붙임 ① (표준서식, 훈련생 작성용)

IT 프로젝트 개발 계획서(안)

(「커피전문점 마이크로서비스 통합 플랫폼 (TORI COFFEE)」)

2025.12.18~2026.01.20

훈련과정명 (소속)	클라우드 데브옵스 프론트엔드&백엔드 자바(JAVA) 풀스택 (중앙정보기술인재개발원)
팀 명	브리또
팀장 성명	서재홍

팀원 성명	박윤호
팀원 성명	하성호
팀원 성명	이주희
지도교사	정지웅

1. 개 요

□ 프로젝트명

- TORY COFFEE 커피전문점 마이크로서비스 통합 플랫폼

□ 개발 배경 및 필요성

- 마이크로서비스 아키텍처는 소프트웨어를 작은, 독립적인 서비스로 분할하는 개발 방법론임
- 물리적인 하드웨어 인프라 구축 없이 클라우드 플랫폼을 통한 서비스를 구현하고자 함
- 대규모 시스템을 더 효율적으로 구축하고 유지 보수하는 데 도움이 됨
- 서비스별 독립적인 배포 및 확장이 가능하여 운영 효율성 향상이 됨

□ 개발목표

- Java/Spring Boot 기반 마이크로 서비스 통합 플랫폼 구축
- Spring Cloud를 활용한 서비스 디스커버리 및 API 게이트웨이 구

현

- RabbitMQ를 통한 비동기 메시지 처리 시스템 구축
- AWS EKS/Docker 기반 클라우드 배포 환경 구성
- JWT 기반 보안 인증 시스템 구현

□ 기존 서비스(플랫폼)에 관한 고찰

- 마이크로서비스 아키텍처를 구현하고 관리하기 위해 다양한 플랫폼과 도구들이 개발됨
- Kubernetes, Docker, Amazon Web Services (AWS) 등이 있음
- Spring Boot와 Thymeleaf로 코드를 구현하여 Docker를 활용한 AWS EKS 배포

□ 산출물

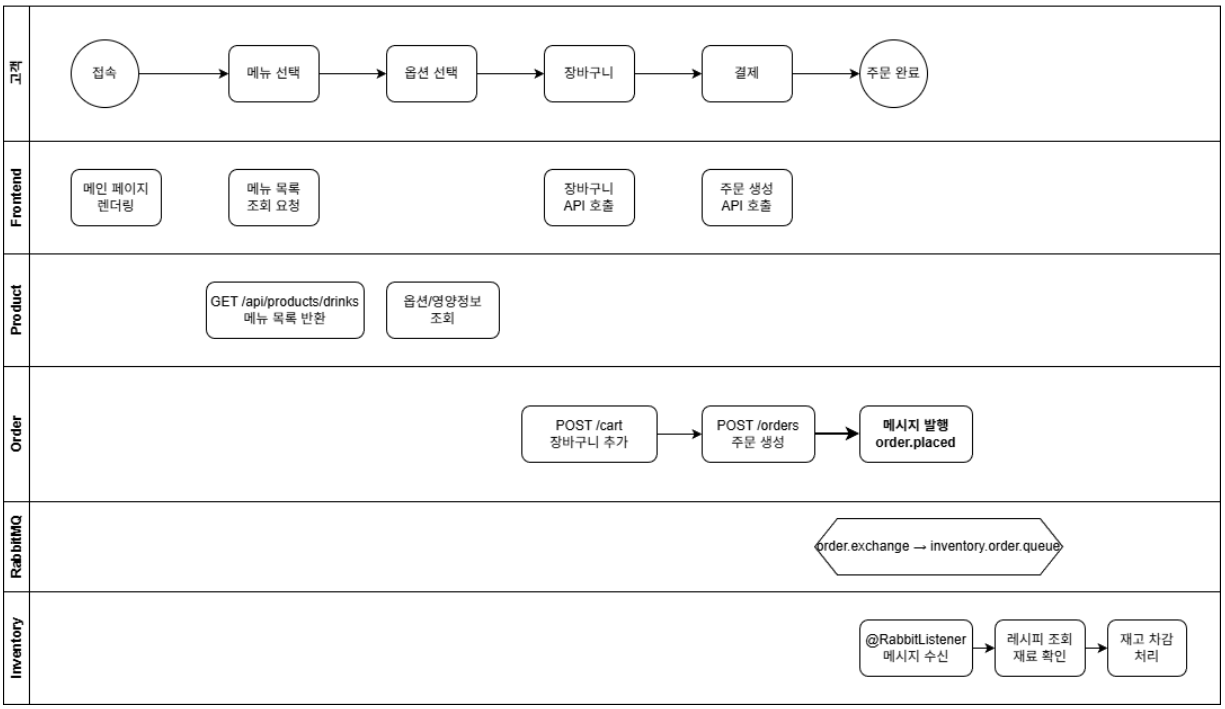
산출물	설명	담당
ERD	데이터베이스 구조 설계서	서재홍
시스템 아키텍처	MSA 구조 및 기술 스택 문서	서재홍
화면설계서	각 화면별 UI 상세 설계	하성호
멘토링 일지		박윤호
발표 PPT	최종 발표 자료	이주희

