Lecture 20 - Contents

An overview of the main sections in this lecture.

This outline is for guidance. Navigate the slides with the left/right arrow keys.

Lecture 20:

Capstone Project

End-to-End Medical AI System

Ho-min Park

homin.park@ghent.ac.kr

powersimmani@gmail.com

Project Overview

프로젝트 범위

- 의료 AI 시스템 전체 개발 과정 경험
- 데이터 수집부터 배포까지 End-to-End 구현
- 실제 임상 환경을 고려한 설계
- 팀 협업 및 프로젝트 관리 실습

최종 목표

- 완성된 의료 AI 시스템 프로토타입
- 상세한 기술 문서 및 발표 자료
- 성능 평가 및 검증 결과
- 배포 가능한 수준의 코드

평가 기준

- 시스템 설계 및 구현 품질 (40%)
- 성능 및 안정성 (30%)
- 문서화 및 발표 (20%)
- 창의성 및 혁신성 (10%)

주요 마일스톤

1. 시스템 설계 및 계획 (Week 1-2)
2. 구현 및 개발 (Week 3-6)
3. 배포 및 검증 (Week 7-8)

4 최종 발표 및 평가 (Week 9)

Part 1/3:

System Design and Planning

1. Requirements Analysis
2. Architecture Planning
3. Technology Stack Selection
4. Data Pipeline Design
5. Model Selection Criteria
6. Integration Points
7. Security Architecture

Requirements Analysis

기능적 요구사항 (Functional)

- 의료 이미지 분석 및 진단 지원
- 실시간 예측 결과 제공
- 사용자 친화적 인터페이스
- 데이터 관리 및 저장 기능
- 성능 모니터링 대시보드

비기능적 요구사항 (Non-Functional)

