JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 24, No. 12, pp. 3121-3130, Dec. 2023

Check for updates

# AWS Greengrass 와 SageMaker를 활용한 1인가구 돌봄 IoT 서비스 개발

최 환 석[1] · 장 승 호[2] · 천 효 림[2] · 김 민 정[2] · 이 우 섭[3*]
[1]카이스트-메가존클라우드 지능형 클라우드 융합기술 연구센터 매니저 [2]국립한밭대학교 정보통신공학과 학사
[3*]국립한밭대학교 지능미디어공학과 교수

# Development of a Single-Person Care IoT service using AWS Greengrass and SageMaker

**Hoan-Suk Choi[1] · Seung-Ho Jang[2] · Hyo-Rim Cheon[2] · Min-Jeong Kim[2] · Woo-Seop Rhee[3*]**

[1]Manager, KAIST-Megazone Cloud Intelligent Cloud Computing Convergence Research Center, Daejeon 34141, Korea
[2]Bachelor's degree, Department of Information Communication Engineering, Hanbat National University, Daejeon 34158, Korea
[3*]Professor, Department of Intelligent Media Engineering, Hanbat National University, Daejeon 34158, Korea

**[요 약]**

1인 가구의 경우 범죄 및 질병 등 비상시 대처 능력이 떨어지고 외출시 원격 케어에 대한 요구(집안 컨디션 케어, 반려동물 대응 등)가 큰 편이다. 본 논문은 AWS 클라우드의 머신러닝 서비스인 SageMaker와 IoT 플랫폼인 Greengrass를 활용하여, 1인 가구 돌봄 IoT 서비스를 제안한다. 제안하는 서비스 제공을 위해 카메라와 각종 센서를 설치한 라즈베리 파이 노드를 설치하였으며, 집안 곳곳을 자유롭게 이동하며 상황을 전달할 수 있는 지능형 이동체를 활용한다. AWS IoT Greengrass를 기반으로 IoT 디바이스를 등록하고, 관리하며, 클라우드 플랫폼과 연동한다. 또한 SageMaker를 기반으로 사용자 인식을 위한 모델을 학습/배포하여 효과적인 지능형 서비스를 제공한다.

**[Abstract]**

Single-person households face increased vulnerability to emergencies such as crime and illness, emphasizing the necessity for remote care solutions, including home and pet care. This paper introduces a novel approach to single-person care IoT services, leveraging AWS cloud-based machine-learning service, SageMaker, and its IoT platform, Greengrass. The proposed service involves the deployment of Raspberry Pi nodes equipped with cameras and various sensors, alongside intelligent mobile devices capable of dynamic movement within the household for video delivery. The system efficiently registers, manages, and interfaces with cloud platforms through AWS IoT Greengrass. Furthermore, it enhances its capabilities by training and distributing models for user recognition based on SageMaker, delivering intelligent services tailored to individual needs.

**색인어 :** 클라우드, 머신러닝, IoT 서비스, 1인 가구 돌봄, AWS
**Keyword :** Cloud, Machine Learning, IoT Service, Single-Person Care, AWS

**http://dx.doi.org/10.9728/dcs.2023.24.12.3121**



**Received** 09 November 2023; **Revised** 14 December 2023
**Accepted** 22 December 2023

***Corresponding Author; Woo-Seop Rhee**

**Tel:** +82-42-821-1749
**E-mail:** wsrhee@hanbat.ac.kr

## Ⅰ. 서 론

초창기 IoT 서비스는 수집된 데이터를 클라우드에서 저장 및 처리하여 결과를 전달하는 형태로 개발되었다. 그러나, 연결된 디바이스가 많아지고, 지연에 민감한 서비스가 등장하면서, 사용자 및 IoT 디바이스에 가까운 위치 즉, 에지(Edge)에서 데이터를 분석 및 처리하는 방법이 요구되고 있다[1]. 에지 컴퓨팅은 IoT 데이터의 발생 위치와 가까운 곳에서 실시간으로 대용량 데이터를 처리하여 지연을 최소화하고, 네트워크에 모든 데이터를 전달하지 않으므로 보안상 장점이 있다. 또한 대용량 데이터의 신속한 처리에 유리하여 효과적인 AI/ML 기반의 지능형 IoT 서비스 제공이 가능하다[1].

본 논문에서는 클라우드 기반 지능형 IoT (Internet of Things) 서비스 제공을 위해 AWS (Amazon Web Service) Greengrass를 활용하여 IoT 디바이스를 등록, 관리하고, 클라우드 기반 기계학습 기술인 AWS SageMaker를 활용하여 클라우드 기반 머신러닝을 활용한 1인 가구를 위한 지능형 IoT 서비스를 개발하였다.

제안하는 서비스는 댁내의 다수 디바이스를 통해 다양한 지능형 서비스를 제공한다. 실내 온도, 습도, 미세먼지 등 공기질을 모니터링하는 실내 케어 기능을 제공하며, 사용자와 상호작용하는 지능형 이동 디바이스를 배치하여 실내공간을 자유롭게 이동하면서 곳곳의 실시간 영상을 제공한다. 또한 AWS SageMaker를 기반으로 사용자 인식 모델을 학습하여 사용자가 손쉽게 인식할 대상을 등록할 수 있으며, 인식을 위한 모델의 훈련 배포가 용이하다. 만약 허용하지 않는 사람이 등장했을 경우 이미지를 저장하고, 인식 결과를 AWS QLDB (Quantum Ledger Database)에 저장하여 위변조를 막고 신뢰성을 확보한다.

이를 위해 본 논문은 2장에서 서비스 개발에 활용한 클라우드 기반 머신러닝 기술을 소개하고, 3장에서 제안하는 시스템 구조 및 구현 환경을 설명한다. 4장에서는 주요 기능별 프로시저와 구현 결과를 제시하고 5장에서 결론을 맺는다.

## Ⅱ. IoT 서비스를 위한 클라우드 기반 머신러닝 기술

### 2-1 클라우드 기반 IoT 플랫폼

대표적인 클라우드 기반 IoT 플랫폼인 AWS IoT Greengrass는 IoT 애플리케이션 구축, 배포 및 관리 기능을 제공하는 클라우드 서비스이다. 사용자는 AWS IoT Greengrass를 사용하여 학습된 기계학습 모델을 배포하고, 디바이스가 생성하는 데이터를 통해 로컬에서 추론을 처리하며, 디바이스 데이터를 필터링 및 관리할 수 있다. 또한 로컬 네트워크의 다른 디바이스 뿐 아니라 AWS IoT 코어(Core)와 안전한 통신수단을 제공하여, IoT 데이터를 안전하게 클라우드로 전달할 수 있다. AWS IoT Greengrass는 구성 요소(Component)라고 하는 소프트웨어 모듈을 기반으로 에지 애플리케이션을 구축할 수 있으며 에지 디바이스를 AWS 서비스 또는 타사 서비스와 쉽게 연결할 수 있다[2].

