**Key DB Tools 및 배포 Guide**

2023.01.16

문서개정이력

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **버전** | **작성일** | **작성자** | **변 경 내 용** | **비고** |
| 1 | 2023.01.16 | 최진우 | 최초 작성 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목 차

1. KeyDB 소개----------------------------------------------------------------------------------------
2. KeyDB Tools---------------------------------------------------------------------
3. 배포 프로세스--------------------------------------------------------------------------------
4. 향후 과제 ----------------------------------------------------------------------------------
5. **KeyDB 소개**

파토스 상용서버에서 관리 하는 고객사의 단말기들의 인증키 발급 및 관리 하는 것을 말한다.

현재, 파토스에서는 주로 소방(베가웍스),도로공사,해양에너지,스카니아 등의

업체의 단말기를 관리하고, 고객사에서 발주 요청이 들어오면

즉각적으로 단말기의 Uid나 mac주소를 적절한 해시함수를 적용하여,

단말 인증 처리를 해준다.

키등록 원리는 아래의 깃에서 소스 코드를 참고.

기존 소스는 c++,postgresql로 구성되어 있다.

* Git : ssh://git@git.fatos.biz:30022/source/tools/site-key.git
* DB connect
  + Host: service.fatos.biz
  + Port : 15432
  + Database : fatosdb
  + 계정정보 : 서버담당자에게 문의
  + Schema : sitekey.fatos\_service, sitekey\_fatos\_authex

fatos\_service는 각 사이트들의 사이트번호(site),사이트 사이드번호(site\_sub)를 관리 하는 스키마이며,

fatos\_authex는 각 사이트 안에서의 단말 인증키들을 관리하는 스키마이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<fatos\_service Schema의 구성>

텍스트이(가) 표시된 사진

자동 생성된 설명

<fatos\_service Schema 내용의 일부>

텍스트이(가) 표시된 사진

자동 생성된 설명

<fatos\_authex Schema의 구성>

텍스트이(가) 표시된 사진

자동 생성된 설명

<fatos\_authex Schema의 내용의 일부>

텍스트이(가) 표시된 사진

자동 생성된 설명

<sitekey 단순 쿼리 조회 예>

후술할 2장에서는 기존 소스를 dart & flutter로 구현한 keyDB Tools를 설명.

1. **KeyDB Tools**

Flutter로 웹 어플리케이션을 구현하였고,

Api는 Node.js를 활용하여,구현하였다.

<소스 코드>

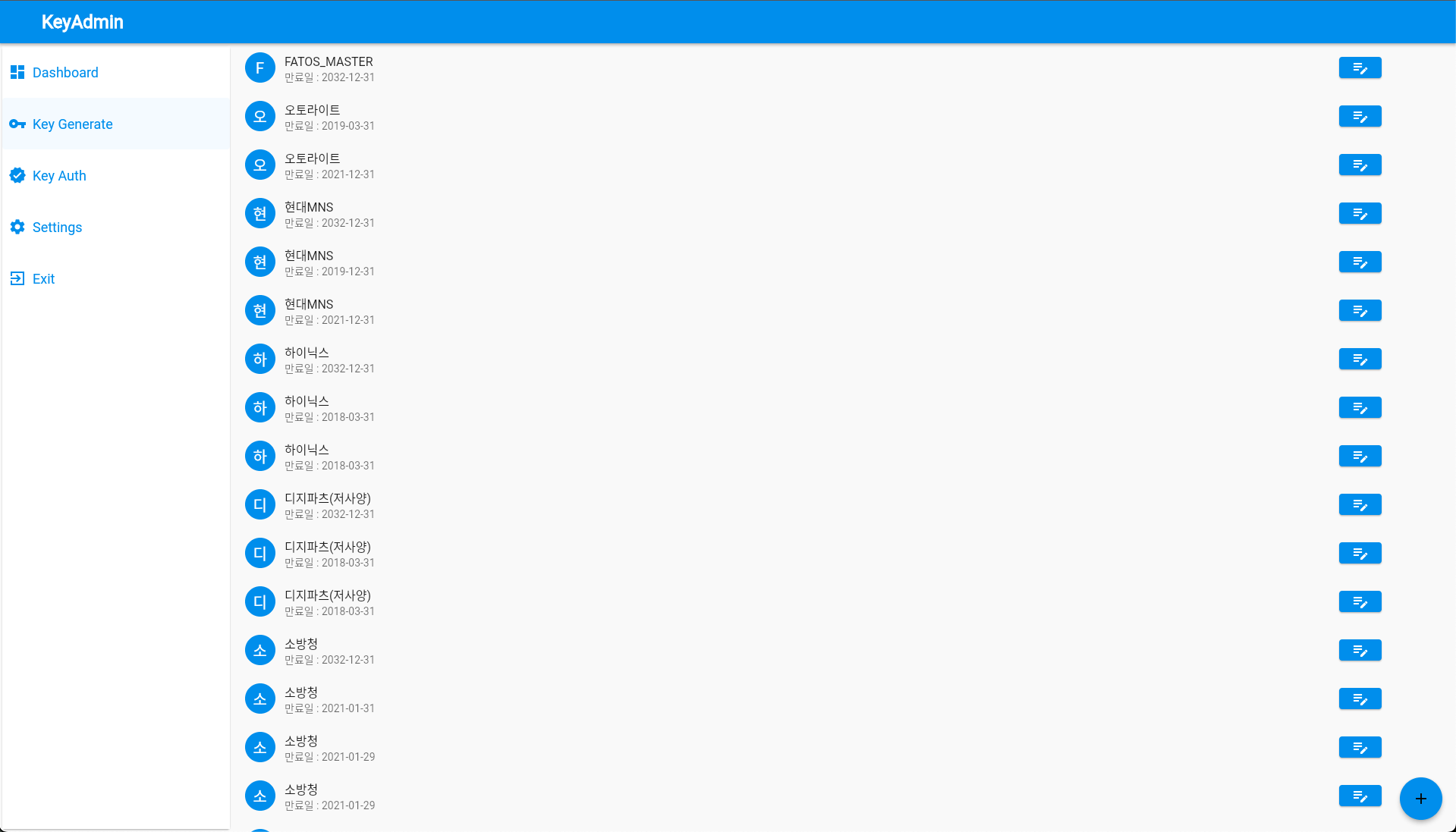
* Git : ssh://git@git.fatos.biz:30022/jinwou.choi/key\_admin\_tool.git (플러터)
* Git : ssh://git@git.fatos.biz:30022/jinwou.choi/NodePG.git (NodeJS)
* 개발 tool : Android Studio, Code, DBeaver 등

