IOT 프로젝트 보고서

스마트 쓰레기통(5조)

과목명 : IOT

교수명 : 최차봉 교수님

학과 : 소프트웨어학과

팀원: 2016125035안일훈

2017126106전예림

2016125061이지원

2016125065이효중

1. 프로젝트 선정 배경 및 개요

1.1 배경

본인이 사는 아파트 단지 외부에 있는 빌라단지에 쓰레기통이 있는데, 중부대학교가 최근에 고양캠퍼스를 내가 사는 동네에 지어서, 신축빌라가 세워졌다. 그럼에도 빌라들이 버리는 쓰레기장의 크기는 그대로 이기 때문에, 항상 쓰레기통이 넘쳐서 주변에 까지 쓰레기가 버려지고 동네 미관을 해치는 것을 목격하고 생각하게 되었다.

1.2 프로젝트 목적

단순한 쓰레기통이 아니라, 쓰레기통에 IOT 기술을 활용하여, 쓰레기통의 유무를 파악하고, 쓰레기업체가 넘치지 않도록 관리하는 목적을 가짐. 추가적으로 쓰레기통이 꽉 찼을 때, 옆에 쓰레기를 쌓아서 넘쳐 흐르지 않도록, 쓰레기통을 수동으로 눌러서 가용공간을 확보하는 기능을 추가하여, 효율적인 쓰레기장을 만들 수 있도록 하기 위함이 있다.

1.3 프로젝트 요구사항

1) 현재 쓰레기통의 용량을 체크해야 함.

2) 쓰레기통의 위치를 표시함.

3) 꽉 찬 쓰레기통들을 효율적으로 수거할 수 있도록 최적화 경로 표시가 가능 해야 함.

4) 사람 접근 시, 주의를 줄 수 있도록 장치부착이 필요함.

5) 가용 공간을 확보 할 수 있도록 수동 장치를 부착하고, 요구해야 함

1. Framework

2.1 프로젝트 시나리오

1) 쓰레기통 상태 확인

초음파 센서를 통해서 앞에 물체가 있다면, 쓰레기통이 꽉 찬 것으로 판단

2) 꽉 찼으면, 상태를 shadow에 report하고, dynamoDB에 저장함

거리를 조정하여, 어떤 물체가 앞에 있다면 쓰레기통이 꽉 찬 것으로 판단하고, shadow에 publish를 함. 규칙을 이용하여, dynamoDB에 상태를 업데이트 함 (state : Full or NotFull)

3) dynamoDB의 값을 가져와서 쓰레기통이 꽉 찼음을 지도에 표시 + 최적화 경로 계산

Javascript를 이용해 값을 추출하고 maps.html에 get으로 값을 넘겨줌

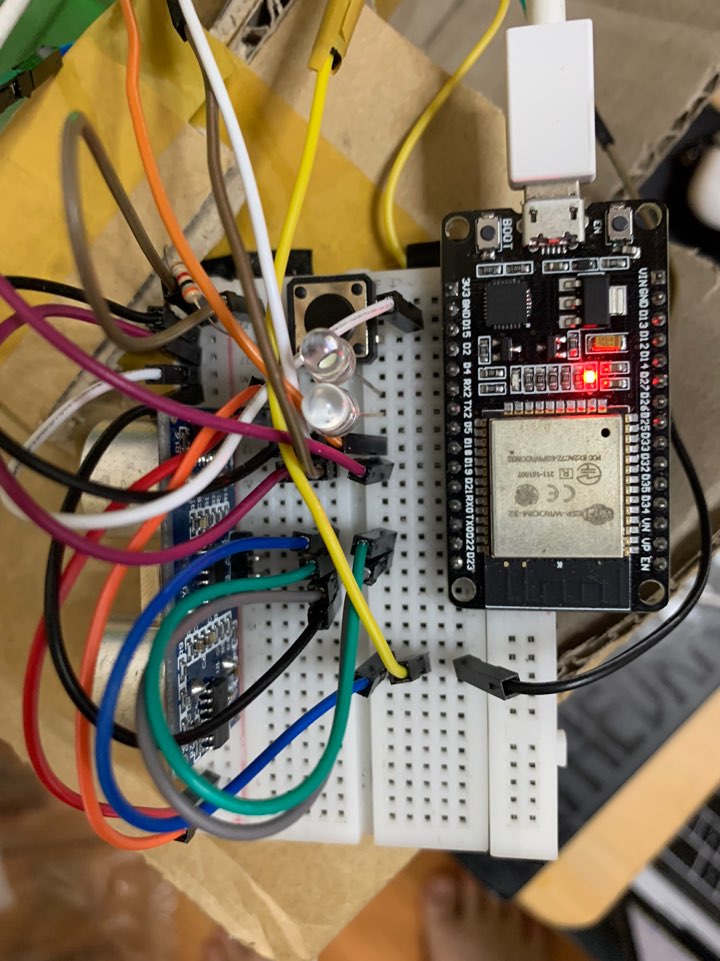
Maps.html안에 js코드를 이용하여, 각 쓰레기통의 상태를 확인하고, maps에 어떻게 표시 되어야 하는지 값을 변경하고 최종적으로 카카오맵에 line과 마커를 표시함

