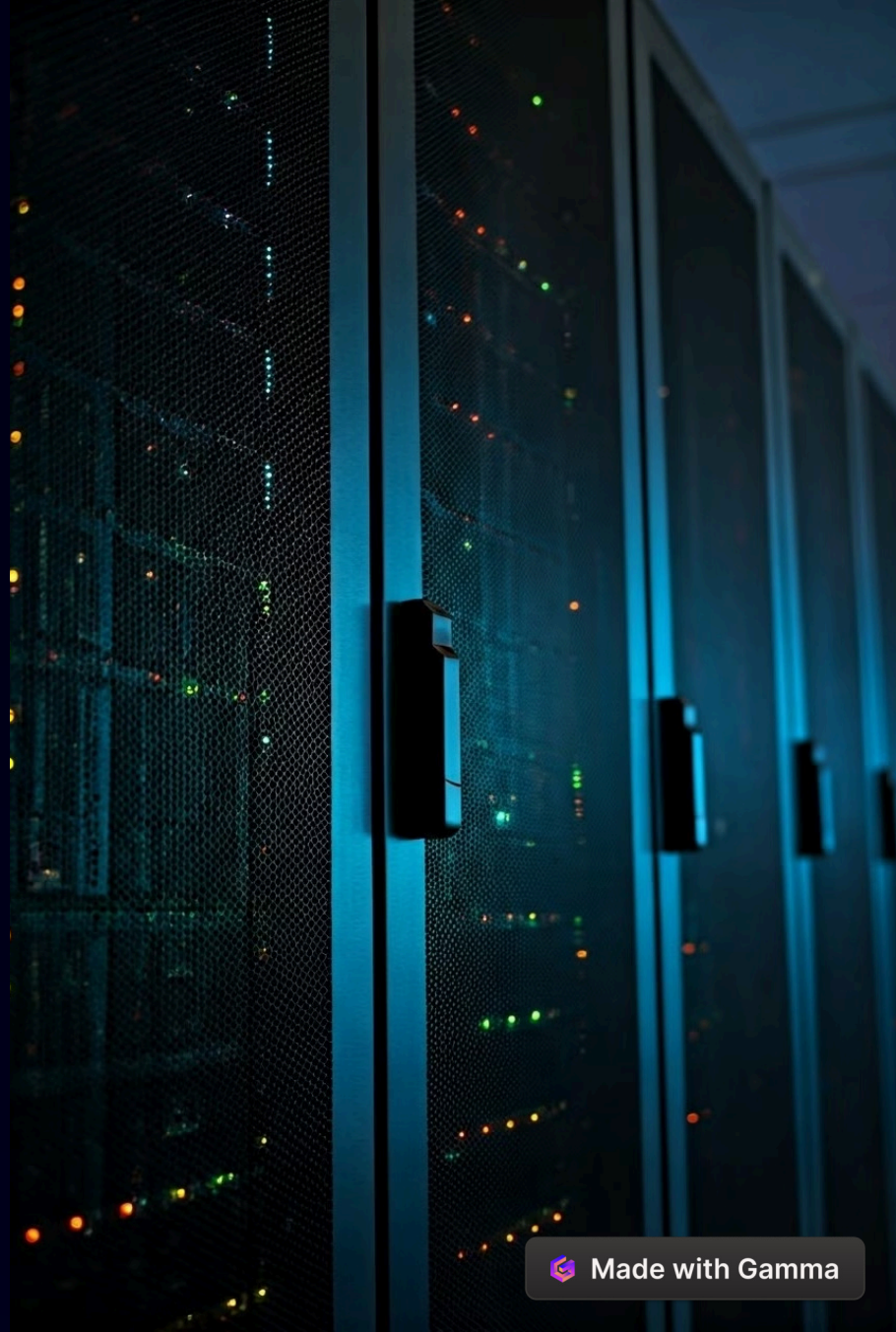
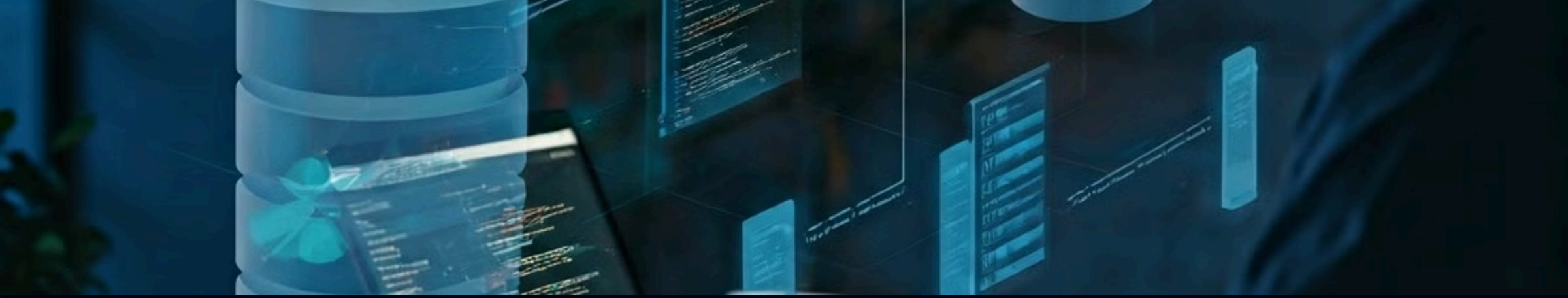
Node.js와 MongoDB 통합: 스무스한 데이터 관리

