종합설계 프로젝트 수행 보고서

프로젝트명	치매 환자 배회 관리 IOT 시스템
팀번호	S1-6
문서제목	수행계획서(O) 2차발표 중간보고서(O) 3차발표 중간보고서(O) 4차발표 중간보고서(O) 최종결과보고서(O)

2020.12.04.

팀원 : 신정희 (팀장)
이준규
이영주
지도교수 : 최진구 교수 (인)
지도교수 : 노영주 교수 (인)

문서 수정 내역

작성일	대표작성자	버전(Revision)	수정내용	
2019.01.22	신정희(팀장)	1.0	수행계획서	최초작성
2020.02.28		2.0	2차발표자료	상세 설계 추가
2020.05.02		3.0	3차발표자료	시험결과추가
2020.06.27		4.0	4차발표자료	시험결과추가
2020.12.04		5.0	최종결과보고서	결론 추가

문서 구성

진행단계	프로젝트 계획서 발표	중간발표1 (2월)	중간발표2 (4월)	학기말발표 (6월)	최종발표 (10월)
기본양식	계획서 양식	계획서 양식	계획서 양식	계획서 양식	계획서 양식
포함되는 내용	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~3) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~4) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~5) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~7) 참고자료	 I II III

이 문서는 한국산업기술대학교 컴퓨터공학부의
“종합설계” 교과목에서 프로젝트
“치매 환자 배회 관리 IOT 시스템” 을 수행하는
(S1-6, 신정희, 이영주, 이준규)들이 작성한 것으로 사용하기
위해서는 팀원들의 허락이 필요합니다.

목 차

Ⅰ. 서론

1. 작품선정 배경 및 필요성 ····· - 4 -
2. 기존 연구/기술동향 분석 ····· - 7 -
3. 개발 목표 ····· - 7 -
4. 팀 역할 분담 ····· - 8 -
5. 개발 일정 ····· - 8 -
6. 개발 환경 ····· - 9 -

Ⅱ. 본론

1. 개발 내용 ····· - 10 -
2. 문제 및 해결방안 ····· - 11 -
3. 시험시나리오 ····· - 12 -
4. 상세 설계 ····· - 13 -
5. Prototype 구현 ····· - 21 -
6. 시험/ 테스트 결과 ····· - 24 -
7. Coding & DEMO ····· - 28 -

Ⅲ. 결론

1. 연구 결과 ····· - 35 -
2. 작품제작 소요재료 목록 ····· - 35 -

참고자료 ····· - 36 -

Ⅰ. 서론

1. 작품선정 배경 및 필요성

경제 수준의 상승 및 의학의 발전에 따른 평균수명의 연장과 사회문화적 변화에 의한 출산율의 감소에 따라 노인 인구의 비율이 빠르게 증가하면서, 치매의 발견과 조기 치료의 중요성이 부각되고 있다.

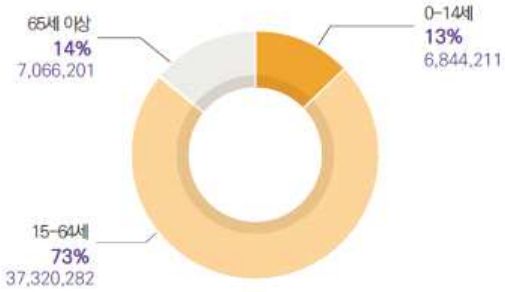


그림 1. 2017년 인구구성비 (단위: 명, %)

2017년 기준 대한민국의 전체인구 51,230,704명 중 65세 이상 노인 인구는 7,066,201명으로 13.8%를 차지하면서 고령사회에 진입하였다. 서울시는 2019년 9월 10일 “8년 후인 2027년에는 서울시내 65세 이상 인구 비율이 21.2%(201만 4000명)로 초고령사회가 될 것으로 전망된다”고 밝혔다.

노인의 대표적 질환인 치매의 유병률 또한 고령사회가 진행됨에 따라 가파르게 증가하고 있다.

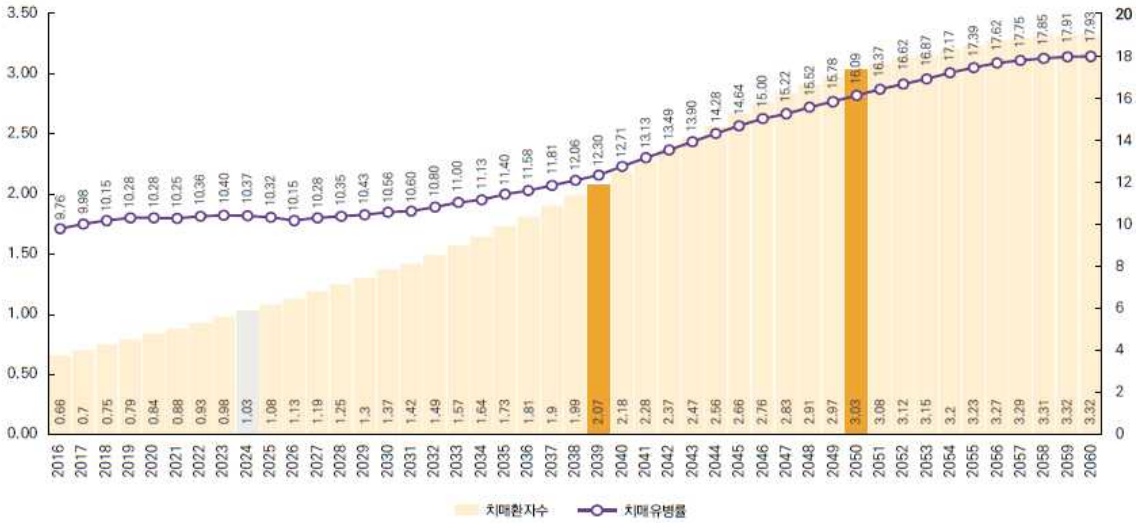


그림 2. 연도별 치매환자 수 및 유병률 추이(2016-2060) (단위: 백만 명, %)

중앙치매센터의 보고에 따르면 65세 이상 노인인구의 치매환자 비율은 2017년 기준 약 10.0%로 추정된다. 2016년에는 66만 명이었으나 2024년에 백만 명, 2039년에 2백만 명, 2050년에 3백만 명을 넘어설 것으로 예상된다.

또한, 치매는 환자 본인 뿐 아니라 보호자에게도 가족관계가 악화되거나 신체적, 경제적 부담을 갖는 등의 다양한 부양스트레스를 일으키는 것으로 나타났다. 더욱

이 치매노인을 부양하는 보호자의 경우 일반 노인이나 신체적 어려움, 다른 질병을 가지고 있는 환자의 보호자에 비해 보다 많은 시간적, 육체적 노동을 통해 일상생활에 영향을 받고 신체적, 정신적 부담과 더불어 경제적 지원까지 떠맡아야하기 때문이다.

하지만, 이러한 어려움에도 불구하고 치매가 진단된 이후 인지기능의 저하를 이전 수준으로 회복시킬 수 있는 치료법은 아직 없으며, 오직 치매의 위험 요인을 사전에 조절하여 발병률을 낮추고, 최대한 이른 시점에서의 진단과 그에 맞는 적절한 치료를 통해 치매의 경과속도를 저하시키는 방법만이 현실적으로 시행 가능한 대응책이라 볼 수 있다.

현재 치매는 일반적인 기능의 상실을 억제하기 위한 약물 치료에 의존하는 경우가 대부분이지만, 그 효과는 제한적이라고 할 수 있다. 최근 많은 연구들에서 운동이 뇌의 노화에 긍정적인 영향을 주는 것과 함께 인지기능 장애와 치매의 예방과 예후에도 효과가 있다는 것에 초점을 맞추고 있으며, 운동은 노화로부터 노쇠, 장애 및 사망률을 줄이는 중요한 수단으로 인식되어지고 있다. 또한 운동은 치매에 걸린 사람들의 기능적인 능력, 신체기능, 및 신경 정신과 같은 증상들을 개선하는 것으로 알려져 있으며, 더불어 노인이 운동에 참여하는 비율이 점차 증가함에 있어 치매환자들이 운동에 참여하였을 경우의 위험성 또한 무시할 수 없게 되었다.

