


캡스톤 디자인 I

종합설계 프로젝트

프로젝트 명	책봇 (기업 맞춤형 웹사이트 비서)
팀 명	37조 -수윳현모
문서 제목	결과보고서

Version	1.2
Date	2024-MAY-24

팀원	고 강현 (조장)
	곽 다윗
	구 형모
	김 은수

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24


CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인I 수강 학생 중 프로젝트 “책봇”을 수행하는 팀 “수윳현모”의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 “수윳현모”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

문서 정보 / 수정 내역


Filename	결과보고서-책봇.doc
원안작성자	고강현, 곽다윗, 구형모, 김은수
수정작업자	고강현, 곽다윗, 구형모, 김은수

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2024-05-16	김은수	1.0	최초 작성	
2024-05-20	곽다윗	1.1	추가 작성	문서 추가 작성
2024-05-23	구형모/고강현	1.2	추가 작성	문서 추가 작성 및 보완

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

목 차

1	개요	4
1.1	프로젝트 개요	4
1.2	추진 배경 및 필요성	4
2	개발 내용 및 결과물	5
2.1	목표	5
2.2	연구/개발 내용 및 결과물	6
2.2.1	연구/개발 내용	6
2.2.2	시스템 기능 및 구조 설계도	6
2.2.3	활용/개발된 기술	6
2.2.4	현실적 제한 요소 및 그 해결 방안	6
2.2.5	결과물 목록	6
2.3	기대효과 및 활용방안	6
3	자기평가	7
4	참고 문헌	7
5	부록	7
5.1	사용자 매뉴얼	7
5.2	운영자 매뉴얼	7
5.3	배포 가이드	7
5.4	XXX 매뉴얼	7
5.5	XXX에 대한 기술 문서	7

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

1 개요

1.1 프로젝트 개요

“**챗봇(기업형 웹사이트 AI 비서)**”는 상담 문의 복잡도가 늘어나고 있는 다양한 기업 웹사이트에 사용자 응대 효율성을 높일 수 있는 서비스이다. 기존에는 많은 기업들이 자사 웹사이트에 사용될 챗봇을 직접 만들면서, 많은 서버 운영과 개발 비용을 사용해서 챗봇을 개발했다. 이는 많은 기업들에게 큰 부담으로 작용 하고 있다.

이러한 문제를 해결하기 위해서 “**챗봇(기업형 웹사이트 AI 비서)**”는 맞춤형 챗봇 Package를 제공한다. 사용자가 입력한 키워드, PDF, 웹사이트 주소등 여러 정보를 기반으로 맞춤형 챗봇을 만들 수 있다. “**챗봇(기업 맞춤형 웹사이트 비서)**”를 사용한다면 기존의 단순한 챗봇 제작과 비교해서 시간적으로나 비용적으로 효율성을 높일 수 있다.

1.2 추진 배경 및 필요성

이번 프로젝트의 필요성은 맞춤형 챗봇의 늘어나는 수요에 있다. 기존 챗봇은 미리 설정해둔 답변이나 다수의 선택을 기반으로 사용자에게 응대하는 방식이다. 하지만, 이제는 많은 기업들과 사람들은 단순한 챗봇이 아닌 맞춤형 AI 챗봇을 원하기 시작했다. 챗봇을 필요로 하는 기업에 각각의 사용자에게 맞는 맞춤형 챗봇을 개발할 수 있도록 Package를 제공한다면 좋은 반응을 얻을 것으로 기대된다.

이번 프로젝트의 추진 배경은 AI에 대해 잘 모르더라도 자사의 서비스에 맞는 맞춤형 챗봇을 쉽게 생성할 수 있도록 하는 기술 구현과 챗봇의 챗봇을 쉽게 본인의 서비스, 회사에 적용 가능한 Package 제공에 초점을 맞췄다.


2 개발 내용 및 결과물

2.1 목표

“**ai 챗봇 개발에 지식이 없는**”, “**챗봇을 개발할 비용이 부족한**” 기업들에게 자신만의 챗봇을 설정하고 서비스 할 수 있게 해주는 서비스를 제공한다.

- 기능적인 측면

상담 문의 복잡도가 증가함에 따라, 많은 기업들이 자사 도메인에 맞춤형된 챗봇을 만들려고 합니다. 이런 수요에 맞춰서, AI를 모르더라도 손쉽게 “**기업 맞춤형 챗봇**”을 만들 수 있도록

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수릿현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

도와주는 챗봇 Package를 개발하는게 목표입니다.


- 기술적인 측면

1. 할루시네이션

기존의 chat봇의 방식의 경우 이상한 정보나 없는 정보도 대답을 해주는 할루시네이션 현상이 일어난다. 하지만 이러한 현상은 기업형 챗봇에는 허용 가능한 이야기가 아니다. 정확한 서비스에 대한 정보를 알려주어야 하는 기업형 챗봇에는 이러한 문제가 발행하면 안되기에 RAG를 사용하여 할루시네이션을 극복하는 것을 목표로 한다.

2. 비싼 비용, 어려운 지식

사이트를 외주를 맡기거나 구매하여 사용하는 기업들의 경우 프로그래밍, 그리고 챗봇 제작에 대한 기술 지식이 부족하다. 이러한 기업들은 챗봇을 제작하려해도 비싼 돈으로 외주를 맡기거나 직접 공부하고 채용하여 시간을 쏟아야 한다. 시간, 비용, 지식의 문제점을 쉽고 간단한 제작과 적용 방법으로 해결하는 것을 목표로 한다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수릿현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

2.2 연구/개발 내용 및 결과물

2.2.1 연구/개발 내용

● Server

- Chatbot AI

본 프로젝트의 가장 큰 기술이자 중요한 기술 중 하나인 **chatbot ai**이다. 본 프로젝트의 틀을 짤때 **open ai**의 서비스 방식을 참고하여 없는 것들을 확인 후 진행하였다. 처음에는 **kobert**를 사용한 직접 학습한 모델을 사용할 것을 고려하였으나 더 많은 **gpu**, 자원 들이 들어간 **GPT**모델을 사용하는 것이 성능과 시간적으로 훨씬 이득이 될 것이라 생각하여 방법을 변경하였다.

Open ai의 모델을 사용하여 프로젝트를 진행 하기 위해서는 아래와 같은 기술적 방법이 필요하다.

1. 커스텀 챗봇 제작
2. 회원 별 커스텀 챗봇 저장
3. 할루시네이션 제거
4. 여러 데이터 학습 방법

이렇게 4가지의 기술적 요소가 가능한지에 따라 프로젝트의 방향이 정해진다고 판단하여 테스트를 먼저 진행하였다.

