

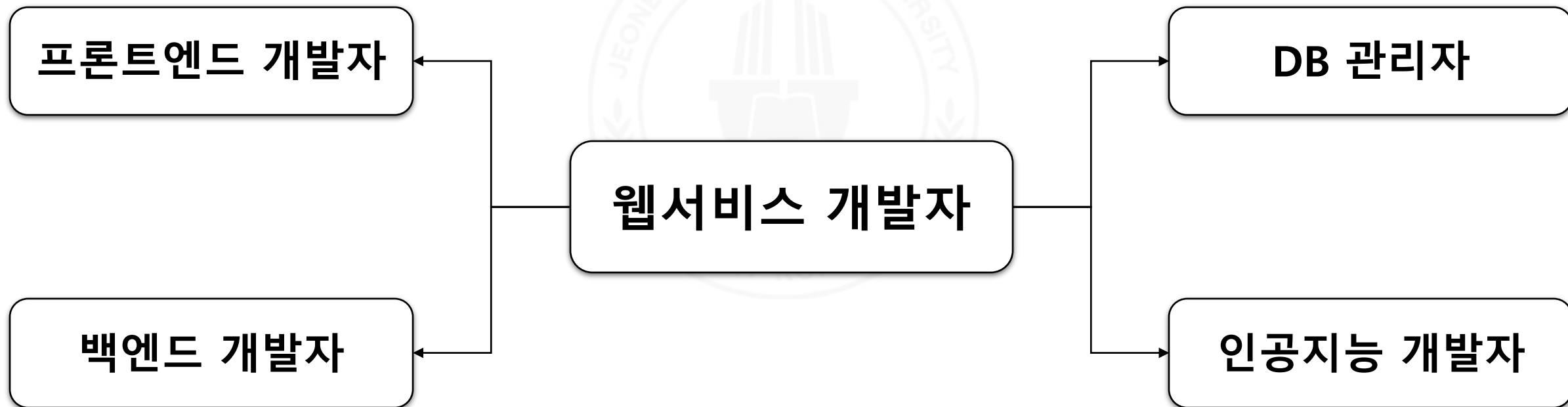
01 CHAPTER

개발 로드맵

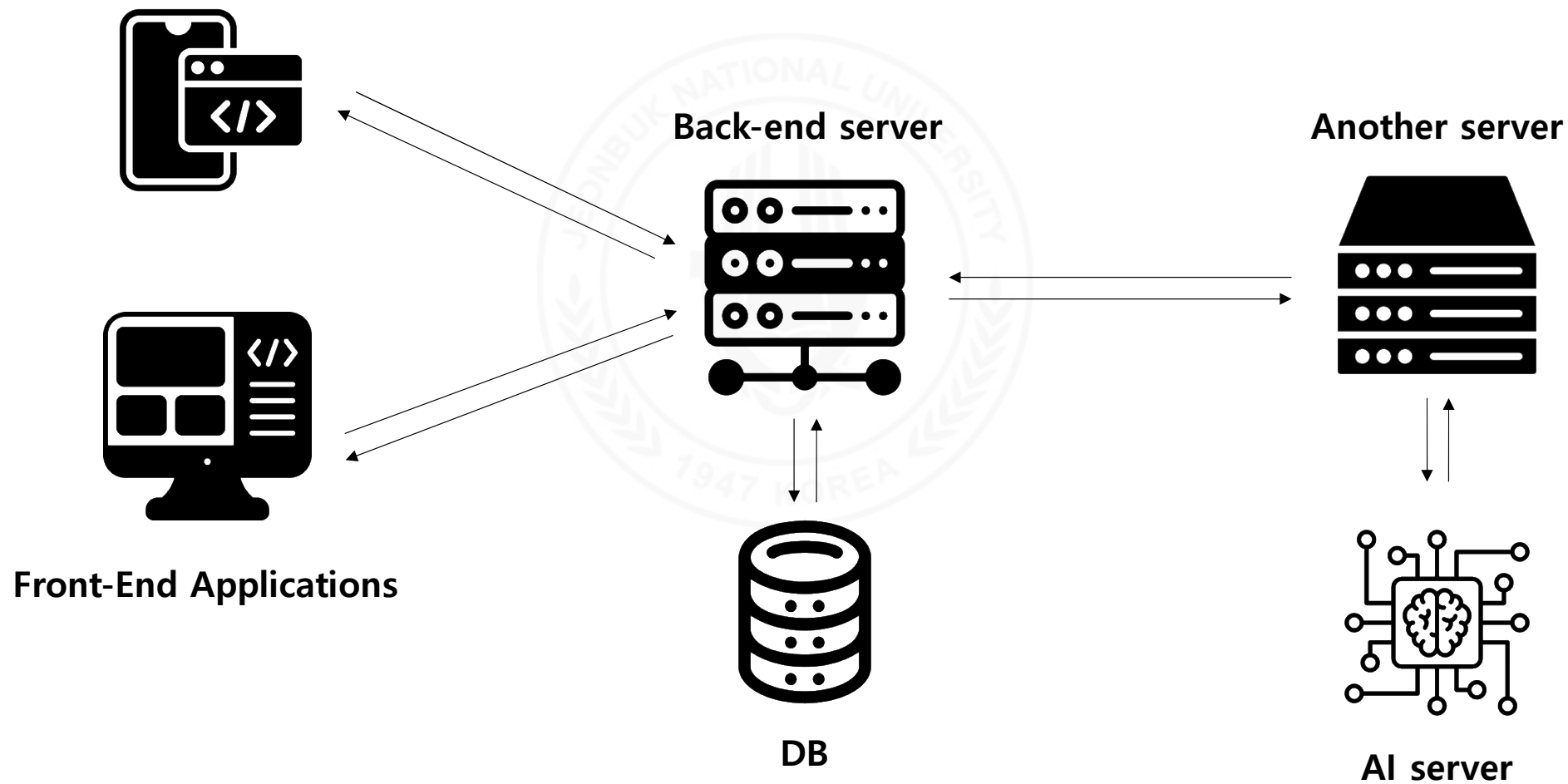
- 웹서비스 개발자의 역할
- 웹서비스의 구성
- 웹서비스와 백엔드

웹서비스 개발자의 역할

웹서비스 개발자?



웹서비스 개발자?



웹서비스 개발자의 역할

웹서비스 개발자?



| 웹서비스 개발자?

Web Programming 기초

HTML, CSS, JavaScript

웹 프로그래밍의 기초를 이해하려면 **HTML(구조)**, **CSS(스타일)**, **Javascript(인터랙티브 기능)**와 같은 필수 언어를 배우는 것에서 시작해야 합니다.

Backend Technologies

백엔드 개발을 위해서는 Python, Java, Node.js와 같은 **서버 측 프로그래밍 언어**를 배우는 것이 중요합니다. MySQL이나 MongoDB와 같은 **데이터베이스**를 이해하는 것도 필수적입니다.

Version Control Systems

Git과 같은 버전 관리 시스템을 사용하여 **협업 및 코드 관리**를 할 수 있는 능력은 백엔드 개발 프로젝트에서 필수적입니다.

웹서비스 개발자?

