졸업 프로젝트 최종 보고서

프로젝트 제목 : AlphaFit

담당교수 : 김경창 교수님

학번 / 이름 : B311018 김도현

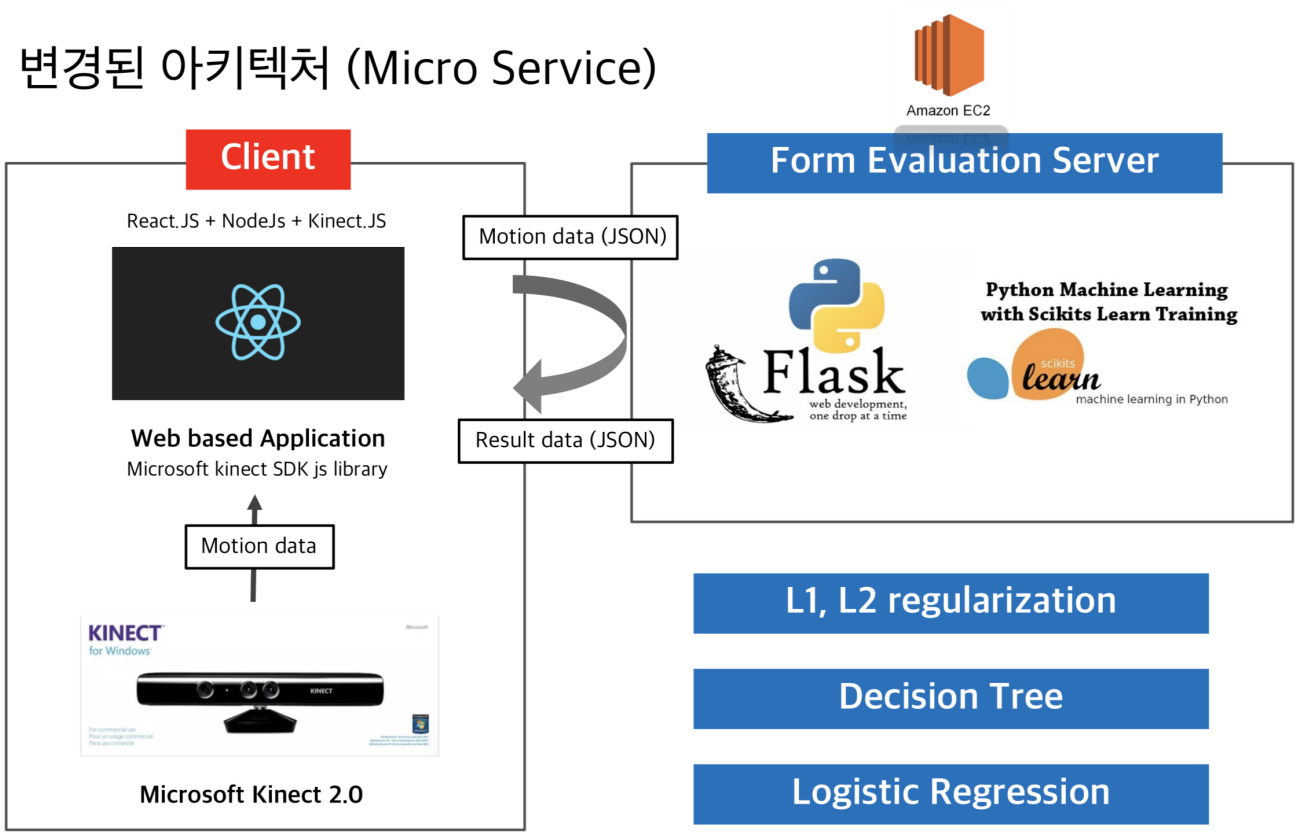
1. 프로젝트 개발 동기 / 의의

건강한 라이프스타일은 생각보다 평범한 운동에 달려있습니다.
그러나 많은 훈련자들이 올바르지 못한 운동 자세로 고통받고 있고, 부상의 위험성을 늘이며 운동이 주는 건강의 이득을 감소시키고 있습니다. 운동 자세에 대한 자세한 피드백을 받으려면, 일반적으로 퍼스널 트레이너가 필요합니다. 저희는 이러한 운동 자세에 대한 코칭을 **머신러닝** 으로 해결하고 싶었습니다.