<연령대별 생활체육 참여율 및 참여 빈도>

(대상 : 응답자 전체, 단위 : %, %p)

구분	전혀 안 함	한 달 3번 이하	주 1회	주 2회	주 3회	주 4회	주 5회	주 6회	주 7회	주 1회 이상		차이(%p)
										17년	18년	
전체	28.0	9.8	9.8	11.8	18.2	7.3	8.1	2.4	4.6	59.2	62.2	3.0
남성	25.5	13.0	12.9	13.0	15.4	6.6	7.2	2.5	3.9	60.1	61.6	1.5
여성	30.5	6.7	6.7	10.5	21.1	8.0	8.9	2.4	5.3	58.4	62.8	4.4
10대	32.9	9.9	12.9	11.9	14.2	4.9	8.7	2.0	2.6	60.4	57.2	-3.2
20대	30.8	10.0	6.6	13.2	18.7	7.0	9.0	2.8	1.9	55.2	59.3	4.1
30대	29.6	8.5	9.2	13.0	17.4	9.6	8.7	2.1	2.0	60.7	61.9	1.2
40대	23.5	10.8	13.5	11.5	20.8	7.0	8.2	2.2	2.4	60.4	65.7	5.3
50대	22.3	13.2	11.3	11.0	20.4	7.2	7.8	2.0	4.8	60.4	64.5	4.1
60대	27.8	8.6	8.7	11.9	16.5	8.5	8.1	1.8	8.1	61.7	63.6	1.9
70대 이상	34.4	5.8	4.1	9.1	16.9	6.1	5.5	4.4	13.6	54.8	59.8	5.2

그림 3. 2018년 국민생활체육 참여 실태조사 결과 [문화체육관광부 제공]

이를 토대로 치매환자의 보다 안전한 운동과 치매노인을 간호하는 데 있어 가장 어려움을 주는 정신행동증상 중 하나인 배회에 대한 관리가 용이하며, 동시에 보호자의 비교적 자유로운 행동반경에 도움을 주는 어플리케이션과 웨어러블 장치인 스마트밴드를 만들고자 한다. 배회는 치매노인 당사자에게 체중감소, 언어능력 저하의 가속화, 낙상과 골절 등 사고위험성 노출을 포함한 단순한 사고 및 부정적 결과에서부터 조기 사망에 까지 이르게 되는 치명적인 결과를 초래할 뿐 아니라, 간호

를 제공하는 가족 및 전문 의료인들에게도 매우 우려스럽고 많은 시간이 소요되며, 지치게 만드는 증상으로 인식됨으로써 신체적 · 정신적으로 여러 심각한 영향을 미친다.

따라서 환자의 위치를 보호자가 실시간으로 확인할 수 있도록 하여 운동 및 배회에 대한 보호자의 걱정과 부담을 줄이고 치매 개선에 긍정적인 효과를 불러올 수 있는 어플리케이션을 개발할 필요성이 있다. 또한 웨어러블 장치(스마트 밴드)에 다양한 기능을 구현하여 약물 치료의 효율성을 높임과 동시에 환자 관리에 용이함을 제공할 수 있다.

2. 기존 연구/기술동향 분석

Samsung Health	
기능	만보기를 비롯한 다양한 활동 관리 가능
	식사, 수면 등 자신에게 알맞은 목표 설정 가능
	단계별 운동 프로그램을 설정하고 실행 가능

LOMY	
기능	NFC 기능으로 팔찌에 사용자의 정보(사진, 이름, 보호자의 연락처 등)을 입력하고 수정할 수 있음
	NFC 태그 시 사용자의 정보 확인 가능

37도 스마트 밴드	
기능	호흡, 기분변화, 수면패턴 모니터링
	혈압, 피로도, 심장박동 모니터링
	헬스 만보계

세 가지의 스마트 밴드 모델을 비교 및 분석 해 본 결과 Samsung Health와 LOMY의 경우 보호자 어플리케이션과의 연동 및 상호작용이 없어 보호자가 관리할 수 없다는 불편점이 있고 37도 스마트밴드의 경우에는 보호자 어플리케이션과 연동이 되어 있지만 자동 업데이트 시스템이 없어 사용자가 수동으로 업데이트를 해야 한다는 불편점이 있다.

따라서 환자의 스마트밴드를 통해 환자의 심박수를 수집하고 전달하여 서버에 저장한다. 보호자의 어플리케이션으로 수집한 정보를 보내고 시간을 정해 자동으로 환자의 정보를 보호자의 어플리케이션으로 업데이트를 시키는 기능을 구현하여 현재 있는 제품들과의 차별성을 둘 예정이다.

3. 개발 목표

환자의 위치를 시각화하여 보호자에게 보여줌으로써 치매환자의 운동과 배회에 관한 보호자의 불안감을 줄이고 환자가 복용하는 약에 대한 알림을 줌으로써 약을 꾸준히 섭취할 수 있게 하여 환자가 가지고 있는 합병증에 대한 예방을 높이고 약물치료의 효율성을 높이며 진행을 늦춘다.

또한 어플리케이션을 통해 환자의 운동량을 관리함으로써 환자 본인이 자신의 운동량 추이를 확인할 수 있고 치매 진행을 늦추는 것에 도움을 주는 것을 목표로 한다.

4. 팀 역할 분담

	이준규	이영주	신정희
자료수집	- 치매 환자 치료방법 - 서버 관련 API	- 관련 프로젝트 탐색 - 블루투스 관련 수집	- Smart Band 정보
설 계	- 어플리케이션 관련 기능 알고리즘 설계	- 블루투스 연동과 서버 설계	- 스마트 밴드(하드웨어)관련 설계
구 현	- 어플리케이션 위치 구현 - 어플리케이션 운동량 구현	- 스마트 밴드와 어플리케이션 관련 블루투스 기능 구현 - 어플리케이션과 서버 연결 기능 구현	- Smart Band 진동 센서 구현 - Smart Band 심박 수 구현(어플에 심박수 전달)
테 스 트	- 스마트 밴드에서 데이터 정보(심박 수, 운동량)정보 전송 테스트 - 어플리케이션과 서버 연동 테스트 - 통합테스트 및 유지 보수		

5. 개발 일정

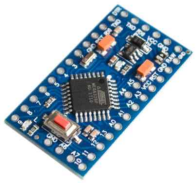
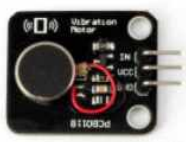
항목	추진사항	12월	1월	2월	3월	4월	5월	6월	7월	8월
요구사항 정의 및 분석	- 요구사항 정의 및 분석 - 요구사항 명세									
시스템 설계 및 상세설계	- 시스템 설계 - 상세 설계									
구현	- 코딩									
시험 및 데모	- 스마트 밴드 및 어플리케이션 시험 - 통합 시험 - 완전성 보강									
문서화 및 발표	- 중간보고서 작성 (중간보고서 및 사용자 매뉴얼 작성)									
최종 보고서 작성 및 패키징	- 최종 보고서 작성 및 패키징 (문서 사용법, 프로그램, 데모 동영상 등)									

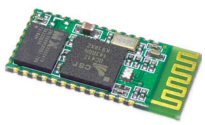
6. 개발 환경

- 소프트웨어 개발 환경

안드로이드 스튜디오	MySQL	아두이노 IDE	Apache Tomcat
언어 : JAVA	데이터베이스	언어 : C++	버전 : 9.0 언어 : PHP

- 하드웨어 개발 환경

				
아두이노 프로 미니 3.3V	리튬폴리머 배터리 모음 3.7V	리튬 배터리 충전 모듈	심장박동 측정 센서 DM447	진동 모터 모듈 DM159
동작전압 : 3.3V 클럭 주파수 : 8MHz	전압 : 3.7V	입력전압 : 5V USB TYPE : C타입	공급전압 : 5V	동작전압 : 5V


HC-06 블루투스 모듈
신호 범위 : 약 10m

Ⅱ. 본론

1. 개발 내용

- 어플리케이션

<보호자>

- 치매 환자에 대한 위치 정보를 어플리케이션을 통해 확인함으로써 환자에 대한 불안감을 줄이고, 실제 실종률을 감소시키는 것을 목표로 한다.
- 기간에 따른 치매 환자의 진행률을 짐작하고, 그에 대한 방안을 빠르게 준비할 수 있도록 한다.
- 웹 서버를 이용하여 환자와 연동 기능 구현
- 웹 서버와 연동하여 환자의 실시간 위치 확인 기능 구현
- 웹 서버와 연동하여 환자의 오늘의 운동량 일정 주기에 대한 운동량에 대한 평균치를 계산하여 확인 기능 구현
- 환자가 복용하는 약이 추가 되었을 경우 보호자가 환자에게 복용해야할 약 리스트를 서버로 전달 기능 구현
- 환자의 실시간 심박수 확인 기능 및 응급 상황에 대한 확인 기능 구현

