SW개발/HW제작 설계서

프로젝트 명 :인공지능 기반의 객체 검지 기술을 응용한 스마트 물류 창고

2024. 08. 30

(팀명) A4B1

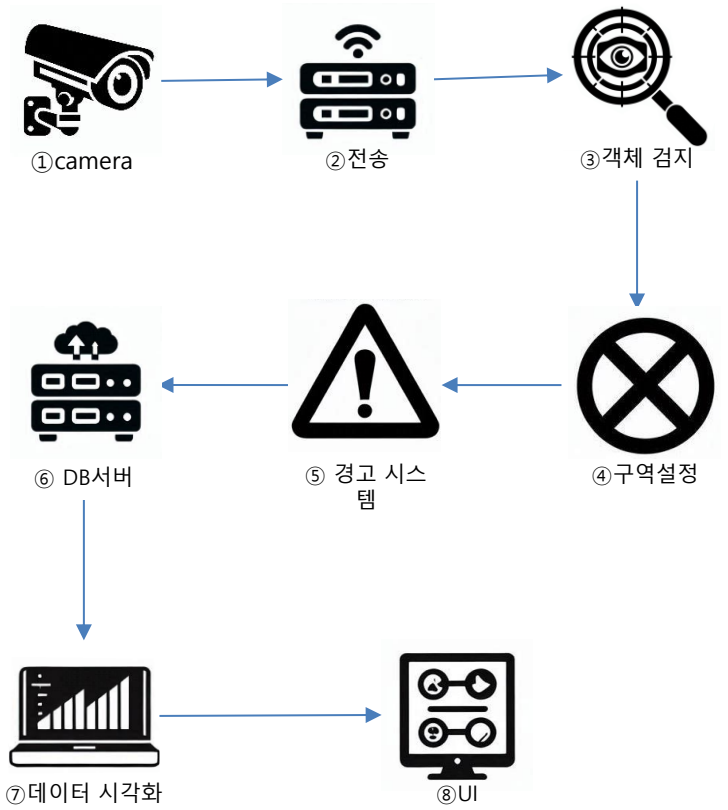
| 요구사항 정의서

유형	요구사항 ID	요구사항명	기능 ID	기능명	세부사항
기능적 요구사항	A01	실시간 객체 검지	A01-01	다양한 객체 검지	사람, 지게차 등 다양한 객체들을 실시간으로 검지
			A01-02	객체 검지 결과 표출	검지된 객체들의 위치와 상태를 실시간으로 표출
	A02	보안 및 안전 관리	A02-01	안전, 위험 구역 설정	안전 구역과 위험 구역 설정 기능
			A02-02	위험구역 실시간 경고	작업자의 위험 구역 접근 시 실시간 경고 시스템
			A02-03	안전거리 유지	작업 차량과 작업자 간의 안전거리 유지 경고 시스템
			A02-04	객체 경고 및 알림 시스템	특정 객체(예:지게차)에 대한 경고 및 알림 시스템

| 요구사항 정의서

유형	요구사항 ID	요구사항명	기능 ID	기능명	세부사항
기능적 요구사항	A03	데시보드 및 모니터링 시스템	A03-01	객체 검지 결과 표시	실시간 객체 검지 결과를 시각적으로 표시
			A03-02	경고 및 알림 현황 표출	실시간 경고 및 알림 현황 표출
			A03-03	웹사이트	사용자 친화적인 인터페이스를 갖춘 웹사이트
비기능적 요구사항	B01	성능	B01-01	데이터 처리	원활한 실사간 데이터 처리 능력
			B01-02	객체 검지	신뢰할 수 있는 정확도의 객체 검지 성능
	B02	확장성	B02-01	확장 가능성	향후 시스템 확장 가능성 고려
	B03	보안	B03-01	사용자 인증	사용자 인증 및 권한 관리 기능

| 서비스 구성도 - 서비스 시나리오



객체 검지 시스템

① 특정 구역(거리 등) 내 설치된 카메라로 촬영한다.

② 카메라에서 수집한 영상을 전송한다.

