Wrap-Up Report, RecSys09

좋아요구독추천AI Team

Team 소개

👍 **좋**아요**댓**글**구**독**알**림~

> BoostCamp AI Tech 4기 추천 시스템 9조, 좋아요… 팀!
>
> **T4056_김찬호**
> **T4096_배성수**
> **T4171_이지훈**
> **T4196_정소빈**
> **T4210_조원삼**

우리팀의 목표

“가치있는 서비스를 기획, 개발하자!”

- #기획
- #협업
- #성장
- #소통
- #문제 해결
- #추천 마스터

목차

1-1. 프로젝트 개요

▼ 프로젝트 주제

🎮 **인디게임 추천**

유저에게 맞춤형 인디게임을 추천해 주는 서비스 개발

▼ 협업

- Github
- Slack
- Gather.town
- Zoom

▼ 활용 장비 및 재료

개발환경

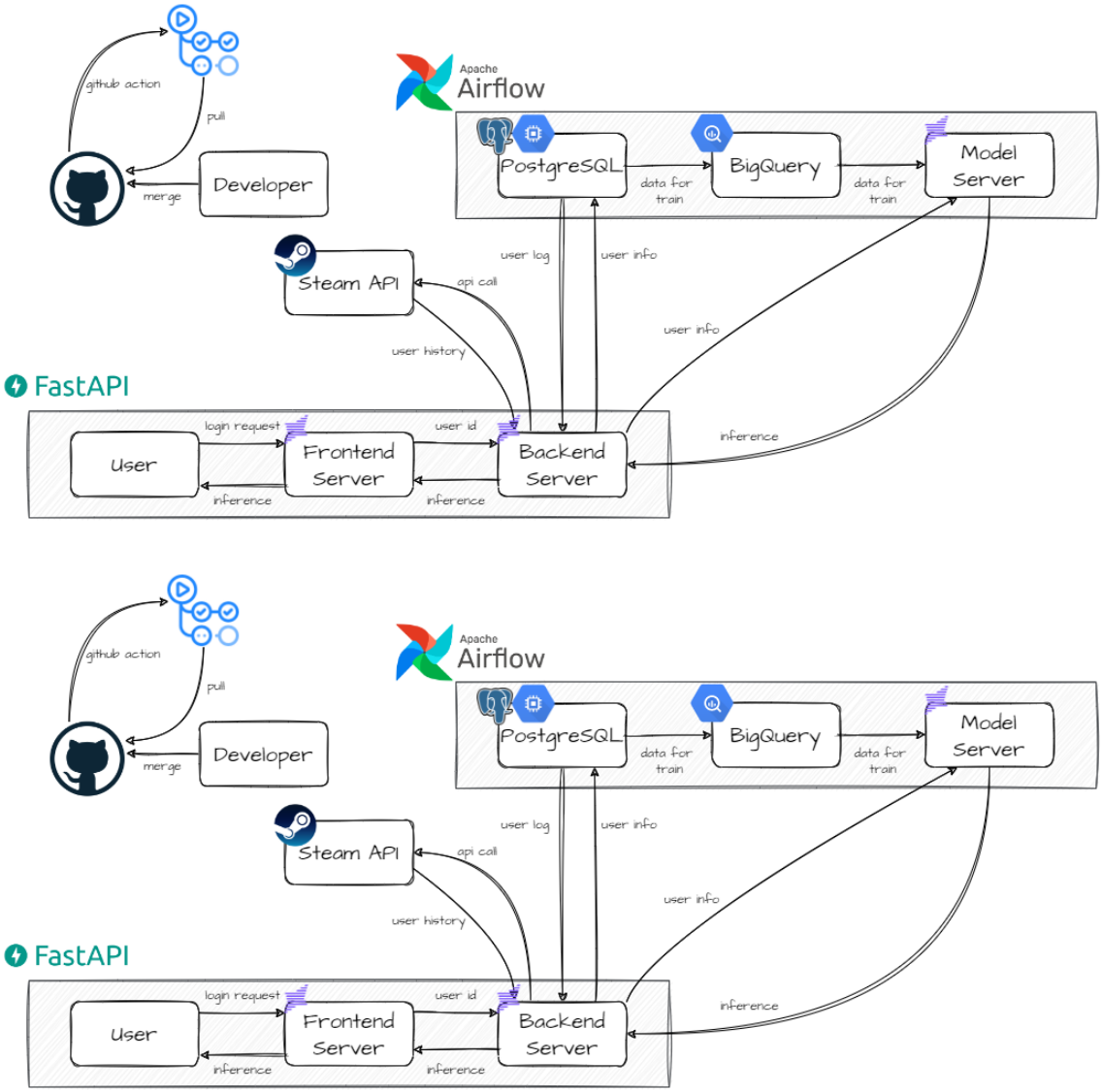
AI Stage Server

- OS: Ubuntu 18.04.5 LTS
- GPU: Tesla V100-SXM2-32GB

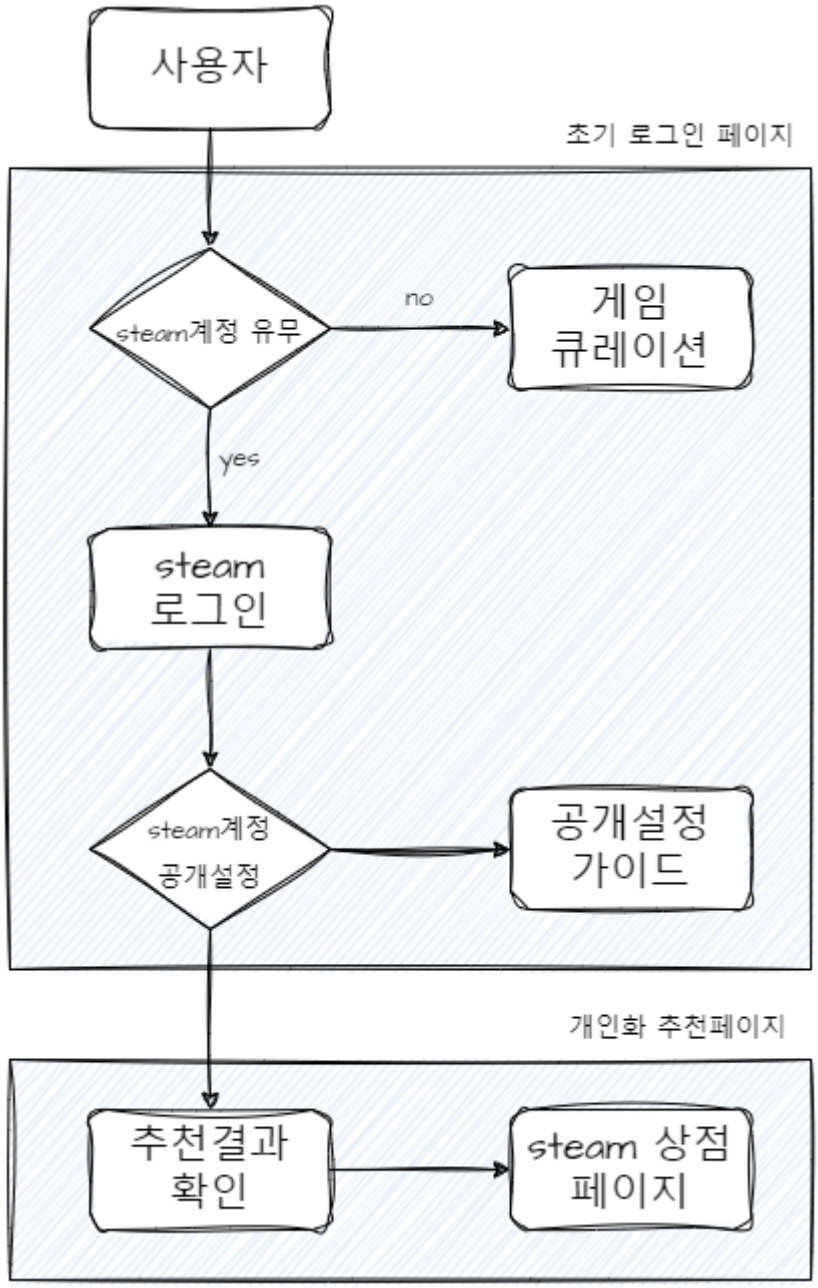
Tools

- Python
- Pytorch
- Github Action
- Airflow
- GCP
- Javascript
- Css
- HTML
- PostgreSQL

