

산학협력 프로젝트 결과보고서

과제명	MQTT를 활용한 반려어 관리 IOT 플랫폼 개발		
협력기관명	(주)나우테스테크놀로지	과제멘토	대표이사 임중권
참여인원	(총 6명) 기업체 1명, 참여교수 1명, 학부과정 4명 학부생: 오경석, 이지민, 추승윤, 박성원 담당교수: 고석주		
수행기간	2019.09.02.~12.20	유형	중기
추진배경	<ul style="list-style-type: none"> ○ 4차 산업의 핵심인 IoT 디바이스의 각종정보를 수집, 분석 및 제어하는 서버 플랫폼에 사용되는 표준 프로토콜 중에 가장 많이 사용되는 MQTT 프로토콜의 이해와 open source 활용 능력을 학습하고 플랫폼을 개발한다. ○ 현재 상용화 되어있는 IoT서비스 플랫폼 중 가정용 관상어의 관리 등에 이용 가능한 서비스 플랫폼은 극히 드물며 제한적임에 따라 새로운 서비스가 필요하다. 		
목표 및 내용	<ul style="list-style-type: none"> ○ 반려어 관리를 위해 IoT 플랫폼 활용 및 구축 <ul style="list-style-type: none"> - 기존의 번거롭던 반려어 육성과정에서 필요한 정보를 수치화 및 분석 - 서버를 통한 각 정보들을 실시간 통합관리 및 시각화 ○ IoT device 의 각종정보를 수집 분석 및 제어 하는 서비스 플랫폼 개발 <ul style="list-style-type: none"> - IoT device에서 각종 센서 정보를 수집함 - IoT 단말에서 수집된 정보를 MQTT 표준 프로토콜을 이용하여 서버로 전송 - 어플리케이션을 통한 사용자 UI제공 및 센서정보 시각화 - 먹이급여장치 등의 자동화 시스템 제공 <div style="text-align: center;"> <pre> graph LR subgraph ARDUINO A1((ARDUINO)) A2((ARDUINO)) A3((ARDUINO)) end subgraph AWS AWS[Server] end subgraph Mobile Phone[Smartphone] end subgraph Firebase FB[Database] end A1 -- "Sensor Data" --> AWS A2 -- "Sensor Data" --> AWS A3 -- "Sensor Data" --> AWS AWS -- "Control" --> Motor[Motor] Motor -- "State" --> AWS AWS -- "Sensor Data" --> Phone Phone -- "Control" --> AWS Phone -- "Push Alert" --> FB FB -- "Sensor Data" --> Phone </pre> </div> <ul style="list-style-type: none"> ○ 기존의 IoT제품들이 단일 기능을 제공하는 경우에도 가격대가 높고 소비자의 맞춤 서비스가 이루어지지 않음 ○ 개발비용이 적으며 간단한 조작만으로 효율적인 서비스 제공이 가능하고 확장성이 보장된 통합관리 서비스의 개발방안을 제시 ○ 소비자가 언제든지 자신의 필요에 맞게 센서의 추가와 제거가 용이 하도록 인프라가 넓은 매체를 사용 ○ 확장이 용이한 인프라와 저렴한 가격으로 기존의 단일 서비스를 제공하던 제품의 가격보다 저렴하게 통합관리 서비스 제공 ○ 어플리케이션 활용 및 사용자 UI/UX 구현으로 간단한 조작으로 사용가능 		

수행결과

- 적은 개발비용으로 통합적인 IoT서비스 플랫폼이 제공 가능함을 보임
- 클라우드 서버 이용 및 데이터베이스 구현 및 활용
- IoT장치 개발완료 및 각 센서의 동작 확인
- 먹이급여 장치 등의 자동화 장치를 구현
- 어플리케이션 이용 센서정보, 가이드 수신 및 급여장치 선택적 조작가능
- 서버 데이터 이용 다양한 기능 추가 제공 가능
- 기존의 제품에 비해 저렴한 IoT 통합관리 플랫폼 구현

<사용자 어플리케이션>



<IoT하드웨어 디바이스 완성본 & 피딩장치>



1. 과제 수행 배경

현대 사회에서 사물 인터넷 즉 IoT의 필요성과 그 활용성은 상당히 잘 알려져 있으며 많은 개발이 진행되고 있다. 4차 산업의 핵심 기술인 IoT기술을 활용하여 생활 속에 적용 된 사례는 이미 수도 없이 많으며 많은 분야에서 사람의 인력을 대체하여 자동화와 원격화에 사용되고 효율적인 생활을 가능하게 해주고 있다. 산업 전반 뿐 아니라 우리 생활 속에서도 뿌리 깊게 자리매김하고 있으며 조금 더 편리한 서비스를 제공하기 위해 발전하고 있다.



< IoT시장은 연평균 22.6%의 성장률을 보인다. >

대형 가전뿐만 아니라 중/소형 가전에도 IoT기술이 적용되어 제품사이의 연결성이 강화되는 추세이며 향후 IoT기술은 기기간 연결에 그치지 않고, 보다 많은 사물에서 빅데이터가 생성되면서 부가가치를 창출하는 방향으로 발전할 것으로 전망된다. IoT 기술의 적용은 어떠한 사물이든 제한이 없으며 또 다른 핵심기술인 VR,AR등 기술의 접목이 용이하여 활용성이 무궁무진하며 다양한 플랫폼으로의 발전을 거듭하고 있다.

그러나 IoT기술은 발전 단계이며 아직은 적용되지 못한 분야나 사용하기에 까다로운 분야가 많이 존재하고 있다. 사람의 판단이 매우 중요한 분야의 경우 아직까지 보조의 역할로만 쓰이고 있으며 사람의 생활에 필요한 개발은 많이 진행되어 왔으나 생물/애완/반려 즉 펫 라이프 분야의 경우 소비자의 수요에 비해 제품군이 한정적이다.

따라서 펫라이프 분야. 그중에서 반려어의 케어에 도움을 줄 수 있는 IoT플랫폼을 개발하고자 한다. 현대 사회에서 1인가구의 증가로 인해 반려견, 반려묘, 반려어 등 함께 의지하며 살아갈 수 있는 애완동물의 수요가 급격히 증가하였다. 그중 반려어의 경우 키우는 것이 목적이 아닌 인테리어 용도로 분양을 받는 사례도 증가하면서 꾸준한 수요를 보이고 있다. 그러나 개체별 까다로운 사육 방법과 번거로운 관리요건으로 인해 자동 케어 시스템의 필요성이 꾸준히 증가해왔으며 수요에 비해 현재 개발된 제품군은 통합적인 플랫폼이 아닌 단일 측정제품에 불과한 것이 현실이다.

또한, 현재 시중의 제품들은 IoT 표준이 정해지지 않아 자사의 플랫폼만을 호환되도록 제품을 만드는 사례가 많으며 이에 따라 소비자들은 자신의 필요에 따라 제품을 선택하고 활용할 수가 없는 경우가 많다.