2-1 Flutter 기반의 key admin tool

텍스트이(가) 표시된 사진

자동 생성된 설명

<초기 화면 – 로그인 (firebase에서 계정 관리)>



<초기 화면- 로그인 후>

텍스트이(가) 표시된 사진

자동 생성된 설명

<소스 구성>

main.dart : 처음 로그인 상태에 대한 코드

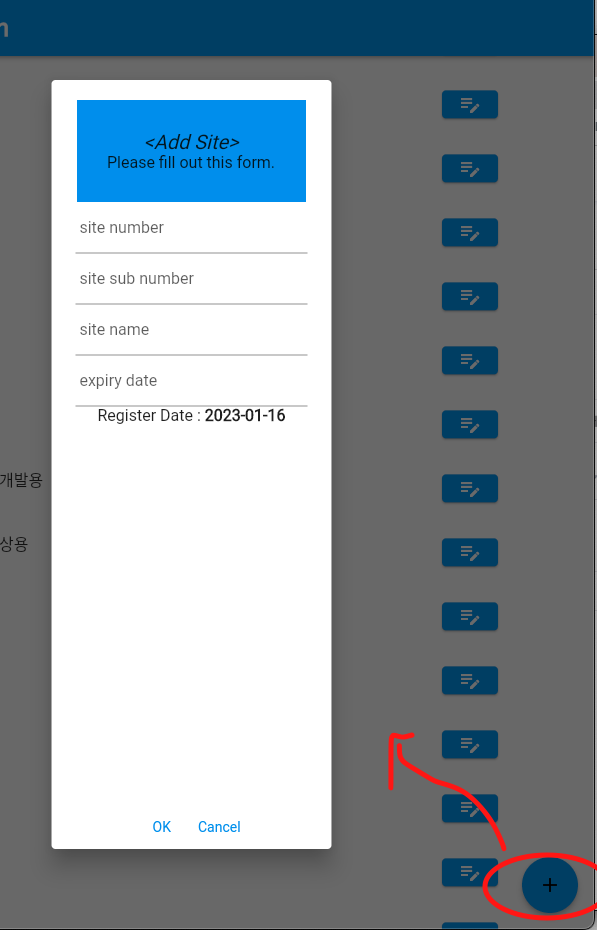
homepage.dart: 대시보드,인증,생성,설정,나가기의 navigator 를 구현한 코드