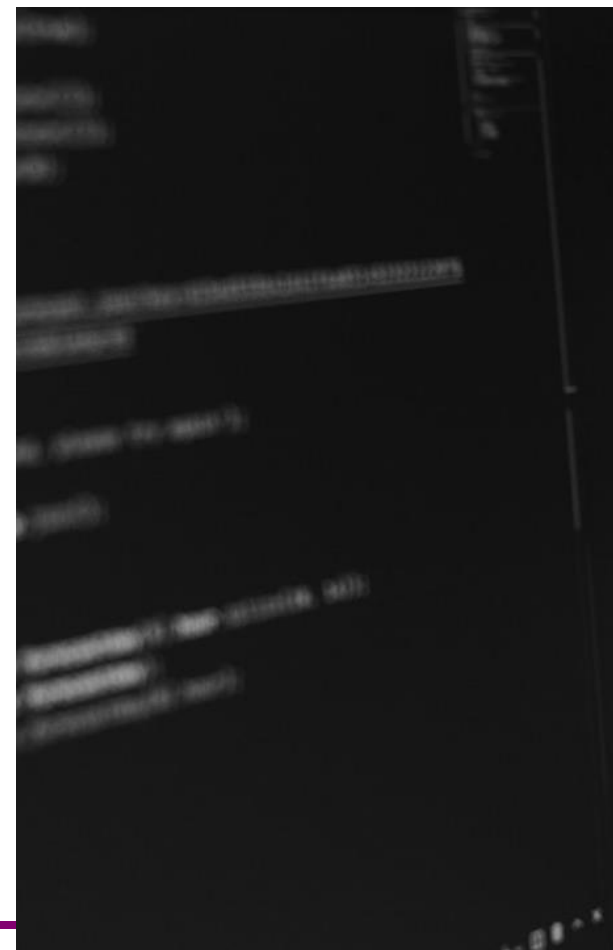
Backend Frameworks and Libraries

개발 관련 라이브러리

인기 있는 백엔드 프레임워크인 Django, Flask, Express.js를 공부하면 개발 프로세스를 간소화하고 **백엔드 구성 요소에 대한 즉시 사용 가능한 솔루션**을 개발할 수 있습니다.

배포 관련 라이브러리

Axios, Lodash, requests와 같은 라이브러리를 백엔드 개발 프로젝트에서 활용하면 **기능성을 향상**시키고 일반적인 작업을 단순화할 수 있습니다.



| 웹서비스 개발자?

APIs and Microservices

API Development

API를 이해하고 RESTful 서비스를 설계하면 백엔드 시스템의 다양한 구성 요소 간의 **원활한 통신**을 가능하게 합니다.

Microservices Architecture

마이크로서비스 아키텍처를 구현하면 복잡한 시스템을 더 작고 독립적인 서비스로 분해하여 **백엔드 애플리케이션의 확장성과 유연성을** 촉진합니다.

Security Considerations

API 설계 및 마이크로서비스 아키텍처에서 보안 문제를 해결하는 것은 **데이터 보호와 백엔드 시스템의 무단 접근 방지**에 매우 중요합니다.

웹서비스 개발자?

Deployment and Monitoring

Deployment Best Practices

Docker를 사용한 컨테이너화 및 Kubernetes를 통한 오케스트레이션과 같은 배포 모범 사례를 따르면 백엔드 애플리케이션의 **효율적인 배포 및 확장이 보장**됩니다.

Performance Monitoring

Prometheus, Grafana, New Relic과 같은 모니터링 도구를 구현하면 백엔드 애플리케이션의 **성능 지표를 추적**하고 문제를 **해결**하는 데 도움이 됩니다.

웹서비스 개발자?

Optimization and Scalability

Caching Strategies

Redis 또는 Memcached와 같은 캐싱 메커니즘을 구현하면 자주 접근하는 데이터를 저장하고 **데이터베이스 부하를 줄여 성능을 향상**시킬 수 있습니다.

Load Balancing

Nginx 또는 HAProxy와 같은 로드 밸런서를 사용하면 여러 서버에 들어오는 **트래픽을 분산**시켜 최적의 자원 활용 및 확장성을 보장합니다.

Horizontal Scaling

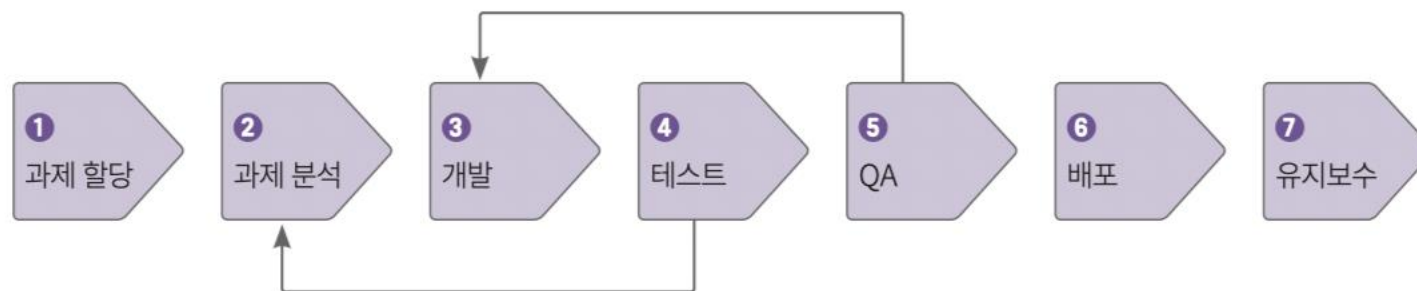
더 많은 서버 또는 컨테이너를 추가하여 백엔드 시스템을 수평으로 확장하면 증가하는 **사용자 부하를 처리**하고 일관된 성능을 보장합니다.

웹서비스 개발자의 역할

웹서비스 개발자?