③ 실시간 영상에서 객체를 검지한다.

④ 구역을 설정한다.

⑤침입자 발생 등 이벤트 발생시 경보를 발생한다.

⑥ 경보 발생 내역을 DB에 저장한다.
⑦ 실시간 객체 검지 결과 및 알림 현황을 시각화한다.

⑧ 사용자 모니터링 인터페이스에 표시한다.

| 서비스 흐름도

1. 실시간 객체 검지

2. 보안 및 안전 관리

3. 대시보드 및 모니터링 시스템

1. 실시간 객체 검지

카메라 → YOLO 객체 검지 모듈 → 데이터베이스 서버

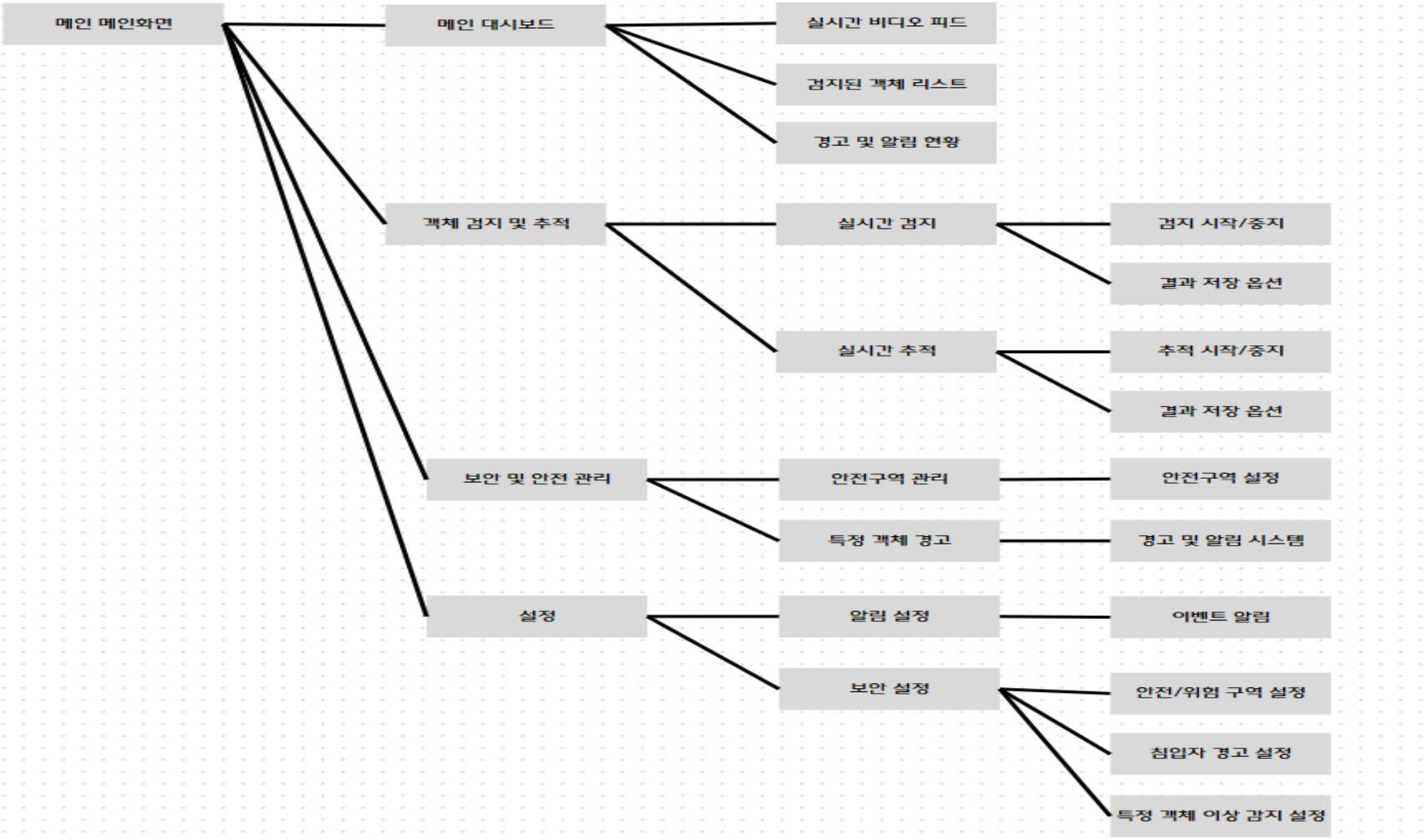
2. 보안 및 안전 관리

① 침입/불법주차 구역 설정 모듈 → 객체 검지 모듈
② 침입구역 경고 시스템 → 데이터베이스
③ 불법주차 경고 시스템 → 데이터베이스
④ 배회 인식 경고 시스템 → 데이터베이스

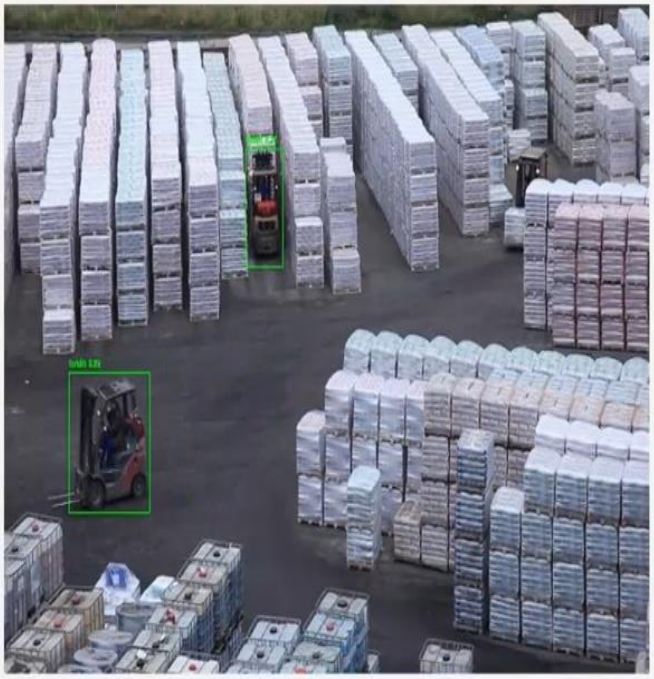
3. 대시보드 및 모니터링 시스템

객체 검지 모듈 → 데이터베이스 서버 → 사용자 인터페이스

| 메뉴 구성도


메인 메인화면
메인 대시보드
실시간 비디오 피드
검지된 객체 리스트
경고 및 알림 현황
객체 검지 및 추적
실시간 검지
검지 시작/중지
결과 저장 옵션
실시간 추적
추적 시작/중지
결과 저장 옵션
보안 및 안전 관리
안전구역 관리
안전구역 설정
특정 객체 경고