4) 사람이 접근 할 때, 경고를 하도록 함

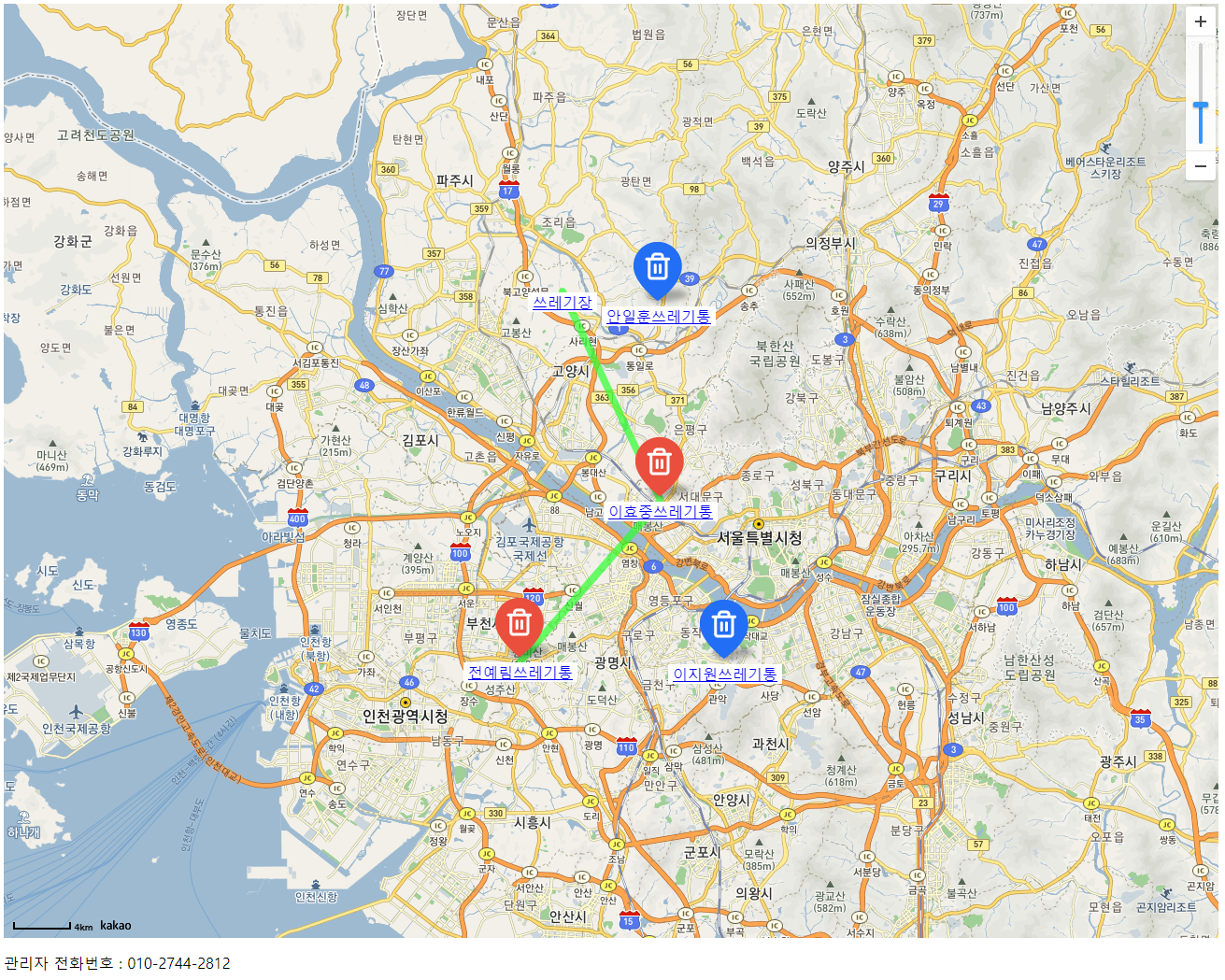
이 부분은 스피커가 없어서 빨간색 LED를 점멸하는 것으로 대체함

경고 메시지는 쓰레기통에 잘 버려주세요. 쓰레기통에 음료를 버리지 마세요. 등등