2. 주요 기능

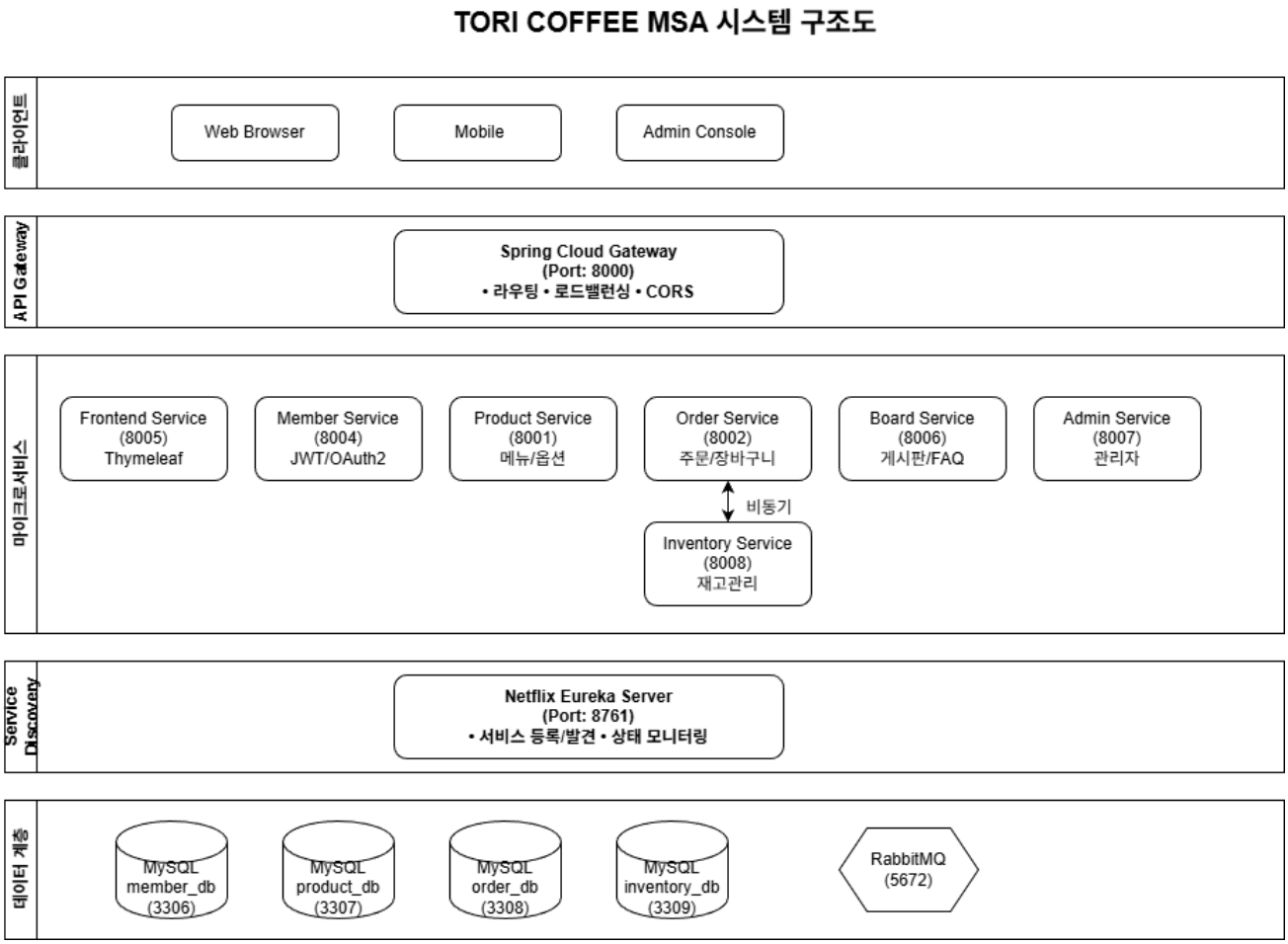
주요 기능 명칭	주요 기능에 관한 상세 설명(명세)
서비스 디스커버리	Netflix Eureka 서버를 구축하여 마이크로 서비스 등록 및 발견
API 게이트웨이	Spring Cloud Gateway를 구축하여 서비스 라우팅 및 로드벨런싱
메시지 큐잉	RabbitMQ를 통한 마이크로 서비스 간 메시지 발행 및 구독
회원 관리	회원 등록, 로그인, 인증관리
상품 관리	커피 메뉴 등록, 조회, 옵션 및 영양정보 관리
주문 관리	장바구니, 주문 생성, 주문 상태 관리, 결제 처리
재고 관리	재료 재고 현황 관리, 주문 시 자동 재고 차감