이에 따라 기존의 제품보다 저렴하고 좀 더 확장성과 활용성이 높은 반려어의 관리에 필요한 통합적인 플랫폼을 개발하여 서버를 통한 자동화를 목표로 프로젝트를 진행하였다.

2. 과제 목표

수조의 반려어 생태환경 관리를 위한 IoT센서 모듈을 제작하고 클라우드 서버를 이용하여 데이터를 관리 및 재전송의 과정을 거쳐 사용자가 어플리케이션을 통해 정보를 수신 받아 현재 상태를 확인하고 먹이 급여 등의 자동화가 이루어지는 것을 목표로 프로젝트를 수행한다.

인프라와 저렴한 가격이 보장된 아두이노를 이용하여 IoT장비를 제작 및 각종 라이브러리를 사용하여 클라우드 서버에 연결하여 정해진 시간 간격으로 센서정보를 서버로 전송하도록 설계하여 하드웨어장치를 완성한다.

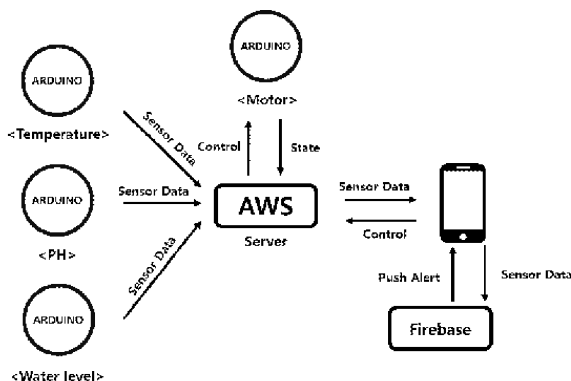
시간과 장소에 영향을 받지 않고 실시간 데이터 처리가 가능하도록 로컬 서버가 아닌 클라우드 서버 중 AWS의 클라우드 서버를 이용하며 개인의 맞춤 서비스가 이루어 져야 하기 때문에 데이터 전송 및 처리에 대하여 보안이 구현되어야 한다. 구체적 보안 구현은 IAM서비스를 통해 3개의 키를 이용하여 SSL기반의 공용키 인증방식을 도입하여 사용하며 인증된 기기가 서버에 연결될 수 있도록 인증키 등록 과정을 거치도록 한다. 기기와의 통신은 네트워크망을 이용하여 MQTT프로토콜 통신 채널을 개설 후 브로커에 의해 제어 되도록 설계하며 각각의 데이터는 등록된 토픽에 따라 구독된 기기로 브로드캐스트 하도록 한다.

수신된 센서 및 컨트롤 데이터는 JSON형태의 데이터로 전송되며 데이터베이스에 타임스탬프에 따라 저장하도록 하여 관리 및 불러오기가 가능하도록 구성한다.

어플리케이션은 서버로부터 사용자가 원할 때 센서 데이터를 받아 현재 수조의 상태를 확인할 수 있도록 하며 데이터를 편집 및 가공하여 여러 형태로 출력하여 정보를 제공한다. 또한, 먹이 급여 장치의 자동화에 대한 시간 정보 및 수동 조작이 가능하도록 기능을 제공한다.

먹이 급여 장치의 경우 서버와 연결되어 자신의 현재 상태를 공유하며 일정 주기마다 먹이 급여를 수행하고 어플리케이션으로부터 서버에 명령이 전달되면 즉시 동작하도록 구현한다.

위의 모든과정을 통해 기존의 제품보다 저렴하나 반려어 양육에 대해 단일 기능이 아닌 통합적 관리 기능을 수행하도록 설계 및 개발을 진행하도록 한다.



3. 과제 수행 결과

1. 구현 상세 내용 및 사용 툴

1-1. 서버

<MQTT 프로토콜>



통신은 MQTT 프로토콜을 이용하여 각 토픽을 구독하는 형태로 데이터 전송이 이루어진다. 저전력 기기에 적합하고 데이터 크기에 제한이 없으며 QOS를 지원하기 때문에 프로젝트에 사용하게 되었다.

<AWS Cloud Server>



AWS에서 제공하는 클라우드 서버를 이용하여 개인 계정으로 개설 및 프로젝트 진행에는 무료티어 계정을 이용한다. 서버에는 각 센서 모듈에 사용될 사물 토픽을 구분하고 이벤트별로 토픽을 생성 및 분류하여 사용한다. 각각의 토픽을 통해 수신되는 데이터들을 구분하고 처리하기 위해 MQTT Broker를 구현하여 사용하였으며 브로커는 토픽을 구분하여 해당 토픽에 대해 각각 기기들의 구독여부를 확인하고 브로드캐스트하는 역할을 수행한다.

<Mongo DB>



메시지 형식으로 JSON형식의 데이터를 사용함으로써 JSON데이터의 뭐리가 가능하고 인덱스의 관리가 쉬워 데이터베이스로 이용하였다.

서버로 수신된 데이터는 각 사물의 메타데이터와 비교하여 올바른 형식의 메시지인지 비교확인 후 Value값을 추출하여 데이터베이스로 넘기고 각각의 해당 토픽 채널로 브로드 캐스트 된다. 데이터베이스로 넘어온 센서정보는 각 타임스탬프에 의해 인덱스에 저장되고 어플리케이션의 센서값 변화량 및 센서정보 업데이트시 불러와 사용하도록 하였다. 데이터베이스 내의 각 데이터의 편집은 서버 및 데이터베이스 계정 인증을 통해서만 편집이 가능하도록 하였다.

<AWS IAM>



개인이 사용하는 서비스이기 때문에 보안 방안은 필수적이다. 때문에 AWS IAM 툴을 이용해서 서버에 등록된 각 클라이언트들의 등록과 인증 방안을 구현하였으며 인증에는 SSL기반의 공용키/개인키/디바이스ID 등을 이용해서 인증을 진행하게 된다. 어플리케이션의 경우 서버의 지역코드와 디바이스에 할당된 ID, 엔드포인트 고유주소를 이용해서 인증을 진행하여 이용자가 등록하지 않은 기기는 정보의 업데이트 및 수신을 할 수 없도록 하였다.

클라우드 서버를 사용하기 때문에 24시간 언제든지 플랫폼의 동작 및 서비스 이용이 가능하며 서버 내에서 사물의 변경과 설정이 용이하도록 하였고 현재 서버의 상태 및 통계정보 모니터링이 가능하다.