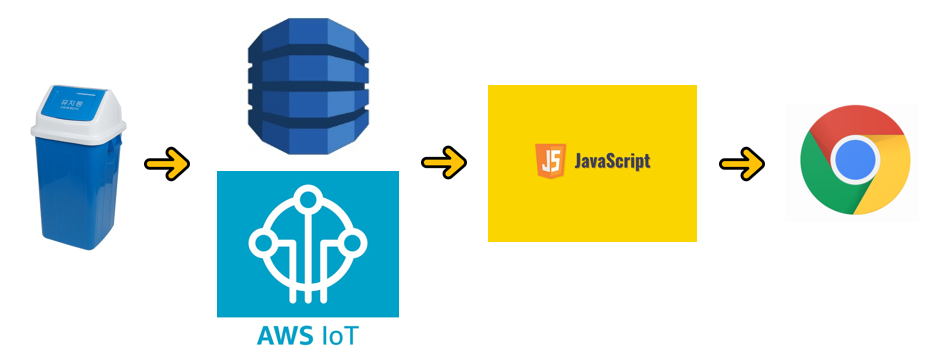
2.2 시스템 구성도



2.3 어플리케이션 구상도



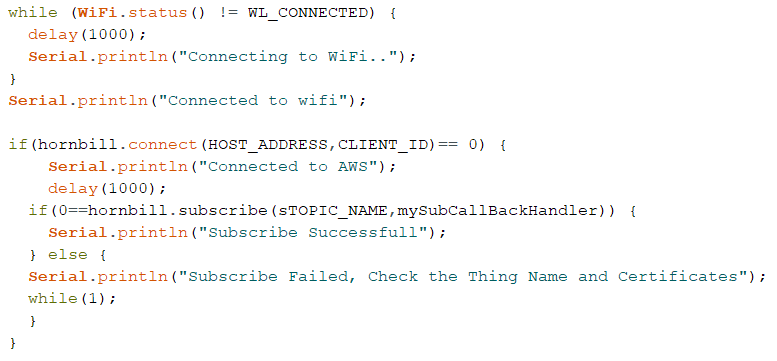
1) 웹페이지 메인 화면 – 쓰레기통 위치표시 + 꽉 참(빨간색)/안 참(파란색) + 수거경로(초록선)