### 2-2 클라우드 기반 기계학습 서비스

AWS SageMaker는 완전 관리형 기계학습 서비스로 머신러닝 모델을 쉽고 빠르게 구축하고 학습시킬 수 있으며, 학습된 모델을 프로덕션 지원 호스팅 환경에 직접 배포할 수 있다. Jupyter Notebook 인스턴스를 제공하여 탐색 및 분석에 필요한 원본 데이터의 쉬운 접근을 지원하며 분산된 환경 내 대규모 데이터를 효율적으로 실행하는 데 최적화된 다양한 머신러닝 알고리즘들을 제공한다. 또한 특정 워크플로우에 맞게 조정되는 유연한 분산형 학습 기능을 제공하며 클릭 몇 번으로 모델을 학습하고 안전하고 확장 가능한 모델 배포를 지원한다[3].

## Ⅲ. 제안하는 클라우드 기반 1인가구 IoT 서비스

그림 1은 본 논문에서 제안하는 지능형 IoT 서비스를 제공하기 위한 시스템 구성도이다. 지능형 서비스를 제공하기 위해 두 가지 타입의 IoT 디바이스를 활용한다. Greengrass 코어 디바이스는 데이터를 수집하고 클라우드에서 학습된 모델을 기반으로 지능형 서비스 제공을 위한 추론을 실행한다. Greengrass 클라이언트 디바이스는 다양한 위치의 데이터 수집을 위해 다수의 디바이스로 구성 가능하며, 코어 디바이스와 통신하며 데이터를 수집하고, 추론 결과에 따라 다양한 기능을 실행할 수 있다.

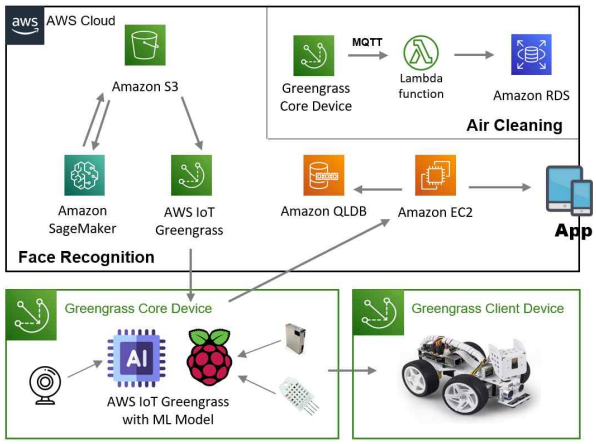



**그림 1.** 제안하는 시스템 구성도
**Fig. 1.** Proposed system overview

IoT 디바이스에서 수집된 데이터는 AWS 클라우드로 전송된다. 수집된 데이터는 AWS의 서버리스 솔루션인 람다

(Lambda)[4]를 통해 비동기적으로 처리된다. 본 시스템에서는 IoT 디바이스에서 데이터가 수집되면 미리 정의된 람다 함수에 의해 AWS 데이터베이스 솔루션인 RDS[5]에 수집된 데이터를 저장한다. 지능형 서비스 제공을 위해 AWS SageMaker로 특정 추론을 위한 모델을 생성/학습하고, 학습된 모델은 추론을 수행할 Greengrass 코어 디바이스로 배포된다. 또한 제안하는 시스템은 타겟 이벤트의 영속적인 저장을 위해 AWS QLDB를 활용한다. QLDB는 완전 관리형 원장 데이터베이스로, 중앙의 신뢰 기관이 소유하는 투명하고, 변경 불가능하며, 암호화 방식으로 검증 가능한 트랜잭션 로그를 제공한다[6]. 이를 기반으로 원장의 데이터 변경 내용을 추적하며 완전하고 검증 가능한 방법으로 기록을 유지할 수 있다[7].

### 3-1 하드웨어 개발 환경

제안하는 시스템의 Greengrass 코어 디바이스는 라즈베리파이 (Raspberry Pi) 4B[8]를 사용하였고, 영상 데이터 수집을 위한 Intel RealSense F455 카메라 모듈[9], 공기질 데이터 수집을 위해 미세먼지 센서 PMS7003과 온습도 센서 DHT22를 설치하였다.

제안하는 시스템은 다양한 공간 정보를 수집하기 위해 그림 2와 같이 Greengrass 클라이언트 디바이스로 구동되는 지능형 이동체를 구현하였다. 지능형 이동체는 컨트롤러로 라즈베리파이 4B가 장착되어 있으며 구조체의 머리 부분에 pi 카메라[10]를 설치하여 영상 데이터를 수집할 수 있도록 하였다. 4개의 바퀴에는 모터가 설치되어 사용자의 제어에 따라 원하는 곳으로 이동하며 스피커를 장착하여 경고음, 음성 출력 등을 지원할 수 있게 구성하였다.


**그림 2.** Greengrass 코어 디바이스[11]와 클라이언트 디바이스
**Fig. 2.** Greengrass core device and client device

### 3-2 웹앱 구현 환경

제안하는 시스템의 사용자 인터페이스는 그림 3의 개발환경을 기반으로 웹앱으로 구현하다. 프론트엔드는 Vue.js 로 구현하였으며, 백엔드는 Python-Flask, 웹 스크래핑을 위해 BeautifulSoup4[12]를 활용하였다. 데이터베이스는 AWS의 RDS를 이용하였고 MariaDB 엔진을 활용한다.

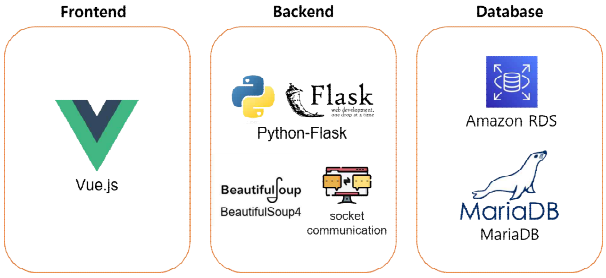



**그림 3.** 웹앱 구현 환경
**Fig. 3.** Implementation environment of Web App

## Ⅳ. 지능형 IoT 서비스 설계 및 구현

본 논문에서 제한하는 지능형 IoT 서비스는 1인 가구를 위한 서비스로 온도, 습도, 미세먼지 등 공기질을 모니터링 하는 실내 케어 기능과 더불어 사용자와 상호작용하는 지능형 이동체를 통해 댁내 공간을 자유롭게 이동하면서 집안 곳곳의 실시간 영상을 제공한다. 또한 AWS SageMaker를 기반으로 사용자의 얼굴 모델 생성/학습하여 집안에 들어오는 사람의 얼굴을 인식하여, 등록하지 않은 사람이 침입했을 경우 이를 인식하여 침입자의 이미지를 QLDB에 저장하여 위변조를 막고 신뢰성을 확보하여 침입의 증거로 활용하도록 한다.