<MQTT프로토콜이 주로 사용된 것을 볼 수 있다>

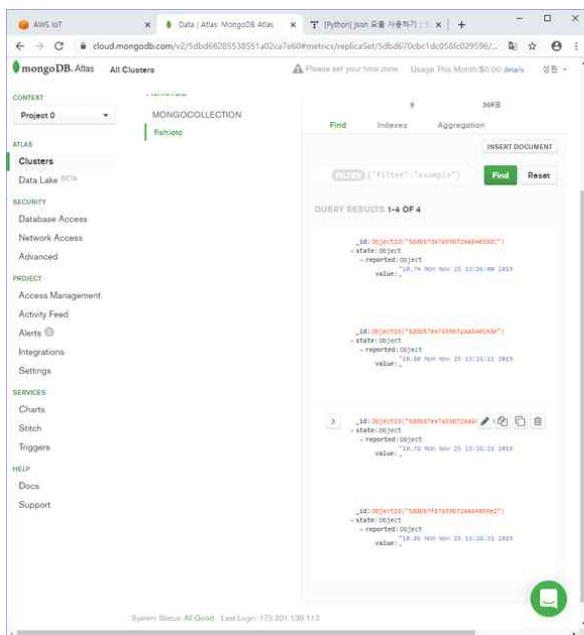
새도우 상태:

```
{
  "reported": {
    "value": "9.8 sun oct 13 02:09:25 2019Wn"
  }
}
```

메타데이터:

```
{
  "metadata": {
    "reported": {
      "value": {
        "timestamp": 1574315165
      }
    }
  },
  "timestamp": 1574580602,
  "version": 2021
}
```

<서버에 등록된 센서의 메타데이터 (PH센서)>



<데이터베이스 내부 모습>

사물 새도우 업데이트

\$aws/things/PH-Sensor/shadow/update

사물 새도우 업데이트 승인

\$aws/things/PH-Sensor/shadow/update/accepted

사물 새도우 문서 업데이트

\$aws/things/PH-Sensor/shadow/update/documents

사물 새도우 업데이트 거부

\$aws/things/PH-Sensor/shadow/update/rejected

사물 새도우 가져오기

\$aws/things/PH-Sensor/shadow/get

사물 새도우 가져오기 승인

\$aws/things/PH-Sensor/shadow/get/accepted

사물 새도우 가져오기 거부

\$aws/things/PH-Sensor/shadow/get/rejected

사물 새도우 삭제

\$aws/things/PH-Sensor/shadow/delete

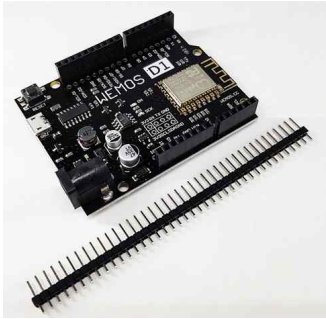
<이벤트별 토픽분류>



< 서버에 보안 인증키가 등록된 모습>

1-2. IoT센서모듈

<아두이노 WEMOS D1R2>



시장의 인프라가 크고 가격이 저렴한 아두이노를 선택하였으며 그중에도 WIFI모듈이 결합된 WEMOS D1R2모델을 사용하였다.

IoT 기기를 제작하기 위해서는 서버와의 연동으로 데이터를 주고받는 과정이 필요한데 블루투스의 경우 인터넷 네트워크는 이용할 수 없기에 WIFI모듈이 내장된 제품을 선택하여 사용하였다. 각각의 센서의 모듈화를 위해서 센서를 분리하고 각각의 아두이노에 센서 하나씩 할당하여 사용하였다. 때문에 원하는 센서만 사용하거나 원하는 센서를 추가하는 것이 쉽도록 제작하였다.

<PAHO Library>



AWS 서버와의 연결을 위해 아마존 웹서비스에서 기본 제공하는 라이브러리는 아두이노의 UNO계열의 모델만 지원하기 때문에 PAHO라이브러리를 이용하여 서버와의 MQTT통신이 가능하도록 일반적이지 않은 방식을 이용하였다.

<Arduino IDE>

개발툴은 일반적으로 많이 사용하는 아두이노의 기본 개발툴을 이용하여 개발을 진행 하였다.

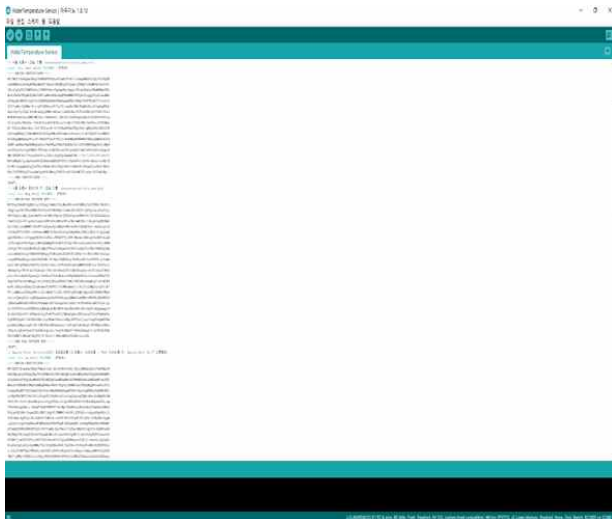
<Sensor>



아두이노 호환 센서들을 이용하여 요소별 측정치를 얻을 수 있도록 제작하였다. 센서는 온도, 수위, PH 3가지의 일반 측정 센서를 이용하였으며 아두이노용 모터센서를 이용하여 자동 먹이급여장치를 제작하였다. 아두이노 호환 센서들을 이용했기 때문에 대부분 가격이 저렴하고 센서의 인프라가 넓어 원하는 기능을 추가하고 빼는 등의 선택적 사용에 용이하다. 온도의 경우 측정치의 소수 첫째자리 까지 데이터로 이용하며 수위의 경우 측정높이 범위인 0~1500까지 범위내의 측정치를 퍼센트로 환산하여 활용하며 pH측정치의 경우 소수 둘째까지 이용한다. 모터의 경우 현재 동작 상태를 1과 0으로 체크하여 서버에 전송하게 된다.

각각의 센서 모듈은 내장된 Public/Private/Device ID를 사용하여 서버와 연결 시 인증키로써 활용하도록 하였으며 각 모듈은 서로 다른 토픽을 서버로부터 생성하여 메시지 전송에 이용한다. 일반 센서 측정치의 경우 10초에 한 번씩 업데이트 이벤트가 발생하고 생성된 메시지가 메타데이터 형식과 일치할 경우 메시지가 업데이트 되게 된다. 모터의 경우 서버에 기록된 마지막 먹이급여시간을 기준으로 6시간 후에 자동으로 동작하도록 설정 하였으며 어플리케이션을 통해 사용자가 원할 때 언제든지 먹이급여가 가능하도록 모터 명령 메시지를 아래와 같이 할당하여 사용하였다.