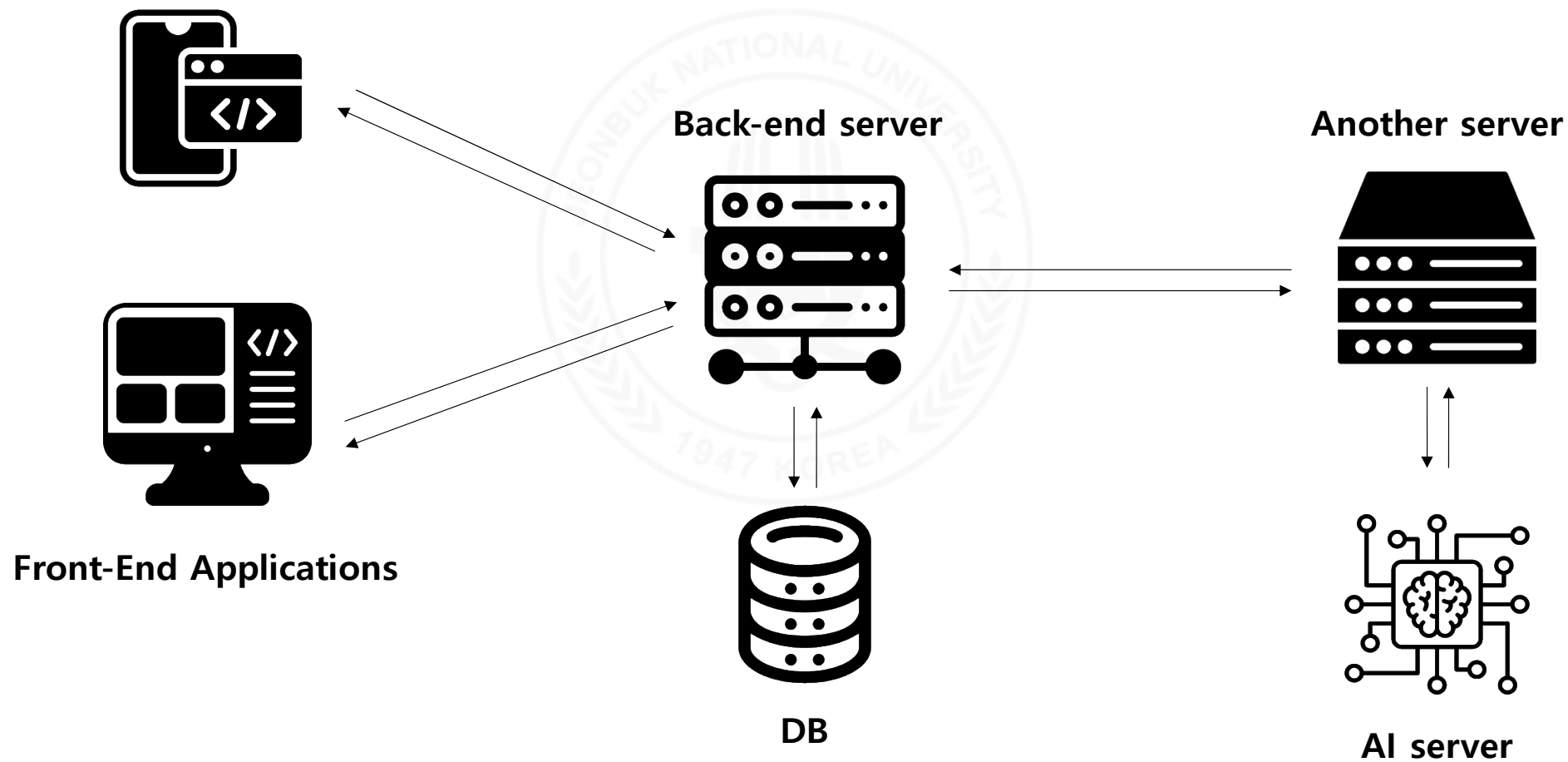
과제 할당 → 과제 분석 → 개발 → 테스트(리뷰) →
QA 및 버그 수정 → 배포 → 유지보수 순서로 진행