dashboard.dart : 로그인 후 최초 화면인 각 사이트 정보,추가,편집,삭제 기능을 구현한 코드

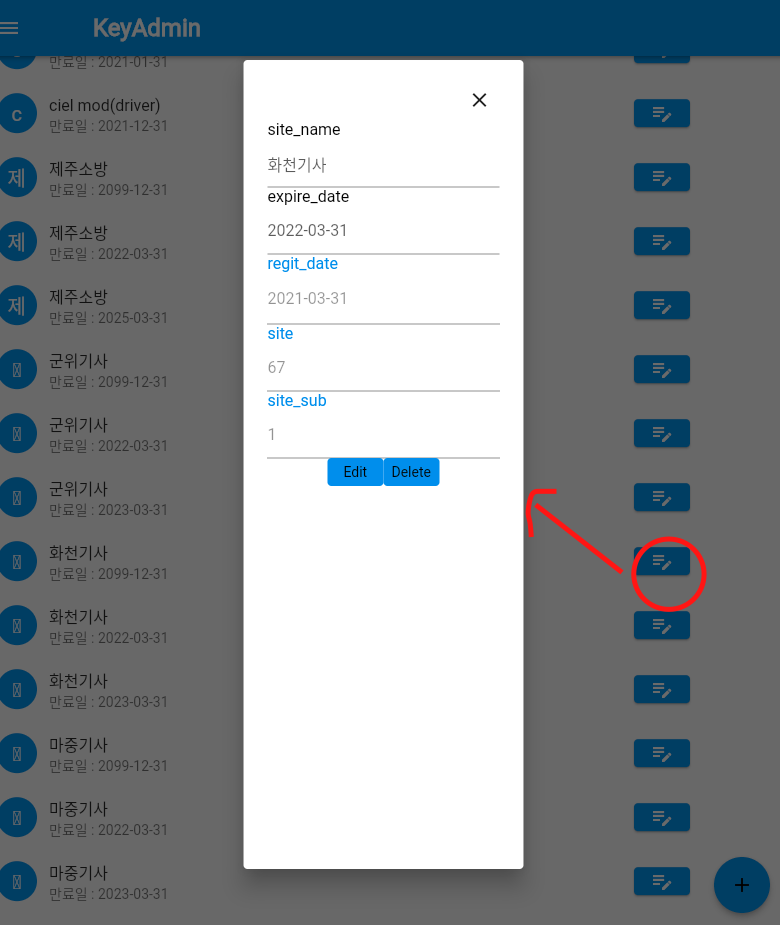
keyauth.dart : 각 사이트를 점검하여 , 잘못된 인증키가 있는지 점검하는 기능을 구현한 코드

keygen.dart : 가장 중요한 부분으로써, 단말 인증에 대한 모든 것을 구현한 코드.

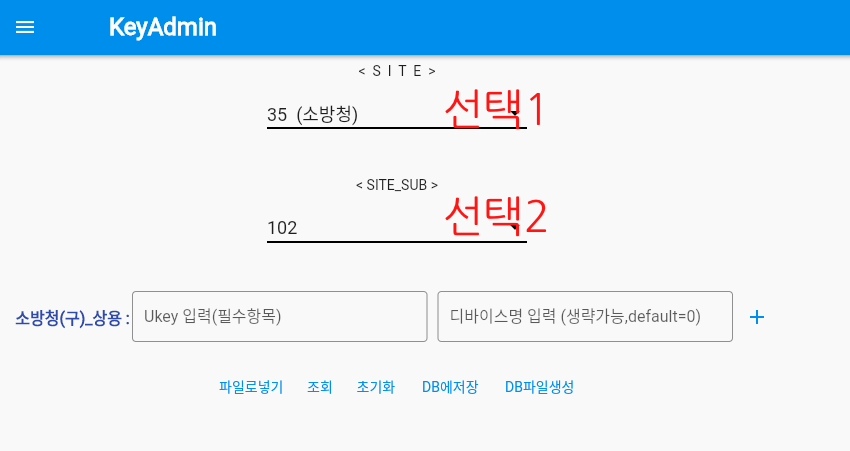
**# 자세한 내용은 소스 코드에 주석 내용 확인!**



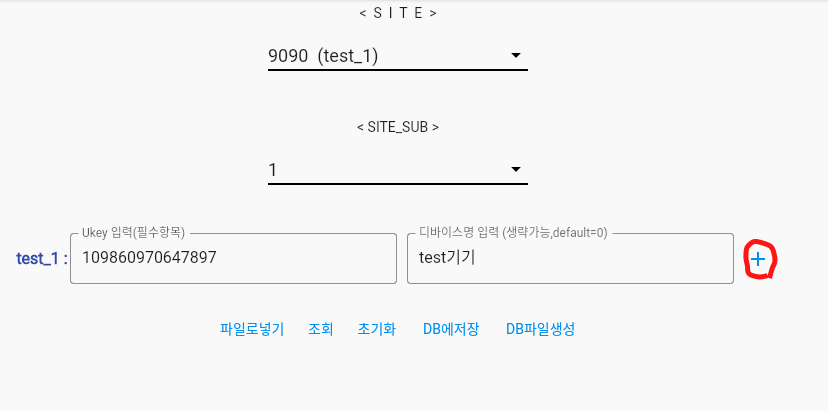
<dashboard 페이지에서 우측 하단 + 버튼을 클릭하여 특정 업체의 사이트를 추가>