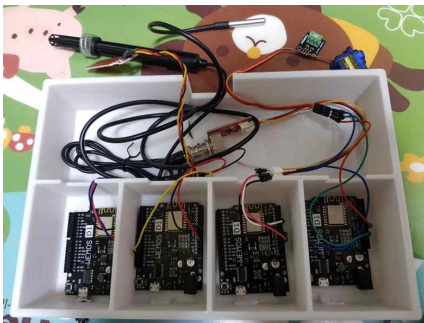
```
Sensor message : { "state": { "reported": { "value": "? Date ₩ time" } } }  
Motor message : " { \"state \ \ ₩\": { \"desired \ \ ₩\": { \"motor \ \ ₩\": 1 } }, \ ₩\"client Token \ ₩\":  
 \ ₩\"Feeder \ ₩\" }"
```



<인증기가 내장된 모습>



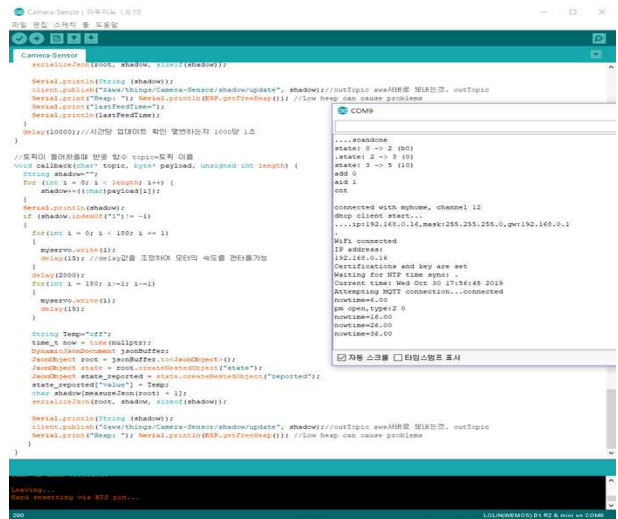
< Publish 코드>



<센서모듈 전체>



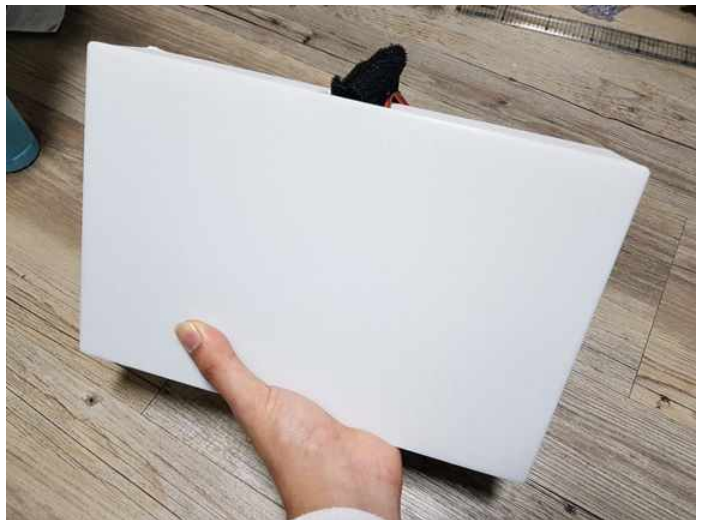
<먹이 급여 장치>



<서버연결 단계의 시리얼 모니터 모습>



<센서 메시지 업데이트 승인>



<케이스 작업 완료 후 완성모습>

1-3. Android Application

<Android Studio>

안드로이드 어플리케이션 개발 출로는 안드로이드 스튜디오를 사용하였다.

<Android AWS Library>

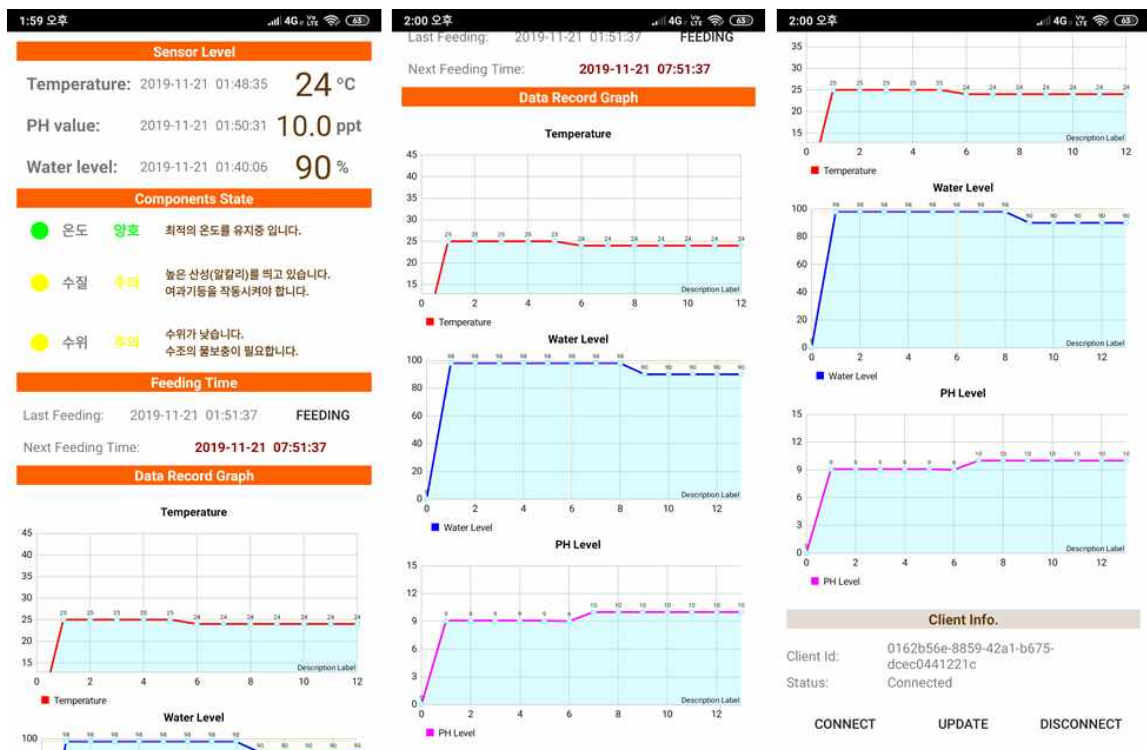


서버와의 MQTT통신을 이용하기 위해 AWS의 안드로이드용 라이브러리를 이용하였으며 앞서 이야기한 보안 인증을 위해 서버로부터 디바이스 ID를 할당받아 서버 연결 시 서버의 지역코드와 함께 정확히 서버를 구분하고 등록된 후에만 메시지를 수신 받도록 이중보안을 구현하였다. 서버의 토픽에 대해 선택적으로 구독 수정이 가능하고 메시지 수신 및 푸시가 가능하도록 구현하였다.

<FireBase project>

위험 수준의 측정데이터 수신 시에 사용자의 앱에 푸시알림을 올려주기 위해 Firebase의 프로젝트를 안드로이드 스튜디오의 현재 개발 프로젝트와 연결하여 이용하였다.