### 4-1 실내 케어 서비스

#### 1) 실내 케어 서비스 프로시저

그림 4는 실내 공기질을 모니터링하기 위한 실내 케어 서비스의 프로시저이다. 사용자는 웹앱에서 실내 케어 기능을 선택하는 즉시 온습도와 미세먼지 데이터 측정 요청을 백엔드(WAS:Web Application Server)에 전달한다. WAS는 코어 디바이스와 소켓 통신을 통해 연결되고 온습도 요청 메시지를 전달받은 코어 디바이스는 센서 값을 측정해 배포(Publish)한다. 이후 측정 완료 메시지를 반환하고, 동시에 IoT 코어에 측정한 센서값을 전달한다. AWS IoT 코어가 센서값을 받으면 이것이 람다 함수를 호출하는데, 이 람다 함수는 전달받은 센서값을 완전 관리형 데이터베이스 솔루션인 RDS에 삽입하는 코드를 수행한다. 결과적으로 사용자 요청

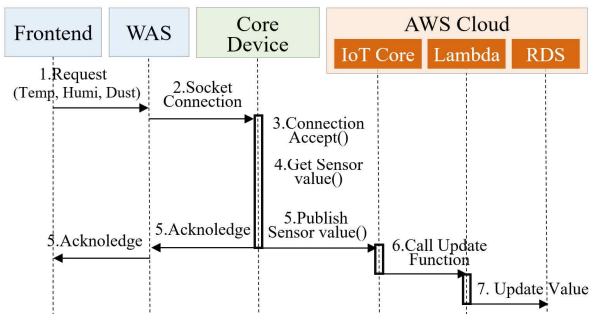



**그림 4.** 실내 케어 서비스 프로시저
**Fig. 4.** Indoor care service procedure

을 통해 코어 디바이스의 센서값이 RDS에 전달되며, 사용자는 측정된 온습도, 미세먼지 농도, 측정 시간을 UI에서 확인할 수 있다.

#### 2) 실내 케어 데이터 저장

본 시스템에서는 센서 데이터 업데이트를 위해 비동기 컴퓨팅 서비스인 AWS 람다를 활용하였다. 람다 함수를 생성하고, 해당 함수가 실행되는 조건인 '규칙'을 정의한다. 여기서는 이동체에 전달되는 모든 값이 변경되는 경우, 실행되어야 하므로, 규칙 속성에서 SQL의 모든 속성값을 정의하였다. 다음 실행할 타겟 람다 함수를 선택하면 된다.

다음으로 람다 함수 트리거를 위한 구성이 필요하다. AWS IoT의 Custom IoT rule에 이전에 구성한 규칙 속성의 ARN을 입력하여 해당 람다의 호출 설정을 완료한다.

마지막으로 실제 람다가 호출되었을 때 수행해야 할 내용을 정의하기 위해, 그림 5와 같은 센서데이터 업로드 코드를 람다 함수에 입력한다. 구현시에는 RDS에 센서데이터를 삽입하는 코드를 작성하였으며 이를 통해 RDS에 온도, 습도, 미세먼지 농도, 측정 시간을 삽입한다.

```
Procedure onDataReceived (event : SensorDevice )
  // Extract sensor data from event parameter
  temp := event.getTemperature( )
  humi := event.getHumidity( )
  PMS := event.getPMData( )
  // Get current time
  currentTime := getCurrentTime( )
  // Connect to AWS database
  awsConnection := connectToAWSDatabase( )
  // Upload data to AWS database
  uploadDataToAWSDatabase ( awsConnection, temp, humi, PMS, currentTime )
End Procedure
```

**그림 5.** 센서 데이터 업로드 슈도코드
**Fig. 5.** Sensor Data Upload Pseudo Code

| | idDHT_data | temp | humi | pms | nowtime |
|---|---|---|---|---|---|
| | 46 | value1 | value2 | value3 | value4 |
| | 47 | value1 | value2 | value3 | value4 |
| | 48 | 25.9*C | 64.1% | %dum | 2022-11-11 22:27:27 |
| | 49 | 25.9*C | 64.2% | 20um | 2022-11-11 22:33:42 |
| | 50 | 26.1*C | 63.9% | 20um | 2022-11-11 22:38:58 |
| | 51 | 25.9*C | 64.5% | 24um | 2022-11-11 22:52:25 |
| | 52 | 26.0*C | 64.4% | 20um | 2022-11-11 22:52:35 |
| | 53 | 26.0*C | 64.4% | 18um | 2022-11-11 22:53:10 |
| | 54 | 26.0*C | 64.4% | 20um | 2022-11-11 22:59:33 |
| | 55 | 26.0*C | 64.5% | 19um | 2022-11-11 23:06:36 |
| | 56 | 26.1*C | 64.2% | 22um | 2022-11-11 23:13:25 |
| | 57 | 26.0*C | 64.5% | 21um | 2022-11-11 23:13:33 |

**그림 6.** 센서 데이터 수집 결과 (RDS)
**Fig. 6.** Sensor data collection result

그림 6은 위에서 정의한 람다 함수를 통해 RDS의 MariaDB에 데이터 업로드한 결과를 보인다. idDHT_data가 46, 47인 데이터는 AWS 람다 함수를 테스트 값이고 이후에 실제 데이터가 쌓여가는 것을 확인할 수 있다.

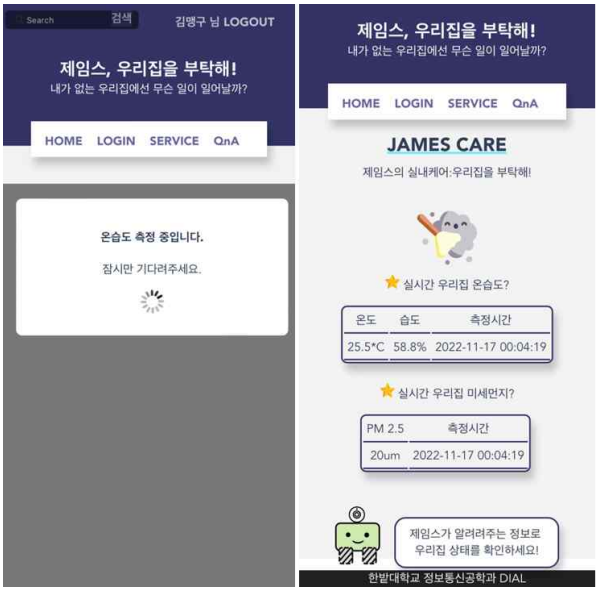
#### 3) 실내 센싱 데이터 요청

데이터 센싱 및 전달을 위해 백엔드와 코어 디바이스는 소켓을 설정하여 센서값 요청 및 전달을 처리한다. 백엔드는 센서값이 필요할 때 연결된 소켓을 통해 센서 측정 요청 메시지를 전송하며, 센서값이 정상적으로 수신되면 소켓을 종료하고 해당 값을 반환한다.