▼백엔드 개발자가 수행하는 업무 순서



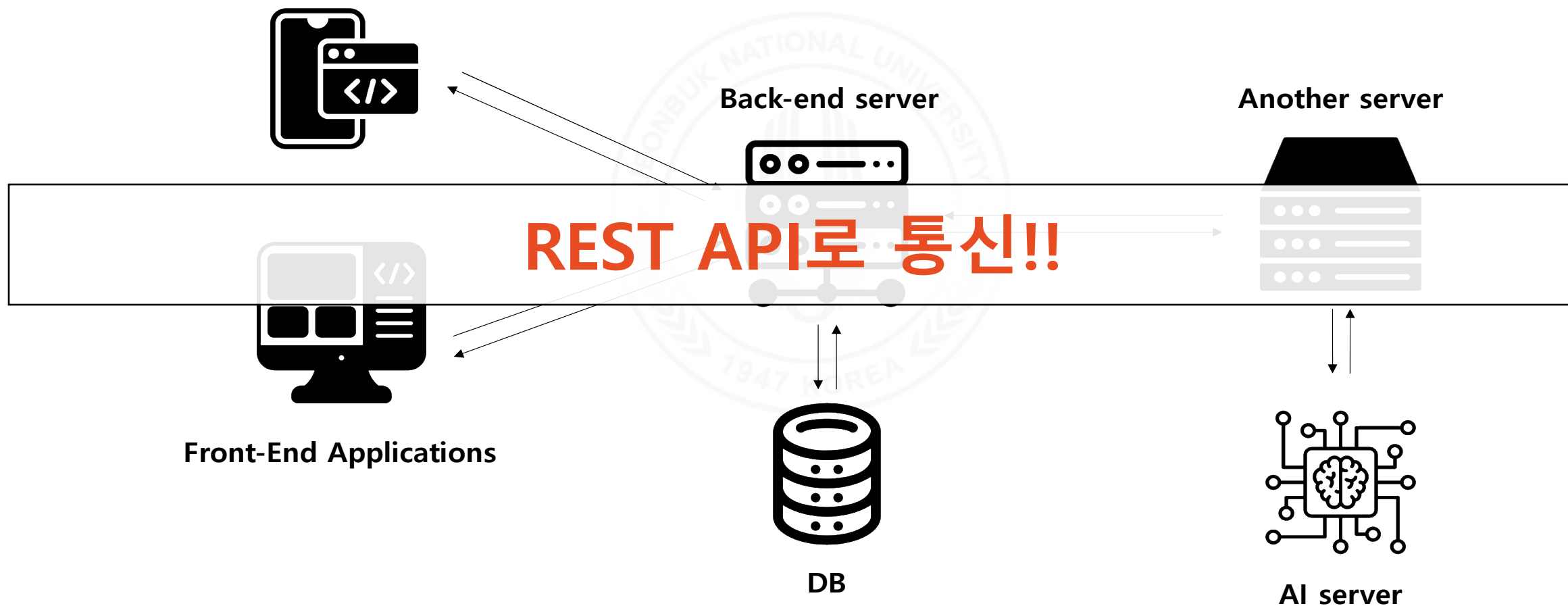
웹서비스의 구성

웹서비스 구성



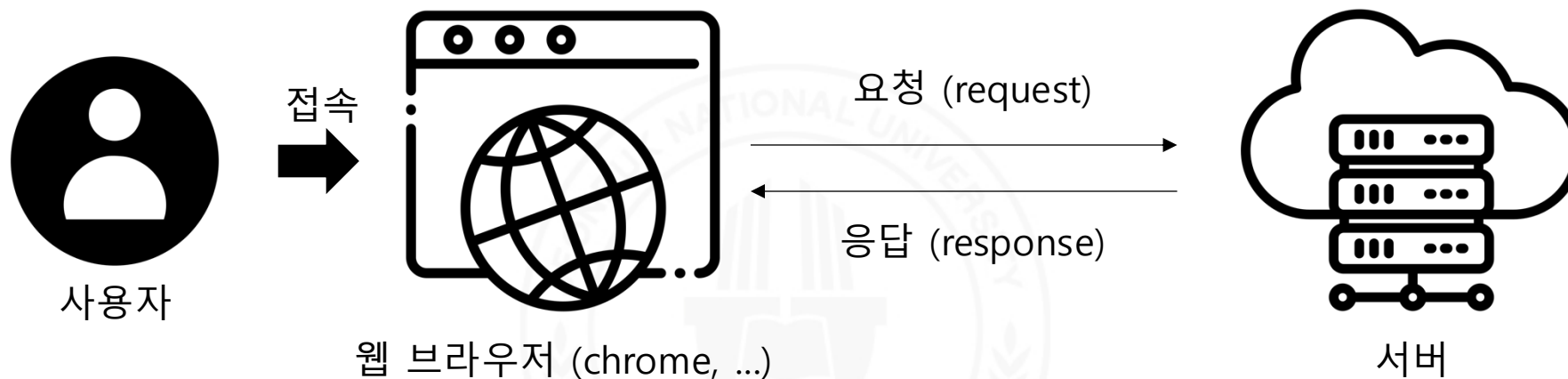
웹서비스 개발자의 역할

웹서비스 구성



웹서비스 개발자의 역할

웹서비스 구성

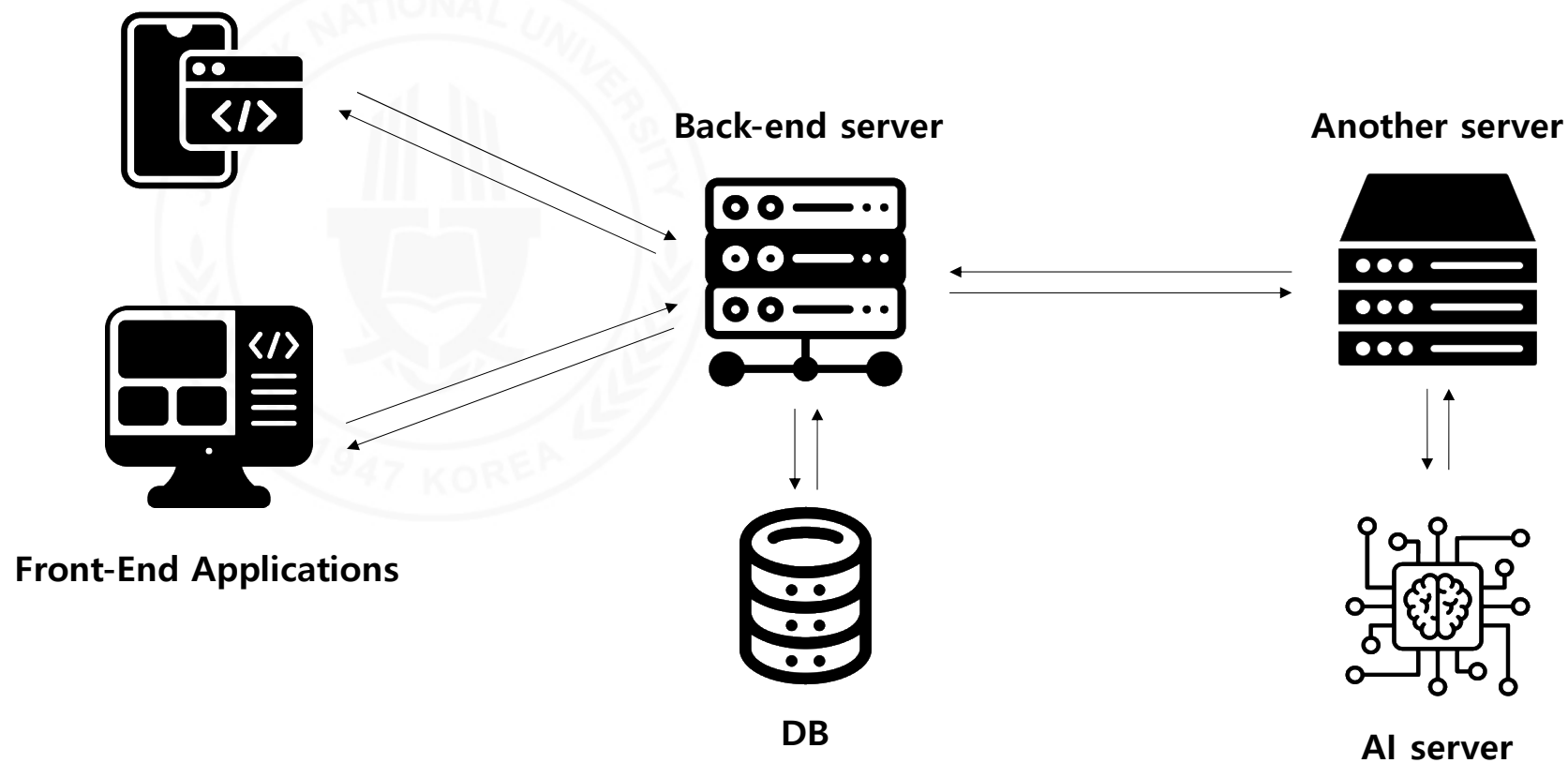


웹 API의 역할

- 서버와 데이터베이스안의 리소스에 접근
 - 데이터베이스의 정보를 누구나 열람하면 곤란하기에, 필요에 의해서만 열람 가능.
 - API는 접근 권한이 인가된 사람에게만 서버와 데이터베이스에 접근하도록 함.
- 모든 요청과 응답을 표준화
 - 아이폰을 쓰던 갤럭시 폰을 사용하든 상관없이 동일한 API를 사용하기 때문에 클라이언트의 요청과 서버의 응답을 하나의 API로 표준화 할 수 있음.