앱 실행시 서버로부터 실시간으로 측정정보를 수신 받아 각각의 출력형태로 변형하여 사용하며 피딩을 위한 메시지 푸시버튼을 사용할 수 있도록 하였다. 스크롤뷰 형식의 앱에서 제공되는 기능으로는 현재 측정수치와 마지막 업데이트 시간이 상단에 표시되고 해당 수치에 따른 현재 상태와 조치사항이 가이드로 아래쪽에 표시된다. 중단에는 마지막 먹이급여 시간과 다음 먹이급여 예정시간이 표기되며 오른쪽의 피딩버튼을 통해 사용자가 언제든지 먹이급여가 가능하도록 구현하였다. 아래쪽으로는 측정치의 변화량을 한눈에 볼 수 있도록 컴포넌트별 그래프를 제공하며 가장 아래쪽에는 서버에 등록시 사용한 클라이언트 아이디와 현재 서버와의 연결 상태 및 앱 연결 오류시 리셋을 위한 버튼들이 생성 되어있다.



```

void initIoTClient() {
    Region region = Region.getRegion(MY_REGION);
    mqttManager = new AWSSlotMqttManager(clientId, CUSTOMER_SPECIFIC_ENDPOINT);
    mqttManager.setKeepAlive(10);
    AWSSlotMqttLastWillAndTestament lwt = new AWSSlotMqttLastWillAndTestament(.willTopic: "my/lwt/topic",
        willMessage: "Android client lost connection", AWSSlotMqttQos.QOS0);
    mqttManager.setMqttLastWillAndTestament(lwt);
    mlotAndroidClient = new AWSSlotClient(AWSMobileClient.getInstance());
    mlotAndroidClient.setRegion(region);
    keystorePath = getFilesDir().getPath();
    keystoreName = KEYSTORE_NAME;
    keystorePassword = KEYSTORE_PASSWORD;
    certificateId = CERTIFICATE_ID;
    try {
        if (AWSSlotKeystoreHelper.isKeystorePresent(keystorePath, keystoreName)) {
            if (AWSSlotKeystoreHelper.keystoreContainsAlias(certificateId, keystorePath,
                keystoreName, keystorePassword)) {
                Log.i(LOG_TAG, "msg: " + "Certificate " + certificateId
                    + " found in keystore - using for MQTT.");

                clientKeystore = AWSSlotKeystoreHelper.getIoTKeystore(certificateId,
                    keystorePath, keystoreName, keystorePassword);
                runOnUiThread() -> {
                    btnConnect.setEnabled(true);
                };
            } else {
                Log.i(LOG_TAG, "msg: " + "Key/cert " + certificateId + " not found in keystore.");
            }
        } else {
            // ...
        }
    } catch (Exception e) {
        // ...
    }
}

```

< App Init code >

```

public void subscribeClick(final View view) {
    String topic = "";

    if (subscribeCount == 3) {
        topic = "$aws/things/Saturday/shadow/update";
        subscribeCount = 1;
    } else if (subscribeCount == 2) {
        topic = "$aws/things/PH-Sensor/shadow/update";
        subscribeCount++;
    } else if (subscribeCount == 1) {
        topic = "$aws/things/WaterTemperature-Sensor/shadow/update";
        subscribeCount++;
    }

    Log.d(LOG_TAG, "msg: " + "topic = " + topic);

    try {
        mqttManager.subscribeToTopic(topic, AWSSlotMqttQos.QOS0,
            (topic, data) -> {
                runOnUiThread() -> {
                    try {
                        String message = new String(data, CharsetName.UTF_8);
                        Log.d(LOG_TAG, "msg: " + "Message arrived:");
                        Log.d(LOG_TAG, "msg: " + "Topic: " + topic);
                        Log.d(LOG_TAG, "msg: " + "Message: " + message);

                        if (topic.equals("$aws/things/WaterTemperature-Sensor/shadow/update")) {
                            String tapmessage = new String(message);
                            // ...
                        }
                    } catch (Exception e) {
                        // ...
                    }
                }
            });
    } catch (Exception e) {
        // ...
    }
}

```

< Topic Subscribe Code >

```

public void publishClick(final View view) {
    final String topic = "$aws/things/Camera-Sensor/shadow/update";
    final String msg = "{\"state\":{\"desired\":{\"motor\":1}},\"clientToken\":\"Feeder\"}";

    try {
        mqttManager.publishString(msg, topic, AWSSlotMqttQos.QOS0);
    } catch (Exception e) {
        Log.e(LOG_TAG, "msg: " + "Publish error.", e);
    }

    Last_Feeding_Update(view);
}

```

< Message Publish Code>

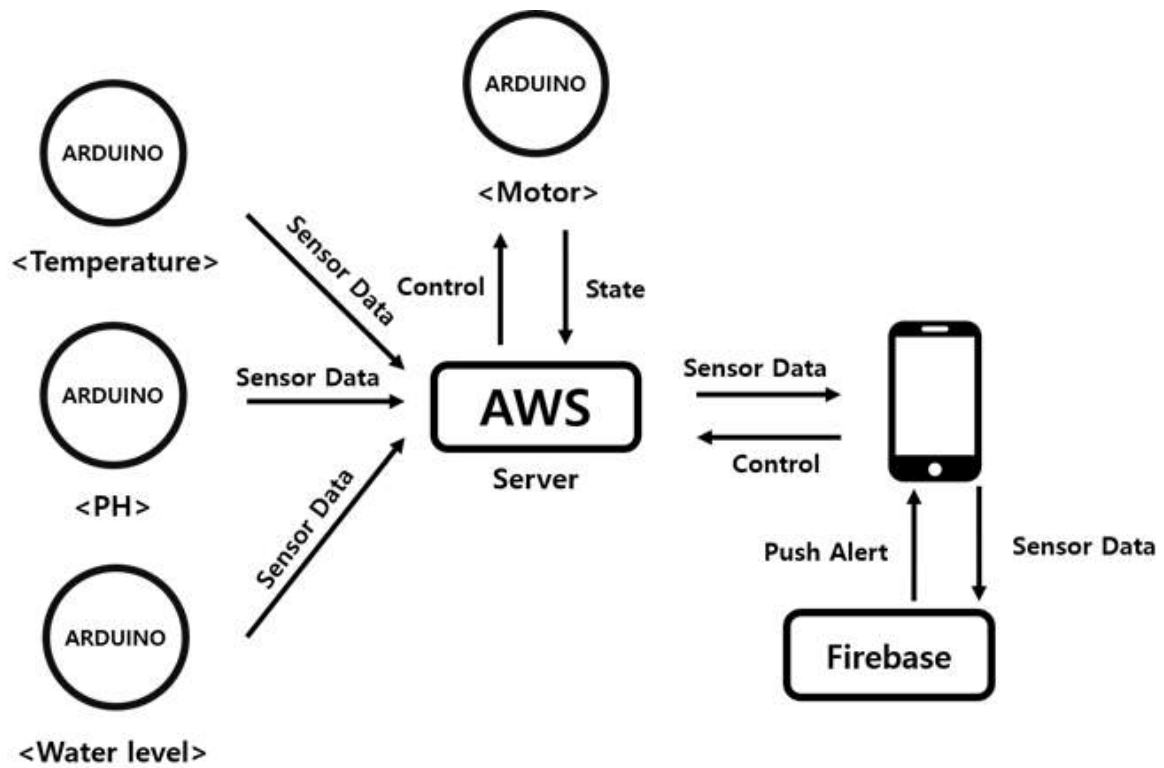
```

{
    "Version": "1.0",
    "IdentityManager": {
        "Default": {}
    },
    "CredentialsProvider": {
        "CognitoIdentity": {
            "Default": {
                "PoolId": "ap-northeast-2:b41578fa-2b3d-47db-b",
                "Region": "ap-northeast-2"
            }
        }
    }
}