경고 및 알림 시스템
설정
알림 설정
이벤트 알림
보안 설정
안전/위험 구역 설정
침입자 경고 설정
특정 객체 이상 감지 설정

| 화면 설계서



영역 설정

위험구역 설정

제한구역 설정

감지된 이벤트

- 2024-08-30 16:30:53 - 제한 구역 (Confidence: 0.8787035346031189)
- 2024-08-30 16:30:51 - 제한 구역 (Confidence: 0.889342725276947)
- 2024-08-30 16:30:51 - 제한 구역 (Confidence: 0.8957871198654175)
- 2024-08-30 16:30:50 - 제한 구역 (Confidence: 0.9176096320152283)
- 2024-08-30 16:30:49 - 제한 구역 (Confidence: 0.9022634625434875)

| 화면 설계서

기능 번호	A03-01
기능 명	객체 검지 결과 표시
기능설명	메인 대시보드, 실시간 객체 검지 영상을 통해 객체 검지 결과를 표시. 경고, 알림 표시
처리내용	- 실시간 객체 검지 영상을 분석하여 경고 및 알림 현황을 표출. 경고 표시는 팝업 형식으로 표시. 알림은 시간 순으로 표시.
비고	- 연결되는 기능 : A03-02 (경고 및 알림 현황 표출)
요구사항 명	대시보드 및 모니터링 시스템