<환자>

- 일상생활 중 갑작스러운 치매 증상이 나타날 경우, 스마트 밴드를 통해 주변 사람들이 환자의 간단한 정보를 알 수 있게 한다.
- 치매 환자의 경우, 합병증을 가지기 쉬우므로 그에 관련된 약을 어플리케이션에 등록시켜 해당하는 약의 복용 시간에 알림이 설정되게 한다.
- 스마트밴드와 블루투스로 연동하여 심박수 정보를 실시간으로 체크 기능 구현
- 자신의 개인정보 입력과 서버에 개인정보 및 보호자 정보 입력 및 전달 기능 구현
- 환자의 복용 약과 복용시간을 설정 및 보호자로부터 약의 정보를 수신 기능 구현
- 스마트 폰 내의 GPS를 이용하여 현재위치 확인 및 서버로 현재 위치(주소, 위도, 경도)정보 및 서버로 전달 기능 구현
- 스마트 폰 내의 자이로센서를 이용하여 하루의 운동량(걸음 수)측정 및 서버로 데이터 전달 기능 구현

- 스마트 밴드
  - 스마트 밴드 내의 심박 수 센서를 이용하여 환자 정보를 어플리케이션으로 보내는 기능 구현
  - 환자 애플리케이션에서 받은 정보를 통해 환자의 약 복용 시간에 맞춰 제때 약을 복용할 수 있는 알림을 주는 역할

- 서버 및 DB
  - 아마존 AWS의 EC2를 사용하여 Apache Tomcat 9.0 웹 서버 구축해 서버 내 MySQL DB 연동
  - 환자 어플리케이션의 개인정보를 수신하여 보호자 어플리케이션으로 전달하는 기능 구현
  - 환자 어플리케이션에서 심박, 걸음 수, 블루투스 연결 상태 정보, 위치 정보를 수신하여 저장 기능 및 보호자 어플리케이션으로 전달기능 구현
  - 일정기간이 지났을 때(1주, 1달, 3개월 등), 그 기간 동안의 데이터(환자 어플리케이션에서 받아온 정보)의 평균치를 구하고 새 평균치 저장 기능 구현 및 이전 데이터 삭제기능 구현
  - 데이터베이스 테이블

    patient(p_no, p_name, p_phone, p_birth, g_phone, g_info)
    guardian(g_no, g_name, g_birth, p_no)
    distance(p_step, p_distance, d_time, p_no)
    heartrate(p_heart, d_time, p_no)
    medicine(m_name, m_num, m_time, p_no)

2. 문제 및 해결방안

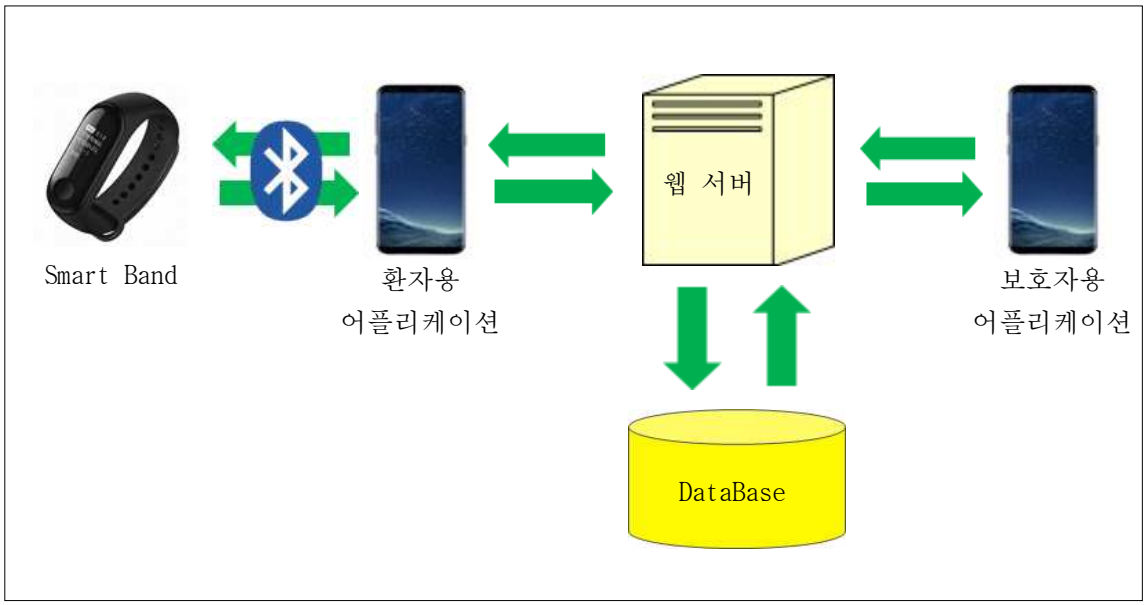
Q1) 웨어러블 장치의 구현에 있어 아두이노 우노를 쓸 경우 웨어러블 장치의 크기가 너무 커져 효율성이 떨어짐

A1. 아두이노 우노가 아닌 아두이노 프로 미니를 통해 만들고자 하는 스마트 밴드의 전체적인 크기를 줄임 또한 웨어러블 장치의 디자인과 크기보다는 기능 구현에 중점을 둠

Q2) 블루투스 연결이 해제 되었을 경우에 대한 처리 방안을 어떻게 할 것인가

A2. 스마트폰과 연결할 때 스마트폰에 블루투스 장치(스마트 밴드)에 대한 데이터를 넘어주고 그 데이터를 통해 현재 연결 상태가 connect 상태인지 disconnect 상태인지에 대한 정보를 보호자에게 전달한다.

3. 시험 시나리오



- 환자용 어플리케이션과 보호자용 어플리케이션을 환자와 보호자 폰에 각각 설치한다.
- 환자용(혹은 보호자용) 어플리케이션에서 보호자 등록번호(혹은 환자 등록번호)를 입력하여 각 어플리케이션을 연결한다.
- 블루투스를 통해 환자의 어플리케이션과 스마트 밴드를 연동한다.
- 스마트 밴드를 통해 심박 수를 측정한다.
- 스마트 밴드에서 측정한 정보를 환자 어플리케이션으로 전달한다.
- 어플리케이션에 복용 약에 대한 시간을 설정하여 해당하는 시간에 진동을 통해 환자에게 복용 시간을 알린다.
- 앞서 말한 환자의 위치, 심박 수, 복용 약 그리고 어플리케이션에 저장된 환자의 걸음 수 정보를 서버에 올려 치매 환자의 운동량 추이를 비교하고 통계치를 계산한다.
- 서버에 저장된 내용들을 보호자 어플리케이션에 전달한다.
- 보호자는 환자의 위치 정보, 운동량의 추이와 현재 환자의 심박 수를 확인 가능하다.
- 또한 보호자의 어플리케이션에서 환자의 복용 약에 대한 입력이 가능하다.

4.상세 설계

1) Device

```
void setup()
{
  Serial.begin(115200);    //serial 모니터
  bt.begin(9600);          //블루투스
  interruptSetup();                  //2ms 마다 인터럽트 발생 심박 측정
}
```

void setup()

- 블루투스 통신과 인터럽트를 위한 setup() 함수

```
void interruptSetup()
{
  TCCR2A = 0x02;      //모드설정 : CTC 모드 -> 타이머 카운터 모드
  TCCR2B = 0x06;      //
  OCR2A = 0X7C;       // CTC 모드 설정시, 124와 같아질 때 인터럽트 발생
  TIMSK2 = 0x02;      // 사용할 타이머 인터럽트 ENABLE.
  sei();              // MAKE SURE GLOBAL INTERRUPTS ARE ENABLED
}
```

void interruptesetup()

- 2ms 마다 심박을 측정할 수 있게 설정하는 타임인터럽트 함수

```
ISR(TIMER2_COMPA_vect) //인터럽트가 실행될 때 넘어오는 함수
// Timer2가 124로 카운트 될 때 트리거 됨
{
  cli();                                      // 외부 영향 없애기
  Signal = analogRead(pulsePin);              // 심박수 읽음(핵심)
  sampleCounter += 2;                         // ms 단위
  int N = sampleCounter - lastBeatTime;       // IBI 측정 후 경과 시간

  if(Signal < thresh && N > (IBI/5)*3) // 최저점 찾는 함수
    {
      if (Signal < T) // T is the trough
      {
        T = Signal; // keep track of lowest point in pulse wave
      }
    }

  if(Signal > thresh && Signal > P) //최고점 찾는 함수
    {
      P = Signal;
    }
```

인터럽트 발생 시 실행되는 함수, 심박 signal을 읽어온다.

```
if (Signal < thresh && Pulse == true) //심박에서 비트가 끝나고?? 순간 쉬는 시간일때?
  {
    Pulse = false;
    amp = P - T;                              // 진폭 구하기
    thresh = amp/2 + T;                       // thresh 재설정
    P = thresh;
    T = thresh;
  }

if (N > 2500)
  {                             //2.5초 이상 비트가 측정되지 않으면 초기화
    thresh = 512;
    P = 512;
    T = 512;
    lastBeatTime = sampleCounter;
    firstBeat = true;
    secondBeat = false;
  }
```