```

< 디바이스 ID 할당 >

1-4. 시스템 구조도




1-5. 데모영상


<https://youtu.be/qkRG1P55mkl>

2. 개발 진행 사항

2-1. 개발 회의록

번호	1	
일시	2019. 9. 3 10:00 - 11:00	
장소	공대 카페	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 기업 요청 개발 내용 확인 2. 가능 기술내용 확인 3. 프로젝트 구현 내용 확인 4. 재료비 산정 	


번호	2	
일시	2019. 9. 4 18:00 - 18:45	
장소	온라인 화상회의 진행(ZOOM 이용)	
참여자	오경석, 박성원, 이지민, 추승윤, 임중권(Mentor)	
내용	<ol style="list-style-type: none"> 1. 구현 기반 기술 내용 확립 2. 사용 툴 및 개발 방향 토의 3. 프로젝트 목표 확립 4. 구현에 필요한 기술 정리 	


번호	3	
일시	2019. 9. 6 18:00 - 18:30	
장소	IT융복합관 245	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 프로젝트 구체적 구현 내용 토의 2. 개발 방향 수정 3. 개발 스케줄 작성 4. 기기 구현 방안 수정 5. 서버 구축 계획 	

번호	4	
일시	2019. 9. 9 21:00 - 22:30	
장소	Discord(온라인 회의)	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 프로젝트 구체적 목표 확립 2. 개발방향 수정 및 재검토 3. 개발 단계 수정 4. IoT기기의 개발 목적 설정 5. 반려어 관련 조사 	

번호	5	
일시	2019. 9. 10 20:00 - 22:30	
장소	Discord(온라인 회의)	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 각 센서별 필요 측정요소 조사 2. 구현 가능 기능 선정 3. 개발 툴 및 구현 가능성 검토 4. 기능의 사용방안 결정 5. 웹뷰 개발 방안 조사 	

번호	6	
일시	2019. 9. 11 12:00 - 12:15	
장소	Direct Call(음성통화)	
참여자	오경석, 임중권(Mentor)	
내용	<ol style="list-style-type: none"> 1. 반려어 대상 기능 구현 방향 보고 2. 개발 계획 및 수정사항 검토 3. 재료관련 및 필요 툴 검토 4. 개발 계획 보고 	

번호	7	
일시	2019. 9. 18 18:00 - 19:00	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 서버 개발 사항 정리 2. 클라우드 서버 플랫폼 선정 3. 계정 개설 및 서버 구현 작업 4. 센서 모듈 개발 환경 구축 	

번호	8	
일시	2019. 9. 24 18:00 - 20:30	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 서버 MQTT채널 개설 방안 토의 2. 데모 센서 제작 3. 서버 연결 오류 사항 검토 4. 센서 업데이트 데이터 형식 지정 	

번호	9	
일시	2019. 9. 30 15:00 - 22:30	
장소	오경석 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 데모 센서 측정 데이터 설정 2. 서버 보안 구현 3. 서버와 센서 연결 환경 구축 4. MQTT 채널 개설 및 토픽 설정 5. 센서 퍼블리쉬 기능 구현 	


번호	10	
일시	2019. 10. 2 18:00 - 18:30	
장소	Discord(온라인 회의)	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 서버 MQTT채널 개설 완료 및 세부설정 결정 2. 센서 모듈 서버 연결 시 인증키 설정 	

번호	11	
일시	2019. 10. 5 21:00 - 21:30	
장소	Discord(온라인 회의)	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 수위센서 모듈 완성 2. 센서 서버 연결 테스트 3. 센서 업데이트 데이터 형식 수정 4. 사물 메타데이터 확정 5. 웨도우 업데이트 승인 확인 	

번호	12	
일시	2019. 10. 9 18:00 - 22:30	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 업데이트 데이터에 대한 브로커 동작 설정 2. 센서 측정치 표기사항 결정 3. 측정센서 모듈 수위, 수질 작업완료 4. 서버 승인 데이터 확인 5. 인증키 동작 확인 6. 측정정보 사용자 시각화 작업 토의 7. 웹뷰에서 어플리케이션으로 전환 8. 어플리케이션 개발환경 구축 9. 데이터 처리방안 검토 	




번호	13	
일시	2019. 10. 16 18:00 - 24:00	
장소	오경석 학우 집	
참여자	오경석, 박성원	
내용	<ol style="list-style-type: none"> 1. 어플리케이션 데모버전 개발 2. 라이브러리 연결 및 서버 연결 경로 결정 3. MQTT 통신 채널 개통 4. 라이브러리 오류 수정 및 데이터 형식 설정 5. 서버에 등록된 센서 토픽 연결 6. 브로드 캐스트 된 데이터 수신 구현 7. 어플리케이션 구현 기능 결정 8. 센서 측정값 동기화 및 시각화 완료 9. 가이드 출력 작업 진행 	

번호	14	
일시	2019. 10. 18 15:00 - 24:00	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 측정 센서 모듈 기능 구현 및 서버 연결 완료 2. 인증키 동작 및 보안 서비스 확인 3. 어플리케이션 서버 연결 설정 수정 4. 임시 사용자 UI 구현 5. 서버 측정 데이터 업데이트 설정 완료 6. 데이터베이스 작업환경 구축 7. 어플리케이션 측정데이터, 가이드 메시지 출력완료 9. 센서 하드웨어 내구성 보강작업 8. 추가 기능 검토 	

번호	15	
일시	2019. 10. 30 21:00 - 21:30	
장소	Discord(온라인 회의)	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 추가 기능 관련 의견 제시 2. 제시된 의견에 따른 개발 가능성 검토 3. 개발 방안 및 작업 소요시간 검토 4. 자동 먹이 급여 장치 추가 검토 5. 재료 산정 및 주문진행 6. 서버 및 어플리케이션 개편 검토 7. 데이터 베이스 연결 작업 시작 	

