물류 SW 개발자
김유석 입니다.

2024 PORTFOLIO

물류 관리자에서 개발자로
SCM 개발자
김유석입니다.

김유석 / Yuseok Kim

최소한의 공간으로 최대한의 효율을 낼 수 있는 프로그램을 고민하고 있습니다.

저의 고민된 코드 한 줄이 수백, 수 천명의 근로자들의 한 걸음을 줄인다고 생각하며 개발에 임하고 있습니다.

각 업장에 적합한 피킹 , 적치 시스템을 프로그램으로 구현하고자 합니다.

Career

롯데 글로벌 로지스

🗓️ 2021.04 ~ 2023.07(2년 3개월)

👷 유통물류 3팀 유니클로

업무 내용

- WMS 입출고 데이터 관리
- 21FW, 22SS, 22FW 반품 스케줄 수립
- DAS 분류, 작업 관리

수행 프로젝트

👷 한·일 무역 전쟁 TFT팀 팀장

- 6,000평 버퍼 센터 운영
- 간선 배차 관리
- 적치 시스템 변경으로 3,000평 감평 (임대료 감축)
- 도급사 인원 관리, 고객 사 관리

Skills

BackEnd

- JAVA
- Spring Boot
- JPA
- JUnit

FrontEnd

- Vue
- React
- Next.js
- JavaScript
- TypeScript

Database

- MySQL
- Oracle
- MyBatis
- MongoDB
- Redis

Certification

2024.12 정보처리기사(예정)

2024.09 SQLD

2023.07 물류관리사

2021.03 컴퓨터 활용

2020.08 2종 보통 운전 면허

2016.09 국제 무역사

Experience

삼성 청년 SW 아카데미 이수

2024.01 ~ 2024.06 (6개월)

교육 내용

컴퓨터 사고력 및 SW문제 해결 능력 강화

Java 언어 활용 및 문법 이해

Spring, Vue를 활용한 웹 개발 기술

DB 설계, RDBMS 활용

수행 프로젝트

축구 동호회 매칭 시스템 “SSACCER”

(SpringBoot, Vue3, MyBatis, Mysql)

삼성 청년 SW 아카데미 수료

2024.06 ~ 2024.12 (6개월)

교육 내용

6인 1팀 자기주도 프로젝트 수행

공통 프로젝트 : 모바일 웹 디자인 및 기본 구성 (7주)

특화 프로젝트 : 인공지능 언어 모델 구현 (8주)

자율 프로젝트 : 자유 주제 프로젝트 구현 (5주)

수행 프로젝트

공통 : 재고 관리 시스템(WMS) “FITBOX”

특화 : 무인 매장 관리 시스템 “AutoStore”

자율 : CSV 파일 AI 모델 분석 시스템 “말하는 DA로 "

프로젝트 상세:

창고 관리 시스템 (WMS)

최소한의 공간으로 최대 생산성

창고 관리 시스템 (WMS)

👦 역할

Back-End | 팀장

👫 역할

Back-End 4 | Front-End 2

🏆 성과

삼성 전자 우수 프로젝트 수상

📎 링크

GitHub

ERD 설계

⚙️ 개발 환경

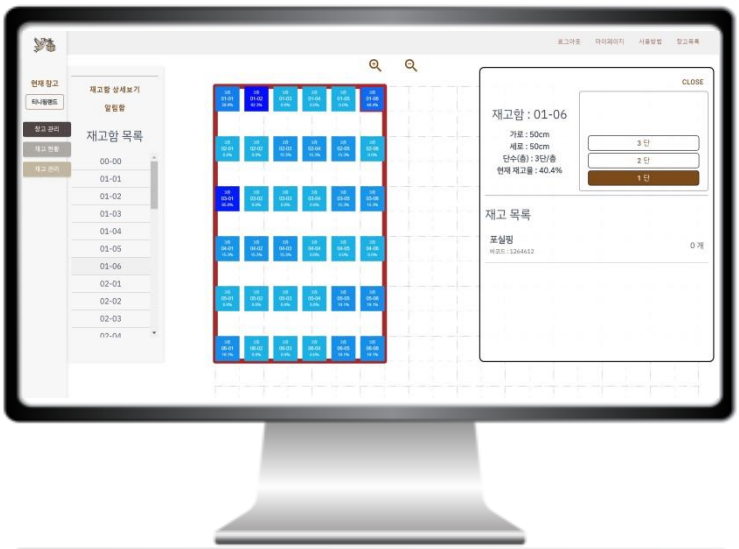
Front-End : JavaScript | Next.js | Konva | React-Chart

