마이크로서비스 아키텍처 톺아보기

스타트업에서 MSA를 도입하면서 느꼈던 경험

백엔드 개발자 2년차 남혜민

발표자 소개

Career

#스타트업 #운영/기획 #PM #백엔드개발 #제품

Interest

#아키텍처 #DDD #FinOps #AWS

개요

1. 들어가기
2. MSA의 배경
3. MSA의 특징
4. 적용 사례
5. 정리

들어가기

MSA?

마이크로서비스 아키텍처

JD: 우대사항

주요업무

- 신규 서비스에 대한 Back-end Application 개발

자격요건

- Java 또는 kotlin 중 하나 이상의 언어에 능숙하신 분
- Spring framework, Spring Boot 기반 개발에 숙련되신 분
- 효율적인 비즈니스 로직 설계와 성능 최적화에 탁월하신 분
- RESTful API 설계 및 구현 경험이 있으신 분
- Kubernetes, Docker 기반 개발 경험이 있으신 분
- 학사 이상

우대사항

- 컴퓨터 공학 또는 전산 관련 전공인 분
- MSA(Micro Service Architecture) 설계 및 개발 경험이 있으신 분
- 질문과 공유를 좋아하고 함께 성장하려는 협업 마인드를 가진 분
- 자기 주도적이고 빠른 실행력을 가진 분

주요업무

- 광고 시스템 백엔드, 어드민 툴 개발 및 유지보수
- 광고 노출 최적화 및 데이터 서빙 개발
- 광고 랭킹 로직 최적화 및 budget 관리 개발
- 광고 결제 및 정산 개발
- 시스템 성능 최적화 및 모니터링

자격요건

- Java/Kotlin 중에서 하나 이상의 프로그래밍 언어에 능숙한 분
- Spring Framework, Spring Batch, ORM(JPA, Hibernate)를 잘 이해하고 있고 적절히 활용할 수 있으신 분
- NoSQL(Redis, Elastic Search, Kafka) 개발 및 운영 경험이 있으신 분
- Restful API 설계/구현 경험이 있으신 분
- 복잡한 도메인을 빠르게 이해하고 설계와 구현부터 운영까지 해본 경험이 있으신 분

우대사항

- 광고 시스템에 전반적인 이해가 있는 분
- MSA(Microservice Architecture), EDA(Event Driven Architecture) 개발 경험 있으신 분
- 코드 리뷰에 긍정적이고, 테스트 및 이해하기 쉬운 좋은 코드 작성에 관심이 많으신 분