코어 디바이스인 라즈베리파이는 자신에게 설치된 센서값을 제공하기 위해 소켓 서버로 동작하며, 센서 데이터 요청을 받으면, GetDhtValue 함수를 실행하여 연결된 온습도 및 미세먼지 센서의 데이터를 수집하고 소켓을 통해 전달한다.

#### 4) 실내 케어 수행 결과

앞서 설명한 바와 같이 라즈베리파이를 통해 수집된 데이터는 RDS 데이터베이스에 업로드되며, 결과적으로 그림 7과 같이 사용자는 웹앱을 통해 측정된 온습도 및 미세먼지 농도를 확인할 수 있다.




* screen shot of app operation.
**그림 7.** 실내 케어 서비스 구현 결과
**Fig. 7.** Indoor care service implementation result

### 4-2 환경 모니터링을 위한 이동형 홈 CCTV 서비스

환경 모니터링 기능은 사용자가 지능형 이동체를 조작하여 원하는 장소로 이동시키고, 설치된 카메라를 통해 실시간으로 영상을 제공한다.

### 1) 홈 CCTV 서비스 프로시저

그림 8은 지능형 이동체 제어 프로시저이다. 사용자가 지능형 이동체의 이동을 위해 웹앱의 특정 버튼을 누르면 소켓 통신을 통해 엣지 디바이스에 이동 방향이 전달되며, 이를 수신한 엣지 디바이스(지능형 이동체)는 적합한 액추에이터를 동작 시킨다.

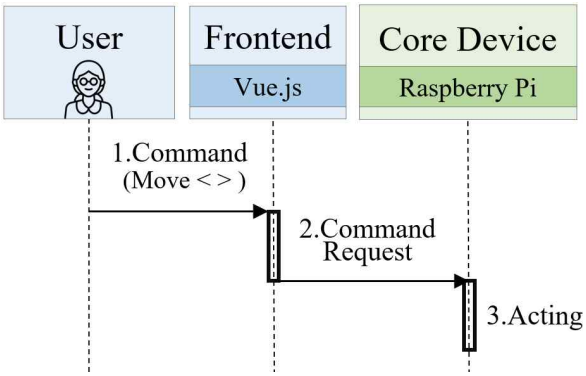



**그림 8.** 지능형 이동체 제어 절차
**Fig. 8.** Intelligent moving device control procedure

그림 9는 이동체 카메라의 영상 수신 프로시저이다. 사용자 웹앱을 통해 영상을 요청하면, 백엔드에 영상 수신을 위한 함수가 호출되어 코어 디바이스에게 mjpeg-streamer[13] 서버 실행 요청을 보낸다. 코어 디바이스는 영상 송출을 위한 서버를 실행하고, 영상 수신을 위한 URL을 프론트엔드로 전달하며, 프론트엔드에서 영상을 재생한다.

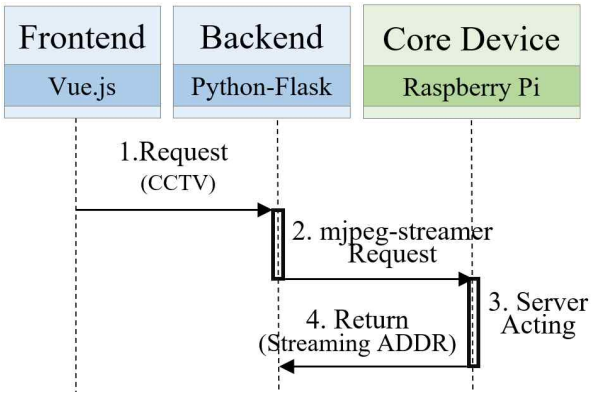



**그림 9.** 이동체 영상 수신 절차
**Fig. 9.** Video receiving procedure of moving device

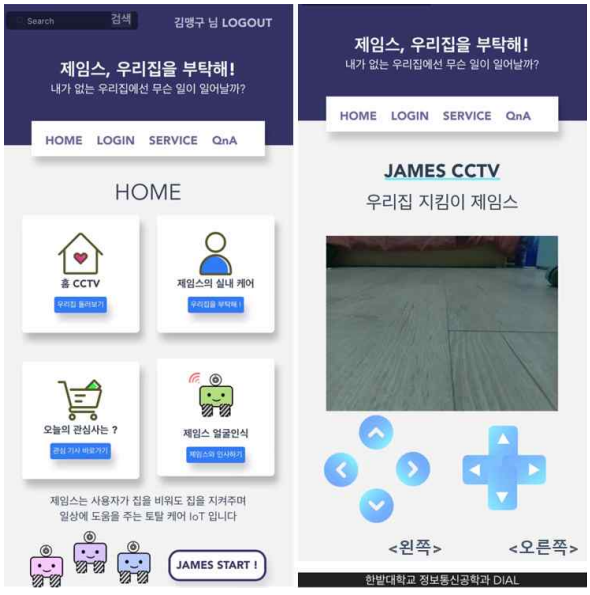
### 2) 지능형 이동체 제어

본 논문에서 제안된 지능형 이동체는 IoT Greengrass 클라이언트 디바이스로 구성하였다. 지능형 이동체의 이동(전/후진, 좌/우 회전)과 카메라 각도 조절을 위해 총 8개의 이동 함수를 구성하였다. 이를 통해 사용자는 지능형 이동체를 집 안의 원하는 장소로 이동시키고, 카메라를 상하좌우로 조작하여 다양한 각도로 집 안을 살펴볼 수 있다.