1. 커스텀 챗봇 제작

커스텀 챗봇을 만들기 위해 먼저 **open ai api**를 살펴 보았다. **open ai api**를 좀더 쉽고 잘 활용하기 위해 **langchain**을 사용하였고 **chatgpt**처럼 간단한 발화문에 대한 답변이 잘 나오는 것을 확인하였다. 하지만 본 프로젝트에 필요한 기술은 커스텀 챗봇이다. **gpt**모델을 커스텀 할 방법들을 고안하고 찾아보았다. 첫번째에는 **finetuning**방식을 생각하여 조사를 진행하였다.

```

from openai import OpenAI
client = OpenAI()

client.files.create(
    file=open("train.jsonl", "rb"),
    purpose="fine-tune"
)


```

fine tuning 방식의 경우 학습 데이터를 정제하고 정제된 데이터를 위의 사진과 같은 방식으로 학습 시켜 본인의 의도에 맞는 챗봇을 구성하는 방식이다. 이러한 방식의 문제점을 조사중에 몇가지 발견하였다. 먼저 위 사이트의 목적은 비용이 최대한 적은 방식으로 챗봇을 제공해야한다. 하지만 **fine tuning**을 진행하려면 데이터를 정제해야하고 적은 데이터가 아닌 꽤 많은 데이터가 필요하다. 그리고 **fine tuning**을 진행하는 것조차 비용이 추가된다. 비용의 경우 아래와 같다.

모델	Chat-GPT fine-tuned	ChatGPT (4K)	GPT-4 (8K)
Training (학습)	\$0.0080	(-)	(-)
Input usage (인풋 텍스트)	\$0.0120	\$0.0015	\$0.0300
Output usage (생성)	\$0.0160	\$0.0020	\$0.0600

그렇게 엄청난 비용은 아니지만 비용이 계속 추가되는 방식으로 진행을 한다면 의미가 없을 것이라 판단하고 다른 방식을 찾아보았다. 찾던 중 **RAG** 방식을 발견했다. **RAG**란 **Retrieval-Augmented Generation**의 약자이며 외부에 데이터를 두어 참고하는 방식을 동작한다. 기존의 **fine tuning**과 다른 점은 **fine tuning**의 경우 학습 되어있는 모델에 값을 추가로 넣어 가중치를 변경하는 방식이다. 이러한 방식은 특정 **downstream task**에는 좋은 성능을 보인다. 하지만 할루시네이션의 문제점 없는 데이터에 대한 정보가 들어올 경우 할루시네이션이 다시 발생한다. 이러한 경우 다시 **fine tuning**을 해야하는데 이러한 과정이 너무 번거롭고 비용도 추가적으로 든다. 이러한 방식을 해결한 방법이 **vector db**를 사용한 **RAG**이다. 외부의 데이터를 임베딩 시켜 저장하고 내부의 **gpt**모델의 정보와 외부의 데이터를 같이 활용한다. 이러한 방식 덕분에 할루시네이션을 쉽고 빠르게 대응할 수 있다는 장점이 있다.

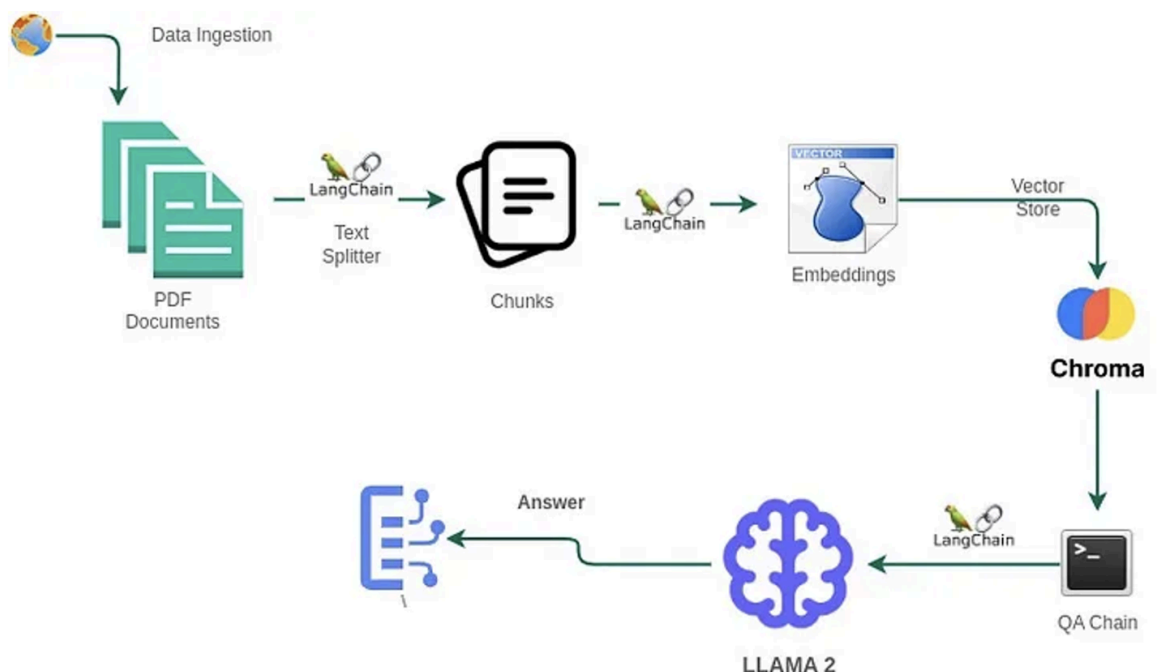
이제 이러한 **RAG**방식을 어떻게 기술에 접목 시켜 사용할 수 있을까를 고민하고 테스트 하였다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

먼저 langchain을 사용하여 gpt모델을 가져가고 추가적으로 vector db를 활용하는 방식을 구상하였다. vector db는 무료인 chroma db를 고려하였다. 먼저 간단한 chroma db를 활용한 커스텀 챗봇을 아래와 같은 순서로 테스트 진행하였다.

- Web에서 문서 가져오기
- chunk 단위로 문서 자르기
- OpenAI의 Text Embedding 모델을 사용해서 임베딩하기
- Chroma Vector DB 에 저장하기
- ChatGPT 모델을 기반으로 검색하기


위와 같은 방법으로 간단한 url링크로 커스텀 챗봇이 잘 작동하는 것을 확인하였다.



chroma db와 gpt모델을 사용하는 구조는 위와 같다.

2. 회원 별 커스텀 챗봇 저장

vector db를 활용하여 정상적으로 커스텀 하는 것을 위에서 확인하였다. 하지만 문제점이 한가지

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

발생했다. 위의 테스트에 활용한 **chroma db**의 경우 **langchain** 패키지 안에 들어있는 **chroma db**를 활용한 방식이다. 이러한 방식에는 **chroma db**를 사용할때 **local file**로 저장되며 유저별로 저장을해서 불러오는 방식이 없다는 것을 발견하였다. 이렇게 된다면 사용자 별로 데이터를 다른 파일명으로 저장해서 해결해야했다. 하지만 이러한 방식은 보안, 용량상의 문제가 있다고 판단하여 다른 방식을 고안하였다. 먼저 **chroma db** 자체적으로 따로 저장하는 방식이 있는지 살펴보았다. 살펴본 결과 **collection** 단위로 분류하여 저장할 수 있다는 것을 확인하고 테스트를 진행하였다.


```
# 컬렉션의 항목 수를 출력합니다.
print(
    "현재 저장된 Collection 의 개수는 ",
    langchain_chroma._collection.count(),
    " 개 입니다.",
)
```

현재 저장된 Collection 의 개수는 51 개 입니다.