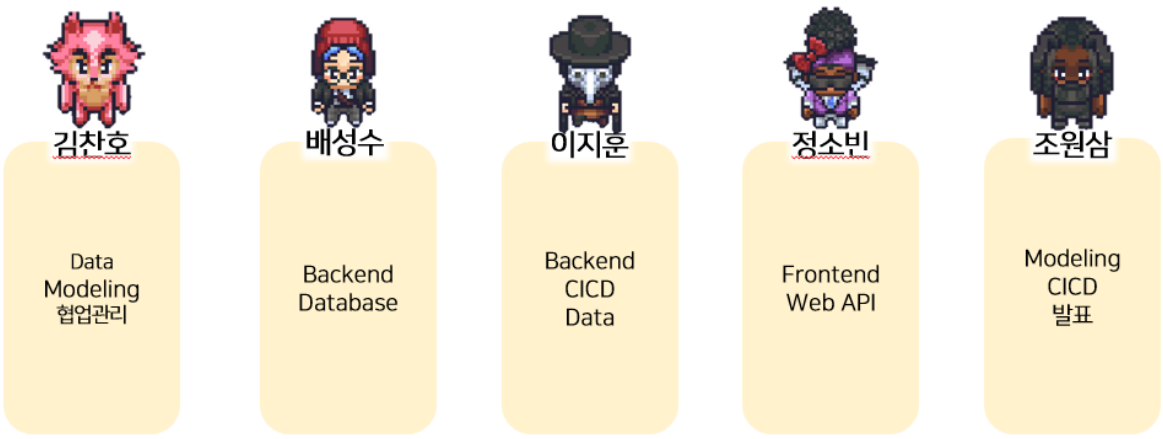
▼ 프로젝트 구조도


github action
pull
merge
Developer
Apache
Airflow
PostgreSQL
data for train
BigQuery
data for train
Model Server
Steam API
api call
user log
user info
user history
user info
FastAPI
User
login request
Frontend Server
user id
Backend Server
inference
inference
inference
github action
pull
merge
Developer
Apache
Airflow
PostgreSQL
data for train
BigQuery
data for train
Model Server
Steam API
api call
user log
user info
user history
user info
FastAPI
User
login request
Frontend Server
user id
Backend Server
inference
inference
inference

▼ 사용자 흐름도



1-2. 프로젝트 팀 구성 및 역할



- **T4056_김찬호:**
  - 노션, 게더타운 등 협업 관리
  - Recbole을 활용하여 다양한 모델 실험 및 비교
  - EASE 모델 유저 프리로 인퍼런스 구현
  - Curation 기능 구현
- **T4096_배성수:**
  - Backend API서버 설계 및 구현
  - Database 설계 및 구현
- **T4171_이지훈:**
  - Inference API서버 설계 및 구현
  - CICD & Slack 알람 기능 구현
  - 데이터 EDA 및 정합성 검증
- **T4196_정소빈:**
  -
- **T4210_조원삼:**
  - NCF/EASE 모델 설계 및 구축
  - BigQuery 구축
  - Airflow 설계 및 구축
  - 발표