- 응답 시간 < 5초
- 99.9% 시스템 가용성
- HIPAA 규정 준수
- 확장 가능한 아키텍처
- 데이터 암호화 및 보안

요구사항 매트릭스

정확도 (Accuracy > 95%)	HIGH
응답 속도 (< 5s)	HIGH
데이터 보안	HIGH

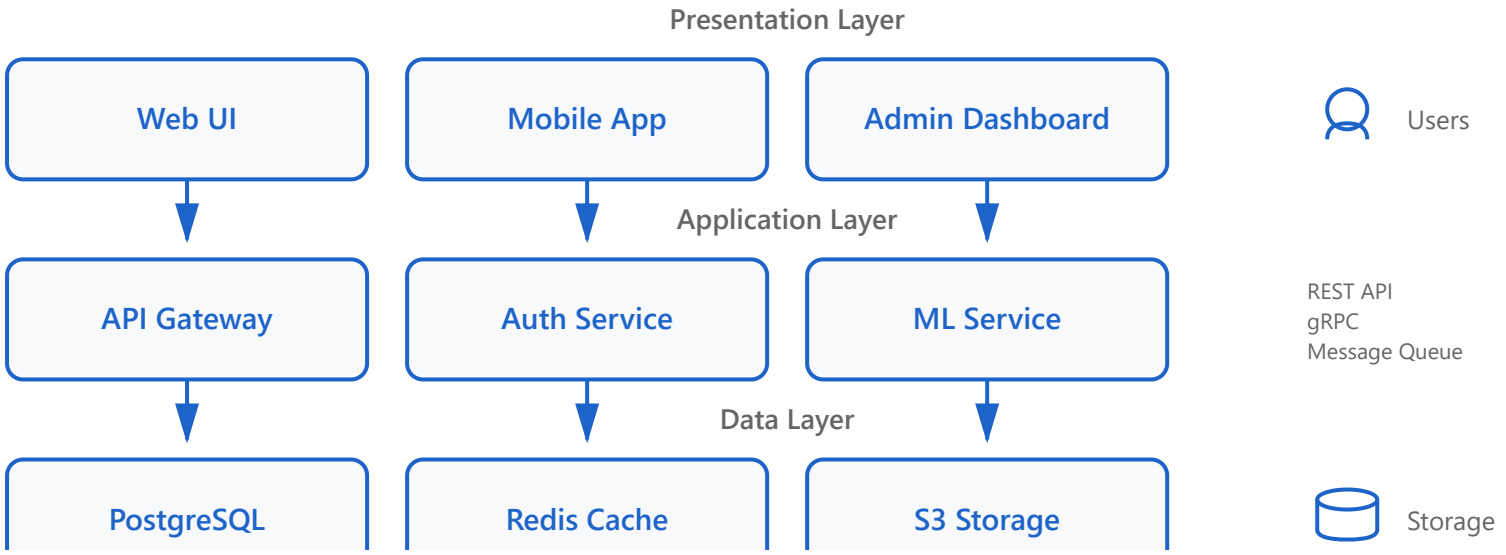
우선순위 설정

- 필수 (Must Have): 핵심 진단 기능
- 중요 (Should Have): 모니터링 시스템
- 선택 (Nice to Have): 고급 시각화

사용자 경험 MEDIUM

- 미래 (Future): AI 모델 자동 업데이트

Architecture Planning - System Components



핵심 설계 원칙

- 마이크로서비스 아키텍처
- 확장 가능한 구조
- 느슨한 결합 (Loose Coupling)

통신 프로토콜

- REST API for synchronous
- Message Queue for async
- gRPC for ML inference

Technology Stack Selection

Backend

- Python (FastAPI)
- Node.js (Express)
- PostgreSQL Database
- Redis Cache

Frontend

- React.js
- TypeScript
- TailwindCSS
- D3.js for visualization

ML/AI

- PyTorch / TensorFlow
- ONNX Runtime
- MLflow for tracking
- Weights & Biases

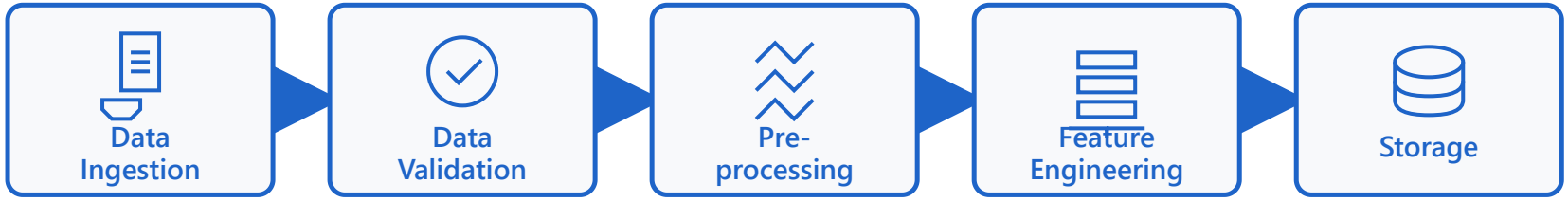
Infrastructure

- Docker / Kubernetes
- AWS / GCP
- CI/CD (GitHub Actions)
- Terraform

선택 기준 (Selection Criteria)