<dashboard 페이지에서 특정 사이트의 우측 편집 버튼 클릭하면 , 사이트정보와 편집,삭제 기능 사용 >



<keygen 페이지에서 site, site\_sub의 선택을 통해 키 등록할 준비 >



텍스트이(가) 표시된 사진

자동 생성된 설명

<keygen 페이지, ukey,device명 입력, 리스트에 일시적으로 쌓임>

* Keygen의 기능

-파일로 넣기 : ukey와 device가 명시된 텍스트 파일 삽입하면 , 자동으로 리스트화 해주는 기능

(ex)site-key/00.input 파일 넣으면 됨.but한글미지원,테스트중)

-조회 : 특정 사이트의 키 값들을 조회

-초기화 : 기 등록 된 리스트의 값을 삭제

-DB에 저장 : 리스트의 값들을 db에 저장 그리고 txt 파일 생성

-DB파일생성 : 최종파일 .db 생성

텍스트이(가) 표시된 사진

자동 생성된 설명

<생성된 .db 파일의 내용 : 단말기들의 serialkey가 오름차순으로 나열되 있는 것임>

테이블이(가) 표시된 사진

자동 생성된 설명

<KeyAuth 항목의 키 검증 기능>

2-2. Node.js의 CRUD 및 API 구현

플러터에서 선보인 db 조회, 등록 , 수정 , 삭제 같은 기능들은

Node.js에서 쿼리를 통해 구현하였으며, 각 쿼리마다 API 주소 부여,

다음은 NodePG 디렉토리의 api.js와 connection.js소스 코드의 일부이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

<connection.js – 데이터베이스 연결 될 포트를 정의 하는 부분>

텍스트이(가) 표시된 사진

자동 생성된 설명

<api.js 소스코드의 일부-주소 선언 및 CRUD 구현 >

2-3 사용법

1. Node에서 먼저 포트를 열어야 함,

Node 프로젝트 경로에서 node api.js 혹은 node api.js & lt -p 3300 --subdomain mogoskeyadmin2 --print-\n-requests 실행

텍스트이(가) 표시된 사진

자동 생성된 설명

* Node api.js 뒤에 lt란?

Localtunnel의 약자로서, 외부 망에서 localhost를 도메인을 유저가 지정한 임의의 것으로 무료로 접속하게 끔 해주는 기능이다. (아직 회사의 nginx도메인을 따지않은 상태라 그렇다)

환경에 따라 다르겠지만 port는 3300으로 하였고 subdomain은 mogoskeyadmin2로 지정.

1. 웹 앱 실행

안드로이드스튜디오나 code로 key\_admin\_tool 프로젝트 실행

* (크롬)빌드 => 로컬호스트버전으로 사이트 조회 등이 이상 없다면 1)2)과정 정상 수행 된 것.

1. 베타 버전 앱 실행

(현재기준) 1)번처럼 포트 열고, <https://keyadmin-1e9d1.web.app/> 접속하면 된다.

1. 웹 앱 배포 하기

로컬에서 코드 수정 작업하고, 이를 베타 버전에도 적용 시키려면,

일련의 과정이 필요.

-Key\_admin\_tools 프로젝트 경로 내 터미널 실행!

-firebase init

텍스트이(가) 표시된 사진

자동 생성된 설명

< 이 곳에서 4번째 항목 hosting: configure…… 선택하고 스페이스바 누르기>

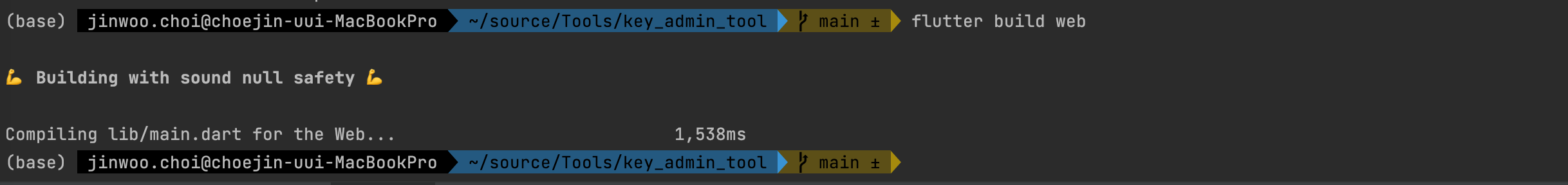
텍스트이(가) 표시된 사진

자동 생성된 설명

<나오는 질문에는 위와 같이 입력>

-위와 같은 작업 끝나면 , 호스팅 준비가 마친 상태

-flutter build web 입력



<flutter build web 정상 처리 결과>

firebase deploy –only hosting 입력

텍스트이(가) 표시된 사진

자동 생성된 설명

<명령어 입력 후, 정상적인 결과 화면 , 맨 아래 hosing URL 출력>

텍스트이(가) 표시된 사진

자동 생성된 설명

<정상적인 결과 화면>

1. **배포 프로세스**

고객사에서 단말 인증 요청 올 때, 어떤 조치를 해야 하는지에 대한 가이드.