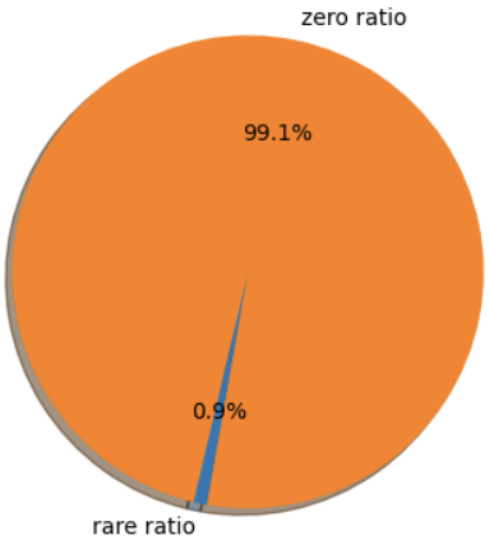
1-3. 프로젝트 수행 절차 및 방법

1-3-1. 데이터 및 DB

데이터

데이터 분류	데이터 설명	활용	크기
유저-아이템 상호작용 데이터	Steam Video Game and Bundle Data, USCD에서 수집한 데이터 중 유저-아이템 상호작용 데이터	추천시스템 모델 학습	5153209*5
게임 아이템 메타 데이터	2022 Steam Games Kaggle에서 수집한 게임 메타 데이터	아이템 필터링, 큐레이션,콘텐츠 기반 추천시스템 모델	55691*23
유저 데이터	유저 ID를 받아 Steam API를 이용해 직접 받아오는 유저-아이템 상호작용 데이터	개인화 추천시스템 모델 Input	-

EDA



- **User-Item 데이터 셋의 희소행렬 비율 : 약 0.9%**
  - 각 유저별 아이템의 상호 작용이 21개 이상인 유저들을 대상으로 분석을 진행했기에 **콜드 스타트** 유저 문제를 방지할 수 있었음
  - **필터링 후 총 유저 수** : 47825, **총 아이템 수** : 10978

- 위의 시각화 자료를 확인하면 희소행렬의 비율 역시 약 0.9%로 **추천 시스템 모델링에 적용**하기에 적합한 데이터 셋이라고 할 수 있음

DB

1-3-2. 모델

두 가지 고려사항

- **성능과 Inference 속도의 Trade-off**: 좋은 성능을 갖고 있더라도 실 서비스 예측이 느리면 문제
- **User Free & Item Cold-start 문제**: 실시간으로 추가되는 아이템들과 보유 데이터의 차이로 인한 cold-start문제, 개별 유저의 상호작용 데이터를 사용하기 위한 user free 모델 문제

Embarrassingly Shallow Autoencoders for Sparse Data : EASE

$$\min_{B} \quad ||X - XB||_F^2 + \lambda \cdot ||B||_F^2$$ → Closed form solution의 목적함수

Input: data Gram-matrix $G := X^\top X \in \mathbb{R}^{|\mathcal{I}| \times |\mathcal{I}|}$ → U*I의 입력데이터를 I*I로 변환해 사용

데이터 로드 포함, 학습과 예측과정 포함 12-13초 이내

Precision@10 기준, 전체데이터 0.61 / 정제데이터 0.35의 성능

1-3-3. 백엔드

FastAPI를 통해 구현, 크게 **API Server**와 **Inference Server** 분류

API Server

- 프론트엔드, 인퍼런스 서버 그리고 데이터베이스를 연결해주는 역할로 프론트엔드에 필요한 API를 제공하고 인퍼런스 결과를 데이터베이스에 저장해주는 서버를 구축