커뮤니티 지원	성능 & 확장성	보안 & 규정 준수
개발 생산성	비용 효율성	유지보수 용이성

Data Pipeline Design



ETL 프로세스

- Extract: 다양한 소스에서 데이터 수집
- Transform: 정제 및 변환
- Load: 데이터베이스에 저장

주요 고려사항

- 데이터 품질 검증
- 확장 가능한 아키텍처
- 실시간 처리 지원

Model Selection Criteria

모델 비교

- 정확도 vs 속도 트레이드오프
- 모델 크기 및 메모리 사용량
- 추론 시간 및 처리량

선택 기준

- Task에 맞는 모델 아키텍처
- 사전 학습 모델 활용
- Fine-tuning 전략

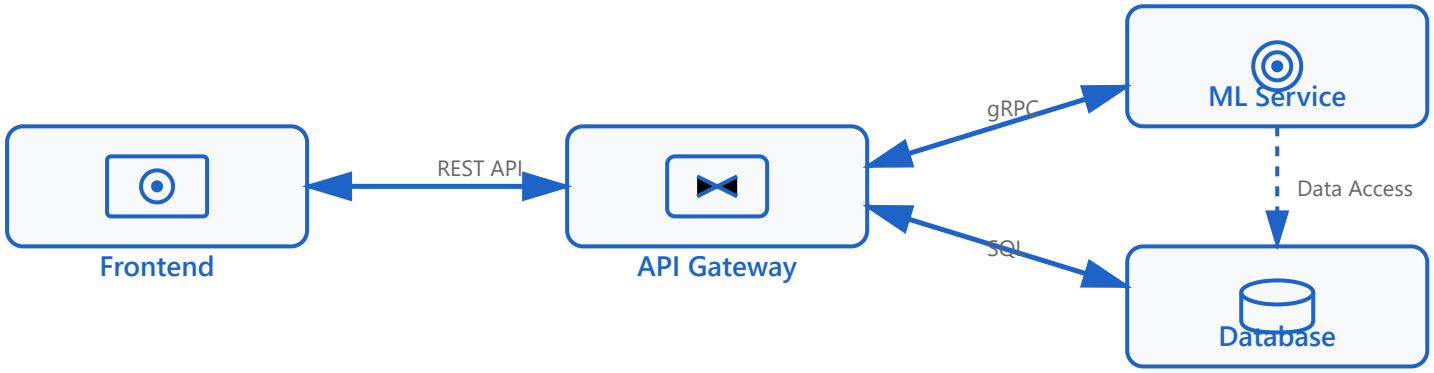
성능 메트릭

- Accuracy, Precision, Recall
- F1 Score, AUC-ROC
- 추론 시간 및 FLOPs

실용적 고려사항

- 배포 환경 제약
- 유지보수 및 업데이트
- 비용 효율성

Integration Points



시스템 통합 맵

- Frontend ↔ API Gateway
- API Gateway ↔ ML Service
- ML Service ↔ Database

API 설계

- RESTful API 엔드포인트
- Request/Response 형식
- 에러 핸들링 전략

인터페이스 정의

- 데이터 교환 포맷 (JSON)
- 인증 및 권한 관리

통합 테스트

- End-to-End 테스트
- 성능 테스트

- 버전 관리 전략

- 부하 테스트

Security Architecture

보안 레이어

- 네트워크 보안 (방화벽, VPN)
- 애플리케이션 보안 (인증/인가)
- 데이터 보안 (암호화)

위협 모델

- SQL Injection 방지
- XSS/CSRF 공격 차단
- 데이터 유출 방지

접근 제어

- 역할 기반 접근 제어 (RBAC)
- 다중 인증 (MFA)

규정 준수

- HIPAA 준수
- GDPR 준수

- 감사 로깅

- 데이터 보호 정책

Part 2/3:

System Implementation

1. Data Collection & Processing
2. Model Training Pipeline
3. Evaluation Framework
4. API Development
5. Frontend Interface
6. Testing Strategies

