Open Source SW Contribution

Architecture Design

제출일	2023.04.19	담당교수	송인식 교수님
과목	오픈소스 SW 기여	프로젝트명	Dr.Bot
학번 / 이름			
32197256 박성우		32173575 이해주	

1. 사용기술

1.1. 프로트엔드

프론트엔드로 사용할 기술은 Xcode 를 이용할 것이다. Xcode 는 애플의 통합 개발 환경(IDE)로 iOS 를 기반으로 한 앱을 구현할 것이기 때문에 Xcode 개발환경, 언어로는 Swift 를 사용할 계획이다.

• Xcode : Xcode 는 기본적으로 객체지향 프로그래밍 언어로 작성된 클래스의 내용을 시각화 하거나 검색 등을 할 수 있는 기능인 문서 브라우저, 클래스 브라우저가 있으며 StoryBoard, PreView 등 인터페이스 빌더와 연계가 매우 매끄럽다. 또한 Git 과 연동이 가능하여 관리가 편리하다는 큰 장점을 가지고 있다. 더불어 디버그 네비게이터, 디버그 메모리 그래프 등을 이용해 모니터링과 디버그가 가능하다. 이러한 점에서 해당 IDE 를 채택했다.

• Swift : 스위프트란 iOS, macOS, watchOS, tvOS 를 개발하기 위해 애플에서 제공하는 직관적인 프로그래밍 언어로 기존의 애플 운영체제용 언어인 Object-C 와 함께 공존할 목적으로 만들어 졌으며, 문법은 파이썬 언어라고 초창기 발표 때 알려졌다.

그간 발전된 모든 프로그래밍 언어를 참고하여 사용하기 편하고 보기 좋은 문법을 구사하려고 노력하였으며 개발자들이 원하던 현대적이고 세련된 문법을 구현했다. 또한 스위프트는 다중 프로그래밍 패러다임을 채용한 다중프로그래밍 언어이다.

애플이 공개한 문서에 의하면 스위프트의 특징은 크게 3 가지로 나뉜다.

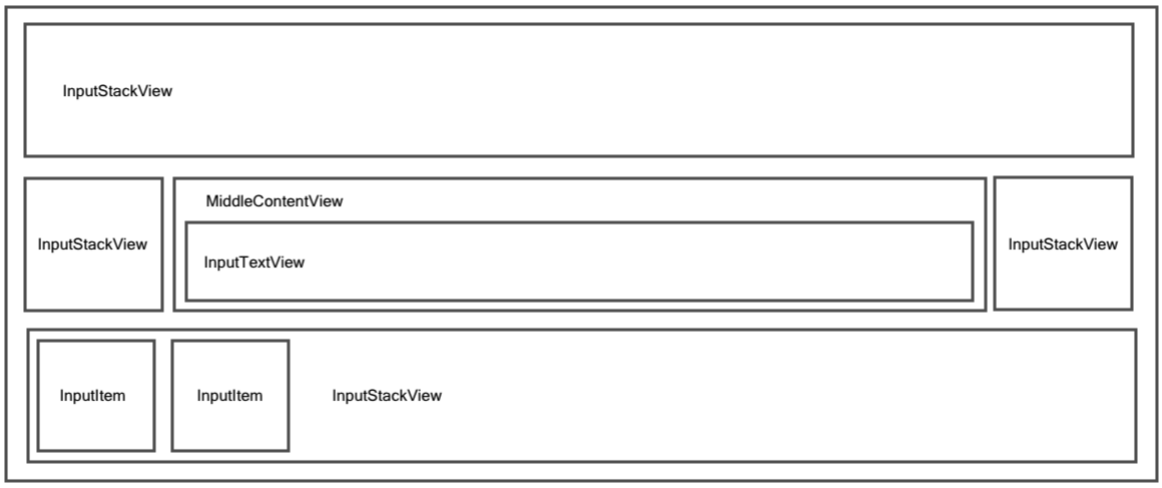
구분	설명
안전성	• 프로그래머가 저지를 수 있는 실수를 엄격한 문법을 통하여 버그를 미리 방지 → 강제적이라고 느껴질 수 있으나 문법적 제재는 실수를 줄이는데 큰 도움 • 옵셔널, Guard 문, 오류처리, 타입통제 등을 통해 안전한 프로그래밍 구현 • 불안전한 코드의 전체 클래스를 제거함으로써 변수는 사용 전에 항상 초기화, 배열 및 정수에 대한 오버플로우 검사 수행, 메모리 자동관리
속도 및 성능	• C 언어를 기반으로 한 C++, Object-C 와 같은 프로그래밍 언어를 대체하려는 목적으로 개발되었기 때문에 성능을 최대화 한 C 언어에 가깝게 구현

