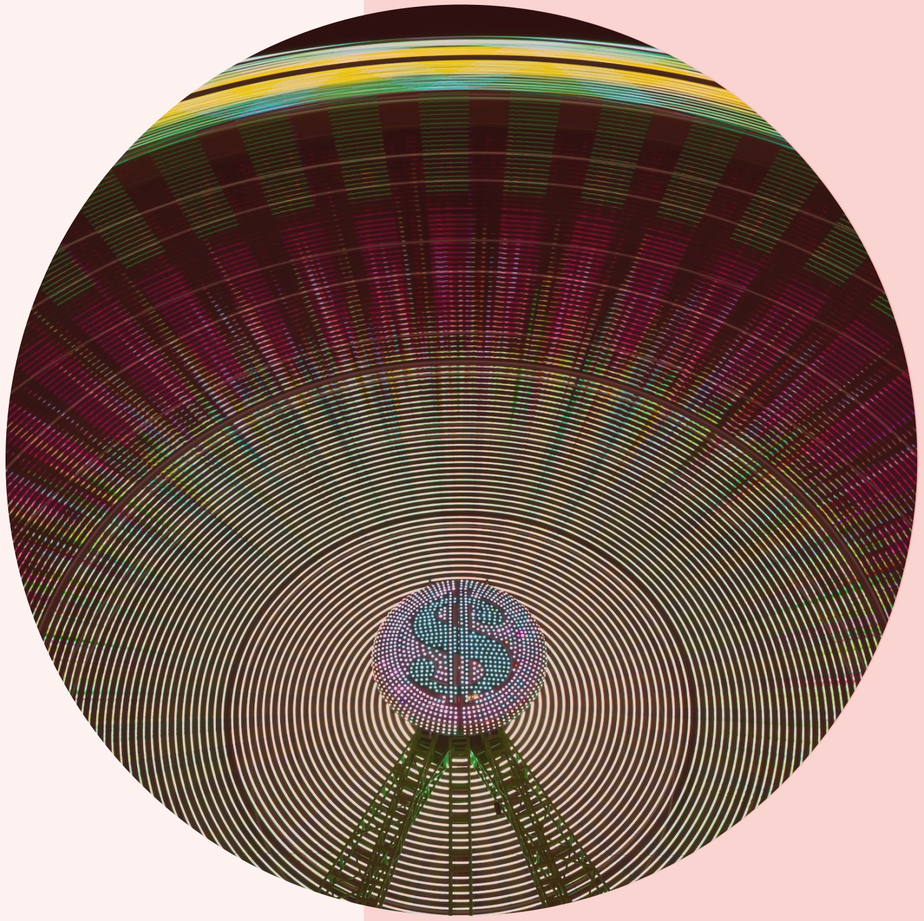
Node.js 웹소켓을 활용한 실시간 통신 시스템 구축



소개

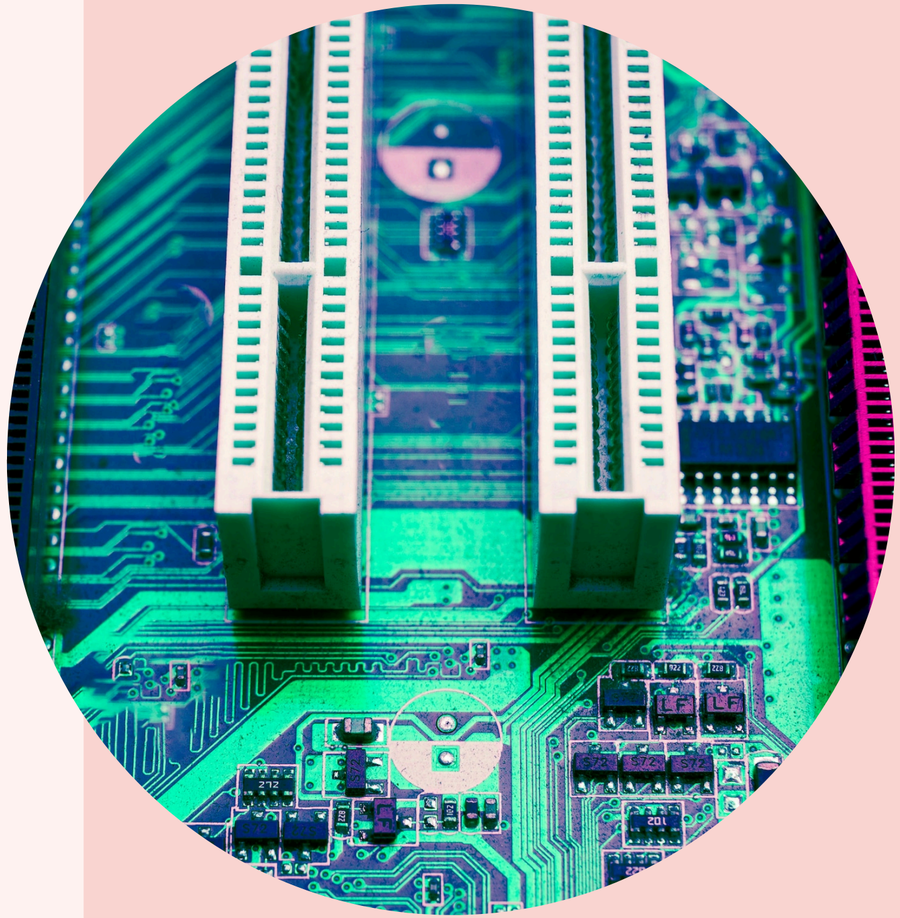
Node.js 웹소켓을 활용한 **실시간 통신 시스템** 구축에 대해 알아보겠습니다. 이 발표에서는 웹소켓의 기본 개념, Node.js와의 통합, 그리고 실시간 애플리케이션을 개발하는 방법을 다룰 것입니다.

