혁신성장 청년인재 집중양성 사업

Win:G

인디 음악 홍보 플랫폼

2020 년 06 월 20 일

클라우드 기반 마이크로 서비스 전문가 양성 과정

A 반 3 조 - 노말(NoTalk)

이종화, 이대희, 이민기, 탁성건

목 차

1. 프로젝트 개요 1

1.1 프로젝트 기획 배경 및 목표 1

1.2 구성원 및 역할 1

1.3 프로젝트 추진 일정 2

2. 프로젝트 현황 3

2.1 시장분석 3

2.2 기존 서비스의 한계 4

2.3 차별화 핵심 전략 기술 4

3. 프로젝트 개발 결과 5

3.1 개발방법론 및 도구 5

3.1.1 애자일 방법론 채택 이유 및 적용 5

3.1.2 협업에 사용된 도구 5

3.2 Jira 를 통해 본 스프린트 7

3.2.1 로드맵 7

3.2.1 누적 플로우 다이어그램 8

3.3 프로젝트 구조 및 파일 9

3.3.1 MSA 아키텍처 9

3.3.2 DevOps Pipeline 10

3.3.3 Docker Swarm 클러스터 - Visualizer 10

3.3.4 RESTful API - Swagger 11

3.3.5 프로젝트 트리 12

3.4 주요 기능 13

3.4.1 로그인 - Google Login API 활용 13

3.4.2 마이 페이지 14

3.4.3 스트리밍 14

3.4.4 라이브 & 채팅 15
3.4.5 후원 16
3.4.6 콘서트 정보 17
3.4.7 통합검색 17
4. 기대 효과 18
4.1 기대 효과 18
4.2 향후 개선 사항 19
4.2.1 기술적 개선 사항 20
4.2.2 서비스적 개선 사항 20
5. 개발 후기 22

1. 프로젝트 개요

1.1 프로젝트 기획 배경 및 목표

마이크로서비스 전문가 양성과정을 통해 배운 Infra, Front-end, Back-end, CI/CD 내용을 충실히 반영하며, 마이크로서비스 아키텍처에 적합한 Win:G 프로젝트를 기획하였습니다. Win:G 란 뮤지션들에게 날개를 달아준다는 뜻의 Wing 과 목표를 이루는 이룬다는 뜻의 Win + Goal 을 합쳐서 만들었습니다.

저희는 인디 뮤지션들이 편향된 음악 시장에서 음악 활동을 하면서 어려움을 느낀다는 것과 사용자들이 본인들이 좋아하는 뮤지션들을 접하기 힘들다는 문제점을 해결하기 위해 Win:G 플랫폼을 기획했습니다.

Win:G 는 인디 뮤지션과 대중들의 만남의 장을 만드는 것을 목표로 하여 음원 스트리밍 서비스, 실시간 Live 서비스, 인디 뮤지션 후원 서비스, 공연 정보 서비스를 제공하는 서비스 플랫폼입니다.

사용자들은 Win:G 를 통해 개성 있고 실력 있는 인디 뮤지션들의 음악을 들을 수 있고, 언제 어디서든 인디 뮤지션들의 실시간 공연 영상을 시청할 수 있습니다. 그리고 좋은 음악이 있지만 열악한 음악 환경 때문에 음반 제작이 힘든 뮤지션들은 후원 서비스를 통해 후원을 받을 수 있으며, 뮤지션별 공연 날짜 및 공연 정보 확인이 가능합니다. Win:G 는 뮤지션과 사용자들의 즐거운 놀이터를 목표로 합니다.

1.2 구성원 및 역할

이름	전공	역할	구현 부분
이종화	수학과	팀장	Music Service

			CI/CD
이대희	수학과	팀원	Concert Service
이민기	경영정보학과	팀원	User Service
탁성건	정보제어학과	팀원	Support Service Live Service

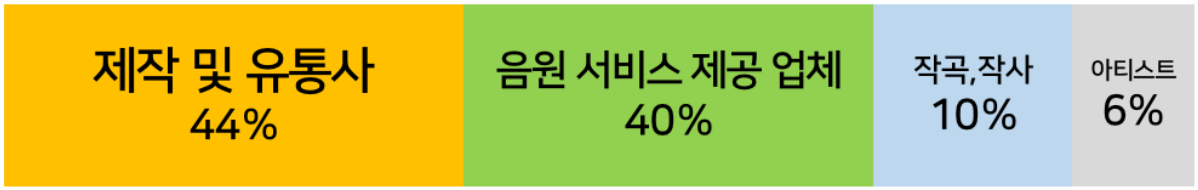
1.3 프로젝트 추진 일정

구분	기간	활동
사전 기획 및 아이디어 회의	01.28 ~ 01.31	아이디어 회의 및 개발 주제 선정 강의 시간 내 발표
서비스 도출 및 기획	02.12 ~ 02.23	시장분석 및 기획안 도출
기획안 구체화 및 협업 툴 설정	02.24 ~ 02.28	서비스 기획안 확정 Slack, Jira, Confluence, GitHub 세팅
1 차 스프린트 (요구사항 정의서, Prototyping)	04.13 ~ 04.17 04.23 ~ 04.25	UI Prototype 제작(Marvel App 활용) 요구사항 정의서 작성
2 차 스프린트 (모놀리식 아키텍처)	04.27 ~ 04.28	GitHub Project Repository 생성 프로젝트 환경설정
	05.03 ~ 05.19	Front-end 개발
	05.20 ~ 06.01	Back-end API 개발 및 연동
3 차 스프린트 (마이크로서비스 아키텍처 & CI/CD)	06.01 ~ 06.10	마이크로서비스 분할 Docker Image 생성
	06.08 ~ 06.14	Docker Swarm 클러스터 구축 Jenkins 를 활용한 배포 자동화
4 차 스프린트 (최종 발표 준비)	06.15 ~ 06.19	최종 보고서 작성 시연 영상 및 발표 PPT 제작

2. 프로젝트 현황

2.1 시장분석

KBS2 에 따르면 국내 음원 수익분배 구조에서 아티스트가 차지하는 몫은 6% 정도이며, 작사와 작곡에 참여하는 경우 10%를 가져간다고 합니다. 이는 음원으로 발생한 수익이 1,000 원일 경우, 아티스트가 받게 될 수익은 최대 150 원 정도라는 것입니다.

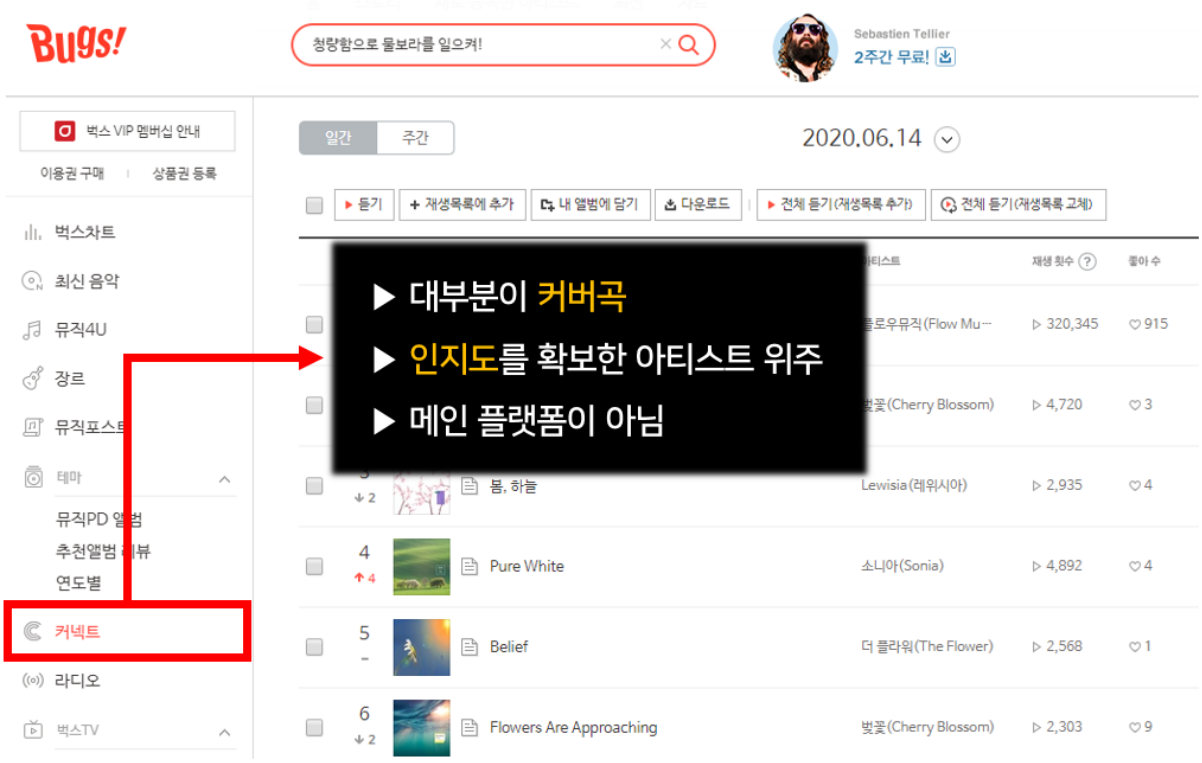


한국문화예술위원회의 "인디 음악의 현주소에 대한 다각적 분석과 발전 방향에 대한 연구(2013)" 논문에 따르면 인디 뮤지션들이 음악 활동을 하면서 가장 어려운 점을 경제적 어려움(33%)과 열악한 음악 환경(25%)을 뽑았습니다. 또한 이들이 불공정하다고 느끼는 것은 편향된 음악 시장 (36%), 지나치게 적은 출연료 (19%), 편향적인 미디어 환경 (18%) 으로 답했습니다.

