Tuyệt vời! Vậy hãy bắt đầu từ **tầng thấp nhất**, nơi tất cả mọi thứ trong ASP.NET Core Web API bắt đầu: **khi trình duyệt gửi một HTTP request, điều gì xảy ra?**

**⚙️ Tầng 0: Hệ điều hành & Network Stack (TCP/IP)**

1. Trình duyệt gửi HTTP request tới một địa chỉ như https://localhost:5001/api/books
2. Hệ điều hành (Windows, Linux) dùng TCP/IP stack nhận gói tin đó và chuyển tới cổng được mở.
3. Cổng đó đang được lắng nghe bởi **Kestrel** hoặc **IIS**.

**🏗️ Tầng 1: Web Server (Kestrel & IIS)**

**✅ Kestrel**

* Là web server **thuần C#**, đa nền tảng, mặc định của ASP.NET Core.
* Kestrel:
  + Mở socket TCP để lắng nghe request từ OS.
  + Đọc byte từ socket → tạo HttpRequest/HttpResponse.
  + Đẩy request vào **ASP.NET Core pipeline**.

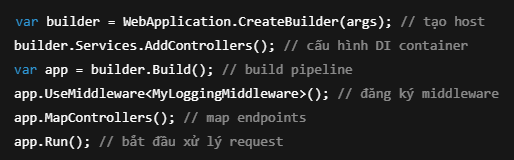
**🔁 Khi dùng IIS (chỉ Windows):**

* IIS **không trực tiếp xử lý request ASP.NET Core**.
* Nó dùng module AspNetCoreModuleV2 để chuyển tiếp request tới **Kestrel**.
* IIS = reverse proxy (chuyển tiếp request nội bộ tới Kestrel).

📌 **Kestrel luôn là engine xử lý chính**.

**🔧 Tầng 2: Host & Startup (Program.cs / Startup.cs)**

**Program.cs là nơi tạo Host Builder, khởi tạo hệ thống.**



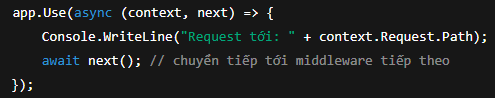
📌 app.Run() là nơi Kestrel bắt đầu lắng nghe request và đẩy request vào pipeline.

**🌀 Tầng 3: Middleware Pipeline (Request → Response)**

**Ý tưởng:**

* Mỗi HTTP request sẽ đi qua chuỗi các **middleware** theo thứ tự.
* Middleware giống như các bộ lọc: logging, auth, routing, v.v.
* Mỗi middleware nhận một HttpContext và RequestDelegate.

Ví dụ:



**🧭 Tầng 4: Endpoint Routing**

* UseRouting(): phân tích URL → xác định controller/action nào phù hợp.
* MapControllers(): cho phép định tuyến tới các controller có [Route].

**📦 Tầng 5: Controller, Model Binding, Action**

* Sau khi route match thành công → ASP.NET Core gọi Controller.Action
* **Model Binding** sẽ bind dữ liệu từ body, query, route → tham số method.



**🔚 Tầng 6: Response → Quay về client**

* Kết quả từ action → ObjectResult, JsonResult, StatusCodeResult...
* ASP.NET dùng **output formatters** (thường là JSON via System.Text.Json) để serialize.
* Sau đó, ghi nội dung vào HttpContext.Response.Body → gửi lại Kestrel → hệ điều hành → client.