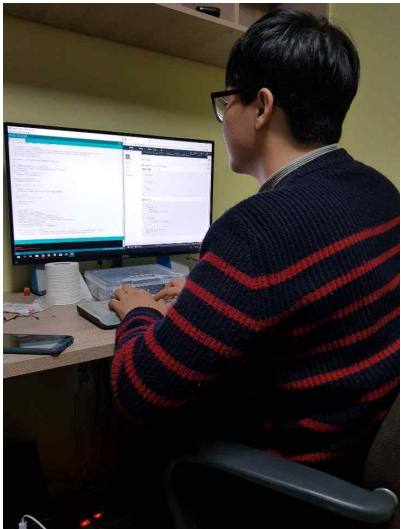
번호	16	
일시	2019. 11. 01 21:00 - 21:30	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	1. 중간 발표자료 작성	

번호	17	
일시	2019. 11. 06 15:00 - 19:00	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 피딩모터 제작 및 모듈화 2. 피딩 모터 서버 연결 및 메타데이터 설정완료 3. 모터 회전각 설정 및 동작 상태 구분 설정 4. 서버 모터 명령 체계 구축 5. 어플리케이션 피딩메뉴 개설 6. 컨트롤 메시지 푸쉬 기능 구현 7. 서버 업데이트 승인 확인 	

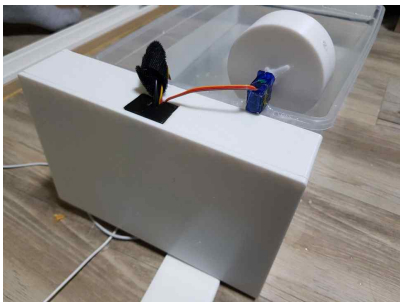
번호	18	
일시	2019. 11. 08 20:00 - 23:30	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 어플리케이션 UI개편 및 UX수정 2. 사용자 선택적 피딩기능 추가 3. 사물 업데이트 지연시간 단축 4. 모터 주기적 동작 확인 5. 데이터 베이스 구축 작업 진행 	

번호	19	
일시	2019. 11. 13 18:00 - 20:30	
장소	Discord(온라인 회의)	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 진행 작업 에러사항 검토 2. 수정방안 검토 3. 추가 기능 검토 	

번호	20	
일시	2019. 11. 19 18:00 - 18:20	
장소	온라인 화상회의 진행(ZOOM 이용)	
참여자	오경석, 박성원, 이지민, 추승윤, 임종권(Mentor)	
내용	1. 현재 진행사항 보고 및 제품 패키징 제안	

번호	21	
일시	2019. 11. 20 17:00 - 23:30	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 데이터베이스 데이터 관리 체계 확립 2. 인증된 기기정보 구분 구현 3. 인증된 기기에서만 불러오기 및 데이터 편집 가능하도록 구현 4. 데이터 축적 및 데모 데이터 측정 5. 어플리케이션 데이터 베이스 연결 6. 이전 측정치 불러오기 작업 진행 	

번호	22	
일시	2019. 11. 27 15:00 - 20:00	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 데이터 베이스 구축완료 2. 어플리케이션 연동 작업 완료 및 피딩메뉴 완료 3. 데이터 불러오기 및 수신용 인덱스 구축 4. 컴포넌트별 변화량 그래프 추가 5. 그래프 UI 디자인 조정 6. 확대 및 축소 터치 이벤트 구현 	

번호	23	
일시	2019. 11. 28 17:00 - 19:00	
장소	이지민 학우 집	
참여자	오경석, 박성원, 이지민, 추승윤	
내용	<ol style="list-style-type: none"> 1. 하드웨어 패키징 작업 진행 2. 센서 측정 오류 수정 3. 데모영상 촬영 	

2-2. 일정표

[illegible]

4. 기대 효과 및 활용 방안

1. 현대인의 반려어 양육에 대한 진입장벽을 낮춤

바쁜 현대인들에게 반려동물에 대한 인기는 갈수록 높아지나 시간 부족과 반려동물의 양육에 시간을 투자할 수가 없는 경우가 많아 양육을 포기하는 경우가 많다. 반려어는 양육 난이도가 높고 조금만 신경 써주지 않는다면 폐사 하는 경우가 많기에 반려동물 중에서도 관리가 까다롭다. 그러나 수조의 정확한 수치제어 및 규칙적인 관리만 이루어진다면 생명주기를 보장할 수 있는 개체가 반려어 이다.

따라서, 반려어에 대한 통합관리 시스템이 주기적인 먹이 급여와 실시간 수치측정 및 현재 상황을 자동으로 체크해준다면 크게 신경 쓰지 않고도 안정적으로 기를 수 있는 환경을 제공한다. 반려어에 대해 양육을 잘 할 수 있을지 고민하는 경우 이처럼 진입장벽을 낮춰 줌으로써 긍정적인 영향을 끼칠 것으로 기대된다.

2. 저가형 IoT기기의 가격 경쟁력과 영향

기존의 IoT기기들은 단일 기능을 제공함에도 가격대가 20만원 이상으로 높게 책정되는 경우가 많았다. 이에 본 프로젝트를 통해 단일 대상에 대한 통합적 서비스를 저렴한 가격을 통해 개발 및 제공 할 수 있다는 것을 증명하면서 시장에 새로운 동기를 부여하고 저렴한 제품으로 소비자들의 구매가 이어진다면 다시 한번 IoT시장이 성장하는 형태의 긍정적 효과를 기대할 수 있다.

3. IoT시장의 성장에 따른 클라우드 서비스 성장

기존의 내부 네트워크만을 이용하여 서비스를 제공하던 것과 달리 본 프로젝트에서는 클라우드 서버를 이용하여 서비스를 제공 및 데이터를 관리하는 방안을 제시함으로써 시간, 공간에 제약을 받지 않고 서비스를 활용할 수 있도록 하였다. 이는 사용자들에게 좀 더 편리한 사용 환경을 제공하며 더 나아가 다른 서비스와의 융합 및 연계의 확장성을 보장한다. 이러한 클라우드 서비스를 급성장 중인 IoT시장에서 활발하게 사용된다면 클라우드 서비스 또한 필수적으로 함께 성장과 발전을 거듭할 것이다.

4. 플랫폼 개혁과 확장성

현재 IoT의 표준이 확립되지 않은 IoT시장에서 출시 된 제품들은 자사의 플랫폼만을 고집하는 경우가 많고 타 서비스와 연동이 불가능한 경우가 많다. 그러나 본 프로젝트를 통해 각각의 기기를 모듈화 시켜 쉽게 확장이 가능하도록 하며 인프라가 넓은 매체를 사용함으로써 향후 다양한 서비스의 플랫폼이 공유하고 융합 및 확장이 가능하도록 장려하는 효과를 기대할 수 있다.

