

2024 산학협력프로젝트 (캡스톤
디자인)

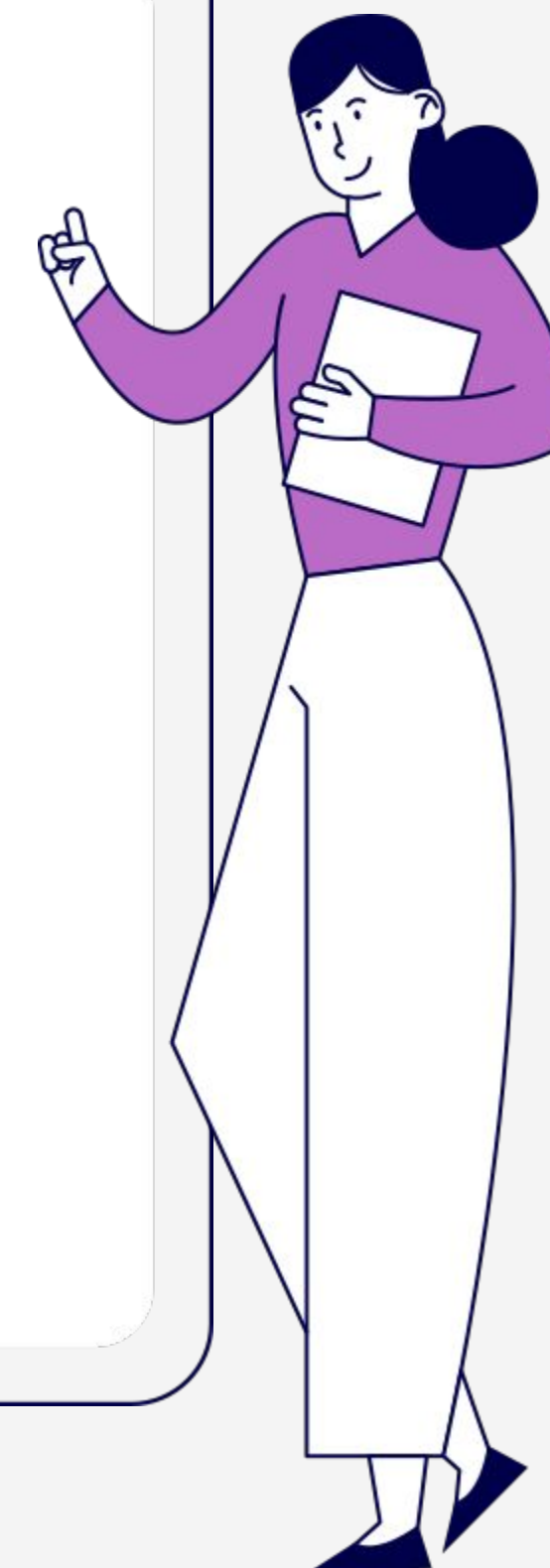
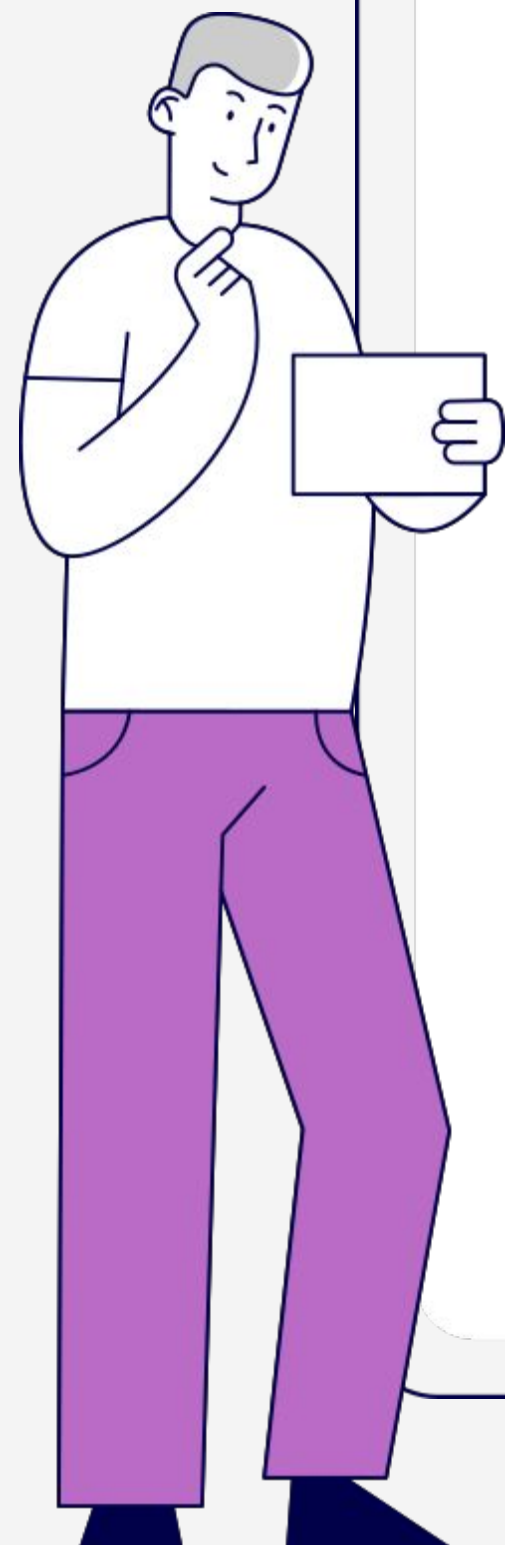


데굴 (The Gull):

자연어처리 기반 팀 내 참여도 및 기여도 분석 시스템을 활용한
협업 보조 Slack Bot

Team. MATE (임한빈 서재오)