읽어온 심박 signal의 10개의 값을 받아와 평균내어 오차를 줄이고 1분의 심박수를 측정하는 함수

```
    for(int i=0; i<=8; i++)
      {

        rate[i] = rate[i+1];                  // 가장 오랜된 값 버림
        runningTotal += rate[i];
      }

    rate[9] = IBI;                            // 가장 최신 IBI값 넣음
    runningTotal += rate[9];
    runningTotal /= 10;                       // IBI값 평균
    BPM = 60000/runningTotal;  // 1분의 BPM 측정
    QS = true;
  }
}
```

값들을 재설정하여 다시 심박수를 잴 수 있게 만들어 주는 함수

```
void loop(){
  if (QS == true) // 심박수를 찾은 경우
    {
     serialOutputWhenBeatHappens();    //시리얼 확인 + 앱에 BPM 수신
      QS = false;
    }
    delay(1000);
}
```

아두이노의 loop함수로 스마트밴드의 기본 사이클 담당

```
void serialOutputWhenBeatHappens()
{
 if (serialVisual == true) //
   {
     if(BPM<50||BPM>130){
     // Serial.println("BPM..");//시리얼 확인용
      bt.println(BPM); //안드로이드로 보내기 // 좀 더 보완할 예정
     }
     else{
      // Serial.println(BPM); //시리얼 확인용
      bt.println(BPM); //안드로이드로 보내기
     }
   }
```

측정된 BPM값을 어플리케이션으로 송신

2) 환자용 Application 부분

```
@Override
public void onSensorChanged(SensorEvent event){
    Calendar cal=Calendar.getInstance();
    int Hour=cal.get(Calendar.HOUR_OF_DAY);
    int minute=cal.get(Calendar.MINUTE);
    int second=cal.get(Calendar.SECOND);
    Log.v( tag: "시간",Integer.toString(minute));
    Log.v( tag: "시간",Integer.toString(Hour));
    if(event.sensor.getType()==Sensor.TYPE_STEP_COUNTER){
        walkCount++;
        if(Hour==0&&minute==0&&second<=10){
            walkCount=0;
        }
        WalkView.setText("걸음수 :"+Integer.toString(walkCount));

    }
}
```

void onSensorChanged(SensorEvent event)

SensorEventListener 내에 있는 가상 메서드로 정의 되어 있는 메서드로 센서가 사용 될 때 마다 호출 되는 함수이다 반환 값은 없으며 핸드폰 내에 센서 사용이감지 될 때 마다 사용되는 call back 함수 걸음이 감지 될 때마다 사용되어 걸음 수 1씩 증가시켜주는 메서드

```
private void checkPermission() {
    boolean fineLocationRationale = ActivityCompat.shouldShowRequestPermissionRationale( activity: this, Manifest.permission.ACCESS_FINE_LOCATION);
    int hasFineLocationPermission = ContextCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION);
    if (hasFineLocationPermission == PackageManager.PERMISSION_DENIED && fineLocationRationale)
        showDialogForPermission( msg: "위치정보 승인 해야함");
    else if (hasFineLocationPermission == PackageManager.PERMISSION_DENIED && !fineLocationRationale) {
        showDialogForPermissionSettings( msg: "위치 정보 승인 거부");
    } else if (hasFineLocationPermission == PackageManager.PERMISSION_GRANTED) {
        //퍼미션 허용
        if (googleApiClient.isConnected() == false) {
            //googleapi와 연결이 안되어있다면
            googleApiClient.connect();
        }
    }
}
```

void checkPermission()

GPS 연결을 위한 권한 허용 및 인터넷 접속을 위한 접근허용을 위한 메서드

```
@Override
public void onMapReady(final GoogleMap map) {//지도의 초기세팅
    googleMap = map;
    //런타임 퍼미션 요청 or 활성 요청 대화상자 보이기전 초기위치로 이동
    setDefaultLocation();
    googleMap.getUiSettings().setMyLocationButtonEnabled(true);//자기위치버튼활성화
    googleMap.animateCamera(CameraUpdateFactory.zoomTo( v: 15));//카메라 이동할수있게 활성화
    googleMap.setOnMyLocationButtonClickListener(() -> {
            mMoveMapByAPI = true;//위치에 따른 카메라 이동 활성화
            return true;
    });
    googleMap.setOnMapClickListener((latLng) -> { //맵클릭 });
    googleMap.setOnCameraMoveStartedListener((i) -> {
            if (mMoveMapByUser == true && mRequestingLocationUpdates) {
                mMoveMapByAPI = false;//위치에 따른 카메라 이동 비활성화
            }
            mMoveMapByUser = true;//사람이 움직일수 있게해줌
    });
```

void onMapReady(final GoogleMap map)

지도의 초기세팅을 위한 메서드

```
@Override
public void onConnected(Bundle bundle) {
    if (mRequestingLocationUpdates == false) {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            int hasFineLocationPermission = ContextCompat.checkSelfPermission( context: this,
                    Manifest.permission.ACCESS_FINE_LOCATION);
            if (hasFineLocationPermission == PackageManager.PERMISSION_DENIED) {
                ActivityCompat.requestPermissions(mActivity, new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, PERMISSIONS_REQUEST_ACCESS_FINE_LOCATION);
            } else {
                //퍼미션을 가지고 있고 현위치 업데이트함수 호출
                startLocationUpdates();
                googleMap.setMyLocationEnabled(true);//현재위치 활성화
            }
        } else {
            //현재위치 업데이트
            startLocationUpdates();
            googleMap.setMyLocationEnabled(true);
        }
    }
}
```

void onConnected(Bundle bundle)

초기 실행에서 퍼미션허용이 다 되었는지 혹은 이후 실행에서 퍼미션 허용이 다 되었는지를 확인 하고 위치 정보 네트워크 정봐 다 켜져있는지 확인하는 메서드

```
public void setDefaultLocation() {
    mMoveMapByUser = false;//사용자가 이동 불가
    LatLng DEFAULT_LOCATION = new LatLng( v: 37.56,  v1: 126.97);//위치 서울로 설정(GPS  안켜져있거나 인터넷 연결이 안되 구글 api를 가져올수 없음)
    String markerTitle = "위치정보 가져오기 불가";
    String markterSnippet = "위치 퍼미션과 GPS 활성 요부 확인";

    if (currentparker != null) currentparker.remove();//마커가 있다면 마커 제거
    MarkerOptions markerOptions = new MarkerOptions();
    markerOptions.position(DEFAULT_LOCATION);
    markerOptions.title(markerTitle);
    markerOptions.snippet(markterSnippet);
    markerOptions.draggable(true);
    markerOptions.icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_RED));
    currentparker = googleMap.addMarker(markerOptions);

    CameraUpdate cameraUpdate = CameraUpdateFactory.newLatLngZoom(DEFAULT_LOCATION,  v: 15);
    googleMap.moveCamera(cameraUpdate);
}
```

void setDefaultLocation()

초기 설정을 위하거나 퍼미션중 오류가 생겼을 경우 호출되는 함수 초기 위치 설정을 GPS 상의 위치로 하는 것이 아니라 한 지정된 곳으로 위치를 설정함

```
public void setCurrentLocation(Location location, String markerTitle, String markerSnippet) {
    mMoveMapByUser = false;
    //사용자가 맵움직일수 없게 변경
    if (currentparker != null) currentparker.remove();
    LatLng currentLatLng = new LatLng(location.getLatitude(), location.getLongitude());
    MarkerOptions markerOptions = new MarkerOptions();
    markerOptions.position(currentLatLng);//마커의 위치
    markerOptions.title(markerTitle);//마커를 눌렀을때 정보창에 표시되는 문자열
    markerOptions.snippet(markerSnippet);//제목 아래에 표시되는 추가 텍스트
    markerOptions.draggable(true);//마커이동가능
    //이에 추가로 Icon//visibility/Flat or Billboard(평면 또는 빌보드 방향)
    //lotation(회전),zindex(그리기순서),Tag
    currentparker = googleMap.addMarker(markerOptions);
    if (mMoveMapByAPI) {
        Log.d(TAG,  msg: "setCurrentLocation:mGoogleMap moveCamera" + location.getLatitude() + " " + location.getLongitude());
        CameraUpdate cameraUpdate = CameraUpdateFactory.newLatLng(currentLatLng);//카메라 위치변경
        googleMap.moveCamera(cameraUpdate);
    }

}
```

void setCurrentLocation(Location location,String markerTitle,String markerSnippet)