홍대에서 활동하는 인디 뮤지션들은 500~600 팀 정도이지만 라이브 클럽은 20 개 남짓입니다. 또한 수요가 많은 힙합 공연과의 경쟁에서 밀려, 인디 뮤지션이 설 자리는 더더욱 없어지고 있습니다. 홍대 상권의 젠트리피케이션으로 인해 기존의 인디 뮤지션들이 홍대를 떠나는 상황입니다.

이러한 인디 뮤지션들을 위해 현재 나와 있는 서비스로는 대표적으로 벅스 커넥트가 있습니다.

2.2 기존 서비스의 한계



'벅스 커넥트'의 경우 인디 음악 차트, 새로운 아티스트 소개, 인디 음악 추천 등의 기능을 제공하고 있습니다. 하지만 대부분의 곡들이 유명 아티스트들이 발표한 곡을 편곡해 부른 커버곡이며, 이미 인지도를 확보한 아티스트 비중이 높습니다. 또한 메인 플랫폼이 아니기 때문에 '커넥트'라는 서비스는 주목 받기 어렵습니다.

2.3 차별화 핵심 전략 기술

위에서 언급한 인디 뮤지션의 어려움, 기존 서비스의 한계를 해결하기 위해 Win:G 는 뮤지션 대상의 직접적인 후원, 유튜브 스트리밍 서비스를 제공할 것입니다.

첫 번째는 뮤지션에게 직접적인 후원입니다. 단순히 뮤지션의 음원을 구매 및 스트리밍하는 것만으로 수익이 발생하지 않고 결제 오픈 API 아임포트를 사용하여 뮤지션의 계좌에 바로 송금이 가능한 서비스를 제공하고 있습니다. 추후 Toss, 카카오페이 등의 서비스를 추가하여 보다 편리한 후원 방식을 제공할 계획입니다.

두 번째는 Youtube Data API v3 를 사용한 유튜브 스트리밍 서비스입니다. 예상 서비스 이용자가 10~20 대 인만큼 가장 이용률이 높고 접근성이 높은 플랫폼인 유튜브를 선택했습니다. 뮤지션이 버스킹하는 모습을 실시간 스트리밍으로 Win:G 사이트에서 시청할 수 있으며, 과거 뮤지션의 버스킹 영상, 뮤직비디오를

플레이 리스트에서 바로 시청할 수 있습니다. 라이브 버스킹 영상을 유튜브 스트리밍 서비스로 제공하기 때문에 인디 아티스트의 유튜브 홍보로 자연스럽게 이어지게 됩니다.

Win:G 서비스의 최종적인 목표는 '오픈 플랫폼' 제공입니다. 인디 뮤지션들의 접근이 용이한 홍대, 신촌 등에 합주를 위한 공간을 제작하여 뮤지션들에게 공연 장소로 대여할 것입니다. 인디 뮤지션들에게 각자 필요한 장비를 빌려주고, 라이브 버스킹 영상을 송출하는 촬영 서비스도 도입할 것입니다. 또한, 상/하반기에 1 회씩 경연 프로그램을 개최하여 뮤지션들에게 공연 기회를 제공할 수 있도록 할 것입니다.

3. 프로젝트 개발 결과

3.1 개발방법론 및 도구

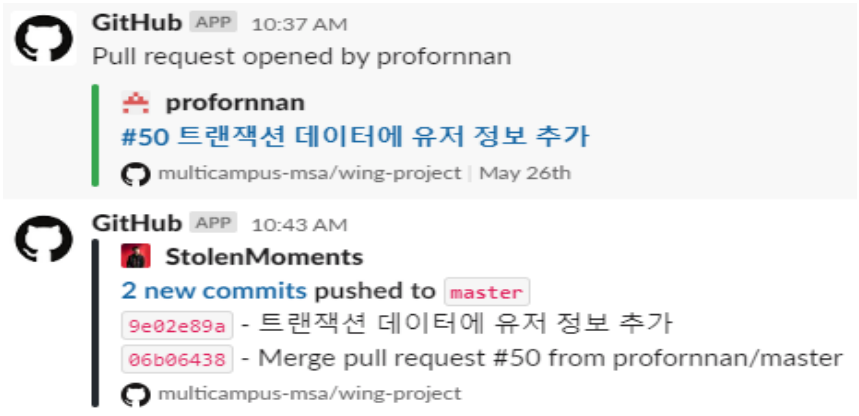
3.1.1 애자일 방법론 채택 이유 및 적용

프로젝트의 생명주기 동안 반복적인 개발을 촉진하는 애자일 방법론을 채택해서 이슈에 대해 유연하게 대처하고 요구사항을 충분히 반영할 수 있도록 하였습니다.

스프린트를 나눠서 계획적으로 개발을 할 수 있게 하였고 데일리 스크럼을 통해 어제 있었던 이슈, 오늘 해야 할 일, 의문점을 서로 교환하여 진행 상황에 대한 인지와 맡은 일에 대한 책임감을 상기시켰습니다. 지속적인 테스트와 오류 수정으로 동작하는 프로그램을 목표로 하고 이후에 완성도를 높여갔습니다.

3.1.2 협업에 사용된 도구

메시지 채널은 Slack 이용하였습니다. 문제나 궁금증에 관해 묻거나 기술 적용에 있어서 참고하면 좋은 팁 등을 공유하는 용도로 사용하였습니다. Confluence, Jira, GitHub 를 연동하여 프로젝트 진행에 관한 모든 상황을 실시간으로 반응할 수 있게 하였습니다.



Confluence 를 통해 개발과정에서 나오는 이슈 해결법, 기술 공유, 회의록 등을 문서화하여, 정리하고 기록하는 팀 문화가 정착되도록 했습니다.

Win:G Project

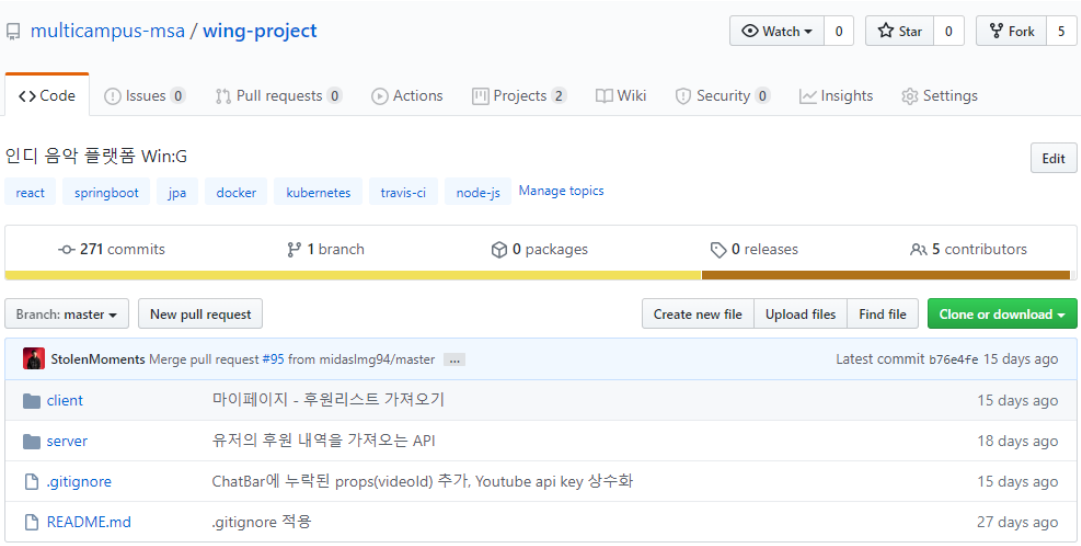
인디 음악 홍보 플랫폼

Core team

	역할(정)	역할(부)	담당 업무
강인호(멘토)	멘토	멘토	멘토
이종화(팀장)	Full-Stack	Full-Stack	Music Service, CI/CD
이대희	Front-End	Back-End	Concert Service
이민기	Back-End	Front-End	User Service
탁성건	Back-End	Front-End	Support Service, Live Service

이슈 트레킹에는 Jira 를 사용했습니다. 전체적인 프로젝트 이슈 관리, 스프린트 별 각자의 역할과 기능 명시 등을 이용했습니다. 프로젝트의 특성 중 하나인 점진적 구체화로 인해 초기의 계획대로 완성되는 프로젝트는 거의 없으므로 프로젝트 중 변하는 상황에 대응하기 위해 Jira 를 사용한 것이 많은 도움이 되었습니다.