기존에도 운동동작에 대한 평가 시스템은 존재했습니다. 그러나 **머신러닝** 을 적용한 케이스는 거의 찾을 수 없었습니다. 예를 들어 이화여대 대학원에서도 키넥트 센서를 이용한 영상 처리 및 운동동작 피드백 제공에 관한 연구가 있었지만, 이는 기계학습 없이 자체 개발한 알고리즘으로 처리하였고, *athos*, *Nike training* 등 다양한 트레이닝 프로그램들이 나와 있으나 알고리즘을 적용한 케이스 일 뿐 머신러닝을 이용한 케이스는 거의 못했습니다.

저희는 일반적인 하체운동인 맨몸스쿼트에 집중하여 이 문제를 해결하고자 하였습니다. 저희의 목적은 Kinect 로부터 얻어진 동작 데이터와 머산러닝 동작 판별기를 통해 사용자들이 운동 동작에서 어떠한 부분을 개선할 수 있는지 보여주고 싶었습니다.

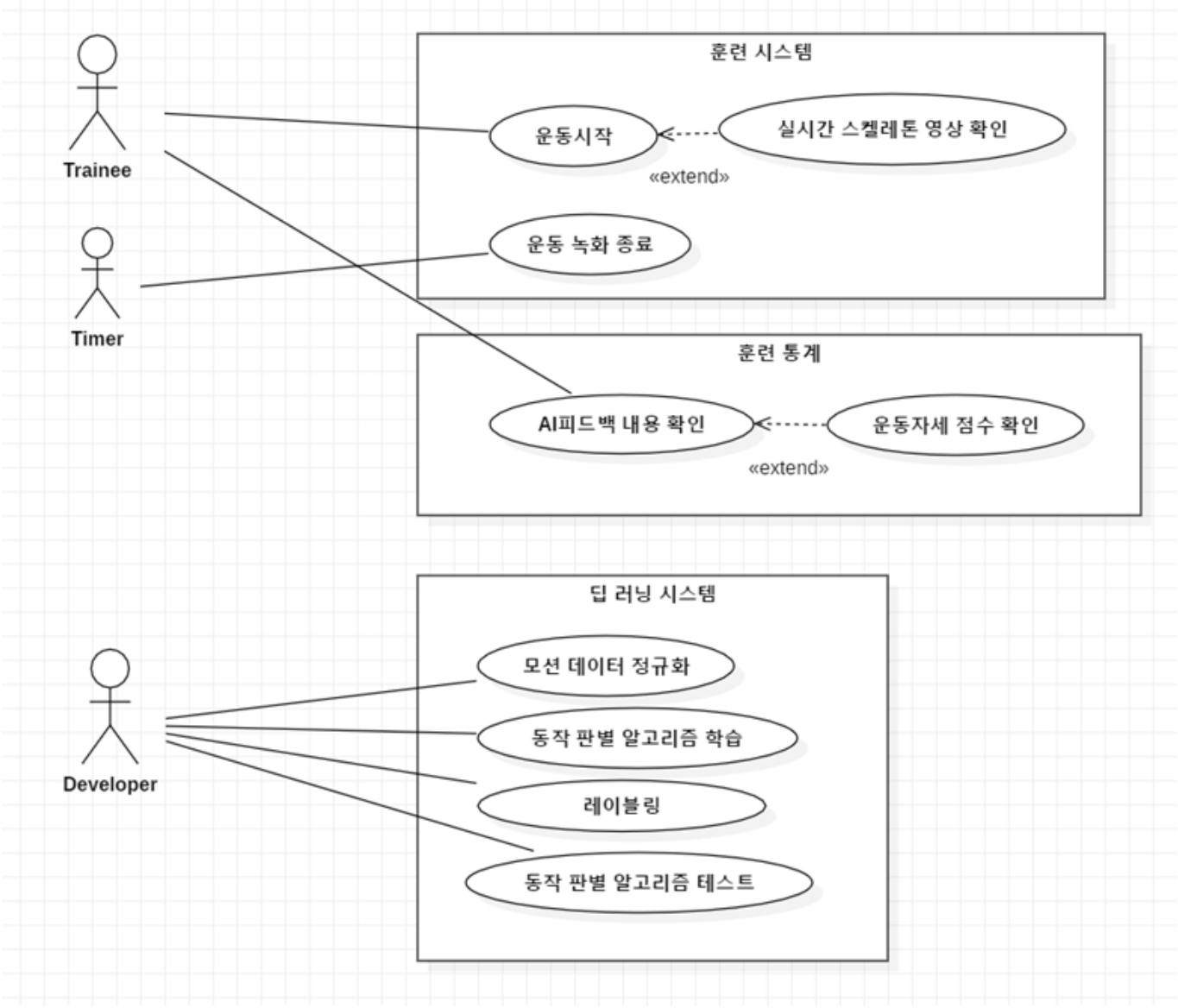
2. 아키텍쳐 / 기술 스택



- **Client** 는 최근 현업에서 가장 수요가 많은 자바스크립트 기반 Frontend Framework 인 React.js 를 사용하여 효율적으로 화면 컴포넌트를 구성하여, SPA (Single Page Application) 을 제작하였습니다.
- 또한 원활한 키넥트 기기 - 웹 어플리케이션 간 연동, 그리고 HTTP 통신을 위해 Node.Js 기반 클라이언트 서버를 구성하였습니다. Node.Js 는 서버 구축이 간단하고, JSON 형태의 데이터를 쉽게 핸들링 할 수 있어 좋았고, 키넥트 디바이스 관련 모듈이 존재하여 디바이스 연동 또한 용이했습니다.
- **Server** 은 구축이 간단하며, 머신러닝 모델과 Merge 가 용이한 Python Flask 서버 프레임워크를 사용하여 생산성을 높일 수 있었습니다.
  또한 Amazon Web Services 의 EC2 Instance (Linux Ubuntu) 클라우드 서버를 활용하여 물리장비 없이 간단하게 인프라 환경을 구축할 수 있었습니다.
- **Maching Learning** 프레임워크는 불필요한 기능이 많은 텐서플로우 대신, 사용이 간편하며 여러 수학적 연산을 제공하는 Scikit-Learn 을 사용하였습니다.