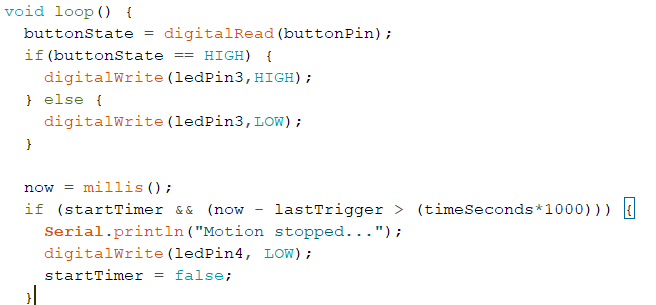
2) 통합 구상도

3. 코드 설명

3.1) 와이파이 연결 / aws 연결



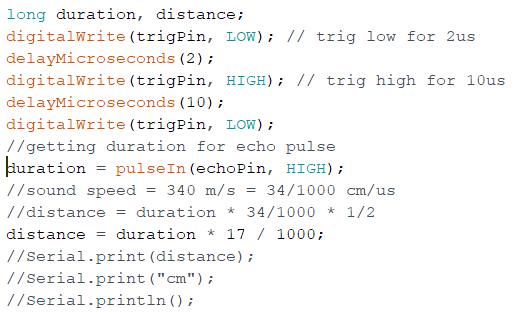
3.2) 버튼 구현 / PIR 센서 타이머구현



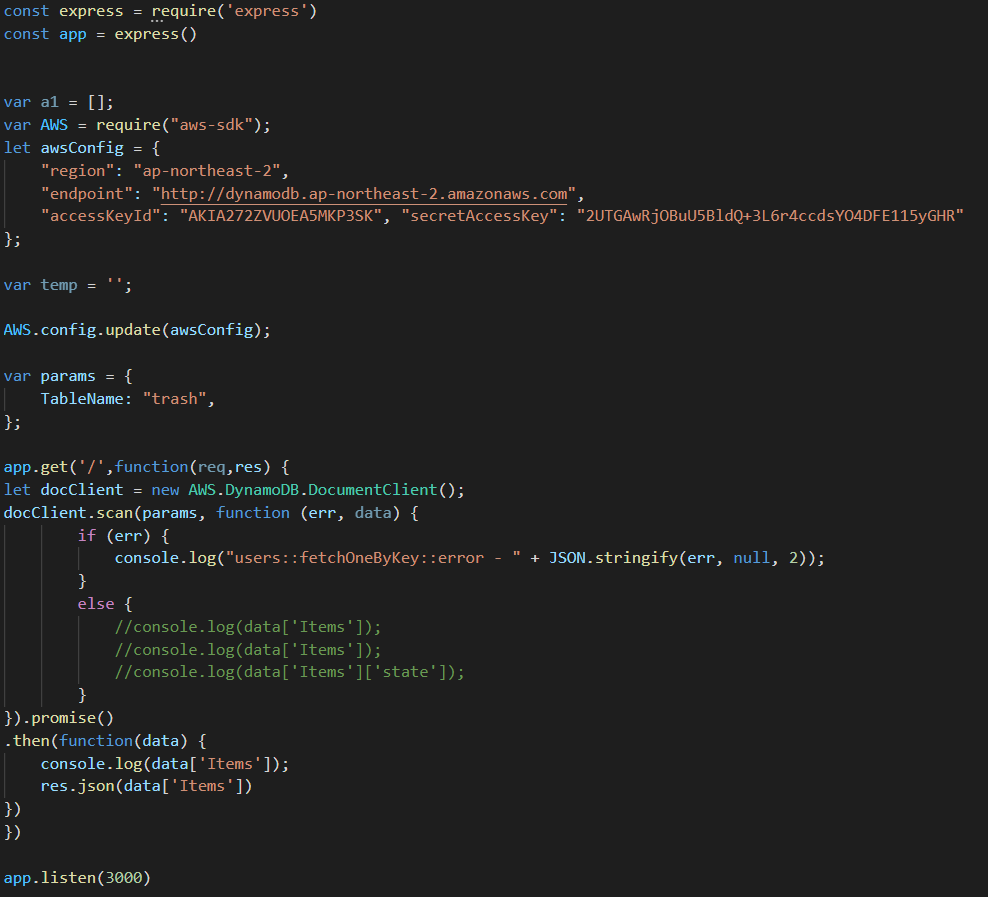
3.3) publish 가 일어나는 코드



3.4) 초음파 센서 거리측정 코드

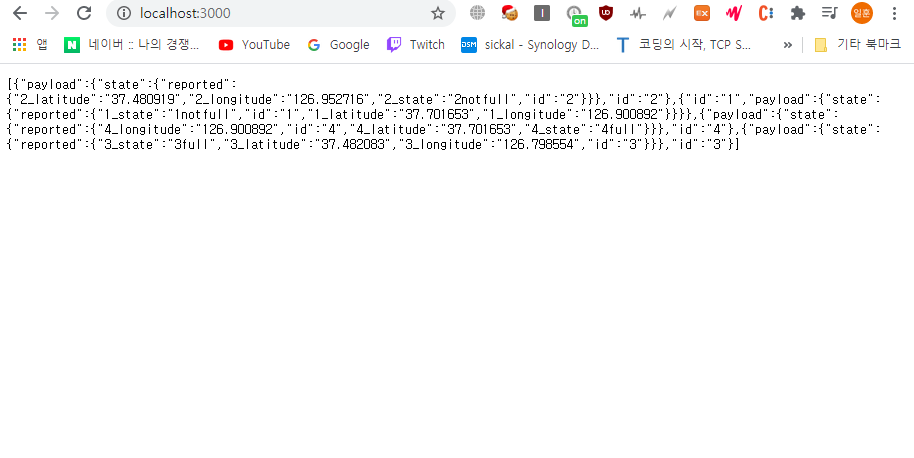


3.5) JS코드