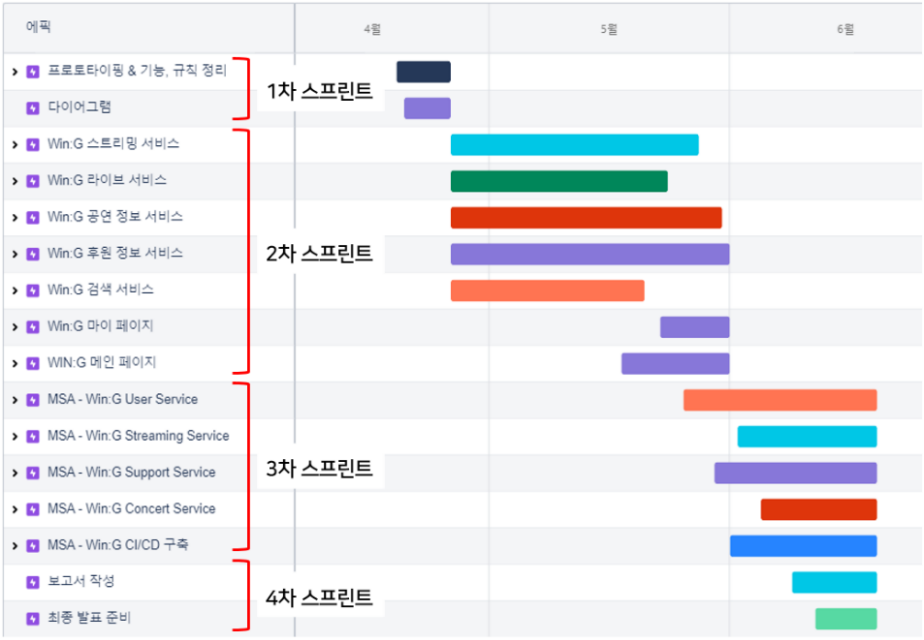
형상 관리 도구로는 GitHub 를 사용하였습니다. 프로젝트 중 업무를 나눠 작업하고 합칠 때 GitHub 를 통해 pull request, 코드 확인, 승인의 흐름을 따라 충돌을 최소화하였습니다.



프로젝트 진행에 대해서 멘토님께 조언을 받거나 진행 상황을 체크할 때 Zoom 을 이용해 화상회의로 진행했습니다.

3.2 Jira 를 통해 본 스프린트

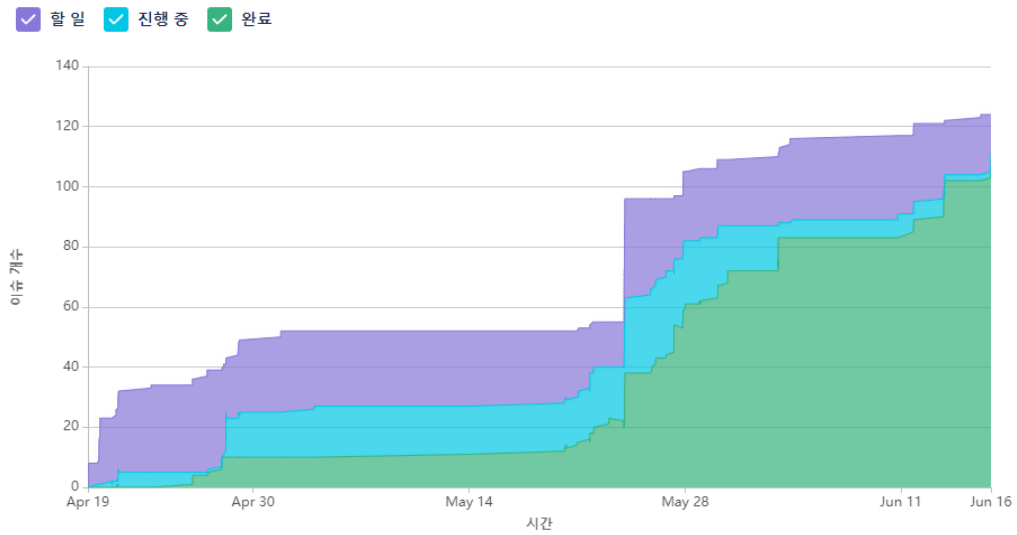
3.2.1 로드맵



개발 기간 동안 Jira 를 활용하여 총 4 개의 스프린트를 진행하였습니다.

- 1 차 스프린트 : 개발을 위한 준비 작업 및 UI Prototyping
- 2 차 스프린트 : 모놀리식 아키텍처 구현
- 3 차 스프린트 : 마이크로 서비스 아키텍처 구현 & CI/CD
- 4 차 스프린트 : 최종 발표 준비

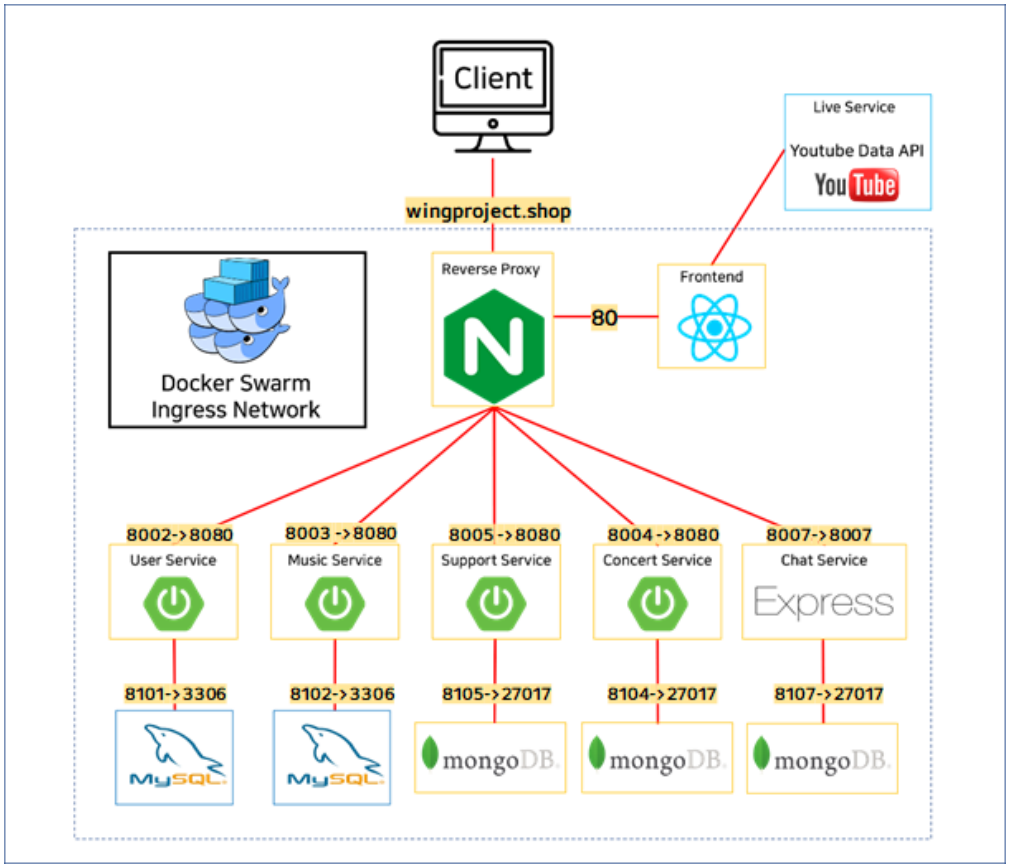
3.2.1 누적 플로우 다이어그램



5 월 20 일까지는 수업으로 발생한 이슈가 적었으며, 본격적으로 프로젝트가 시작된 후 이슈의 비율이 증가한 것을 확인할 수 있습니다. 발생한 이슈는 체계적인 관리를 통해 해결해 나갔습니다.

3.3 프로젝트 구조 및 파일

3.3.1 MSA 아키텍처



Win:G 서비스를 MSA 구조로 설계했습니다. 3 개의 GCP VM 인스턴스에 1 개의 Manager Node, 2 개의 Worker Node 를 구성하였고, 도커 스웜을 활용하여 컨테이너 오케스트레이션을 구성했습니다.

각 서비스는 독립적인 Database 를 가지고 있으며 User Service 의 DB 를 제외한 나머지 서비스의 DB 는 도커 컨테이너에 구성했습니다.

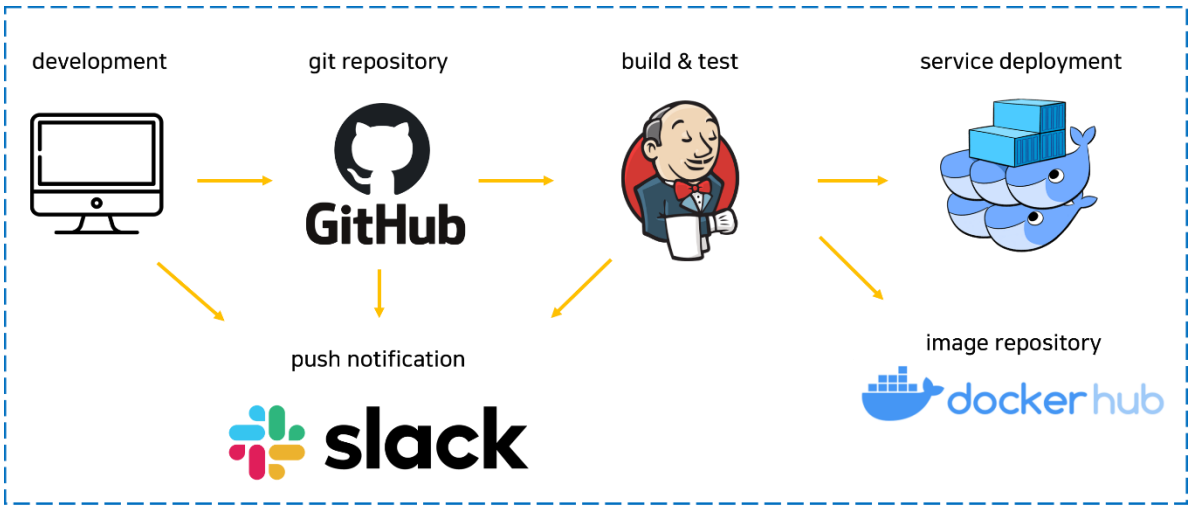
MSA 는 변경이 있는 서비스는 다른 서비스에 운영적인 측면에서 영향을 주지 않기 때문에 새로운 서비스를 추가시킬 때 전체를 다운시키는 등의 일을 할 필요가 없습니다. 또한 각 서비스들이 별도의 서비스로 구분되어 있어서 다른 서비스의 기술적 제약 없이 다양하게 사용할 수 있습니다.