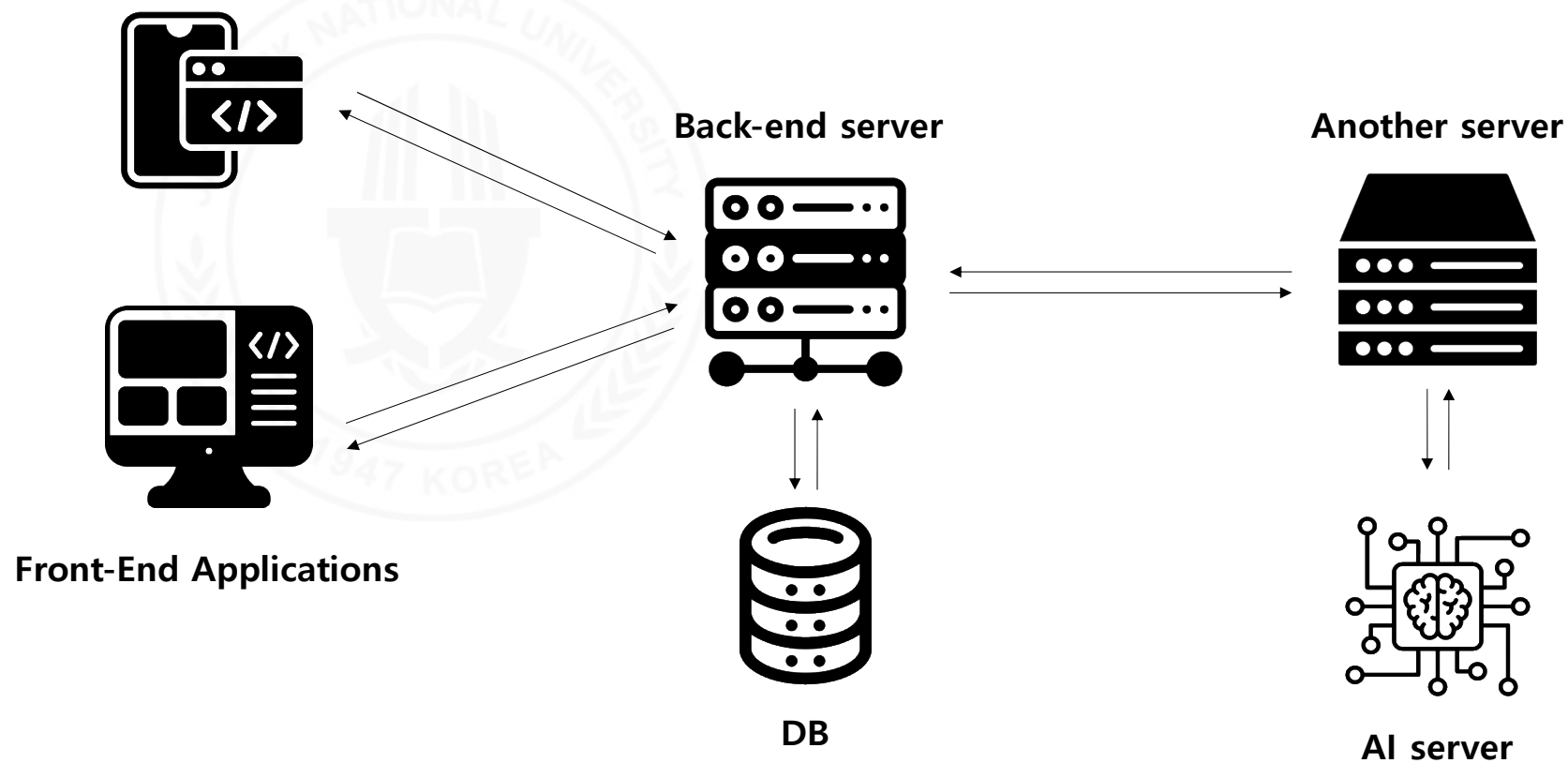
웹서비스 구성

Front-End



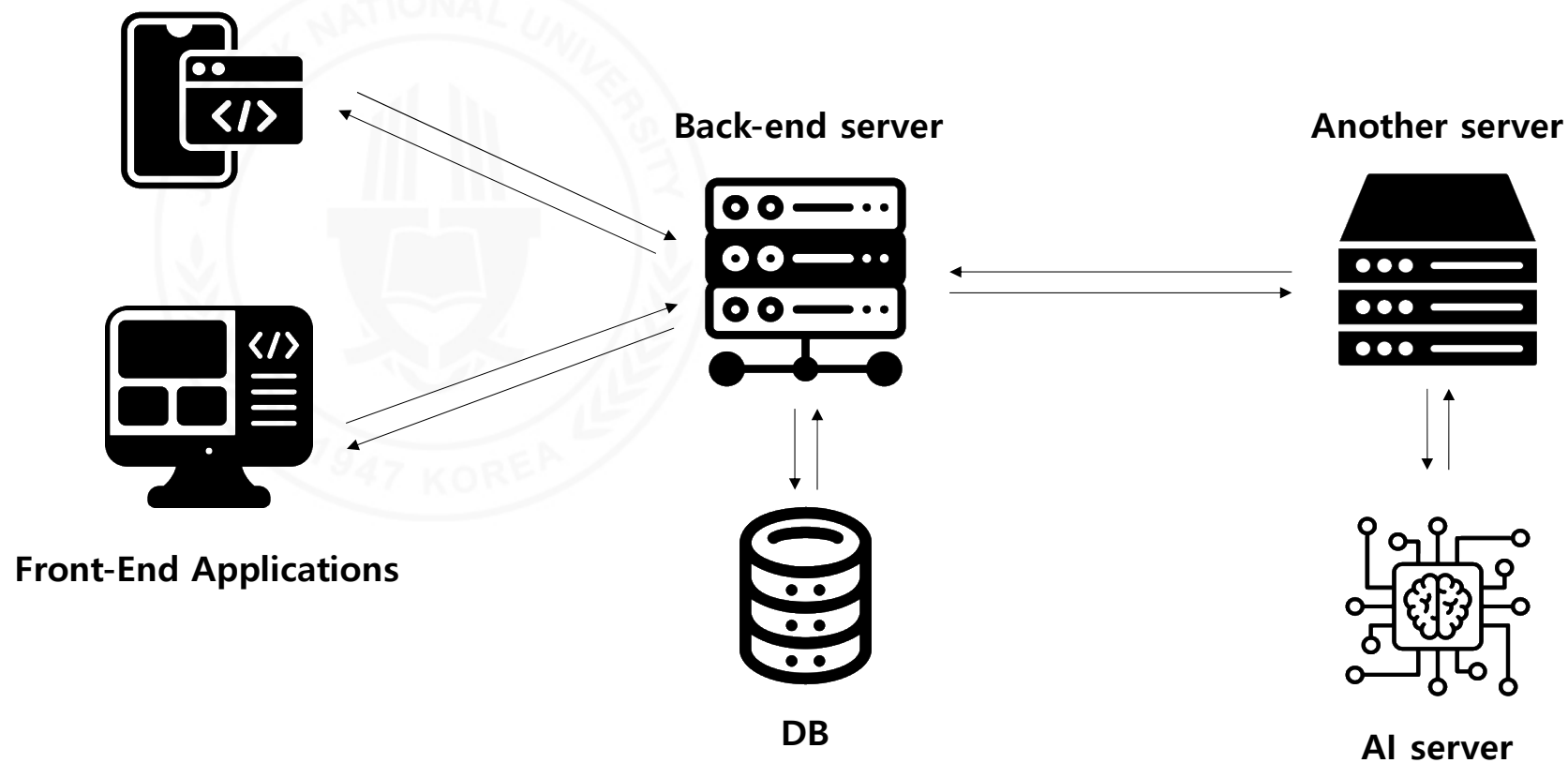
웹서비스 구성

Back-End



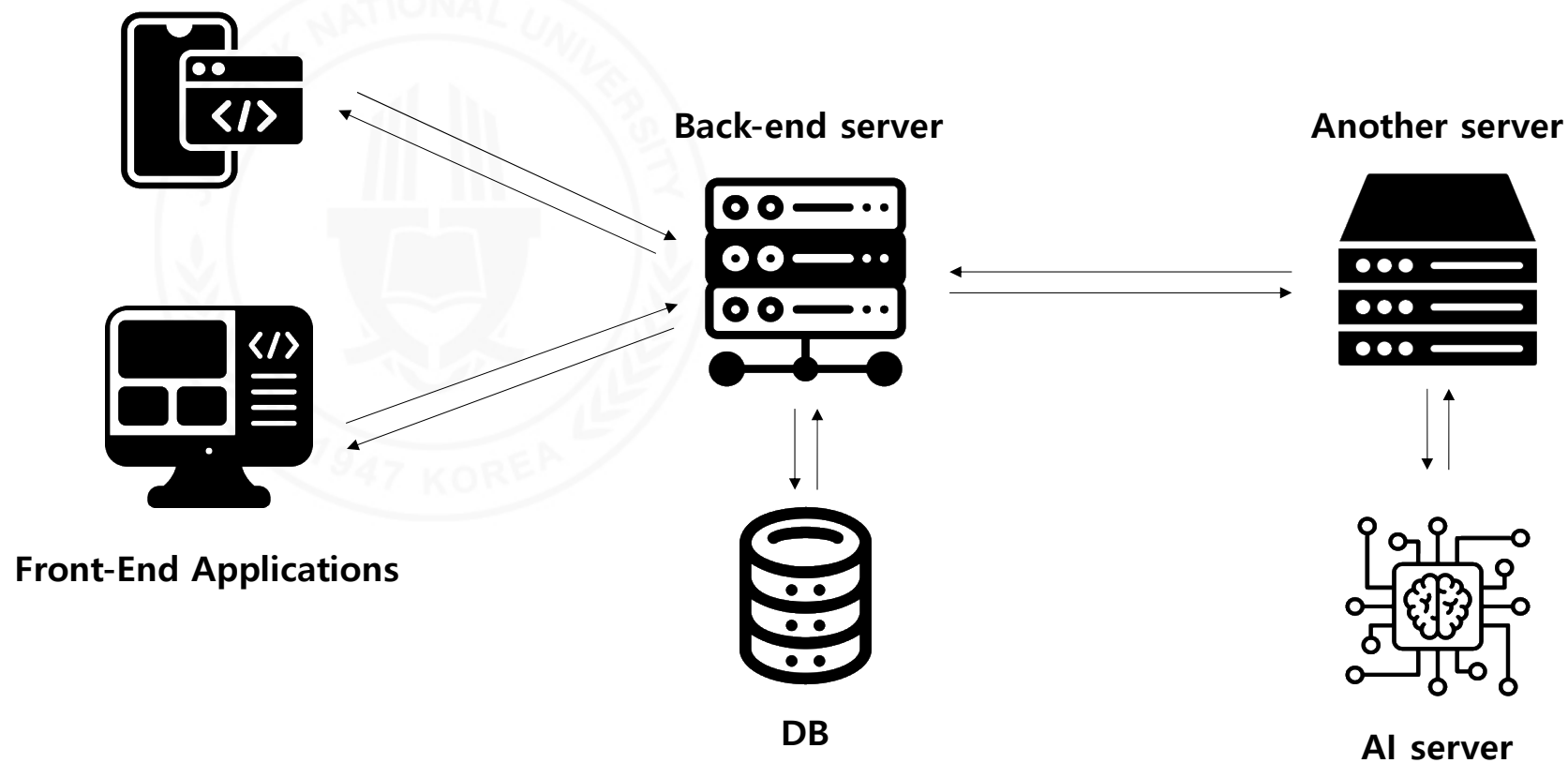
웹서비스 구성

Database



웹서비스 구성

AI Service



