**🔹 1. Socket (TCP/UDP)**

* **Tầng hoạt động:** Transport
* **Dùng cho:** Realtime app (game, chat, Gmail noti), low-level network control
* **Loại:** TCP (đảm bảo), UDP (nhanh, không đảm bảo)

**➕ Ưu điểm:**

* Linh hoạt, hiệu năng cao
* Có thể dùng cho các giao thức tự thiết kế riêng

**🔹 2. HTTP/HTTPS**

* **Tầng hoạt động:** Application Layer
* **Dùng cho:** Giao tiếp client-server, REST API, web app
* **Thường chạy trên:** TCP

**➕ Ưu điểm:**

* Đơn giản, phổ biến, dễ debug
* Hỗ trợ bảo mật (HTTPS, TLS)
* Có HTTP/2, HTTP/3 hỗ trợ multiplexing, hiệu suất cao hơn

**🔹 3. WebSocket**

* **Tầng hoạt động:** Application Layer (trên TCP)
* **Dùng cho:** Realtime bidirectional communication (chat, noti, trading)
* **Cách hoạt động:** Khởi tạo từ HTTP rồi nâng cấp thành kết nối 2 chiều

**➕ Ưu điểm:**

* Gửi/nhận dữ liệu song song 2 chiều
* Tốt cho realtime web, nhưng nhẹ hơn socket thuần

**🔹 4. gRPC (Google RPC)**

* **Tầng hoạt động:** Application (trên HTTP/2)
* **Dùng cho:** Microservices, API high performance
* **Định dạng:** Protocol Buffer (binary, nhỏ, nhanh hơn JSON)

**➕ Ưu điểm:**

* Performance cao hơn REST
* Streaming, bidirectional
* Dễ tích hợp đa ngôn ngữ

**🔹 8. SignalR (ASP.NET)**

* **Tầng hoạt động:** Application (trừu tượng WebSocket/HTTP Long Polling)
* **Dùng cho:** Realtime web app .NET (chat, live dashboard)

**➕ Ưu điểm:**

* Dễ dùng, tự động chọn giao thức tốt nhất (WebSocket ưu tiên)
* Có state management

**🔹 9. FTP, SFTP**

* **Tầng hoạt động:** Application Layer
* **Dùng cho:** Truyền file

**➕ Ưu điểm:**

* Được dùng phổ biến trong backend/CI/CD
* SFTP bảo mật qua SSH

Bạn đã có hệ thống **web đọc sách online** dùng ASP.NET Core Web API với giao thức **HTTP + REST API**. Bây giờ bạn muốn **mở rộng hoặc cải tiến** để dùng thêm:

* gRPC
* SignalR
* Socket (TCP/UDP)

**⚙️ 1. Giao thức bạn đang dùng: HTTP + REST API**

* **Cách hoạt động:** Client gửi request → Server trả response qua HTTP.
* **Kiểu truyền:** One request → One response.
* **Dữ liệu:** JSON (text), dễ debug, nhưng nặng hơn binary.
* **Ưu điểm:** Phổ biến, dễ tích hợp mọi frontend, dễ test (Postman).
* **Nhược:** Không phù hợp với **realtime, streaming**, hoặc high-performance internal services.

**2. gRPC**

* **Tầng hoạt động:** Application Layer trên **HTTP/2**
* **Dữ liệu:** Binary (Protocol Buffer – nhỏ gọn hơn JSON)
* **Kiểu gọi:** Remote Procedure Call (gọi hàm từ xa)
* **Tính năng nổi bật:**
  + One-to-one, **bi-directional streaming**
  + Tốc độ nhanh, nhẹ, tốt cho microservices

**✅ Ưu điểm:**

* Nhanh hơn REST (binary, multiplexing)
* Hỗ trợ streaming và realtime nhẹ
* Tốt cho backend-to-backend

**❌ Nhược điểm:**

* Không chạy tốt trên trình duyệt (cần gRPC-Web hoặc proxy)

**Gợi ý dùng gRPC:**

|  |  |
| --- | --- |
| **Microservices nội bộ (Search, Recommender)** | gRPC cho hiệu năng cao |

**3. SignalR**

* **Tầng hoạt động:** Application Layer (WebSocket, fallback HTTP Polling)
* **Dữ liệu:** JSON hoặc MessagePack (binary nhẹ)
* **Kiểu truyền:** **Realtime 2 chiều** giữa client và server

**✅ Ưu điểm:**

* Giao tiếp realtime dễ cài đặt với .NET
* Server gọi trực tiếp tới client (push notification)
* Tự động fallback nếu không có WebSocket

**❌ Nhược điểm:**

* Khó scale hơn REST nếu không dùng Redis hoặc Azure SignalR
* Không phù hợp với batch hoặc RPC

**Gợi ý dùng SignalR:**

| Tình huống | Ứng dụng |
| --- | --- |
| User đang đọc sách → nhận thông báo realtime khi có bình luận mới | Push noti bằng SignalR |
| Admin tạo sự kiện realtime (thông báo hệ thống) | Gửi đến toàn bộ người dùng qua SignalR |
| Realtime live chat giữa user - admin | Dùng SignalR để tạo kênh chat |

**4. Socket (raw TCP/UDP)**

* **Tầng hoạt động:** Transport Layer
* **Dữ liệu:** Tùy định dạng (binary, JSON,...)
* **Kiểu truyền:** Toàn quyền kiểm soát (packet, stream, datagram)

**✅ Ưu điểm:**

* Hiệu năng cao nhất (không có overhead HTTP)
* Linh hoạt tuyệt đối

**❌ Nhược điểm:**

* Cần tự quản lý kết nối, bảo mật, lỗi,...
* Khó tích hợp trình duyệt frontend
* Không hỗ trợ sẵn trong ASP.NET như SignalR