Data Collection & Processing

데이터 수집

- 의료 이미지 데이터셋 확보
- 메타데이터 수집 및 라벨링
- 데이터 품질 검증

데이터 처리

- 이미지 정규화 및 리사이징
- 데이터 증강 (Augmentation)
- 배치 처리 최적화

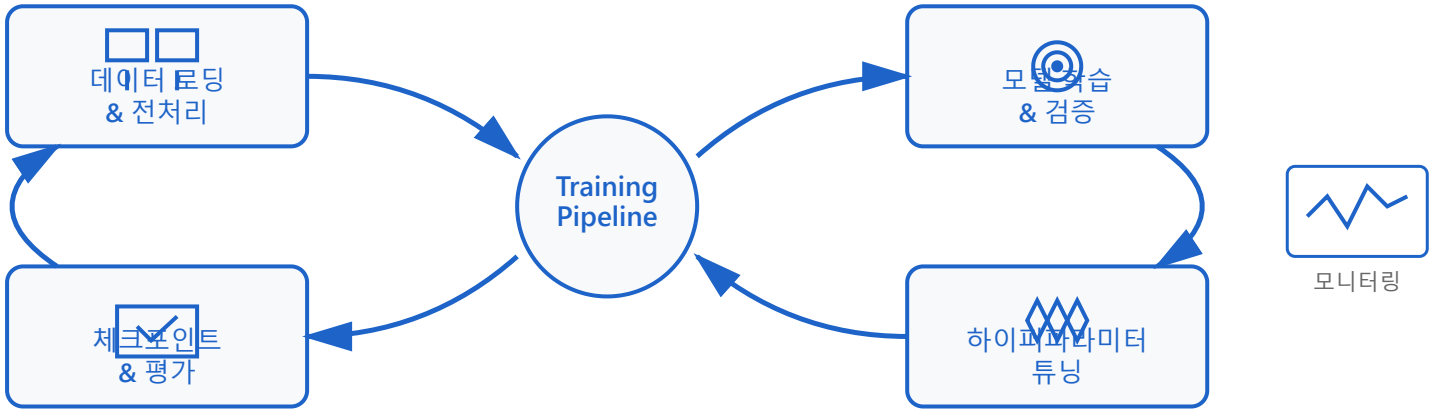
데이터 플로우

- 실시간 데이터 파이프라인
- 배치 처리 스케줄링
- 데이터 버전 관리

구현 도구

- Pandas, NumPy
- OpenCV, PIL
- Apache Airflow

Model Training Pipeline



학습 프로세스

- 데이터 로딩 및 전처리
- 모델 학습 및 검증
- 하이퍼파라미터 튜닝

파이프라인 구성

- 자동화된 학습 워크플로우
- 체크포인트 저장
- Early Stopping

모니터링

- 학습 손실 추적

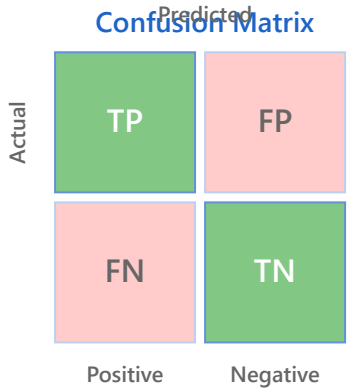
도구 활용

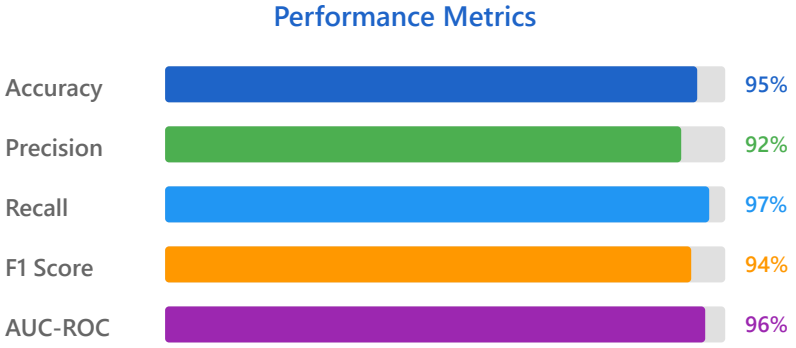
- PyTorch Lightning

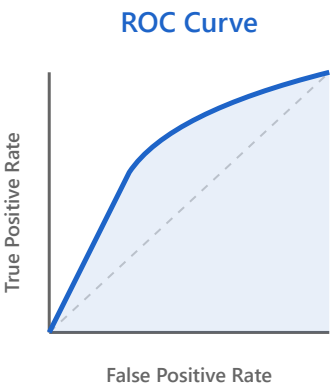
- 검증 메트릭 모니터링
- 리소스 사용량 확인

- MLflow
- Weights & Biases

Evaluation Framework







평가 메트릭

- Accuracy, Precision, Recall
- F1 Score, ROC-AUC
- Confusion Matrix

테스트 전략

- Hold-out Test Set
- K-Fold Cross Validation
- Stratified Sampling

성능 분석

- Error Analysis
- Feature Importance

검증 프로세스

- 독립적인 테스트 데이터
- 실제 환경 시뮬레이션

- Model Interpretation