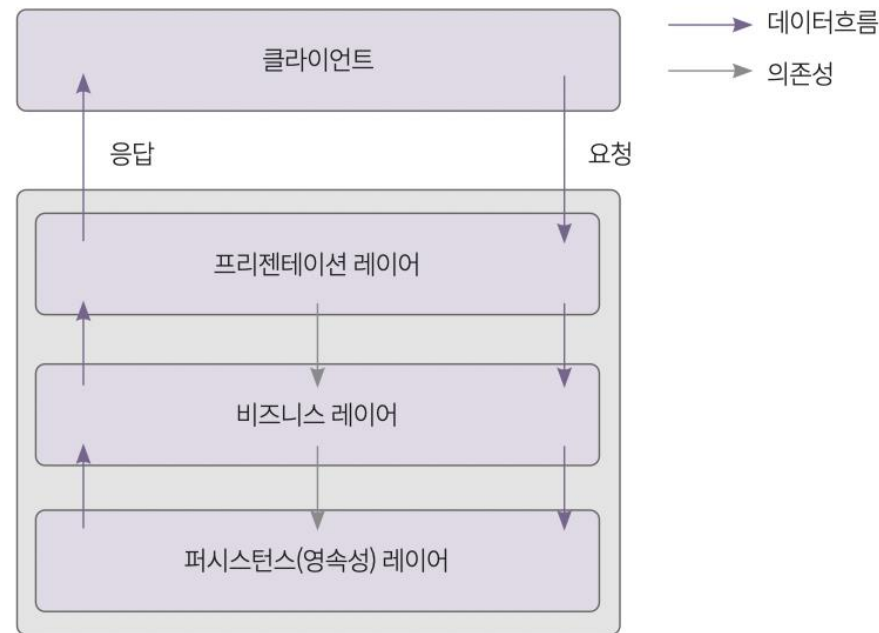
웹서비스와 백엔드

웹서비스와 백엔드

계층형 아키텍처

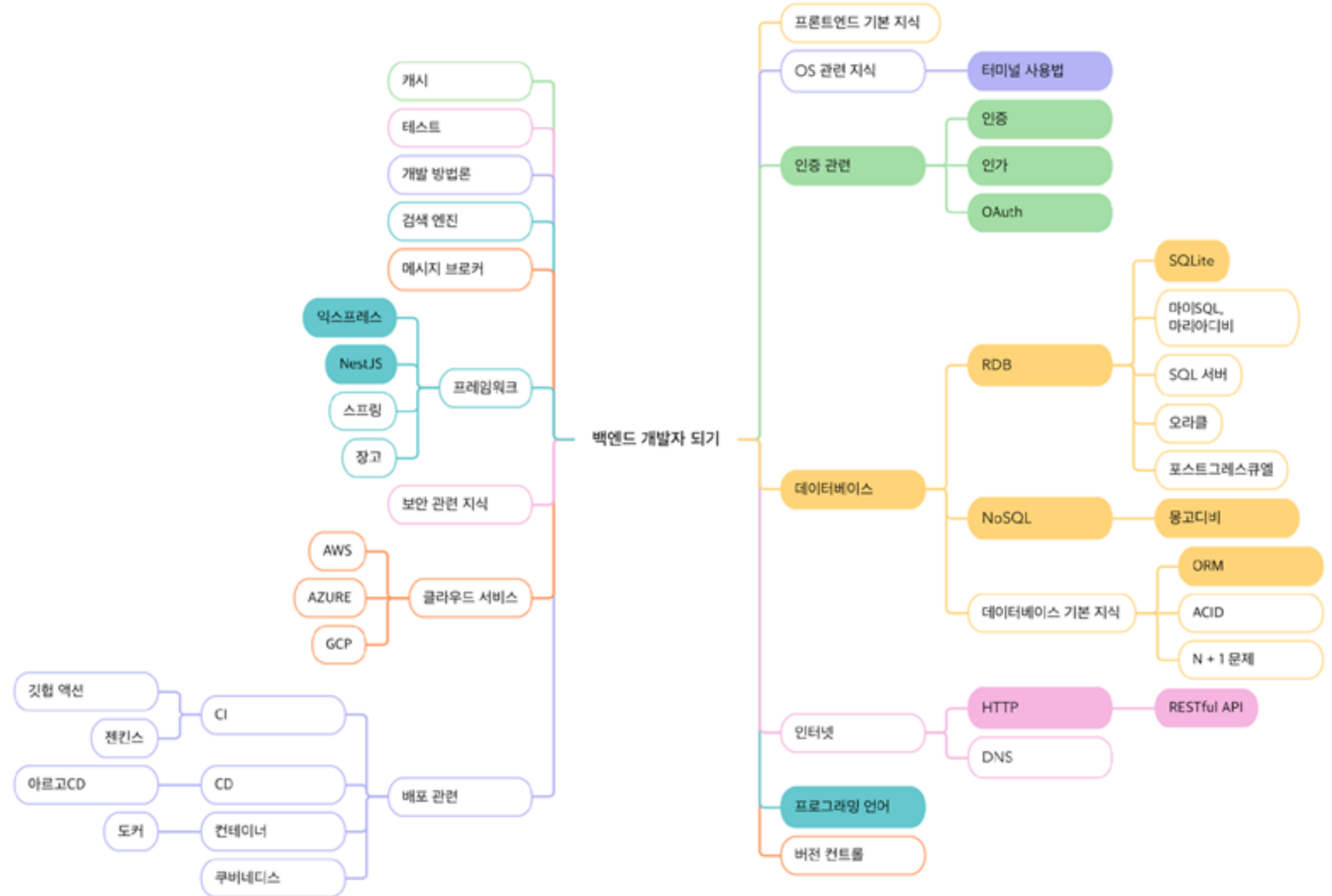
- 소프트웨어를 몇 가지 계층으로 나누어 만드는 방식
- 가장 흔한 아키텍처 형태 - 단순함, 이해하기 쉬움
- 각 계층이 논리적으로 분리됨 -> 소규모 애플리케이션에서 많이 채택
- 각 계층은 주어진 역할만을 수행, 계층간 의존성은 단방향임