현재 위도와 경도를 위치하여 현재 위치를 설정해주는 메서드

```
private String getCurrentAddress(LatLng latlng) {//현재 주소 반환
    //지오코더 GPS를 주소로 변환
    Geocoder geocoder = new Geocoder( context: this, Locale.getDefault());
    List<Address> addresses = null;
    try {
        addresses = geocoder.getFromLocation(latlng.latitude, latlng.longitude,  maxResults: 1);


    } catch (IOException ioException) {
        Toast.makeText( context: this,  text: "지오코더 서비스 사용불가", Toast.LENGTH_LONG).show();
        return "지오코더 서비스 사용 불가";
    } catch (IllegalArgumentException illegalArgumentException) {//부적절 인자를 받았을때
        Toast.makeText( context: this,  text: "GPS 좌표 오류", Toast.LENGTH_LONG).show();
        return "잘못된 GPS 좌표";
    }
    if (addresses == null || addresses.size() == 0) {
        Toast.makeText( context: this,  text: "주소 미발견", Toast.LENGTH_LONG).show();
        return "주소 미발견";
    } else {
        Address address = addresses.get(0);
        return address.getAddressLine( index: 0).toString();
    }
}
```

String getCurrentAddress(LatLng latlng)

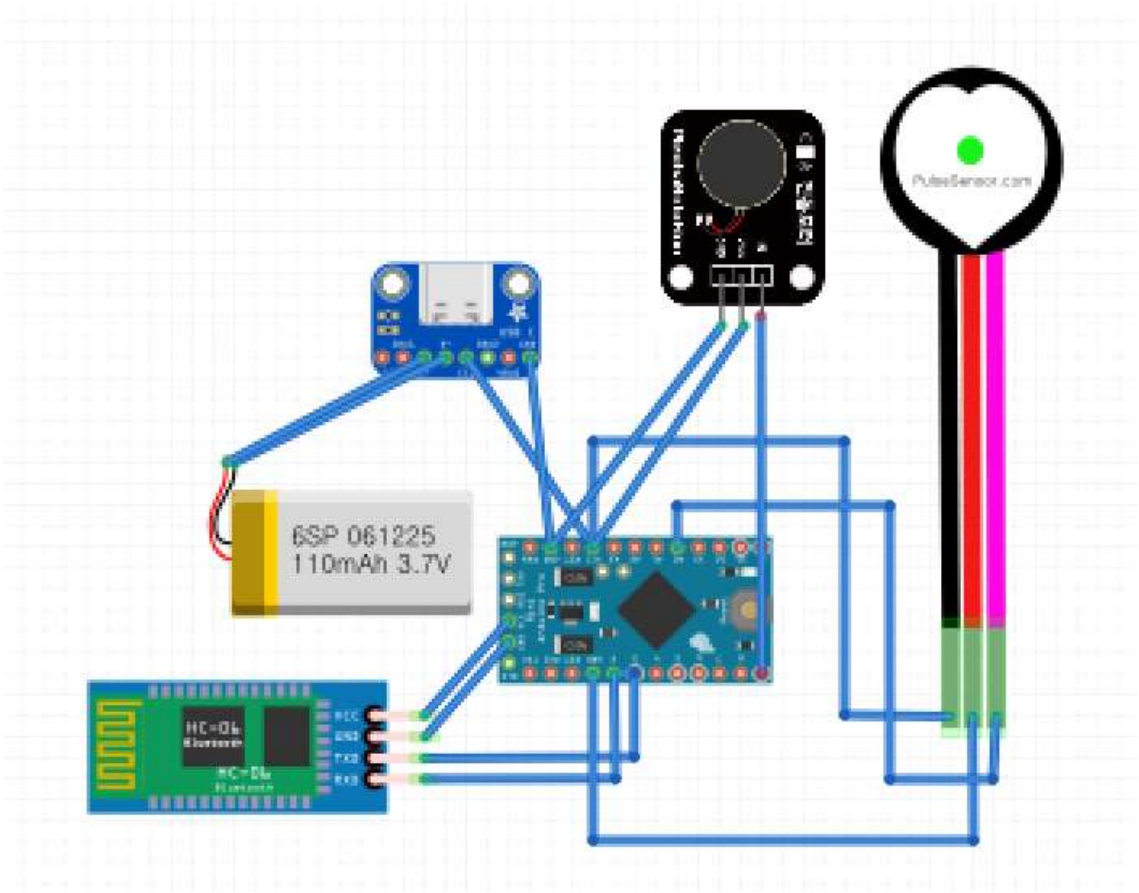
인자로부터 현재 위치의 위도와 경도를 받아와서 Geocoder를 이용해 위도와 경도를 주소로 변환해주는 메서드 이후 주소를 String으로 변환

```
@Override
public void onLocationChanged(Location location) {//GPS 위치가 변경되었을때 오버라이딩
    currentPositon = new LatLng(location.getLatitude(), location.getLongitude());//위도와 경도 얻어옴
    String markerTitle = getCurrentAddress(currentPositon);
    String markserSnippet = "위도:" + String.valueOf(location.getLatitude()) + "경도 :" + String.valueOf(location.getLongitude());
    setCurrentLocation(location, markerTitle, markserSnippet);
    mCurrentLocation = location;//최근위치 변경
    prelat = mCurrentLocation.getLatitude();
    prelon = mCurrentLocation.getLongitude();
}
```

void onLocationChanged(Location location)

GPS 위치가 변경되었을 때 호출되는 콜 백 메서드 위치 정보를 수정하고 위도와 경도 변경

3) 회로도(아두이노)

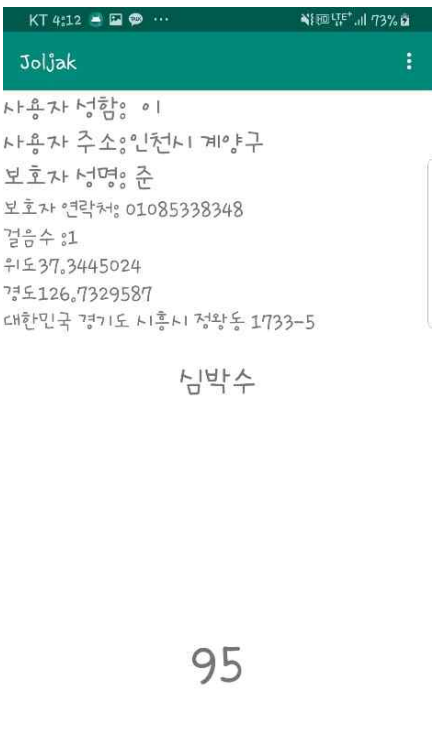


5. Prototype 구현

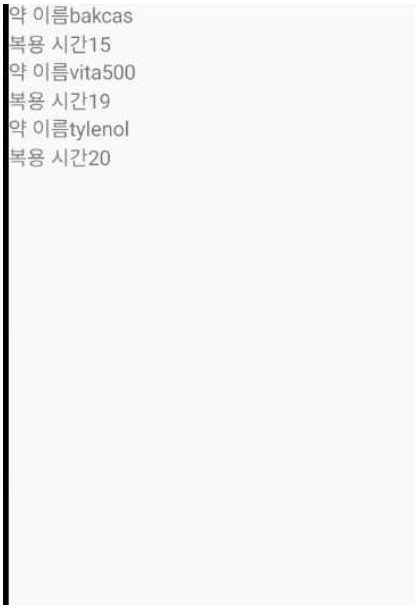
1) 환자용 어플리케이션

- 로그인 및 회원가입 화면

- 메인 화면(위치, 걸음수 및 심박수 표시)

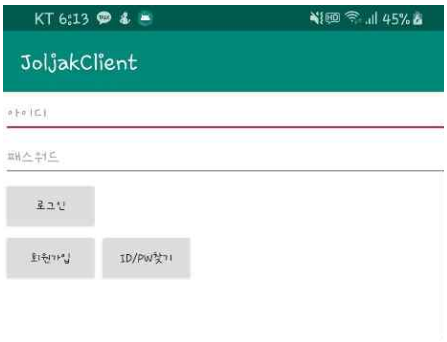


- 복용 약 설정 화면

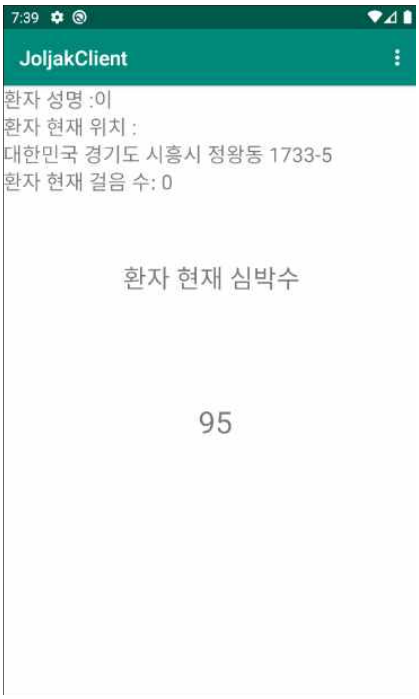


2) 보호자용 어플리케이션

- 로그인 화면



- 환자 정보 확인 화면(위치)



3) 환자 앱

- 포그라운드

6. 시험/테스트 결과

1) 환자용 어플리케이션

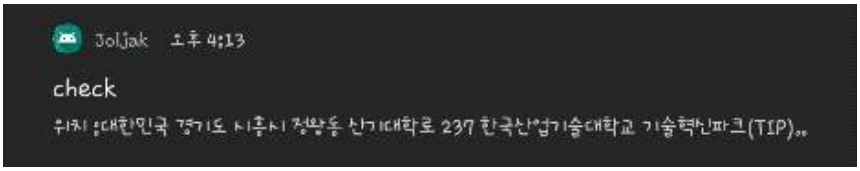
- 회원가입

자신의 신상 정보를 기반으로 회원가입 시 데이터베이스에 입력이 된다.

