

(양식4-1)

캡스톤디자인 결과보고서

학년도-학기	2018년 2학기	학부(과)	컴퓨터학과	
교과목명	졸업 프로젝트(캡스톤디자인) I		담당교수	이재훈 교수님
팀 명	백 키 퍼 (Back Keeper)			
프로젝트명	백 키 퍼 (Back Keeper)			
팀장 (대표학생)	학과(부)/학년	컴퓨터학과/3학년	성명	문기준
	E-mail	shangmoon@korea.ac.kr	휴대전화	010-4766-9137(정지 중) +65-8447-9430(싱가포르 번호)
팀원	학과(부)/학년		성명	
	학과(부)/학년		성명	
	학과(부)/학년		성명	
	학과(부)/학년		성명	
프로젝트 내용				
<p>1. 프로젝트 수행 결과</p> <p><input type="checkbox"/> 개발 배경 및 목적</p> <p>저는 어렸을 때부터 잘못된 허리 자세로 인해 통증을 자주 느꼈습니다. 그래서 허리에 좋다고 하는 스트레칭 동영상을 찾아보거나 필라테스 수업을 수강하면서 적극적으로 해결하려 했지만 오랜 시간 의사에 앉아있어야 하는 학생이라는 신분 때문에 통증을 완전히 해소하기 어려웠습니다. 헬스장, 필라테스 수업, 또는 병원을 가는 등의 적극적인 해결책은 시간이 많을 때 효과적이었으나 저와 같은 학생들은 시간이 적은 상황에서 요통은 참고 인내해야 하는 고통이었습니다. 주변 사람들, 트레이너 그리고 병원 관계자들과 얘기를 나누며 경험을 공유하면서 이와 같은 고통은 저만 겪는 문제가 아니라는 것을 알게 되었습니다. 국민건강보험공단의 통계자료에 따르면 2006년부터 2012년까지 7년 동안 척추수술 인원이 84% 증가했으며, 이에 따라 수술 후 환자가 허리를 지속적으로 관리할 수 있도록 도와주는 재활병원이나 재활센터에 대한 수요가 급증하고 있다고 합니다.</p> <p>이와 같은 흐름을 고려하였을 때 본 프로젝트의 목표는 허리통증에 영향을 미치는 다양한 변수를 고려하여 한명 한명의 사용자에게 맞춤화된 스트레칭 정보를 제공하는 플랫폼을 구축하는 것입니다. 허리 건강에 대한 많은 제품이 나오고, 다양한 물리치료 방법과 보조장구가 출시되는 상황에서, 4차 산업 혁명의 주요 주제로 떠오르는 IoT 기술과 기존의 요통 치료법을 결합하려고 합니다. 이를 통해 저와 같이 허리 통증으로 불편함을 겪는 학생들에게 오랜 시간 공부하면서도 올바른 자세를 유지할 수 있는 IoT 헬스케어 제품을 제시할 것입니다. 그리고 더 나아가 만성적인 요통을 겪는 중장년층들에게도 장기적으로 도움을 줄 수 있는 헬스케어 플랫폼을 준비하는 것이 본 프로젝트의 목적입니다.</p> <p><input type="checkbox"/> 개발 방향성 확립을 위한 실험 및 연구</p> <p>본 프로젝트의 목적상 보다 정확한 자세 교정을 할 수 있도록 제품을 개발해야 해기에 시중에 실제로 존재하는 여러 자세교정 제품을 직접 구매하여 참고하였습니다. 조사를 해본 결과 현재 시장에 존재하는 자세교정 제품에는 총 3가지가 있었습니다. 각각의 제품의 유형에는 각각의 장단이 있었습니다.</p>				

첫 번째는 견고한 고정대를 기반으로 몸에 직접 부착하여 장기간 자세를 올바른 자세를 유지하게 해주는 자세교정밴드 및 벨트입니다. 이러한 벨트 및 벨트의 경우 착용 시에는 효과적으로 자세를 잡아 줄 수는 있어도 착용과 동시에 동반되는 밴드의 압박감에 그 답답함을 이겨가며 일을 지속하기 힘들다는 큰 단점이 있었습니다. 또한, 비착용 시 기존의 잘못된 자세로 바로 돌아가게 되어 낮은 지속력을 보여줍니다. 두 번째는 최근 수험생들 사이에서 인기가 많았던 고급 방석인 벤런스온 시트입니다. 고탄성 특수 폴리



머 소재로 만들어진 이 제품은 약 10만원 상당의 고가의 제품으로 사용자의 체압을 분사시키며 진동을 흡수하여 판한한 바른 자세를 유지 시켜준다고 합니다. 하지만 실제로 제품을 구매하여 이용을 해보니 보통에 방석에 비해 통풍이 잘되어 시원하다는 느낌을 주었을 뿐 자세에 직접적인 영향을 주고 있다고 생각하기는 힘들었습니다. 세 번째는 거북목 교정을 위해 미국에서 개발된 웨어러블 기기인 알렉스입니다. 알렉스는 IoT제품의 일종으로 옆에 보이시는 그림과 같이 귀에 기기를 걸어 목의 뒷부분과 맞닿는 압력센서를 이용하여 사용자의 경부의 층만도를 측정하는 것입니다. 만약 거북목 자세 또는 경부 건강에 나쁜 자세를 취하게 되면 기기가 스스로 진동 통해 사용자에게 알려주게 됩니다. 또한, 이 기기는 핸드폰의 전용어플과 연동이 되어 24시간 스스로의 주기별 자세 데이터를 확인할 수 있습니다. 이는 마치 사용자가 실시간으로 자세교정 PT를 받는 것과 같은 느낌을 줄 수 있다는 기대감이 컸지만 막상 기기를 사용해본 결과 크게 두 가지 단점을 볼 수 있었습니다. 연결이 너무 자주 끊긴다는 사실과 제품 이름에서도 볼 수 있듯 목의 자세 이외에 자세교정의 또 다른 요소인 척추에 대한 교정은 전혀 다루고 있지 않는다는 것입니다.

이러한 제품들을 여럿 사용해본 결과 저는 이번 프로젝트의 방향성을 보다 쉽게 결정할 수 있었습니다. 사용해본 제품들을 통해 올바른 자세를 유지할 수 있는 IoT 헬스케어 제품을 개발하는 지표로 두 가지로 잡게 되었습니다. 첫 번째는 사용자가 부담 없이 항상 사용할 수 있는 제품이어야 한다는 것입니다. 두 번째는 자세교정 밴드와 같이 수동적으로 이루어지는 교정이 아닌 스스로의 잘못된 자세를 인지하고 고칠 수 있는 지표를 제시할 수 있어야 한다는 것입니다. 위에 언급한 3가지 제품 중 첫 번째 항목에 무엇보다 잘 부합했던 것이 바로 두 번째 제품인 벤런스온 시트입니다. 수험생들은 물론 직장인들에게도 많이 판매되었던 이 제품은 장시간 한곳에 앉아 있어야 하는 모든 사람들을 대상으로 거부감 없이 사용될 수 있는 방석의 형태를 띠고 있었기에 큰 인기몰이가 가능했다고 판단했습니다. **이에 본 프로젝트의 제품의 유형을 “방석”으로 지정하였습니다.** 이어 여기서 두 번째 지표가 등장하게 됩니다. 사용자가 스스로 자세를 교정할 수 있는 지표를 주기 위하여 방석의 형태에 IoT플랫폼을 입히도록 설계하였습니다. 세 번째 제품인 “알렉스”처럼 전용 어플을 통해 센서의 데이터를 친 사용자적으로 제공하는 플랫폼을 기획하였습니다.

프로젝트의 개발에 앞서 하드웨어 및 소프트웨어 개발에 있어서 도움이 될 참고자료들을 수집하는 것이 첫 번째 과정이었습니다. 이 과정에 있어 제일 큰 도움을 주신 분이 바로 이재훈 교수님입니다. 저는 이번 학기의 직전 학기인 2018년도 1학기에 이재훈 교수님은 사물인터넷 수업을 들었고 이를 통해 이번 프로젝트를 위해 필요한 기본적인 지식을 갖추게 되었습니다. 이에 IoT플랫폼의 하드웨어 및 소프트웨어에 대한 개발과정에 대한 정보들은 대부분 이전 수업의 자료를 통해 보충했으며 필요한 일부 부품 또한 교수님을 통해 전달 받았습니다. 또한, 어플 개발의 경우 여름방학 때 학교에서 제공했던 IOS어플 개발 과정을 통해 얻은 기반 지식으로 개발 과정을 밟아 나갔습니다. 이어 실제 개발과정에 대한 설명을 드리겠습니다.