AWS dynamoDB에 접속하여 payload를 가져와서 data를 뿌려줌

3.6) 3000 port로 접근하면 뿌려줌



3.7) maps.html은 카카오맵 api를 활용하여

마커와 최적화경로(직선)을 표시해줌

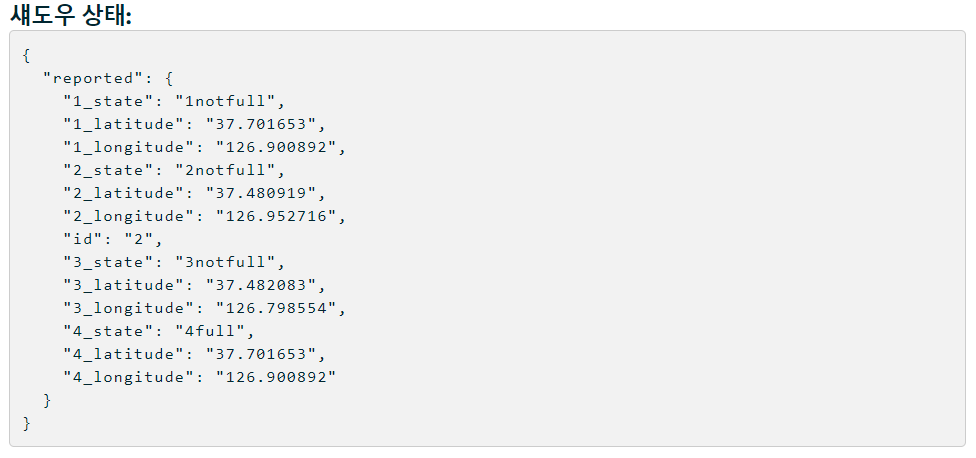
최적화 경로는 탐욕알고리즘을 사용하여, 시작위치인 쓰레기처리장에서 가장 가까운 곳을 다음으로 정하고 그곳에서 다시 최단거리를 다음 목적지로 정하는 알고리즘을 이용하여 최적화경로를 설정합니다. (greedy 알고리즘 특성상 실제 최적이 아닐 수도 있음)

자세한 내용은 첨부된 maps.html 파일을 확인 해주세요.