- 통계적 유의성 검증

API Development

REST API

- GET /predict - 예측 요청
- POST /upload - 데이터 업로드
- GET /status - 상태 확인

GraphQL

- 유연한 쿼리 인터페이스
- 타입 시스템
- 실시간 구독

API 문서

- OpenAPI/Swagger
- 자동 문서 생성
- API 테스트 도구

보안

- JWT 인증
- Rate Limiting
- HTTPS 암호화

Frontend Interface

UI 디자인

- 직관적인 사용자 인터페이스
- 반응형 디자인
- 접근성 고려

핵심 기능

- 이미지 업로드
- 실시간 예측 결과
- 결과 시각화

사용자 플로우

- 로그인 및 인증
- 데이터 업로드
- 결과 확인 및 다운로드

기술 스택

- React.js
- TypeScript
- TailwindCSS

Testing Strategies

단위 테스트

- 함수 레벨 테스트
- 모듈별 독립 테스트
- Pytest, Jest 활용

통합 테스트

- 컴포넌트 간 상호작용
- API 엔드포인트 테스트
- 데이터베이스 통합

E2E 테스트

- 사용자 시나리오 테스트
- 전체 워크플로우 검증
- Selenium, Cypress

테스트 커버리지

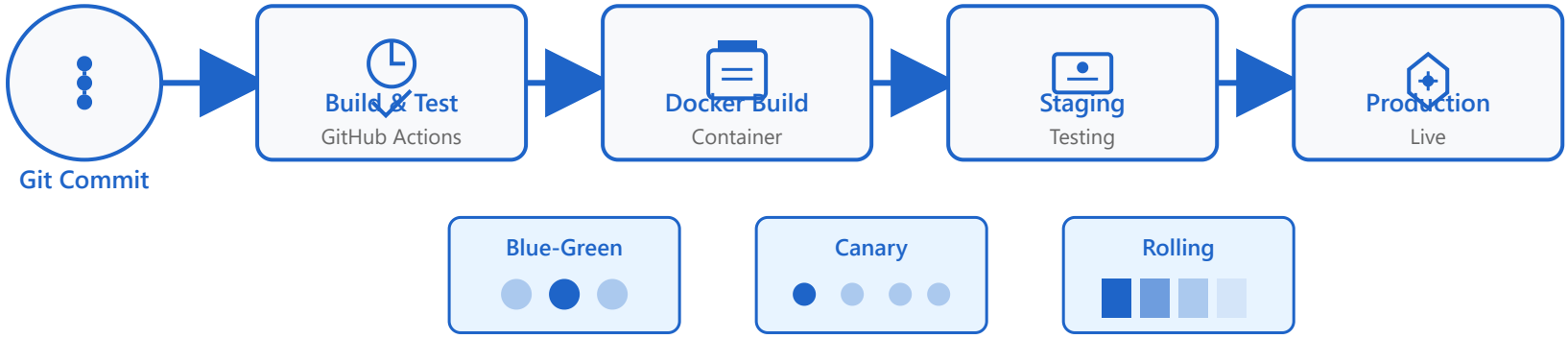
- 코드 커버리지 > 80%
- Critical Path 100% 커버
- 자동화된 테스트 실행

Part 3/3:

Deployment and Validation

1. Deployment Pipeline
2. Performance Monitoring
3. Clinical Validation Study
4. User Acceptance Testing
5. Documentation & Training
6. Maintenance Planning