서비스별로 개발하려는 언어가 다른 경우, 신기술을 적용해야 하는 경우에도 다른 서비스와 의존성을 고려하지 않아도 전체 서비스가 동작하는 데 아무 지장이 없습니다.

각 서비스는 RESTful 한 방식으로 통신하기 때문에 API 명세에 맞는 형식의 JSON 타입의 데이터를 전달해주기만 하면 원하는 결과를 얻을 수 있습니다.

이외에도 개별적인 Scale-Out 이 필요한 경우, 하나의 서비스 때문에 전체를 Scale-Out 하는 모놀리식과는 달리 MSA 에서는 부하가 걸리는 특정 서비스만 Scale-Out 하면서 유동적이고 효율적으로 운영할 수 있습니다.

3.3.2 DevOps Pipeline



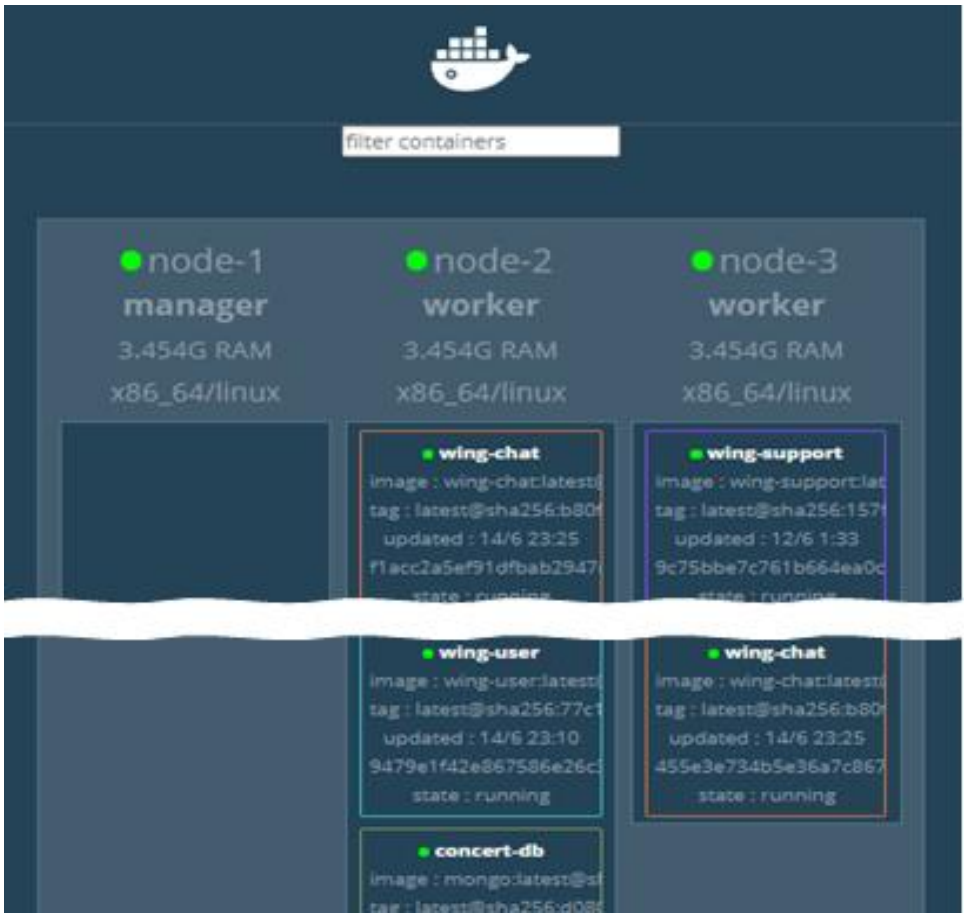
Jenkins 와 Docker Swarm 을 사용하여 CI(빌드 자동화)와 CD(배포 자동화)를 구축했습니다.

GitHub 에 push 를 하면 Jenkins 에서 test 와 build 과정이 실행됩니다. 실행 결과를 Jenkins Dashboard 에서 확인하는 것이 번거롭기 때문에 slack 으로부터 notification 을 받을 수 있도록 플러그인을 설치했습니다.

CI 프로세스를 통과한 코드는 각 서비스가 가지고 있는 Docker Hub 의 image 로 생성이 되고, 실행 중인 Docker Service 가 있다면 서비스에 수정사항이 release 될 수 있도록 구성했습니다.

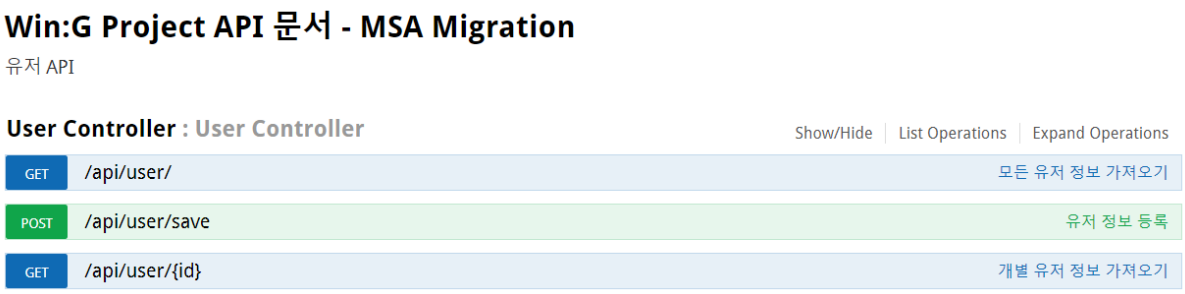
3.3.3 Docker Swarm 클러스터 - Visualizer

Visualizer 를 이용해서 스웜 클러스터에 컨테이너 그룹이 배치된 모습을 시각화하였습니다. 서비스 장애 발생 시에도 지속해서 서비스가 가능하도록 replicas 옵션을 통해 각 서비스마다 설정한 개수만큼 정상 동작하는 컨테이너들을 유지하도록 하였습니다.



3.3.4 RESTful API - Swagger

Swagger 를 이용해 개발과 동시에 인터페이스에 필요한 엔드포인트, 요청 및 응답 문서의 형식을 쉽게 확인, 참조할 수 있도록 했습니다.



Win:G Project Support API 문서

Support Controller : Support Controller

Show/Hide | List Operations | Expand Operations

GET	/api/support/	전체 후원 정보 조회
POST	/api/support/	후원 정보 저장
GET	/api/support/artist/{id}	후원 정보를 아티스트 ID로 조회
DELETE	/api/support/uid/{uid}	후원 정보 삭제
GET	/api/support/uid/{uid}	후원 정보를 UID로 조회
GET	/api/support/user/{id}	후원 정보를 유저 ID로 조회

Win:G Project API 문서 - MSA Migration

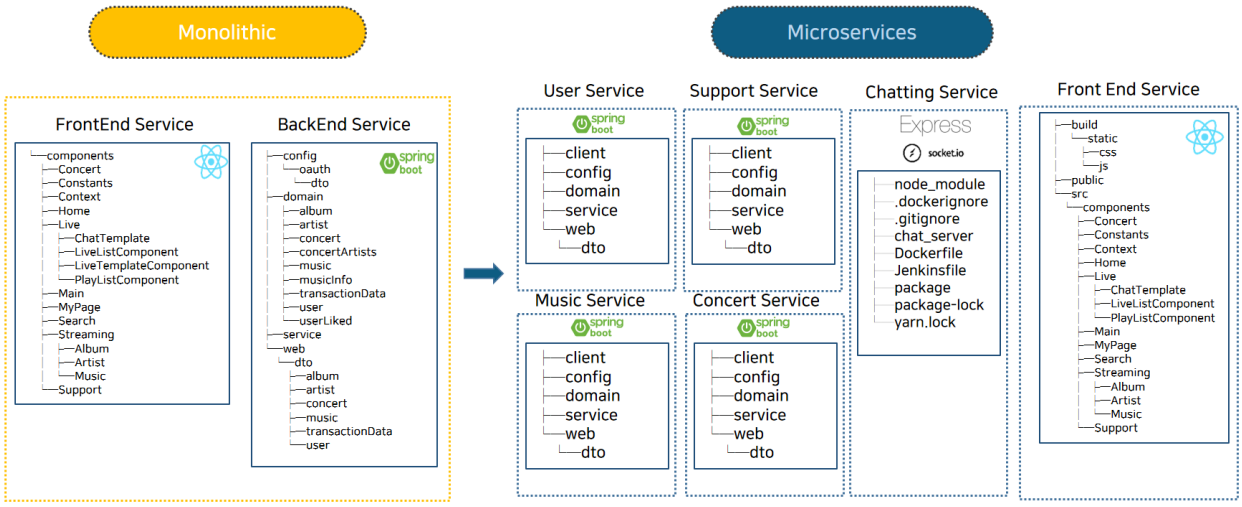
콘서트 API

Concert Controller : Concert Controller

Show/Hide | List Operations | Expand Operations

GET	/api/concert	콘서트 검색
POST	/api/concert	콘서트 저장
DELETE	/api/concert/{id}	콘서트 삭제
GET	/api/concert/{id}	콘서트 상세 검색
PUT	/api/concert/{id}	콘서트 수정