Microservice

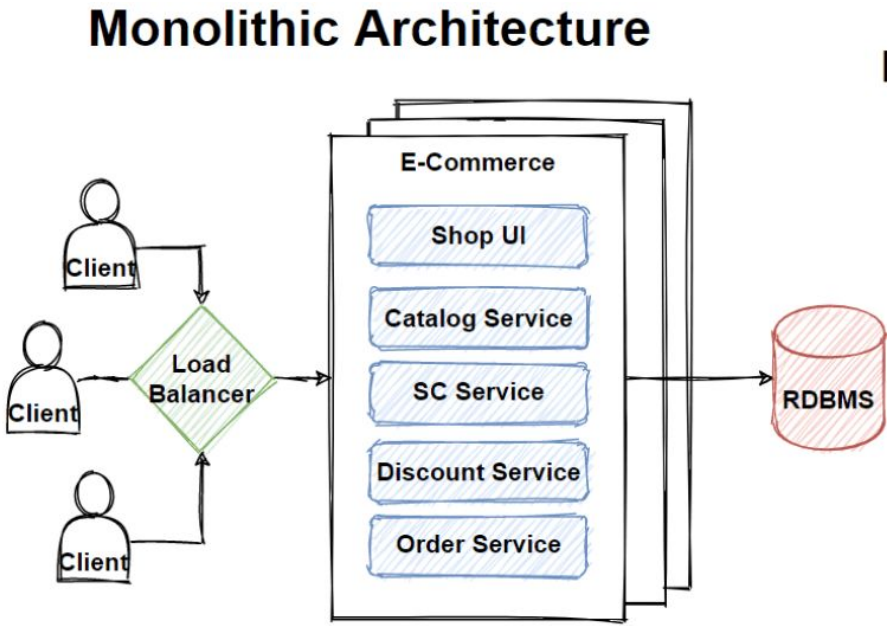
마이크로서비스

애플리케이션이 느슨하게 결합된
서비스의 모임으로 구조화하는
서비스 지향 아키텍처(SOA) 스타일의
일종인 소프트웨어 개발 기법

MSA 이전

모놀리식 (Monolithic) 아키텍처

애플리케이션의 모든 기능이 하나의 독립적인 단위로 배포되는 아키텍처



https://stories.salesforcecodex.com/2023/05/salesforce/what-is-monolithic-architecture