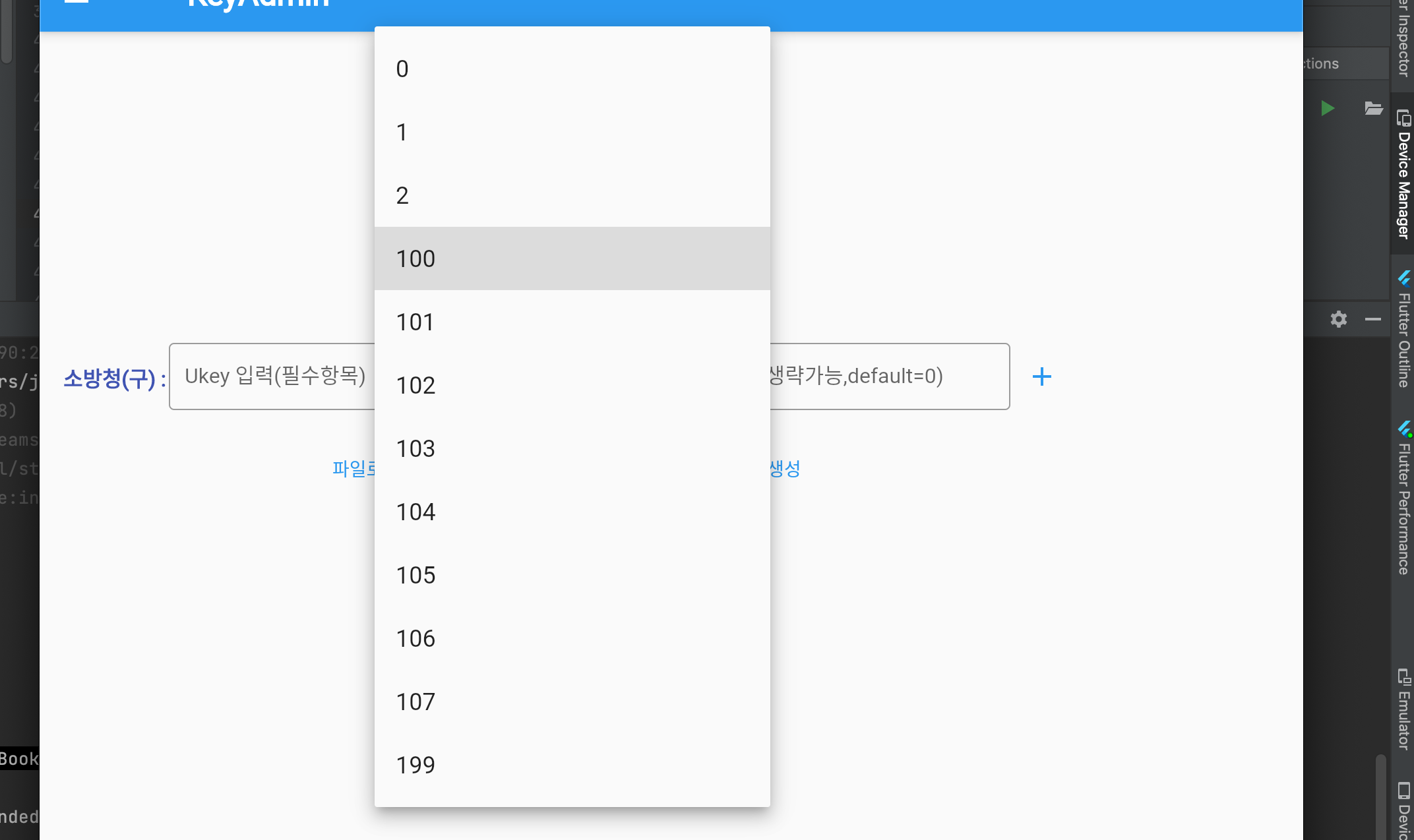
3-1) 고객사 마다 다른 serial값 생성 방식

주로 단말기 요청 들어 오는 곳은 소방(베가웍스), 도로공사(지금inc)이다.

그 외 해양,스카니아 등이 있는데 이에 대한 차이를 설명한다.

**<소방청 베가웍스>**

먼저 , 소방은 현재 사이트 번호가 35이며, 사이트 서브 번호는 100~107 그리고 199로 이루어져 있다.



소방쪽에서 단말 인증 요청이 들어 올때마다, 단말 인증 요청 정보를 엑셀파일로 보내준다.