- 위치 확인 화면

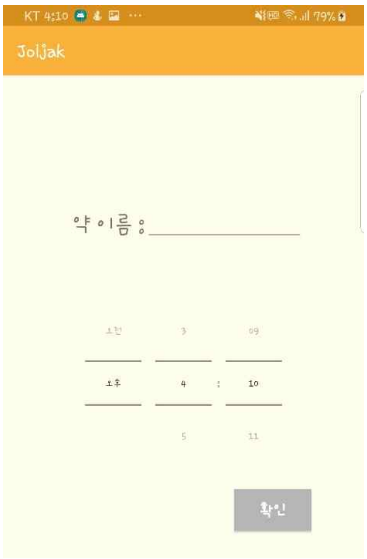
구글 API를 통해 자신의 위치를 지도로 볼 수 있다.

- 포그라운드



Notification에서 구글 API를 통해 얻은 자신의 위치를 확인할 수 있다.

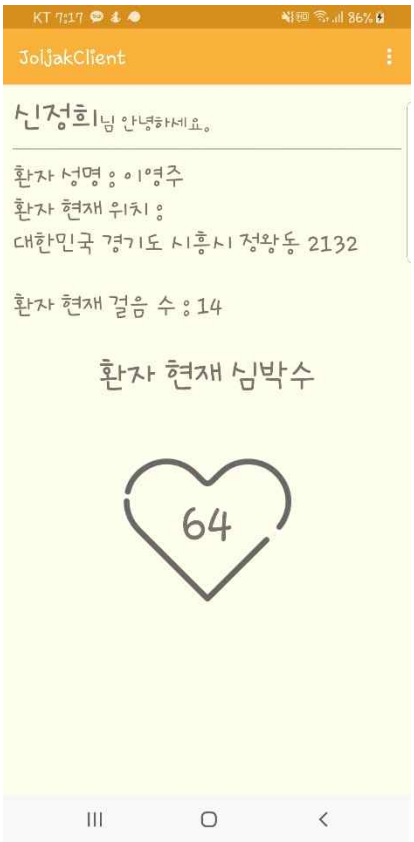
- 약 복용 시간 알림 설정



약 이름과 시간을 지정하면 해당 시간에 아두이노에서 진동이 울림

2) 보호자용 어플리케이션

- 메인 화면

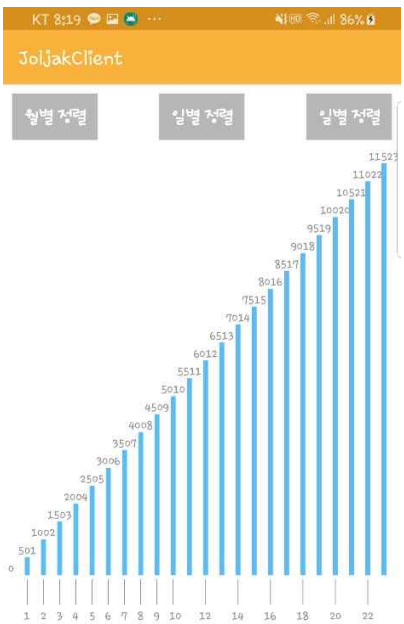


로그인 시 연결된 환자의 심박 수와 위치(현 위치) 확인 가능

- 위치 확인 화면

서버의 저장된 환자의 위치를 받아와 지도로 표시한다.

- 운동량 확인



통계 버튼을 클릭했을 때 환자의 운동량에 대한 그래프를 확인할 수 있다.
(위 그림은 일별 정렬을 나타낸 그래프이다.)

* 위 그래프는 실제로 모든 데이터를 측정할 수 없어 임의로 조작하여 넣은 데이터로 만들어낸 그래프이다.

3) 하드웨어

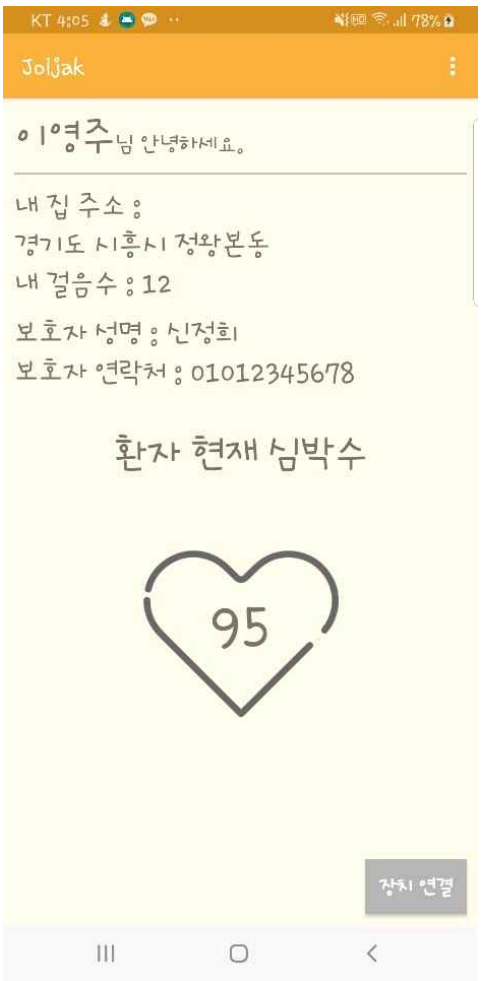
- 심박수 측정

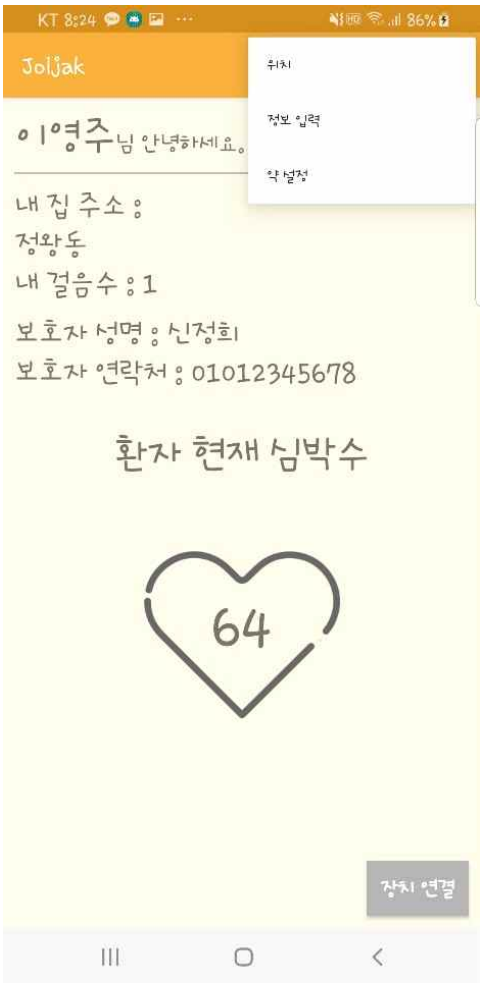
```
BPM: 74
BPM: 73
BPM: 75
BPM: 77
BPM: 78
BPM: 79
BPM: 78
BPM: 78
BPM: 79
can't find BPM
BPM: 68
BPM: 64
can't find BPM
can't find BPM
```

측정된 심박 수를 표시하고 심박을 측정하지 않을 경우에는 “can’t find BPM”이라는 문구가 출력된다.