3.3.5 프로젝트 트리

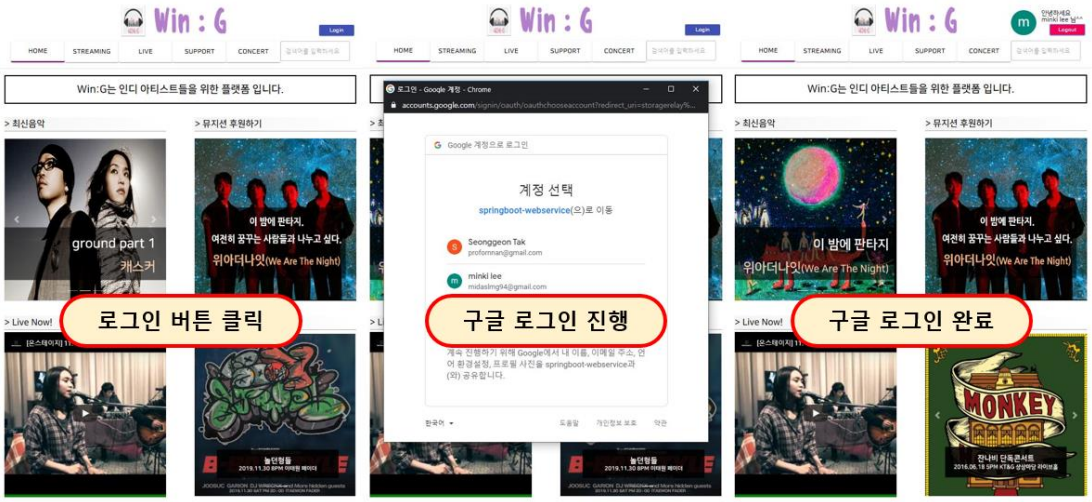


저희 팀은 기존 모놀리식과 MSA 의 차이를 경험해 보기 위해, 최초 모놀리식 형식으로 개발을 완료한 후 MSA 기반으로 변경하는 형식으로 프로젝트를 수행했습니다.

모놀리식에서는 프론트엔드, 백엔드 각각 하나의 프로젝트로 구성되었던 것이 MSA 에서는 5 개의 백엔드 서비스로 나눠진 것을 확인할 수 있습니다.

3.4 주요 기능

3.4.1 로그인 - Google Login API 활용



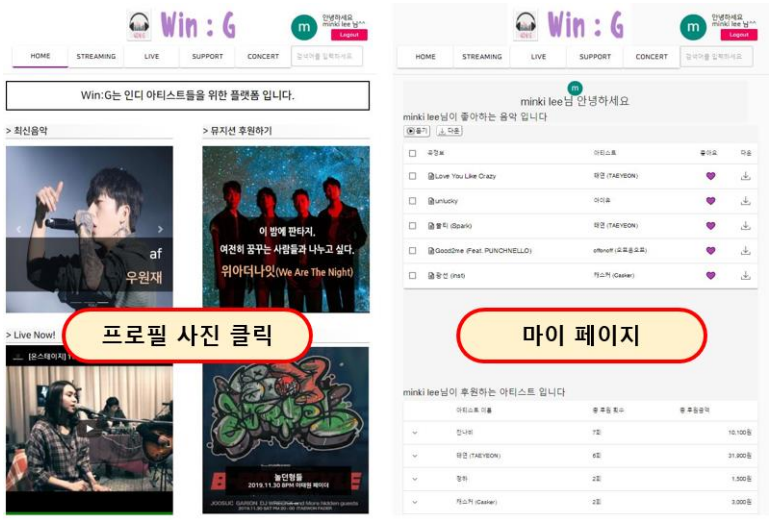
소셜 로그인을 제공하는 서비스들이 많아지면서 사이트 자체에서 회원가입을 하지 않고 구글 Login API 를 활용하여 구글 계정으로 로그인 할 수 있도록 구현하였습니다.

Google Cloud Platform 에서 제공하는 Google SDK 를 사용하여 로그인한 계정의 정보를 받아 왔고, OAuth2 인증을 거쳤습니다.

서버는 클라이언트에서 보낸 데이터의 비동기 처리(API 통신)를 위해 ajax 통신이 필요합니다. ajax 라이브러리인 axios 를 사용했습니다. 이렇게 받아온 유저 정보는 Google Cloud Platform 의 SQL 인스턴스에 저장하여 외부 서비스에서 호출하여 사용할 수 있도록 하였습니다.

로그인 한 유저가 다른 사이트로 이동하거나 새로 고침 했을 시 Win:G 서비스를 이용하면서 유지해야 하는 정보이지만, 사이트를 빠져나간 뒤에는 갖고 있지 않아도 될 정보이기 때문에 sessionStorage 를 사용해서 Win:G 서비스를 이용하는 동안 유저의 정보를 세션에 담아 재로그인 하는 일이 없도록 했습니다.

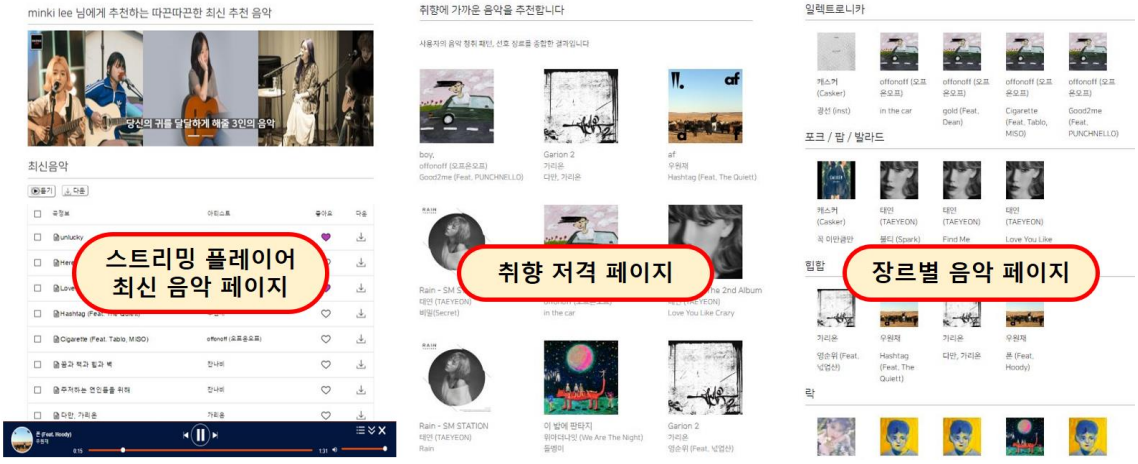
3.4.2 마이 페이지



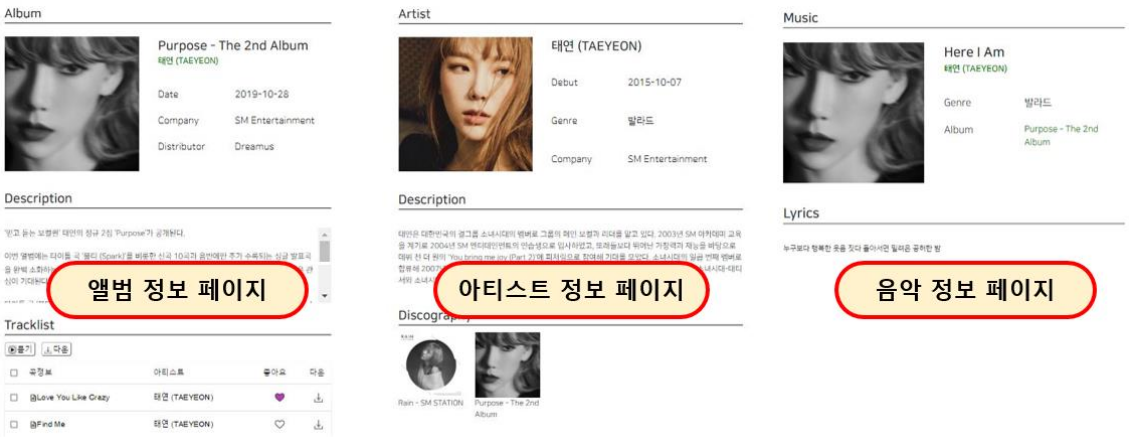
프로필 사진을 클릭하여 마이 페이지에 접속하면 좋아요 한 음악, 후원 아티스트 목록을 표시하였습니다. '좋아요'로 되어있는 음악을 취소하면 목록에서 사라지도록 하였습니다.

후원리스트를 보여주며 과거의 후원리스트까지 보여줄 수 있도록 하였습니다. 후원리스트는 가장 많은 후원을 한 뮤지션 순서로 보여줄 수 있도록 구성하였습니다.