$HOME/source/Tools/site-key/keyDB요청파일/소방/119현장지원시스템(M119)\_단말기-KeyDB관리\_20230110.xlsx 파일로 예시

테이블이(가) 표시된 사진

자동 생성된 설명

<파일에는 단말기의 uuid(ukey)와 만료일자,센터/지역대명(device) 등의 정보가 있다.>

이를 keyadmintool의 keygen 항목에서 등록 -> DB에저장 -> DB생성

DB에 저장하면 , 상용db서버에 insert 되며,

DB생성을 클릭하면 output 파일로 35.fireagency-4\_0\_{datetime}.db가 생성된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이 파일에는 현재 등록한 단말기 외에 이전에도 등록된 단말기들 정보까지 누적된 인증키들이 오름차순으로 나열되어 있다.

생성되는 원리의 코드를 분석 해보자.

nodePG/api.js의 265번째 줄 혹은 mgl\_make\_authkey.cpp부분 258번째 줄 참고

app.post('/authex/:site/:site\_sub/:ukey', (req,res)=>{

const authex = req.body;

let insertQuery = `insert into sitekey.fatos\_authex(site,site\_sub,ukey,apikey,basickey,serialkey,created,expire,device)\

values(${req.params.site},${req.params.site\_sub},'${req.params.ukey}',md5(concat(${req.params.site},'\_',${req.params.site\_sub})),\

md5(concat('${req.params.ukey}','\_',${req.params.site})),md5(concat('${req.params.ukey}','\_',${req.params.site})),\

(select current\_date),\

(select to\_date(expiredate,'YYYY-MM-DD') expiredate from sitekey.fatos\_service where site = ${req.params.site} and site\_sub = ${req.params.site\_sub}), 0)`

client.query(insertQuery,(err,result)=>{

if(!err){

res.send('성공!!!!')

}

else{console.log(err.message)}

})

client.end;

})//fatos\_authex 데이터 삽입

Ukey를 가지고, apikey,basickey,serialkey를 만드는데,

Apikey는 공통적으로 사이트번호에 언더바 그리고 사이트서브 번호를 붙여서 만들어지고 이를 md5 함수를 적용.

md5(concat(${req.params.site},'\_',${req.params.site\_sub})),\

Basickey는 ukey+언더바(\_)+사이트번호를 붙여서 만들어지고 이를 md5 함수를 적용 하는 원리이다.

md5(concat('${req.params.ukey}','\_',${req.params.site}))

Serialkey도 basickey와 같은게 가장 중요한 특징이다.

md5(concat('${req.params.ukey}','\_',${req.params.site}))

고객사에 SerialKey를 배포하는게 핵심이다.

Uuid를 입력을 다하면 DB생성하기에서 아래와 같은 코드가 실행된다.

app.get('/getauthex2/makedb/:site/:site\_sub',(req,res)=>{

client.query(`SELECT DISTINCT on (SERIALKEY) SERIALKEY FROM sitekey.fatos\_authex WHERE (SITE = 1 AND SITE\_SUB = 0) OR (SITE = ${req.params.site} AND SITE\_SUB >= 100 AND EXPIRE >= CURRENT\_TIMESTAMP) ORDER BY SERIALKEY ASC`, (err,result)=>{

if(!err){

res.send(result.rows);

}

});

client.end;

})//fatos\_authex db 생성 쿼리 (소방청 전용)

기본적으로 사이트1,사이트서브0(파토스전용masterkey)과 사이트는 35 사이터서브번호는 100부터 199까지의 모든 시리얼 키를 모아 오름차순으로 나열하는 과정임을 알 수 있다.

파일이 완성되면 ,

txt파일은 /$HOME/source/Tools/site-key/00.input에 저장.

Db파일은 /$HOME/source/Tools/site-key/01.output에 저장.

dump파일은 /$HOME/source/Tools/site-key/02.backup에 저장.

고객사 요청 파일은 /$HOME/source/Tools/site-key/ keyDB요청파일 에 저장.

그리고 만약을 위해 백업본을 따야한다

pg\_dump를 활용한다.

Ex) 터미널에서

pg\_dump --dbname=postgresql://master:masterpass11@service.fatos.biz:15432/fatosdb --schema=sitekey -f /$HOME/source/Tools/site-key/02.backup/dump-sitekey-20230113.sql

입력하면,

텍스트이(가) 표시된 사진

자동 생성된 설명

키 값들이 .sql 파일 형태로 저장되어 있다.

