최종발표

Team 사다리

20171234 하재민 - 20171057 정신우 - 20170191 김성현

Content

- 프로젝트 개요
- 결과물 시연
- 아키텍쳐
- 미구현 요소
- 추후 개선 과제

프로젝트 개요

Chat Kiosk

Chat-GPT 를 활용한 인공지능 Kiosk 제작

프로젝트 개요

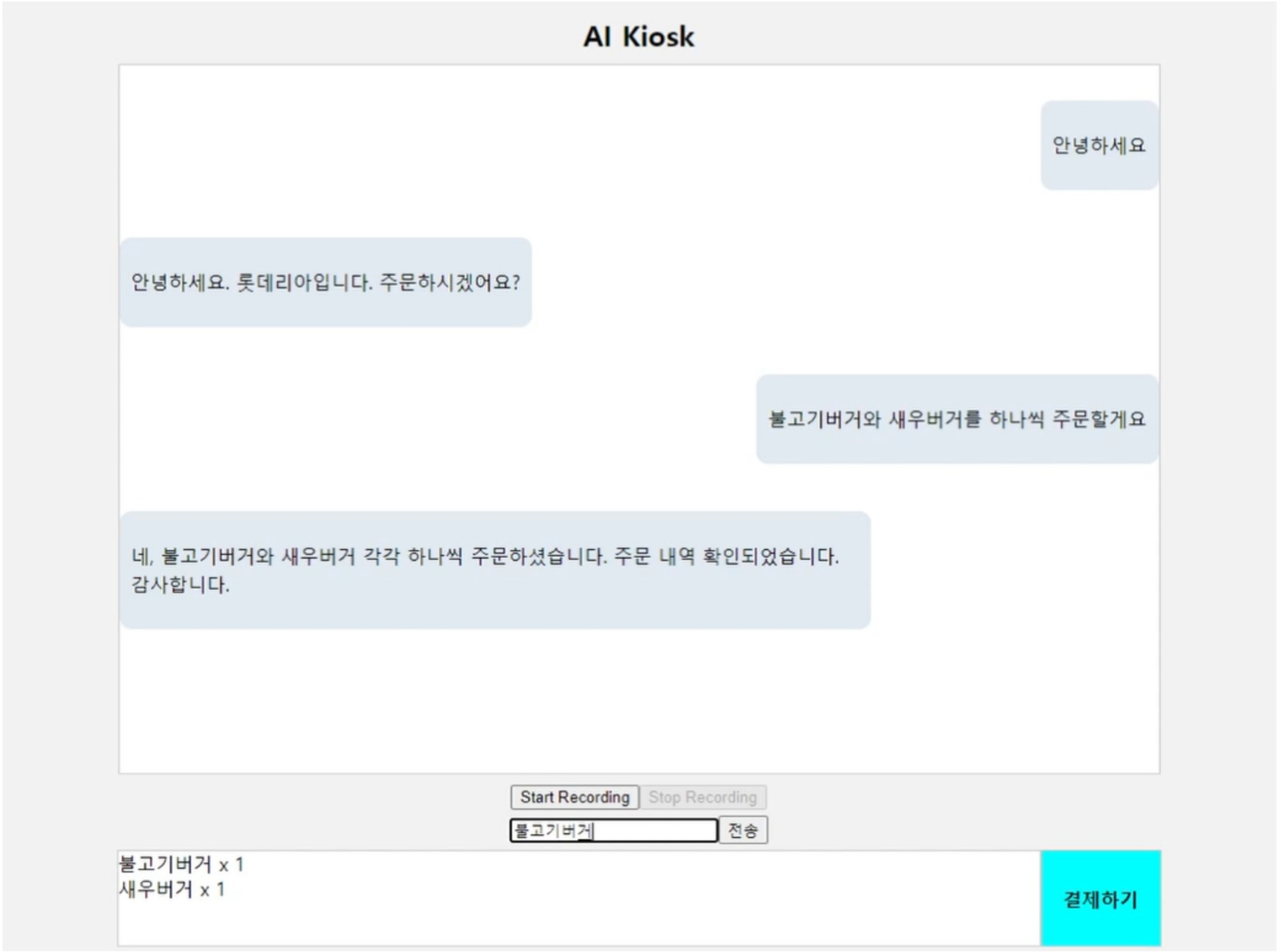
Open AI사의
ChatGPT를 중심으로
고성능의 LLM을 저비용에
이용가능하게 되었다.