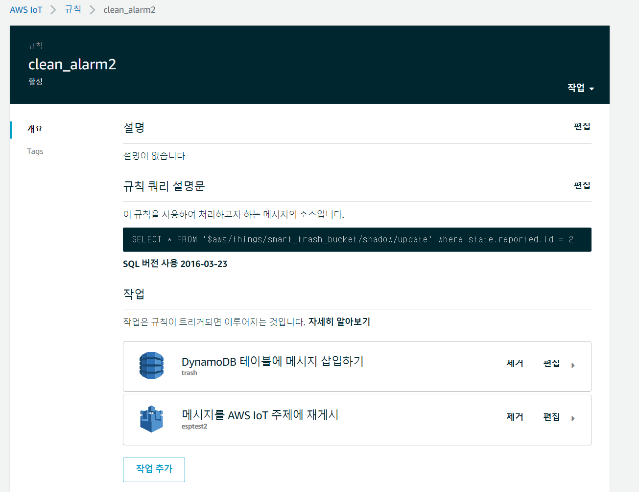
4. AWS 설정

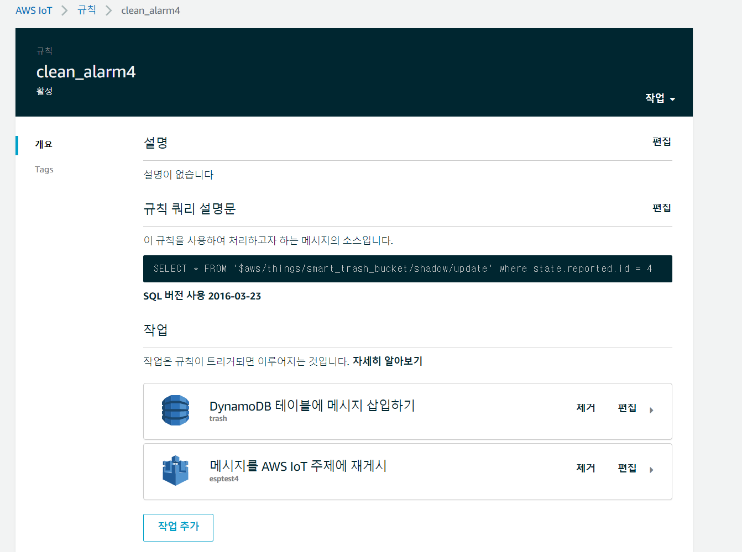
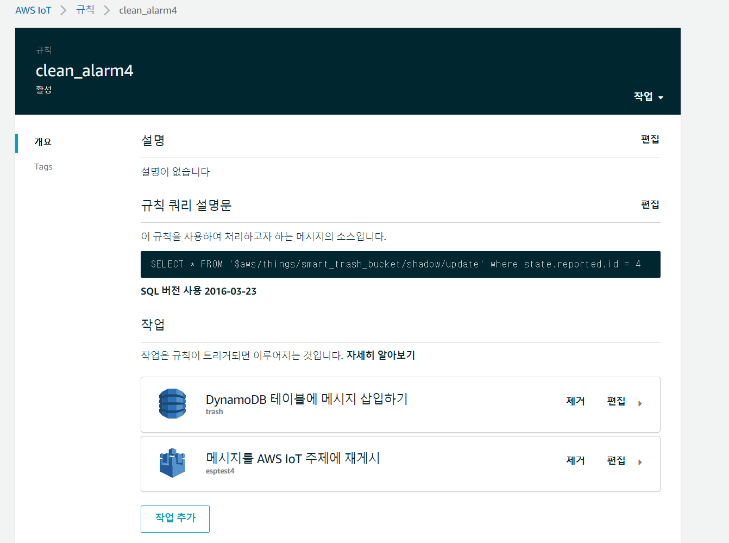
4.1) 사물&섀도우





