1. **Hub**

Trong ASP.NET MVC, "hub" thường được sử dụng để thực hiện các tính năng thời gian thực (real-time) thông qua SignalR. Hub là một phần của SignalR, là một thư viện cho phép truyền tải dữ liệu theo thời gian thực (real-time) giữa máy chủ và trình duyệt của người dùng.

Hub cung cấp một API để giao tiếp hai chiều giữa máy chủ và trình duyệt của người dùng. Nó sử dụng các kết nối WebSockets, Server-Sent Events hoặc các kỹ thuật fallback khác để giao tiếp giữa hai bên. Khi một tín hiệu (signal) được gửi từ máy chủ đến trình duyệt của người dùng, SignalR sử dụng JavaScript để cập nhật nội dung trên trang web của người dùng mà không cần phải tải lại trang web.

Trong ASP.NET MVC, bạn có thể sử dụng Hub để xây dựng các tính năng thời gian thực như chat, thông báo, cập nhật dữ liệu trực tiếp, vv.

1. **Docker**

Docker là một nền tảng ảo hóa dùng để đóng gói, phân phối và chạy ứng dụng trong một môi trường độc lập với hệ thống máy chủ. Với Docker, bạn có thể đóng gói ứng dụng và các phụ thuộc của nó trong một container, và sau đó phân phối và chạy container đó trên bất kỳ hệ thống nào có hỗ trợ Docker.

Trong ASP.NET MVC, Docker cung cấp một cách tiếp cận mới cho việc phát triển và triển khai ứng dụng. Bằng cách sử dụng Docker, bạn có thể đóng gói ứng dụng của mình và các phụ thuộc của nó trong một container, và sau đó phân phối và triển khai container đó trên bất kỳ máy chủ nào có hỗ trợ Docker.

Việc sử dụng Docker trong ASP.NET MVC có thể giúp bạn dễ dàng quản lý các phụ thuộc của ứng dụng và đảm bảo rằng ứng dụng của bạn sẽ chạy đúng trên bất kỳ hệ thống nào có hỗ trợ Docker. Nó cũng giúp cho việc triển khai ứng dụng trở nên đơn giản hơn và nhanh chóng hơn, bằng cách đóng gói ứng dụng trong một container duy nhất và sau đó triển khai container đó trên bất kỳ máy chủ nào có hỗ trợ Docker.

1. **Log4net**

Log4net là một thư viện ghi log được sử dụng trong các ứng dụng .NET để ghi lại các thông tin hệ thống và sự kiện trong quá trình thực thi của ứng dụng.

1. **Owin**

OWIN (Open Web Interface for .NET) là một giao diện lập trình ứng dụng (API) độc lập với nền tảng cho phép các framework ứng dụng web được đóng gói thành các thành phần middleware (phần mềm trung gian) có thể tái sử dụng được.

OWIN được thiết kế để giảm thiểu sự phụ thuộc vào các framework và máy chủ web cụ thể, giúp cho các ứng dụng web có thể độc lập hơn và dễ dàng di chuyển giữa các nền tảng khác nhau. Thay vì phải sử dụng các đối tượng và phương thức của máy chủ web cụ thể, các ứng dụng web có thể sử dụng OWIN để kết nối với các thành phần middleware có thể sử dụng chung.

1. **Class Startup**

Class Startup trong ASP.NET là một class quan trọng trong quá trình khởi tạo ứng dụng web. Nó được sử dụng để cấu hình các dịch vụ (services) và middleware (phần mềm trung gian) trong ứng dụng.

Trong ASP.NET Core, Class Startup được sử dụng để định cấu hình ứng dụng bằng cách sử dụng Dependency Injection (DI) để đăng ký các dịch vụ và cấu hình các middleware. Class Startup được gọi khi ứng dụng khởi động để cấu hình các dịch vụ và middleware trước khi xử lý các yêu cầu HTTP.

Class Startup chứa hai phương thức chính là ConfigureServices và Configure. Phương thức ConfigureServices được sử dụng để đăng ký các dịch vụ của ứng dụng trong container DI. Phương thức Configure được sử dụng để cấu hình các middleware và xác định cách xử lý các yêu cầu HTTP.

Ví dụ, trong phương thức ConfigureServices, bạn có thể đăng ký các dịch vụ như Entity Framework, Identity, Swagger, Redis, RabbitMQ, ...

Trong phương thức Configure, bạn có thể thêm các middleware để xử lý các yêu cầu HTTP như xác thực, bảo mật, định tuyến, đăng ký dịch vụ, ...