게시판	공지사항, FAQ, 1:1 문의 및 답변 관리
관리자 기능	회원 관리, 상품 관리, 문의 관리, 통계

커피 주문 처리 흐름도 (비즈니스 프로세스)



3. 시스템 구조

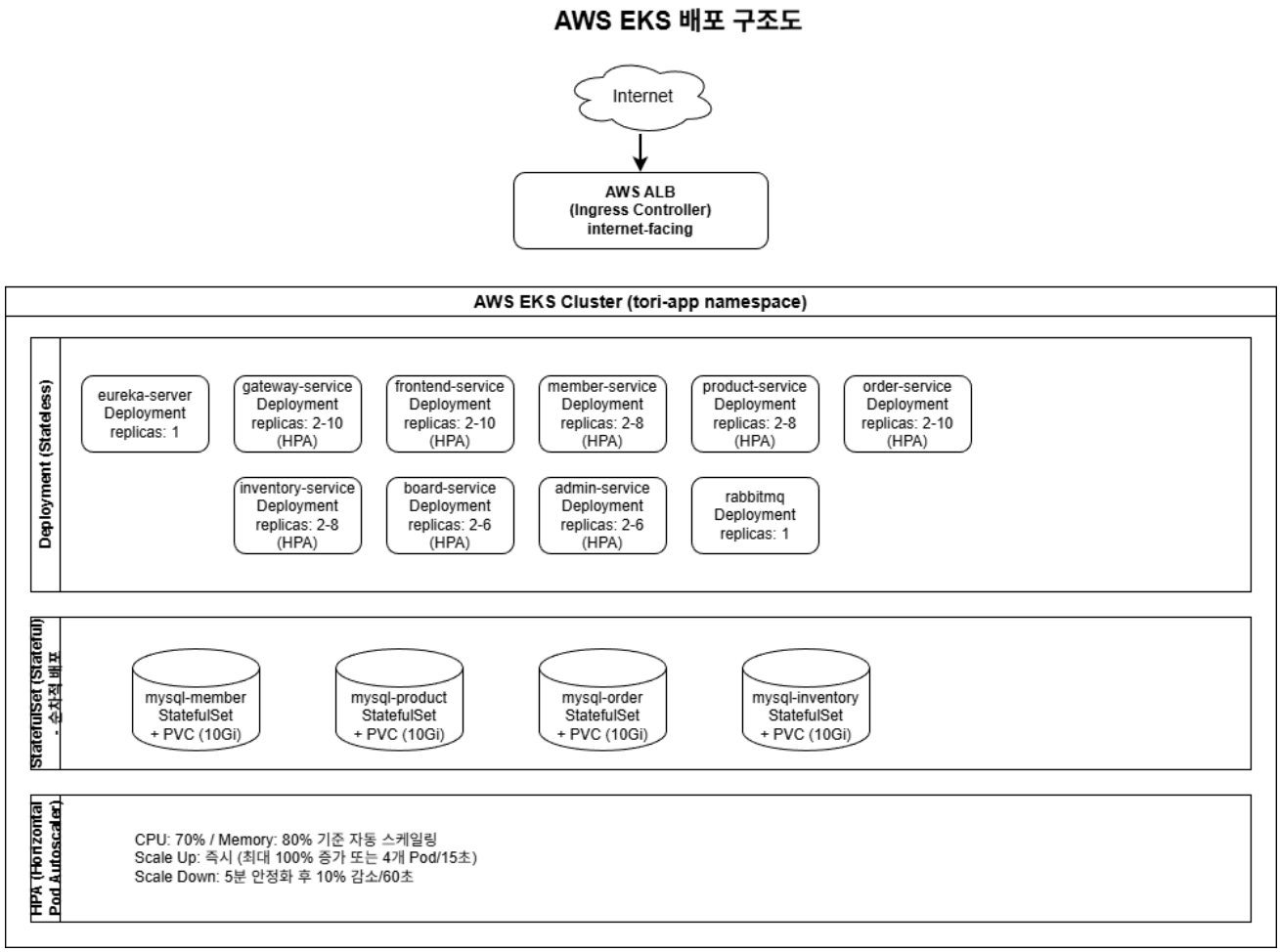


3.1 구성요소

구성요소	설명	개발환경 및 구현방법
Service Discovery	마이크로서비스 등록/발견, 상태 모니터링	Netflix Eureka Server (8761)

API Gateway	요청 라우팅, 로드벨런싱, CORS 처리	Spring Cloud Gateway (8000)
DB	회원 상품, 주문, 재고 데이터 관리	MYSQL 8.0 (서비스별 분리)
Mwssage Queue	주문 시 재고 서비스로 메시지 비동기 전달	RabbitMQ 3.x
Backend Services	회원, 상품, 주문, 게시판, 재고, 관리자 서비스	Spring Boot 3.1.5, JPA