학과 : 인공지능학부 책임교수 : 김만제



CONTENTS

프레젠테이션 목차 안내

1 프로젝트 배경

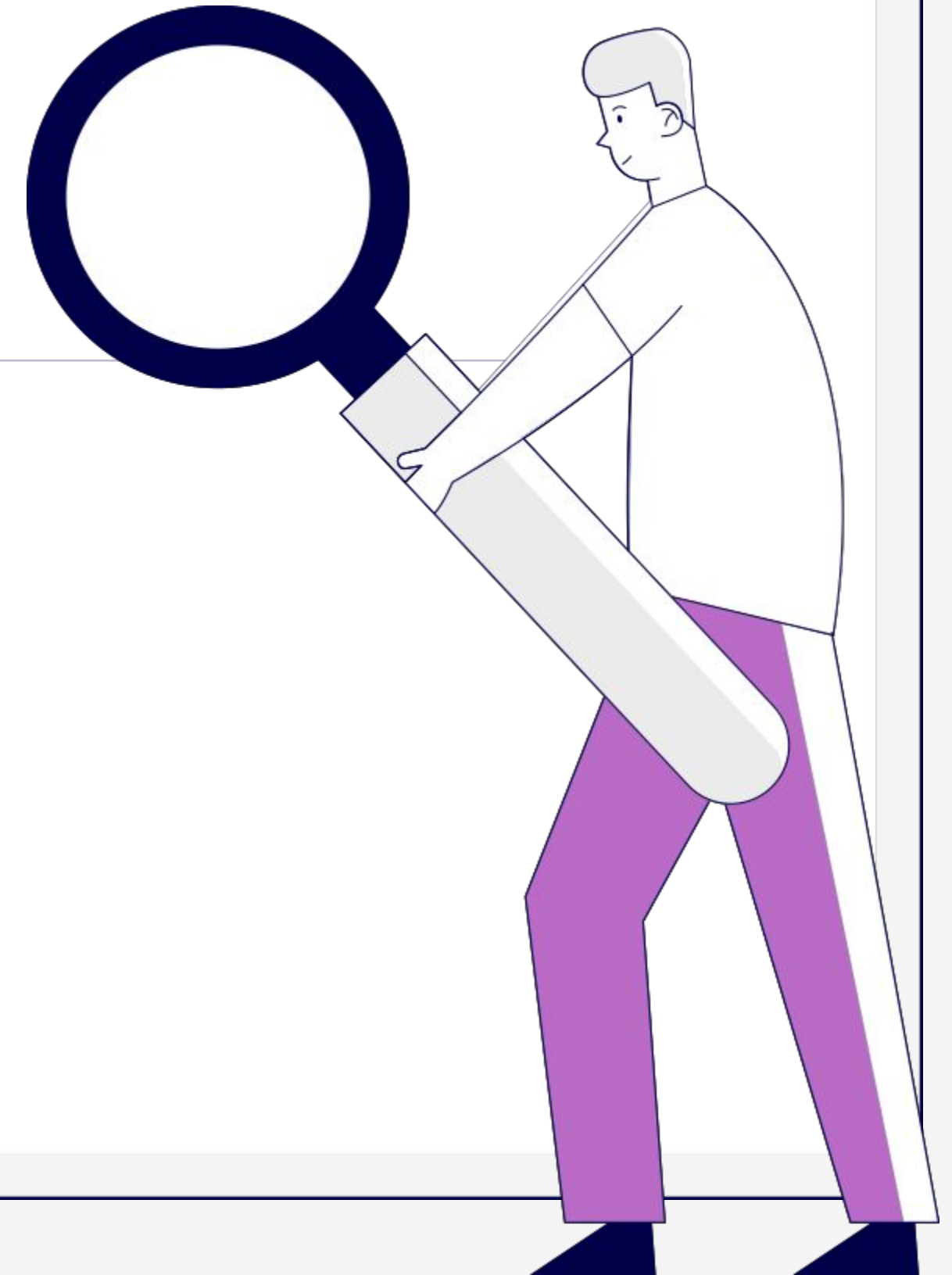
2 기존 방법 분석 및 한계점

3 프로젝트 목적

4 제안 아이디어

5 진행 상황 및 향후 계획

6 Q & A



01

프로젝트 배경



문제 인식

왜 사람들은 팀플을 꺼려할까 ?



협업



무임승차

> 팀플 '무임승차'

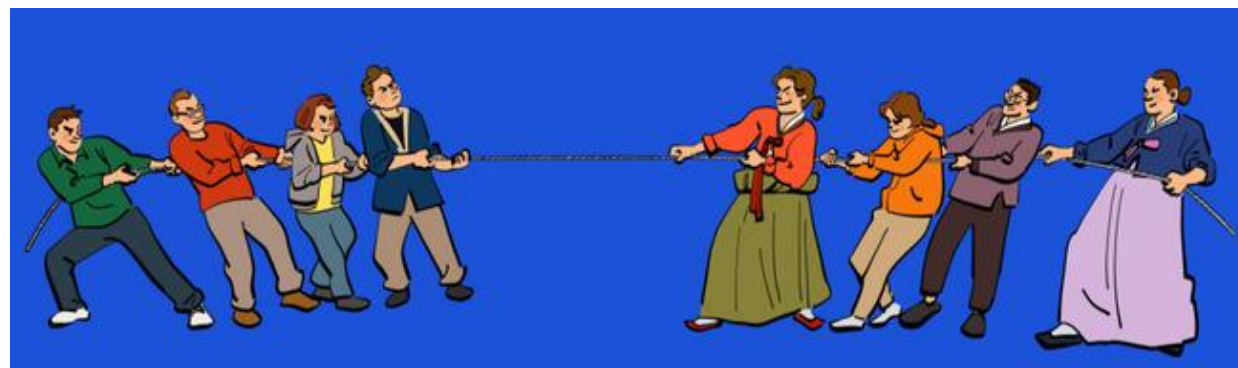
누구는 많이 일하고, 누구는 놀다가 무임승차하고 ...
결과가 좋은 나쁜 서로 감정만 상하고,
'팀 프로젝트'라고 하면 부정적인 감정만 떠오르게 만드는 경험들
팀플 과정에서의 팀원 간 불만과 갈등을 문제라고 인식

문제 정의

과연 팀원 개개인의 역량 차이가 팀원 간 감정 싸움이나 프로젝트 붕괴의 근본적인
원인일까?

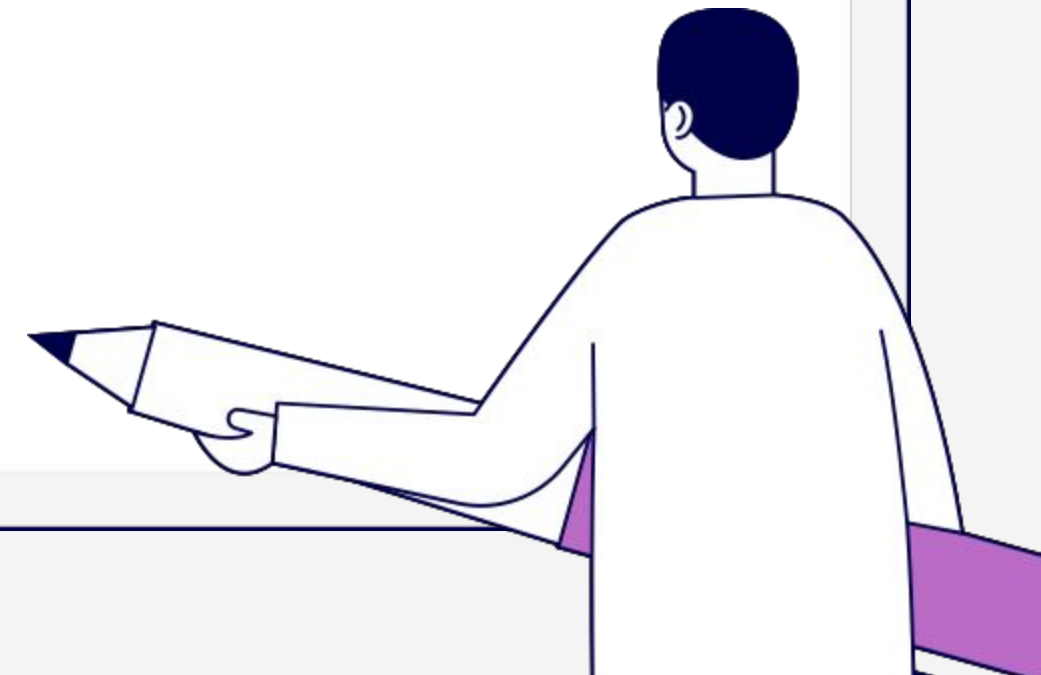
▶ 링겔만 효과

- 한 사람이 줄다리기를 할 때보다 참가자가 증가할수록 개인이 줄다리에 들이는 힘이 적어진다는 실험 결과
- 집단에서는 한 사람이 '나 하나쯤이야'라는 생각으로 게으름을 부려도 티가 나지 않아 결국 개인이 줄다리에 들이는 힘이 줄어든다는 점을 시사 - 한국심리학신문