모놀리식 아키텍처 장점

단순성

개발, 배포, 테스트가 간단함

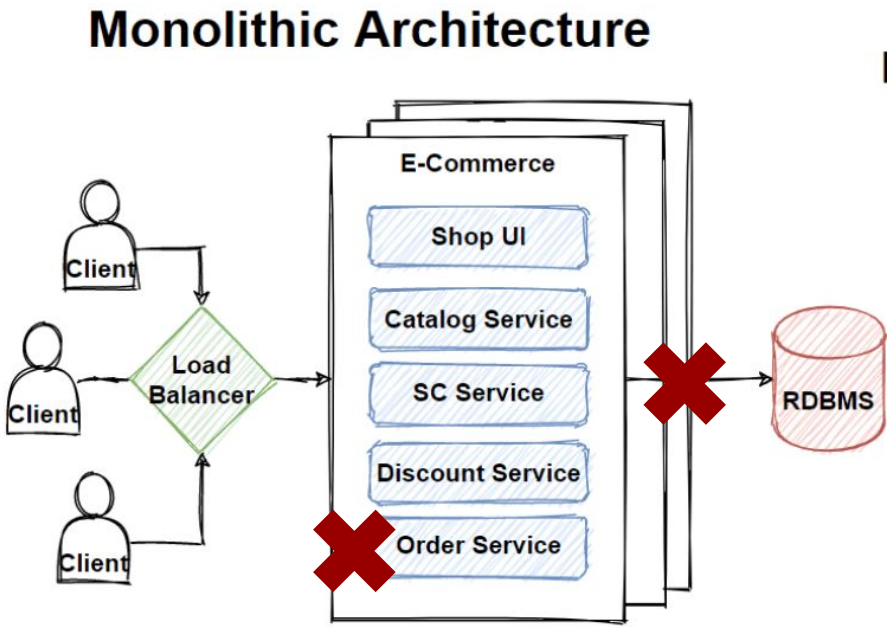
낮은 복잡도

복잡한 통신이나 분산 시스템이 필요 없음

비용 효율성

인프라 및 인력에 대한 투자 감소

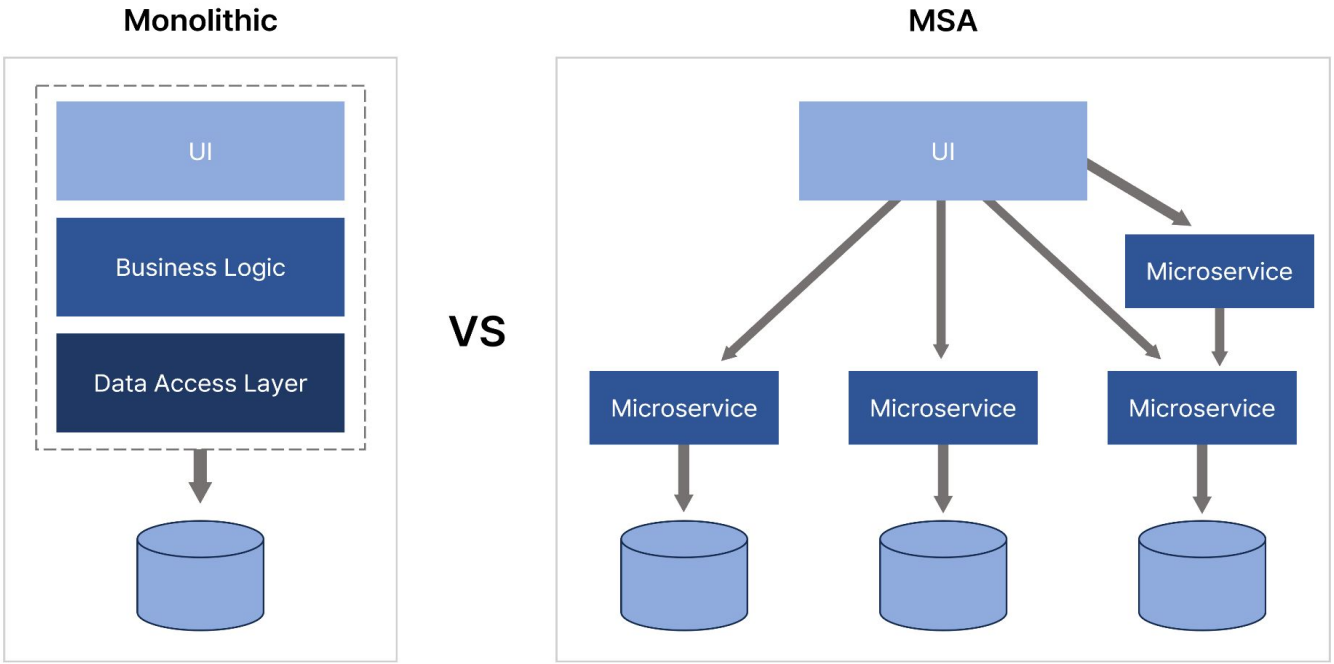
모놀리식의 한계


Monolithic Architecture
Client
Client
Client
Load Balancer
E-Commerce
Shop UI
Catalog Service
SC Service
Discount Service
Order Service
RDBMS

모놀리식 아키텍처 단점

MSA 특징

MSA(Microservice Architecture) 의 등장



https://metanetglobal.com/bbs/board.php?bo_table=tech&wr_id=38

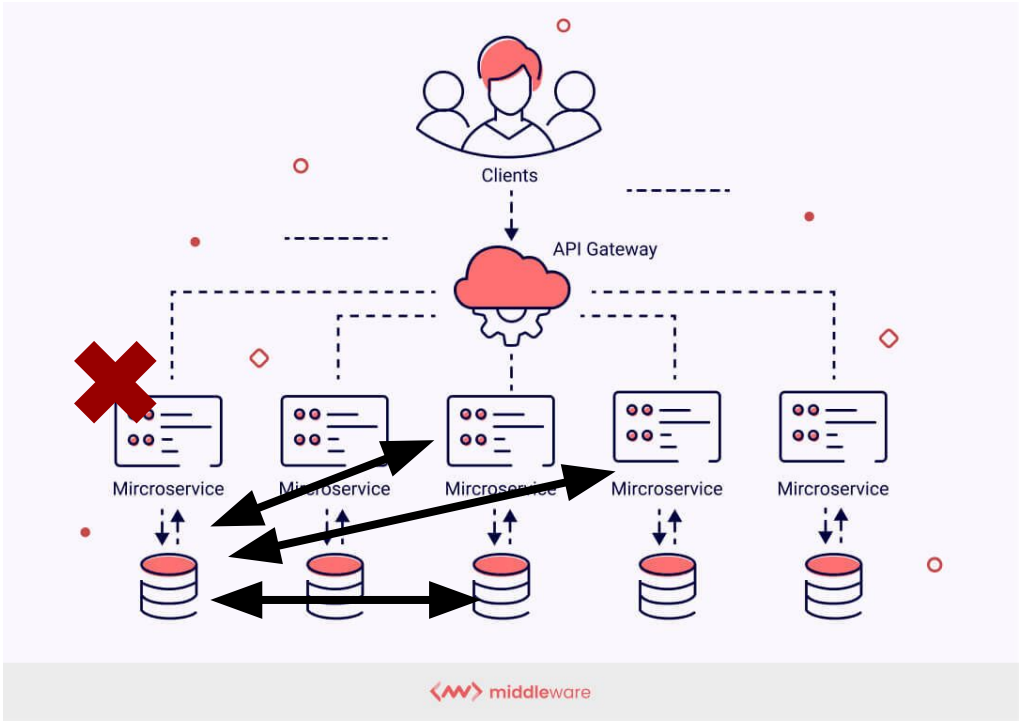
MSA의 장점

대규모 서비스 확장

관리 용이성
(기술 다양성)