전/후진 이동함수는 호출되는 시간만큼 모터를 순/역방향으로 회전시키며, 우회전, 좌회전시에는 원하는 각도값을 입력하여 진행방향을 조정한다. 또한 카메라 각도 조정의 경우에도 각도 값을 입력받아 원하는 만큼 카메라를 조정한다. 백엔드는 그림 10과 같이 위와 같은 이동 메시지의 내용에 따라 실행할 내용을 구분한다.

```
def handle command ( command , cs , addr ) :
    if 'Go' in command :
    _thread.start_new_thread (JamesGo, (cs, addr ))
    elif 'Back' in command :
    _thread.start_new_thread (JamesBack, (cs, addr
))
    elif 'Right' in command :
    angle = int ( command.split ( " " ) [ 3 ])
    _thread.start_new_thread (JamesRight, (cs, addr,
angle))
    elif' Left' in command :
    angle = int ( command.split ( " " ) [ 3 ] )
    _thread.start_new_thread (JamesLeft, (cs, addr,
angle))
    elif' Pitch' in command :
    direction, angle = command.split ( " " ) [ 3 ], int
( command.split ( " " ) [ 4 ])
    _thread.start_new_thread (handlepitch, (cs, addr,
direction, angle )
```

**그림 10.** 지능형 이동체 제어 슈도코드
**Fig. 10.** Intelligent moving device control pseudo code




*screen shot of app operation.
**그림 11.** 홈 메뉴 및 홈 CCTV 서비스
**Fig. 11.** Home Manu and Home CCTV service

#### 3) 실시간 영상 전송 기능 구현결과

그림 11은 제안하는 서비스의 메인화면 및 실시간 영상 전송 결과를 보여준다. 사용자는 홈 화면에서 제공하는 기능을 선택할 수 있으며, 홈 CCTV 메뉴를 선택 할 경우, 오른쪽과 같이 실시간 영상을 확인할 수 있으며, 배치된 버튼을 통해 지능형 이동체를 제어할 수 있다.

### 4-3 사용자 얼굴인식 서비스

사용자 얼굴인식 기능은 AWS SageMaker를 통해 얼굴을 학습하여 등록 사용자를 인식하며, 허용되지 않은 사람이 인식될 경우 이벤트를 기록한다. 사용자 영상 획득을 위해 라즈베리파이로 구성된 AWS IoT Greengrass 코어디바이스에 Pi 카메라를 활용하며, 인식결과 전달을 위해 클라이언트 디바이스인 지능형 이동체를 활용한 복합 서비스이다.

#### 1) 사용자 얼굴인식 서비스 프로시저

사용자 얼굴인식 서비스 기능을 위해서는 먼저 인식할 사용자의 얼굴인식 모델을 생성하고 대상의 얼굴 데이터를 학습해야 한다.

그림 12는 얼굴인식 학습을 위한 프로시저이다. 얼굴인식 학습은 AWS SageMaker 오토파일럿으로 수행하였다. 얼굴인식을 위해 특정 사용자의 얼굴 사진을 AWS S3 버킷[14]에 업로드하고 이를 기반으로 모델을 튜닝한다. 모델 생성이 완료되면, S3에 모델 튜닝 정보를 저장하고 생성된 모델 중 최적의 모델을 선택한다. AWS IoT 코어에서 S3 버킷에 저장된 모델을 AWS IoT Greengrass 구성 요소로 연결하고, AWS SageMaker의 Experiments를 활용하여 코어 디바이스에 모델을 배포하면 얼굴인식 기능을 수행할 수 있다. 이와 같은 방법을 통해 코어 디바이스에 모델을 저장하거나 추론 코드를 넣지 않아도 얼굴인식 기능을 수행할 수 있다.

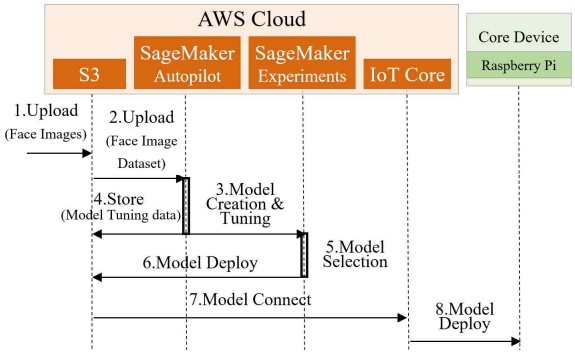



**그림 12.** 사용자 얼굴인식을 위한 모델 학습 프로시저
**Fig. 12.** Training procedure for User Face Recognize

그림 13은 학습된 얼굴인식 모델을 기반으로 실제 사용자 얼굴을 인식하는 프로시저이다. 코어 디바이스에 배포된 소켓 서버가 얼굴 인식 기능을 실행한다. 코어 디바이스에서 Pi 카메라를 통해 입력된 영상을 기반으로 얼굴인식 추론을 실행하며, 인식 결과를 클라이언트 디바이스에 전달한다. 만약 정상 사용자로 인식될 경우, 이 결과를 수신한 지능형 이동체는 사용자의 이름을 호출하는 특정 기능을 실행한다. 만약 Pi 카메라 영상이 허용되지 않은 침입자로 인식된 경우, 즉시 카메라 이미지가 캡처되고 이벤트 발생 시간과 함께 영상 파일이 AWS S3 버킷에 업로드되며 Amazon QLDB에 S3 URL을 저장한다.

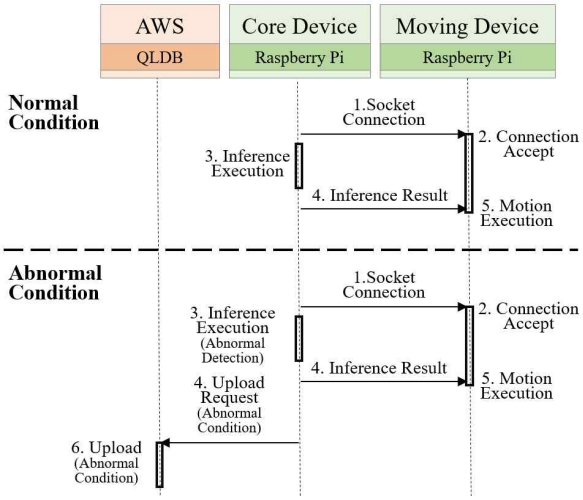



**그림 13.** 사용자 얼굴인식 프로시저
**Fig. 13.** User Face Recognize Procedure

#### 2) SageMaker 기반 얼굴인식 모델 구축

얼굴인식을 위한 모델 학습을 진행하기 위해서는 학습할 이미지 데이터가 필요하다. 먼저 OpenCV 라이브러리와 웹캠을 사용하여 영상을 촬영한 뒤, 그 영상을 프레임 단위로 캡처하여 jpg 파일로 저장한다. SageMaker 오토파이럿을 사용하기 위해서는 최소 500개의 데이터가 요구된다. 본 논문에서는 589개의 데이터를 사용했다.