문제 정의

팀원 간의 참여도 불균형으로 인한
프로젝트 진행 효율 저하



02

기존 방법 분석 및 한계점



기존 방법 분석 및 한계점

❖ Slack, Trello, Jira, Asana 등의 협업 도구

✓ 특징:

•업무 및 프로젝트 관리 지원:

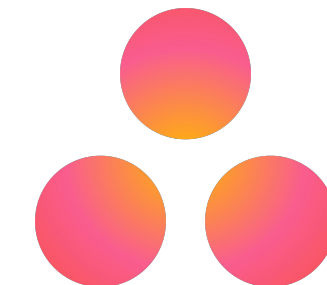
작업 할당, 진행 상황 추적, 마감일 관리 등을 통해 팀의 프로젝트를 관리

•팀 협업 기능:

댓글, 태그, 파일 공유 등을 통해 팀원 간 원활한 소통과 협업을 지원

•통합 및 확장성:

다양한 외부 도구와 연동하여 기능을 확장하고 업무 효율성을 높일 수 있음



기존 방법 분석 및 한계점

❖ Slack, Trello, Jira, Asana 등의 협업 도구

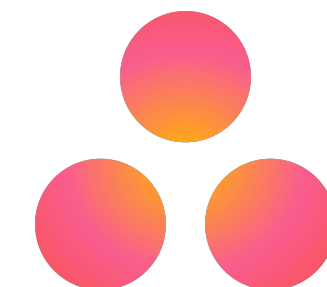
✓ 한계점:

•비정형적 협업 지원의 한계:

브레인스토밍이나 아이디어 회의 등의 비정형적인 협업 상황에서
균형 잡힌 참여를 유도하는 기능이 부재

•추가 기능 사용 시 비용 부담:

고급 기능이나 추가 통합을 활용하려면 유료 플랜으로
업그레이드해야 하며, 이는 예산이 제한된 팀에게 부담이 될 수 있음



기존 방법 분석 및 한계점

❖ Otter.ai

✓ 특징:

• 자동 음성 인식 및 전사:

회의나 대화의 음성을 실시간으로 텍스트로 변환하여 기록

• 공유 및 협업:

전사된 내용을 팀원들과 공유하고 함께 편집

• 다양한 통합:

Zoom 등 화상 회의 도구와 연동하여 회의 내용을 자동으로 기록



기존 방법 분석 및 한계점

❖ Otter.ai

✓ 한계점 :

•내용 이해의 한계:

음성을 텍스트로 변환하지만, 내용의 의미나 맥락을
심층적으로 분석하지는 않음

•언어 지원의 제한:

영어 등에 최적화되어 있으며, 다른 언어 지원이 제한적

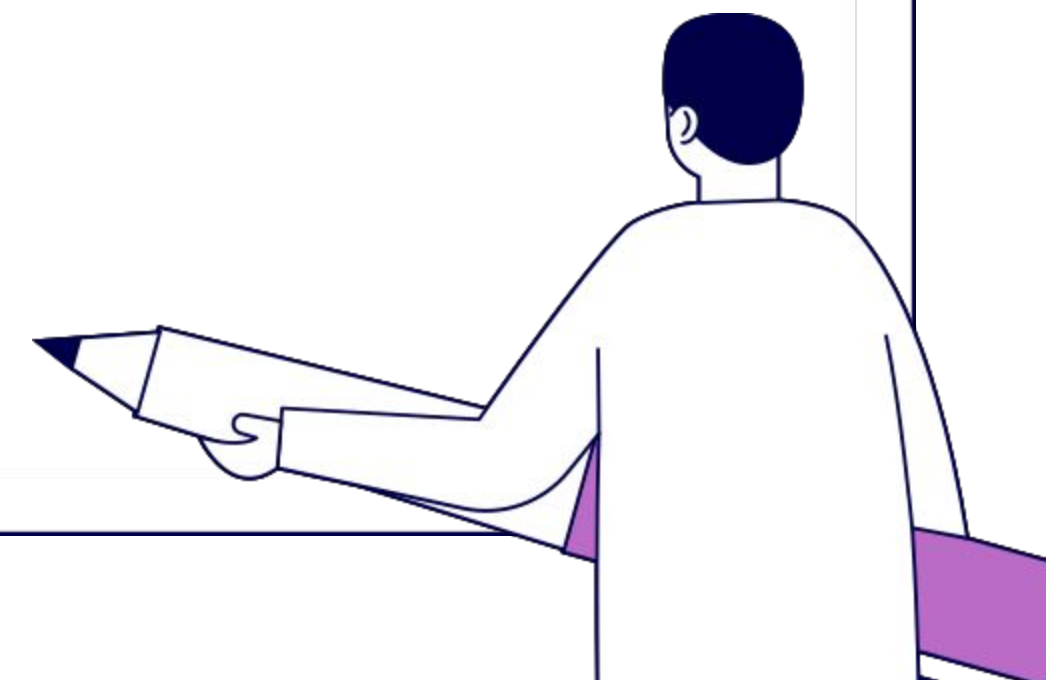
•정확도의 제약:

발음, 억양, 배경 소음 등에 따라 전사 정확도가 떨어질 수 있음



프로젝트 목적

팀원들의 적극적 참여 유도를
통한 원활한 협업 환경 구축



03

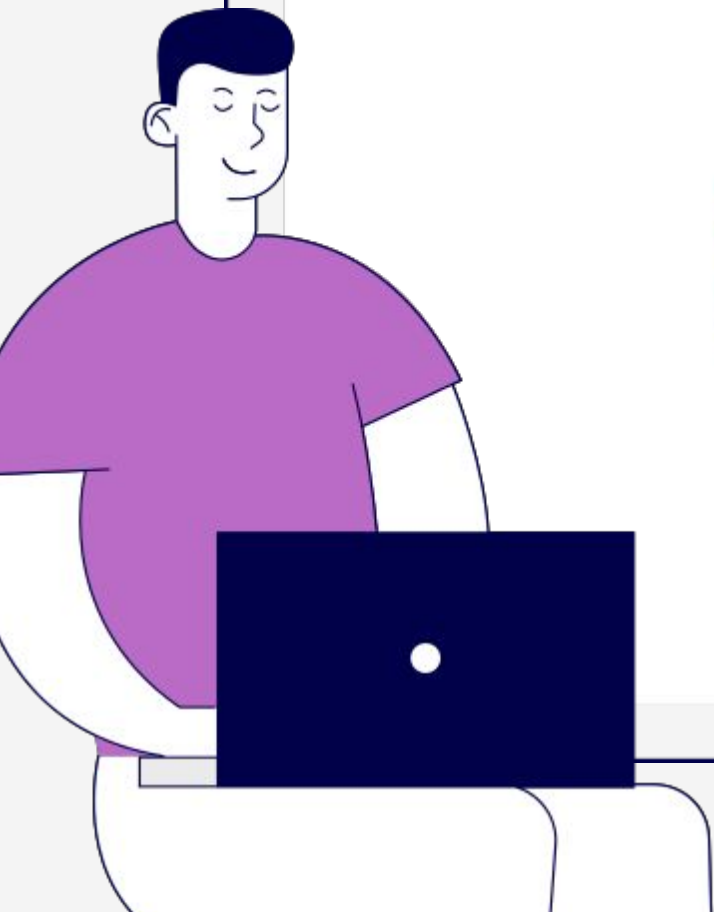
제안 아이디어




아이디어

데굴 (THE GULL):