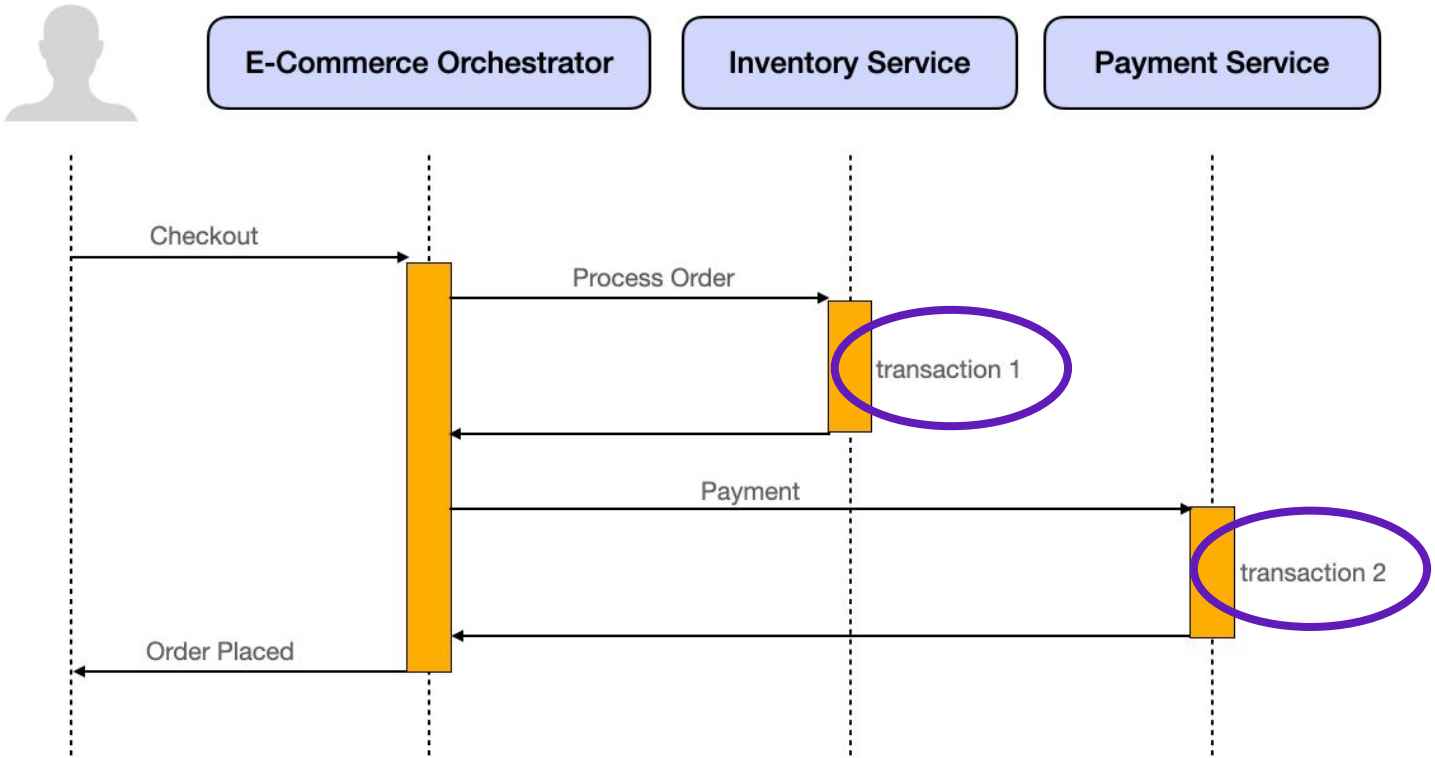
무중단 관리

MSA의 한계



https://middleware.io/blog/microservices-architecture

분산 트랜잭션



https://middleware.io/blog/microservices-architecture

MSA의 단점

분산 시스템의 복잡성

초기 개발 및
운용 비용 부담 증가

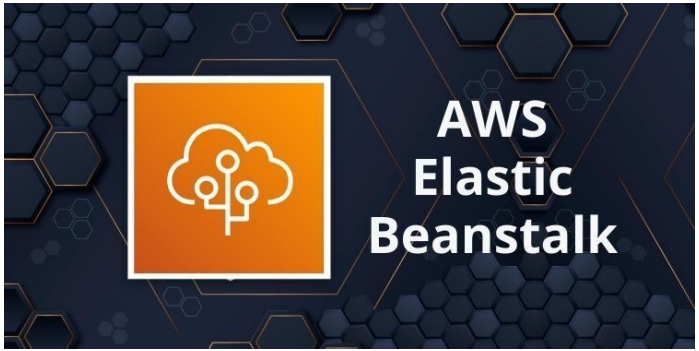
네트워크 지연