□ 개발 과정에서의 실험 및 연구

본 프로젝트의 개발 과정에서의 실험 및 연구 절차는 크게 두 가지로 나누어 볼 수 있습니다. 이는 개발 착수 전 설계단계에서의 실험 및 연구과정과 개발 착수 후 개발단계에서의 실험 및 연구과정입니다.

1. 개발 착수 前: 설계단계에서의 실험 및 연구과정

1. 보드 선택:

저는 이번 프로젝트를 진행하기 위해서 IoT의 기반 보드인 3가지 보드를 직접 만지고 비교해 보았습니다. 이는 아두이노, 라즈베리파이, 그리고 삼성이 개발한 아틱 보드입니다. 아두이노와 라즈베리 파이의 경우 지난 학기 “사물 인터넷” 수업을 통해서 접했습니다. 아틱 보드의 경우 이번 프로젝트를 진행하면서 여러 공모전에 참여하면서 단가가 비싸고 상용화 된지 얼마 되지 않아 범용성과 활용도에서 다른 보드에 비해 아직은 비약한 아틱 보드를 접하게 되었습니다. 이 세 보드에 대한 실험 및 연구에 대한 자세한 내용은 도표로 정리하여 “**2. 프로젝트 수행과정 및 결과물 설명**”에서 자세히 설명하였습니다. 결과적으로 아두이노 보드를 선택하게 되었습니다.

2. 센서의 선택:

방석 속의 압력 선세를 선택함에 있어 제일 중요한 요소는 두 가지 였습니다. 첫 번째는 사용자가 앓아도 전혀 센서의 존재에 대해 몰랐을 만큼 부피가 작고 유연성이 있어야 한다는 것이었습니다. 두 번째는 자세에 대한 데이터를 받아와야 하기 때문에 최대한 많은 FSR(압력)센서가 고르게 퍼져 있어야 한다는 것이었습니다. 이에 총 5가지의 센서를 구매하여 테스트를 진행하였습니다. 이에 5가지 센서에 대한 내용은 그림으로 정리하여 “**2. 프로젝트 수행과정 및 결과물 설명**”에서 자세히 설명하였습니다. 결과적으로 “mdxs-16-5610”를 선택하게 되었습니다.

3. 무선 통신 방식의 선택:

방석센서의 데이터를 무선으로 기기에 전송할 수 있는 무선 통신 방법에는 두 가지가 있습니다. Bluetooth와 WiFi가 그것입니다. 물론 전송속도와 사용하는 모듈의 차이에서도 그 둘의 차이를 볼 수 있지만 고작 32개의 센서의 데이터를 전송하기 때문에 전송속도는 크게 영향이 없었지만 제일 중요한 것은 효율적인 데이터 저장에 있습니다. Bluetooth의 경우 기기와 아두이노가 다른 기기와의 부가적인 연결 없이 직접적으로 연결되기 때문에 보다 더 실시간에 가까운 데이터를 받을 수 있다는 장점이 있지만 해당 데이터를 저장하고 활용하려면 전송 받은 기기가 그 데이터를 직접 데이터베이스에 저장해야 하는 번거로움이 있습니다. 하지만 WiFi의 경우 직접 클라우드와 연동되어 클라우드가 직접 데이터를 관리하며 기기는 클라우드에서 직접 데이터를 받아올 수 있음으로 보다 안정적인 처리가 가능합니다. 또한 Bluetooth의 경우 master 하나가 연결 가능한 slave의 수가 대부분 7개로 제한되어 있지만 WiFi의 경우 Server에 접속할 수 있는 client의 수는 7개보단 확연히 많기 때문에 센서의 데이터에 접근 가능한 기기들의 수에도 제한이 생기기 마련입니다. 그래서 저는 이번 프로젝트에 WiFi를 사용하였습니다.

4. 어플리케이션 개발 OS의 선택:

이번 선택에 있어서 두 OS에 대한 연구나 실험을 진행한 것은 아닙니다. 어플리케이션 개발 경험에 전무였던 저에게 지난 여름방학에 고려대학교 정보대학에서 진행하였던 Swift를 이용한 IOS 어플리케이션 개발 수업을 들었기 때문에 수업을 기반으로 어플리케이션을 만들어야 했기에 IOS를 선택하였습니다.

2. 개발 착수 後: 개발단계에서의 실험 및 연구과정

1. 개발에 쓰인 모든 부품들 리스트: 표로 정리하여 “**2. 프로젝트 수행과정 및 결과물 설명**”에 적었습니다.

2. 방석센서의 하드웨어 작업:

mdxs-16-5610 FSR센서의 경우 31개의 FSR센서를 담고 있기 때문에 해당 센서의 단자들을 보다 쉽게 아두이노의 포트에 연결시키기 위해 32채널 쉴드를 연결해야 합니다. 이를 연결하는 방법은 그림로 정리하여 “**2. 프로젝트 수행과정 및 결과물 설명**”에 적었습니다. 센서를 쉴드에 연결했다면 쉴드가 센서의 데이터를 아두이노로 전달할 수 있게끔 각각의 포트에 배선 작업을 해야 합니다. 이 배선 작업 역시 그림로 정리하여 “**2. 프로젝트 수행과정 및 결과물 설명**”에 적었습니다. 이렇게 센서와 쉴드 그리고 쉴드와 아두이노를 연결했다면 방석 세서의 하드웨어 작업은 끝이 납니다.

3. 방석센서의 Arduino 스케치 코딩 작업:

이어서 아두이노를 통해 31개의 FSR센서의 값을 받아와 이를 통해 사용자의 자세에 대한 기준을 내려야 합니다. 각각의 FSR 센서의 값은 float변수 형태로 받아지기 때문에 이를 단순히 출력하고 받아오는 것 만들로는 아무런 결과를 이루지 못하기 때문에 저는 이 작업을 두 번의 코드에 걸쳐서 만들어냈습니다. 첫 번째 코드는 31개의 FSR센서의 값을 그대로 읽어와 Serial 창에 출력해 내는 것과 두 번째는 이 값을 통해 사용자의 무게중심을 계산하여 무게가 치중되는 방향을 출력할 수 있게끔 변환했습니다. (예: 앞이면 front, 뒷면 back, 왼쪽이면 left, 오른쪽이면 right) 해당 코드에 대한 발전과정을 보여드리기 위해 “**2. 프로젝트 수행과정 및 결과물 설명**”에 코드를 받을 수 있는 URL을 올렸습니다.

4. Wifi 모듈(ESP-01)모듈의 펌웨어 업그레이드:

ESP-01모듈의 경우 시제품이 나온 후 많이 시간이 흘렀기 때문에 이 모듈을 사용하기 위해서는 꼭 펌웨어를 업그레이드 시켜야 합니다. 펌웨어 업그레이드에 대한 모든 내용은 “**2. 프로젝트 수행과정 및 결과물 설명**”보다 구체적으로 적었습니다. 해당 과정이 완료되었다면 이젠 ESP-01모듈을 아두이노와 연결하여 WiFiEsp.h 헤더 파일과 함께 실제 와이파이 모듈로써의 역할을 할 수 있게 됩니다.

5. 아두이노와 ESP-01모듈로 Server구축, 하드웨어 작업:

이전 세팅에서 이젠 두 개의 배선만 옮겨 준다면 모든 하드웨어 세팅이 완료됩니다. 그 두 개의 배선은 아두이노와 ESP-01모듈의 TX와 RX연결선입니다. 이 두 연결선을 아두이노 디지털 포트 6번에 ESP-01모듈의 TX그리고 아두이노 디지털 포트 7번에 ESP-01모듈의 RX를 연결하게 되면 모든 하드웨어 배선이 끝이 납니다. 보다 나은 이해를 위해 배선을 다시 한번 그림으로 정리하여 **2. 프로젝트 수행과정 및 결과물 설명**에 올렸습니다.

