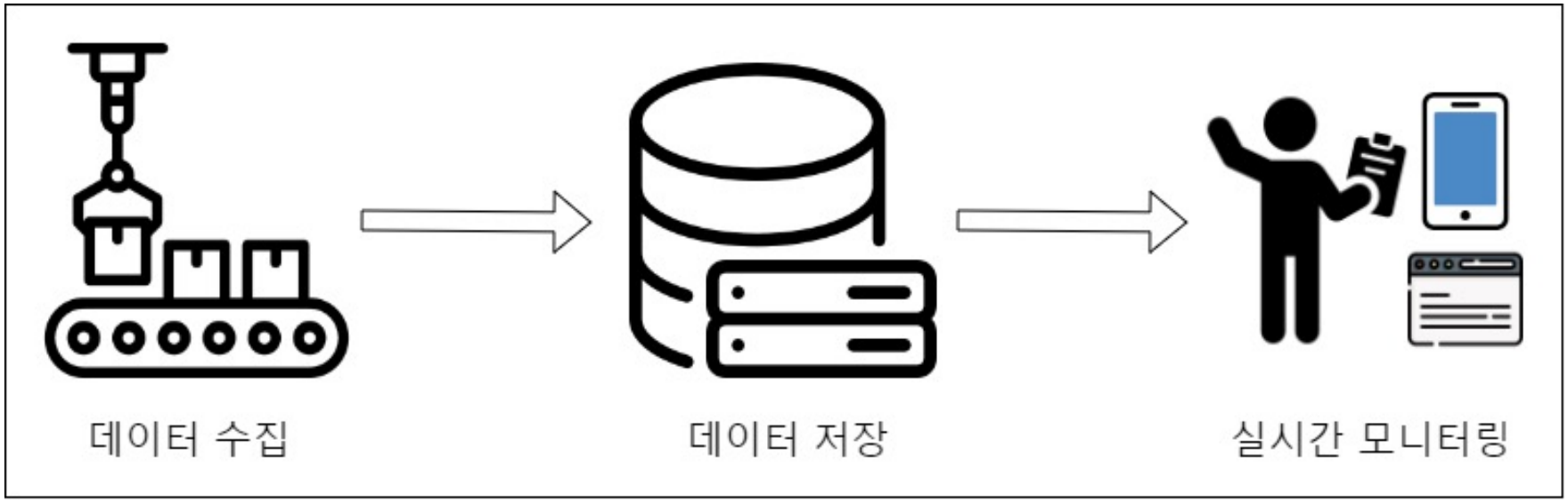
Smart Factory

Part . Server

91714806 허정운

: Intro



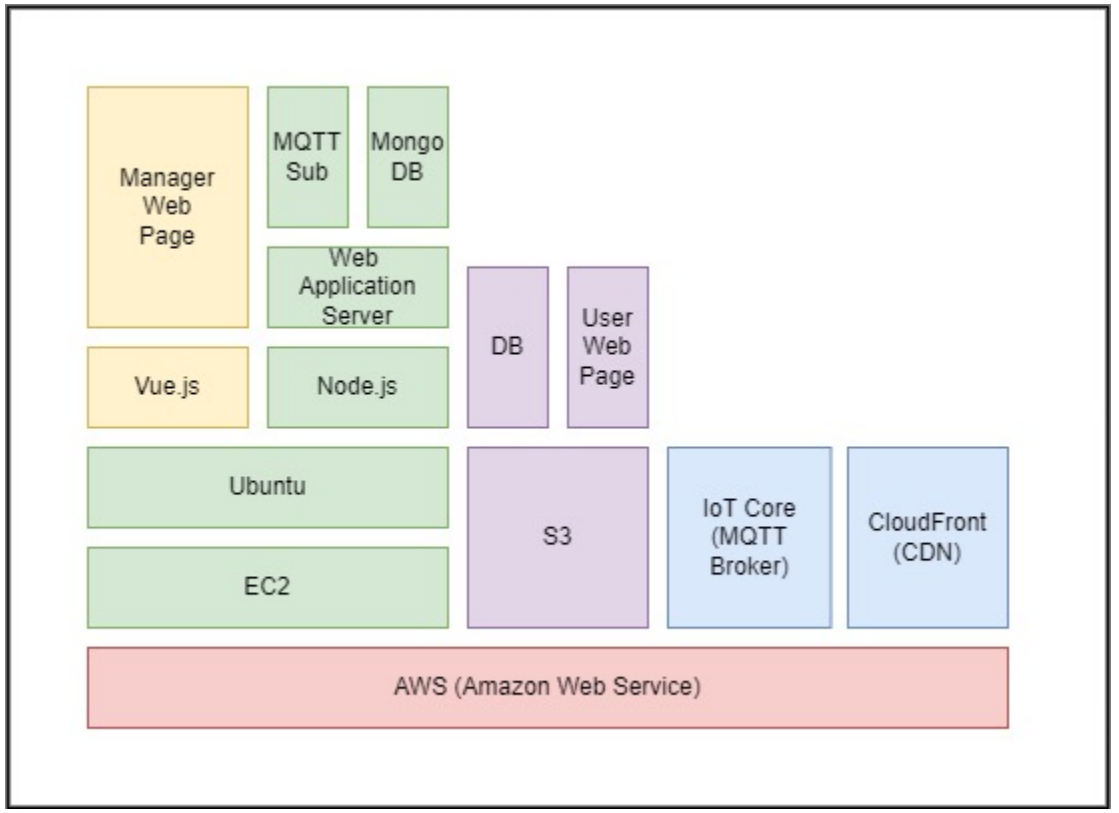
: Architecture & Design

1) EC2 Instance (Ubuntu/Node.js)
- Koa HTTP Server
- Mongo DB
- MQTT Subscribe
- Management Web Server(vue.js)