적용 사례

모놀리식

사례: 모놀리식의 장점 - 단순성

- MVP 목표: 학생에게 특정 검사를 수행하고, 그에 따른 학습 훈련을 수행하는 서비스
  - 복잡한 요구사항이 없기 때문에 **Elastic Beanstack** 로 구현



사례: 모놀리식의 장점 - 비용 효율성

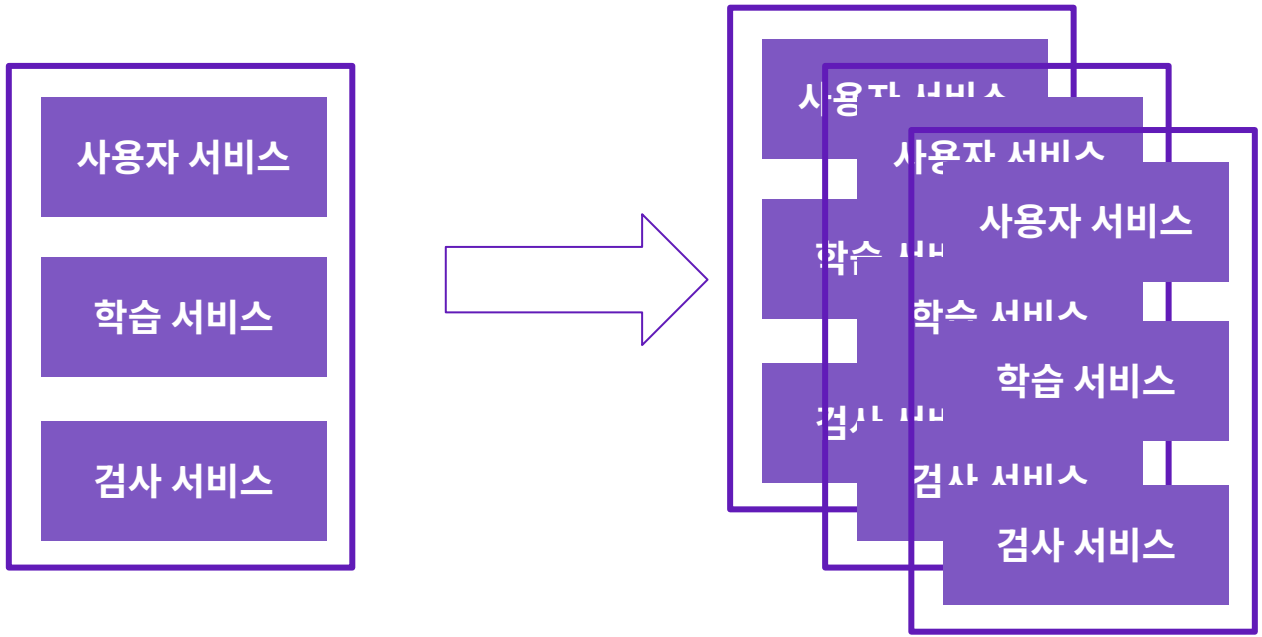
- 1인 백엔드 개발자 10일만에 MVP 수준의 서버 구축
- (극단적이지만) AWS Free Tier 로도 충분히 MVP 구현 가능