Node.js와 MongoDB를 통합하면 강력한 웹 애플리케이션을 만들 수 있습니다. 이 프레젠테이션에서는 Mongoose를 활용한 데이터베이스 연결 설정부터 CRUD 작업 구현, 데이터베이스와 서버 간 데이터 흐름 이해까지 전 과정을 다룹니다.

작성자: 현욱





Mongoose를 활용한 데이터베이스 연결 설정

1 신속한 연결 설정

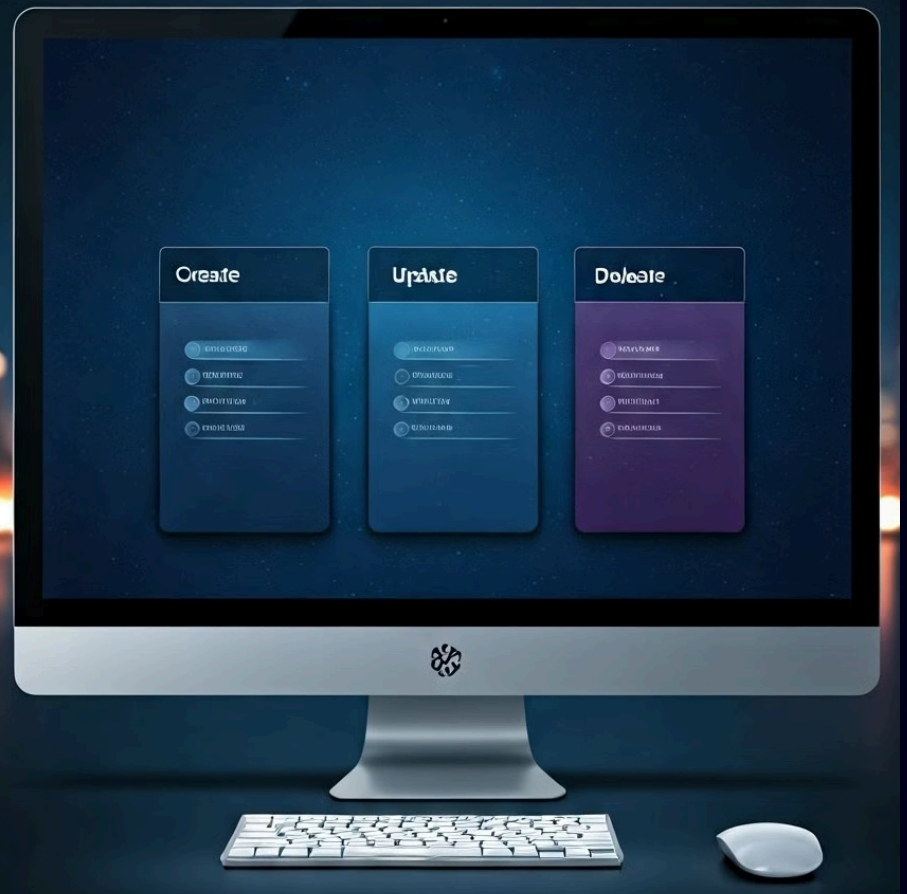
Mongoose를 통해 MongoDB 데이터베이스와 빠르게 연결할 수 있습니다.

2 모델 정의

Mongoose 스키마를 사용하여 데이터베이스 테이블의 구조를 정의합니다.

3 스키마 유효성 검사

Mongoose는 데이터 입력 시 스키마를 기반으로 유효성을 검사합니다.



CRUD 작업 구현하기

1. **Create**
   새로운 데이터를 데이터베이스에 저장합니다.
2. **Read**
   데이터베이스에서 데이터를 조회합니다.
3. **Update**
   기존 데이터를 수정합니다.