3.4.3 스트리밍

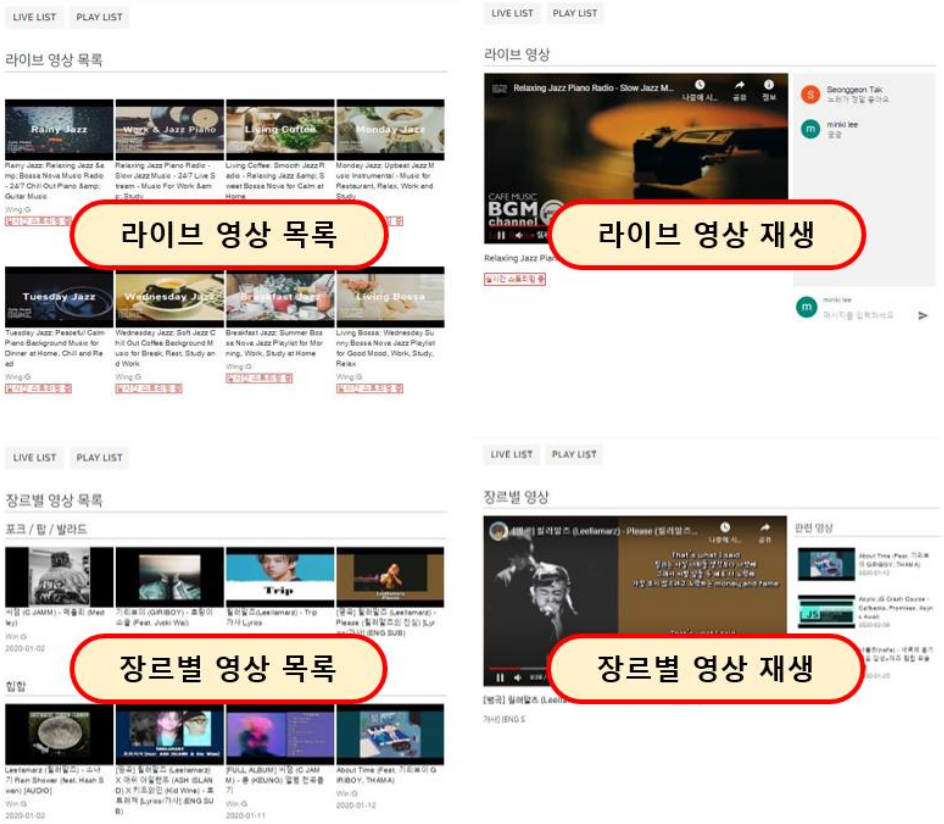


최신음악과 장르별로 음악 목록을 볼 수 있도록 페이지를 구현했습니다. 그리고 AWS S3 에 음원을 저장해두고 react-mp3-player 라는 node.js 모듈을 활용해 스트리밍 서비스를 구현했습니다. 플레이어는 화면 하단에 고정된 상태입니다.



아티스트, 앨범, 곡의 상세 정보를 제공하는 페이지입니다. 해당 아티스트, 앨범, 곡의 고유한 ID 를 가지고 REST API 에 요청해 정보를 가져온 후 페이지에 출력하는 방식으로 구현했습니다.

3.4.4 라이브 & 채팅

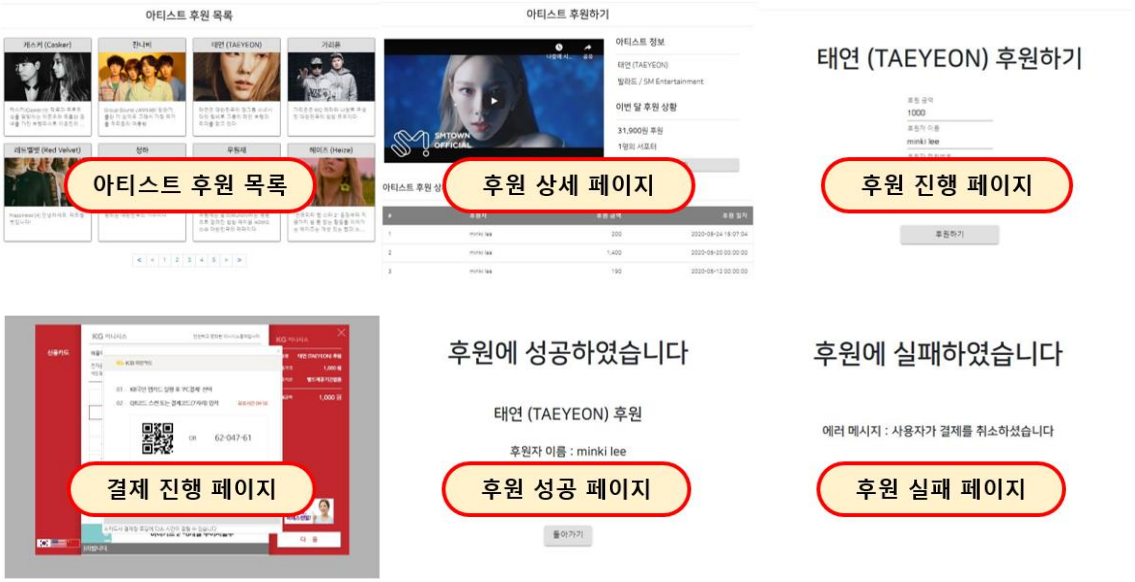


라이브 서비스에서는 Youtube Data API v3 를 이용하여 사용자들이 인디 뮤지션들의 공연을 어떤 장소에서든 접할 수 있도록 했습니다.

Youtube Data API v3 를 이용해 받아온 영상 정보를 라이브 영상 목록 페이지에 띄우고, 영상을 클릭할 경우 라이브 영상 재생 페이지로 이동합니다. 라이브 영상을 보면서 실시간으로 소통을 할 수 있도록 socket.io 를 이용해서 채팅 서비스를 구현했습니다.

장르별 영상 페이지에서는 장르별로 뮤지션의 지난 라이브 영상이나 뮤직비디오를 시청할 수 있도록 구성하였습니다.

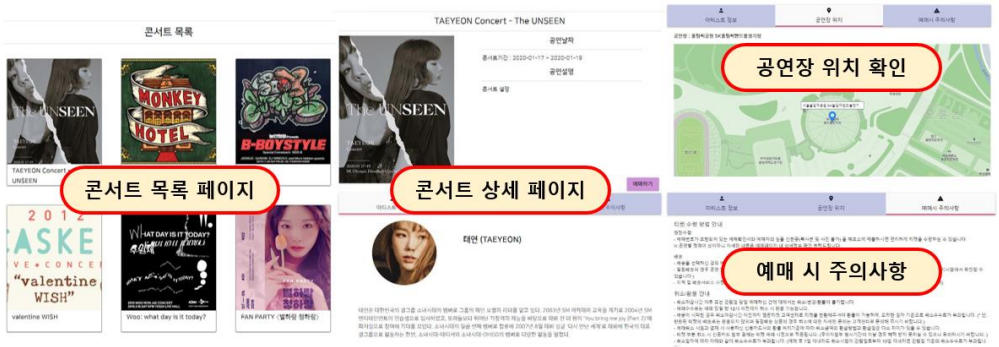
3.4.5 후원



후원 페이지에서 사용자가 좋아하는 아티스트를 후원할 수 있도록 아임포트 결제 연동 및 결제 API 를 적용했습니다.

아티스트 후원 목록에서 후원하고 싶은 아티스트를 클릭하면 후원 상세 페이지를 볼 수 있습니다. 상세 페이지에서는 홍보 영상과 후원 현황을 볼 수 있으며, 후원하기 버튼 클릭 시 후원 진행 페이지로 넘어갑니다. 후원자의 정보를 입력하고 후원하기 버튼을 클릭하면 아임포트 결제 연동을 통해 결제를 진행하게 되며, 후원 성공 또는 실패 페이지로 넘어갑니다. 후원에 성공한 경우 후원 정보를 DB 에 저장합니다.

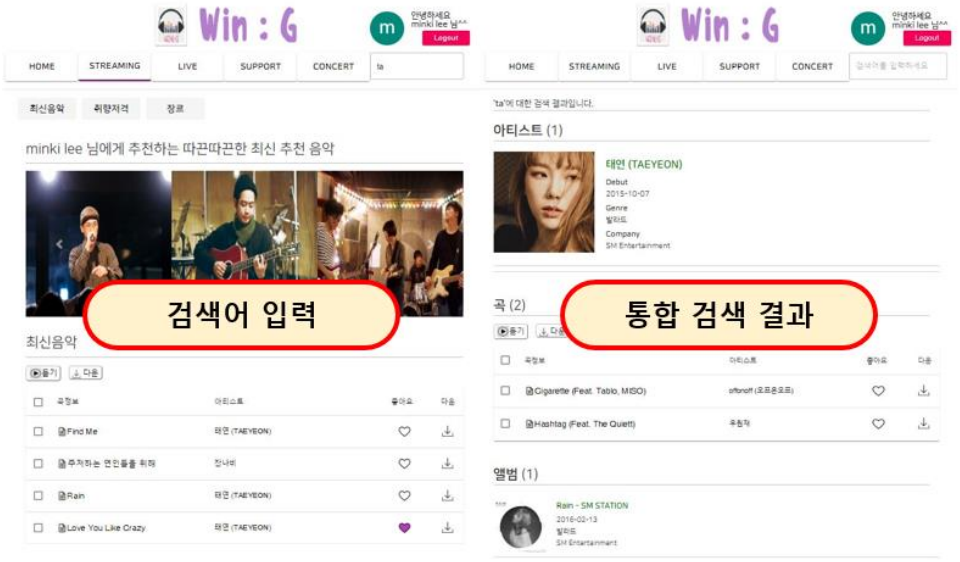
3.4.6 콘서트 정보



콘서트 상세 페이지에서 공연장 위치를 볼 수 있도록 카카오 지도 API 를 적용했습니다.

지도를 생성하는 API 를 불러오고 지도를 띄우는 컴포넌트를 작성했습니다. 서버를 통해 DB 에 저장되어있는 장소 값을 불러와서 API 의 함수로 키워드를 기반으로 검색해서 지도를 생성하는 코드를 작성했습니다. 마커를 생성하는 코드도 같이 작성해서 좀 더 보기 편하게 만들었습니다.