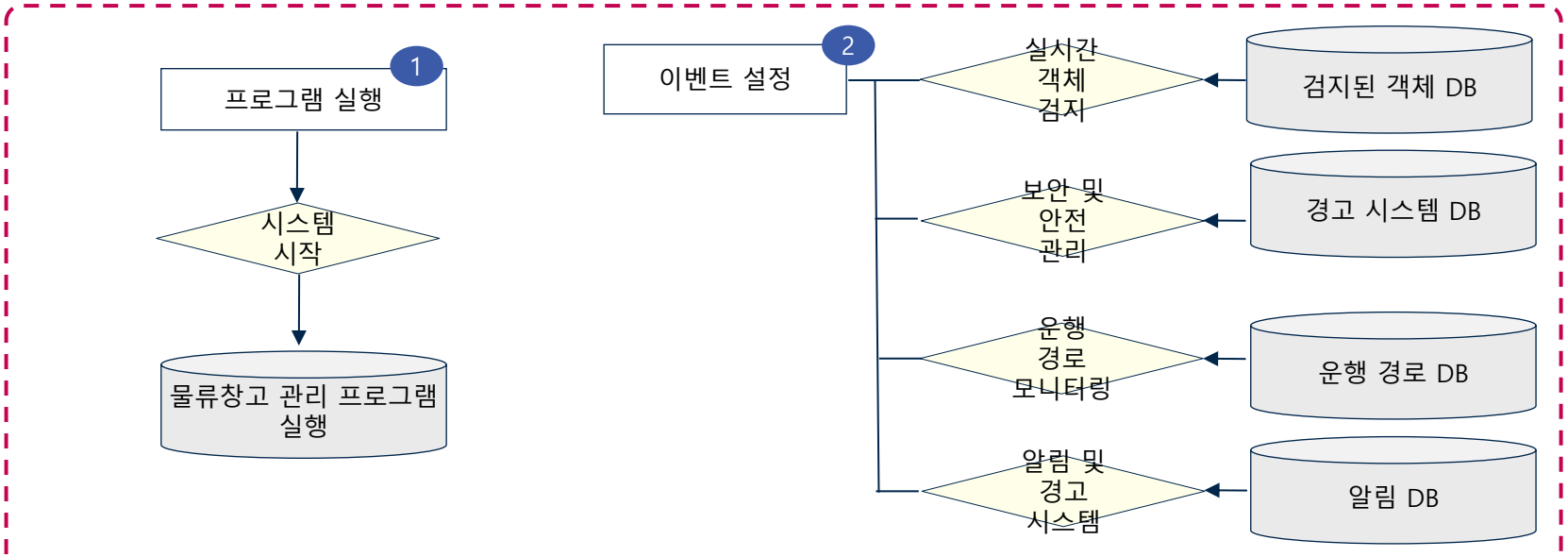
| 기능 처리도(기능 흐름도)

실무 산출물 형식

프로그램 ID	YOLO_SmartWarehouse	프로그램 명	스마트 물류창고 관리 프로그램	작성일	2024 . 07 . 14 .	Page	1/3
개요	YOLO 기반의 객체 검지 기술을 통해 물류창고 내의 다양한 객체를 실시간으로 검지하여 안전 및 보안 관리를 제공하는 프로그램					작성자	박재현