5. 반려어 시장에서의 새로운 가능성

반려어에 대한 IoT기기들은 현재 많은 제품이 존재하지는 않으며 단일 기기만을 제공함에도 가격대가 높아 일반 구매자가 사용하기는 힘들다. 따라서, 저렴한 반려어 관리 시스템의 출시를 통해 반려어 시장의 새로운 관리기기 표준으로써 구매율을 보장하고 새로운 가능성을 제시할 수 있을 것으로 기대된다.

5. 주요 산출물

논문	
제목	관상어 관리를 위한 스마트 시스템 구현
저자	오경석, 박성원, 이지민, 추승윤, 고석주, 임중권
학회	한국정보과학회 - 2019한국소프트웨어종합학술대회(KSC2019) 학부생/주니어 논문 경진 대회 2019. 12. 20

관상어 관리를 위한 스마트 시스템 구현

오경석¹, 박성원, 이지민, 추승윤, 고석주, 임중권

경북대학교

naurytm@gmail.com, tt0410tt@naver.com, blapls@naver.com, cnuudgh3333@naver.com

, sjkoh@knu.ac.kr, jglim@nautech.com

Implementation of Smart System for Pet Fish Management

Kyoung-Buk Oh¹, Sung-Won Park, Ji-Min Lee, Seung-Yun Choo, Seok-Ju Koh, Joong-Kwon Lim

KYUNGPOOK NATIONAL UNIVERSITY

요약

본 논문은 클라우드 서버와 아두이노 센서값을 이용하여 통합적인 관상어 관리를 IoT서비스를 제공하는 것을 구현한 것에 관한 논문이다. 기존의 사물인터넷 플랫폼은 특정 업체만의 서비스만 제공하거나 통합 서비스의 경우에도 서비스 구축에 높은 비용이 발생하였다. 또한 각 기기간의 사물인터넷 환경 보장이 정확히 확인되지 않은 상태에서 각 개발 회사들의 제품이 한 회사의 IoT시스템과 통합되지 않는 문제점이 존재하여 확률이 높지 않다. 본 논문에서는 현재 코마코를 중심으로 한 통합적인 IoT서비스가 확인되지 못한 관상어의 관리에 대한 서비스를 세부적이고 효율적으로 제공할 수 있음을 보여줌과 동시에 클라우드 서버와 저가형 상용칩을 활용하여 충분히 통합적인 IoT시스템 구축이 가능하다는 것을 입증함으로써 사용자들의 효율적인 관상어 관리의 지원 가능성과 기존의 사물인터넷 플랫폼의 문제점이 쉽게 해결 가능함을 보이고 시간이 더욱 더 많은 분야에 적용되고 적용되는 것을 목표로 한다.

1. 서론

4차 산업혁명이라는 용어가 등장한 후, 4차 산업혁명 분야 시장은 매년 성장하고 있으며, 특히 4차 산업의 핵심 분야인 사물인터넷(IoT)과 클라우드(Cloud) 서비스 시장은 매년 비약적인 성장을 이루고 있으며 그중 사물인터넷(IoT) 시장은 국내시장 추이 8조 8천억원 규모에 연평균 22.8%성장이라는 놀라운 시장 성장을 보여주고 있다.[1]



그림 1. 사물인터넷(IoT) 국내 시장의 성장 추이[2]

이런 성장에 맞추어 플랫폼 및 다양한 규모의 기업, 스타트업 기업들은 다양한 분야에서 활용 가능한 사물인터넷 서비스 플랫폼을 개발 및 제공하고 있으며 플랫폼 자율 관리, 환경, 재난 등 규모의 범위에 활용되는 서비스부터 개별 기기의 동작, 개인 생활 습관, 가정 등 특정 목적을 위한 소규모의 서비스까지 다양한 시도가 이루어져 서비스되고 있다. 그중 가정에서 사용하는 IoT시

스템은 빠르게 성장하고 있으며 이미 SKT, KT, LG 등 메이저 통신사들 중심으로 스마트홈 기술의 통합적인 IoT시스템을 제공하고 있고 중국의 샤오미 등에서 만원 플랫폼에서 모바일과 연동할 수 있는 저가형 IoT 제품을 빠르게 출시하고 있다. 이처럼 사물인터넷 시장이 커지고 있으며 앞으로 더욱 많은 성장의 가능성을 지니고 있다.

그러나 시장의 발전에도 불구하고 개인 사용자의 일반 가정에서는 통합적인 서비스가 기존의 플랫폼을 사용되고 있지 않으며 서비스의 제공 대상도 제한적이다. 특히나 현재 가정에서의 또 다른 가족으로 여겨지는 반려동물들 대상으로 제공되는 서비스는 그 개수가 매우 제한적이고 까다로운 조건과 높은 가격으로 인해 많이 쓰이지 못하고 있다. 첫 번째, 고양이와 같은 포유류는 대상으로 활용가능한 제품은 만일 기능 제공을 위해 서서히 출시되고 있지만 파충류, 관상어 등의 비주류는 여전히 반려동물들 대상으로는 활용할 수 있는 제품이 제한적이고 높은 가격대를 자랑한다.

현재 우리나라 반려동물 시장의 수요는 폭발적으로 증가하고 있으며 그중 중간 활용과 장식적 환경, 인테리어 등의 이유로 반려어의 인기가 급상승하며 반려동물 순회 3위를 기록했다. 한국 관상어 협회에 따르면 지난해 한국과 중국의 관상어 생산량이 전년 대비 40%증가하였으며 약 1조 8000억원에 육박한다고 나타났다.[3]

이 같은 관상어의 수요증가에도 불구하고 현재 IoT서비스 플랫폼의 개인 사용자를 위한 만일 대상의 통합적 플랫폼의 개발과 설치비용이 높은 편이고 반려어의 경우 각종 조건이 까다로워 사물인터넷의 혜택을 적용하기 어려웠다. 현재 시장의 상용화된 제품들은 저가의 플랫폼만을 제공하는 경우가 많아 확장성이 떨어져서 무용지민

6. 참여인력(세부)

지도교수	소속	컴퓨터학부		성명	오경석
참여인력 (산업체)	기업명	성명	직위	전화	Email
	나우테스 테크놀러지	임중권	대표이사	01022859620	jglim@nautestech.com
과 제 참 여 학 생	소속(학과)	학위과정 (성별)	학번	성명	담당업무
	컴퓨터학부	학사과정 (남)	2014097054	오경석	팀장, 서버구축 어플리케이션 개발
	컴퓨터학부	학사과정 (남)	2014097065	이지민	센서모듈 제작
	컴퓨터학부	학사과정 (남)	2014097085	추승윤	센서모듈제작
	컴퓨터학부	학사과정 (남)	2014097035	박성원	서버, 데이터베이스, 클라이언트 연결