여러가지의 데이터를 각각 다른 이름의 collection에 저장하고 제대로 저장되었는지 확인하였다. 확인 결과 데이터 별로 따로 저장 가능하며 추후에 데이터를 추가할 수 있다는 것까지 확인하였다. 다음은 이러한 chroma db를 langchain과 같이 활용할 수 있는 방식을 살펴보았다. langchain chroma db라는 패키지가 있다면 langchain에서 나오기 전에도 활용할 수 있는 방식이 있을 것이라고 판단하였다.

```
from langchain_community.vectorstores import Chroma
```

langchain_community내에 벡터 스토어가 있는 것을 확인하였고 그중에 chroma 클래스가 있는것을 확인하고 살펴보았다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수릿현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

```

langchain_chroma = Chroma(
    # PersistentClient를 전달합니다.
    client=persistent_client,
    # 사용할 컬렉션의 이름을 지정합니다.
    collection_name="chroma_collection",
    # 임베딩 함수를 전달합니다.
    embedding_function=stf_embeddings,
)

```

살펴본 결과 chroma객체를 생성할때 collection의 이름을 전달하여 제작할 수 있는 것을 확인하였고 유저 별로 각각 다른 데이터를 주입하고 가져와서 사용이 가능한지 테스트 하였다.

```
> Entering new AgentExecutor chain...
```

```
Invoking: `cusomter_service` with `{'query': '채무자 법'}`
```

```
1. 채무자회생법 제564조 제1항 제1호, 제658조, 제321조에 따르면, 채무자는 파산관재인·감사위원 또는 채권자집회(이하 '파산관재인 등'이라 한다)의 요청이 있으면
```

각각 collection에 다른 법에 대한 정보를 넣고 테스트를 진행하자 위와 같이 잘 나오는 것을 확인하였다.

3. 할루시네이션 제거

프로젝트를 진행하며 챗봇 구성에 집중하여 초기에는 프롬프트를 이용한 할루시네이션 설정과 커스텀이 이루어졌다. 이러한 방법을 사용하면 chroma db에 없는 결과의 경우 아예 없다고 말하고 찾아주지 않는 모습을 보였다.


```

Example:
User Input: "What meetings do I have tomorrow?"
RAG Retrieval: [project status update at 10 AM, client discussion at 3 PM]
Response: "Good morning! You have two meetings scheduled for tomorrow: the project status update at 10 AM and the client discussion at 3 PM.

Now respond to following User input, based on RAG Retrieval.
User input: {question},
RAG Retrieval: {data}
Response:

```

위 이미지는 설정한 프롬프트이다. RAG방식을 사용할 것임을 알리고 example로 친절하게 대답을 해주는 발화문을 작성하여 설정하였다. 이러한 방식으로 챗봇을 구동 할 시 할루시네이션이 제대로 제거 되는 것을 확인했다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

대화

You:


환경친화적 자동차의 개발 및 보급 촉진에 관한
법률에 대해 알려줘

챗봇:

죄송합니다. 해당 질문에 알맞은 답변을 찾을 수 없습니
다.

👍 🔄

이러한 방식을 사용하면 기업에서 설정한 서비스에 대한 설명만을 할 수 있을 것이라고 판단했다. 하지만 기업에서 챗봇의 챗 기능과 더불어 기능을 가져가고 싶으면 어떻게 하지? 라는 생각을 하게 되었다. 그래서 할루시네이션을 알려주되 없는 정보는 gpt모델 내에서 검색해서 가져오게 하는 것을 생각하고 두가지 버전을 준비했다. 위 사진의 방식의 경우 chroma db에 query로 cosin값이 가장 가까운 즉 거리가 가장 가까운 데이터 3개를 가져와서 프롬프트에 합성시키고 gpt에 보내는 방식으로 이루어졌다. 이렇게 진행하니 chroma db에 없을 경우 gpt에서도 결과를 검색해주지 않는 것을 보았는데 이러한 방식이 아니라 순서를 다르게 진행하는 것을 생각하고 테스트 하였다. 먼저 langchain의 chroma db 패키지를 불러왔다. 그리고 검색하기전 검색에 필요한 리트리버를 만들기 위해 기존의 리트리버 초기화시켜준다. 그 후 원하는 검색 기준으로 리트리버를 생성하여 tool로 저장해둔다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

```

langchain_chroma = Chroma(
    # PersistentClient를 전달합니다.
    client=chroma_client,
    # 사용할 컬렉션의 이름을 지정합니다.
    collection_name=name,
    # 임베딩 함수를 전달합니다.
    embedding_function=stf_embeddings,
)

retriever = langchain_chroma.as_retriever() # 리트리버 초기화
llm = ChatOpenAI(temperature=config_normal['TEMPERATURE'], # 창의성 (0.0 ~ 2.0)
                  max_tokens=config_normal['MAX_TOKENS'], # 최대 토큰수
                  model_name=config_normal['MODEL_NAME'], # 모델명
                  openai_api_key=OPENAI_API_KEY
                  )

tool = create_retriever_tool(
    retriever,
    "cusomter_service",
    "Searches and returns documents regarding the customer service guide.",
)

```

그 후 검색에 활용할 system 명령을 지정한다. 이 방식은 프롬프트를 생성하는 것과 동일한 기능을 한다. 이렇게 만든 prompt와 tools를 이용하여 agent로 만들고 chain설정으로 검색의 단계를 설정한다.

```


system_message = SystemMessage(
    content=(
        "You are a nice customer service agent."
        "Do your best to answer the questions."
        "Feel free to use any tools available to look up "
        "You provide a service that provides answers to questions"
        "relevant information, only if necessary"
        # 할루시네이션.
        "If you don't know the answer, just say that you don't know. Don't try to make up an answer."
        # 한국어
        "Make sure to answer in Korean."
    )
)

prompt = OpenAIFunctionsAgent.create_prompt(
    system_message=system_message,
)

# 위에서 만든 llm, tools, prompt를 바탕으로 에이전트 만들어주기
agent = OpenAIFunctionsAgent(llm=llm, tools=tools, prompt=prompt)

```

이러한 방식으로 챗봇을 제작하였다. 기업별로 다른 방식의 챗봇을 원할 수 있기에 api로 나누어 두개다 사용 가능하게 하였다. 아래는 챗봇의 대화 결과이다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

대화

You:

환경친화적 자동차의 개발 및 보급 촉진에 관한
법률에 대해 알려줘


챗봇:

해당 질문에 적합한 답변을 찾을수는 없지만 '국가 법령
정보 센터'에서 찾은 '환경친화적 자동차의 개발 및 보
급 촉진에 관한 법률'은 다음과 같습니다. 이 법은 환경
친화적 자동차의 개발 및 보급을 촉진하기 위한 종합적
인 계획 및 시책을 수립하여 추진하도록 함으로써 자동
차산업의 지속적인 발전과 국민 생활환경의 향상을 도
모하며 국가경제에 이바지함을 목적으로 한다.