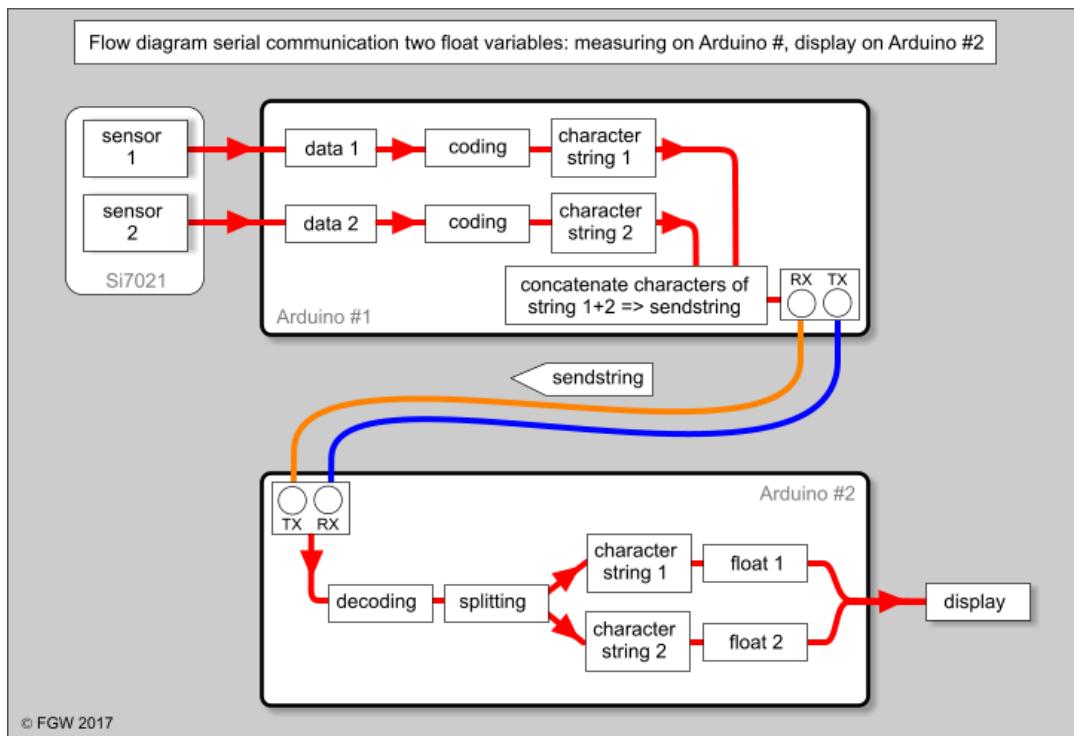
6. 아두이노와 ESP-01모듈로 Server구축, Arduino 스케치 코딩 작업:

해당 코드의 경우 WiFiEsp.h 헤더 파일을 이용하여 전체적인 코드를 작성하게 됩니다. 코드의 진행은 다음과 같습니다. ESP-01모듈이 입력된 와이파이를 찾아 연결하고 연결된 인터넷을 통해 포트 Local Host의 포트 80으로 Server를 만들게 됩니다. 이에 해당 와이파이에 연결된 모든 기기들은 이 서버에 접속할 수 있게 되는 것입니다. 그리고 Server는 접속된 client가 있을 때 마다 반복적으로 데이터를 받아서 http 스크립트 형식의 언어로 client에게 보내집니다. 이는 연결된 client 기기의 웹브라저 상에 원하는 포맷으로 출력되게 되며 이는 연결이 지속되어 있다면 5초에 한번씩 새로고침이 되게 코드를 작성하였습니다. 이렇게 작성된 코드 역시 URL로 “**2. 프로젝트 수행과정 및 결과물 설명**”에 올렸습니다.

7. 방석센서 아두이노와 Server 아두이노의 Software Serial 통신:

방석센서 아두이노와 Server 아두이노의 통신은 Software Serial로 연결됩니다. 이를 위해 두 아두이노의 Rx와 Tx는 서로 교차되어 연결되어야 합니다. 이를 통해 서로의 Serial Monitor 창의 데이터를

바이트 단위로 읽어올 수 있게 됩니다. 바이트 단위의 데이터를 받기 위해 데이터를 받는 입장인 Server 아두이노의 코드에 다음과 같은 작업을 하는 코드를 추가합니다. 먼저 Serial 통신에 들어오는 데이터가 있다면 이를 Serial.read함수를 통해 바이트를 읽어오게 되고 읽어온 각각의 바이트 단위의 데이터를 한 line 씩 하나의 String변수 안에 저장하게 됩니다. 이 String 값을 각각의 필요한 부분으로 잘라서 다시금 String 변수를 각각의 변수 타입에 맞게 변환하여 (예: 무게 중심을 나타내는 float라면 string을 float로 변환) http 형식으로 클라이언트에게 넘겨주게 됩니다. 이를 그림으로 도식하게 된다면 아래와 같습니다. 이렇게 작성된 코드 역시 URL로 “[2. 프로젝트 수행과정 및 결과물 설명](#)”에 올렸습니다.



여기서 sensor1 과 sensor2는 무게중심의 x좌표와 y좌표로 대치되며 화살표의 마지막에 위치하는 display는 http 형식으로 각각의 client에게 보내는 과정을 의미합니다. 저는 하나의 스트링 안에 있는 데이터를 보다 효율적이게 decode하기 위하여 방석센서 아두이노의 코드를 약간 수정하여 “쏠려있는 무게 중심의 방향”f “x좌표의 무게중심”s “y좌표의 무게중심”l 이렇게 스트링 안에 데이터를 f(first) s(second) l(last)로 분리하여 Server 아두이노에게 보내게 됩니다. 이를 이용하여 Server arduino는 보다 정확하게 String을 각각의 데이터에 해당되는 부분만 분리해 낼 수 있게 됩니다.

8. 반석센서 아두이노에 진동모터 부착:

사용자가 잘못된 자세를 취할 경우 결고를 해주는 진동 센서를 연결하는 작업입니다. 진동 센서의 경우 방석센서 아두이노에 연결되게 됩니다. 이를 연락하는 방법은 간단합니다.



왼쪽 그림과 같이 3개의 단자로 이루어져 있는 진동 모터를 GND는 방석센서 아두이노의 GND에 VCC는 해당 아두이노의 5v에 연락하고 진동 모터를 작동시키는 IN단자에는 앞, 뒤, 좌, 우에 각각 부여하고 싶은 아두이노 디지털 GPIO핀에 연결하면 됩니다. 그렇다면 코드도 간단합니다. 무게중심이 치주되는 Front, Back, Left, Right가 출력되는 라인 바로 다음 줄에 해당 핀으로 전압을 흘리면 됩니다. 하지만 여기서 큰 문제가 발생합니다. 그러면 사용자가 바른 자세로 돌아 온다면 진동을 멈춰주는 코드는 어떻게 작성해야 하는가에 대한

문제입니다. 이는 기존에 Front, Back, Left, Right를 출력하게 만드는 if문을 그대로 부등호만 반대로 만들어 loop안에 넣어 GPIO출력을 다시 0으로 돌려놓는다면 금방 해결되는 것이었습니다. 해당 코드 역시 URL로 “**2. 프로젝트 수행과정 및 결과물 설명**”에 올렸습니다.

9. IOS 어플리케이션 개발:

IOS 어플리케이션 개발에 대한 부분은 “**2. 프로젝트 수행과정 및 결과물 설명**”에 보다 구체적으로 설명했습니다.

□ 최종 결과물

본 프로젝트의 결과물은 크게 세 부분으로 나누어져 있습니다. 첫 번째는 아두이노를 통한 방석센서 및 WiFi모듈의 하드웨어 개발이며 두 번째는 해당 하드웨어를 동작시키기 위한 아두이노 스케치(소프트웨어)의 개발입니다. 그리고 마지막 결과물은 센서와 함께 자세교정을 도와줄 IOS어플리케이션입니다.

1. 아두이노 하드웨어: 하드웨어의 중심에는 두 개의 아두이노 UNO가 있습니다. 센서와 동작을 제어하는 아두이노는 방석센서인 mdxs-16-5610와 32채널 쉴드를 통해 연결되어 있으며 추가적으로 4개의 진동 모터와도 연결되어 있습니다. 또 다른 아두이노는 ESP-01 WiFi 모듈과 연결되어 있으며 이 두개의 아두이노들은 Software Serial 통신을 위해 서로의 Rx 와 Tx가 교차되어 연결되어 있습니다.

2. 아두이노 IDE 스케치: 아두이노 IDE 스케치는 각각의 아두이노의 기능을 부여합니다. 방석센서가 연결된 아두이노에게는 FSR 센서를 통해 사용자 자세의 불균형도(무게중심)를 계산하며 심하게 틀어져서 앉을 경우 진동모터로 약한 알림이 울리도록 설정되어 있습니다. 이렇게 계산된 무게중심은 Software Serial을 통하여 또 다른 아두이노로 전달됩니다. 또한, 또 하나의 아두이노의 경우 전달 받은 방석 센서의 데이터를 WiFi 모듈을 통하여 실시간으로 Internet을 통하여 Server를 구축하고 이는 웹 브라우저를 탑재한 모든 디바이스가 Client로 동작하여 정보를 확인할 수 있습니다.