프로젝트 개요


낮은 비용으로 고성능의
무인 포스기 솔루션
정보 취약층
접근성 상승
악성 고객
대응 가능
운영비 절감
인간 친화적
키오스크
직원 심리적
스트레스 감소

개발물 시연

결과물 시연



채팅을 이용한
GPT와의 대화를
통해 주문

아키텍쳐

Web Brower

Client

React

Ubuntu

Express Server

REST API

Completion

STT Module

TTS Module

Amazon EC2

Amazon RDS

주요 프로젝트 구조

API 서버 설계

- GET /menu/:id
- POST /completion/new-session
- **POST /completion**
- **POST /voice/stt**
- **POST /voice/tts**

아키텍쳐

POST /Completion

- 서비스의 가장 중요한 로직
- openai의 api로 gpt-turbo-3.5모델 사용
- 사용자의 요청에 대한 적절한 대답을 생성하여 응답
- 각 주문의 정보를 서버 세션DB에 유지
    - POST /completion/new-session
- 자연어 응답과 로직 구현 용 명령어를 응답함

아키텍쳐

POST /Completion

open ai의 api는
입력 prompt와
응답 content를 토큰화해
토큰당 가격을 책정한다.
입력 프롬프트를 잘짜서 최소한의
토큰으로 좋은 응답을 만들어낸다.

아키텍쳐

POST /Completion

- 1회 생성요청시 토큰한도 존재

- 자연어로 코드를 조작해야함

- 목표와 상반되는 구현방향

-> 프롬프트 엔지니어링

아키텍쳐

POST /Completion 토큰 최소화 전략

OrderManager

- shop_id : String
- shop_name : String
- state : OrderState
- step : Int
- dialogue : Array<Chat>
- menu : Array<Item>
- orders:
- requested_command : Array<Command>
- total_token : int

*server Session DB에 저장됨

각 session은 orderManager의 필드로 Prompt를 동적으로 구성한다.

Fields

아키텍쳐

POST /Completion 토큰 최소화 전략

- 장문의 Prompt는 토큰사용량이 높고 Prompt의 신뢰성이 떨어진다.

- 자연어 응답 생성과 명령어 추출을 별개의 Prompt를 사용

extractCommand

ceateReply

아키텍쳐

POST /Completion 토큰 최소화 전략

```
Convert context to command.
Command declare follow name list [add, rm, ask]
Each command must start with a command name in name list.
Command can be mutiple. Separate each command with a "/"
If second arg is exist, the arg declare follow list [${menu}]
If the second argument is a similar one from the menu, use the id of the similar one.
When you convert, follow example
    if context is about adding items from orderlist -> add id count
    if context is about removing items from orderlist -> rm id count
    if context is asking for agreement-> ask
        Command "ask" cannot exist alone.

    other context -> -
    `
```

```
Role : You're ${"롯데리아"} restaurant order assistant.
Situation : taking an order
OrderState : ${orderManager.state}
Menu : {${JSON.stringify(orderManager.menu)}}.
Rule :
   1. Never reply unrelated to the order.
   2. You're job is just take order, NOT SERVE ITEM.
   3. Talk only korean.
   4. You NEVER contain menu list in reply.
   5. as short as possible.
   6. You must check the menu
   7. Don't ask if there is anything else to order.
   ${manual}
```

해당 prompt로 주문 처리 시 평균 10000 토큰 사용.
gpt-3.5-turbo는 $0.002/1k tokens, 회당 $0.02 사용

아키텍쳐

POST /Completion 로직

1. 사용자 자연어 입력
2. 사용자 자연어에서 명령어 추출
3. 사용자 자연어에 대한 응답 생성
4. 응답에서 동작을 명령어로 추출
5. 명령어 실행으로 주문정보 갱신
6. 응답과 실행한 명령어 결과 반환