📄 🔍

4. 여러 데이터 학습 방법

본 사이트에서 챗봇을 학습하기 위한 방법은 간단하되 여러가지 방식이어야 한다고 생각하고
기획하였다. 그래서 팀원들과 초기에 생각한 것이 pdf이다. pdf를 올리기만하면 학습이 되는
방식을 생각하였다. 그리하여 pdf를 업로드 하고 안의 text를 가져와서 학습에 이용하는 것으로
하였다. 초기에는 s3에 올려 주소값을 보내는 것으로 사용하려 했으나 s3의 비용 때문에 파일을
그냥 업로드 하는 것으로 결정하였다. 파일을 업로드 하고 업로드된 파일을 임의의 uuid
제목으로 저장한다. 그리고 저장된 제목을 다시 client에 반환하여 어떤 pdf가 저장되어있는지
알려준다. client에서는 학습 요청을 보내면 가지고 있는 pdf의 제목을 같이 보내주어
server에서 학습하게 하는 방식이다. 이러한 방식을 구상할 때 도중에 나갈경우 같이 학습을
시키지 않고 업로드 하면 파일이 남아 쌓이게 되는 문제가 발생하였다. 그래서 페이지에서
나갈때, 학습할때 에러가 발생했을때 가진 pdf들을 삭제하는 요청을 날리게 하여 해결하였다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

파일로 학습하기

파일을 선택해주세요 (PDF만 지원함)

파일 선택 선택된 파일 없음

제출

pdf를 이용한 학습을 완료하고 이러한 방식만 사용한다면 결국 학습할 데이터를 기업측에서 직접 제작하여 넣어야 한다는 문제점이 있다는 것을 알게 되었다. 기업만이 아닌 개인의 이용자가 사용할때도 고려한다면 더 쉬운 방법이 있어야 한다고 생각했다. 그리하여 구상한 것이 키워드와 url주소이다. url주소의 경우 기존의 학습 테스트 방식이었기에 쉽게 구현하였다. bs4를 이용한 크롤링 방식으로 보내진 url의 주소에 있는 데이터를 모두 가져오는 방식이다. 이러한 방식을 활용하여 키워드 데이터 추가도 구현하였다. 나무위키, 사전 같은 데이터를 얻어올 수 있는 사이트들을 선별하고 받아오는 키워드를 해당 사이트에서 데이터를 가져와서 학습하게 설정하였다.

주소로 학습하기

본인의 웹사이트 주소를 입력해주세요.

웹사이트 주소


제출

키워드로 학습하기

관련된 키워드를 추가 해주세요

+

제출

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

- Backend

여러 백엔드 프레임워크들이 있지만, chatbot 응답에 사용될 API와 REST API를 모놀리식으로 개발하기 위해서, Python 언어로 작성된 최신 웹 프레임워크인 FastAPI를 사용했다.

Backend 기능 구현에는 다음과 같다.

1) 이메일 회원가입/로그인 & google oauth2 회원가입/로그인


/auth/sign-in, /auth/sign-up API에 다음과 같은 requestbody를 줘서 type으로 email 로그인, google 로그인을 구별했다.

```
{
  "type": "string",
  "token": "string",
  "company_name": "string",
  "email": "string",
  "password": "string",
  "picture": "string",
  "name": "string"
}
```

2) 유저 정보 가져오기

/user/me 에 JWT 토큰 값을 보내면 현재 유저의 정보를 다음과 같이 반환해준다.

```
{
  "_id": "6644e28b30...",
  "type": "email",
  "email": "...@...com",
  "picture": null,
  "name": null,
  "company_name": "",
  "create_time": 1715790475,
  "client_id": "6A7DF7FDDFCA4...",
  "exp": 1718461871
}
```

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수릿현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

3) 유저 세션유지에 필요한 JWT 토큰 발급

유저가 로그인시에 JWT토큰을 발급하기위해 만료기간 `expires_delta`를 정해주고 프로젝트만의 `secret`로 암호화해서 유저에게 다시 보내준다. 유저는 request마다 JWT로 권한을 인증하게 된다.

```

async def create_jwt_token(data: dict, expires_delta: Union[timedelta, None] = None):
    data['_id'] = str(data['_id'])
    to_encode = data.copy()
    if expires_delta:
        expire = datetime.utcnow() + expires_delta
    else:
        expire = datetime.utcnow() + timedelta(minutes=15)
    to_encode.update({"exp": expire})
    encoded_jwt = jwt.encode(to_encode, setting.JWT_SECRET, algorithm=setting.ALGORITHM)
    return encoded_jwt

```

4) Request 마다 권한 유효성 체크

유저는 매번 Request를 보낼때 마다 Authorization에 JWT 토큰을 담아 보낸다. 서버는 이를 매번 확인하고 권한을 인증한다.

```

oauth2_scheme = OAuth2PasswordBearer(tokenUrl="token")

@user_router.get("/me")
async def get_user(token: str = Depends(oauth2_scheme)):
    user_dict = await service.get_user_in_token(token)
    return user_dict


```

5) Chatbot 요청에 필요한 API키 발급

회원가입 중에 chatbot 사용에 필요한 API를 발급한다.

6) 파일 업로드 & 파일 삭제

파일 업로드는 `multipart/form-data` 로 pdf를 입력받는다. 파일은 서버 디렉토리 안에 저장된다. chatbot 학습에 파일이 사용되고 나서는 클라이언트에서 삭제 요청이 들어오는데, 이때 포함된 파일 이름으로 서버에서 파일을 삭제한다.

 <div> 국민대학교 컴퓨터공학부 캡스톤 디자인 I </div>	결과보고서		
	프로젝트 명	чек봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

```

@file_router.post("/upload-pdf")
async def upload_pdf(file: UploadFile = File(...)):
    if file.content_type != "application/pdf":
        raise HTTPException(400, detail="Invalid document type")

    content = await file.read()
    filename = f"{str(uuid.uuid4())}.pdf" # uuid로 유니크한 파일명으로 변경
    with open(os.path.join(setting.UPLOAD_DIR, filename), "wb") as fp:
        fp.write(content) # 서버 로컬 스토리지에 이미지 저장 (쓰기)

    return {"filename": filename}

@file_router.post("/delete-pdf")
async def upload_pdf(filename: RemoveFileDto):
    file_path = os.path.join(setting.UPLOAD_DIR, filename.filename)
    if os.path.isfile(file_path):
        os.remove(file_path)
        return {"message": "remove {filename.filename}"}
    else:
        return {"message": "file not found"}

```

Client

1. 패키지화

기존 웹사이트의 디자인을 해치지 않고, 개발이 용이하도록 모든 디자인과 코드, 로직을 패키징하여 유저(기업)에게 제공한다. 패키지는 NPM에 배포되어 있으며, 누구나 다운로드 할 수 있도록 만들었다.

@checkbot/checkbot
1.6.0 • Public • Published 2 days ago

Readme
Code (Beta)
5 Dependencies
0 Dependents
29 Versions
Settings

This package does not have a README. Add a README to your package so that users know how to get started.