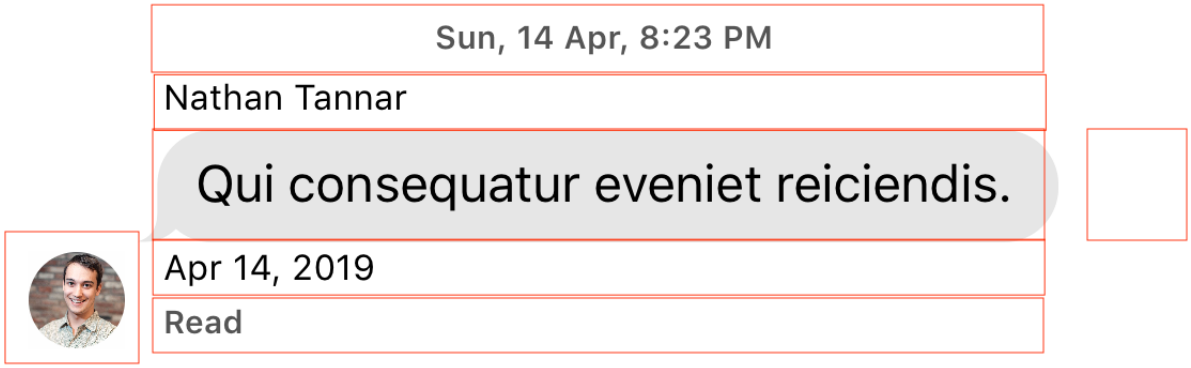
	• 실행 속도의 최적화 뿐만 아니라 컴파일러의 지속된 개량을 통해 더 빠른 컴파일 성능을 구현
표현성	• 순수하게 함수에 전달된 인자 값만 결과에 영향을 주고 상태 값을 가지지 않는다. → 즉, 어떠한 상황에서 프로그램을 실행하더라도 일정하게 같은 결과를 도출할 수 있으며 대규모 병렬처리, 멀티 코어의 환경에서 효율적인 프로그래밍이 가능 • 프로토콜 지향 프로그래밍 언어로 캡슐화, 추상화, 접근 제어 등의 기능들을 구조체와 열거형에서 구현 가능 • 인스턴스보다 값 타입을 사용 함으로서 더 나은 효율성과 오류 최소화, 참조로부터 자유로움을 추구

• 라이브러리 목록

```
import UIKit
import MessageKit
import InputBarAccessoryView
import Firebase
```

① MessageKit





MessageKit 은 전반적인 GUI 구성을 위해 사용하는 라이브러리로 주요 모듈은 MessageViewController: UICollectionView 를 상속하여 구현되었다. 이를 상속해 ChatViewController 를 생성할 예정이다. 내부적으로는 messageCollectionView 프로퍼티가 사용 가능하다. 더불어 이 라이브러리를 다운받으면 AccessoryView 위에 위치한 InputBar 를 사용할 수 있어 InputBarAccessoryView 도 import 하여 같이 사용해줄 예정이다.

```
class ChatViewController: MessagesViewController {
        override func viewDidLoad() {
                super.viewDidLoad()
                messagesCollectionView.messagesDataSource = self
                messagesCollectionView.messagesLayoutDelegate = self
                messagesCollectionView.messagesDisplayDelegate = self
        }
}
```

위와 같은 델리게이트를 이용해 전체적인 UI 를 구성해 줄 예정이다.

② Firebase

NoSQL 기반 클라우드 데이터베이스로 데이터를 저장, 동기화 하고 JSON 으로 저장하면서 Swift 에서 지원이 되기 때문에 이를 이용하고자 한다.

```
{
  "channels": [{
    "MOuL1sdbrnh0x1zGuXn7": { // channel id
      "name": "Puppies",
      "thread": [{
        "3a6Fo5rrUcBqhUJcLsP0": { // message id
          "content": "Wow, that's so cute!",
          "created": "April 12, 2021 at 10:44:11 PM UTC-5",
          "senderId": "YCrPJF3shzWSHagmr0Zl2WZFBgT2",
          "senderName": "naturaln0va",
        },
        "4LXlVnWnoqyZEuKiiubh": { // message id
          "content": "Yes he is.",
          "created": "April 12, 2021 at 10:40:05 PM UTC-5",
          "senderId": "f84PFeGl2yaqUDaSiTVeqe9gHfD3",
          "senderName": "lumberjack16",
        },
      }]
    },
  }]
}
```

이후 채팅 내역은 Firebase 라이브러리 설치 시 함께 이용할 수 있는 Firestore 를 이용해 채팅내역을 자동으로 저장하여 차후 사용자가 검색을 할 수 있도록 하고자 한다.