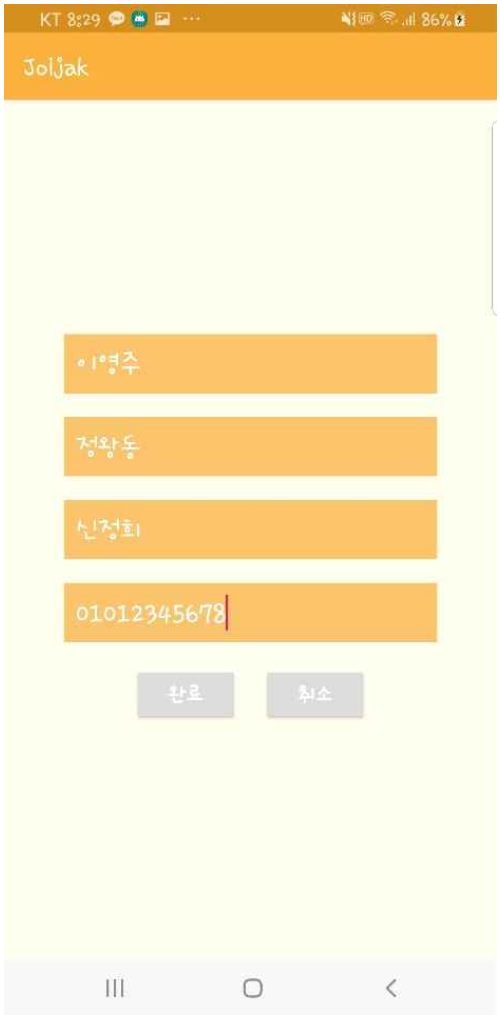
7. Coding & DEMO

1) 환자용 어플리케이션

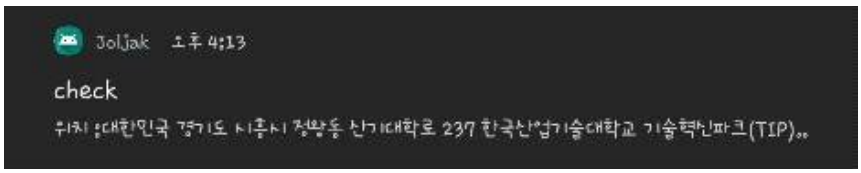




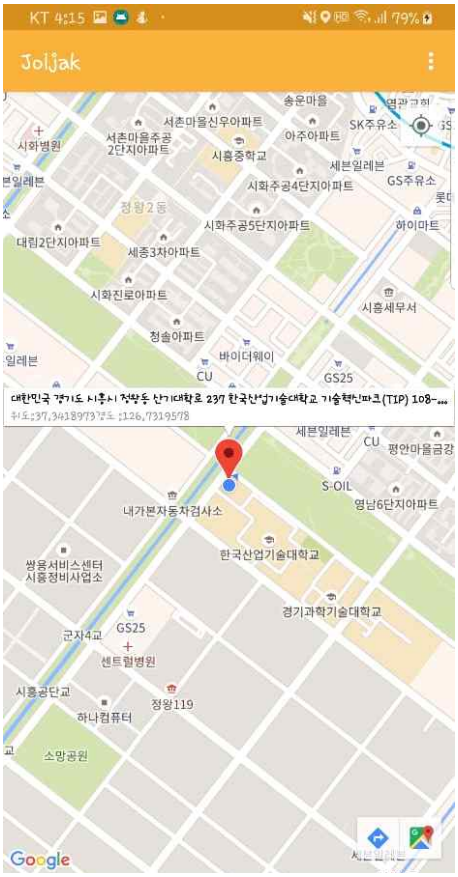
메인 화면에서 자신의 정보와 보호자의 개인정보를 알 수 있다. 또한 자신의 심박수를 확인할 수 있고 장치 연결 버튼을 통해 블루투스 장치와 연결할 수 있다. 메뉴 버튼을 클릭하면 오른쪽과 같은 창이 뜬다.



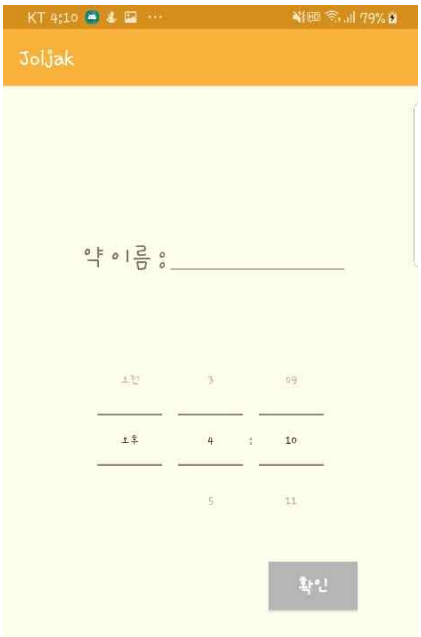
정보 입력 메뉴를 통하여 환자의 정보를 입력하여 어플을 사용할 수 있게 한다.



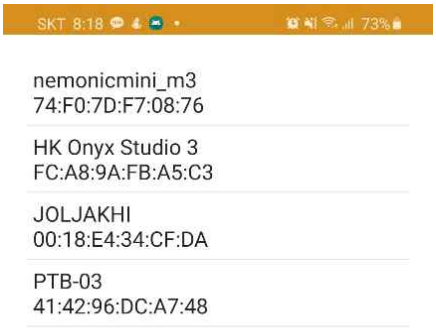
지도로 환자의 위치를 알 수 있을 뿐만 아니라 Notification으로 어플에 들어갈 필요 없이 환자의 위치와 걸음수를 알 수 있다.



위치 메뉴를 클릭했을 때 자신의 위치를 지도로 확인 가능하다.

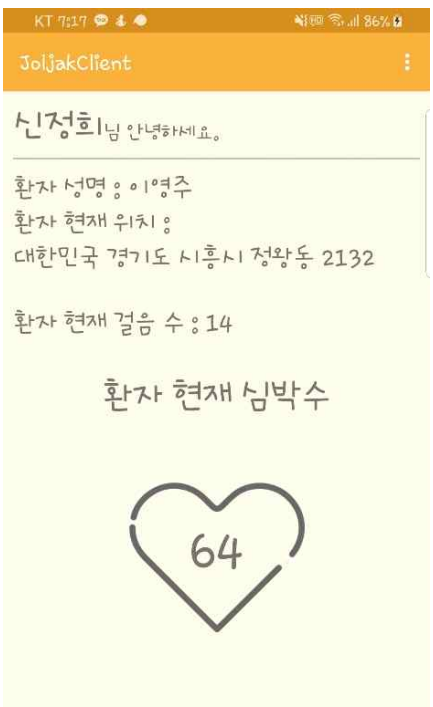


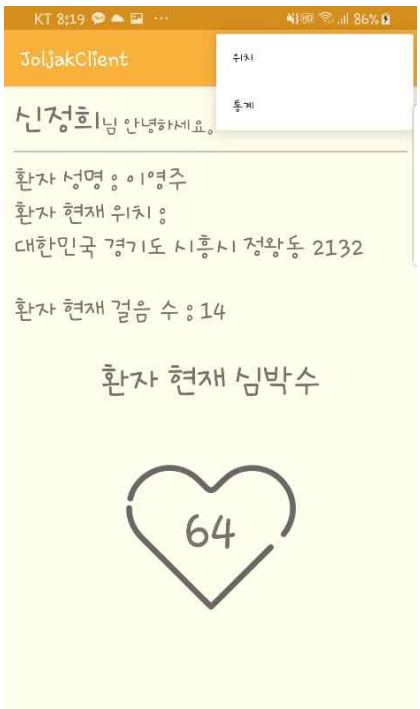
약 설정 메뉴를 통해 복용 약의 이름을 정하고 복용 시간을 설정하여 시간이 되면 하드웨어에서 진동이 울리도록 한다.



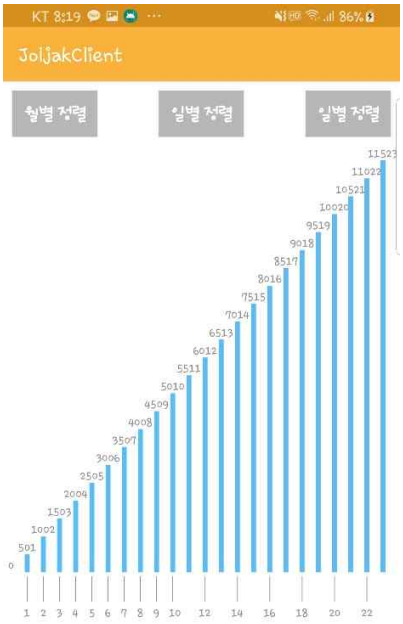
장치 연결 버튼을 통해 블루투스 연결할 device를 찾을 수 있다.