웹소켓의 이해

웹소켓은 클라이언트와 서버 간의 **양방향 통신**을 가능하게 하는 프로토콜입니다. HTTP 요청과 달리, 웹소켓은 지속적인 연결을 유지하여 실시간 데이터 전송을 지원합니다.

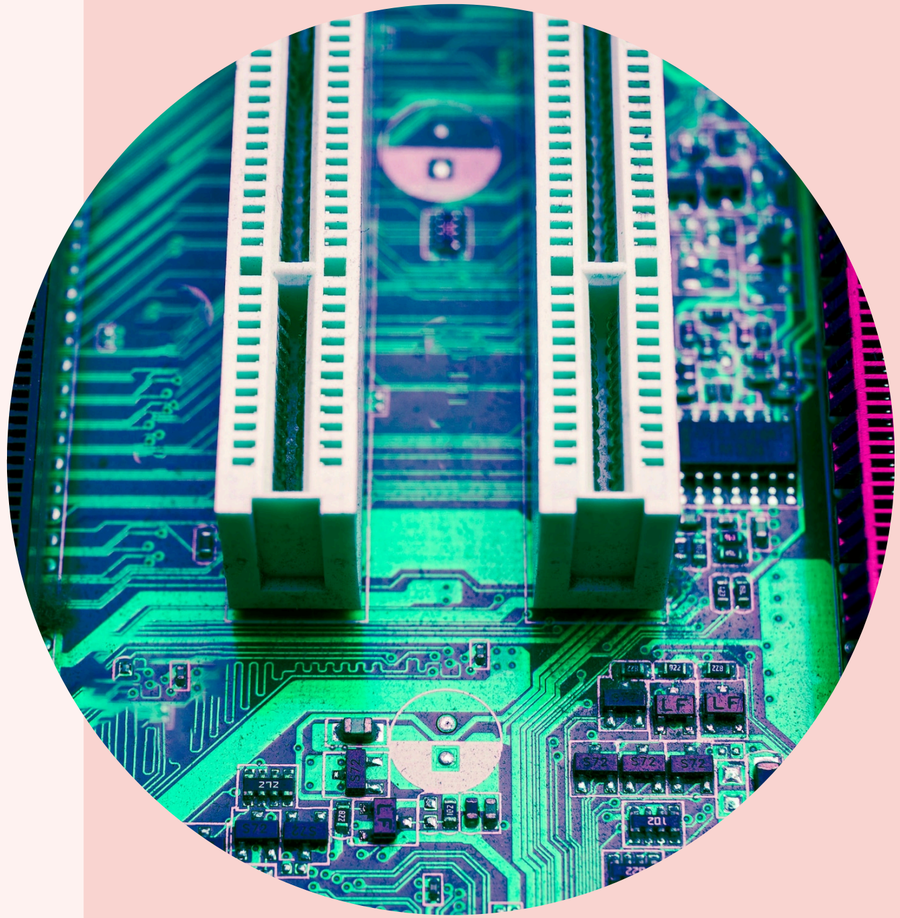
Node.js의 장점

Node.js는 비동기 I/O 모델을 기반으로 하여 높은 **성능**과 **확장성**을 제공합니다. 이러한 특성 덕분에 웹소켓을 활용한 실시간 애플리케이션 개발에 적합합니다.



설치 및 설정

Node.js와 **웹소켓 라이브러리**(예: Socket.IO)를 설치하는 방법을 설명합니다. 이를 통해 실시간 통신을 위한 기본 환경을 구축할 수 있습니다.



서버 구현

Node.js를 사용하여 **웹소켓 서버**를 구현하는 방법을 살펴보겠습니다. 클라이언트의 연결 요청을 처리하고, 메시지를 송수신하는 기본 코드를 작성합니다.

클라이언트 구현

웹소켓 클라이언트를 구현하여 서버와 연결하고, **메시지 전송** 및 수신을 처리하는 방법을 설명합니다. 이를 통해 실시간 통신의 흐름을 이해할 수 있습니다.

실시간 애플리케이션 사례

실시간 채팅, 게임, 협업 도구 등 다양한 **실시간 애플리케이션**의 사례를 소개합니다. 이러한 애플리케이션에서 웹소켓의 활용도를 이해할 수 있습니다.

결론

Node.js와 웹소켓을 활용하여 **효율적인 실시간 통신 시스템**을 구축할 수 있습니다. 이 기술을 통해 다양한 애플리케이션을 개발하고, 사용자 경험을 향상시킬 수 있습니다.

Thanks!