4.2) rule

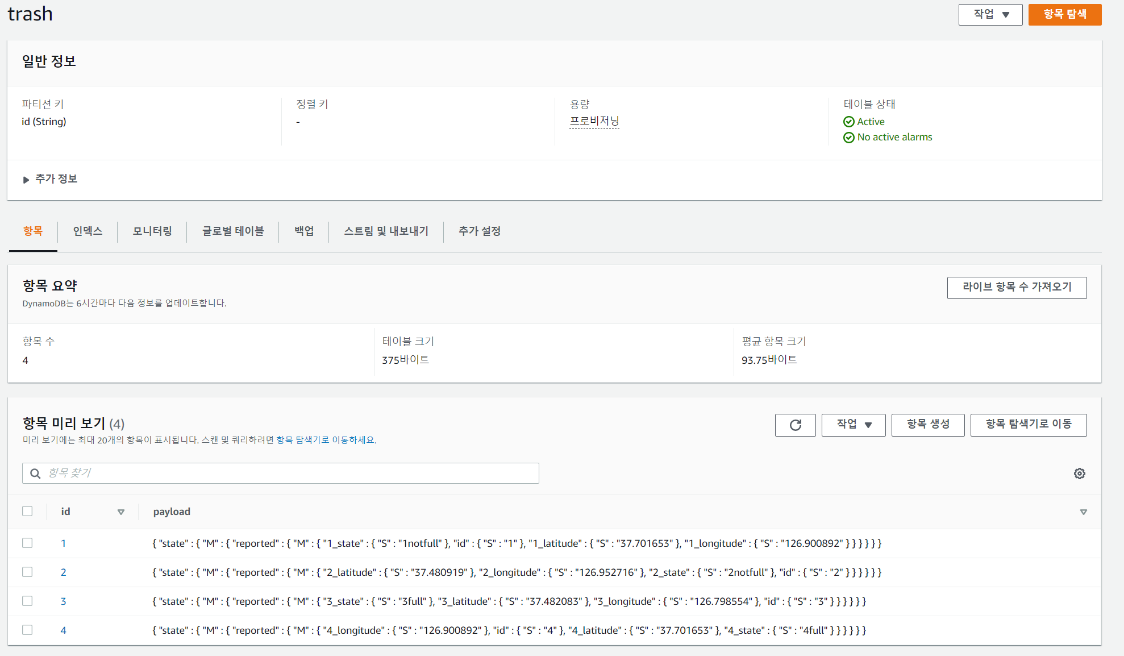




각각의 룰은 id에 따라서 trash 테이블에 id에 맞는 값을 넣어준다.

각각의 사물은 고유 id를 가지고 있고, 코드에 구현되어 각각 업로드 된다.

4.3) DynamoDB



5. 문제점 및 해결방안

Aws Shadow api를 이용하여 shadow에 접근 못하는 문제

-> DynamoDB를 이용하여, shadow를 저장하고, JS를 DB에 연결하여, 값을 가져와서 maps.html에서 처리하게 함.

CORS 문제

-> html에 JS코드가 DB에 접근하여 값을 가져온 NodeJs에 GET을 날렸을 때, 값을 못가져오는 문제였는데, chrome에 추가 확장프로그램인 access-control-allow-origin을 활용하여, 해결 하였다.

DB에서 가져온 데이터 payload문제

-> NodeJS에서 전처리 하지 않고 console.log() 로 GET에 응답후 maps.html의 js코드에서 stringify로 전처리후 배열로 처리하고 각 배열(id당)에 (id)notfull이 들어있는지 확인 후 미리 선언된 position배열을 수정하여 최종적으로 카카오맵에 표시 되도록 해서 해결

기여부분

안일훈 : Nodejs코드(서버), html코드 ,esp32(PIR와LED구성 및 시연)

espcam설정,DynamoDB연동(NodeJS코드에 부착하여 관리)

전예림 : html코드,카카오맵api/자바스크립트 (최적화경로,마커)

이효중 : 보고서 esp32 코드(초음파센서 코드와 버튼구현)

이지원 : 테스팅 및 esp32 사물구성(aws iot,섀도우,사물설정)