Frontend	웹 사용자 인터페이스, API 프록시	Thymeleaf, RestTemplate
Container	서비스 컨테이너화 및 오케스트레이션	Docker, Kubernetes (AWS EKS)

3.2 Kubernetes 배포 구조



구성요소	Workload Type	설명

MYSQL (4개)	Statefulset	영구 스토리지(PVC), 순차적 배포/삭제, Headless Service
RabbitMQ	Deployment	Stateless 메시지 브로커
Spring 서비스	Deployment	Stateless 애플리케이션, HPA로 자동 스케일링
Ingress	ALB	AWS ALB 로드벨런서, 경로 기반 라우팅

HPA

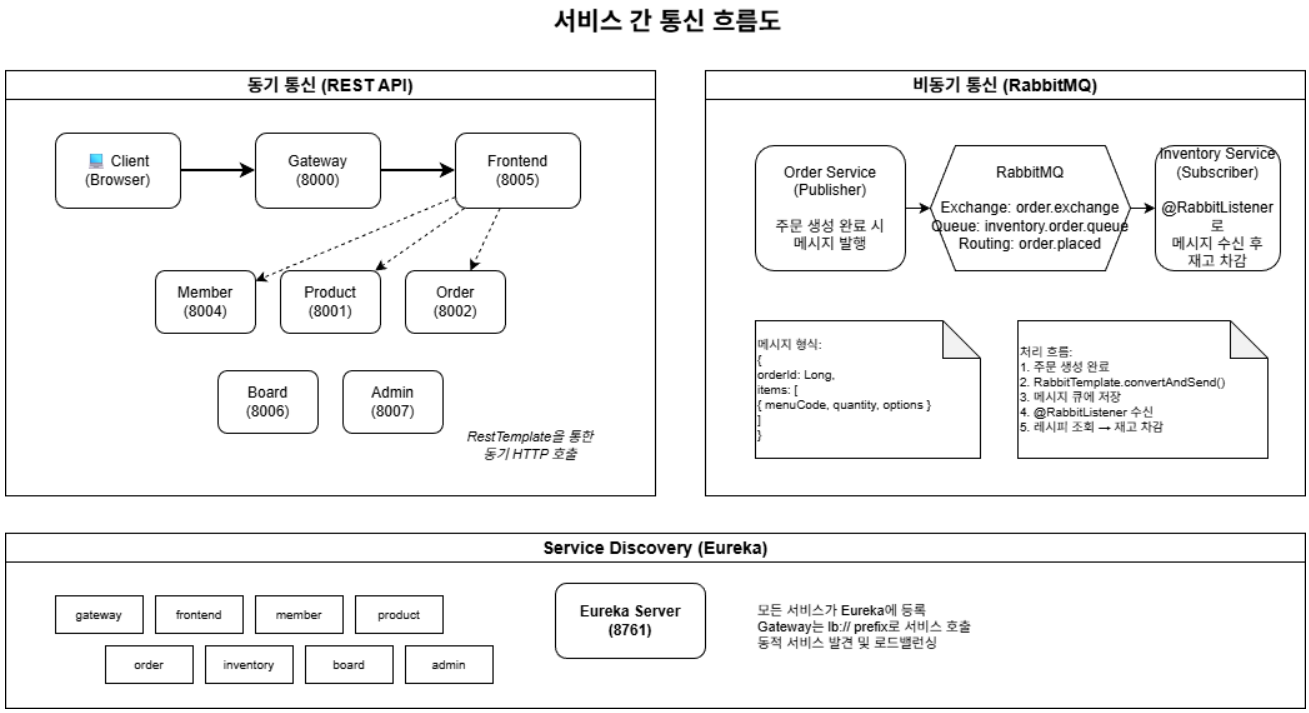
- CPU 70%, Memory 80% 기준 자동 스케일링
- 최소 2개 ~ 최대 10개 pod