2) 보호자용 어플리케이션





메인 화면에서 환자의 위치와 심박 수를 확인할 수 있다. 메뉴 버튼을 클릭하게 되면 오른쪽과 같은 창이 뜬다.

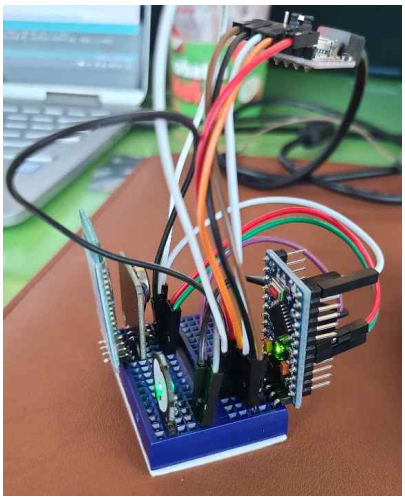


통계 버튼을 클릭했을 때 환자의 운동량에 대한 그래프를 확인할 수 있다.
(위 그림은 일별 정렬을 나타낸 그래프이다.)

* 위 그래프는 실제로 모든 데이터를 측정할 수 없어 임의로 조작하여 넣은 데이터로 만들어낸 그래프이다.

위치 버튼을 클릭했을 때 환자의 현재 위치를 지도로 볼 수 있다.

3) 하드웨어



핸드폰과 블루투스로 연결되어 설정된 알람 시간에 진동이 울리고 심박센서를 통해 환자의 심박 수를 측정하고 블루투스를 통해 측정된 수를 핸드폰에 전송한다.

4) 데이터베이스

테이블	실행	행	종류	데이터정렬방식	크기
distance	보기 구조 검색 삽입 비우기 삭제	287	InnoDB	utf8mb4_general_ci	32.0 KB
location	보기 구조 검색 삽입 비우기 삭제	2	InnoDB	utf8mb4_general_ci	16.0 KB
medicine	보기 구조 검색 삽입 비우기 삭제	0	InnoDB	utf8mb4_general_ci	16.0 KB
patient	보기 구조 검색 삽입 비우기 삭제	2	InnoDB	utf8_general_ci	32.0 KB
4개 테이블 계		291	InnoDB	utf8mb4_general_ci	96.0 KB

데이터베이스는 총 4개의 테이블이 생성되었다. patient 테이블은 환자의 개인 정보를 저장하고 medicine 테이블은 약 이름과 약 알람 시간을 관리한다. location 테이블은 환자의 위치를 위도와 경도로 나타내어 관리하며 distance 테이블은 환자의 운동량을 관리한다.

- patient 테이블

#	이름	종류	데이터정렬방식	보기	Null	기본값
1	p_id	varchar(40)	utf8_general_ci		아니오	없음
2	p_pw	varchar(40)	utf8_general_ci		예	NULL
3	p_name	varchar(40)	utf8_general_ci		예	NULL
4	p_phone	varchar(40)	utf8_general_ci		예	NULL
5	p_birth	int(11)			예	NULL
6	h_heart	int(11)			예	NULL
7	p_latitude	double			예	NULL
8	p_longitude	double			예	NULL

-

- medicine 테이블

	#	이름	종류	데이터정렬방식	보기	Null	기본값
☐	1	p_id	varchar(100)	utf8mb4_general_ci		아니오	없음
☐	2	m_name	varchar(100)	utf8mb4_general_ci		아니오	없음
☐	3	m_time	varchar(100)	utf8mb4_general_ci		아니오	없음

- location 테이블

	#	이름	종류	데이터정렬방식	보기	Null	기본값
☐	1	p_name	varchar(100)	utf8mb4_unicode_ci		아니오	없음
☐	2	p_longitude	double			아니오	없음
☐	3	p_latitude	double			아니오	없음

- distance 테이블

	#	이름	종류	데이터정렬방식	보기	Null	기본값
☐	1	p_id	varchar(40)	utf8_general_ci		아니오	없음
☐	2	walk	varchar(40)	utf8_general_ci		예	NULL
☐	3	datetime	varchar(100)	utf8_general_ci		예	NULL

Ⅲ. 결론

1. 연구 결과

최근 치매 환자가 급증함에 따라 환자들의 실종률이 증가하고 치매 환자를 보호하는 과정에서 생기는 사회적·경제적 문제가 언론에 많이 노출이 된다.

기존 시종 웨어러블 장치들은 NFC 태그를 찍어야 보이거나 운동량 및 현재 건강상태만 측정할 수 있었으나 갑작스런 위기 상황에 대응 할 수 없거나 데이터를 자동으로 업데이트 하지 않고 수동으로 업데이트 해야만 하는 문제점이 있었다. 따라서 본 제품은 치매 환자의 갑작스런 위기 상황에 대응 할 수 있도록 보호자에게 실시간으로 위치 정보 및 심박 수 운동량 데이터를 수집하여 보호자에게 데이터를 전송해 현재 상태와 위치 등을 알려준다. 이를 통해 보호자가 환자의 대한 관심도를 증가시키며 위기 상황에 대한 대응이 가능하게 하여 치매환자 보호에 있어 좀 더 효율적이고 보호자로 하여금 불안감을 감소시켜 심리적 불안감과 치매 환자를 보호에 발생하는 사회적 문제 또한 감소 할 수 있도록 개발하였다.

블루투스의 연결이 해제되거나 블루투스와 센서 값의 충돌로 인해 심박 수 측정값이 정상적으로 측정되지 않는 문제점과 충전을 목적으로 장치를 해제하였을 때 휴대전화와 장치를 둘 다 가지고 나가지 않았을 경우 사용할 수 없다는 문제점이 발생하였다. 블루투스 연결의 문제와 센서 값의 충돌은 측정값이 순차적으로 증가하거나 감소할 경우를 제외하고 갑작스럽게 높아진다거나 낮아질 때 데이터 전송을 하지 않거나 평균치를 구함으로서 불편함을 최소화 할 예정이며 사용자가 장치와 휴대전화 둘 다 나가지 않았을 경우를 대비하여 일정 기간 심박 수가 측정되지 않았을 경우 보호자 애플리케이션에 알람이 울리게 하여 문제점을 최소화 할 예정이다.

2.작품제작 소요재료 목록

아두이노 프로 미니 3.3V 동작 전압 : 3.3V 클럭 주파수 8MHz

리튬 폴리머 배터리 모음 3.7V 전압 : 3.7V

리튬 배터리 충전 모듈 리튬 : 입력전압 5V, USB타입 : C타입

심장박동 측정 센서 DM447 :공급 전압 5V

진동 모터 모듈 DM 159 :동작 전압 5V

블루투스 HC-06 :신호범위 10m

아크릴 판

아두이노 USB to UART 모듈 : SZH-EK072

참고자료

- 이재홍. 『아마존 웹 서비스를 다루는 기술』. 길벗(2014년)
- 남효정, 황성희, 김유정, 김기웅(2018). “대한민국 치매현황 2018”. 중앙 치매 센터
- 정화철. (2017). 치매노인 보호자의 부양스트레스가 자살생각에 미치는 영향. 한국콘텐츠학회논문지, 17(11), 167-182.
- MY LOMY. “LOMY”, http://lomy.co.kr/?page_id=2238, (2019.12.20)
- SAMSUNG. “SAMSUNG Health”. https://www.samsung.com/sec/apps/samsung-health/, (2019.12.20)
- 엘레파츠 전자부품쇼핑몰. “37도 스마트밴드”. https://eleparts.co.kr/goods/view?no=3868999, (2019.12.20.)
- 나무위키. “대한민국 고령사회”. https://namu.wiki/w/%EA%B3%A0%EB%A0%B9%EC%82%AC%ED%9A%8C. (2019.11.30)
- 동아닷컴. “고령사회 치매”. http://www.donga.com/news/article/all/20190910/97363404/1. (2019.09.10)
- 황현숙, 고윤성, 반가운, 김창수. (2016). “치매환자의 보호를 위한 스마트 앱 개발”
- 권대원. (2016). “블루투스 비콘을 화룡한 실내위치기반 치매환자 모니터링 시스템에 관한 연구”. 디지털융복합연구.
- 양은혜. (2016). "치매예방 통합운동 프로그램이 노인여성의 인지기능, 건강 체력 및 심장·뇌혈관계에 미치는 영향"
- 윤병곤, 김종진, 곽이섭. (2019). 운동이 치매 환자의 수면 관련 인자, 치매 관련 인자에 미치는 영향. 생명과학회지, 29(6), 740-746.
- 송준아, 박재원. (2011). 요양시설 치매노인의 배회 관리를 위한 시설기반 그룹 걷기 프로그램의 효과: 예비연구. , 13(1), 37-47.