<피치 측정 프로그램>

오픈소스SW기초 2분반

10조 팀프로젝트

조장 : 허윤서()

조원 : 김형준()
이상진()
이찬혁()

- 목차-

1. 프로젝트 개요

‘피치 측정 프로그램’은 사용자의 음성이나 음악 신호를 입력받아 소리의 높낮이, 즉 피치를 분석하고 시각화하는 웹 기반 애플리케이션이다. 기존의 전문 음향 분석 툴은 접근성이 낮고 인터페이스가 복잡하여 일반 사용자가 활용하기 어렵다는 한계가 있다. 이에 본 프로젝트는 오픈소스 라이브러리와 웹 프레임워크(Node.js)를 기반으로 누구나 쉽게 사용할 수 있는 피치 측정 서비스를 제공하는 것을 목표로 한다.

주요 기능으로는 • 음성 파일 업로드 및 관리, • 시간별 파형 및 피치 측정, • 파일 간 비교 분석, • 실시간 마이크 입력 피치 측정 및 시각화 등이 있다. 이를 통하여 음악 교육 및 다양한 실생활 영역에서 활용 가능하도록 설계한다.

본 프로젝트는 약 3개월간 진행되며, 핵심 기능만 구현한 최소 기능 제품을 우선 개발 한 뒤, 단국대학교 학생들을 대상으로 실제 사용 환경을 고려하는 것을 목표로 한다. 개발이 완료되면 간단한 사용자 테스트를 통해 기능의 실효성 및 편의성을 검토하며 피드백을 받아 개선할 예정이다.

2. 필요성 및 문제정의

대부분의 일반 사용자들은 • 노래 연습 • 발표 준비 • 외국어 발음 교정 • 악기 튜닝 등 다양한 상황에서 자신의 음성이나 소리의 높낮이를 확인하고 싶어 하지만, 기존의 상용 프로그램은 가격이 비싸거나 전문가용으로 복잡하여 접근이 쉽지 않다. 반대로 무료로 제공되는 일부 애플리케이션은 정확도가 낮거나 단순 수치 출력에만 그쳐, 실질적인 도움이 되지 않는 경우가 많다.

현재 사용 가능한 서비스들은 • 전문가 지향적 인터페이스 • 실시간 분석 부족 • 시각화 미흡 • 사용자 맞춤형 기능 부재 등 여러 문제점을 가지고 있다. 특히 교육용이나 개인 학습용으로 활용하기 위해서는 간단하면서도 정확한 피치 측정과 직관적인 시각화 기능이 필수적이지만, 이를 충족하는 오픈소스 기반 솔루션은 드문 상황이다.

따라서 본 프로젝트는 이러한 문제점을 해결하기 위해서는 누구나 쉽게 접근할 수 있는 웹 기반 피치 측정 프로그램을 Node.js 환경에서 개발하고자 한다. ‘피치 측정 프로그램’은 이러한 문제 인식을 바탕으로 기획된 프로젝트로, 이를 통해 학생, 일반 사용자, 음악 사용자들이 자신의 음성이나 음악을 분석하고 개선할 수 있는 환경을 제공하며, 나아가 오픈소스 협업을 통한 확장성과 활용성을 높이는 것을 목표로 한다.

3. 핵심 기능 및 개발 전략

3.1 핵심 기능

• 음성 파일 업로드 및 관리
- 파일 업로드 (WAV, MP3 등 다양한 포맷 지원)
- 파일 삭제 기능
- 카테고리 또는 폴더 생성 및 삭제

• 시간별 파형 및 피치 측정
- 파일 선택 시 시간별 파형 표시 (Wavesurfer.js 활용)
- 오디오 재생 및 정지 기능
- 되감기 및 빨리감기 기능
- Web Audio API 기반 피치(Hz) 검출
- 시간별 피치 그래프 시각화 (Chart.js, Canvas API 등)
- 분석 결과 저장 기능 (CSV, 이미지 등)