▼ 계층형 아키텍처 예시



웹서비스 개발자의 역할

웹서비스와 백엔드



웹서비스와 백엔드

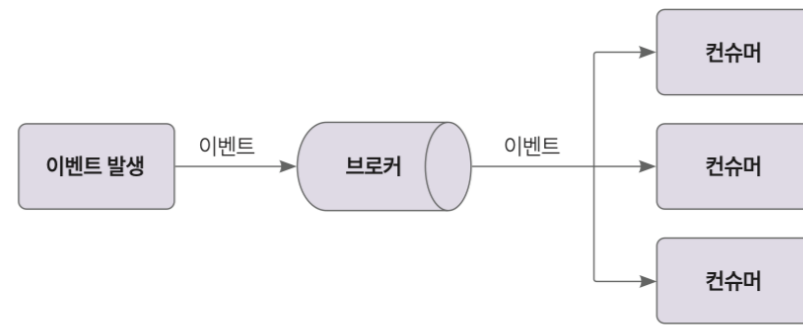
- 리눅스 또는 윈도우즈 서버등의 서버 운영체제 이해와 학습이 필수
- 명령줄 인터페이스(CLI)에 익숙해야함.
- 콘솔용 에디터 사용가능 해야함.
- 프로토콜(HTTP은 필수적으로 이해해야함)
- DNS : 도메인 주소를 IP로 변경해주는 것
- 웹서버와 WAS
- 데이터베이스
- 프로그래밍 언어
- 테스트
- 코드 배포

웹서비스 구성

이벤트 기반 아키텍처

- 이벤트의 상태 변화에 대응하는 소프트웨어 설계 패턴
- 이벤트 : 시스템에 영향을 주는 상황 의미 (사용자 로그인, 버튼 클릭 등)
- 구성 : 프로듀서 (이벤트 발생시킴) + 컨슈머 (전달하는 브로커 이벤트 수신)
- 모든 요청을 비동기로 처리하므로 확장성이 좋음

▼ 이벤트 기반 아키텍처 예시

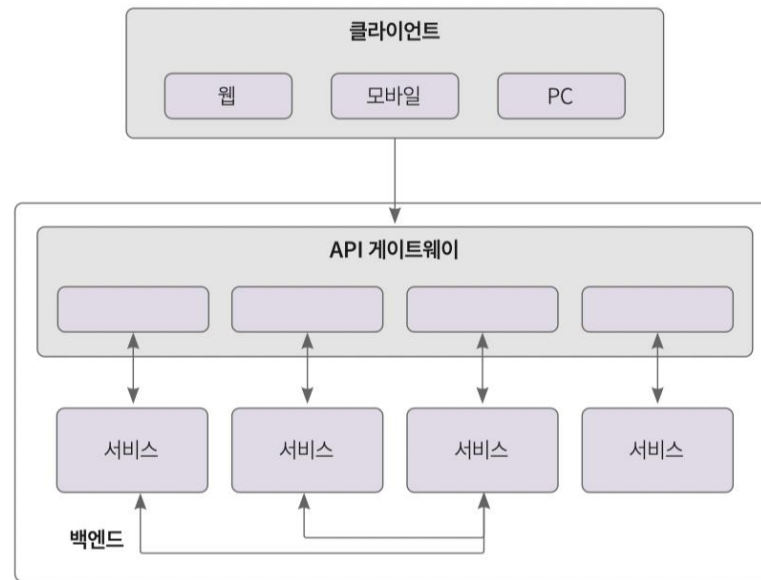


웹서비스 구성

마이크로 서비스 아키텍처

- 시스템을 여러 개의 작은 서비스로 나누어서 관리하는 설계 방식
- 각 서비스는 독립적으로 개발, 배포, 운영됨 - 구조적으로 분리됨
- 서비스 분리 - 독립적으로 업데이트 및 스케일링이 가능함
- 대규모 시스템 구축에 적합
- 각 서비스는 하나의 도메인을 책임지는 형태로 쪼개는 것이 좋음

▼ 마이크로서비스 아키텍처 예시



웹서비스 구성

백엔드 프로그래밍 언어

: 자바, 파이썬, C++, C#, 자바스크립트를 많이 사용

- 파이썬 : 데이터 분석 분야에서 인기 많음
- C++: 성능이 중요한 게임 서버 개발에서 많이 사용
- 자바 : 스프링 기반의 웹서비스에서 많이 사용
- C# : 닷넷 프레임워크, 마이크로소프트 기반 인프라와 함께 많이 사용
- 자바스크립트 : FE, BE에서 모두 인기