3.4.7 통합검색

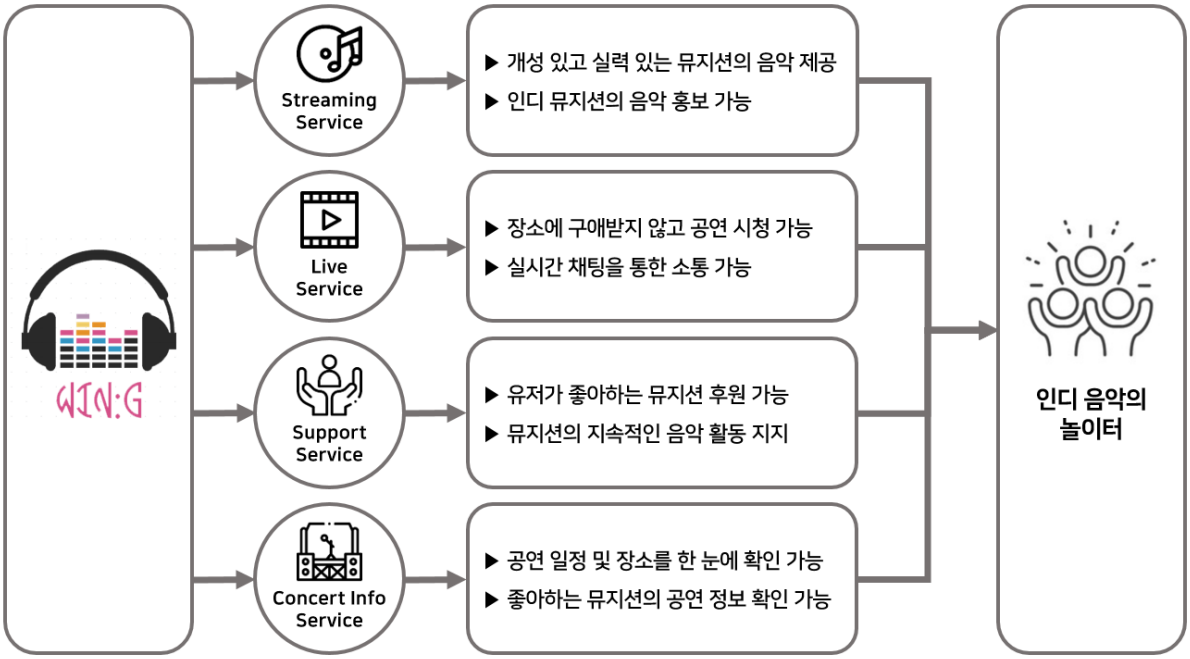


사용자들이 원하는 정보를 찾을 수 있도록 통합 검색 서비스를 제공합니다.

사용자가 검색창에 검색어를 입력하고 Enter 를 누르면 데이터베이스에서 아티스트, 음악, 앨범 정보를 이름 기반으로 검색해 결과 페이지에 출력하게 됩니다.

4. 기대 효과

4.1 기대 효과



Win:G 는 밴드와 사용자 간의 교류를 도와 뮤지션에게는 안정적인 수입을 얻고 홍보할 수 있는 플랫폼을 제공하고, 사용자에게는 취향에 따른 음악을 듣고 후원할 수 있는 소통의 장을 제공할 것입니다.

첫 번째, 음원 스트리밍 서비스를 통해 새로운 인디 음악을 쉽게 접할 수 있는 장소를 제공하겠습니다. 유저 입장에서는 편향된 음악 시장에서 벗어나 개성 있고 실력 있는 인디 뮤지션들의 음악을 쉽게 접할 수 있습니다. 뮤지션들의 입장에서는 본인들의 음악을 홍보할 수 있어, Win:G 가 새로운 소통의 장으로서의 역할을 수행할 것으로 기대합니다.

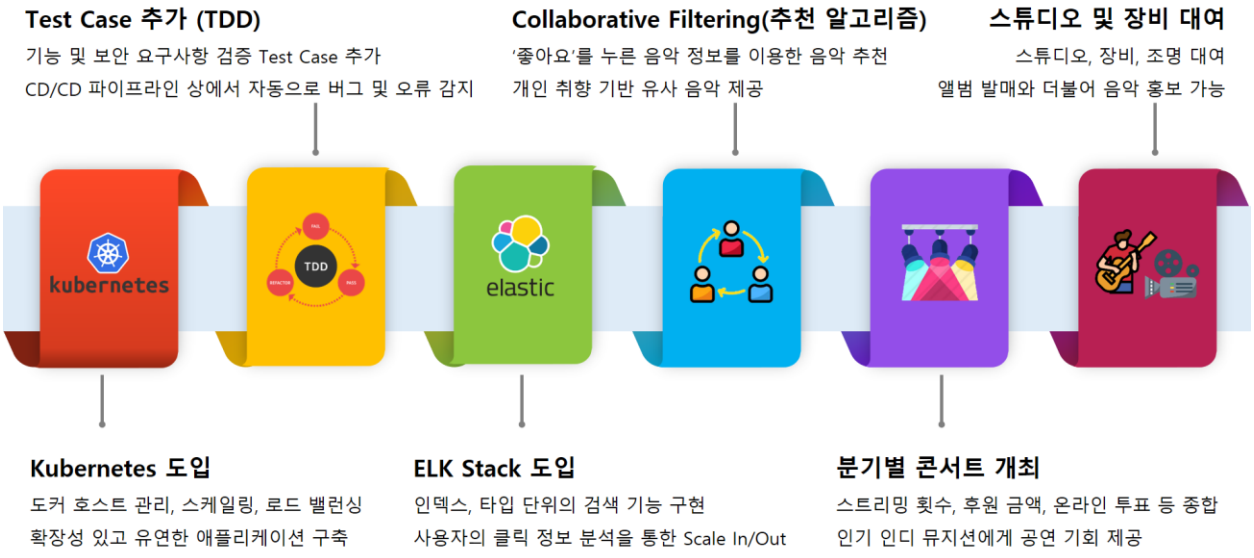
두 번째, 실시간 Live 서비스를 통해 사용자와 뮤지션 간의 소통을 돕겠습니다. Win:G 의 유저들은 공연 장소에 직접 가지 않아도 자신들이 좋아하는 뮤지션들의 공연을 어떤 장소에서든 접할 수 있습니다. 덕분에 뮤지션들은 장소에 구애받지 않고 사용자들에게 본인의 음악을 들려줄 수 있으며, 실시간 채팅을 통해 팬들과 소통할 수 있습니다.

세 번째, 인디 뮤지션 후원 서비스를 통해 지속적인 음악 활동을 지원합니다. 유저 입장에서는 본인이 좋아하는 뮤지션을 후원해 뮤지션이 지속해서 활동할 수 있도록 응원할 수 있습니다. 뮤지션은 후원금을 통해 안정적인 환경에서 음악 활동에 몰입할 수 있어 Win:G 가 뮤지션을 지지하는 플랫폼으로서의 역할을 수행할 수 있습니다.

네 번째, 공연 정보 서비스를 통해 인디밴드의 공연 정보를 한눈에 확인할 수 있습니다. 유명한 가수들의 공연 정보가 담긴 포스터와는 달리 라이브 클럽에서 공연하는 인디밴드들의 포스터에는 공연 장소, 인디밴드 이름, 공연 시간 등을 확인하기 어렵습니다. Win:G 를 통해 자신이 좋아하는 뮤지션의 공연 일정, 공연 장소를 확인하기 쉽게 공연정보 서비스를 제공하여 유저들의 편의성을 높였습니다.

4.2 향후 개선 사항

COVID-19 로 인해 프로젝트 기간이 2 개월에서 1 개월로 단축되어 전체 개발 범위를 2 단계로 나눴습니다. 1 단계에 적용할 기술 및 서비스들은 구현을 완료하였으며, 향후 2 단계에서 적용할 내용을 기술적, 서비스 관점에서 정리해 보았습니다.



4.2.1 기술적 개선 사항

Test Case 추가 (TDD)

버그 또는 오류를 CI/CD 파이프라인 상에서 자동으로 걸러낼 수 있도록 기능 및 보안 요구사항을 검증하는 테스트 케이스를 추가, 보완할 예정입니다. 현재는 기능 테스트 위주로만 테스트 케이스가 작성되어 있는데, 비기능 요구사항에 대해서도 테스트할 수 있도록 테스트 케이스를 보완할 것입니다.

Kubernetes 도입

쿠버네티스를 이용해 컨테이너를 운영함으로써, 애플리케이션 배포 외에도 다양한 운영 관리 업무를 자동화할 예정입니다. 현재 Win:G 프로젝트는 여러 도커 호스트를 도커 스웜을 이용해 클러스터로 묶어서 관리하고 있습니다. 추후 쿠버네티스를 도입해 도커 호스트 관리, 컨테이너 배치, 스케일링, 로드 밸런싱, 헬스 체크 등의 기능을 통해 더욱 확장성 있고 유연한 애플리케이션을 구축할 것입니다.

ELK Stack 도입

ELK Stack 을 도입하여 검색 기능을 구현하고 사용자의 사이트 이용 패턴을 분석할 수 있는 기능을 추가할 예정입니다. 현재의 검색 기능에서는 데이터베이스에서 이름 기반의 검색만 가능하지만, ELK Stack 을 적용한다면 인덱스, 타입 단위로 사용자가 원하는 정보에 가까운 검색 결과를 도출할 수 있습니다. 또한, 사용자의 사이트 클릭 정보를 분석하여 이용자가 몰리는 시간대에 해당 서비스 Scale In/Out 을 통해 유연하고 효율적인 서버 관리를 가능하게 해줍니다.

음악 추천 서비스 도입 - Collaborative Filtering(추천 알고리즘)

Collaborative Filtering 을 통해 사용자가 '좋아요'를 누른 음악 정보를 이용하여 다른 음악을 추천할 수 있습니다. 이 기술을 통해 개인 취향을 기반으로 한 유사 음악을 제공하면서 사용자들에게 더 많은 음악을 노출하게 되고, 사이트에 대한 사용 경험을 높일 수 있습니다. 따라서 고객만족도를 높이면서도 이윤 창출 및 증대에 이바지할 것입니다.