Back-End : SpringBoot (3.3.1) | JPA | MySQL

Infra : AWS EC2 | Jenkins | Nginx Blue & Green

프로젝트 개요

📅 2024.07.08 ~ 2024.8.16 (7주)

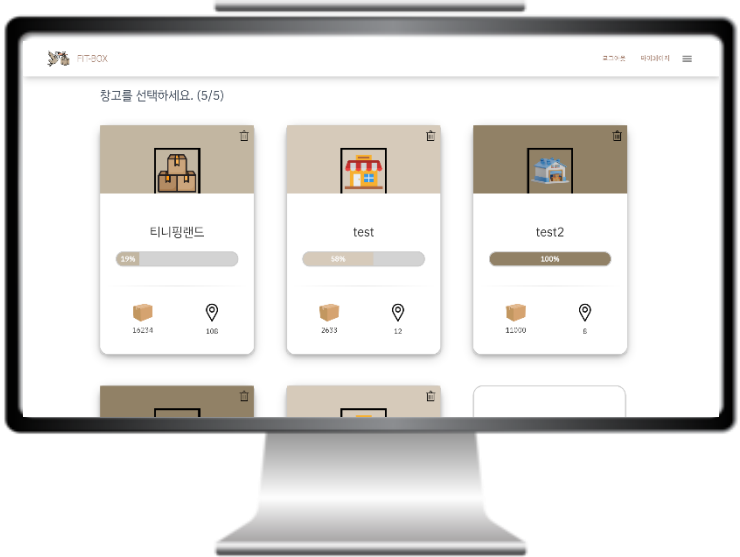


최단 거리 피킹 시스템 구현

- 재고 소진 시 2~3 단 재고 보충 시스템
- 2~3 단 재고 소진 시 발주 알림 시스템

창고 사용량 추적 관리 시스템

- 색상으로 로케이션 사용률 표시
- 압축 시스템으로 상시 최소 공간 활용



Excel 상품 등록, 추가, 출고

- Excel 파일로 대량 상품 등록, 출고 시스템

다중 창고 사용 기능

🖥️ 담당 업무

창고 CRUD , Rest API 개발
User CRUD, 결제 API
OAuth2 소셜 로그인 API
Spring Security
JWT 토큰 관리

🚚 물류 서비스 시사점 및 생각

- 창고의 사용량을 화면에 표시하여 비생산적인 공간 추적 및 관리
- RDBMS를 활용 물류 데이터 처리시 초기에 Indexing 반정규화 고려 (대규모 데이터 처리)

❓ 문제 상황

상품 조회 및 출고 시 5번의 Join이 발생하여 속도 개선이 필요

Ex) When 1개 사업체 -> 3개 창고 -> 1,000로케이션-> 4,000단 -> 4,000개 상품

쿼리 성능 2~3분 **O(n * m)**

n = 로케이션 수 | m = 상품 수

⁉️ 해결 방안

수정 삭제가 적은 창고 테이블에 Indexing 적용

```
@Entity  김유석
@Getter
@Setter
@SQLRestriction("status_enum = 'Active'")
@Table(name = "warehouse", indexes = {
    @Index(name = "idx_business_id", columnList = "business_id")
})
```

수정 삭제가 많은 상품을 상위 테이블에 반정규화

```
/**
 * @param warehouseID
 * @return
 * 창고 상품을 조회하는 쿼리 (반정규화된 필드를 사용)
 */
@Query("SELECT p FROM Product p " +  2 usages
    "JOIN p.floor f " +
    "WHERE f.warehouseId = :warehouseID")
List<Product> findByWarehouseId
(@Param("warehouseID") Long warehouseID);
```

‼️ 문제 해결

Ex) When 1개 사업체 -> 3개 창고 -> 1,000로케이션-> 4,000단 -> 4,000개 상품

쿼리 성능 40초~1분 30초 **O(n)**

n = 로케이션 수

프로젝트 상세:
무인 매장 관리 시스템 (AutoStore)

CCTV 매장 이상현상 감지

무인 매장 관리 시스템 (AutoStore)

역할

Front-End (Electron.exe) | 팀장

역할

Back-End 3 | Front-End 2 | AI 영상 분석 1

개발 기간

2024.09.19 ~ 2024.10.10 (8주)