대표적인 웹 프레임워크 : express

- Nest JS, Next JS도 급성장 중

▼ 백엔드 프로그래밍 언어

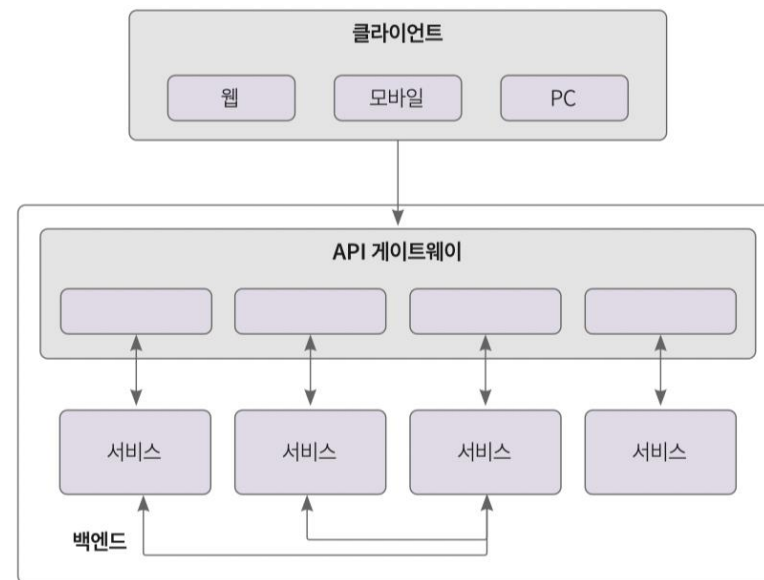


웹서비스 구성

데이터베이스

- 검색과 축적이 쉽도록 정리된 데이터의 모음
- 일반적으로는 데이터베이스 소프트웨어를 의미
- 크기는 RDB와 NoSQL로 나뉨

▼ 마이크로서비스 아키텍처 예시



웹서비스 구성

데이터베이스 – RDB

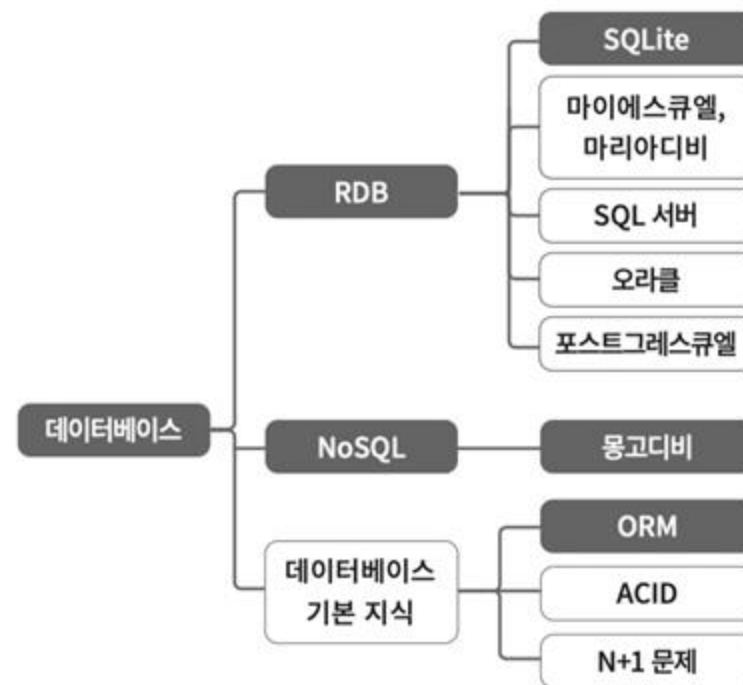
- ACID 트랜잭션
 - Atomicity 원자성 / 일관성 Consistency / 격리성 Isolation / 내구성 Durability을 의미
- SQL
 - Structured Query Language의 약자
 - 쿼리(데이터 검색)를 하는 프로그래밍 언어

웹서비스 구성

데이터베이스 - NoSQL(Not Only SQL)

- RDB의 문제들을 해결하기 위해 등장
- 클러스터 지원 -> RDB에 비해 스케일 아웃이 쉬움
- 테이블이 아닌 다른 방법으로 데이터 모델링을 함
 - Ex. 키 밸류, 컬럼, 오브젝트
- 키-밸류 스토어 : 다이나모디비, 카우치베이스가 유명
- 문서 스토어 : 몽고디비
- 와이드 컬럼 스토어 : 테이블과 유사하게 행과 열을 사용
 - 행마다 열의 타입이 다를 수 있음
 - 아파치의 카산드라가 유명

▼ 주요 데이터베이스

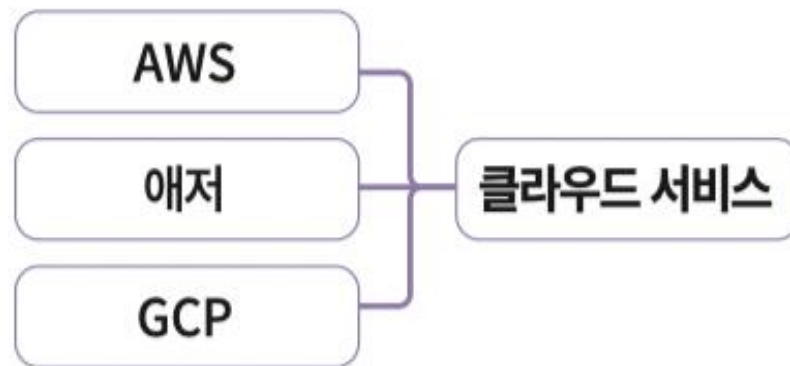


웹서비스 구성

클라우드 서비스

- 이전 : 많은 기업들이 IDC에 서버를 설치한 후 서비스
- 매우 많은 서버의 CPU와 메모리, 네트워크를 나누어서 서비스
- 서버의 자원을 탄력적으로 제공 가능함
- 클라우드를 제공하는 회사 : AWS, AZURE, GCP 등
- 2022년 1분기 기준 3사가 65%의 점유율 가짐
- 사용한 만큼만 요금을 냄

▼ 주요 클라우드 서비스



웹서비스 구성

클라우드 컴퓨팅의 유형

- IaaS(Infrastructure as a Service) : 서버 인프라 환경 서비스를 제공
 - 서버, 스토리지, 네트워크와 같은 컴퓨팅 자원을 사용
- PaaS(Platform as a Service): 애플리케이션 개발 및 실행 환경을 제공
 - 헤로쿠, AWS 일레스틱 빈스토크, 애저 앱서비스, GCP의 앱엔진등이 있음
- SaaS(Software as a Service): 브라우저 기반으로 소프트웨어를 사용할 수 있게 해줌
 - 마이크로소프트 오피스 365, 어도비 크리에이티브 클라우드 등

웹서비스 개발자의 역할

웹서비스 구성

▼ 클라우드 3사의 기능 비교표

기능	AWS	Azure	GCP
컴퓨팅	EC2, Elastic Beanstalk	Virtual Machines, AKS	Compute Engine, Kubernetes
스토리지	S3, EBS, Glacier	Blob Storage, ADLS	Cloud Storage, Cloud SQL
데이터베이스	RDS, DynamoDB, Redshift	Cosmos DB, Azure SQL	Cloud SQL, Bigtable
네트워킹	VPC, Route 53, Direct Connect	Virtual Network, ExpressRoute	Virtual Private Cloud, Cloud DNS
관리 & 거버넌스	IAM, CloudWatch, CloudFormation	Azure Policy, Azure Monitor	IAM, Stackdriver, Deployment Manager
인공지능, 머신러닝	SageMaker, Rekognition, Transcribe	Azure Machine Learning, Cognitive Services	Cloud AI Platform, Cloud ML Engine
사물 인터넷	IoT Core, Greengrass	IoT Hub, IoT Central	IoT Core, Cloud IoT Core
빅데이터	Redshift, Kinesis, EMR	HDInsight, Data Lake	BigQuery, Dataflow, Dataproc
개발자 툴	CodeStar, CodeCommit, CodeBuild	Azure DevOps, Visual Studio	Cloud SDK, Cloud Code, Cloud Build

JS 생태계

- 자바스크립트는 1995년 넷스케이프에서 HTML을 동적으로 표현할 목적으로 만든 모카(Mocha)라는 이름의 스크립트 프로그래밍 언어로 탄생
- 자바스크립트의 구조와 기능은 ECMAScript에 정의되어 있음
- ECMAScript의 최신버전은 ECMAScript 2021(ES11)

JS 생태계 - 패키지 매니저

- npm과 yarn이 대표적
- npm은 Node.js를 설치하면 함께 설치됨
- npm은 패키지 다운로드 속도문제, 보안문제를 가지고 있어, 이런 문제를 개선한 yarn을 사용하기도함
- 심볼릭 링크를 사용한 pnpm도 있음

JS 생태계 - 빌드 도구

- 빌드도구가 매우 많음
- 실제로는 여러개의 자바스크립트 파일을 하나로 뭉쳐주는 번들링 도구임
- 가장 유명한 것은 웹팩(webpack), 그러나 느림
- ES6이후에는 표준 모듈 시스템을 지원하는 ESBuild, Vite 등이 있음.
- ES6이전이라면 웹팩을 사용해야함



Thanks!

질문은 여기에...

ksl@jbnu.ac.kr