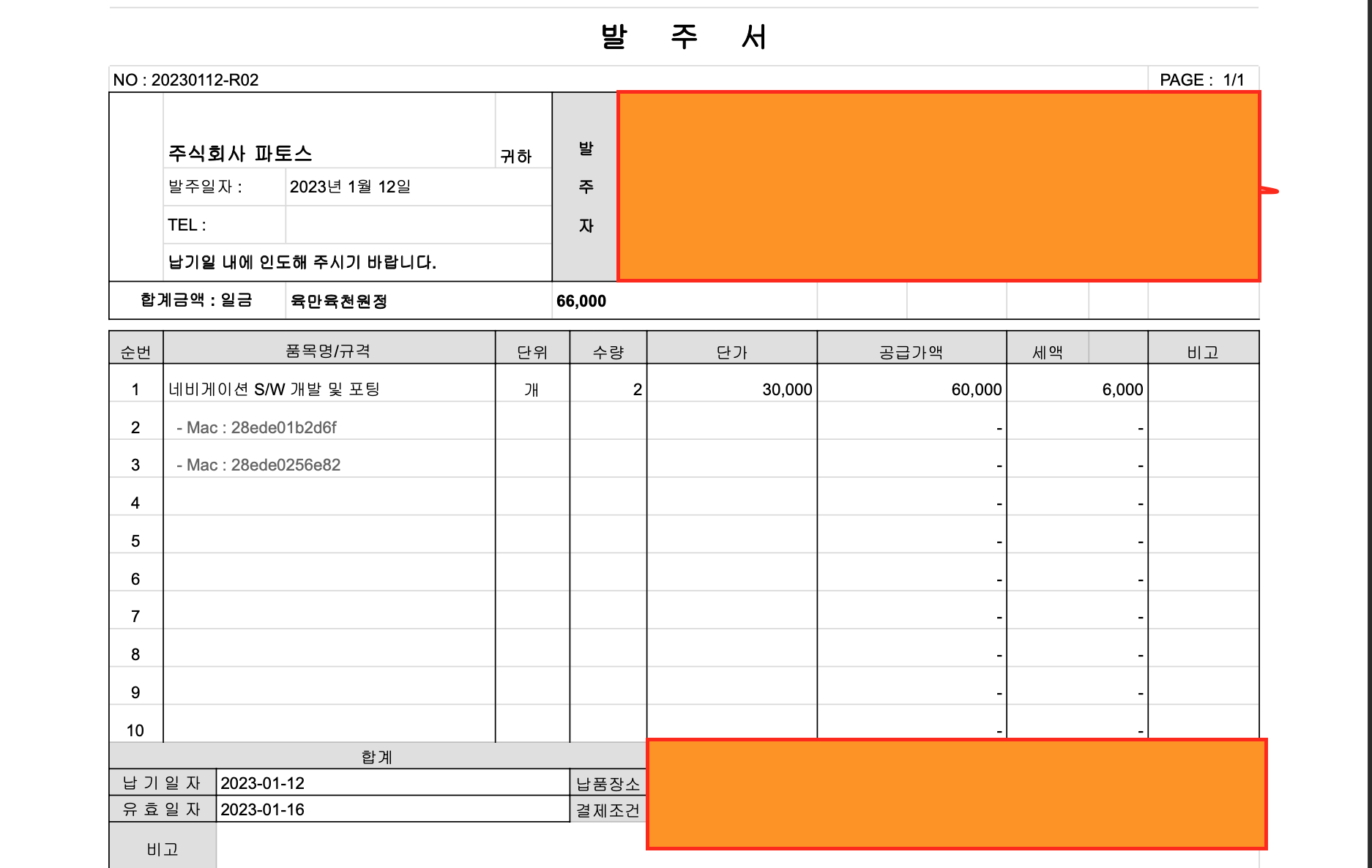
끝으로 고객사 베가웍스 담당자 (서건희 대리([sluckhy17@naver.com)](mailto:sluckhy17@naver.com)))에게

db파일을 직접 첨부해서 보내면 된다.

**<도로공사>**

도로공사는 사이트 번호 23. 사이트 서브 번호 2이다.

지금아이앤씨에서 단말 인증 요청이 들어 올때마다, 단말 인증 요청 정보를 pdf형태로 보내준다.



<요청 발주서 예시>

Keydb 등록 단계 전까지는 소방이랑 다를 바 없지만,

생성 알고리즘과 output 파일은 크게 다르다. 3가지 특징이 있음

이에 대해 반드시 숙지할 것!

먼저 생성 알고리즘은 apikey와 basickey까지 생성 알고리즘은 소방과 같다.

특징 첫번째 ) 문제는 serialkey가 다른 포맷과 다르다.

app.post('/authex2/:site/:site\_sub/:ukey/:device', (req,res)=>{

const authex = req.body;

let insertQuery = `insert into sitekey.fatos\_authex(site,site\_sub,ukey,apikey,basickey,serialkey,created,expire,device)\

values(${req.params.site},${req.params.site\_sub},'${req.params.ukey}',md5(concat(${req.params.site},'\_',${req.params.site\_sub})),\

md5(concat('${req.params.ukey}','\_',${req.params.site})),md5(concat('${req.params.ukey}','\_fatosauto')),\

(select current\_date),\

(select expiredate from sitekey.fatos\_service where site = ${req.params.site} and site\_sub = ${req.params.site\_sub}), '${req.params.device}')`

client.query(insertQuery,(err,result)=>{

if(!err){

res.send('성공!!!!')