쿼리 작성과 데이터 핸들링

고급 쿼리 작성

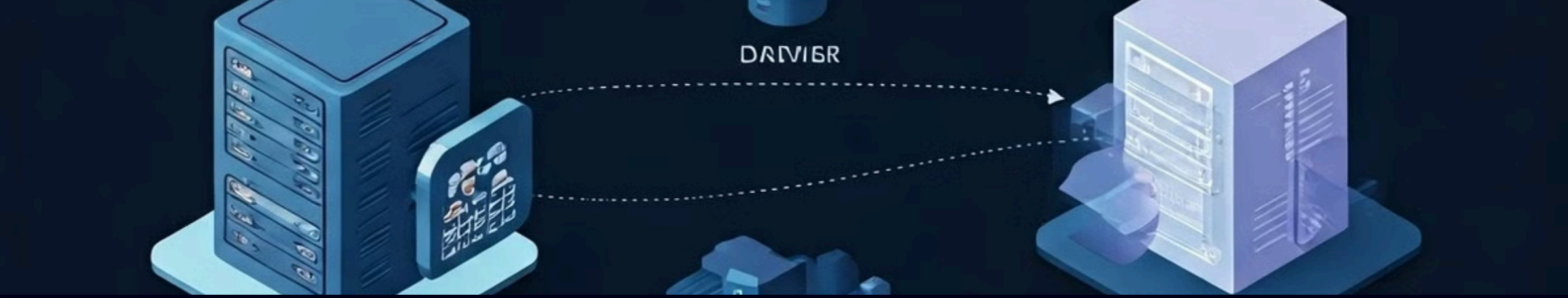
Mongoose는 다양한 쿼리 메서드를 제공하여 복잡한 데이터 조회가 가능합니다.

데이터 필터링

조건, 정렬, 페이지네이션 등을 통해 데이터를 효율적으로 필터링할 수 있습니다.

관계형 데이터 처리

Mongoose의 참조 기능을 사용하면 관계형 데이터를 쉽게 처리할 수 있습니다.



데이터베이스와 서버 간 데이터 흐름 이해하기

1

요청 전송

클라이언트 요청이 Node.js 서버로 전달됩니다.

2

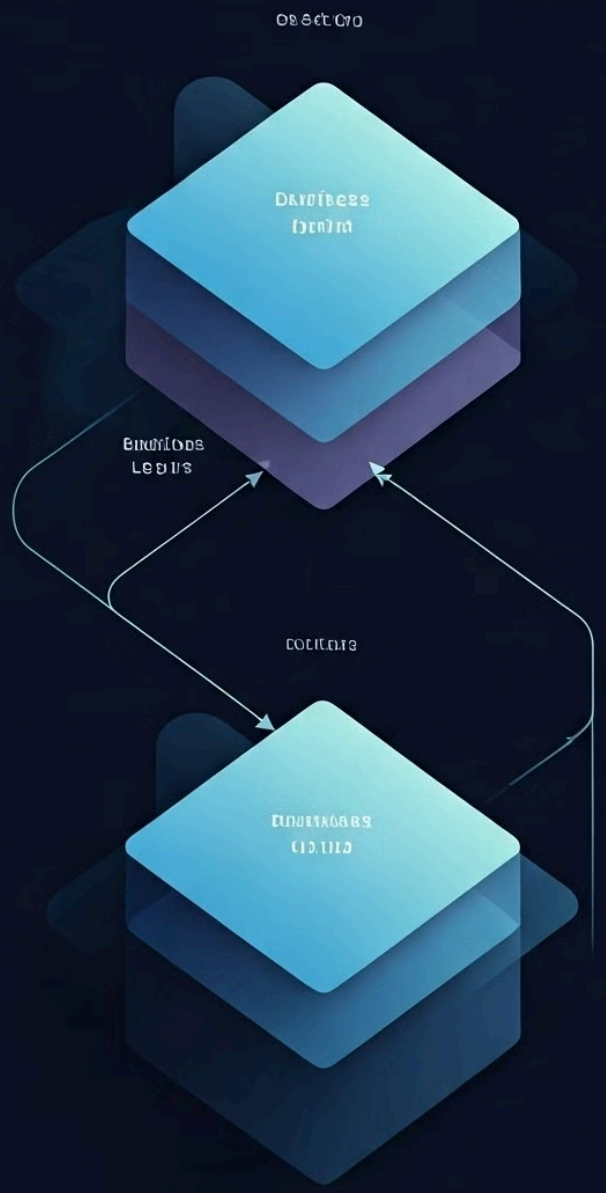
데이터 조회/수정

서버에서 MongoDB 데이터베이스와 상호작용합니다.

3

응답 반환

데이터베이스에서 조회된 데이터가 클라이언트에 전송됩니다.



비즈니스 로직과 데이터 처리 분리

비즈니스 로직

애플리케이션의 핵심 기능을 구현합니다.

데이터 처리

데이터베이스와의 상호작용을 처리합니다.

API 서버

비즈니스 로직과 데이터 처리를 통합합니다.

에러 처리와 예외 핸들링

예외 감지

데이터베이스 작업 중 발생할 수 있는 예외를 감지합니다.

사용자 친화적 에러 메시지

적절한 에러 메시지를 통해 사용자에게 정확한 정보를 제공합니다.

안정적인 프로세스

예외 처리를 통해 애플리케이션의 안정성과 신뢰성을 높입니다.

마무리: 통합 프로세스 요약 및 실습 방향

이번 프레젠테이션에서는 Node.js와 MongoDB의 통합 과정을 살펴보았습니다. Mongoose를 활용한 데이터베이스 연결 설정, CRUD 작업 구현, 데이터베이스와 서버 간 데이터 흐름 등을 다루었습니다. 다음으로는 직접 실습하며 이 내용을 익히시기 바랍니다.