2번의 경우 AI로 생성된 응답은 요청을 되묻기도 하기때문에
사용자의 응답에 대해 긍정/부정 명령어를 추출

아키텍쳐

POST /Completion 로직

사용자 입력의 명령어

INFO :
- 제품 정보 요구
- 응답 생성 prompt에 필요한 정보 추가

YES :
- 지난 step의 명령어 실행 동의

NO :
- 지난 step의 명령어 실행 거부

생성된 응답의 명령어

ADD :
- 주문목록에 제품 추가

RM :
- 주문목록에 제품 제거

STATE :
- 주문 진행을 위한 상태 변경요청

ASK :
- 현재 step의 명령어 실행을 보류하고 다음 step에서 사용자 허락 요구

아키텍쳐

POST /Completion

```
{
    step: Int
    current_state : OrderState,
    total_token : int
    result : [
        {
            reply : String,
            commands : [...Command]
        },
        ...
    ]
}
```

응답예시

아키텍쳐

POST /voice/stt 및 /voice/tts

- 음성파일을 String으로 변환하여 응답
- GoogleCloud Speech 사용
- Python 모듈로 제작
- 요청시 자식프로세스로 Python 코드 실행
- 표준입출력으로 프로세스간 바이너리 데이터 입출력

아키텍쳐

API

- 두개의 프로젝트 생성
- 각 프로젝트에서 필요한 API key 발급

이름	ID
STTS	stts-385607
TTSTest	ttstest-387513

STT

TTS

아키텍쳐

API

음성 요청

Speech-to-Text에는 음성 인식을 수행하는 세 가지 주요 방법이 있습니다. 이들 방법은 다음과 같습니다.

- **동기 인식**(REST, gRPC)은 오디오 데이터를 Speech-to-Text API로 보내고, 해당 데이터를 인식하고, 모든 오디오가 처리된 후 결과를 확인합니다. 동기식 인식 요청 대상은 길이가 1분 이하인 오디오 데이터로 제한됩니다.
- **비동기 인식**(REST 및 gRPC)은 오디오 데이터를 Speech-to-Text API로 보내고, *장기 실행 작업*을 시작합니다. 이 작업을 사용하여 주기적으로 인식 결과를 폴링할 수 있습니다. 최대 480분 길이의 오디오 데이터에는 비동기식 요청을 사용하세요.
- **스트리밍 인식**(gRPC만 해당)은 gRPC 양방향 스트림에 제공되는 오디오 데이터를 인식합니다. 스트리밍 요청은 마이크에서 라이브 오디오 캡처 용도와 같은 실시간 인식 용도로 설계되었습니다. 스트리밍 인식은 오디오 캡처 중에 중간 결과를 제공하므로, 사용자가 계속 말하는 중에도 결과를 표시할 수 있습니다.

요청에는 오디오 데이터는 물론 구성 매개변수도 포함됩니다. 다음 섹션은 이러한 유형의 인식 요청, 요청이 생성하는 응답, 해당 응답 처리 방법을 자세하게 설명합니다.

스트리밍 인식은 트래픽과 비용 제한

아키텍쳐

API

권장사항

이 문서에는 Speech-to-Text API에 음성 데이터를 제공하는 방법에 대
응답 시간을 합리적으로 개선하고 효율성과 정확성을 높이기 위해 마련
매개변수 내에 있을 때 Speech-to-Text API를 가장 효율적으로 사용할