1.2. 백엔드

백엔드에서 사용할 프레임 워크는 파이썬을 활용하여 개발할 예정이므로 google colab 을 사용하여 개발할 예정이다.

1) 사용 기술 및 프레임워크

- Python 3.7.1 이상 버전 사용

- Colab: 구글 CoLaboratory 는 클라우드 기반 개발 환경이다. 구글의 CPU, TPU, RAM 을 사용해서 '주피터 노트북'과 같은 개발 환경을 클라우드 서비스로 제공해준다. 구글 드라이브를 통해서 데이터를 읽고 쓸 수 있으니까, 인터넷만 되면 어디서든 클라우드 하드웨어와 소프트웨어를 사용해서 자신의 컴퓨터 환경에 영향을 받지 않고 작업을 할 수 있다.

- Firebase: 데이터베이스로는 Firebase 를 사용할 예정이다. Firebase 는 모바일 및 웹 애플리케이션을 만들기 위해 2011 년 google firebase 사에서 개발했다. 앱 개발의 다양한 측면을 간소화하여 개발자가 시간과 노력을 절약할 수 있도록 도와준다. 쉽게 말해 BaaS(Backend-as-a-Service)플랫폼으로 제공되는 포괄적인 도구 및 서비스 제품군으로, 개발자가 모바일 및 웹 애플리케이션을 모두 쉽게 생성, 실행 및 확장할 수 있도록 한다. 실시간 데이터베이스, 인증, 스토리지, 호스팅 및 기타 기능을 제공하며 모두 단일 플랫폼에서 관리된다. Firebase 의 대표 기능이라고 할 수 있는 Firestore 는 nosql 서비스 이고, 1 기가 바이트 저장까지는 무료이며 한달에 문서 쓰기 60 만번, 문서 읽기 150 만번, 문서 삭제 60 만번 까지 무료이다.

- API(text-davinci-003): GPT-3.5 모델은 자연어 또는 코드를 이해하고 생성할 수 있다. 가장 유능하고 비용효율적인 모델은 gpt-3.5-turbo 모델인데 채팅에 최적화되어 있지만 기존 모델들도 잘 동작한다. 그 중에 text-davinci-003 모델은 curie, Babbage 또는 ada 모델보다 더 나은 품질, 더 긴 출력 및 일관된 지침 준수로 모든 언어 작업을 수행할 수 있다. 또한 텍스트 내에 완성 삽입을 지원한다. 다빈치(Davinci) 모델은 가장 유능한 모델 제품군이며 다른 모델이 수행할 수 있는 모든 작업을 수행할 수 있을 뿐만 아니라 적은 지침으로도 수행할 수 있다. 특정 청중을 위한 요약 및 창의적인 콘텐츠 생성과 같이 콘텐츠에 대한 많은 이해가 필요한 응용 프로그램의 경우 Davinci 가 최상의 결과를 생성할 것이다. 다만 향상된 기능에는 더 많은 컴퓨팅 리소스가 필요하므로 Davinci 는 API 호출당 비용이 더 많이 들고 다른 모델에 비해 속도가 조금 느리다. 하지만 많은 종류의 논리 문제를 해결하고 등장인물의 동기를 설명하는데 꽤 능숙하며 인과 관계에 관련된 어려운 문제를 잘 해결한다는 장점이 있다. 챗봇 모델 api 는 'text-davinci-003'를 tuning 하여 사용할 예정이다.