3. 어플리케이션: IOS기반의 어플리케이션으로 Swift언어로 구현되었다. 해당 어플리케이션은 사용자의 정보(나이, 운동 빈도, 몸무게, 키, 허리 통증 부위)를 입력 받아 이를 통해 사용자에게 맞는 최적의 스트레칭 방법을 스마트폰 어플리케이션으로 알리며 사용자의 잘못된 자세에 대한 정보를 누적하여 보다 사용자가 쉽게 스스로의 자세에 대해 확인할 수 있도록 여러 기능을 제공합니다.

이 세 가지 결과물의 설계도 및 실제 구동 영상은 “**2. 프로젝트 수행과정 및 결과물 설명**”에 보다 구체적으로 설명하겠습니다.

앞으로 진행될 발전 가능 사항 또한 “**2. 프로젝트 수행과정 및 결과물 설명**”에 보다 구체적으로 설명하겠습니다.

2. 프로젝트 수행 과정 및 결과물 설명(사진, 그래프, 표 활용)

ex) 위 내용 중 설계 및 실험 과정과 결과물에 대해 사진, 그래프, 표 등을 활용하여 구체적으로 설명

개발 착수 前: 설계단계에서의 실험 및 연구과정(위 내용 참고)

1. 보드 선택:

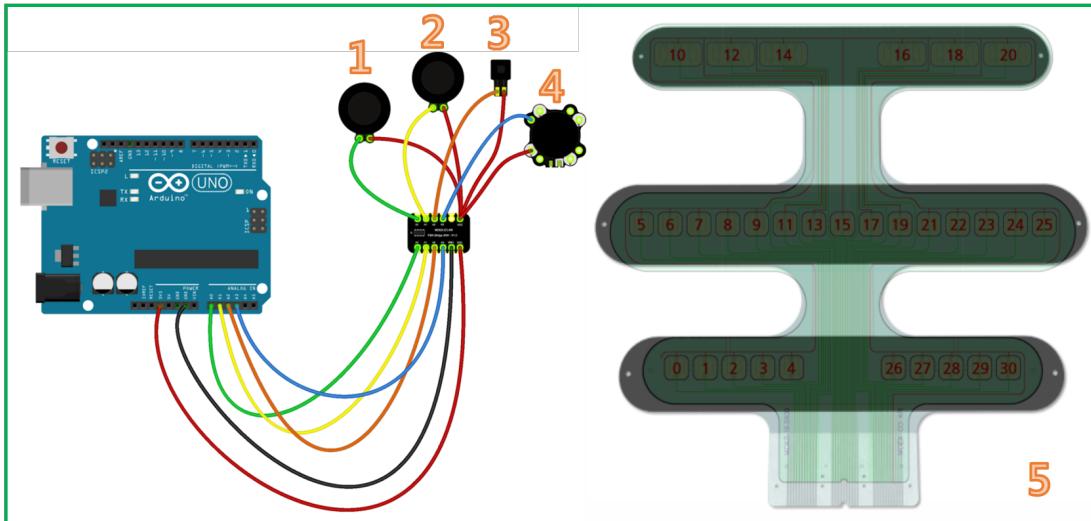
ARTIK		아두이노		라즈베리파이	
장점	단점	장점	단점	장점	단점
높은 성능	5v 지원 x	싼 단가		연산처리가 강함	비싼 단가
	기존 라이브러리 사용 불가	소형화에 특화		비디오나, 카메라, 복잡한 수치 계산같은 그래픽 처리에 특화	외부기기 제어에 약한 경향
	제일 비싼 단가	센서에서 데이터를 받아오고 모터 등을 제어하는 작업에 특화			센서 및 모터의 라이브러리 설치가 제한적-> 라즈베리 파이 전용 라이브러리가 없으면 직접 드라이버를 구현해야 함
		별도의 IDE로 어떤 컴퓨터 운영체제에서도 코딩 가능			라즈베리 파이는 키보드, 마우스, 모니터를 모두 연결하고 코딩 (번거로움)

아틱 보드의 경우 라즈베리파이와 아두이노와 비교했을 때 그 성능은 제일 뛰어납니다. 아틱 보드에 대한 교육을 삼성에서 5일 가량 받고 난 후 이 보드는 이번 프로젝트에 적합하지 않다는 결론을 내렸습니다. 일단 성능에 비해 아직까지 아틱 보드에 활용 가능한 압력 센서를 시중에서 구할 수 없었습니다. 또한 수많은 센서와 모터 통신장치등이 5v를 사용하는데 라즈베리 파이의 경우 5v출력이 없습니다. 이에 대비해 GPIO핀을 아두이노와 유사하게 설계하여 아두이노용 쉴드를 사용 가능하게 만들었지만 아두이노 스케치에서 쓸 수 있는 라이브러리를 불러오지 못한다는 큰 단점이 있어 WiFi 나 Bluetooth 같은 통신 모듈을 불러올 수 없었습니다. 어플과의 통신이 필수적인 저의 프로젝트에 치명적인 단점이기에 아두이노와 라즈베리파이 사이에서 고민을 하게 되었습니다.

일단 결과부터 말씀을 드리자면 전 아두이노를 선택했습니다. 그 이유는 여러 가지가 있습니다. 일단 단가의 차이입니다. 저의 프로젝트의 최종적인 목표는 상용화입니다. 그래서 라즈베리 파이의 1/10의 단가를 가지고 있는 아두이노가 원가를 낮추기 위한 방법 중 하나였습니다. 또한 방식이라는 외형을 띠고 있는 만큼 사람이 앉아도 눈치채지 못할 정도의 크기 여야 합니다. 아두이노는 소형화에 특화된 Arduino Pico 모델이 있을 정도로 그 크기를 크게 줄일 수 있습니다. 또한 아두이노는 센서에서 데이터를 받아오고 모터 등을 제어하는 작업이 훨씬 편리합니다. 이는 각각의 마이크로 컨트롤러의 차이에서 나오게 됩니다. 마이크로 프로세서(라즈베리파이)는 외부기기 제어에 약한 경향이 있어, 트랜지스터가 외부 기기를 다룹니다. 이에 반해 연산처리가 강하다는 장점을 갖고 있습니다. 그래서 비디오나, 카메라, 복잡한 수치계산같은 그래픽 처리에는 라즈베리파이가 더 적합합니다. 결과적으로 센서의 데이터를 받아 전공하는 역할을 담당하게 될 저의 프로젝트에 보드엔 아두이노가 훨씬 적합하다는 판단을 했습니다.

2. 센서의 선택:

제가 구매한 5가지의 압력 센서는 RA9P, RA9, QA6P, RA18P-DIY와 32채널 FSR 센서입니다. 각각의 센서들은 아래에 보여진 그림과 같은 형태를 띠고 있습니다.



RA를 이름으로 가지고 있는 센서들의 경우 모두 낱개의 FSR센서와 같습니다. 이는 모두가 VIN이라는 아두이노의 전압포트와 연결되는 단자와 ADC포트와 연결되는 또 하나의 단자가 있습니다. 이 때문에 큰 문제가 발생하게 됩니다. 최대한 많은 FSR 센서가 고르게 분배 되어야 되는 방식 센서의 조건에 만족하려면 적어도 아두이노가 FSR센서의 수 만큼의 전압 포트와 ADC 포트가 있어야 하기 때문입니다. 물론 브레드보드를 활용하여 연결하는 방법이 있을 수도 있지만 방식 안에 그 딱딱한 보드가 들어가게 된다면 아무도 그 방식을 사용하지 않을 것입니다. 또한 저는 4채널 브릿지를 활용하여 위에 사진과 같이 여러 FSR 센서를 묶어보려고 하였지만 이 또한 고작 4개가 고작이며 아두이노 UNO보드의 특성상 6채널을 이용하여 최대 6개의 FSR 센서가 최대 연결 가능한 숫자였습니다. 이러한 시행착오를 통해 이러한 단점을 극복할 수 있는 센서를 찾고 있던 도중 31개의 FSR 센서를 3개의 레이어로 나누어 고르게 분배한 mdxs-16-5610를 찾게 되었고 이를 통해 고작 6개의 Digital 포트만으로 32개의 FSR 센서의 정보를 모두 받아올 수 있었습니다.