4.2.2 서비스적 개선 사항

로그인 서비스

구글 외에 다른 소셜 로그인 구현과 자체 회원가입 도입을 추진할 것입니다. 리스너 계정 이외에도 아티스트 계정 그룹을 추가하여, 아티스트가 본인의 음악, 공연을 셀프 등록할 수 있도록 서비스할 예정입니다. 후원 서비스를 사용할 때도 후원자 목록 확인이 가능하도록 연계할 것입니다.

스트리밍 서비스

스트리밍 서비스 부문에서는 각 음원을 블록체인과 분산저장시스템을 통해 저장함으로써, 음원의 저작권 보호 및 음악 파일에 대한 보안을 강화할 것입니다. 또한, 아티스트가 자신의 음악에 대한 권리를 보장받을 수 있도록 블록체인 기술을 이용해 사용자의 음악 이용 내역을 확인할 것입니다. 음원이 어떻게 사용되고 몇 번 재생되었는지 투명하게 알 수 있으므로 아티스트들의 정확한 수익을 산정할 수 있게 됩니다.

후원 서비스

후원 서비스의 경우, Win:G 계좌를 통해 아티스트에 지원될 후원금을 관리하고 아티스트 후원 시 앨범이나 굿즈를 홍보 및 마케팅의 일환으로 제공할 것입니다. 이는 서비스 수익체인 면에서 아티스트의 만족도, 사용자의 충성도를 고취시켜 장기적인 후원을 도울 수 있고, 궁극적으로 Win:G 의 성장에 힘을 실어주게 될 것입니다.

콘서트 서비스

크롤링을 활용해 예매사이트에 있는 정보를 가져와서 뮤지션들의 콘서트 정보를 Win:G 사이트에서 보여줄 예정입니다. 그리고 현재는 공연 예매 사이트 링크를 제공하지만, 라이브 카페와 협의를 통해 Win:G 사이트 내에셔 뮤지션들의 공연 정보를 확인하고 예매까지 가능하도록 서비스를 확장할 것입니다. 이를 통해 사용자들에게 편의성을 제공하고, 이윤을 창출할 수 있습니다.

분기별 콘서트 개최

스트리밍 횟수, 후원 금액, 온라인 투표 등을 종합해 분기별로 인기 인디 뮤지션에게 공연 기회를 제공할 것입니다. 이를 통해 인디 뮤지션들의 플랫폼 유입을 늘리고, 대중들에게 자신을 알릴 기회를 제공할 수 있을 것입니다.

스튜디오 및 장비 대여 서비스

열악한 음악 환경에 있는 인디 뮤지션들에게 스튜디오, 장비, 조명을 대여해 주는 서비스를 제공할 예정입니다. 앨범 발매와 더불어 자신들의 음악을 홍보할 수도 있으며, 해당 서비스 자체가 하나의 콘텐츠가 되어 Win:G 유튜브에 영상이 업로드되면 뮤지션 홍보 및 이윤 창출이 가능합니다.

오늘의 인디 뮤지션 서비스

Win:G 측에서 온/오프라인으로 인디 뮤지션을 찾아가 인터뷰를 진행합니다. 인터뷰이인 인디 밴드가 추구하는 음악적 성향 및 가치관을 대중에게 알리고 그들의 음악을 소개할 수 있는 콘텐츠를 만들 예정입니다.

5. 개발 후기

성명	후기
이종화	6 개월간 클라우드부터 웹 개발, MSA 까지 아주 많은 것을 배웠습니다. 하지만 배운 내용을 깊이 이해하고 능숙하게 다룰 수 있는 경지에 오르지는 못했습니다. 변명이지만,

	마치 밥은 많이 먹었으나 소화할 시간이 부족했고, 운동하며 근육을 만드는 시간이 부족했다고 비유할 수 있습니다. 과정이 끝났으니 배운 내용을 바탕으로 개인 프로젝트를 진행할 계획입니다. 교육을 받으며 배운 내용을 프로젝트에 적용하기 급급했던 것들을 곱씹으며 확실하게 제 것으로 만들기 위해서입니다. 그리고 TDD 라든지, 클린 코드 라든지, 쿠버네티스 라든지 미처 시도하지 못했던 것들을 해볼 생각입니다. 본 과정에서 프로젝트를 하며 프론트엔드, 백엔드, 인프라까지 모두 경험했기 때문에 자신감이 넘치네요. 개발자로서 성장하기 위한 아주 유의미한 시간이었습니다. 3 조 우리 팀원들이 빡빡한 시간 동안 쉽지 않은 프로젝트를 완성하느라 고생 많았고 잘 해줘서 고맙습니다. 강사님, 매니저님 교육과정 동안 잘 배울 수 있도록 해주셔서 감사합니다. 마지막으로 프로젝트 진행에 대한 조언, 그리고 개발자로서 성장할 수 있도록 양질의 조언을 해주신 멘토님 감사합니다. 저는 이만 ㅎㅎ!
이대희	처음 이 과정을 들어오면서 열심히 해야겠다는 마음과는 달리 프론트엔드부터 백엔드, 그리고 인프라까지 개발의 모든 부분을 배우면서 내가 프로젝트를 잘 수행할 수 있을까 걱정이 있었습니다. 아이디어 기획, 요구사항 분석 및 명세서 작성, UI Prototype 제작, 프론트엔드 & 백엔드 개발, 배포까지 프로젝트가 진행되면서 수업 시간에 이해한 것도 실제로 개발해보니 부족하다는 것을 느꼈고 팀원들과 다양한 상황에서 기술에 관해 얘기하다 보니 같이 성장하는 느낌을 받았습니다. 여러 가지 협업 툴을 사용했는데 처음엔 각각의 도구들이 어떤 역할을 하고 어떤 편의성을 주는지 정확히 인지하지 못했습니다. 데일리 스크럼도 처음엔 어떤 것을 말해야 하나라는 부담이 있었습니다. 시간이 지나 개발이 어느 정도 진행된 후에는 협업 도구들이 상황마다 유용 적이고 변화에 대응하는 데 효율적이라고 느꼈습니다. 또, 데일리 스크럼에서는 하나의 팀이라는 의식과 즉각적인 피드백, 신뢰 형성을 느꼈습니다. 비록, 기간이 짧아 계획했던 서비스나 기술적으로 도입하지 못한 것들이 있어 아쉬움이 남지만, 좋은 개발 문화란 이런 것이라고 느끼게 해준 팀원들에게 우선 고맙고 항상 도움 주신 강사님과 매니저님 그리고 방향을 잃지 않게 도와주신 멘토님에게도 감사의 말씀을 전합니다.
이민기	대학교에서 배운 전공 지식이 실무에 어떻게 쓰이는지 항상 궁금했는데 인프라부터 프론트, 백엔드까지 학습하며 프로젝트를 통해 이것들을 다뤄볼 수 있어서 좋았습니다.

	다른 html, css, jsp 정도만 가르쳐주는 다른 국비 교육들과 달리 React, SpringBoot, JPA, Docker, AWS 와 같은 기술들을 배우고 DB 설계부터 배포까지 모놀리식에서 마이크로서비스 마이그레이션을 해보는 프로젝트에 적용하는 좋은 경험을 할 수 있었습니다. 좋은 팀원들과 멘토님을 만난 것이 가장 큰 행운이었습니다. 팀원들 모두가 열심히 성실하게 참여한 덕분에 크게 다툰 적 없이 정해진 방향에 맞게 서로의 이견을 조율해가며 프로젝트를 완성할 수 있었습니다. 바쁘신 와중에도 화상 회의로 프로젝트 진행 방향에 대해 조언을 주신 멘토님께 감사합니다. 6 개월 동안 배운 교육내용과 이를 프로젝트에 적용해보며 많은 것을 배울 수 있었습니다. 교육이 끝난 뒤에도 프로그래밍 학습을 게을리하지 않고, 좋은 개발자가 되기 위해 노력하겠습니다.
탁성건	팀원들과 협업하여 요구사항 분석, UI 프로토타이핑부터 코드 개발, 테스트 및 배포까지의 단계를 진행하면서 소프트웨어 개발에 대해 많은 것을 배울 수 있었습니다. 무엇보다 프로젝트를 모놀리식 아키텍처에서 마이크로서비스 아키텍처로 전환하는 과정에서 MSA 의 장점인 높은 확장성과 손쉬운 배포를 직접 경험할 수 있었으며, 개별적인 서비스들이 모여서 하나의 서비스처럼 동작하는 것이 인상 깊었습니다. 팀원 각자가 관심 있는 부분을 조금 더 깊이 공부하고 알려주는 기술 공유 시간을 가졌고, 애자일 방법론을 채택해 더욱 체계적으로 개발할 수 있었던 것 같습니다. 스프린트를 나눠서 개발하고, 데일리 스크럼을 통해 진행 상황을 인지하는 등 기술적인 부분뿐만 아니라 협업 문화도 경험해 볼 수 있어서 좋았습니다. 프로젝트 기간이 짧아 쿠버네티스, ELK Stack 등 도입하지 못한 부분들이 있어 아쉬움이 남지만 여기서 배운 내용을 토대로 새로운 기술을 탐구하고 도전해 나갈 것입니다. 6 개월간의 교육 동안 많은 도움을 주신 멘토님, 강사님, 매니저님 그리고 같은 반 동료분들에게 감사하다는 말을 전하고 싶습니다.