Keywords

none

Install

Repository
github.com/kookmin-sw/capstone-2024-...


Homepage
github.com/kookmin-sw/capstone-2024-...

Weekly Downloads
910

Version
1.6.0
License
ISC

Unpacked Size
191 kB
Total Files
20

Issues
0
Pull Requests
0

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

(<https://www.npmjs.com/package/@checkbot/checkbot>)

또한, 시멘틱 버저닝을 지원하여 버그 수정이나 신기능에 대해서도 빠르게 대처 가능 하도록 설계 하였다. 차후 major 버전이 업그레이드 될 시에는 따로 migration 가이드도 제공할 예정이다.

위 패키지는 다음방법을 통해 간단하게 설치 할 수 있으며, 웹사이트에서 직접 확인할 수 있다.

사용법

1. 라이브러리 설치하기

```
npm install @checkbot/checkbot
```

2. 챗봇 코드에서 사용하기 (Next.js 기준)

```
"use client";


import ChatBot from "@checkbot/checkbot";
import "@checkbot/checkbot/dist/index.css";

const WrappedChatBot = () => {
  return <ChatBot clientId={/*내 클라이언트 아이디*/} />;
};

export default WrappedChatBot;
```

클라이언트 아이디는 내 프로필에서 확인하세요.

클라이언트 아이디의 경우에는 내 프로필에서 확인 할 수 있다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

안녕하세요!

F

final@naver.com

내 클라이언트 ID

아래 클라이언트 ID를 복사 하여 사용하세요(사용법 참조)

8C824B46F8AC4500B4B4F9091F30DF7F

사용법 바로가기

책봇 리셋하기


2. 로그인 및 회원가입

유저가 사이트에 로그인 / 회원가입을 할 수 있도록 페이지를 만들어야 했다.

로그인

로그인 하기


아직 회원이 아니신가요? [회원가입하기](#)

 Sign in with Google


회원가입

회원가입 하기

이미 회원이신가요? [로그인하러가기](#)

 Sign in with Google

유저가 아이디 비밀번호를 입력하면, 서버에 정해진 규칙에 따라 필요한 요청을 보내 jwt 토큰을 받아 로그인할 수 있도록 했다. 처음에는 서버에 어떻게 요청을 보내야 하고, 어떤 형식으로 보내하는거지? 물어보면서 만드는 과정중에도 이게 맞나? 이렇게 하는 것이 맞나?


 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

생각하면서 만들었고, 어느정도 구현이 되었다고 생각해 서버에 연결을 시켜보려 했지만 바로 에러를 봐야만 했다. 다행히 클라이언트 문제가 아닌 서버쪽 문제여서 해결이 되고 난 후 연결을 해보니 연결이 잘 되는 것을 확인할 수 있었다. 이메일 비밀번호를 통한 회원가입 및 로그인을 구현한 후에 시간이 지나고, 구글 로그인을 구현하지 않았다는 것을 뒤늦게 깨닫고, 구글 로그인 관련으로 찾아보며 구현하려 했지만 처음에는 에러가 나고 실패 했었다. 곰곰이 생각해 보니 구글 로그인 버튼만 보고 로그인만 생각이 매몰되어 서버에 구글을 통한 로그인이 가능하려면 일단 그 유저 정보를 회원가입을 시켜줘야만 가능하다는 것을 뒤늦게 눈치채고, 올바른 순서대로 다시 구현하여 결국 성공했다.

로그인을 성공적으로 수행했다면, 그 로그인 상태가 유지가 되어야 했기에 React의 상태관리 라이브러리 중 하나인 Jotai를 사용해서 구현했다. 처음엔 Nextjs의 서버 컴포넌트와 클라이언트 컴포넌트의 사용을 혼동하여 구현하려 할 때마다 오류가 나고 제대로 잘 구현이 되지 않았었다. 클라이언트 컴포넌트와 서버 컴포넌트를 제대로 구분하여 구현하니 유저의 정보 관련한 부분을 storage에 잘 저장이 되어 로그인을 유지할 수 있도록 했다.

2.2.2 시스템 기능 요구사항

주요 기능	요구사항	완료 / 미완료
회원가입	유저는 이메일, 구글 회원가입을 할 수 있다.	완료
로그인	유저는 이메일, 구글 로그인을 할 수 있다.	완료
유저 세션 유지	JWT 인증 방식으로 유저의 세션을 유지한다.	완료
유저정보	서버로부터 유저정보를 가져올 수 있다.	완료
파일 업로드 및 삭제	데이터 학습에 필요한 파일을 업로드 및 삭제 할 수 있다.	완료
챗봇 API키 발급	유저가 회원가입함에 동시에 챗봇을 사용할 수 있는 API를 발급한다. 해당 API키를 통해 본인이 설정한 챗봇을 자사 홈페이지에 삽입하여 활용할 수 있다.	완료

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수릿현모	
	Confidential Restricted	Version 1.2	2024-MAY-24


챗봇	유저는 챗봇에게 질문을 할 수 있고, 답변을 들을 수 있다.	완료
챗봇 패키지화	유저는 챗봇을 패키지를 통해 설치하며, client_id 이외의 별도의 설정을 필요로 하지 않는다	완료

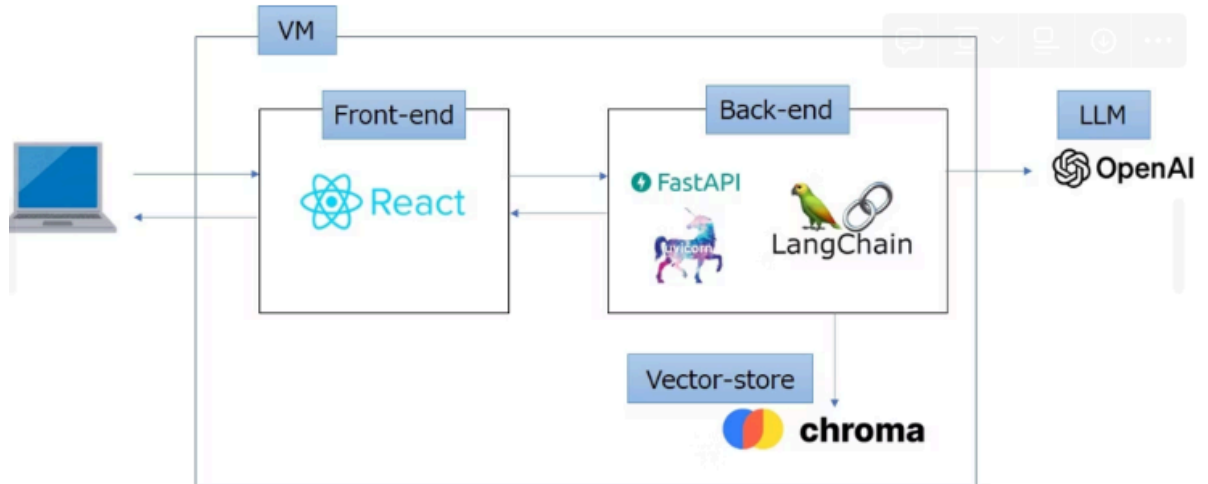
2.2.3 시스템 비기능(품질) 요구사항

항목	설명	달성여부
접근성	인터넷 연결 된 환경에서 언제나 웹 사이트 주소로 접근할 수 있다.	달성
적용성	챗봇 Package를 웹사이트에 손쉽게 적용 할 수 있다.	달성
신뢰성	데이터 학습에 필요한 파일을 업로드 및 삭제 할 수 있다.	달성
효율성	모든 API의 응답속도가 5초 이내여야 한다.	미달성
학습성	유저가 입력한 데이터를 올바르게 챗봇이 학습할 수 있어야 한다.	달성