자연어처리 기반 팀 내 참여도 및 기여도 분석 시스템을 활용한 협업 보조 Slack
Bot



기능 설명




참여도 및 기여도 분석

NLP(자연어처리) 기반으로
팀원들의 참여도와 기여도를
분석



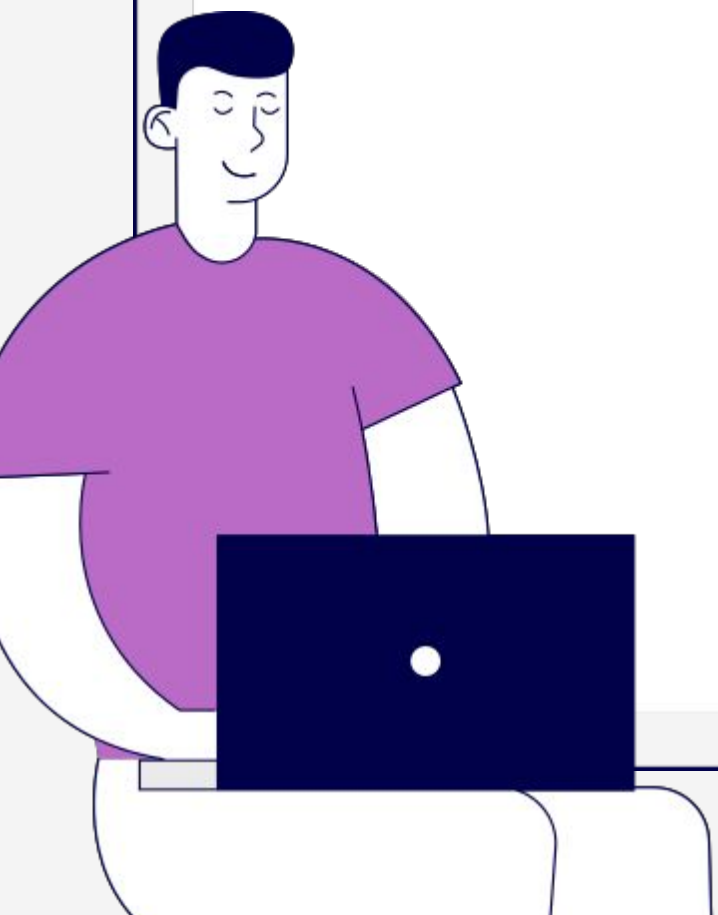
팀장 선정 도우미

첫 회의 분석 기반
팀장에 적합한 팀원 추천



상·벌 설정

참여도나 기여도에 따라
우수 팀원에게 줄 상을,
참여율 저조 팀원에게 줄 벌을
선택적으로 설정



참여도 산출 방법

[국어사전]

참여 (參與) 

1 어떤 일에 끼어들어 관계함.

> 참여도?

- 그 어떠한 주관도 들어가지 않은 '정량적' 척도
- 의미를 반영하지 않은 단순 수치



기여도 산출 방법

국어사전

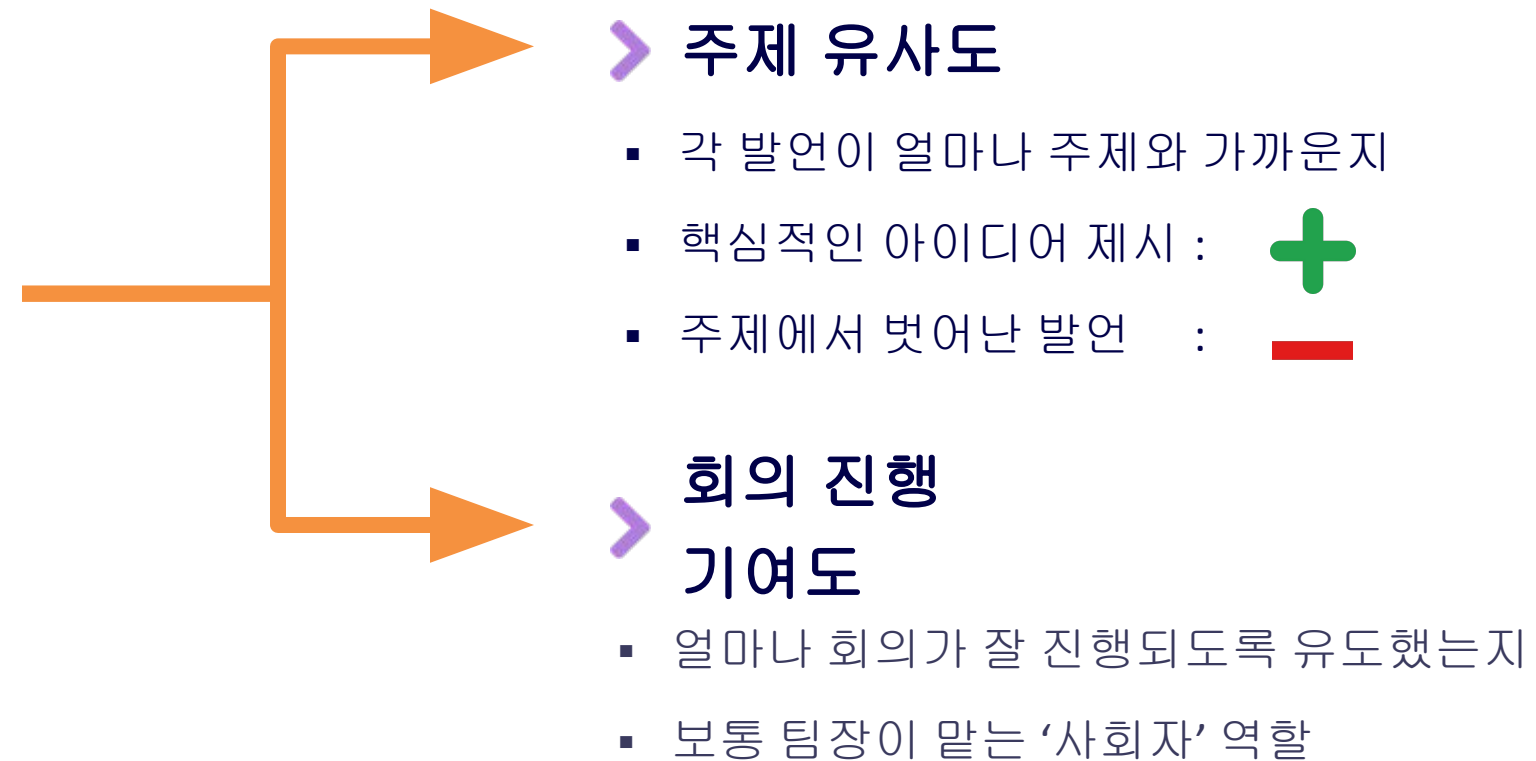
기여 (寄與) 

(명사)

1 도움이 되도록 이바지함.