기능 흐름도

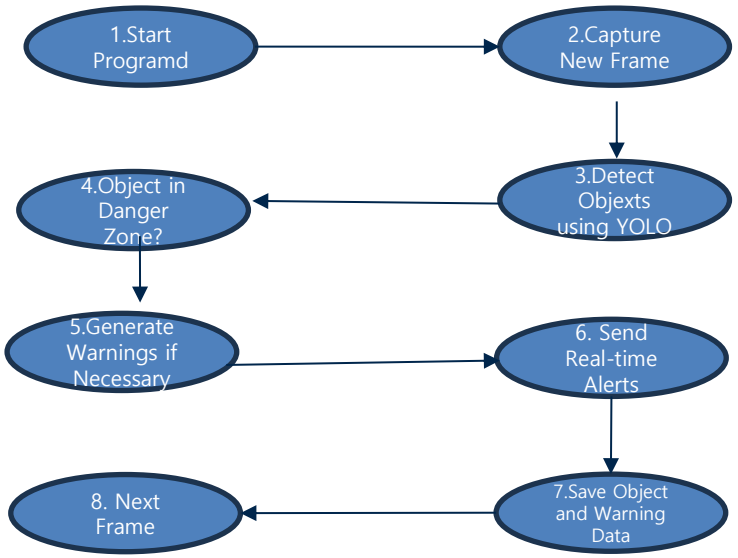
<스마트 물류창고 관리 시스템 기능 흐름도>



1 시스템을 시작하여 물류창고 관리 프로그램을 실행합니다.

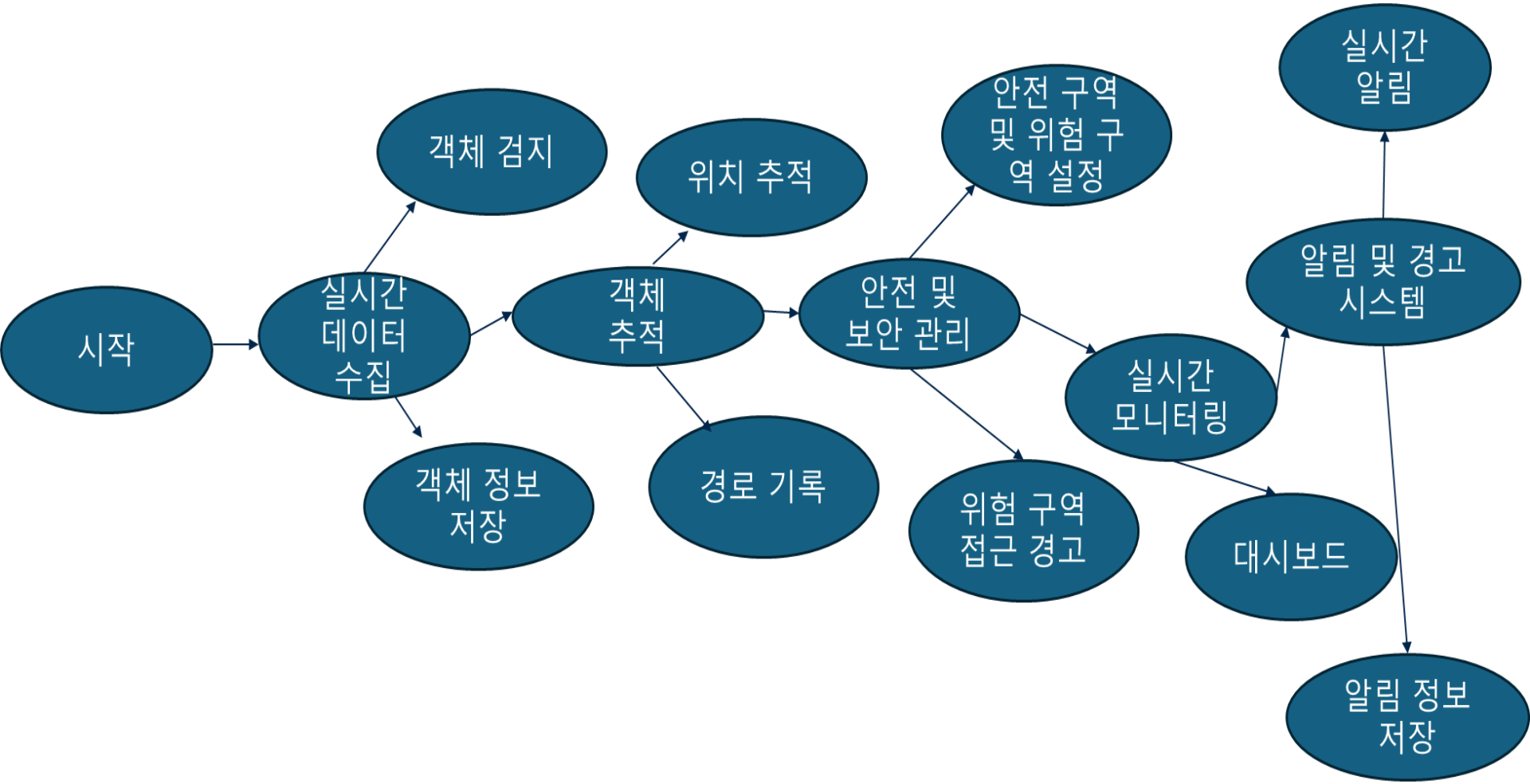
2 사용자는 위험구역, 제한구역을 선택합니다.