2) 라이브러리 및 함수, 파라미터

- OpenAI Python 클라이언트 라이브러리 설치

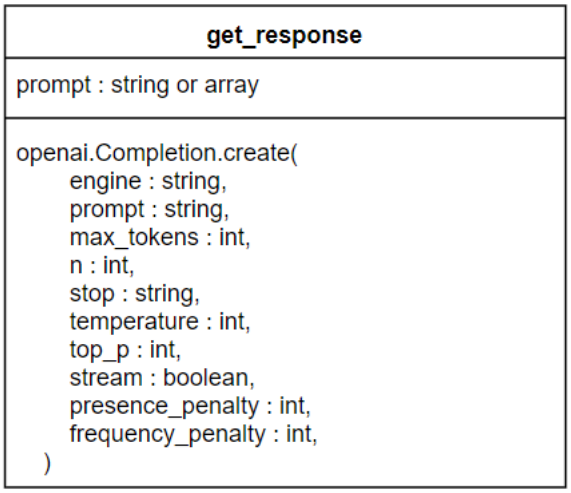
  `pip install openai`

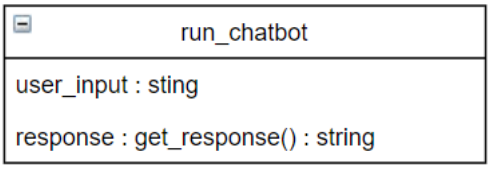
Example request text-davinci-003 python Copy

```
import os
import openai
openai.api_key = os.getenv("OPENAI_API_KEY")
openai.Model.retrieve("text-davinci-003")
```

- 고유 API KEY 값 등 필요한 정보들을 저장하기 위해 dotenv 모듈 설치

  `pip install dotenv`





- get_response(prompt) : api 로부터 답변을 생성하는 함수
  - 인자로 받는 prompt 는 사용자의 질문을 입력 받은 변수를 넘겨 받는다.
  - Response.choices[0].text.strip() 반환

- Openai.Completion.create() : completion 생성
  - engine : 사용할 langage model
  - prompt : completion 을 생성하기 위한 입력 프롬프트
  - max_tokens : completion 을 생성할 때 최대로 생성 가능한 토큰의 수 설정
  - temperature : 샘플링 온도 0~2. 값이 높을수록 다양하게(랜덤하게) 답변, 값이 낮을수록 질문 의도에 집중한다.
  - top_p : temperature 의 대안으로 사용하며 확률질량으로 토큰의 결과를 고려하는 nucleus sampling 기법을 사용한다. top_p 와 temperature 둘 중 하나의 값만을 사용할것을 권장한다.
  - n : 각 prompt 가 생성할 completion 의 개수 설정
  - stream : 부분 진행 상황을 스트리밍 할지 여부.
  - stop : API 가 추가 토큰 생성을 중지하는 최대 4 개 시퀀스. 반환된 텍스트에는 중지 시퀀스가 포함되지 않는다.
  - presence_penalty : -2.0 에서 2.0 사이의 숫자로 양수 값은 지금까지 텍스트에 사용되었는지 여부를 판단하여 새 토큰에 페널티를 주어 모델이 새 주제에 대해 이야기 하도록 한다.

- frequency_penalty : -2.0 에서 2.0 사이의 숫자로 양수 값은 지금까지 텍스트의 기존 빈도를 기반으로 새 토큰에 페널티를 주어 모델이 동일한 줄을 그대로 반복할 가능성을 줄인다.

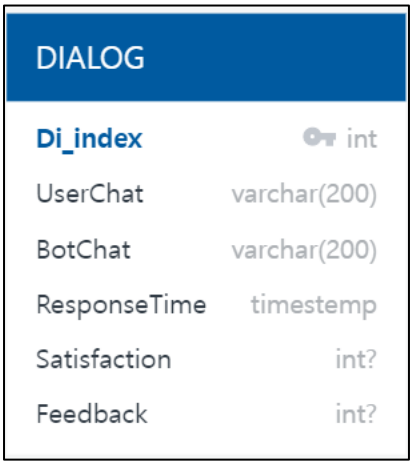
Parameters text-davinci-003 Copy

```
{
  "model": "text-davinci-003",
  "prompt": "Say this is a test",
  "max_tokens": 7,
  "temperature": 0,
  "top_p": 1,
  "n": 1,
  "stream": false,
  "logprobs": null,
  "stop": "\n"
}
```