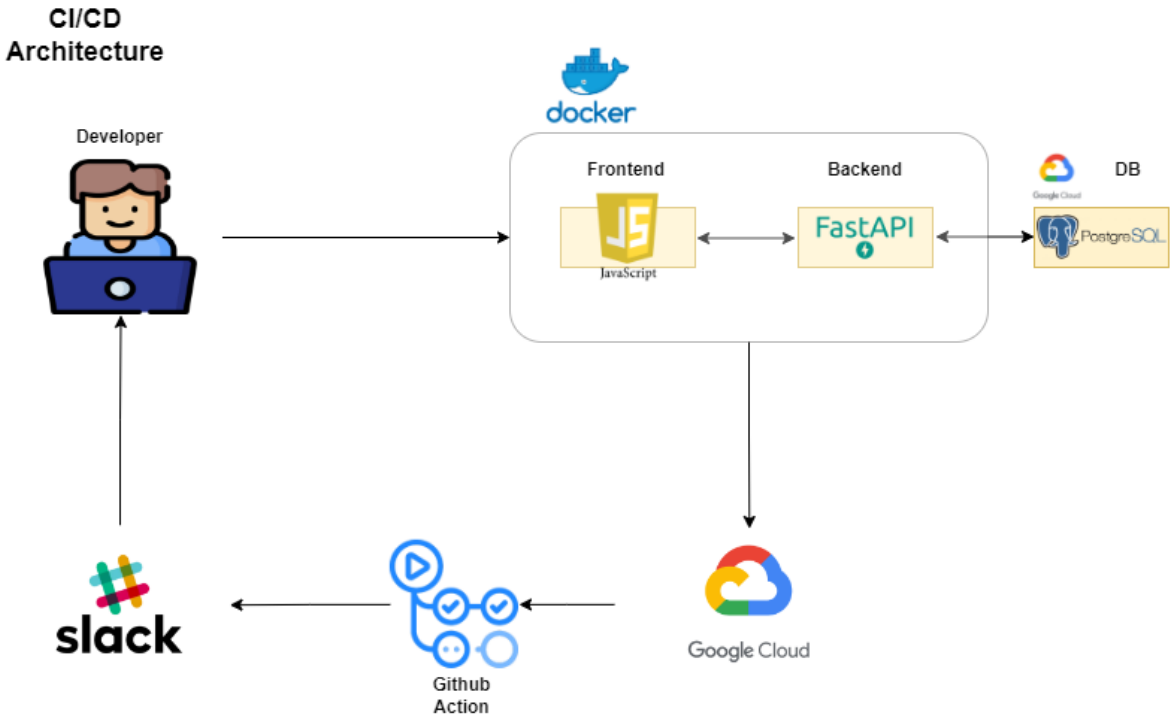
```
api
|-- assets # 이미지와 csv파일 등 기타 파일
|-- core # 세팅, 데이터베이스, 외부 api용
|-- crud # DB에 쓰기, 읽기, 갱신과 삭제 작업
|-- migrations # 데이터베이스 버전 관리
|-- models # 데이터베이스 테이블 구조 관리
|-- routers # API를 용도별로 나눠 관리
`-- schemas # API의 반환과 crud에 필요한 ORM pydantic 클래스용
```

Inference Server

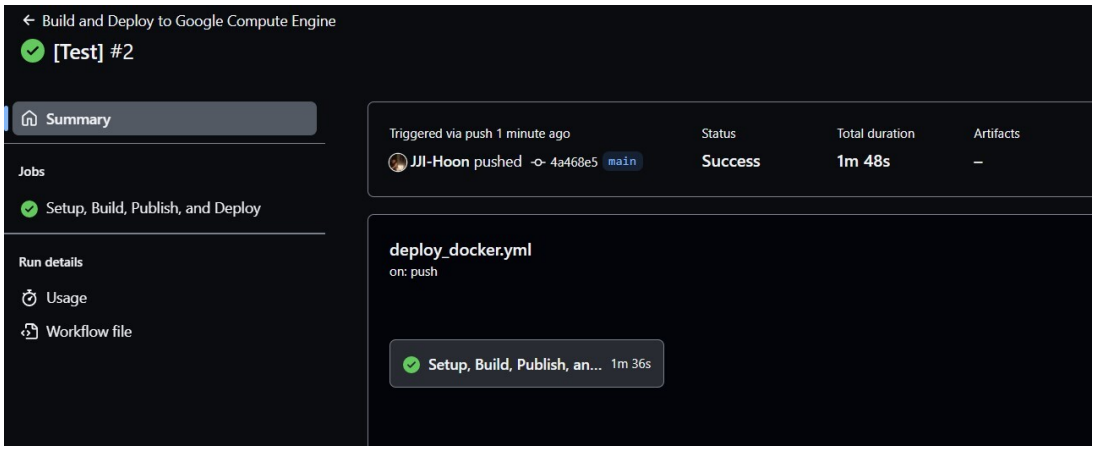
- 백엔드로 부터 전송 받은 유저 히스토리 데이터를 모델에 인풋하여 인퍼런스를 통해 유저의 추천 결과를 백엔드로 다시 전송할 수 있는 서버를 구축