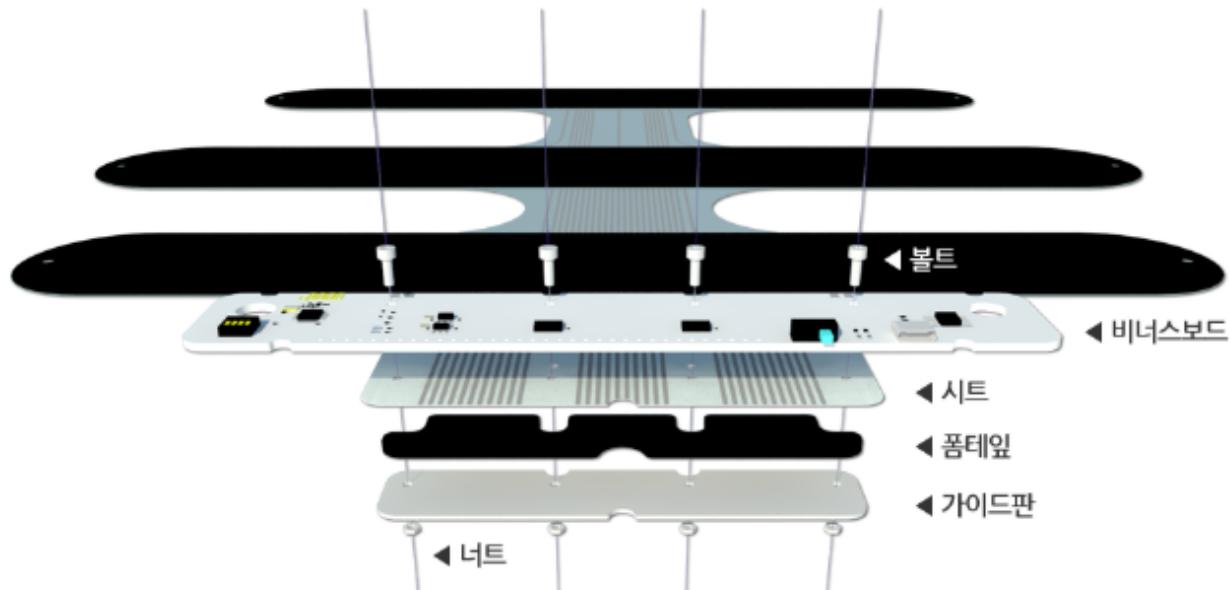
□ 개발 착수 後: 개발단계에서의 실험 및 연구과정

- 개발에 쓰인 모든 부품들 리스트:

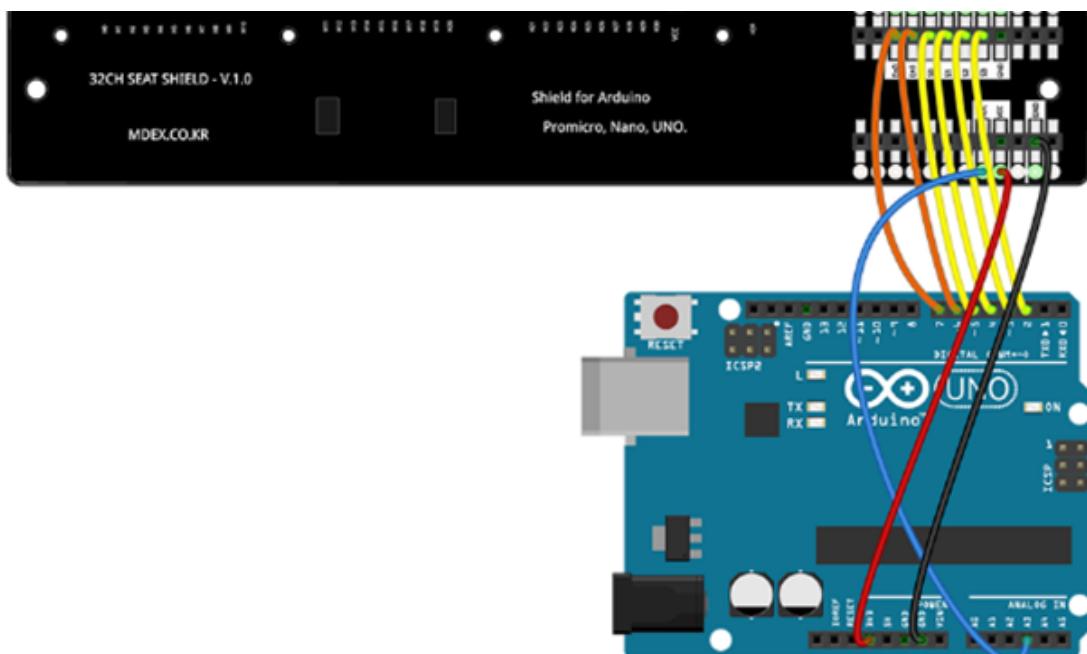
부품종류	부품 상세 이름	개수
진동 모터 모듈	ELB060416	4
아두이노 UNO		3
ESP-8266의 일종인	ESP-01 WIFI 모듈	2
방식 센서	mdxs-16-5610	1
방식 센서 전용	32채널 방식 쉴드	1
암수 배선		34
브레드 보드		1

2. 방석센서의 하드웨어 작업:

mdxs-16-5610와 32채널 쉴드의 연결 방법:



32채널 쉴드와 아두이노 UNO의 연결과정:



3. 방석센서의 Arduino 스케치 코딩 작업:

31개 센서 값만 출력하는 코드 URL:

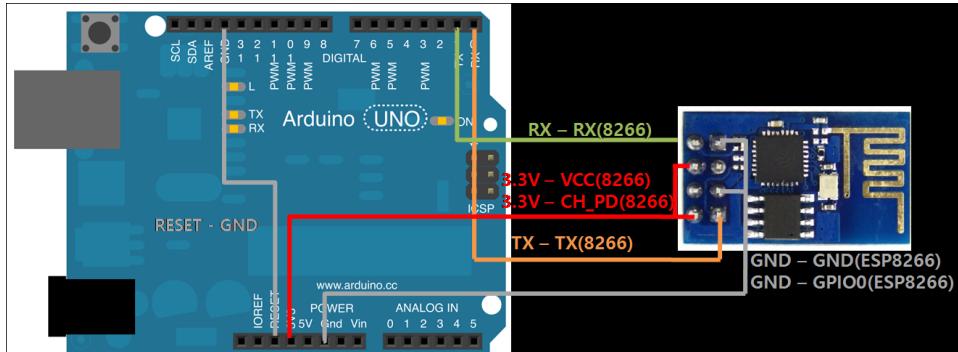
<https://drive.google.com/file/d/1A0erJ5kZb1-ByzkVv6aB4nNBHtfikk9R/view?usp=sharing>

31개 센서 값을 이용하여 무게중심 및 치중된 방향을 출력하는 코드 URL:

<https://drive.google.com/file/d/1HcRByDSq5I4nKCBEAmROchys7nidv-ap/view?usp=sharing>

4. Wifi 모듈(ESP-01)모듈의 펌웨어 업그레이드:

펌웨어 업데이트의 경우 ESP-01모듈을 아두이노에서 사용할때의 배선과 다르기 때문에 주의해야 합니다.



배선이 이렇게 완료되었다면 아래의 두 링크에 들어 있는 ESP-01모듈에 펌웨어와 이를 업로드 시켜주는 “Flash Tool”을 다운로드 받아 합니다.