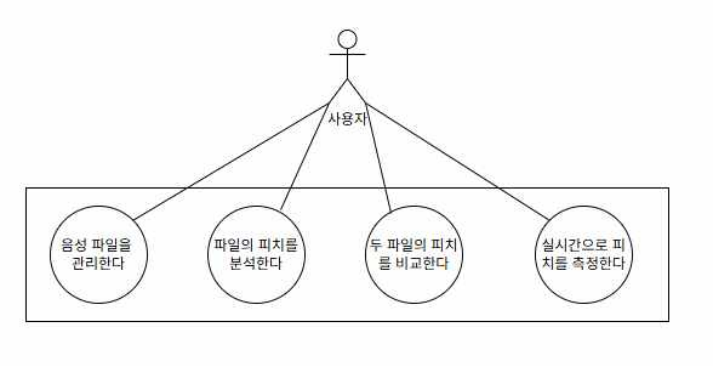
• 파일 간 비교 분석
- 두 개 파일 동시 선택 기능
- 시간별 파형을 위·아래로 배치해 비교 표시
- 각 파일의 피치 정보 표시
- 파일별 재생 및 정지 기능 제공
- 두 파일 개별 컨트롤(되감기, 빨리감기) 지원

• 실시간 마이크 입력 피치 측정 및 시각화
- 브라우저 마이크 권한 요청 및 입력 처리
- 실시간 피치 측정(Hz 단위) 기능
- 시간별 피치 변화 그래프 시각화
- 실시간 파형 표시 및 업데이트

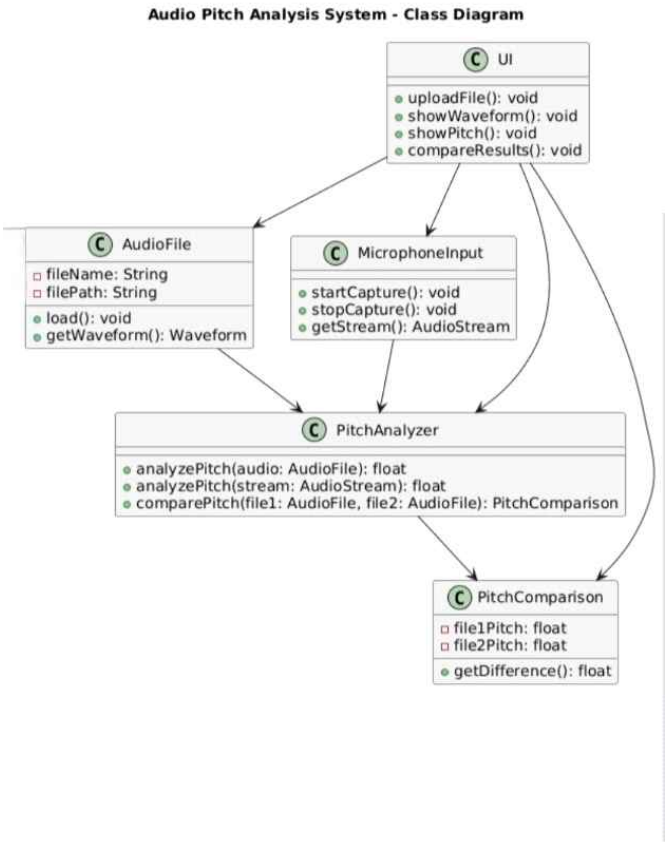
3.2 개발 전략 - (UML 다이어그램, 사용자 시나리오, 화면 및 메뉴 구성 스케치)

