Developer: 어 양록(Yangrok Auh)

시스템 오퍼레이션 자동화

(신규 자동화 개발 프로젝트)

프로젝트.

기간: 2025.07.01 ~ 2025.12.26

현장: PayPay-Card(페이페이 카드)

소속: 시스오페 자동화 팀(5명/서브 리더)

개요.

- 시스템 오퍼레이션 부서의 정형화된 수작업 업무를 자동화
- 계약, 매상, 채권 3개 팀에 대한 입력 파일을 확인하여 데이터 조회 및 검사, 결과 보고서 생성, 엑셀 작성 등의 수작업 프로세스를 Spring Boot 기반 배치 프로그램으로 자동화
- 운영 효율을 극대화하고, 인력 및 운영 비용 등 코스트 절감을 기대

기술 스택 및 툴.

Java / Spring Boot / Junit5 / Oracle / My Btis

S3 / SQS / KEDA / EKS

GitHub / IntelliJ / SQL Developer / Copilot / Confluence

담당 업무.

- 덤프 리스트 추출 자동화(매상 팀)

  덤프 된 매출 데이터가 덤프 대상이 맞는지 검증 후, 매출 명세를 조회 하고 csv파일로 생성해서 해당 부서로 연계.

  - 기존 업무 프로세스 파악 (자동화전 작업)
  - 사양서 작성: 자동화 사양서(가능 범위와 여부), 파일 사양서(입출력 파일)
  - 개발: 파일 입출력, 검증 로직, 데이터 조회 등 비즈니스 로직 구현
  - 기능 테스트: Junit5로 작성 / JaCoCo 사용으로 코드 안전성 높임
  - 통합 테스트: 테스트 사양서 작성, 테스트 데이터 준비
  - 필드 테스트: 실제 운영 환경과 동일한 환경에서 테스트
  - 배포

- 채권 유동화 데이터 패치(채권 팀)

  입력 파일에 특정 항목의 패턴을 체크, 이상 패턴의 경우, 데이터 베이스를 조회 하여

Developer: 어 양록(Yangrok Auh)

검증을 하고, 이상 데이터가 확인이 되면 정상 데이터로 패치 처리 실시 및 csv파일로 출력.

- 기존 업무 프로세스 파악 (자동화전 작업)
- 사양서 작성: 자동화 사양서(가능 범위와 여부), 파일 사양서(입출력 파일)
- 개발: 파일 입력, 청크 처리 및 멀티파트 업로드, 데이터 조회 자동화 구현
  2천만개가 넘는 입력 데이터를 프로그램상에서 일괄 처리하기 위해 청크 처리 로직 구현
  청크 처리한 데이터들을 하나의 결과 파일로 업로드하기 위해 멀티파트 업로드 도입
- 기능 테스트: Junit5로 작성 / JaCoCo 사용으로 코드 안전성 높임
- 통합 테스트: 테스트 사양서 작성, 테스트 데이터 준비
- 필드 테스트: 실제 운영 환경과 동일한 환경에서 테스트
- 배포

- 미수 금액 1엔 발생 체크(채권 팀)

  자동이체 취소가 발생한 고객의 당월 미수금액과 계약 정보를 추출하여 csv파일로 출력.

  - 기존 업무 프로세스 파악 (자동화전 작업)
  - 사양서 작성: 자동화 사양서(가능 범위와 여부), 파일 사양서(입출력 파일)
  - 개발: 파일 입출력, 데이터 조회, 내부 비즈니스 로직 자동화 구현
  - 기능 테스트: Junit5로 작성 / JaCoCo 사용으로 코드 안전성 높임
  - 통합 테스트: 테스트 사양서 작성, 테스트 데이터 준비
  - 필드 테스트: 실제 운영 환경과 동일한 환경에서 테스트
  - 배포

성과.

- 10명이 담당하던 프로세스를 자동화 로직 도입으로 2명 체제 운영으로 전환
- 업무 효율화율 약 80% 상승, 운영 코스트 대폭 절감
- 데이터 처리 자동화로 검증 시간 90% 이상 단축, 오류 발생 0% 유지
- DDD(Domain-Driven Design) 개발로 비즈니스 로직을 명확히 표현하여 가독성 및 추후 확장성 높임
- 팩토리 메서드 적용과 단일책임 원칙 준수로 객체 생성을 일관화하고, 캡슐화를 강화하여 코드 품질 및 확장성·재사용성·유지보수성을 향상

개발 아키텍처

[Presentation Layer]
- exception(자동화 배치 에러 정의)
- runner(자동화 별 실행 클래스/ 서비스를 호출)

[Application Layer]
- exception(서비스 에러 정의)
- service(도메인을 호출 및 조합)
  - chunk
  - download
  - contract
  - ⋮

[Domain Layer]
- model(도메인 클래스/ 비즈니스 로직 / dto)
  - card
  - contract
  - csv
  - ⋮
- repository(DB, S3처리 로직 인터페이스)
  - contract
  - csv
  - sales
  - ⋮
- type(중간 산출물 클래스)

[Infrastructure Layer]
- database
  - entity (my batis)
  - impl (도메인 레포지토리 구현체)
  - mapper (my batis)
- s3
  - config (s3 클라이언트)
  - exception (s3처리 에러 정의)
  - impl (도메인 레포지토리 구현체)
  - shared (s3 핸들링 공통 로직)

[resource]
- my batis(쿼리xml)
- app.properties
- log.xml

자동화 시스템 환경