펌웨어 URL: <http://bbs.espressif.com/download/file.php?id=989>

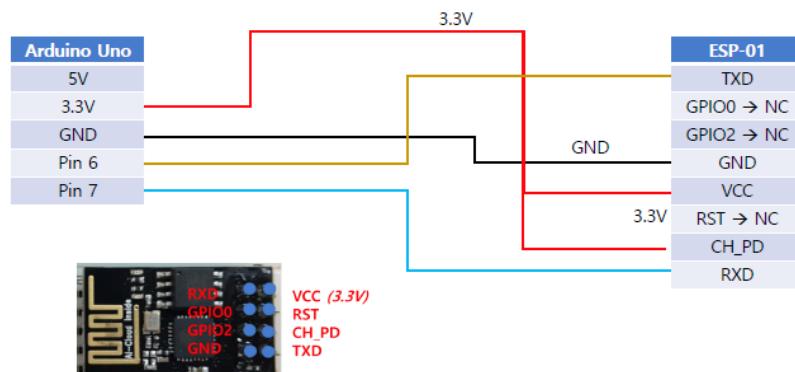
Flash Tool: <http://bbs.espressif.com/download/file.php?id=856>

두 파일을 다운로드 받았다면 일단 아두이노 IDE를 닫습니다. 그리고 Flash Tool 폴더 안에 있는 아래와 같은 파일을 실행시켜야 합니다. “ESP_DOWNLOAD_TOOL_V2.4.exe” 실행을 시켰다면 펌웨어 파일 안에 들어있는 아래의 디렉토리에 있는 파일들을 Download Path Config 아래 업로드시키고 ADDR에 해당 주소를 입력합니다.

```
bin\at\noboot\eagle.flash.bin - 0x00000
bin\at\noboot\eagle.irom0text.bin - 0x40000
bin\blank.bin - 0xfe000
bin\blank.bin - 0x7e000
```

이 모든 것이 끝났다면 Flah size를 8Mbit로 설정하고 COM port를 해당 아두이노가 연결된 포트로 설정합니다. 그리고 Start 버튼을 누르게 되면 펌웨어가 정상적으로 업로드 됩니다. 이제 아두이노를 PC에서 제거하고 ESP-01모듈의 GPIO0과 아두이노 GND에 연결되어 있는 선과 아두이노의 RESET과 GND를 연결하는 선역시 제거합니다. 이후 아두이노를 다시 PC에 연결하고 IDE를 키고 시리얼 모니터를 실행시킵니다. 시리얼 모니터가 열렸다면 스피드를 115200로 바꾸고 Line ending을 Both NL & CR로 바꿔줍니다. 이후 Baud Rate를 수정하기 위해 AT+UART_DEF=9600,8,1,0,0 와 동일한 명령을 콘솔창에 입력합니다. 이후 OK가 떴다면 펌웨어 업그레이드는 완료됩니다.

5. 아두이노와 ESP-01모듈로 Server구축, 하드웨어 작업:



6. 아두이노와 ESP-01모듈로 Server구축, Arduino 스케치 코딩 작업:

URL:

<https://drive.google.com/file/d/1i2iv2HJfqMrwY9Q3aTc9LM5EoJ5IY8nj/view?usp=sharing>

7. 방석센서 아두이노와 Server 아두이노의 Software Serial 통신:

URL:

<https://drive.google.com/file/d/1kMli9MzX41wX6Do7qemSsAtNiCXKopqv/view?usp=sharing>

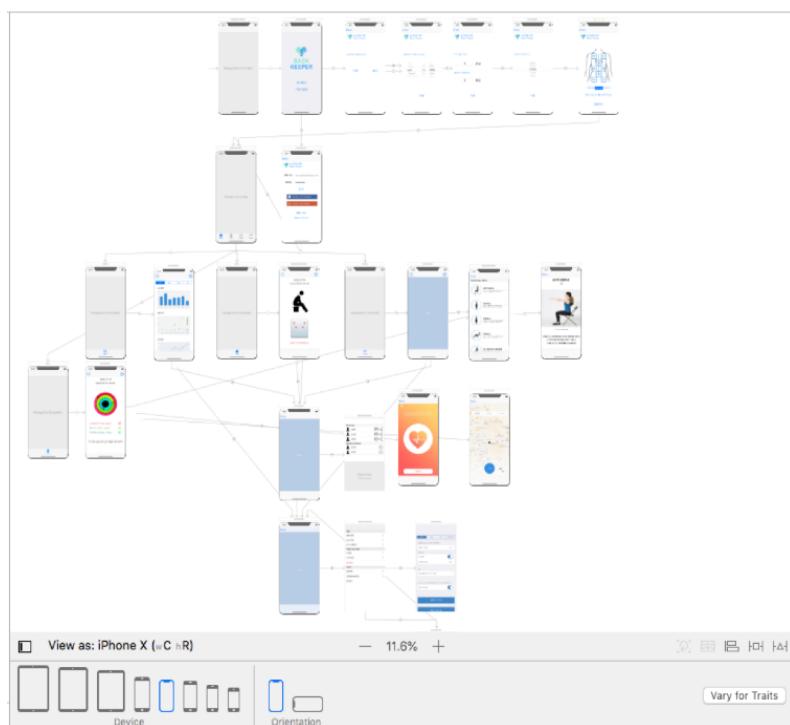
8. 반석센서 아두이노에 진동모터 부착:

URL:

<https://drive.google.com/file/d/1n-MhyKI0pJeN1r1roYzxQYVx4PnRQiaJ/view?usp=sharing>

9. IOS 어플리케이션 개발:

IOS 어플리케이션의 경우 Macbook의 XCode를 이용하여 Swift언어로 작성되었습니다. 어플리케이션의 개발에 앞서 제일 중요한 것은 이 제품을 가장 많이 사용하게 될 사용자들이 보통 어떤 류의 핸드폰을 사용하느냐입니다. 이는 각각의 기기와 OS에 따라 그 개발환경과 방법이 다르기 때문입니다. 하지만 이번 프로젝트에서 저는 이 사항을 고려할 수 있는 기술적 능력이 되지 않아 고려대학교 여름방학 IOS 개발 프로그램을 통해 배웠던 개발환경을 그대로 가지고 와 어플리케이션을 개발 하였습니다. 이에 제가 가지고 있는 iPhone X를 defualt 환경으로 설정하여 개발을 진행하였습니다. 먼저 각 페이지의 디자인과 configuration을 설정한 후 해당 설정 사항에 따라 화면이 Transfer 하는 rule을 설정하여 어플리케이션의 전제적인 tree구조를 만들었습니다. 구축한 트리 구조는 아래 그림과 같습니다.



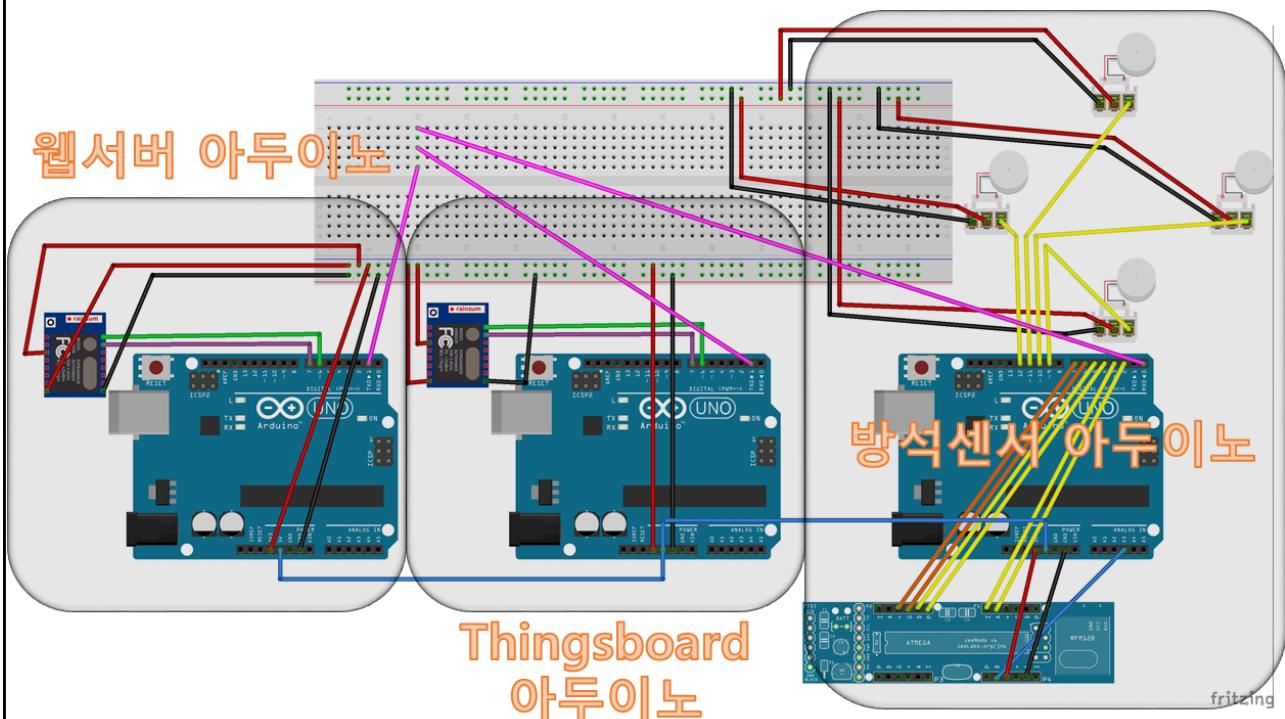
이렇게 구축한 프로그램을 Simulator를 통해 돌려 직접 어플의 화면을 동영상으로 촬영하였습니다. 해당 동영상 역시 아래의 URL로 첨부하였습니다.