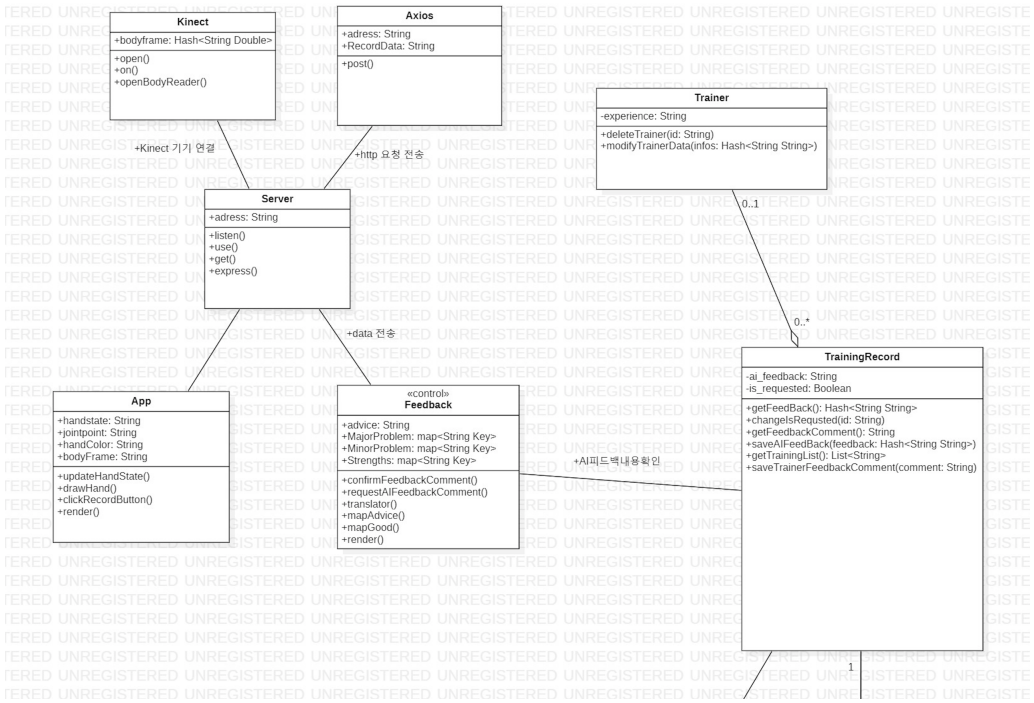
3. Use case diagram

> 초기에는 운동 통계기능이나, 실제 트레이너 피드백 기능 등 부가적인 기능이 더 있었으나, 추후에 머신러닝 모델 고도화나 웨어러블 장비 교체 이슈 등 때문에 개발이 다소 늦어지게 되어, 핵심 요구사항만 개발하는 것으로 타협하였습니다.