➤ 기여도?

- LLM을 기반으로 '정성적' 요소를 수치화한 척도
- 의미가 반영된 수치



!! 일반적인 조별과제, 팀플, 공모전 등의 기획 회의를 가정

기여도 산출 방법

➤ 주제 유사도

1. LLaMA : 회의 주제 추출
2. BERT : 문장을 임베딩 벡터로 변환
3. 코사인 유사도 계산



각 화자 별 발언들의 평균 주제 유사도

➤ 회의 진행 기여도

LLaMA : 4가지 항목에 대해 1~5점으로 평가

- 1) 토론 유도 및 다른 팀원 발언 촉진
- 2) 회의 방향 설정 및 결론 도출 기여
- 3) 회의 목표 달성에 대한 기여
- 4) 협업 촉진 및 의견 차이 좁히기



각 화자 별 발언들의 회의 진행 기여도 합

기대 효과

▶ 모든 팀원들의 최소 참여도 보장

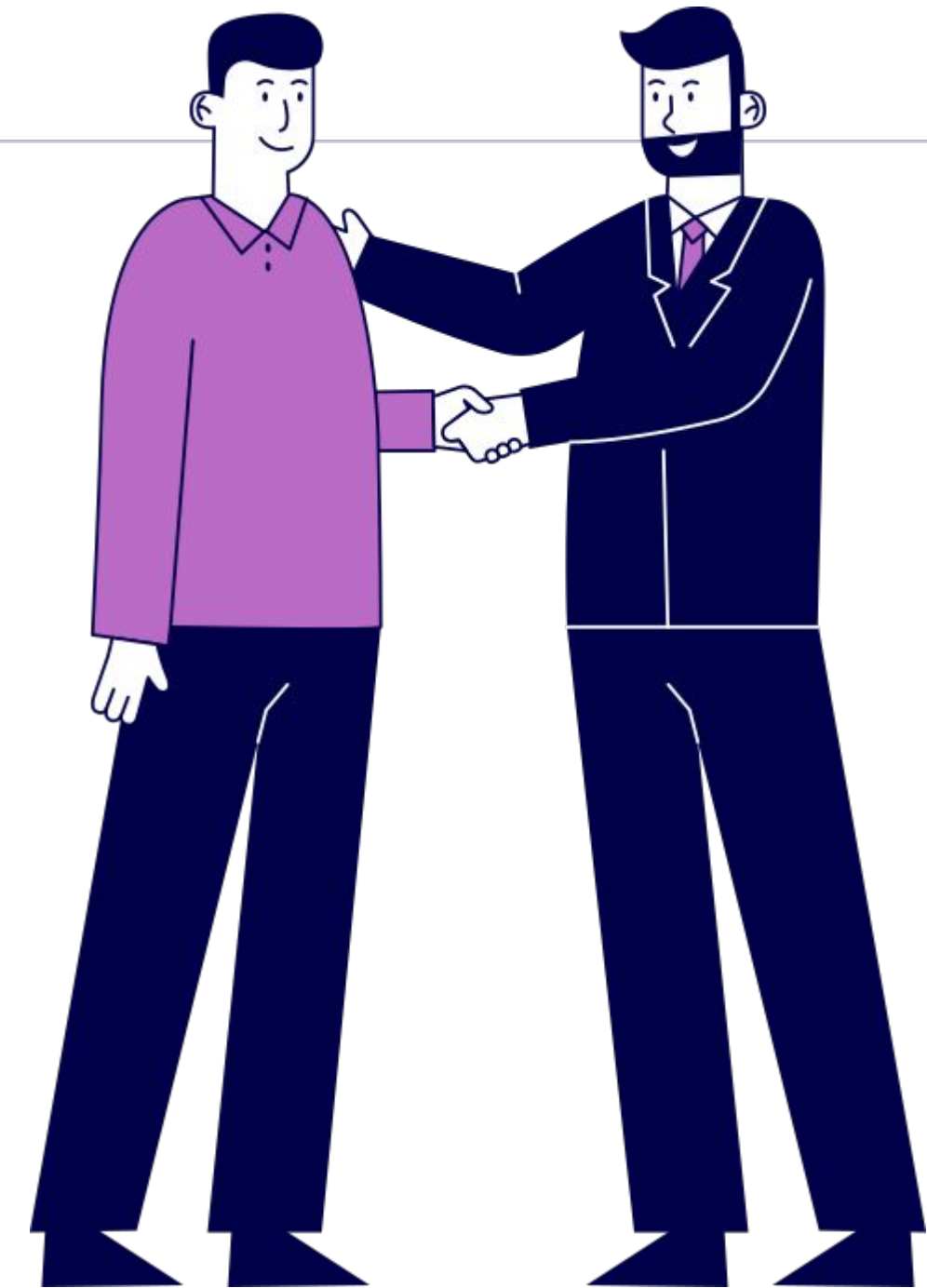
회의 참여도를 비롯한 여러 요인을 반영한 분석을 통해
각 팀원들의 참여도를 수치화하여 최소한의 참여 독려

▶ 팀 내 선의의 경쟁 유도

각자 참여도나 기여도를 확인하여
다른 팀원보다 더 많은 참여와 기여를 하고 싶은 마음이 생기게 함

▶ 상금 분배에 대한 갈등 방지

팀 내 주관적 평가가 아닌, 데이터 기반의 객관적인 참여도, 기여도
분석으로
추후 공모전 상금 분배와 같은 예민한 요소로 인한 불화 예방



04

진행 현황



진행 현황

1. Slack API 연동 및 Slash Command 구현

- Slack API를 연동하여 채팅과 명령어 기능 구현
- /hello와 /participation과 같은 Slash Command 제공

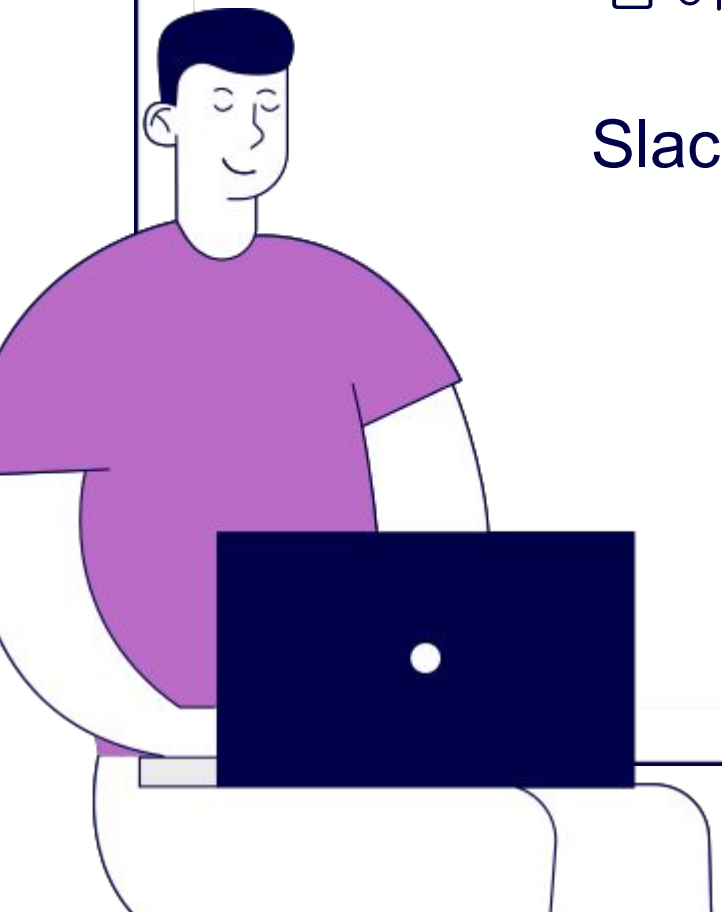
=> 사용자가 명령어를 입력하면 지정된 메시지와 참여도 분석 결과를 Slack 채널에 자동으로 출력