URL: https://drive.google.com/file/d/1Fi-3eo34yNacxj2Fs20PKf56_fuhtg4J/view?usp=sharing

어플리케이션은 여러 기능을 제공합니다. 첫 번째, 회원가입을 통해 각종 인적사항 및 허리통증에 대한 정도를 현재 상태를 파악합니다. 인적 사항으로는 생일, 키, 몸무게, 직업을 순서대로 물어보며 통증이 있는 허리부위를 특정 지을 수 있도록 구현했습니다. 두 번째, 회원가입 이후에 회원으로 로그인을 하면 4가지 기능을 사용할 수 있게 됩니다. 1. "Home" 메뉴: 아두이노 방석 센서와 연동이 되어 실시간으로 자세에 대한 정보를 받습니다. 2. "State" 메뉴: 하루에 스트레칭 횟수와 일어서기 횟수 그리고 걷기 운동에 대한 정보(핸드폰의 GPS를 통한 정보)를 통해 하루 권장량에 어느정도를 채웠는지를 확인할 수 있습니다. 3. "Analysis" 카테고리: 7일 30일 6개월 1년 기준으로 누적된 사용자의 스트레칭 횟수, 일어서기 횟수, 걷기 운동 정도를 볼 수 있습니다. 4. "Training" 메뉴: 자신의 직업에 맞게 가볍게 의자에서도 할 수 있는 간단한 스트레칭 방법에 대한 가이드를 보여줍니다. 부가적인 메뉴로는 친구들을 추가해 서로의 자세 교정을 도울 수 있는가 있지만 아직 완성되지 않은 상태이며 설정창을 통해 특정 주기 별로 알림을 받아 스트레칭을 경고하게 할 수 있는 설정 또한 구현하였습니다.

□ 최종 결과물(설계도, 코드 및 작동영상):

설계도:



해당 그림은 최종 결과물 3가지 부분 중에 하드웨어 부분의 모든 것을 담고 있습니다. 하지만 이 결과보고서를 처음부터 읽으신 분이라면 이야 할 부분이 있을 것 같습니다. 그것은 바로 Thingsboard 아두이노에 대한 부분입니다. 여기서 Thingsboard 아두이노의 기능은 바로 Thingsboard 클라우드와 연동하여 자세 데이터를 클라우드에 전달하며 실시간으로 누적된 데이터를 보다 직관적으로 확인해 볼 수 있게 만드는 아두이노입니다. 이 해당 아두이노에 대한 개발 과정은 “12월 04일 추가 개발 과정”을 보시면 확인하실 수 있습니다.

설계도에 대해 각각의 부품들의 위치를 명확하게 설명하려고 합니다. 웹서버 아두이노와 Thingsboard 아두이노에 연결된 작은 모듈이 바로 ESP-01모듈이며 방석센서 아두이노 아래에 위치하고 있는 보드가 바로 FSR센서와 연결이 된 32체널 쉴드입니다. 또한, 방석센서 아두이노 위에 위치하고 있는 4개의 모듈이 바로 진동 센서입니다. 설계도의 크기가 너무 커질 거 같아 mdxs-16-5610센서는 배제했습니다. 또한, 배치 상 혼란을 막기 위해 mdxs-16-5610센서와 겹치지 않게 끔 설계도를 그렸습니다. 하지만 실제 제품의 경우 진동모터가 mdxs-16-5610센서의 앞뒤좌우에 배치되어 있습니다.

배선의 구조가 잘 보이지 않아 따로 설계도 파일을 URL로 남겨둡니다.

URL: <https://drive.google.com/file/d/1GK9rO4eNlecFjHN41r8qQiEWI7VoO3jK/view?usp=sharing>

최종 완성 코드:

최종 결과물의 코드는 진동 모터 개발 이후의 코드와 동일 하지만 이전 코드 보다 훨씬 정돈되어 보다 쉽게 코드를 이해할 수 있습니다.

URL:

<https://drive.google.com/file/d/1LyWjLgzSR--hTChPRQm-B9nVYDkojiCE/view?usp=sharing>

작동 영상:

Youtube: <https://youtu.be/wbW02bEfn6E>

□ 최종 결과물의 향후 발전 가능성:

발전 가능성 1: 방석센서 아두이노는 항상 길이가 바뀌는 String을 비트 형식으로 변환하여 Serial Software을 통해 다른 아두이노에게 보내게 됩니다. 이 과정에 있어서 아두이노 스케치의 특성상 두 개의 아두이노의 loop()의 주기가 항상 Sync 되지 못하기 때문에 가끔 한줄 이상의 데이터가 전송되거나 데이터가 잘려서 오는 경우도 있다. 이에 두 아두이노 사이의 통신을 Software Serial이 아닌 블루투스로 전환한다면 보다 빠르고 정확하게 받을 수 있을 것입니다.

발전 가능성 2: 아직까지 앱과의 연동이 되어 있지 않아 100프로의 효과를 보고 있지 못한다. 이에 앱과의 연동이 필수 불가결 적입니다.

발전 가능성 3: Thingsboard 아두이노를 통해 Thingsboard 클라우드에 자세 데이터를 누적시켜 보다 직관적으로 사용자가 스스로의 데이터를 볼 수 있게끔 하는 것 또한 좋은 발전 가능성입니다. “12월 04 일 추가 개발 과정”에 구현 과정 설명: 이미 구현 완료

발전 가능성 4: 바른 자세에 대한 보다 정확한 정의를 위해 전문가들이 생각하는 바른 자세에 대해 연구하여 각각의 사용자가 일정기간 스스로의 바른 자세를 유지할 수 있게 된다면 이를 통해 각 사용자들의 바른 자세에 대한 압력값을 측정하여 그 압력 값을 영점으로 두고 x좌표와 y좌표의 무게중심을 계산한다면 보다 정확한 개개인 맞춤형 방석이 될 수 있다고 생각했습니다.

발전 가능성 5: 보고서 초기에 말씀드렸던 거북목 교정 웨어러블 기기인 “알렉스”와 같이 바르지 않은 자세를 탐지해 내는 예민도를 개개인 별로 다르게 설정할 수 있다면 이 또한 보다 Customized 된 기기를 만드는데 일조할 수 있을 것 같습니다.

발전 가능성 6: 현재 제품은 브래드 보드등 너무나도 많은 양의 와이어를 달고 있지만 모든 단자를 납땜하여 연결하고 아두이노 우노 모델 또한 소형 아두이노 모델로 전환 시켜 정말 방식 안에 넣어도 전혀 불편하지 않은 제품을 개발하는 것도 중요한 발전 가능성입니다.

12월 04일 추가 개발 과정:

10. Thingboard를 이용한 데이터 업로드:

Arduino UNO, ESP8266 및 MQTT를 사용한 중계중심 대시보드를 구현하였습니다. 과정은 간단합니다. 먼저, Arduino Uno를 사용하여 수집 된 무게중심 값을 받아 옵니다. 이어 ESP8266을 사용하여 MQTT를 통해 데이터를 업로드하고 Thingsboard를 사용하여 시각화합니다. Thingsboard 는 IoT 장치를 모니터링하고 제어 할 수 있는 오픈 소스 서버 플랫폼입니다. 개인용 및 상업용으로 모두 무료이며 어디서나 배포 할 수 있습니다.

데이터 전송:

방석센서 아두이노에서 Software Serial로 받아온 무게중심 데이터를 Arduino UNO가 ESP8266을 사용하여 WiFi네트워크에 연결합니다. Arduino UNO 는 Arduino 용 PubSubClient 라이브러리를 사용하여 MQTT 프로토콜을 통해 Thingsboard 서버에 데이터를 푸시합니다 . 데이터는 빌트인 사용자 정의 대시 보드를 사용하여 시각화됩니다. Arduino UNO에서 실행되는 응용 프로그램은 매우 간단하고 이해하기 쉬운 Arduino SDK를 사용하여 작성됩니다.

ESP8266 펌웨어:

현재 개발과정에서 WiFiEsp Arduino 라이브러리는 Arduino 보드를 인터넷에 연결하는 데 사용됩니다. 이 라이브러리는 ESP SDK 버전 1.1.1 이상 (AT 버전 0.25 이상)을 지원합니다. 즉, 이전 Server 아두이노를 구축할때에 과정을 그대로 반복하면 됩니다.

구체적인 개발 과정:

A. 스노 보드 구성

1 단계 : Thingsboard 설치

Thingsboard 서버를 가동시켜야합니다. 다음 가이드를 사용하여 Thingsboard를 설치합니다.