1-3-4. CICD

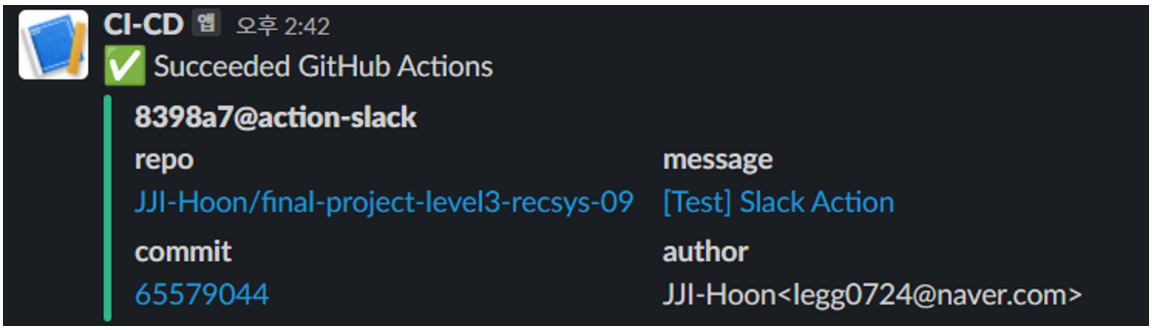
Github action & Slack



- **Docker & Github Action** : 백엔드와 프론트엔드를 도커를 이용해 이미지를 빌드하여, GCP 환경 VM에 업로드 진행 및 Github Action을 활용하여 CI/CD 환경 구축
  - **어려웠던 점** : Github Action 실행 과정에 있어서 gcloud 환경을 세팅하는데 있어서 Python의 버전 충돌로 인한 에러를 마주하여 원인을 학습하는 부분에 고충을 겪음
  - **해결 과정** : 학습을 통해 CI/CD를 진행하는데 있어서 Cloud환경과 사용하는 언어의 버전을 맞춰줘야 함을 알 수 있었고, Workflow에 Python 버전을 업데이트 하는 과정을 통해 해결함
  - **Github Action 구동 자료**



- **Slack 알림 기능 :** 협업을 위해 Github Action의 결과를 Slack에도 알림을 갈 수 있도록 하는 기능을 Github Action에서 구축
  - **Slack Alarm 자료**



Airflow

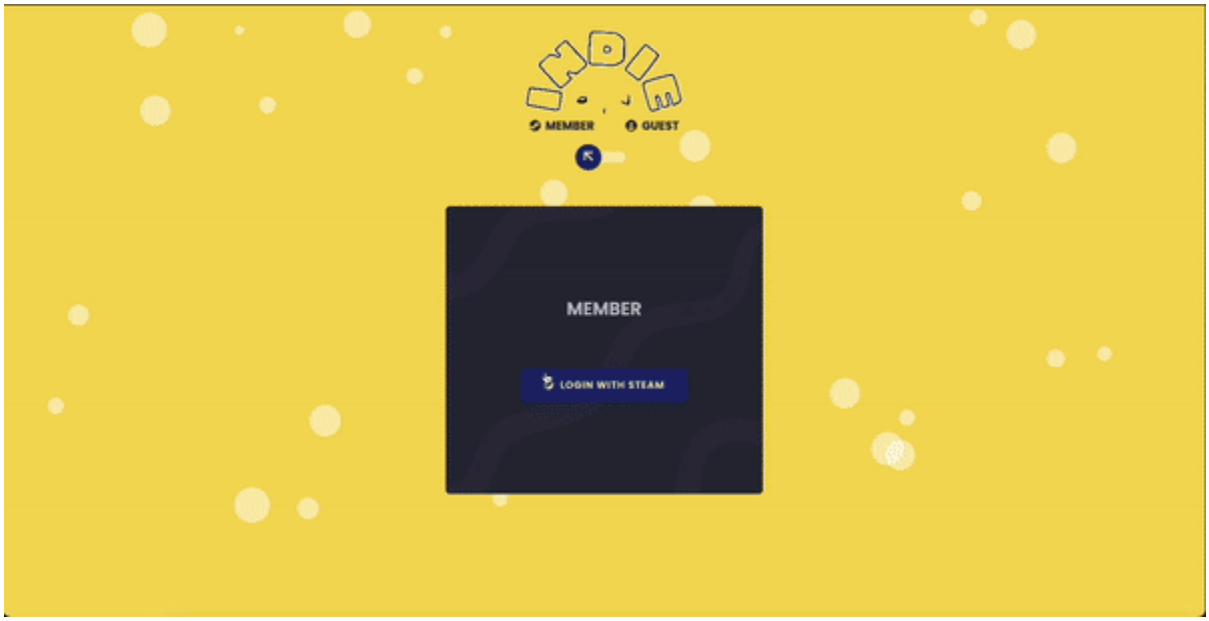
- DataLake에 적제된 신규데이터를 DataWarehouse인 BigQuery에 적재한 뒤, 모델 학습을 통해 성능을 체크하는 하나의 Flow 구축
- 실제 서비스 이후, Online data가 현재 모델 성능에 주는 영향을 파악해 추가 개선