진행 현황

2. 참여도 분석 기능

- Slack 대화 내용을 바탕으로 각 사용자의 발화량(글자 수)과 발화 빈도(메시지 수) 계산
- 참여도 분석 결과는 matplotlib의 바(bar) 차트를 생성하고,
Slack 채널에 이미지를 업로드하여 공유하는 방식으로 구현



N

홈

DM

내 활동

더 보기

NOTT

플랜 업그레이드

초안 및 전송됨

채널

랜덤

일반

async_test

bot-test

middle_test

test2

채널 추가

다이렉트 메시지

H Hanby

seojaeohcoder

G Gull The 나

직장 동료 추가

앱

데굴이

앱 추가

NOTT 검색

middle_test

메시지 캔버스 추가 파일

"캡스톤 주제 무엇으로 할까" 입니다

오늘

G Gull The 오후 5:12

seojaeohcoder 오후 5:12

Hanby 오후 5:12

seojaeohcoder 오후 5:12

B I S | | | | | | | | | |

#middle_test에 메시지 보내기

slack_test.py - Visual Studio Code

+ 앱 추가

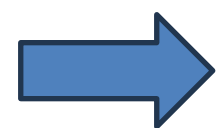
+ Aa 😊 @ 📺 🎤 📝



진행 현황

3. 기여도 분석 기능

- LLaMA 모델을 활용해 회의 진행에 대한 기여도를 평가하고,
발언마다 토론 유도, 회의 방향 설정, 목표 달성, 협업 촉진 측면에서 점수를 부여
- 각 발언의 점수는 대화 로그에 기반하여 집계되고, 시각화한 분석 결과를 Slack 채널에 공유



로컬 환경에서 모델 작동 X (Docker 메모리 이슈)

IBM Watsonx.ai

- LLM Serving

The screenshot displays the IBM Watsonx.ai dashboard for a user named Kate. The interface is divided into several sections:

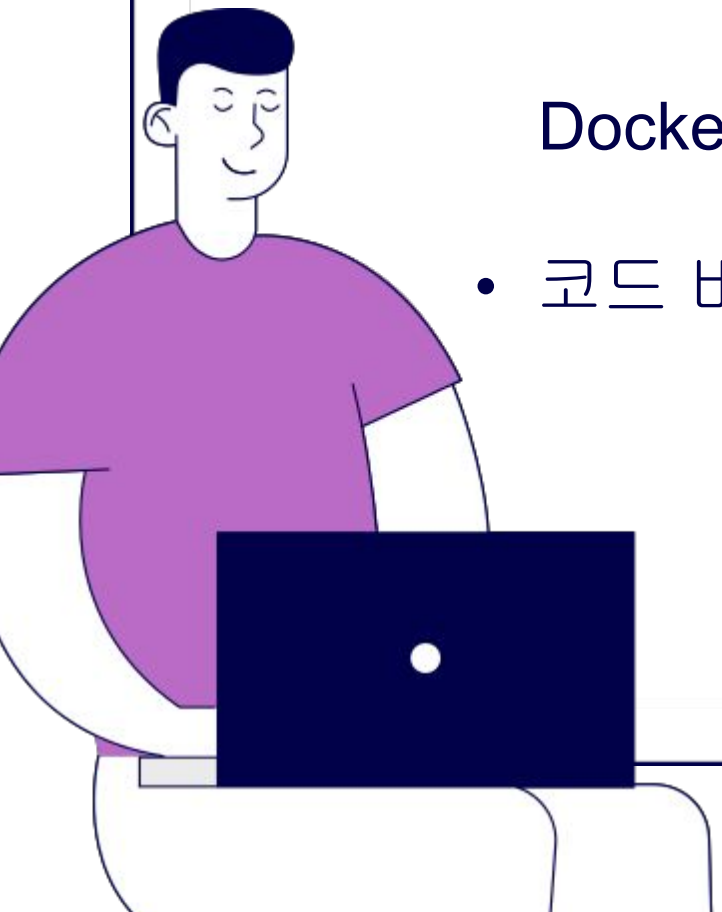
- Welcome, Kate:** A top section with a greeting and a description: "Train, deploy, validate, and govern AI models responsibly." Below this is a button labeled "Customize your journey".
- Open in: Kate's Sandbox:** A section with four cards for different tasks:
 - Experiment with foundation models and build prompts:** with Prompt Lab
 - Tune a foundation model with labeled data:** with Tuning Studio
 - Request or track models in AI use cases:** with AI governance
 - Build machine learning models automatically:** with AutoAI
- Jump back in:** A section with a "Recently visited pages" link and four project links:
 - Project / flights-data_shaped.csv
 - Home / Projects
 - Library 1 / Sample data for bank marketing
 - Deployment space / Bank marketing model
- Discover:** A section with a "Resource hub" and a "Featured" area.
 - Resource hub:** A table with three rows:

Foundation models	Prompts	→
Explore foundation models from Hugging Face and other third-parties depending on your use case.	Data	→
	Projects	→
 - Featured:** A section with a diamond icon, the text "granite-13b-chat-v2", and a description: "The Granite 13 Billion chat V2".

진행 현황

4. Docker 및 Git을 활용한 개발 환경 구축

- 도커(Docker)를 사용하여 일관된 개발 환경을 제공하고, 다양한 라이브러리 버전 충돌 문제 해결
- 도커 이미지에 프로젝트 의존성 패키지를 미리 설치하고,
Docker Compose를 활용해 서버 실행과 네트워크 설정 자동화
- 코드 버전 관리에는 Git을 활용, 프로젝트의 진행 상황을 추적하고, 팀원 간의 원활한 협업 지원