2.2.4 시스템 구조 및 설계도

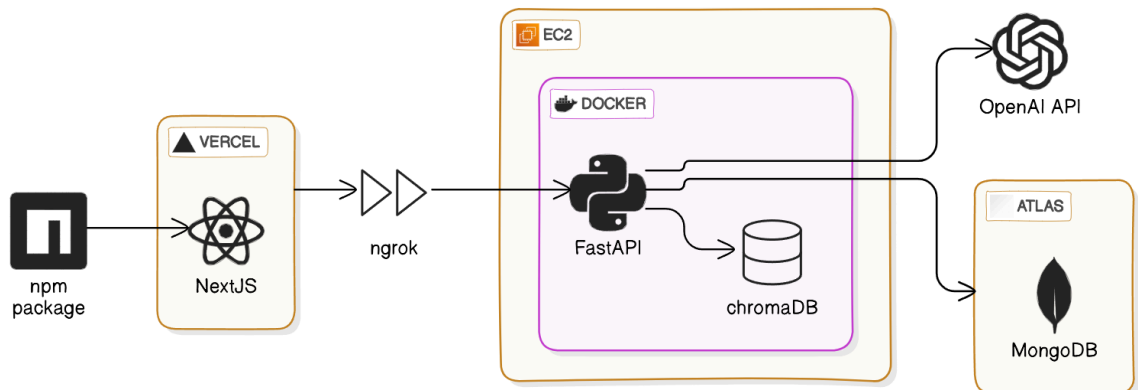
프로젝트 기획 단계에서는 크게 next.js, fastapi, langchain을 사용하는 것을 목표로 구조를 제작하였다. 처음의 구조는 아래와 같다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24




간단하게 ec2 내에 FE와 BE, Vector store를 한번에 올려 사용할 계획이었다. 하지만 그렇게 진행할 경우 ec2하나가 꺼지면 모든 서버가 마비되는 문제가 생겼다. 그리고 몽고 db를 ec2내에 배포하였는데 해킹이 당해 돈을 요구하는 문제도 발생했다. 이러한 문제점을 해결하기 위해 먼저 Fastapi, chroma db를 도커화 해서 환경을 일치시키고 mongo db의 경우 보안이 더 확실하고 좋은 Atlas cloud 서비스를 이용하였다. ec2 인스턴스에 도커화한 것들을 배포하고 외부의 atlas mongo를 연결했다. 그리고 openai api 주소를 fastapi에서 접근 허용 하게 하여 api를 활용하여 사용하였다. 하지만 이렇게 진행하다보니 ssl인증이 된 client와 ssl 인증이 안되서 http로 배포되고 있는 서버의 데이터 교환이 안되는 문제가 생겼다. 이러한 문제를 파악하고 아래와 같은 구조로 변경하였다.

Architecture



인증을 구현하기 위해 nginx를 사용하려 하였지만 DNS 도메인을 구매할 비용이 없어 비용없이 ssl 인증을 할 수 있는 방법을 찾아보았다. 그렇게 Ngrok을 사용해서 포트포워딩으로 ssl인증을 진행하였고 진행한 서버 주소에 api를 요청하는 방식으로 구현하였다. 하지만 ngrok의 단점은 일본서버에서 배포되기에 기존의 chatbot response time보다 더 오래걸리는 문제가 발생하였다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

이러한 점은 추후 비용을 추가하여 한국서버에서 동작하게 구매를 하거나 nginx를 사용하여 인증을 할 계획이다.

2.2.5 활용/개발된 기술

1) 프론트엔드


사용 기술	설명
Next.js	<ul style="list-style-type: none"> - Vercel이 개발한 오픈 소스 react 프레임워크 입니다. - 서버 사이드 렌더링을 통해 클라이언트에 주는 부담을 줄일 수 있습니다.
tailwindcss	<ul style="list-style-type: none"> - 유틸리티 우선 CSS 프레임워크 입니다. - 일관된 스타일로 홈페이지를 디자인 할 수 있었습니다.
React.js	<ul style="list-style-type: none"> - Facebook이 개발한 Javascript 라이브러리 입니다. - 웹 사이트 개발을 보다 쉽게 할 수 있습니다.
Jotai	<ul style="list-style-type: none"> - React를 위한 상태 관리 라이브러리 입니다. - 최소한의 API를 제공하여 React에서 상태 관리를 간단하게 만들 수 있습니다.

2) 백엔드

사용 기술	설명
FastAPI	<ul style="list-style-type: none"> - FastAPI는 Python으로 작성된 현대적인 웹 프레임워크입니다. - 빠른 성능, 타입 힌팅, 비동기 지원, 호환성이 장점이라 API서버로 사용하였습니다.
Pydantic	<ul style="list-style-type: none"> - FastAPI와 자연스럽게 통합되는 데이터 검증 및 설정 관리 라이브러리입니다. - 데이터 모델링과 유효성 검사에 사용했습니다.

3) 챗봇


사용 기술	설명
RAG	<ul style="list-style-type: none"> - 챗봇의 커스텀을 진행하기위해 많이 사용하는 fine tuning의 단점을 해결한 방법

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

	<ul style="list-style-type: none"> - 비용은 줄이고 범용성은 넓혀 좀더 다양한 커스텀 데이터를 넣고 활용할 수 있다.
Langchain	<ul style="list-style-type: none"> - open ai api를 사용할때 설정들을 주입시켜주기 위해 활용하였다. - prompt, agent, tools, chain등 챗봇의 동작을 세세하게 설정하고 할루시네이션 방지 설정등을 제작하였다.
Streaming	<ul style="list-style-type: none"> - 챗봇의 대화를 스트리밍 형식을 보내 사용자가 기다리는 시간을 덜 지루하게 만들기 위해 사용하였다. - streaming response라는 것을 사용하여 제작중에 있으며 현재 코루틴 오류로 인해 제대로 작동이 되지 않아 해결하고있는 중이다.
NPM	<ul style="list-style-type: none"> - Node Package Manager의 줄임말로 특정 컴포넌트나 모듈을 패키지와 하여 배포 할 수 있습니다. - 다른 사용자들은 간단한 명령어를 통해 해당 패키지를 설치할 수 있습니다. - 챗봇을 손쉽게 다른 사용자가 설치 할수 있게 만드는데 활용하였다.