}

else{console.log(err.message)}

})

client.end;

코드 중간 부분에 이 부분에서 차이가 난다.

md5(concat('${req.params.ukey}','\_fatosauto')),\

Mac(Ukey)에 언더바 그리고 ‘fatosauto’라는 문구와 합쳐서 md5 해시함수를 적용하기에 , 소방과 큰 차이점 이다.

특징 두번째) output 파일 형태도 다르다.

소방이나 기타 사이트들은 누적된 serialkey들을 오름차순 나열하는 방식으로 생성되지만,

도로공사는 특정 Json 형식의 틀을 생성한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

이에 대한 코드는 key\_admin\_tool의 keygen.dart에 도공버전부분에 따로 코딩되어있다. 191번째줄 ,1203번째 줄에 특히 주목.

특징 세번째) 다른 사이트들과는 달리 누적 된 ukey를 제공하지 않는다. 당일 건에 대해서만 Json에 저장함

app.get('/getauthex2/makedb3/:site/:site\_sub',(req,res)=>{

client.query(`SELECT ukey as mac, SERIALKEY FROM sitekey.fatos\_authex WHERE SITE = ${req.params.site} AND SITE\_SUB = ${req.params.site\_sub} AND created = current\_date ORDER BY SERIALKEY ASC`, (err,result)=>{

if(!err){

res.send(result.rows);

}

});

client.end;

})//fatos\_authex db 생성 쿼리 (도로공사)

Where절에서 created(생성날짜)를 오늘 기준으로만 생성 된 내용만 불러오게 해놓았기 때문이다.

이런 과정으로 .db을 생성하였다면,

소방과 같이 적절한 곳에 생성파일들을 알맞은 경로에 저장

Db파일 전송은 고객사에 드리는게 아닌

황준섭 그룹장님(jshwang@mogoscorp.com)에게 사전에 메일로 공유 해주면 된다.

**<그 외 사이트들>**

소방과는 크게 차이는 없지만,

다른 점이 하나 있다면,

app.get('/getauthex2/makedb2/:site/:site\_sub',(req,res)=>{

client.query(`SELECT DISTINCT on (SERIALKEY) SERIALKEY FROM sitekey.fatos\_authex WHERE (SITE = 1 AND SITE\_SUB = 0) OR (SITE = ${req.params.site} AND SITE\_SUB < 100 AND EXPIRE >= CURRENT\_TIMESTAMP) ORDER BY SERIALKEY ASC`, (err,result)=>{

if(!err){

res.send(result.rows);

}

});

client.end;

})

생성 하는 쿼리문에서, site\_sub <100 이라 되어 있는데,

소방이 100이상의 site\_sub의 serialkey를 모두 모으지만 , 기타 사이트들은 100미만의 모든 site\_sub만 취급하게 끔 설정되어있다.

즉 ,소방 외에 기타 사이트들의 site\_sub 100이상은 유효하지않다.

이 점만 명심하면 소방과는 크게 차이가 없다.

기타사이트는 현재까지 스카니아나 해양에너지 1~2건정도 있었고,

배포는 메일로 파일 전송방식이 아닌 관련 서버에 db파일을 업로드 한다.

업로드 후 webadmin 수정   
(eDBType\_scania\_Fatos, kor\_oceanenergy\_fatos\_5-3$date.db –해양에너지 서버)

**4.향후 과제**

- Key\_admin\_tool 코드가 좀 더 효율적으로 개선되야 한다.

메모리가 꽤 소모되며, 특정 버튼을 누르면 ,

바로바로 반응하는데 문제가 다소 있음.

-사이트와 사이트서브번호를 선택하는 dropdownvalue 반응 느림.

=>조치 : 브라우저의 사이즈를 마우스로 축소 및 확대를 하면, 활성화가 됨

(이는 서버 로딩의 문제가 아닌 flutter\_Web의 한계점이라고 판단)

-flutter web에는 path\_provider 패키지 미지원.(모바일은 가능)

해서, 파일 읽고 쓰는데에는 현재 아스키코드로 읽고 쓰기와 같은 원시적인 방법을 쓸 수 밖에 없었고, 파일 저장 같은건 당연히 수행하기 어려워, default 경로인

다운로드 디렉토리에서 파일을 Output 할 수 있다.

-키 검증 알고리즘 확장 필요.

현재는 단순히 사이트없음,기간만료 등에 대해서만 검증의 판단이 됨.

- 누군가가 이 웹앱을 사용하면 사용 로그를 추적할 수 있는 기능 마련 필요

-정식으로 도메인 발급 받을 필요성 제기