진행 현황

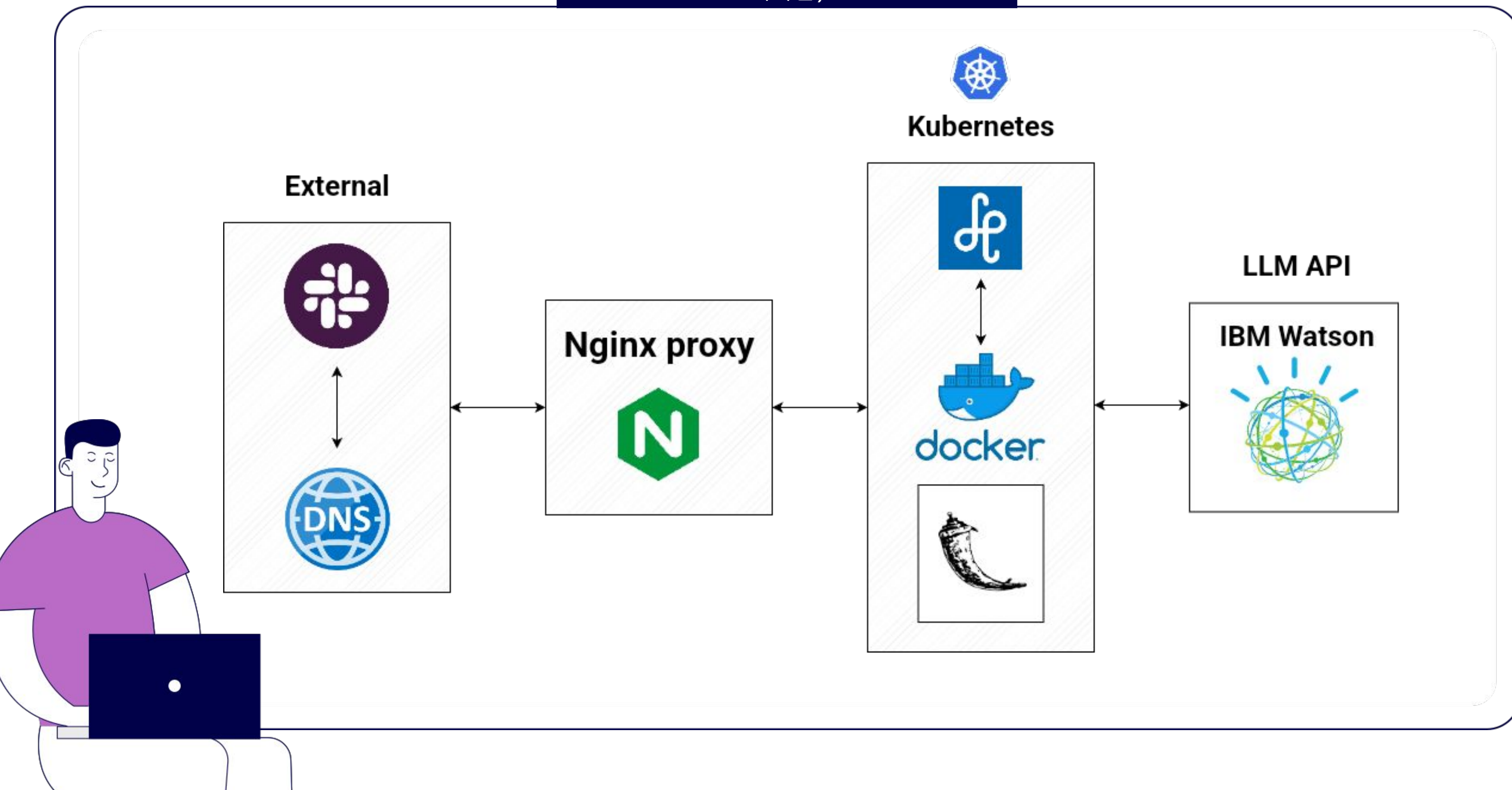
5. DevOps

- NGROK를 통해 로컬 서버를 외부에 노출, Slack과의 API 통신을 수행할 수 있도록 설정
- Flask 웹 프레임워크를 통해 API 요청을 처리

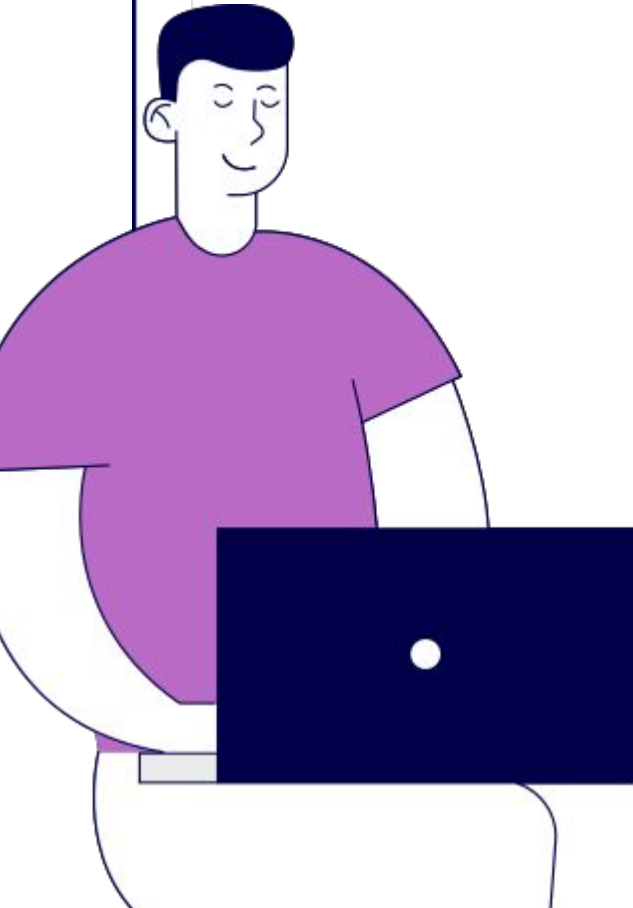
세션이 종료되면
URL이 변경됨

8. Slack Test (Docker NGROK Network Setting)

```
# 네트워크를 만들어 2개의 docker container가 통신이 가능하도록 만들.  
docker network create my_network  
docker network connect my_network <container_name_api>  
docker network connect my_network <container_name_ngrok>  
flask run(IN you api container | API container에서 실행중이어야 NGROK에서 포워딩 가능. 반드시 선행  
docker run --net=host -it -e NGROK_AUTHTOKEN=YOUR_NGROK_AUTHTOKEN ngrok/ngrok:latest http 5000  
  
# 2개의 결과가 동일해야 함. (제대로 통신이 되는지 확인)  
http://localhost:4040  
http://127.0.0.1:5000
```



2024 산학협력프로젝트 (캡스톤 디자인)



The image shows a Docker Desktop interface on a tablet. The interface is dark-themed with a blue header. The header includes the Docker Desktop logo, a 'PERSONAL' badge, and a search bar. Below the header, there's a navigation sidebar on the left with icons for Containers, Images, Volumes, Builds, Docker Scout, and Extensions. The main area displays the details of a container named 'k8s_deploy-container_deploy-job-2pk5k_default_afd93792-835a-40cb-b117-f058fcb5db8d_0'. The container is running the 'thegull:latest' image. Below the container name, there are tabs for Logs, Inspect, Bind mounts, Exec, Files, and Stats. The 'Logs' tab is selected, showing a list of log entries. The log entries show the container's startup process, including applying initial configuration, starting Localtunnel, creating/updating ConfigMap, setting up Kubernetes CronJob, and applying Kubernetes configurations. The final log entry states 'Deployment completed successfully!'.

docker desktop PERSONAL

Search for images, containers, volumes, ...

You can do more when you [sign in](#).

Containers

Images

Volumes

Builds

Docker Scout

Extensions

Containers / k8s_deploy-container_deploy-job-2pk5k_default_afd93792-835a-40cb-b117-f058fcb5db8d_0

k8s_deploy-container_deploy-job-2pk5k_default_afd93792-835a-40cb-b117-f058fcb5db8d_0

d0566171e19a thegull:latest

Logs Inspect Bind mounts Exec Files Stats

2024-10-31 16:00:57 Applying initial configuration using init.yaml...