다음은 얼굴인식을 위한 데이터셋을 만드는 과정으로 로컬에서 라이브러리를 설치해야 하는 번거로움을 피하기 위해 구글 Colab을 사용했다. 이미지를 80x80 픽셀 크기로 만들어 총 6400개의 feature를 생성하였고 이를 그림 14와 같이 589x6401 (1개는 label)의 CSV 데이터셋으로 저장하였다.

| | label | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | ... | 6390 | 6391 | 6392 | 6393 | 6394 | 6395 | 6396 | 6397 | 6398 | 6399 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 27 | 16 | 19 | 26 | 19 | 23 | 19 | 23 | 17 | ... | 68 | 66 | 69 | 71 | 46 | 3 | 70 | 52 | 49 | 66 |
| 1 | 1 | -122 | -117 | -118 | -117 | -116 | -113 | -111 | -98 | -100 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2 | 1 | -108 | -116 | -108 | -104 | -103 | -105 | -108 | -105 | -105 | ... | -94 | -95 | -101 | -103 | 104 | 76 | 85 | 65 | 44 | 0 |
| 3 | 1 | -108 | -101 | -102 | -103 | -104 | -102 | -99 | -102 | -108 | ... | -98 | -102 | 72 | 40 | 22 | 1 | 8 | 19 | 38 | 33 |
| 4 | 2 | 21 | 15 | 19 | 15 | 25 | 17 | 25 | 15 | 20 | ... | 2 | 56 | 52 | 56 | 68 | 40 | 22 | 11 | 19 | 15 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 572 | 2 | 60 | 51 | 35 | 45 | 55 | 62 | 59 | 56 | 30 | ... | 124 | 119 | 120 | 112 | 113 | 101 | 106 | 114 | 124 | 127 |
| 573 | 1 | -120 | -113 | -119 | -115 | -113 | -119 | -103 | -103 | -108 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 574 | 1 | -116 | -116 | -118 | -120 | -115 | -105 | -110 | -104 | -98 | ... | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 575 | 0 | -114 | -77 | -101 | -72 | -71 | -69 | -65 | -64 | -73 | ... | -69 | -69 | -68 | -75 | -81 | -84 | -101 | -114 | 124 | 54 |
| 576 | 0 | 107 | 92 | 114 | -72 | -71 | -67 | -68 | -66 | -128 | ... | -69 | -68 | -70 | -69 | -82 | -92 | -109 | -115 | 50 | 36 |

**그림 14.** Feature 생성 및 CSV 변환 결과
**Fig. 14.** Feature creation and CSV transition result

SageMaker 오토파일럿을 실행하기 위해 앞에서 생성한 CSV 파일을 Amazon S3에 업로드하고 SageMaker 콘솔에 접속한다. SageMaker studio를 통해 Jupyter studio에 접속하고 SageMaker Experiments를 생성한다.

학습을 위한 데이터셋 형식은 CSV, Parquet, Manifest 파일만 가능하며 Target으로 데이터셋의 column 중 하나를 선택한다. 다음으로 머신러닝 예측 모형을 선택할 수 있는데, auto를 선택하면 데이터를 분석하여 자동으로 예측 모형을 결정한다.

SageMaker Experiments가 생성되면 데이터 전처리 과정이 진행된다. 전처리 과정중 과대적합 또는 과소적합을 막기 위해 자동으로 데이터를 섞고 분할한다. 전처리 과정의 결과는 SageMaker Experiments 설정에서 정한 S3 출력 경로에서 확인할 수 있다.

다음으로 Candidate definition generated 과정은 데이터 학습에 적합한 모델 후보들을 생성하는 과정이며, Feature engineering은 과대적합과 데이터 예측의 편향을 방지하기 위해 차원 축소, 스케일링, 변형 과정을 진행한다. 후보 모델 생성 결과도 타겟 S3 버킷에 저장된다.

모델 튜닝 과정은 모델의 하이퍼파라미터를 조정하여 모델의 정확도 손실률을 계산한다. 하이퍼파라미터의 값에 따라 모델의 정확도와 손실률은 다양한 결과가 나오는데, 자동으로 하이퍼파라미터를 다양하게 조정하여 최상의(최저손실률) 머신러닝 모델을 쉽게 얻을 수 있다.

모델 튜닝이 끝나면 하이퍼파라미터 튜닝을 통해 얻은 모든 모델의 정보(하이퍼파라미터의 설정값) 등을 문서화 (오토파일럿 리포트)하여 S3 출력 경로에 업로드한다.

오토파일럿 과정이 모두 끝나면 그림 15와 같이 생성된 모델 후보를 확인할 수 있고, 최상의 모델을 선택할 수 있다. 모델 후보 리스트 좌측 상단에 deploy model 버튼을 클릭하면 endpoint를 생성하여 모델을 배포할 수 있다[15].

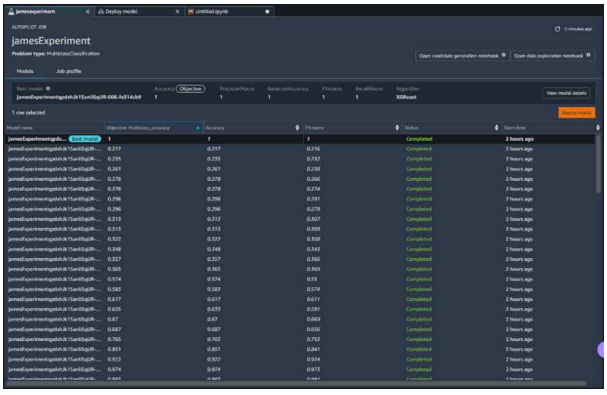



**그림 15.** SageMaker 오토파일럿 모델 훈련 완료
**Fig. 15.** Training complete of SageMaker autopilot model

엔드포인트 생성이 끝나면 모델 배포[16] 상태가 활성화 상태가 되며 그림 16과 같이 배포될 모델의 상세 정보를 확인할 수 있다.

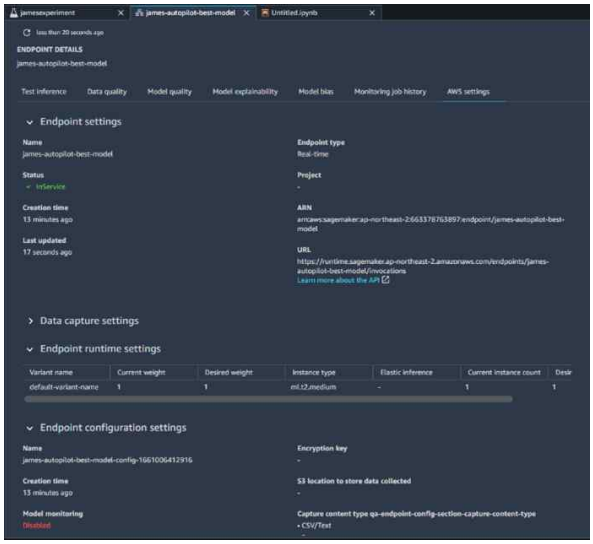



**그림 16.** SageMaker 오토파일럿 최적 모델 상세 정보
**Fig. 16.** Optimal model detail information of SageMaker autopilot model

그림 17은 S3 버킷에 보관된 SageMaker Experiments의 결과물이다. 데이터 전처리 과정, autoML 후보 모델, 모델 훈련 과정, 하이퍼파라미터 튜닝 과정과 모델 검증 결과 등의 보고서와 함께 배포한 모델을 확인할 수 있다.