• UML 다이어그램

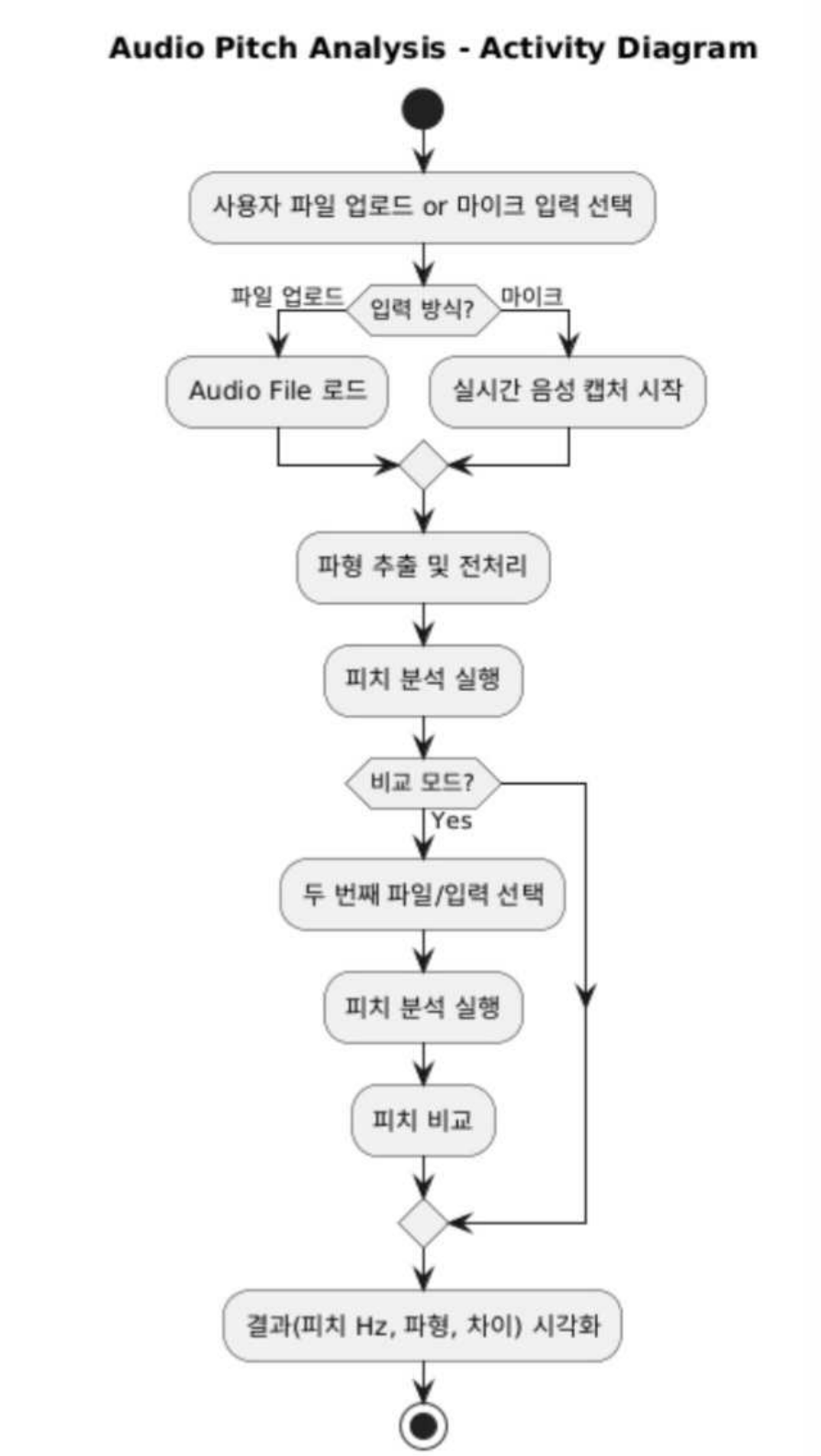
- 유스케이스 다이어그램



- 클래스 다이어그램



- 액티비티 다이어그램


Audio Pitch Analysis - Activity Diagram
사용자 파일 업로드 or 마이크 입력 선택
파일 업로드
입력 방식?
마이크
Audio File 로드
실시간 음성 캡처 시작
파형 추출 및 전처리
피치 분석 실행
비교 모드?
Yes
두 번째 파일/입력 선택
피치 분석 실행
피치 비교
결과(피치 Hz, 파형, 차이) 시각화

• 사용자 시나리오

[시나리오 제목]: 실시간 피치 측정을 통한 외국어 억양 연습

[페르소나]: 24세 취업준비생, 영어 면접 스터디 중

[시나리오]:

1. '실시간 마이크 입력' 기능을 실행한다.
2. 원어민의 의문문 음성을 듣고 피치 그래프의 모양을 확인한다.
3. 마이크에 대고 같은 문장을 따라하며 내 피치 그래프를 실시간으로 확인한다.
4. 원어민의 상승하는 그래프와 자신의 평탄한 그래프의 차이를 즉시 인식한다.
5. 그래프 모양을 거울삼아 문장 끝을 올리는 연습을 반복한다.