2024-10-31 16:00:57 job.batch/init unchanged

2024-10-31 16:00:57 Starting Localtunnel to tunnel the Flask application...

2024-10-31 16:01:02 Localtunnel started successfully. URL: https://long-pigs-jump.local.lt

2024-10-31 16:01:02 Creating/updating ConfigMap from .env file...

2024-10-31 16:01:02 configmap/app-env configured

2024-10-31 16:01:02 Setting up Kubernetes CronJob for IP update...

2024-10-31 16:01:02 cronjob.batch/network-update unchanged

2024-10-31 16:01:02 Applying Kubernetes configurations for deployment and service...

2024-10-31 16:01:03 deployment.apps/thegull-deployment created

2024-10-31 16:01:03 service/thegull-service unchanged

2024-10-31 16:01:03 Your Localtunnel URL is: https://long-pigs-jump.local.lt

2024-10-31 16:01:03 Deployment completed successfully!

05

향후 계획



향후 계획

1. 정기 메시지 저장 및 암호화 관리

- 슬랙 메시지가 90일 후 삭제되므로, 외부 DB 대신 워크스페이스의 메시지를 파일(txt, json)로 암호화하여 서버에 저장
- 추가적인 DB 구축이나 비용을 최소화하고 보안을 강화할 예정

2. 배포 및 서버 관리 (DevOps, CI/CD)

- **Ncloud** 마이크로 서버를 활용해 초기 배포 후, 접속량에 따라 서버를 조정
- 최종 배포 전략은 **오픈소스**로, 사용자가 로컬에서 쉽게 실행할 수 있도록 설치 파일(EXE)을 제공
- 보안 서버 비용 절감을 위해 사용자의 컴퓨팅 자원을 활용해 자동으로 **Kubernetes** 기반의 **Slack Bot**이 실행되도록 설정

향후 계획

로컬 실행 자동화

- **Docker, Kubernetes, NGINX, FreeDNS**를 이용해 로컬 실행 자동화를 구현하여 사용자들이 쉽게 접근하고 서비스 실행 가능
- 유료 버전의 NGROK 없이도 고정된 URL을 제공해 접근할 수 있도록 설정
- ✓ **유동 IP 관리 자동화 (Kubernetes, NGINX, FreeDNS)**
- ✓ **최신 버전 확인 및 알림 (GitHub API)**
- ✓ **직관적인 GUI 기반 관리**

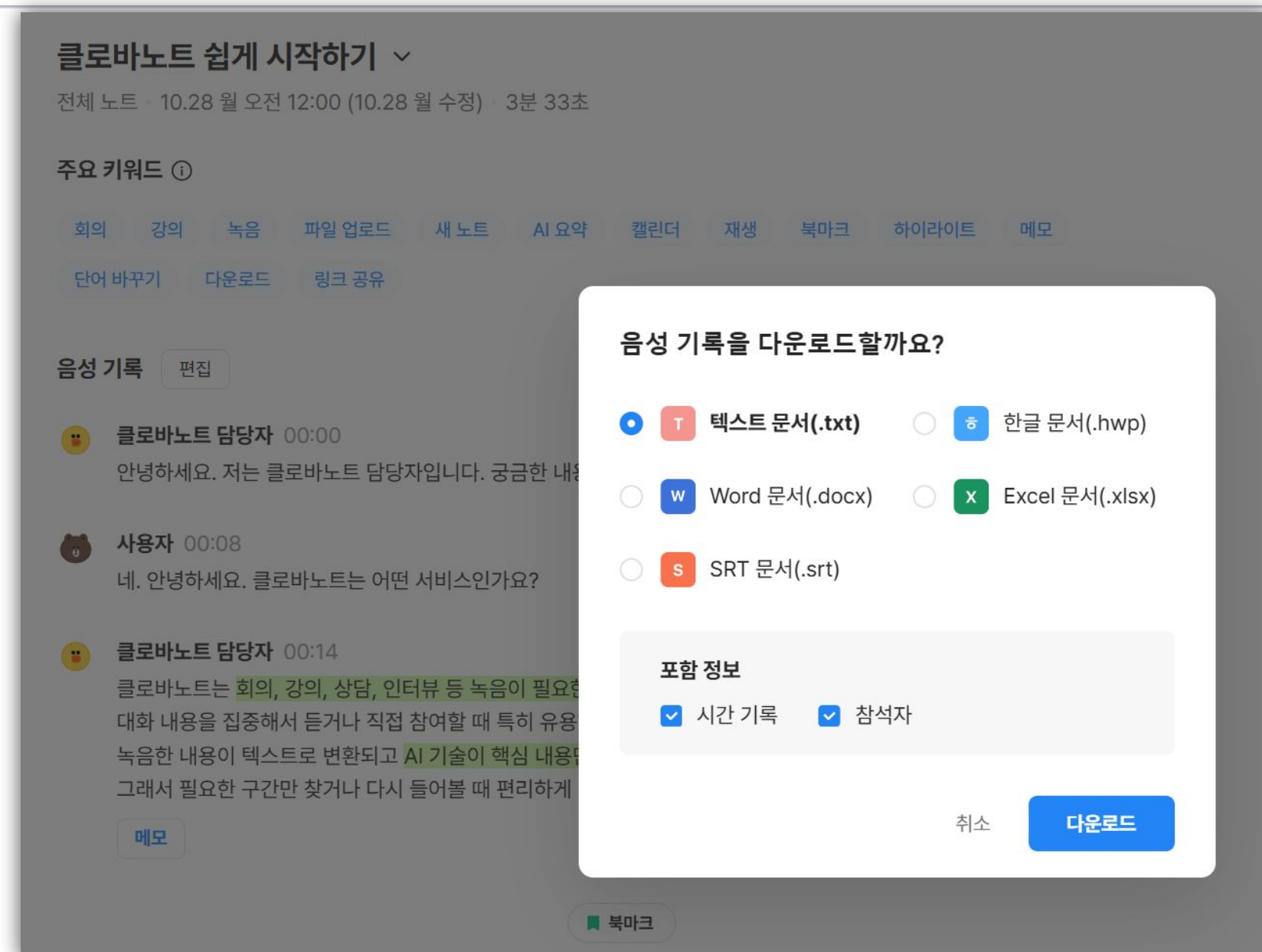
향후 계획

4. 기능 추가

• 클로바노트 연계:

STT 처리된 회의 음성 기록 파일을

Slack Bot으로 받아서 분석



향후 계획

- **LLM 모델 경량화**: 언어 모델을 경량화하여 성능 최적화 및 효율적인 자원 사용
 - IBM Watsonx.ai API
- **UI/UX 개선**: 슬랙 메시지에 캐릭터성을 부여하여 사용자 경험을 향상
- **보상 및 벌점 시스템**: 회의 기여도와 참여도에 따른 보상 및 벌점 시스템 구축
- **감정 인식 모델**: 회의 중 긍정적인 응답을 받은 발언에 가중치를 부여, 발언의 질을 평가

2024 산학협력프로젝트 (캡스톤
디자인)



Q & A

감사합니다

Team. MATE (임한빈 서재오)