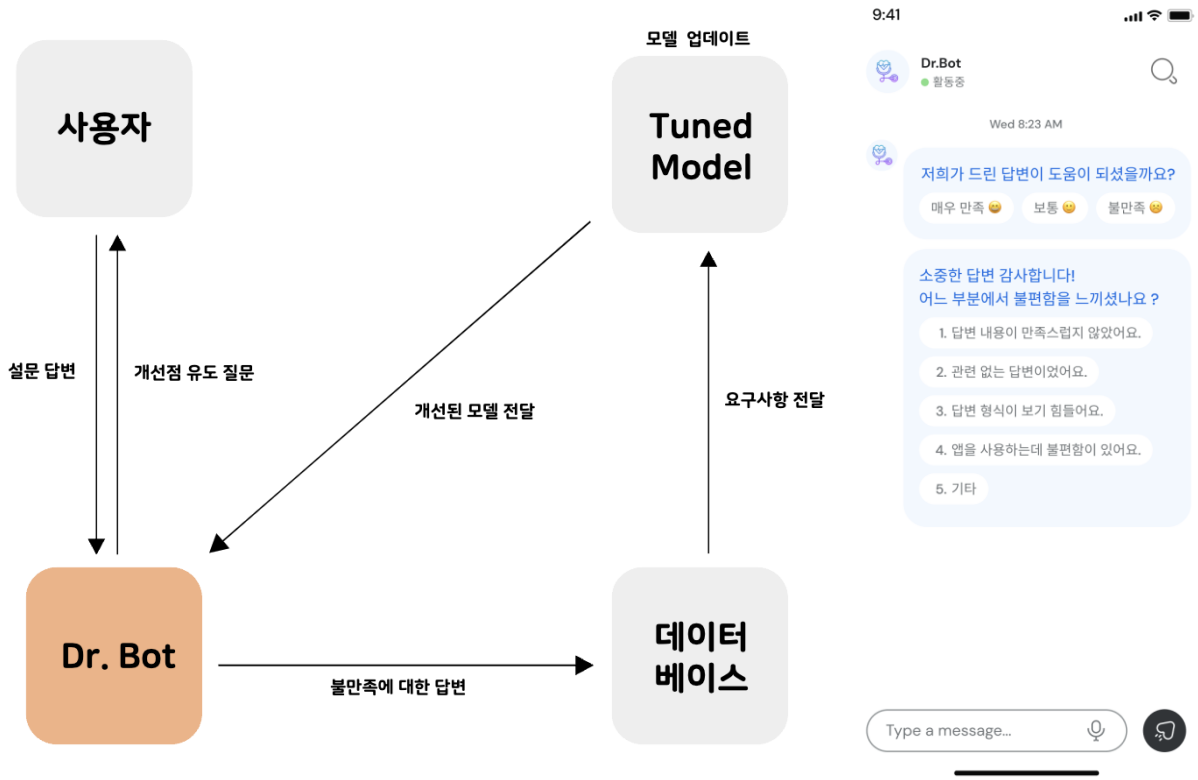
- run_chatbot() : 챗봇을 실행하기 위한 함수

3) DB 설계

- Di_index : 채팅 중 생성되는 말 뭉치에 대한 번호
- UserChat : 사용자의 물 뭉치 도메인
- BotChat : API 를 통해 사용자의 질문에 답한 챗봇의 말뭉치 도메인
- ResponseTime : 챗봇의 답변 했을 때의 시스템 시각 도메인
- Satisfaction : 사용자의 만족도 입력 도메인
- Feedback : 챗봇의 답변에 사용자가 불만족 했을 경우 제시하는 불편사항 카테고리의 선택 입력 도메인



2. 서비스 흐름도



사용자가 앱을 다 사용하고 나면 플랫폼에서는 사용자에게 개선점을 위한 질문을 한다. 개선점을 위한 설문은 만족 / 보통 / 불만족으로 이루어지며 만족 혹은 보통일 경우에는 데이터베이스에 반영되지 않으며, 불만족일 경우에 추가적인 질문이 위 사진과 같이 이루어진다.

사용자가 “Dr.Bot” 앱을 실행하여 사용하면 매 답변의 마지막에 만족도를 조사하도록 설계할 예정이다. 이에 사용자가 불만족을 눌렀을 경우 개선점을 유도하는 질문을 피드백한다. 이후 사용자가 선택한 버튼을 DB 에 저장하여 개선 방향성을 설정하고 그 방향성을 토대로 챗봇 모델을 Tuning 한다. 업데이트된 모델을 다시 제공함으로써 이전의 불편사항을 반영하여 개선되고 있음을 느낄 수 있도록 구현하고자 한다.

3. 참고자료

- 애플 개발자 문서

  https://developer.apple.com/kr/swift/

- Firbase 프로젝트 이해

  https://firebase.google.com/docs/projects/learn-more?hl=ko

- OpenAI API 모델 설명

  https://iotnbigdata.tistory.com/609

- https://github.com/MessageKit/MessageKit/blob/main/Documentation/QuickStart.md
- https://www.kodeco.com/22067733-firebase-tutorial-real-time-chat

- openAI API REFERENCE

  https://platform.openai.com/docs/api-reference/completions/create?lang=python