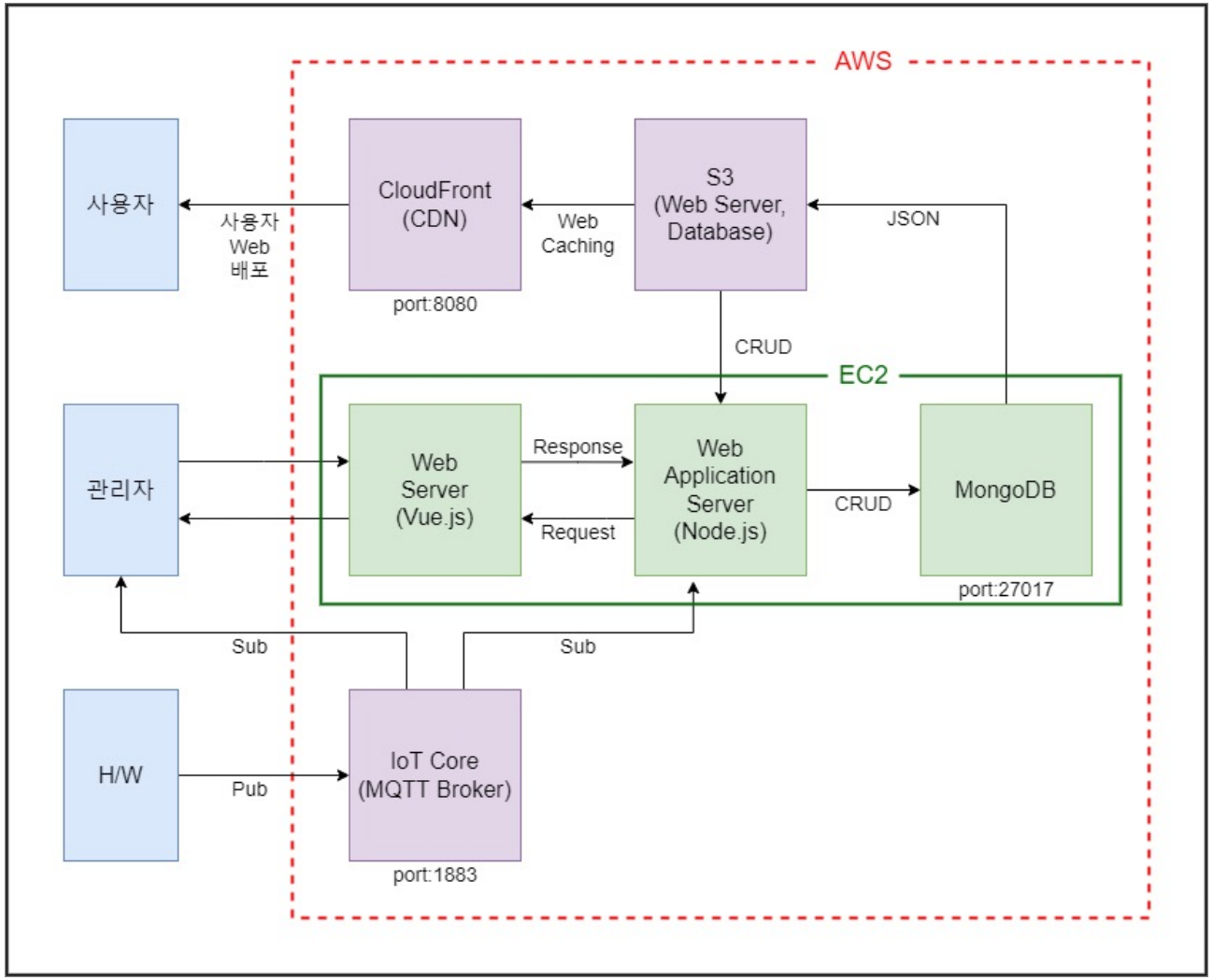
2) IoT Core
- MQTT Broker Server

3) S3 (Storage)
- Database
- User Web Page

4) CloudFront (CDN)



: Lecture


AWS
사용자
사용자
Web
배포
CloudFront
(CDN)
port:8080
Web
Caching
S3
(Web Server,
Database)
JSON
CRUD
EC2
관리자
Web
Server
(Vue.js)
Response
Request
Web
Application
Server
(Node.js)
CRUD
MongoDB
port:27017
Sub
Sub
H/W
Pub
IoT Core
(MQTT Broker)
port:1883

: Test List

분류	내용	결과
데이터 저장	1) 데이터를 Publish에서 지정 Topic으로 JSON 형태로 메시지 발행 2) 해당 Topic을 IoT Core에서 지원하는 Broker Server를 통해 Subscribe하고 있는 WAS에서 메시지를 수령 3) S3에 저장	데이터가 지정된 테이블에 입 력됐는지 상호 비교
데이터의 호출 및 표시	1) 데이터를 Web에서 get 요청 2) DB가 담긴 S3로부터 JSON 형태로 호출 3) Web으로 return	호출한 값과 Web에 표시된 데이터가 일치하는지 확인