이 가이드라인을 따라도 API에서 기대하는 결과를 얻지 못하면 문제해

최적의 결과를 얻는 방법	가능하면 피해야 할
16,000Hz 이상의 샘플링 레이트로 오디오를 캡처합니다.	샘플링 레이트가 낮 예를 들어, 전화 통신 8,000Hz입니다.
무손실 코덱을 사용하여 오디오를 녹음하고 전송합니다. FLAC 또는 LINEAR16을 사용하는 것이 좋습니다.	녹음 또는 전송 시 n 확도가 떨어질 수 있 으면 무손실 FLAC 존을 위해 손실 코덱 WITH_HEADER_B
인식기는 추가적인 주변 소음 제거 없이 백그라운드 음성과 노이즈를 무시하도록 설계되었습니다. 하지만 최적의 결과를 얻기 위해 특히 백그라운드 노이즈가 있는 경우 마이크를 최대한 사용자에게 가깝게 배치합니다.	특히 손실 코덱을 사 릴 수 있습니다.
한 명 이상에서 오디오를 캡처하고 각 사람을 개별 채널에 녹음하는 경우, 각 채널을 개별적으로 보내면 최상의 인식 결과를 얻을 수 있습니다. 하지만 모든 스피커가 단일 채널 녹음에 혼합된 경우, 녹음을 있는 그대로 보냅니다.	여러 사람이 동시에 해석되어 무시될 수
단어와 구문 힌트를 사용하여 이름과 용어를 어휘에 추가하고 특정 단어와 구문의 정확도를 높입니다.	인식기에는 많은 수 다.
짧은 쿼리 또는 명령어의 경우 `single_utterance`를 true로 설정한 채로 `StreamingRecognize`를 사용합니다. 이렇게 하면 짧은 발화에 대한 인식이 최적화되고 지연 시간이 최소화됩니다.	짧은 쿼리 또는 명령 지 않는 것이 좋습니

모든 Speech-to-Text API 동기 인식 요청에는 음성 인식 `config` 필드(RecognitionConfig 유형)가 포함되어야 합니다. `RecognitionConfig`에는 다음과 같은 하위 필드가 있습니다.

- `encoding` - (필수) 제공된 오디오의 인코딩 체계를 `AudioEncoding` 유형으로 지정합니다. 코덱을 선택할 수 있는 경우, 최상의 성능을 원한다면 `FLAC` 또는 `LINEAR16`과 같은 무손실 인코딩을 선택하는 것이 좋습니다. (자세한 내용은 오디오 인코딩을 참조하세요.) 파일 헤더에 인코딩이 포함된 `FLAC` 및 `WAV` 파일의 경우 `encoding` 필드는 필수가 아닌 선택사항입니다.
- `sampleRateHertz` - (필수) 제공된 오디오의 샘플링 레이트(Hz)를 지정합니다. (샘플링 레이트에 대한 자세한 내용은 아래의 샘플링 레이트를 참조하세요.) 파일 헤더에 샘플링 레이트가 포함된 `FLAC` 및 `WAV` 파일의 경우 `sampleRateHertz` 필드는 필수가 아닌 선택사항입니다.
- `languageCode` - (필수) 제공된 오디오의 음성 인식에 사용할 언어와 리전/지역을 포함합니다. 언어 코드는 BCP-47 식별자여야 합니다. 언어 코드는 일반적으로 언어를 나타내는 기본 언어 태그와 보조 리전 하위 태그로 구성됩니다(예: 위의 예에서 'en'은 영어를, 'US'는 미국을 나타냄). (지원되는 언어 목록은 지원되는 언어를 참조하세요.)
- `maxAlternatives` - (선택사항, 기본값은 `1`) 응답에서 제공할 대체 변환 텍스트 수를 나타냅니다. 기본적으로 Speech-to-Text API는 기본 텍스트 변환 한 개를 제공합니다. 다른 대체 변환 텍스트를 평가하려면 `maxAlternatives`를 더 높은 값으로 설정합니다. 인식자가 대체 변환 텍스트 품질이 적절하다고 판단하면 Speech-to-Text는 대체 변환 텍스트만 반환합니다. 일반적으로 대체 변환 텍스트는 사용자 의견이 필요한 실시간 요청(예: 음성 명령)에 더 적합하므로 스트리밍 인식 요청에 알맞습니다.
- `profanityFilter` - (선택사항) 모욕적인 단어 또는 구문 필터링 여부를 나타냅니다. 필터링된 단어는 첫 번째 문자와 별표(나머지 문자)로 표시됩니다(예: f***). 욕설 필터는 단일 단어에 적용되지만 구문 또는 단어 조합으로 된 욕설이나 공격적인 음성을 감지하지 않습니다.
- `speechContext` - (선택사항) 이 오디오를 처리하는 데 필요한 추가 문맥 정보를 포함합니다. 컨텍스트에는 다음과 같은 하위 필드가 있습니다.
  - `boost` - 지정된 단어나 구문을 인식하도록 가중치를 할당하는 값을 포함합니다.
  - `phrases` - 음성 인식 작업의 힌트를 제공하는 단어와 구문의 목록을 포함합니다. 자세한 내용은 음성 적응의 정보를 참조하세요.