4) 인프라

사용 기술	설명
Ngrok	client의 경우 vercel을 사용하여 ssl 인증이 자동으로 되어있으나 기존의 서버는 ssl인증이 되어있지 않아 서로 통신중 에러가 발생하였다. 이러한 오류는 http 사이트에서 https사이트로 데이터를 보내는 과정에서 일어난 에러였다. nginx를 사용한 ssl인증서 발급 방식을 이용하려 하였으나 domain 주소를 구입하지 않은 나머지 포트포워딩으로 해결하였다.
EC2	Aws 내에 존재하는 컴퓨팅 솔루션으로 Elastic Cloud Computing의 약자이다. 서버 인스턴스를 배포하여 Public Access가 가능하도록 하는데에 이용하였다.
Docker	chroma db, fastapi server등 개발자 간의 환경설정값을 맞추기 위해 docker를 이용하였다.
Git Action	Github 내부의 action기능을 활용하여 CD를 진행하였다. 기존의 경우 ec2인스턴스에 접속하여 git pull을 진행하고 docker image를 재 build 후 container를 생성하여 실행했어야 했는데 이러한 명령어를 yml형식으로 작성하고 git master에 merge요청이 올때마다 자동적으로 배포하게 설정하였다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수릿현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

5) 데이터베이스

사용 기술	설명
MongoDB	<ul style="list-style-type: none"> - MongoDB는 NoSQL 데이터베이스 중 하나로, Document-Oriented 데이터베이스 시스템입니다. - 대규모 데이터 처리에 강하고, JSON으로 손쉽게 저장 및 추출이 가능해서 사용했습니다.
Chroma DB	vector db의 한 종류로 학습할 데이터를 벡터화 시켜 저장하는 역할을 수행한다. 이렇게 저장한 벡터 정보들은 gpt모델로 검색할때 거리값으로 유사도를 검색 가능하여 유사 데이터를 가져올때 활용하였다.

2.2.6 현실적 제한 요소 및 그 해결 방안

1) 자사 웹사이트 학습


자사 웹사이트의 경우에는 학습이 안되어 답변을 하지 못하는 이슈가 있었다. 해당 정보를 알지 못하기에 지극히 당연한 일이었다. 우리 팀은 해당 문제를 해결하기 위해 3가지 방식으로 유저에게 챗봇 학습을 허용한다. 첫째, 링크로 학습하기. 웹사이트 링크를 제공하면 서버에서 해당 링크를 크롤링하여 학습한다. 둘째, 키워드로 학습하기. 특정 키워드들을 제공하면 해당 키워드에 최적화된 답변을 제공한다. 셋째, 파일로 학습하기. PDF 파일을 제공하면 포함된 텍스트를 추출하여 학습한다. 위 세가지 방식을 통해 유저는 챗봇을 사용자화 하여 정확한 답변을 고객들에게 제공할 수 있다.

2) 할루시네이션

생성형 AI가 정확하지 않거나 사실이 아닌 정보를 생성해서 그럴 듯하게 답변하는 하는 일이 많았고, 이를 해결해야만 사용자가 원하는 정보를 정확하게 제공할 수 있는 챗봇이 만들어진다. 우리는 Chroma db 와 RAG 방식을 통해서 이를 해결하고자 했고, 할루시네이션이 된 답변을 방지했다.

3) 기획단계에서의 유사한 서비스 발견

AI 시장이 발달함에 따라 자연스럽게 AI 챗봇 서비스를 찾는 사람들이 많아졌고, 그 수요에 맞춰 챗봇을 제공하도록 하는 서비스들이 많이 생기게 되었다. 기존 서비스들과의 차이점을 두어야 했었고 우리는 최대한 간단하고 빠르게 만들 수 있는 방안으로 생각하게 되었다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	체크봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

2.2.7 결과물 목록

번호	결과물	기술문서 유/무
1	웹사이트 (https://capstone-2024-37.vercel.app/)	무
2	API 서버	무
3	챗봇 패키지 (https://www.npmjs.com/package/@checkbot/checkbot)	무

2.3 기대효과 및 활용방안

1) 기업의 사용자 응대 (CS)


자사의 서비스가 복잡해지고 인건비가 상승함에 따라 장기적으로 챗봇은 사용자 응대 관련 업무를 대체 혹은 효율화 할 수 있다. 기제작된 챗봇은 웹사이트 및 사내 서비스 및 다양한 프로젝트에 대해 학습을 통해 상황에 알맞은 답변을 제공하기에, 불필요한 응대 리소스와 시간 모두 절약 가능하다. 장기적으로 챗봇이 고객응대를 대체하는 날이 올것이라고 조심스레 짐작해본다.

2) 나만의 웹서핑 도구


웹사이트 내에 수많은 하이퍼 링크와 참조들이 존재한다. 현대 사회의 유저들은 이를 잘 읽으려 하지도, 읽을 필요도 없다. 그저 웹사이트 우측하단에 존재하는 체크봇 활성화 버튼을 클릭하여 해당 웹사이트의 대한 질문을 하면 그만이다. 국민대학교 소프트웨어학부의 위치와 담당자의 성함과 전화번호를 일일이 찾는데 시간을 허비하지 않아도 된다. 그저 질문하라.

3 자기평가

팀원명	평가
구형모	최근 대두되고 있는 GPT의 원천기술을 활용하여 부가가치를

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24


	<p>창출 하고 싶었던 니즈가 있었고, 이번 프로젝트를 통해 해당 니즈를 여러 BM과 아이디어를 통해 실현시켜본 값진 경험이었다.</p> <p>처음 아이디어를 제시하고 기획을 직접하며 유저 플로우에 대해 많은 고민을 하게 되었다.</p> <p>인프라를 설계하고 팀원들과 솔루션에 대해 토론하고 적용하여 가용성있는 클라우드 설계에 대해 직접 만들어 보는 좋은 계기가 되었다. 다만 비용 문제로 인해 로드밸런서와 실제 DNS 도메인을 통해 테스트 해보지 못한점은 아쉬웠지만, ngrok이라는 솔루션을 직접 제시하여 외부에서 서버에 접속할 수 있도록 하였다.</p> <p>학부에서 클라우드, 네트워크, 알고리즘, 웹 컴퓨팅등을 배운것을 토대로 발전시켜 실제 작동하는 프로젝트를 만들어볼 좋은 기회라고 생각하여 최선을 다하였다.</p>
고강현	<p>이전 GPT가 대두되기 전 kobert, koelectra등 NLP모델을 파인튜닝하고 데이터를 전처리해가며 모델을 제작했던 경험이 있었다. 그 당시에는 GPT만큼의 성능이 나오지 않았기에 이렇게 챗봇 서비스를 개발할줄은 몰랐다. Langchain을 활용하여 GPT를 활용해가며 좀 더 빠르게 정확하게 나오기 위해 열심히 노력해보는 값진 경험이었다. 다만 아쉬운 점은 코루틴 문제로 아직 streaming chat 기능을 활성화 하지 못했다는 점이다.</p> <p>그리고 이번 개발을 진행하면서 개발자 간의 환경 셋팅이 큰 문제로 계속 다가왔다. 이러한 점을 해결하기 위해 도커화를 진행하여 해결했지만 젠킨스 같은 자동 CI/CD툴을 활용하지는 않았기에 Git Action기능을 처음 사용해가며 환경 셋팅을 했다. Ngnix를 이용한 무중단 배포도 진행하려했으나 시간상의 문제로 못한것이 아쉽다.</p> <p>백엔드 디비의 인프라를 구축해가며 해킹의 문제가 있다는 것을 알게 되었고 cloud 서비스를 이용하고 연결하는 경험을 할 수 있어서 좋았다. 인프라를 구축해본 경험이 처음은 아니지만 새로운 cloud 서비스, 새로운 경험들을 해가며 많은것을 배울 수 있었다. 아쉬운 것은 DNS 도메인을 구매하여 ssl 인증을 직접 하려 하였지만 비용 문제로 ngrok으로 대체한 것과 수업시간에 배운 아키텍처 스타일을 적용시켜보지 못한것이 조금 아쉬울 뿐이다. 그래도 이번 프로젝트로 기획, 인프라, 백엔드, ai 개발을 진행하며 모르는 것이 있다면 배워가며 최선을 다해 프로젝트에 임하였다.</p>
곽다윗	<p>어떤 주제로 프로젝트를 진행할까, 프로젝트 시작 당시에 팀내 많은 고민이 있었다. 기획 부분에서 생각보다 많은 시간을 잡아먹었는데 결국에는 IT 개발에서 가장 핫한 LLM과 관련된 프로젝트를 하기로 했고, 최신 기술을 기존에 다른 해결방법이 있던 도메인에 적용할 때에 새로운 가치를 만들어내는 것을 보고 많은 것을 느꼈다.</p>