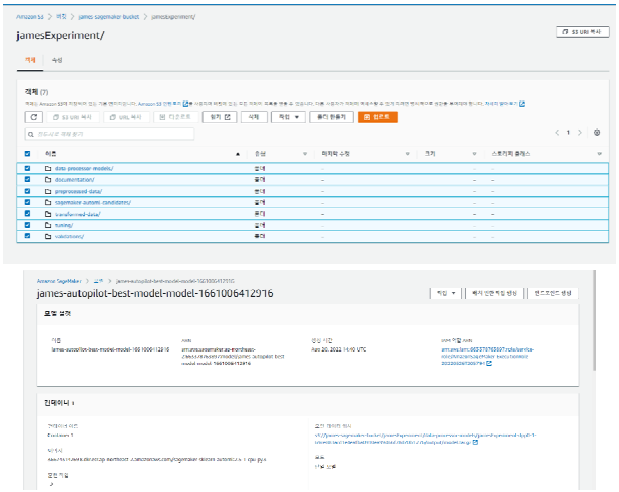



**그림 17.** SageMaker 오토파일럿 결과
**Fig. 17.** Result of SageMaker autopilot

### 3) 사용자 얼굴인식 수행 과정

사용자 얼굴인식을 수행할 코어 디바이스인 라즈베리파이에 구성 요소 배포를 완료하면, 코어 디바이스인 라즈베리파이4에서 소켓 서버가 실행된다.

백엔드 컨트롤러를 통해 얼굴인식 요청이 코어 디바이스에 전달되면 인식 결과를 반환하게 되는데, 사전에 등록된 사용자로 인식된 경우, 이름과 인식 정확도가 반환된다.

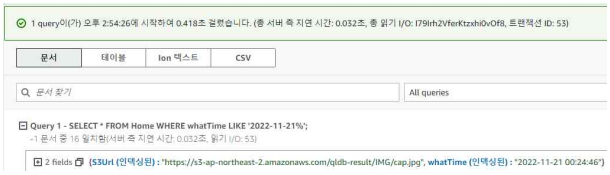
얼굴인식 실험을 위해 min과 hyo 두 사람의 데이터 모델

을 생성하여 학습을 진행하였다. 얼굴인식 결과는 입력된 사진을 기반으로 인식 정확도를 10번 측정하여 평균값을 구한다. 만약 평균값이 일정한 기준값(실험시 0.99) 보다 낮다면 이는 미등록 침입자로 판단한다. 이 경우, 그림 18과 같이 즉시 화면을 캡쳐해서 Amazon S3 bucket에 저장한다. 동시에 그림 19와 같이 이벤트 등록 시간과 캡쳐한 사진의 S3 URL을 Amazon QLDB에 저장하여 추후 객관적 증거로 활용할 수 있도록 한다.


*screen shot of web processing.

**그림 18.** 침입자 이미지 업로드 결과 (S3 Bucket)

**Fig. 18.** Intruder image upload result (S3 Bucket)



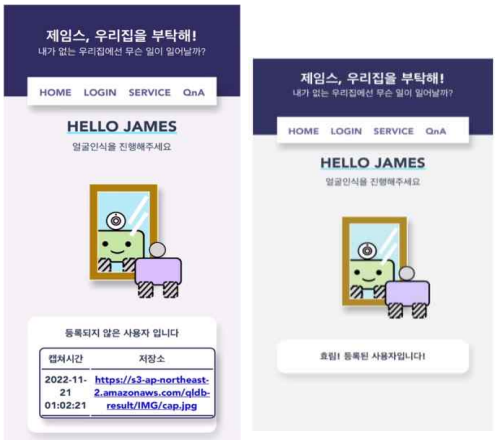

*screen shot of web processing.

**그림 19.** 침입 이벤트 저장 결과 (Amazon QLDB)

**Fig. 19.** Intrude event saving result (Amazon QLDB)

#### 4) 사용자 얼굴인식 수행 결과

사용자 얼굴인식 결과가 도출되면, 클라이언트 디바이스인 지능형 이동체에게 결과를 전달한다. 코어 디바이스와 클라이언트 디바이스는 서로 소켓 통신을 설정하고 얼굴인식 결과인 사용자 이름을 전달한다. 지능형 이동체는 사용자의 이름을 수신하면 TTS(Text to Speech)를 활용하여 스피커로 “Hi, 사용자 이름”과 같이 웰컴 메시지를 재생한다.




*screen shot of app operation.

**그림 20.** 얼굴인식 결과

**Fig. 20.** Face recognize service result

그림 20은 웹앱에서의 사용자 얼굴인식 수행 결과이다. 오른쪽은 등록된 사용자로 인식된 결과로, 사용자 이름이 출력되는 것을 확인할 수 있으며, 왼쪽은 등록되지 않은 사용자의 경우, 해당 이미지를 확인할 수 있는 URL을 보여준다.

## V. 결 론

본 논문은 AWS 클라우드를 기반 머신러닝 및 IoT 솔루션을 활용하여 1인 가구 돌봄 IoT 서비스를 제안하였다. 제안하는 시스템은 영상 인식을 위한 카메라, 공기 질 모니터링을 위한 다양한 센서가 설치되어 있는 IoT 디바이스와, 댁내 곳곳의 상황을 확인하기 위한 지능형 이동체, 그리고 이를 통해 수집된 데이터를 기반으로 지능형 서비스를 제공하기 위한 클라우드 플랫폼으로 구성되어 있다.

이를 위해 본 논문에서는 주요 기능별 프로시저를 제안하였고, 이를 제공하기 위한 시스템 개발 환경을 소개하였으며, AWS IoT Greengrass 기반의 IoT 디바이스 등록, 관리 절차, AWS SageMaker 기반의 지능형 얼굴인식을 위한 모델 학습/배포/인식기능, 지능형 이동체 제어 및 CCTV 전송 기능 등 세부 구축 과정을 기술하였다.