3.3 서비스 포트 구성

서비스	포트	데이터베이스
Eureka Server	8761	

Gateway Service	8000	
Frontend Service	8005	
Member Service	8004	member_db (3306)
Product Service	8001	product_db (3307)
Order Service	8002	order_db (3308)
Board Service	8006	member_db (3306)
Inventory Service	8008	inventory_db (3309)
Admin Service	8007	member_db (3306)

3.4 서비스간 통신 흐름도



4. 프로젝트 개발 일정 (WBS)

4.1 WBS 개요

단계	기간	주요 활동
1. 계획/분석	1주차	목표 정의, 요구분석, 계획서 작성
2. 설계/구현	2주차	시스템 구조 설계, 기능 구현, DB 구축
3. 통합/배포	3주차	기능 통합, 테스트, 배포, 발표

4.2 상세 WBS

작 업 명	담당자	개발 일정			비고
		1 W	2 W	3 W	
1. 계획					
- 목표정의	전체				2 일 소요
- 계획서 작성	전체				
- 요구사항 분석	전체				
- 계획서 작성	전체				
2. 설계					

- 시스템 아키텍처	전체				4일 소요
- ERD 설계	전체				
- 화면 설계	서재홍				
3. 구현					
- Eureka Server	전체				3일 소요
- Gateway Service	전체				
- Member Service	하성호				
- Product Service	이주희				
-Order Service	박윤호				
- Board Service	하성호				
- Inventory Service	박윤호/이주희				
- Admin Service	하성호				
- Front Service	전체				
4. 통합 및 테스트					
- 서비스 통합	박윤호/서재홍				9일 소요
- RabbitMQ 연동	박윤호				
- 단위/통합 테스트	전체				
- 오류 수정	전체				
5. 배포 및 발표					
- Docker 컨테이너화	서재홍				2일 소요
- k8s(EKS) 배포	서재홍				
- 최종 발표자료 작성	전체				

5. 조직 구성 및 업무 분장

5.1 팀원 구성

이름	역할	담당업무
서재홍	시스템 아키텍트/인프라/프론트엔드	Frontend Service 개발, 시스템 아키텍처 설계, Docker/k8s 배포
박윤호	시스템 아키텍트/백엔드 개발/DB 설계	Order service, Inventory Service 개발, RabbitMQ 설계, REST API 구현, ERD 설계, 데이터 모델링
이주희	시스템 아키텍트/백엔드 개발/DB 설계	Product Service, Inventory Service 개발, REST API 구현, ERD 설계, 데이터 모델링
하성호	시스템 아키텍트/백엔드 개발/DB 설계	Member Service, Board Service, Admin Service 개발 REST API구현, ERD 설계, 데이터 모델링

5.2 서비스별 담당자

서비스	담당 업무	담당자
Eureka Server	서비스 디스커버리, 상태 모니터링	전체
Gateway Service	API 라우팅, 로드벨런싱, CORS	전체
Frontend Service	웹 UI, Thymeleaf 템플릿, API 프록시	서재홍
Member Service	회원가입, 로그인, JWT	하성호
Product Service	메뉴 조회, 옵션/영양정보 관리	이주희
Order Service	장바구니, 주문 생성, 결제처리	박윤호
Inventory Service	재고 관리, 주문 시 자동 재고 차감	박윤호, 이주희
Board Service	공지사항, FAQ, 1:1 문의 관리	하성호
Admin Service	관리자 기능, 회원/문의관리	하성호
RabbitMQ	주문.재고 연동 메시지 큐 설계	박윤호
Docker/K8s 배포	컨테이너화, EKS 배포, HPA 설정	서재홍
ERD/DB 설계	데이터 모델링, MySQL 스미카 설계	전체