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윳현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

	<p>개인적으로는 자바 백엔드와 여러 파이썬 프로젝트를 전에 진행했었는데, 이번에는 파이썬 REST API를 전체적으로 경험해볼 수 있어서 좋았다.</p> <p>처음에 프로젝트에 대해 서로의 인식이 달랐다. 그래서 어떤 이슈에 대해서 얘기 할 때에 여러 번 설명을 해야되는 경우가 많았다. 그런데 팀 내에 서로 이해할 수 있는 공통 언어가 만들어지자 개발의 속도가 빠르게 증가하는 것을 보고 의사소통의 중요성도 다시금 느끼게 되었다.</p>
김은수	<p>ChatGPT가 유명해지면서 생성형 AI가 할 수 있는 것이 무궁무진하겠구나 생각을 하고 있었다. 소프트웨어융합최신기술을 들으면서 AI관련으로 지식을 쌓을 무렵 Langchain에 대해서 알게 되었고 언젠가 한번 저걸 써먹으면서 서비스를 한번 만들어 보고 싶다는 생각을 하게 되었다. 이번 프로젝트를 하면서 Langchain을 사용하면서 서비스를 만들면 이런 느낌이겠구나 경험하면서 생각보다 결과물은 잘나오는데 우리가 원하는 최고점의 이상향에 비하면은 많이 부족하다는 것을 보고, 이걸 잘 건드리면서 조율하는 것이 앞으로의 서비스의 방향성이라고 생각했다.</p> <p>이런식으로 프로젝트를 한 경험이 많지 않아서 사람들 사이의 의견 조율과 서로 각자가 필요한게 무엇인지 맞춰가면서 프로젝트를 해나가는 경험을 얻을 수 있어서 좋았다. 흔히 웹사이트에서 가장 첫번째로 하는 일이 회원가입 및 로그인이었었는데 어떤식으로 동작하는지 찾아보지 않았었다. 이번 프로젝트를 통해서 홈페이지는 이런식으로 만들어지고 서비스는 이런 방식을 통해서 만들어지며 아직까진 많은 공부가 필요하고 앞으로도 계속 공부해나가야겠구나 생각하게 되었다.</p>

4 참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
1	서적	우아한 타입스크립트 with react		2023년	우아한형제들	

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	챗봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

2	서적	실전에서 바로 쓰는 Next.js		2023년	미셀 리바	
3	기술 문서	Nextjs Docs	출처	N/A	Vercel	
4	서적	<랭체인LangChain 노트> - LangChain 한국어 튜토리얼kr	출처	2024년	테디노트	
5	기술 문서	Chroma Docs	출처	N/A	Chroma	
6	기술 문서	Jotai Docs	출처	N/A	Daishi	
7	기술 문서	FastAPI Docs	출처	N/A	FastAPI	

5 부록

5.1 사용자 매뉴얼

1) 챗봇 사용하기

사용법

1. 라이브러리 설치하기

```
npm install @checkbot/checkbot
```

2. 챗봇 코드에서 사용하기 (Next.js 기준)

```
"use client";

import ChatBot from "@checkbot/checkbot";
import "@checkbot/checkbot/dist/index.css";


const WrappedChatBot = () => {
  return <ChatBot clientId={/*내 클라이언트 아이디*/} />;
};
```

```
export default WrappedChatBot;
```

클라이언트 아이디는 내 프로필에서 확인하세요.

2) 챗봇 사용자화 하기

- 웹사이트 접속(<https://capstone-2024-37.vercel.app/>)
- 회원가입 및 로그인

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	결과보고서		
	프로젝트 명	책봇 (기업 맞춤형 웹사이트비서)	
	팀 명	수윗현모	
	Confidential Restricted	Version 1.2	2024-MAY-24

- c) 마이페이지 접속(상단 헤더 > 봇 수정하기 클릭)
- d) 링크, 키워드, PDF 제공

5.2 배포 가이드

- 1) 프론트엔드
 - a) git clone <https://github.com/kookmin-sw/capstone-2024-37>
 - b) npm install (패키지 설치)
 - c) next build (빌드)
 - d) next run (배포)
- 2) 서버

1. git clone

```
git clone https://github.com/kookmin-sw/capstone-2024-37.git
```

2. .env setting

```
cd capstone-2024-37/server/b2b
```

위 경로에 .env파일을 생성한다.

3. server start

```
cd capstone-2024-37/server
docker build -t {image name} .
docker run --name {image name} -d --rm -p 9000:9000 {container name}
```

4. client start

```
cd capstone-2024-37/server
yarn install
yarn start or yarn starts
```

5. 접속 및 사용

```
localhost:3000
```

5.3 테스트 케이스

대분류	소분류	기능	테스트 방법	기대 결과	테스트 결과
챗봇	디자인	디자인	유저 매뉴얼내 [챗봇 사용하기] 참조	웹사이트우측 하단에 표시 된다. 클릭 하면 오버레이가 뜬다	성공
	챗봇 답변	올바른 답변을 제공한다	[챗봇 실행] > [새 대화] 질문 입력	상황에 유저가 납득 가능한 올바른 답변을 한다	성공
		제공되지 않은 정보에 대해 답변을 하지 않는다	[챗봇 실행] > [새 대화] 학습되지 않은 정보에 대한 질문 입력	답변을 할 수 없는 내용이라고 답변한다.	성공