| 알고리즘 명세서



1. 시스템을 시작하여 물류창고 관리 프로그램을 실행합니다.
2. 실시간 카메라 스트림에서 새로운 프레임을 입력 받습니다.
3. YOLO 모델을 사용하여 입력된 프레임에서 객체를 검지합니다.
4. 객체 위치가 설정된 구역 내에 있는지 확인합니다.
   Yes: 구역 내에 있으면 경고를 생성합니다.
   No: 구역 밖에 있으면 다음 단계로 이동합니다.
5. 설정 구역에 있는 객체에 대해 실시간 경고를 생성합니다.
6. 생성된 경고를 사용자에게 실시간으로 전송합니다.
7. 검지된 객체와 경고 정보를 데이터베이스에 저장합니다.
8. 다음 프레임을 처리하기 위해 프로세스를 반복합니다.

| 데이터 수집처리 정의서


시작
실시간 데이터 수집
객체 검지
객체 정보 저장
객체 추적
위치 추적
경로 기록
안전 및 보안 관리
안전 구역 및 위험 구역 설정
위험 구역 접근 경고
실시간 모니터링
알림 및 경고 시스템
실시간 알림
대시보드
알림 정보 저장

| 프로그램 - 목록

기능 분류	기능번호	기능 명
ODT (객체 검지)	ODT-01-01	YOLO 모델을 이용한 객체 검지
	ODT-01-02	실시간 비디오 스트림 처리
	ODT-01-03	검지 및 추적결과 저장 및 전송
SEA (보안 및 안전관리)	SEA-01-01	위험 구역 설정
	SEA-01-02	위험 구역 접근 시 경고 알림
	SEA-01-03	지게차 등 작업 차량 경로 모니터링 시스템
	SEA-01-04	작업자 실시간 경고 시스템
	SEA-01-05	특정 객체 경고 시스템
DAS (데시보드 및 모니터링 시스템)	DAS-01-01	실시간 객체 검지 결과 표시
	DAS-01-02	경고 및 알림 현황 표시

| 테이블 정의서 - ERD

경고 및 알림 테이블(Alerts)

항목명	데이터 타입	설명
id	INT	경고 ID
label	String	경고 유형
confidence	Float	관련 객체 ID
timestamp	DATETIME	경고 발생 시간