Deployment Pipeline



CI/CD 구성

- GitHub Actions 워크플로우
- 자동 빌드 및 테스트
- Docker 이미지 생성

배포 플로우

- Development → Staging → Production
- Blue-Green Deployment
- Canary Release

인프라 관리

- Infrastructure as Code (Terraform)

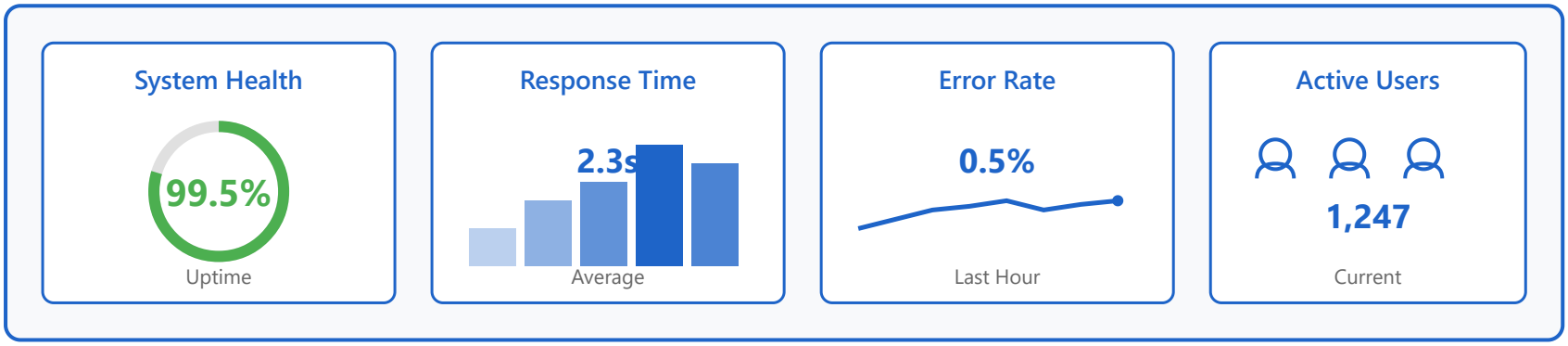
배포 자동화

- 자동 롤백 메커니즘

- Kubernetes 오케스트레이션
- Auto Scaling 설정

- Health Check 모니터링
- 배포 알림 시스템

Performance Monitoring



모니터링 대시보드

- 실시간 시스템 상태
- API 응답 시간
- 에러율 추적

성능 메트릭

- 처리량 (Throughput)
- 지연 시간 (Latency)
- 리소스 사용량

알림 시스템

- 임계값 기반 알람
- 이상 탐지

모니터링 도구

- Prometheus
- Grafana

- Slack/Email 통합

- ELK Stack

Clinical Validation Study

연구 프로토콜

- 연구 설계 및 목적
- 포함/제외 기준
- 샘플 크기 계산

검증 메트릭

- 진단 정확도
- 민감도와 특이도
- 양성/음성 예측도

임상 평가

- 전문의 비교 연구
- 실제 임상 환경 테스트
- 환자 안전성 평가

규제 준수

- IRB 승인
- FDA/CE 인증 준비
- 임상 시험 문서화

User Acceptance Testing (UAT)