회원가입	1) client가 회원가입 요청 시 데이터를 post로 수령 2) 데이터를 쿼리문을 통해 DB에 insert 3-1) 일치하는 데이터가 없을 경우, 성공 메시지를 return 3-2) key 혹은 id가 중복될 경우, 에러 메시지를 return	web에서 입력한 데이터, 서버에서 받은 데이터, DB에 저장된 데이터가 일치하는지 확인
로그인	1) client가 로그인 요청 시 web에서 get 요청 2) 일치하는 데이터가 있는지 쿼리문을 통해 탐색 3-1) 일치하는 데이터가 있는 경우, 성공 메시지를 return 3-2) 일치하는 데이터가 없는 경우, 에러 메시지를 return 3-3) 비밀번호가 다를 경우, 에러 메시지를 return	요청한 아이디로 로그인 되는지 확인
MQTT	1) MQTT Broker(IoT Core)에서 발행되는 모든 메시지를 저장 2) Publish에서 지정 Topic으로 발행되는 메시지를 확인 3) Subscribe에서 메시지가 정상적으로 받아지는지 확인	Pub과 Sub, Broker를 확인하여 데이터가 모두 일치하는지 확인

: Problems

1) 높은 트래픽을 고려하지 않은 서버 구조

2) 보안을 고려하지 않은 설계

Smart Factory

Part . Server

감사합니다.

91714806 허정운