링크

GitHub

ERD 설계

개발 환경

Front-End(PWA) : JavaScript | Next.js | Konva | React-Chart

Front-End(Electron) : Type Script | Next.js | Electron

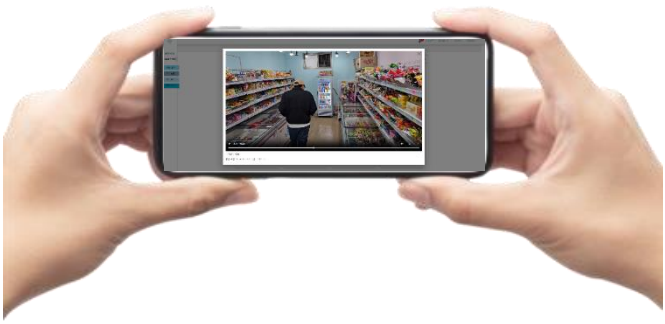
Back-End(Spring) : Spring Boot (3.3.1) | JPA | MySQL | MariaDB | Radis

Back-End(Fast API) : Fast API | python(3.12.5)

Back-End(TensorFlow) : TensorFlow

Infra : AWS EC2 | Jenkins | Nginx Blue & Green

프로젝트 개요

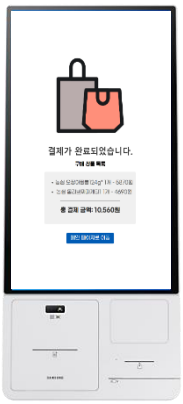


키오스크 자동 상품 인식

- RFID/NFC 칩을 활용한 상품 자동 인식
- 키오스크 exe 파일 배포 환경

수요 예측 서비스

- 수요 예측, 데이터 분석, 추천 서비스
(LSTM 모델, Random Forest 모델)
- 매출 확인, 매출 예측, 인기 상품, 고객 선호도



CCTV 이상감지

- 도난, 파손, 실신, 방화, 흡연
- 두 가지 비디오 인식 모델 사용 앙상블 방식인식 (X3D 모델, Slow Fast 모델)

🖥️ 담당 업무

RFID / NFC 칩 자동 인식(PC/SC 라이브러리)
키오스크 CPU 환경 배포
키오스크 화면 구성(Next.js)
TOSS 결제 API
키오스크 관리 OTP 생성

🚚 물류 서비스 시사점 및 생각

- 물류장비는 장비에 종속적이라서 제조사의 매뉴얼을 잘 읽어볼 것
- 현장 PC를 활용하여 프로그램을 작동한다는 점에서 exe 파일 배포가 필수 적

❓ 문제 상황

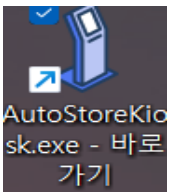
1. 키오스크를 운영하기 위해 불필요한 서버 비용 발생
2. 인코딩한 NFC 칩을 리더기가 인식 못하는 상황
3. 키오스크 인증 방식 및 보안

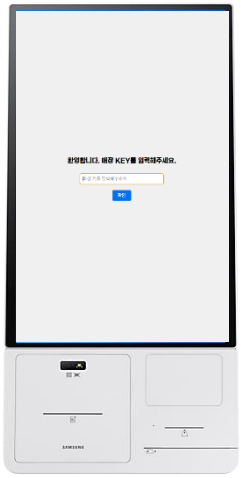
⁉️ 해결 방안

1. 개별 CPU에서 사용되는 키오스크를 고려하여 exe 파일 배포 방식 채택
2. NFC 칩에 저장되어 있는 구역을 순회하여 인코딩 값 추출
3. Redis TTL을 도입, 유효시간 5분을 지정하여 OTP 발행 후 삭제

‼️ 문제 해결

1. 개별 CPU에서 인터넷 환경만 구축되면 실행되는 EXE 파일 배포





2. NFC 칩 4번째 구역에 HEX코드 값 추출 후
ASCII 코드로 변환하여 상품 조회

```
function readSpecificBlock(reader, protocol, blockNumber) {
  const command = Buffer.from([0xff, 0xb0, 0x00, blockNumber, 0x10]);
let fullData = data.toString("hex"); // hex 데이터를 문자열로 변환
console.log(`Block ${blockNumber} data:`, fullData);
let barcode = Buffer.from(fullData, "hex").toString("ascii").trim();
```

3. 사장님 계정으로 OTP 발급 후 5분 뒤 삭제

```
static final int OTP_TTL = 5 * 60;
 ops.set(otpString, deviceIdString, OTP_TTL, TimeUnit.SECONDS);
```

프로젝트 상세:
파일 모델 분석 시스템 (원하는 Da로)

누구나 손쉽게 모델 분석

파일 모델 분석 시스템 (원하는 Da이터로)

역할

Back-End | 팀장

역할

Back-End 4 | Front-End 2 | Infra 1

개발 기간

2024.10.14 ~ 2024.11.19 (5주)