→ 이 시나리오는 눈에 보이지 않는 '억양'을 실시간 그래프로 시각화하여, 사용자가 즉각적인 피드백을 통해 언어 학습의 효율을 극대화하는 과정을 보여줍니다.

• 화면 및 메뉴 구성 스케치

3.3 개발 일정 및 개발 방법

1개월 차 - 기획 및 기본 기능 개발

• 프로젝트에서 필요한 기능을 정리하고, 주요 화면 구성을 위한 UI/UX 시안을 작성

• Node.js(백엔드)와 React(프론트엔드) 개발 환경 세팅

• Github를 통한 버전 관리 및 협업 환경 구축

2개월 차 - 주요 기능 개발 및 테스트

• 파일 관리 기능 구현 (업로드, 삭제, 폴더/카테고리 관리, 상세 페이지)

• 목록에서 파일 선택 시 검색 및 정렬 기능 구현

• Wacesurfer.js를 활용해 오디오 파형 시각화 및 재생 위치 제어 기능 개발

• Web Audio API로 오디오 데이터를 얻고 Pitch detection 알고리즘 (Pitchfinder.js 또는 Meyda.js)을 활용하여 오디오 데이터에서 주파수 검출 기능 구현

• Canvas API/Chart.js/D3.js로 실시간 그래프 표시 구현

• 기능별 단위 테스트 진행 및 오류 수정

3개월 차 - 배포 및 마무리

• Node.js 백엔드를 Render등 호스팅 서비스에 배포

• React 프론트엔드를 Vercel을 통해 배포

• 전체 기능 통합 테스트 및 성능 개선

• 모바일 환경 대응을 위한 UI 조정

• 사용자 피드백을 반영하여 일부 기능 개선 및 프로젝트 마무리

3.4 팀원 역할 분담

조장 허윤서 - UI/브라우저 환경
- React 기반 전체 UI/UX 설계 및 페이지 라우팅
- 메인 페이지, 메뉴, 결과 표시 등 전반적인 UI 구성
- 모바일/브라우저 환경 호환성 점검 및 반응형 디자인 적용
- 시각화 요소(그래프, 파형) 연결

팀원 김형준 - 파일 관리 담당
- 음성 파일 업로드 및 삭제 기능 구현
- 카테고리 및 폴더 생성/삭제 기능 개발
- 업로드된 파일 목록 표시 및 정렬/검색 기능 제공
- 데이터베이스에 파일 메타데이터 저장 및 연동

팀원 이상진 - 측정 & 비교 담당
- 파일 선택 시 시간별 파형 시각화 및 피치 분석 기능 구현
- Wavesurfer.js, Web Audio API 연동을 통한 오디오 재생/정시/되감기/빨리감기 기능 구형
- 두 개 파일 동시 선택 시 파형 비교표시 그래프 구현
- 각 파일별 피치 정보 출력 및 개별 컨트롤 기능 개발

팀원 이찬혁 - 실시간 마이크 입력
- 브라우저 마이크 입력 권한 처리 및 오디오 스트림 수집
- Web Audio API 기반 실시간 피치 분석 알고리즘 연동
- Char.js, Canvas API를 활용한 시간별 피치 변화 그래프 표시