5.3 역할별 상세 업무

역할	담당업무
시스템 아키텍트	MSA 아키텍처 설계, 서비스 간 통신 흐름 설계, Spring Cloud 구성 (Eureka, Gateway), RabbitMQ 메시지 큐
백엔드 개발	각 마이크로서비스 비즈니스 로직 구현, JPA 엔티티 및 Repository 개발, REST API 구현, API 연동
프론트엔드 개발	Thymeleaf 템플릿 개발, UI/UX 구현, 반응형 웹 디자인
DB 설계	ERD 설계, 서비스별 MySQL 데이터베이스 구축, 테이블 설계 및 관계 정의
인프라/DevOps	Docker 컨테이너화, Kubernetes(AWS EKS) 배포, ECR 이미지 관리, HPA설정

6. 개발 환경

6.1 개발 도구

구분	도구	용도
IDE	Intellij IDEA / VS Code	코드 작성 및 디버깅
버전관리	Git, GitHub	소스 코드 버전 관

		리
빌드도구	Gradle	프로젝트 빌드 및 의존성 관리
API 테스트	curl	REST API 테스트
DB 관리	H2 Console(로컬), MySQL(K8S)	데이터베이스 조회/관리
컨테이너	Docker Desktop / Docker CLI	이미지 빌드 및 컨테이너 실행
K8s 관리	kubectl	클러스터 리소스 관리
AWS	AWS Console	EKS, ECR, ALB등 리소스 관리
AWS	AWS CLI	AWS 리소스 CLI 관리
EKS	eksctl	EKS 클러스터 생성/관리

6.2 기술 스택

구분	기술	버전	비고
Language	Java	17	LTS
Framework	Spring Boot	3.1.5	
Cloud Framework	Spring Cloud	2022.0.4	
Service	Netflix Eureka		Spring Cloud

Discovery			Netflix
API Gateway	Spring Cloud Gateway		
ORM	Spring Data JPA/Hibernate		
Database (로컬)	H2		In-memory, 개발용
Database (K8s)	MySQL	8.0	StatefulSet 배포
Message Queue	RabbitMQ	3.X	Spring AMQP
Authentication	JWT (JJWT)	0.11.5	HS512 알고리즘
Security	Spring Security		
Template Engine	Thymeleaf		
Build Tool	Gradle	7.x+	
Container	Docker	Latest	
Orchestration	Kubernetes		AWS EKS
Container Registry	AWS ECR		
Load Balancer	AWS ALB		Ingress Controller
Storage	AWS EBS	gp2	PersistentVolume

7. 기대 효과 및 활용 분야

7.1 기대효과

- 확장성 향상 : 마이크로서비스는 각 서비스를 독립적으로 배포하고 확장할 수 있으므로 시스템의 전체적인 확장성과 유연성을 향상함
- 개발 생산성 증대 : 마이크로서비스는 각 서비스를 작은 단위로 분할하여 개발 및 배포 프로세스를 가속화
- 기술 다양성 : 마이크로 서비스 아키텍처는 각 서비스가 독립적으로 개발되므로 다양한 기술을 적용 가능
- 클라우드 최적화 : 마이크로서비스는 클라우드 네이티브 환경에서 개발되고 운영되는 데 이상적임
- 장애 격리 : 한 서비스의 장애가 전체 시스템에 영향을 미치지 않음

7.2 활용 분야

- 전자 상거래 : 온라인 쇼핑몰, 결제 시스템
- 물류/배송 : 주문관리, 배송 추적 시스템
- F&B : 프렌차이즈 주문/ 재고 관리 시스템

8. 위험 요소 및 대응 방안

위험 요소	영향도	대응 방안
서비스 간 통신 장애	높음	Circuit Breaker 패턴 적용, 재시도 로직

		구현
데이터 일관성 문제	중간	Saga 패턴 검토, 이벤트 소싱 고려
인증 토큰 탈취	높음	HTTPS 적용, 토큰 만료 시간 단축, Refresh Token 활용
서비스 장애 전파	높음	헬스 체크, 자동 복구, 로드벨런싱
배포 실패	중간	Blue-Green 배포, 롤백 전략 수립