사례: 모놀리식의 단점 - 단일 장애점

- 학습 서비스 (학습 생성) 에서 문제가 발생했을 때, 로그인 & 검사 진행도 불가능

사례: 모놀리식의 단점 - 단일 장애점 및 확장성 한계

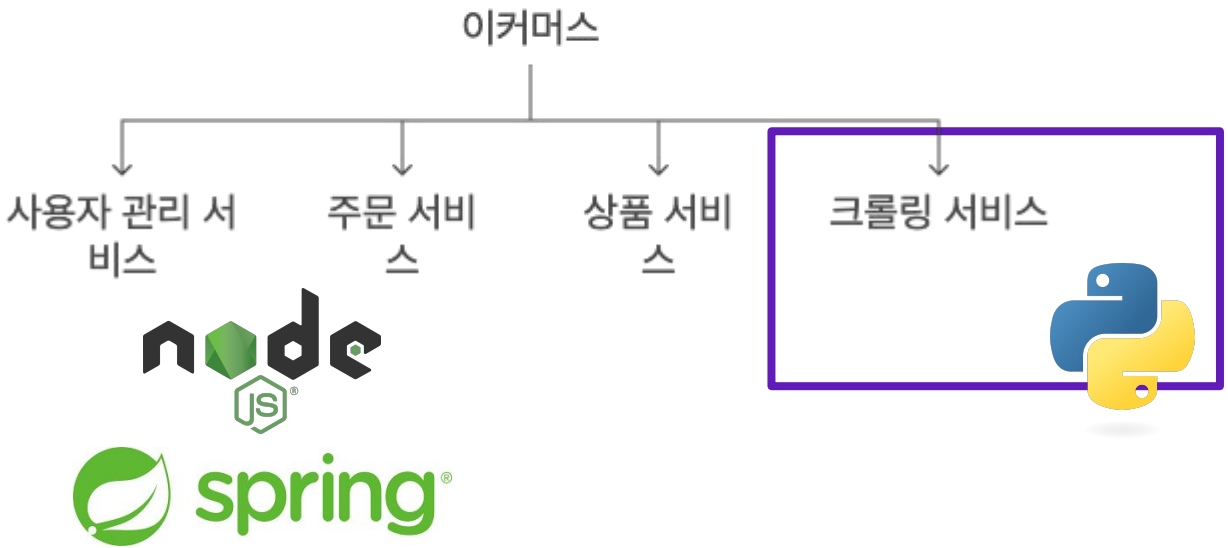
- 수평적 서버 스케일링을 하면 무조건 모든 서비스들이 N개씩 증가



MSA

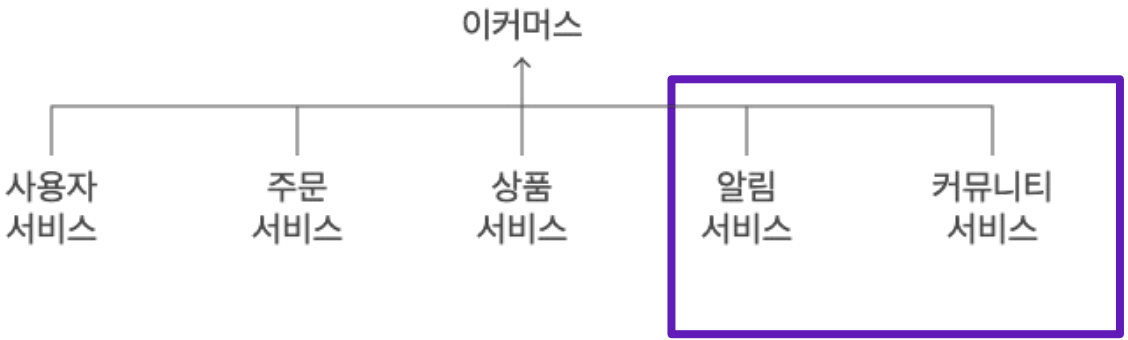
사례: MSA의 장점 - 관리의 용이성

- 각 기능에 따른 다양한 기술 스택 선택 가능



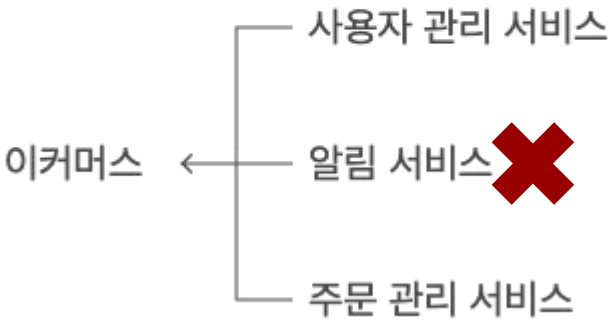
사례: MSA의 장점 - 관리의 용이성

- 한 스프린트 내 신규 기능(알림 / 커뮤니티)을 동시에 2개 나가야 함
- 코드 충돌 적게 추가 개발 가능



사례: MSA의 장점 - 무중단 관리

- 고객 주문 완료 → 카카오톡 알림 발송



사례: MSA의 단점 - 초기 개발 및 운용 비용 부담 증가

- 목표: 3주 안에 제로부터 사용자가 사용할 수 있는 MVP 출시
- 상황: 기획, 디자인도 불명확 → 빠르게 약속된 기간 내에 개발을 해야 함
- 의사결정
  - 개발적 완성도를 위해 MSA 로 8개 서비스로 분리