아키텍쳐

API

STT Module

```
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "google-cloud-sdk/stts-key.json"
```

API key 파일 환경 변수 지정

```
config = speech.RecognitionConfig(
    encoding=speech.RecognitionConfig.AudioEncoding.LINEAR16,
    sample_rate_hertz=sample_rate_hertz,
    language_code="ko-KR",
    audio_channel_count=2,
)
```

encoding, sampleRateHertz, languageCode 지정

```
response = client.recognize(config = config, audio = audio)

for trans in response.results:
    result = {"string": trans.alternatives[0].transcript}
    print(result, end='')
```

표준 입출력으로 서버에 응답

아키텍쳐

API

서버 통신 문제 해결

서버에서 다루는 인코딩 포멧
지원되는 경우는 베타 버전

```
audseg = AudioSegment.from_mp3(RECORDFILE)
audseg.export(TRANSFILE, format="wav")
```

파일 포멧 변환
+ 헤더에 LINEAR16 인코딩 포함

지원되는 오디오 인코딩

Speech-to-Text API는 다양한 인코딩을 지원합니다. 아래 표에는 지원되는 오디오 코덱이 나열되어 있습니다.

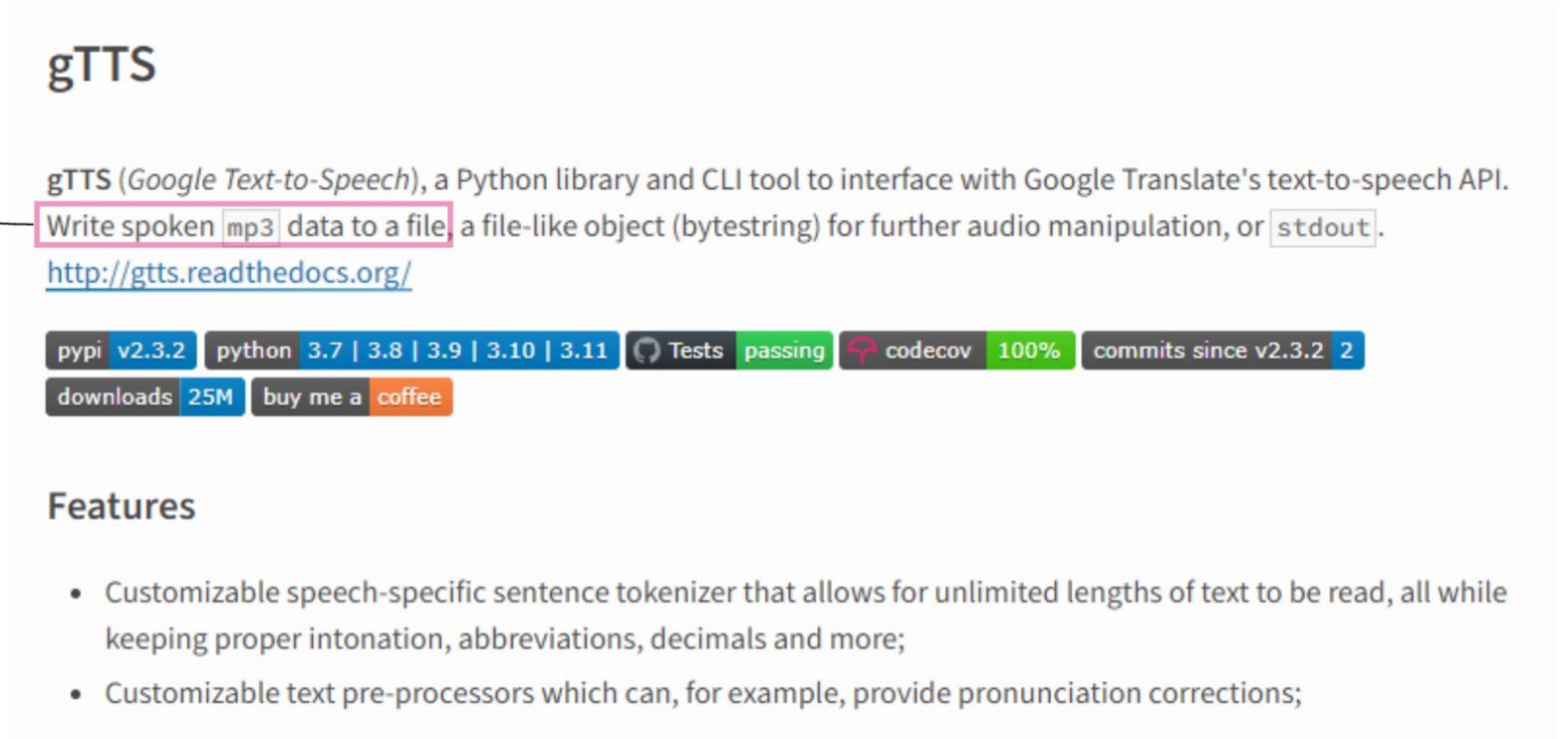
코덱	이름	무손실	사용 참고사항
MP3	MPEG 오디오 레이어 III	아니요	MP3 인코딩은 베타 기능이며 v1p1beta1에서만 사용할 수 있습니다. 자세한 내용은 RecognitionConfig 참조 문서를 확인하세요.
FLAC	무료 무손실 오디오 코덱	예	스트림에 16비트 또는 24비트 필요
LINEAR16	리니어 PCM	예	16비트 선형 펄스 코드 변조(PCM) 인코딩 헤더에는 샘플링 레이트가 포함되어야 합니다.
MULAW	μ-law	아니요	8비트 PCM 인코딩
AMR	AMR-NB(Adaptive Multi-Rate Narrowband)	아니요	샘플링 레이트가 8000Hz여야 함
AMR_WB	AMR-WB(Adaptive Multi-Rate Wideband)	아니요	샘플링 레이트가 16000Hz여야 함
OGG_OPUS	Ogg 컨테이너의 Opus 인코딩된 오디오 프레임	아니요	샘플링 레이트가 8000Hz, 12000Hz, 16000Hz, 24000Hz, 48000Hz 중 하나여야 함
SPEEX_WITH_HEADER_BYTE	Speex 광대역	아니요	샘플링 레이트가 16000Hz여야 함
WEBM_OPUS	WebM Opus	아니요	샘플링 레이트가 8000Hz, 12000Hz, 16000Hz, 24000Hz, 48000Hz 중 하나여야 함