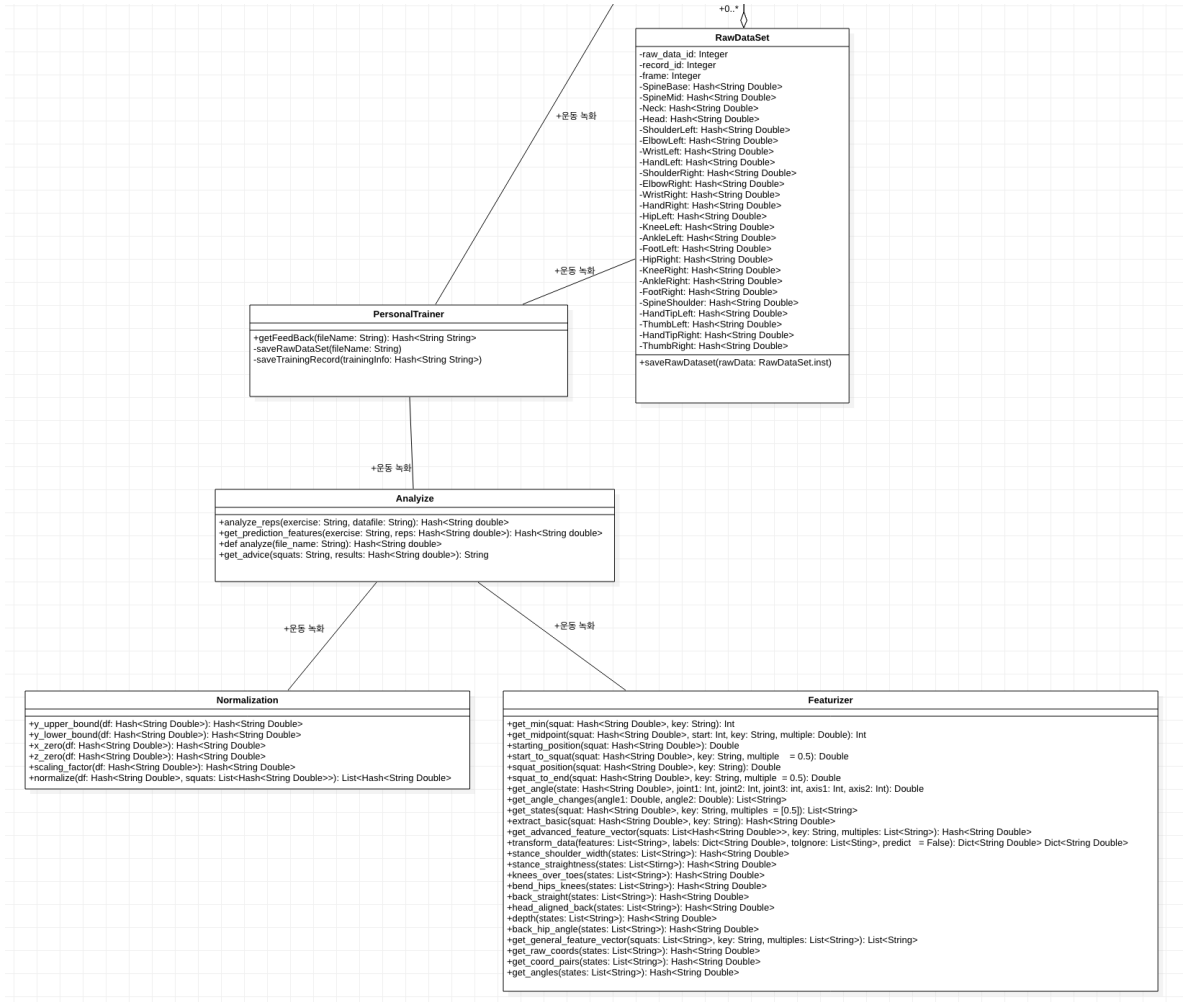


4. Class diagram

(1) Upper Part (Client)



(2) Lower Part (Server)



5. 내가 수행한 것 (프로젝트 회고)

1) Node.js + Express + Kinect.js 를 이용한 Kinect 기기 연동

키넥트 기기는 마이크로소프트에서 만든 동작 녹화 및 인식을 도와주는 하드웨어로, SDK를 제공하고 있으나 대부분이 C#, C++ 언어에 최적화 되어 있습니다. Javascript SDK 도 제공하고 있으나 C#에 비해 기능이 제한적이며 사용한 사례가 거의 없었습니다. 그러나 저는 유려한 화면과 좀더 설치와 실행이 쉬운 웹 어플리케이션으로 제작하고자 하였으며 이를 위해서는 기존 제공하는 javascript SDK를 그대로 사용하는 형태가 아니라 저희 요구사항에 맞게 수정해서 사용해야 했습니다.

저는 이를 위해 NodeJs + Express 라는 자바스크립트 기반 서버 프레임워크와 기존 C++ 기반 SDK를 Wrapping 한 kinect.js 라는 자바스크립트 라이브러리를 이용하여 클라이언트 앱 구동 시 실시간으로 키넥트 기기로부터 동작 데이터를 송신받을 수 있도록 구성하였습니다.

2) 웹소켓과 Canvas element 를 이용한 실시간 skeletical data 스트리밍 및 클라이언트 렌더링

더 나은 어플리케이션 사용성을 위해, 동작 녹화 도중 저희는 실시간으로 저희의 동작 모습을 확인할 수 있도록 하고 싶었습니다. 이 또한 역시 C#이나 C++ 기반 어플리케이션에서는 쉽게 동작하는 모습을 화면에 렌더링 할 수 있도록 제공하는 라이브러리가 제공되었으나, 자바스크립트를 이용해 웹 어플리케이션 html 마크업 화면에 그릴 수 있는 라이브러리는 전혀 제공하지 않고 있었습니다.