  - 설계 및 구축에 2주 넘게 소요
- 결과
  - 기다린 사용자들에게 양해를 구하고 마감 연장
  - ECS 를 선택하여 서버 및 인프라 비용 과도하게 지출

정리

모놀리식 vs MSA 특징 요약

특성	모놀리식	MSA
배포	간단	복잡
확장성	제한적	뛰어남
유지보수	어려움	용이
성능	빠름	네트워크 지연
기술 스택	단일	다양함

마무리

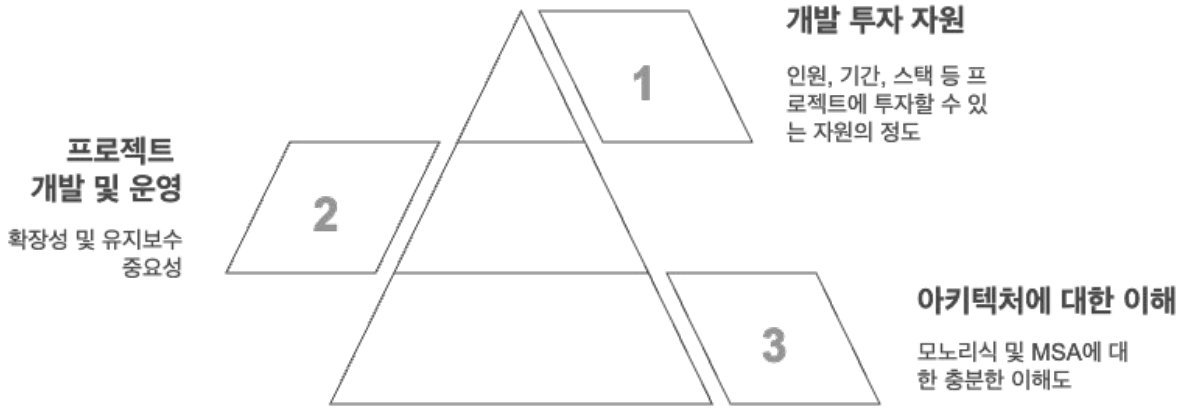
Microservice Architecture

작은 애플리케이션을 쪼개서 독립적인 서비스 단위로 설계된

대규모 서비스 아키텍처 중 하나의 방법

단, **MSA가 항상 만능은 아니다!**

마무리: (스타트업에서) 아키텍처 선택 기준



참고자료

https://microservices.io

https://middleware.io/blog/microservices-architecture/

https://stories.salesforcecodex.com/2023/05/salesforce/what-is-monolithic-architecture/

https://metanetglobal.com/bbs/board.php?bo_table=tech&wr_id=38

Q & A