제안하는 1인가구 돌봄 IoT 서비스는 웹앱을 기반으로 사용자 인터페이스를 구성하여, 실내 공기 질 모니터링 기능, 사용자 인식기능, 이동형 CCTV 기능을 제공한다. 사용자는 웹앱을 통해 댁내 환경을 모니터링하고, 사전에 등록되지 않은 침입자를 구분하며, 이벤트 인식 결과를 완전 관리형 원장 데이터베이스 시스템인 AWS QLDB에 저장하여 변경 불가능하며, 암호화된 형태로 기록을 유지한다. 또한 외부에서도 지능형 이동체의 제어를 통해 댁내 곳곳을 이동하며, 실시간 영상을 제공받을 수 있어, 1인 가구의 댁내 관리에 효과적으로 활용 할 수 있으며, 사용자의 심리적 안녕감을 개선할 수 있을 것으로 기대할 수 있다[17].

향후 본 문서에서 설명하는 개발 결과물을 바탕으로 IoT 디바이스 및 지능형 서비스의 신뢰 관리 기술 분석 등에 활용할 예정이다.

## 감사의 글

이 논문은 2023년도 정부 (과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2020-0-00833, 5G 기반 지능형 IoT 트러스트 인에이블러 핵심기술 연구)

## 참고문헌

[1] Red Hat, What are Cloud Services [Internet]. Available:

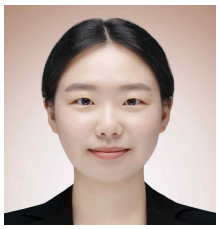
https://www.redhat.com/ko/topics/cloud-computing/what-are-cloud-services
[2] AWS Developer Guide Version 2, What Is AWS IoT Greengrass? [Internet]. Available: https://docs.aws.amazon.com/greengrass/v2/developerguide/what-is-iot-greengrass.html
[3] AWS Developer Guide, What is Amazon SageMaker? [Internet]. Available: https://docs.aws.amazon.com/sagemaker/latest/dg/whatis.html
[4] AWS Developer Guide, What Is AWS Lambd? [Internet]. Available: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html
[5] AWS User Guide, What is Amazon Relational Database Service (Amazon RDS)? [Internet]. Available: https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html
[6] AWS Developer Guide, Getting Started with the Amazon QLDB Console [Internet]. Available: https://docs.aws.amazon.com/qldb/latest/developerguide/getting-started.html
[7] H. S. Choi, J. H. Oh, J. H. Lee, Y. J. Yoon, and W. S. Rhee, "Development of AMB based Reliable Delivery Tracking Service Platform," Journal of Digital Contents Society, Vol. 23, No. 1, pp. 1-4, February 2022. https://doi.org/10.9728/dcs.2022.23.9.1699
[8] Raspberry PI, Raspberry PI Specs [Internet]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/
[9] Intel, Intel RealSense™ ID Solution F455 [Internet]. Available: https://www.intel.co.kr/content/www/kr/ko/products/sku/212561/intel-realsense-id-solution-f455/specifications.html
[10] Eleparts, Rasberrypi Camera Modul V2 [Internet]. Available: https://eleparts.co.kr/goods/view?no=3824792
[11] AWS Developer Guide Version 2, Tutorial: Getting Started with AWS IoT Greengrass V2 [Internet]. Available: https://docs.aws.amazon.com/greengrass/v2/developerguide/getting-started.html
[12] Pypi ORG, beautifulsoup4 4.12.2 [Internet]. Available: https://pypi.org/project/beautifulsoup4/
[13] M. Grinberg, How to Build and Run MJPG-Streamer on the Raspberry Pi [Internet]. Available: https://blog.miguelgrinberg.com/post/how-to-build-and-run-mjpg-streamer-on-the-raspberry-pi
[14] AWS User Guide, What is Amazon S3? [Internet]. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
[15] AWS Developer Guide, SageMaker Autopilot [Internet]. Available: https://docs.aws.amazon.com/sagemaker/latest/dg/autopilot-automate-model-development.html
[16] AWS Developer Guide Version 2, Customize Your Machine Learning Components [Internet]. Available: https://docs.aws.amazon.com/greengrass/v2/developerguide/ml-customization.html
[17] H. J. Kim, "A Cloud Service Architecture for Companion Robot Simulation," *Journal of the Korea Society of Information Technology Policy & Management*, Vol. 14, No. 1, pp. 2785-2791, March 2022.

**최환석(Hoan-Suk Choi)**

2009년 : 한밭대학교 멀티미디어공학과 (공학사)
2011년 : 한밭대학교 멀티미디어공학과 (공학석사)
2018년 : 한밭대학교 멀티미디어공학과 (공학박사)
2015년~현 재: 한국 ITU연구위원회 국제표준전문가
2018년~2020년: 한밭대학교 멀티미디어공학과 박사후연구원
2020년~2021년: 한국과학기술원 전기및전자공학부 위촉연구원
2021년: 주식회사 토브데이터 매니저
2021년~현 재: 카이스트-메가존클라우드 지능형 클라우드 융합기술 연구센터 매니저
※관심분야 : IoT, Social IoT, Semantic Processing, Trust management, Trust Chain, 개인정보보호, GDPR, Cloud Computing, AGI, 스마트 제조

**장승호(Seung-Ho Jang)**

2023년 : 한밭대학교 정보통신공학과 (공학사)
2021년~2022년: 한밭대학교 정보통신공학과 학부연구생
※관심분야 : Application Programming, Cloud Computing, Network Programming, Server Programming



**천효림(Hyo-Rim Cheon)**

2023년 : 한밭대학교 정보통신공학과 (공학사)
2021년~2022년: 한밭대학교 정보통신공학과 학부연구생
2023년~현 재: 메가존클라우드(주) 매니저
※관심분야 : Cloud Computing, IoT, Digital Twin, AWS

**김민정(Min-Jeong Kim)**

2023년 : 한밭대학교 정보통신공학과 (공학사)

2021년~2022년: 한밭대학교 정보통신공학과 학부연구생
※관심분야 : IoT, AWS, Database, AI, Blockchain, Machine Learning, Web

**이우섭(Woo-Seop Rhee)**

1983년 : 홍익대학교 (공학사)
1995년 : 충남대학교 (공학석사)
2003년 : 충남대학교 (공학박사)

1983년~2005년: 한국전자통신연구원 팀장/책임연구원
2005년~현 재: 한밭대학교 지능미디어공학과 교수
2006년~현 재: 한국ITU연구위원회 국제표준전문가
2012년~2013년: 프랑스 Institute TelecomSudParis 방문교수
2018년~2019년: 영국 Liverpool John Moores University 방문교수
※관심분야 : Semantic Processing, Trust Management, Trust chain, IoT, Social IoT