아키텍쳐

TTS Module

API

기본: MP3 파일로 저장



Google Translates' text-to-speech API 사용을 위한 인터페이스를 제공하는 파이썬 라이브러리

해당 모듈 활용하여 TTS 모듈 구현, STT와 달리 파일 입출력으로 서버 통신

```
tts = gTTS(string, lang="ko")
tts.save('{0}'.format(time.strftime('../temp/%Y-%m-%d-%H%M%S.mp3', time.localtime(time.time()))))
```

미구현 요소

- **제작된 음성 API를 클라이언트에 적용실패**
    - 데이터 인코딩 문제로 음성 클라이언트에 적용 실패
- **관리자 페이지 미구현**
    - 개발우선순위에 밀려 미구현
- **배포단계에서 보안문제로 로컬환경에 머무름**
    - 세션에 필요한 쿠키의 보안정책으로 문제발생
    - 이로인한 개발지연
- **클라이언트 추가기능 미적용**
    - REST API로 반환되는 명령어 일부를 미적용

추후 개선 과제

자체 LLM을 사용

- 하드웨어 자원 소요와 학습 데이터 소요로 인하여 프로젝트에서 배제
- 직접 GPT로 구현해본 결과, GPT 파인튜닝하여 사용하면 가성비가 떨어짐
- 프롬프트만으로는 GPT의 할루시네이션으로 불안정
- 오픈소스로 배포된 LLM을 파인튜닝하여 사용하는 것이 궁극적 방향으로 판단됨

Thank You