따라서 저는 웹소켓을 이용하여 실시간으로 kinect 기기에서 송출되는 데이터를 Client Side 로 포워딩하도록 구현하였고, React 기반 클라이언트 측에서 해당 데이터를 이용하여 html canvas element 에 렌더링 할 수 있도록 비즈니스 로직을 구성하였습니다.

```
// NodeJs Client Core Server
// 클라이언트 소켓으로 녹화된 bodyFrame 송신
kinect.on('bodyFrame', function(bodyFrame){
    io.sockets.emit('bodyFrame', bodyFrame);
});
```

```
// React Client
// 클라이언트 소켓으로 녹화된 bodyFrame 수신 및 렌더링
self.socket = io(serverAddress);

self.socket.on('bodyFrame', bodyFrame => {
  let ctx = self.display.current.getContext('2d');
  ctx.clearRect(0, 0, self.display.current.width,
self.display.current.height);
  let index = 0;
  let colors = self.state.colors;

  bodyFrame.bodies.forEach(function(body){
    if(body.tracked) {
      for(let jointType in body.joints) {
        let joint = body.joints[jointType];
        ctx.fillStyle = colors[index];
        ctx.fillRect(joint.depthX * 512, joint.depthY * 424, 10, 10);
      }
```

```
        //draw hand states
        self.updateHandState(body.leftHandState, body.joints[7]);
        self.updateHandState(body.rightHandState, body.joints[11]);
        index++;
      }
    });

});
```

3) AWS + Python Flask API 서버 구축 / 머신 러닝 모듈 결합

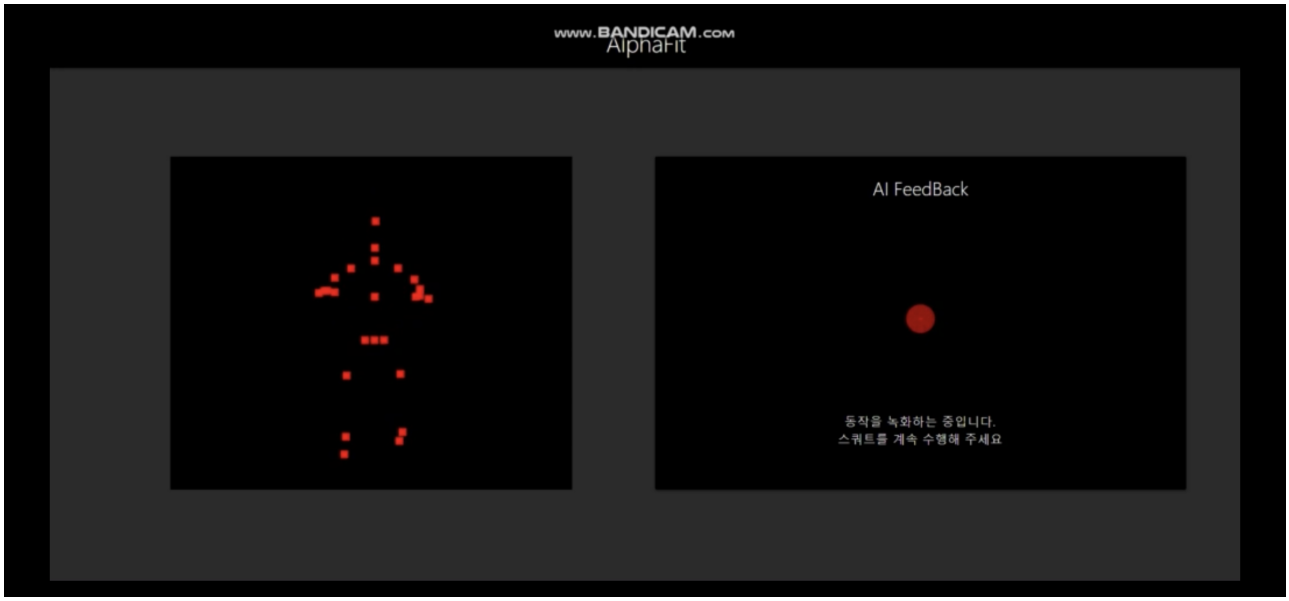
머신러닝 모듈과의 용이한 결합과 편리한 API 개발을 위해 아마존 웹 서비스 Linux ubuntu server 인스턴스 와 Flask Framework 를 이용히여 서버를 구축하였습니다.

```
#  RECEIVE POST METHOD HTTP REQUEST
class AnalyzeRaw(Resource):

    def post(self):
        req_data = request.get_json()
        to_write = open('squatData.txt', 'wb')
        to_write.write(req_data['data'].encode("utf-8"))
        ut.print_success('Data written to file')
        json_res = jsonify(advice())
        ut.print_success('Advice file generated')
        return json_res


api.add_resource(AnalyzeRaw, '/analyze_raw')
```