링크

GitHub

ERD 설계

개발 환경

Front-End: JavaScript | Next.js

Back-End(Spring) : Spring Boot (3.3.1) | JPA | MySQL | MongoDB

Back-End(Fast API) : Fast API | python(3.12.5)

Infra : AWS EC2 | Jenkins | Nginx Blue & Green | S3

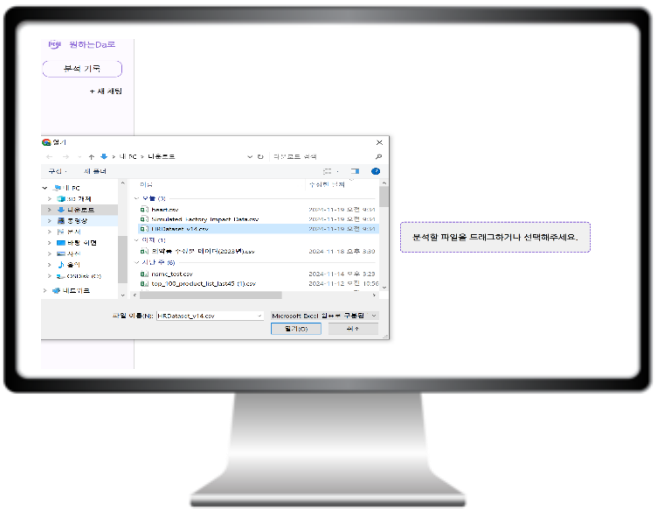
프로젝트 개요

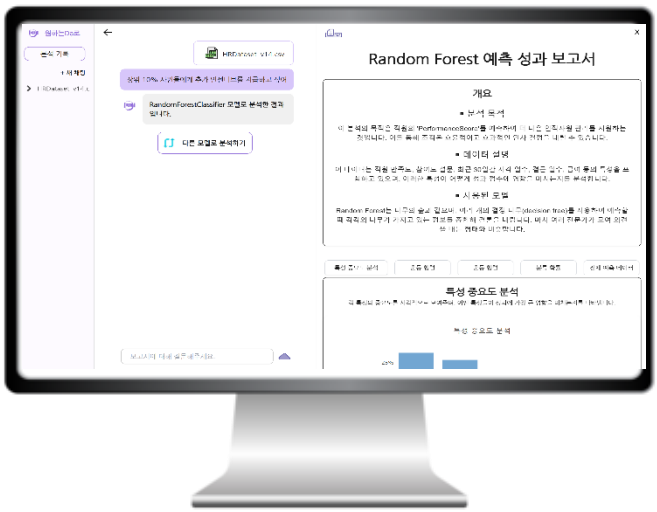
CSV 파일 LLM을 활용하여 전 처리

- CSV 파일 파싱 및 전송
- ChatGPT API를 활용하여 CSV 파일 분석

요구사항 10가지 모델 분석 서비스

- Random forest regression
- Random forest classification
- Logistic regression binary
- Logistic regression multinomial
- Kmeans clustering segmentation
- Kmeans clustering anomaly
- Neural network regression
- Graph neural network analysis
- Support vector machine(classification)
- Support vector machine(regression)





🖥️ 담당 업무

Chat GPT API 전송
요구사항 분석 및 NoSQL 설계
비정형 데이터 분류 및 적재(MongoDB)
Fast API 서버 요구사항 전송
S3 서버 CSV 파일 저장

🚚 물류 서비스 시사점

- LLM을 활용해 출고, 입고 데이터 검증 가능성 (대규모 토큰 비용 문제가 여전히 존재)
- NoSQL 활용하여 대규모 물류 데이터를 처리 시에 속도 처리에 이점, 확장에 용이

❓ 문제 상황

1. GPT Hallucination 응답 후 Fast API 서버 전송 시 에러 문제
2. User가 선택하지 않은 모델을 추적하여 다시 선택

⁉️ 해결 방안

1. Fast API에서 요구하는 값이 비어 있을 경우 Null을 Json 삽입
2. 1)LLM의 결과 값에 isSelected = false 값 삽입,
   2) 이후 선택 시 마다 해당 인덱스 true로 변경
   3) 다른 모델 선택시에 false 값만 반환 및 새로운 ReqeustID 채번

‼️ 문제 해결

1. 필수 값이 Hallucination으로 반환되지 않더라도 Fast API 서버에서 결과 처리 가능

```
modelRecommendations.forEach(model -> {
    model.put("isSelected", false);
    model.putIfAbsent("target_variable", null);
    model.putIfAbsent("id_column", null);
});
```

2. 고객이 선택한 값을 추적하여 선택되지 않은 모델도 다시 선택할 수 있게 됨