4. 위험 요소 및 평가 방법

예상 문제	평가 및 대책
실시간 분석 성능 저하	- 프로젝트 초기에 핵심 기능만 구현한 프로토타입으로 성능을 집중 테스트합니다. - 성능 문제 발생 시, 그래프 갱신 빈도를 조절하거나 데이터 처리 로직을 최적화합니다.
외부 라이브러리 의존성	- 개발 시작 전, 각 팀원이 맡은 라이브러리로 작은 예제(PoC)를 만들어 기술적 장벽을 미리 확인하고 학습 시간을 갖습니다.
모바일 환경 UI 깨짐	- 초기 설계 단계부터 반응형 디자인을 적용하고, 개발 과정에서 지속적으로 모바일 환경 테스트를 병행합니다.
브라우저 호환성 문제	- 매주 정기적으로 주요 브라우저(크롬, 사파리 등)에서 교차 테스트를 진행하여 호환성 문제를 조기에 발견하고 해결합니다.
개발 일정 지연	- Github Projects나 Trello 같은 협업 툴로 진행 상황을 투명하게 공유합니다. - 주간 회의를 통해 일정 지연 여부를 평가하고, 필요시 업무 우선순위를 재조정합니다.
요구사항 변경 및 확장	- 프로젝트 시작 시, 반드시 구현해야 할 최소 기능 제품(MVP)의 범위를 명확히 문서화하고 합의합니다. - 추가 아이디어는 '아이디어 백로그'에 기록해두되, MVP 개발이 완료된 후에 논의하여 반영 여부를 결정합니다.

5. 참고문헌 및 용어 설명

- 참고 문헌

1. Mozilla Developer Network (MDN). (2025). https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API

2. katspaugh. (2025). Wavesurfer.js. https://wavesurfer.xyz/

3. Chart.js Team. (2025) https://www.chartjs.org/docs/latest/

4. Patten, P. (2025). Pitchfinder [Software]. https://github.com/peterkhayes/pitchfinder

5. De Cheveigné, A., & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. The Journal of the Acoustical Society of America, 111(4), 1917–1930. http://audition.ens.fr/adc/pdf/2002_JASA_YIN.pdf

- 용어 설명

피치 (Pitch) : 소리의 높낮이를 나타내는 속성으로, 진동수에 의해 결정됨. 이 프로젝트의 핵심 분석 대상임.

Hz (Hertz) : 주파수의 단위로, 1초 동안의 진동 횟수를 의미함. 피치의 높낮이를 객관적인 수치로 표현할 때 사용됨.

Web Audio API : 웹 브라우저에서 오디오 데이터를 직접 처리하고 합성할 수 있도록 하는 프로그래밍 인터페이스. 실시간 마이크 분석, 피치 검출 등 핵심 기능 구현에 사용됨.

API (Application Programming Interface) : 특정 프로그램의 기능이나 데이터를 다른 프로그램이 접근하여 사용할 수 있도록 미리 정해놓은 통신 규칙.

Node.js : JavaScript를 브라우저 밖(서버)에서도 실행할 수 있게 해주는 환경. 이 프로젝트에서는 백엔드 서버 구축에 사용됨.

React : 사용자 인터페이스(UI) 구축을 위한 JavaScript 라이브러리로, 이 프로젝트에서는 프론트엔드 개발에 사용됨.

Wavesurfer.js : 오디오 파일의 파형(waveform)을 웹 페이지에 시각적으로 그려주는 JavaScript 라이브러리.

Chart.js : 시간의 흐름에 따른 피치 변화와 같은 데이터를 꺾은선 그래프 등 다양한 형태로 시각화해주는 JavaScript 라이브러리.

Pitchfinder.js : 오디오 데이터에서 특정 알고리즘(YIN 등)을 사용하여 피치 값을 찾아내는 JavaScript 라이브러리.

UI (User Interface) : 사용자 인터페이스. 사용자가 프로그램을 편리하게 사용할 수 있도록 구성된 화면, 메뉴, 버튼 등의 시각적 요소.

UX (User Experience) : 사용자 경험. 사용자가 특정 서비스를 이용하면서 느끼는 만족도, 편의성 등 총체적인 경험.

호스팅 (Hosting) : 개발된 웹 애플리케이션을 인터넷 사용자들이 접속할 수 있도록 서버 공간의 일부를 빌려주는 서비스.

최소 기능 제품 (MVP, Minimum Viable Product) : 핵심 기능만 최소한으로 구현한 초기 버전의 제품. 시장 반응을 빠르게 확인하고 피드백을 받아 개선하기 위해 개발함.