1-4. 프로젝트 로드맵


2023년 1월
오늘
일
월
화
수
목
금
토
1월 1일
2
데이터 ...
프로젝트
3
1.3
일일기록
4
1.4
일일기록
5
1.5
일일기록
6
1.6
일일기록
7
8
9
1.9
일일기록
10
1.10
일일기록
11
1.11
일일기록
12
1.12
일일기록
13
14
서비스 강의 수강
이벤트
[기획 PM - 이지훈]
프로젝트
15
16
17
18
19
20
21
[기획 PM - 이지훈]
프로젝트
1.19
일일기록
1.20
일일기록
1.16
일일기록
1.17
일일기록
1.18
일일기록
[추천시스템 PM - 배성수]
프로젝트
22
23
24
25
26
27
28
[추천시스템 PM - 배성수]
프로젝트
일일기록
일일기록
오프라...
프로젝트
설 연휴
휴가
1.25
일일기록
중간 발표 (...
프로젝트
[서빙 PM - 김찬호]
프로젝트
29
30
31
2월 1일
2
3
4
[서빙 PM - 김찬호]
프로젝트
02.02
일일기록
01.30
일일기록
1.31
일일기록
3기 캠...
프로젝트
0
[발표 PM - 조원삼]
프로젝트
2.1
일일기록
이력서 및 ...
이벤트

1-5. 프로젝트 수행 결과

인디게임 추천서비스 IndieAn의 시연 영상



1-6. 자체 평가 의견

잘한 점

- GCP의 서비스를 이용해 목적에 따라 분리된 데이터베이스를 구축했다.
- 레포지토리를 포크를 하여 Github Action를 활용해 CI/CD를 구현했다.
- Airflow로 MLOps 사이클을 만들었다.
- 서버 연동 중 애로사항이 많았으나 협업을 통해 잘 해결했다.
- 서비스 구축의 이면에 필요한 것들을 이것저것 구축을 해봤다.

시도했으나 잘되지 않았던 점

- 추가적인 Item-based 모델과 추가적인 User-free 모델 탐색 및 적용
- 최신 유저 상호작용 데이터를 수집하기 위해 크롤링을 진행했으나 트래픽 제한으로 인해 데이터 수집의 어려움
- BM의 관점에서 이야기를 풀어나가는 것

아쉬운 점

- 유저의 활동 log를 저장해서 서비스에 반영하는 부분까지는 구현하지 못함

프로젝트를 통해 배운점

- 웹 프로토콜과 API 연동 과정
- 실제 서비스를 구현할 때 요구사항을 고려해 프로젝트를 설계하는 방

- 스팀의 파트너 API가 아닌 일반 API를 사용하여 스팀 사용자 기록을 불러올 때, 프로필 공개 설정을 사용자가 수동으로 진행해야 함
- 추천 이외의 서비스 제공
- 서비스의 제공을 하지 못해서 실제 서비스 작동시 마주하는 이슈를 경험해보지 못함
- 모델에 큰 신경을 주지 못하고 간단한 탐색으로만 끝났다.

법

- 실제 서비스 모델 Inference과정에서 속도의 중요성과 User-free model의 구조도