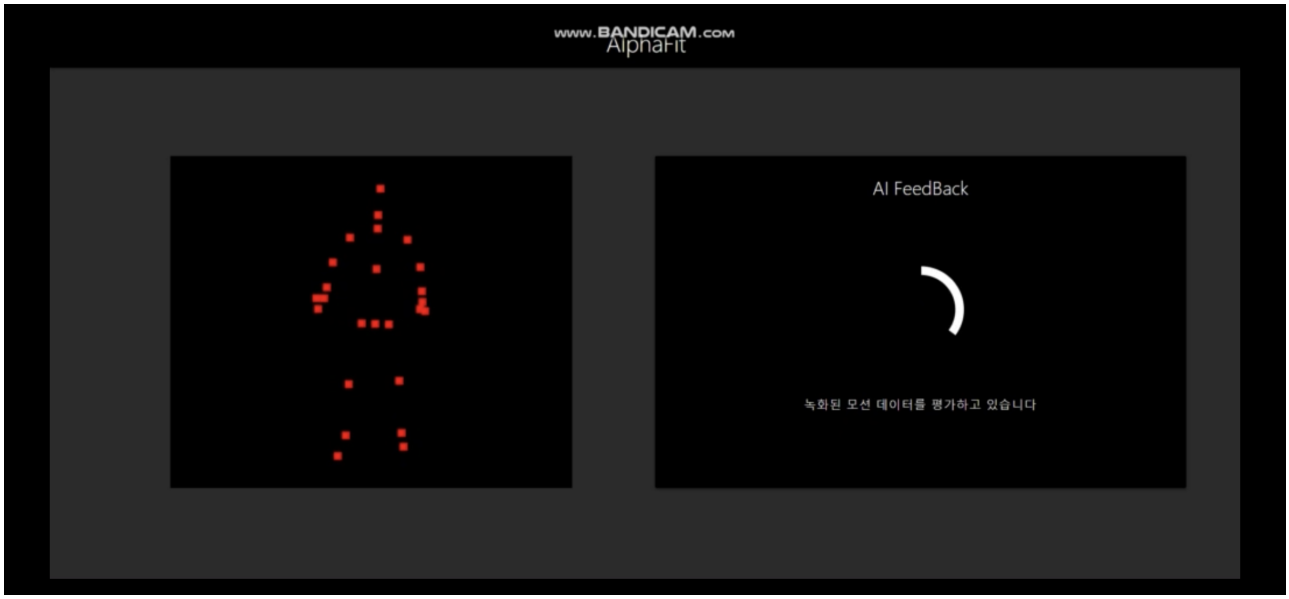
6. 시연 스크린샷

1. 녹화 시작 전

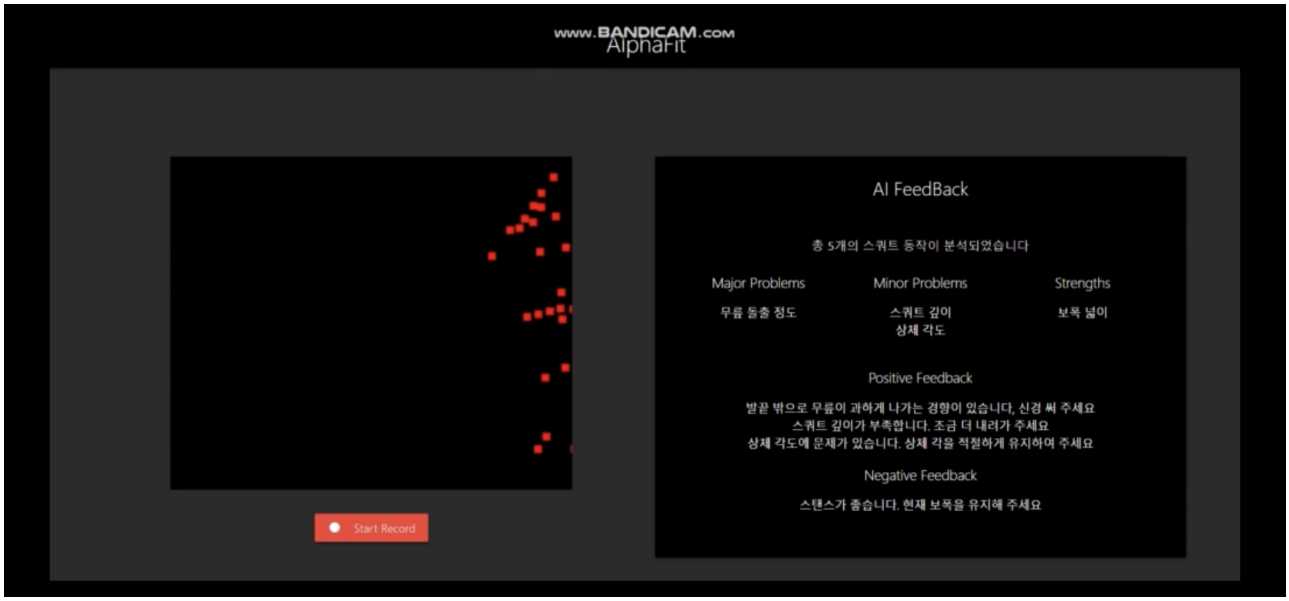
2. 녹화중



3. 평가중



4. 평가완료



7. 결론 / 소감

저는 일반적인 소프트웨어 개발 프로젝트는 많이 경험해 보았지만, 머신러닝을 이용한 협업 프로젝트는 처음이였습니다. 또한 머신러닝 기술을 사용하겠다고 했지만, 동작 분석에 관하여는 생각보다 레퍼런스가 없어 초반에는 정말 막막했던 부분이 많았습니다. 따라서 이러한 부분들을 해소하기 위해 팀원들과 머신러닝 관련한 여러 논문들을 찾아보기도 하고 관련 서적들과 자료들을 참고했던 경험들이 많이 기억에 남습니다.

그리고 팀원들 중 실무 프로그래밍 경험이 있는 제가 자연스럽게 기술 리딩을 맡게 되었고, 상황에 맞는 적절한 프로그래밍 도구들과 프레임워크를 선택하여 역할 분배를 하고 새로운 시도를 해 보는 것 자체가 정말 즐거웠습니다. 비교적 흔히 해볼 수 있는 단순한 구현 뿐 아니라 요구사항 분석부터 각종 다이어그램 표현까지 소프트웨어 공학에서 이야기 하는 설계의 부분들을 실제로 체험해 볼 수 있어서 정말 유익하기도 했습니다. 여름방학 때, 실제 기업에서 인턴을 진행할 때에도 시퀀스 다이어그램이나 클래스 다이어그램을 그려본 경험들이 정말 많은 도움이 되었습니다.

따라서 저는 이 프로젝트를 통해 저는 막연하게 가지고 있었던 머신러닝에 대한 두려움을 많이 덜게 될 수 있었을 뿐 아니라, 서로 다른 배경과 실력들을 가진 팀원들과의 협업, 요구사항 분석부터 그것을 기반으로 구현하는 것까지 모든 부분들을 체험해 볼 수 있었던 최고의 경험이였습니다.