관리자 테이블

항목명	데이터 타입	설명
username	String	사용자 이름
password	String	비밀번호
id	INT	사용자 ID

| 핵심소스코드

```
    }
</style>
<script src="https://cdn.socket.io/4.0.0/socket.io.min.js"></script>
<script>
    document.addEventListener("DOMContentLoaded", () => {
        const socket = io();
        const eventList = document.getElementById("eventList");

        socket.on('new_event', (data) => {
            const eventItem = document.createElement("li");
            eventItem.textContent = `${data.timestamp} - ${data.label} (Confidence: ${data.confidence})`;
            eventList.insertBefore(eventItem, eventList.firstChild);
        });
    });
</script>
```

```
<div class="container">
    <div class="main-content">
        <img src="{{ url_for('video_feed') }}" alt="CCTV Feed" style="width: 640px; height: 480px;">
        <p>감지된 이벤트</p>
        <ul id="eventList"></ul>
        <br>
    </div>

    <div class="right-menu">
        <!-- 침입 이벤트 설정 -->
        <div class="event-group">
            <strong>위험구역 이벤트 설정</strong>
            <button>사람</button>
            <button>동물</button>
            <button>차량</button>
            <a href="{{ url_for('select_roi') }}">
                <button>위험구역 설정</button>
            </a>
        </div>

        <!-- 불법 주정차 이벤트 설정 -->
        <div class="event-group">
            <strong>제한구역 이벤트 설정</strong>
            <a href="{{ url_for('select_roi2') }}">
                <button>제한구역 설정</button>
            </a>
            <button>시간 설정</button>
```

```
            # 위험 구역 dangerzone
            if class_name == 'person' and \
                self.roi_dangerzone[0] < center_x < self.roi_dangerzone[0] + self.roi_dangerzone[2] and \
                self.roi_dangerzone[1] < center_y < self.roi_dangerzone[1] + self.roi_dangerzone[3]:
                detected_events.append({'type': '위험 구역', '신뢰도': conf.item()})
                cv2.rectangle(frame, (int(xyxy[0]), int(xyxy[1])), (int(xyxy[2]), int(xyxy[3])), (0, 0, 255), 3)
                cv2.putText(frame, 'danger zone', (int(xyxy[0]), int(xyxy[1]) - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 0, 255), 2)

            # 제한구역
            elif class_name == 'forklift' and \
                self.roi_restrictzone[0] < center_x < self.roi_restrictzone[0] + self.roi_restrictzone[2] and \
                self.roi_restrictzone[1] < center_y < self.roi_restrictzone[1] + self.roi_restrictzone[3]:
                detected_events.append({'type': '제한 구역', '신뢰도': conf.item()})
                cv2.rectangle(frame, (int(xyxy[0]), int(xyxy[1])), (int(xyxy[2]), int(xyxy[3])), (255, 0, 0), 3)
                cv2.putText(frame, 'restrict zone', (int(xyxy[0]), int(xyxy[1]) - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (255, 0, 0), 2)

            else:
                cv2.rectangle(frame, (int(xyxy[0]), int(xyxy[1])), (int(xyxy[2]), int(xyxy[3])), (0, 255, 0), 2)
                cv2.putText(frame, f"{class_name} {conf:.2f}", (int(xyxy[0]), int(xyxy[1]) - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0)

    # 차와 사람의 거리가 가까울 때 알림
    for obj_id1, obj1 in objects.items():
        if obj1['class'] == 'person':
            for obj_id2, obj2 in objects.items():
                if obj2['class'] == 'forklift':
                    distance = np.sqrt((obj1['center'][0] - obj2['center'][0])**2 + (obj1['center'][1] - obj2['center'][1])**2)
                    if distance < 300:
                        detected_events.append({'type': '근접 알림', '신뢰도': min(obj1['conf'].item(), obj2['conf'].item())})
                        cv2.line(frame, (int(obj1['center'][0]), int(obj1['center'][1])), (int(obj2['center'][0]), int(obj2['center'][1])), (255
                        cv2.putText(frame, 'Proximity Alert', (int((obj1['center'][0] + obj2['center'][0]) / 2), int((obj1['center'][1] + obj2['
```

| 참조- 개발 환경 및 설명

	항목	적용내역
S/W 환경	Visual Studio Code	개발 전 과정에 활용
	Python	파이썬으로 작동되는 부분(YOLO, Flask 등)에 활용
	Windows	개발 전 과정에 활용
	Linux	개발 전 과정에 활용
	YOLO	객체 검지에 활용
	SQLite	DB 구축에 활용
	OpenCV	영상 처리, 분석에 활용
	Flask	기능을 웹 상에 구현하는데 활용
	HTML	웹사이트 템플릿 제작에 활용
	CSS	웹사이트 템플릿 제작에 활용
	JavaScript	웹사이트 템플릿 제작에 활용
	Figma	웹 디자인에 활용
H/W 환경	CCTV	영상 촬영 / 확보에 활용

Thank you