**So sánh JWT + Refresh Token vs ASP.NET Core Identity**

**1. Kiến trúc và Cơ chế Xác thực**

**JWT + Refresh Token (Tùy biến)**

* **Kiểu xác thực**: Token-based authentication
* **Cách gửi request**: Authorization: Bearer <access\_token>
* **Tính chất**: Stateless - server không cần lưu thông tin session
* **Cấu trúc**: Hai token riêng biệt (Access Token + Refresh Token)

**ASP.NET Core Identity (Mặc định)**

* **Kiểu xác thực**: Cookie-based authentication
* **Cách gửi request**: Cookie được gửi tự động bởi trình duyệt
* **Tính chất**: Stateful - server lưu session và thông tin xác thực
* **Cấu trúc**: Sử dụng cookie session và các bảng Identity framework

**2. Cơ chế Lưu trữ và Quản lý Phiên**

**JWT + Refresh Token**

**Access Token (JWT)**

* Chứa thông tin người dùng (claims) được mã hóa
* Thời hạn ngắn (thường 15-30 phút)
* Tự chứa thông tin, không cần query database để validate
* Không thể thu hồi trước khi hết hạn (trừ khi có blacklist)

**Refresh Token**

* Lưu trữ trong database
* Thời hạn dài (vài ngày đến vài tháng)
* Dùng để cấp lại Access Token mới
* Có thể thu hồi bất cứ lúc nào

**ASP.NET Core Identity**

**Cookie Session**

* Chứa session ID, không chứa thông tin nhạy cảm
* Server lưu thông tin session tương ứng
* Có thể thu hồi ngay lập tức

**Database Storage**

* Sử dụng các bảng có sẵn: AspNetUsers, AspNetUserTokens, AspNetUserLogins
* Framework tự động quản lý session, tokens, và user data
* Tích hợp sẵn với Entity Framework

**3. Ưu nhược điểm**

**JWT + Refresh Token**

**Ưu điểm:**

* **Scalability tốt**: Stateless, dễ scale horizontal
* **Cross-domain**: Dễ sử dụng với SPA, mobile apps
* **Microservices friendly**: Token có thể validate độc lập
* **Offline validation**: Không cần query DB để validate Access Token
* **Flexibility**: Tùy chỉnh claims theo nhu cầu

**Nhược điểm:**

* **Phức tạp hơn**: Phải tự implement logic refresh token
* **Security risks**: JWT không thể thu hồi trước khi hết hạn
* **Token size**: JWT có thể lớn nếu chứa nhiều claims
* **Storage management**: Phải tự quản lý Refresh Token trong DB

**ASP.NET Core Identity**

**Ưu điểm:**

* **Đơn giản**: Framework xử lý hầu hết logic xác thực
* **Security**: Tích hợp sẵn nhiều tính năng bảo mật
* **Immediate revocation**: Có thể thu hồi session ngay lập tức
* **Rich features**: 2FA, password reset, email confirmation, etc.
* **Well-tested**: Framework mature, đã được test kỹ

**Nhược điểm:**

* **Stateful**: Khó scale với distributed systems
* **Browser-centric**: Chủ yếu cho web apps, khó dùng với mobile/API
* **Less flexible**: Khó tùy chỉnh deep-level behavior
* **Server dependency**: Mỗi request cần access session store

**4. Khi nào nên sử dụng?**

**Chọn JWT + Refresh Token khi:**

* Xây dựng API cho mobile apps hoặc SPA
* Cần architecture microservices
* Muốn stateless authentication
* Cần cross-domain authentication
* Có requirements đặc biệt về claims/permissions

**Chọn ASP.NET Core Identity khi:**

* Xây dựng traditional web application
* Cần rapid development với full authentication features
* Team chưa có kinh nghiệm implement custom authentication
* Cần tính năng như 2FA, external login providers
* Security requirements cao và muốn dùng solution đã được kiểm chứng