가이드:

<http://www.hackster.io/thingsboard/temperature-dashboard-using-arduino-uno-esp8266-and-mqtt-5e26eb>

2 단계. 기기 프로비저닝

이 단계에는 장치를 Thingsboard에 연결하는 데 필요한 지침입니다.

브라우저에서 Thingsboard 웹 UI (<http://localhost:8080>)를 열고 테넌트 관리자로 로그인합니다.

login : tenant@thingsboard.org

암호 : tenant

'장치'섹션으로 이동하십시오. "+"버튼을 클릭하여 "Arduino UNO Weight Input"라는 이름으로 장치를 생성합니다. 장치가 생성되면 세부 정보를 열고 "Manage credentials"를 클릭하고 "Access Token"필드에 "ARDUINO_DEMO_TOKEN"을 입력하고 "저장"을 클릭합니다. 기기 세부 정보에서 'Device ID'를 클릭하여 기기 ID를 클립 보드에 복사합니다.

3 단계. 대시 보드 제공

대시 보드 파일을 포맷에 맞게 작성합니다. 아래 링크는 제가 작성한 대시보드 포맷입니다. 이를 Import/Export를 이용하여 해당 대시 보드를 사용자의 Thingsboard 인스턴스로 가져옵니다.

<https://drive.google.com/file/d/159ybi65bEVX1wEQBBfQ0A7WJC895KbH2/view?usp=sharing>

B. Arduino UNO 디바이스 프로그래밍

1 단계. Arduino 라이브러리 설치

Arduino IDE를 열고 스케치 -> 라이브러리 포함 -> 라이브러리 관리로 이동하십시오 . 다음 라이브러리를 찾아서 설치합니다.

PubSubClient by Nick O'Leary.

WiFiEsp by bportaluri

Adafruit Unified Sensor by Adafruit

DHT sensor library by Adafruit

2 단계. 스케치 준비 및 업로드.

제가 작성한 MQTT 스케치는 아래 링크에 오렸습니다. 이에 이를 사용하기 위해 스케치에서 다음 상수 및 변수를 편집해야합니다.

MQTT스케치:

https://drive.google.com/file/d/1hGHeiA52G55Hh4ZQNfTDQaYKFaPoex2_/view?usp=sharing

WIFI_AP - 액세스 포인트의 이름

WIFI_PASSWORD - 액세스 포인트 암호

thingsboardServer - WIPI 네트워크에서 액세스 할 수 있는 작업 표시 줄 호스트 / IP 주소입니다.

이후 Arduino UNO 장치를 USB 케이블로 연결하고 Arduino IDE에서 "Arduino / Genuino Uno" 포트를 선택하십시오. "업로드" 버튼을 사용하여 스케치를 컴파일하고 장치에 업로드합니다. 응용 프로그램을 업로드하고 시작하면 mqtt 클라이언트를 사용하여 Thingsboard 노드에 연결을 시도하고 초당 한 번 "X 좌표" 및 "Y 좌표" timeseries 데이터를 업로드합니다.

C. 데이터 시작화

마지막으로 Thingsboard 웹 UI를 엽니다. tenant 관리자로 로그인하여 이 대시 보드에 액세스 할 수 있습니다.

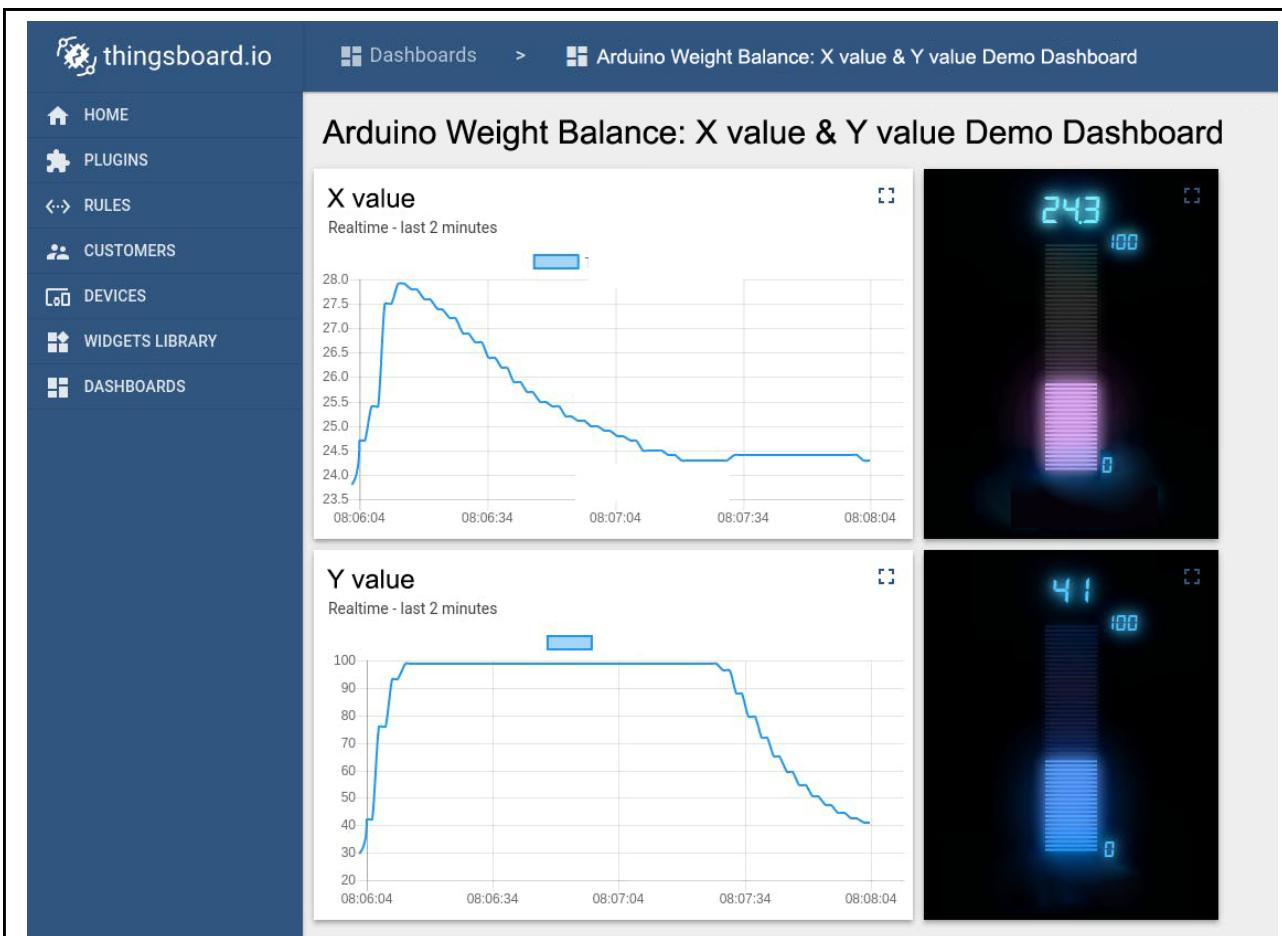
login : tenant@thingsboard.org

암호 : tenant

"Devices" 섹션에서 "Arduino UNO Weight Input"를 찾아 장치 세부 정보를 열고 "Latest telemetry" 탭으로 전환 합니다. 모두 올바르게 구성된 경우 테이블의 "X" 및 "Y" 값의 최신 값을 볼 수 있어야 합니다. 그런 다음 "Dashboards" 섹션을 연 다음 "Arduino Weight Balance: X value & Y value Demo Dashboard" 를 찾아 엽니다 . 결과적으로 무게중심의 X와 Y 좌표를 표시하는 두 개의 시계열 차트와 두 개의 디지털 게이지가 표시됩니다.

D. 결과물

해당 프로세스의 결과물은 다음과 같습니다.



이어 어플리케이션 소스코드에 대한 링크를 실수로 올리지 않아 다시 올립니다.

어플: https://drive.google.com/file/d/1A9cEqiRXgFtX6-eLf0UGDOc9F_f6iSXb/view?usp=sharing

현재까지의 공모전 수상경력:

2018 부산 스마트시티즌 공모전 우수상

스타트업캠퍼스 SDG캠프 결선 대상

위와 같이, 고려대학교 LINC플러스 사업단의 Capstone Design 결과보고서를 제출합니다.

2018 . 12. 04.

팀 장:	(인)
지도교수:	(인)

LINC사업단장 귀하