UAT 계획

- 테스트 시나리오 작성
- 실제 사용자 참여
- 테스트 환경 구축

피드백 수집

- 사용성 평가
- 기능 완성도 검증
- 개선 사항 도출

테스트 케이스

- 일반 사용자 시나리오
- 전문가 워크플로우
- 예외 상황 처리

결과 분석

- 만족도 설문조사
- 문제점 우선순위화
- 개선 계획 수립

Documentation & Training

기술 문서

- 시스템 아키텍처 문서
- API 레퍼런스
- 설치 및 배포 가이드

사용자 매뉴얼

- 사용자 가이드
- FAQ 및 트러블슈팅
- 비디오 튜토리얼

교육 자료

- 온보딩 프로그램
- 실습 워크샵
- 온라인 교육 콘텐츠

문서 관리

- 버전 관리
- 정기적 업데이트
- 다국어 지원

Maintenance Planning

유지보수 일정

- 정기 업데이트 스케줄
- 보안 패치 적용
- 성능 최적화

모니터링 및 지원

- 24/7 시스템 모니터링
- 사용자 지원 체계
- 이슈 트래킹

업데이트 프로세스

- 새로운 기능 추가
- 버그 수정
- 모델 재학습 및 배포

장기 계획

- 기술 부채 관리
- 시스템 개선 로드맵
- 확장성 계획

Project Presentations

발표 구조

- 프로젝트 개요 및 목표
- 시스템 아키텍처 설명
- 주요 기능 데모
- 결과 및 성과

발표 팁

- 명확하고 간결한 전달
- 시각 자료 효과적 활용
- 청중과의 상호작용
- 질의응답 준비

평가 기준

- 기술적 완성도
- 프레젠테이션 품질
- 창의성과 혁신
- 팀워크와 협업

시간 배정

- 발표 15분
- 데모 5분
- 질의응답 5분
- 총 25분

Peer Review Session

평가 루브릭

- 기능 완성도 (25%)
- 코드 품질 (25%)
- 문서화 (20%)
- 창의성 (30%)

피드백 형식

- 건설적인 비평
- 구체적인 개선 제안
- 긍정적인 측면 강조
- 학습 포인트 공유

동료 평가

- 각 팀 상호 평가
- 피드백 문서 작성
- 토론 및 질문
- 베스트 프랙티스 공유

학습 효과

- 다른 접근법 이해
- 문제 해결 아이디어 획득
- 협업 능력 향상
- 비판적 사고 개발

Lessons Learned

기술적 교훈

- 효과적인 아키텍처 패턴
- 최적화 기법
- 피해야 할 함정
- 도구 선택의 중요성

프로젝트 관리

- 시간 관리의 중요성
- 리스크 대응 전략
- 팀 커뮤니케이션
- 일정 조정 경험

개선 권장사항

- 더 나은 설계 방법
- 테스트 전략 개선
- 문서화 프로세스
- 배포 자동화

개인 성장

- 새로운 기술 습득
- 문제 해결 능력
- 협업 능력
- 리더십 개발

Career Opportunities in Medical AI

주요 직무

- ML Engineer
- Data Scientist
- Research Scientist
- Clinical AI Specialist

필요 역량

- Machine Learning 전문성
- 의료 도메인 지식
- 소프트웨어 엔지니어링
- 커뮤니케이션 능력

커리어 경로

- Junior → Mid → Senior Engineer
- Specialist → Lead → Principal
- Research → Product → Management
- Startup Founder

산업 트렌드

- 의료 AI 시장 성장
- 규제 환경 변화
- 원격 의료 확대
- 맞춤형 의료

Course Reflection

학습 여정

- 기초 개념 이해
- 실전 프로젝트 경험
- 팀 협업 경험
- 전문가 네트워킹

주요 성과

- End-to-End 시스템 구축
- 기술 스택 마스터
- 문제 해결 능력
- 포트폴리오 구축

도전과 극복

- 기술적 어려움
- 시간 관리
- 팀 조율
- 성공적 해결

미래 계획

- 지속적인 학습
- 오픈소스 기여
- 네트워킹 확대
- 커리어 발전

Congratulations!

Introduction to Biomedical Datascience 과정 수료

여러분은 의료 AI 시스템 구축의 전 과정을 성공적으로 완료하였습니다.

이제 여러분은 의료 데이터 과학 분야의 전문가로서 첫 걸음을 내디뎠습니다.

🏆 수료증이 발급되었습니다

🌟 포트폴리오 프로젝트 완성

Alumni Network

Career Support

Continuous Learning