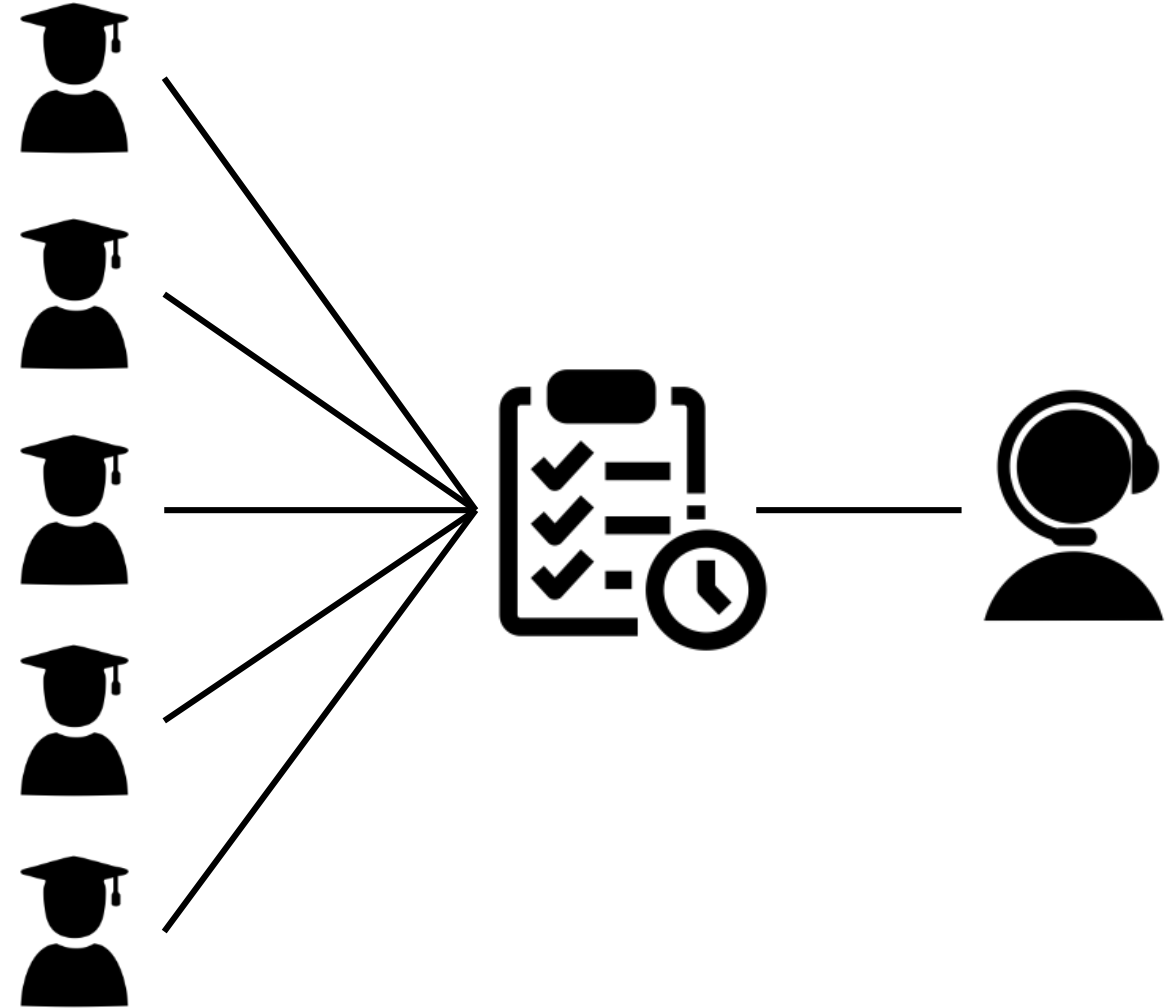# SKKU-GPT

SKKU-GPT - Improved SKKU Chatbot

Contents

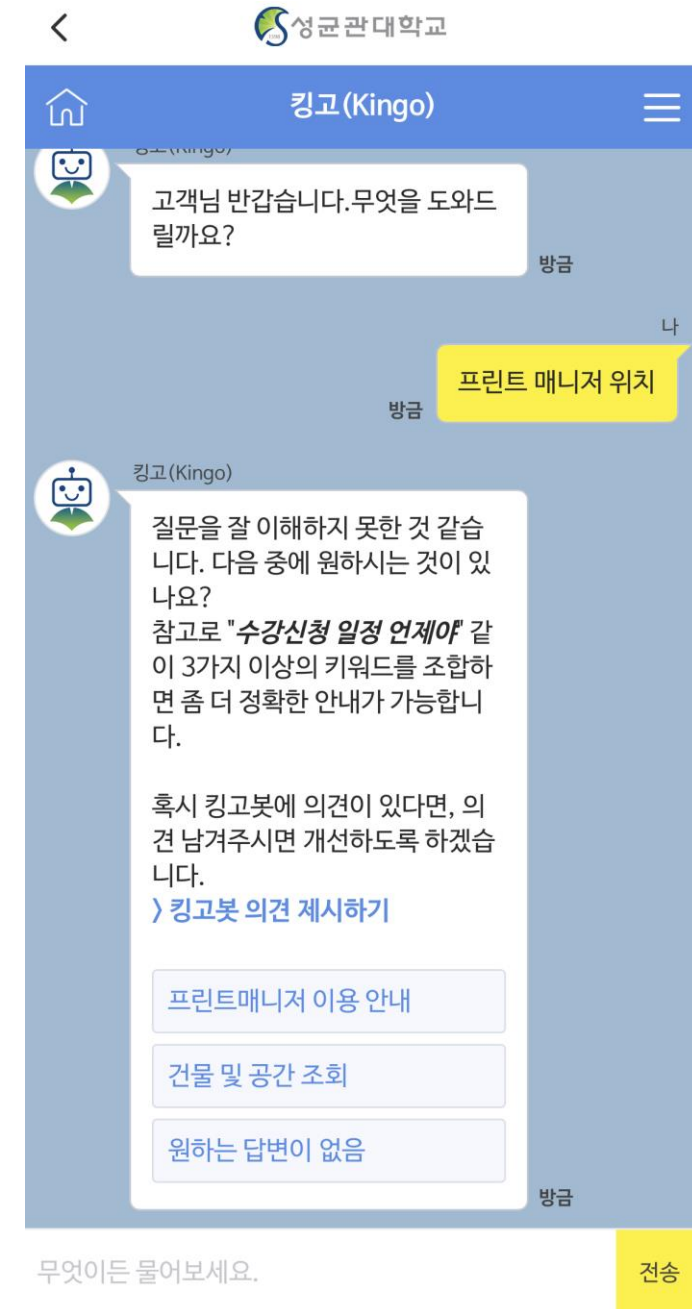# Part 1
Motivation

Motivation

- Administrative office repeatedly handle tasks for a large number of students, such as graduation requirements, scholarships, and course registration

- There is a lot of work burden because the administrative office has to give the same answer to a large number of students

- Finding related school rules on the school website is quite time-consuming and inconvenient work

- Some students find it difficult to call the administration directly

Motivation

Students are able to use chatbot via KINGO-M Application

- Though questions are simple, sometimes, chatbot cannot make appropriate answer
  - Increasing user's input
  - Cannot receive useful answer by multiple questions

- Application has to be restarted because of short session time
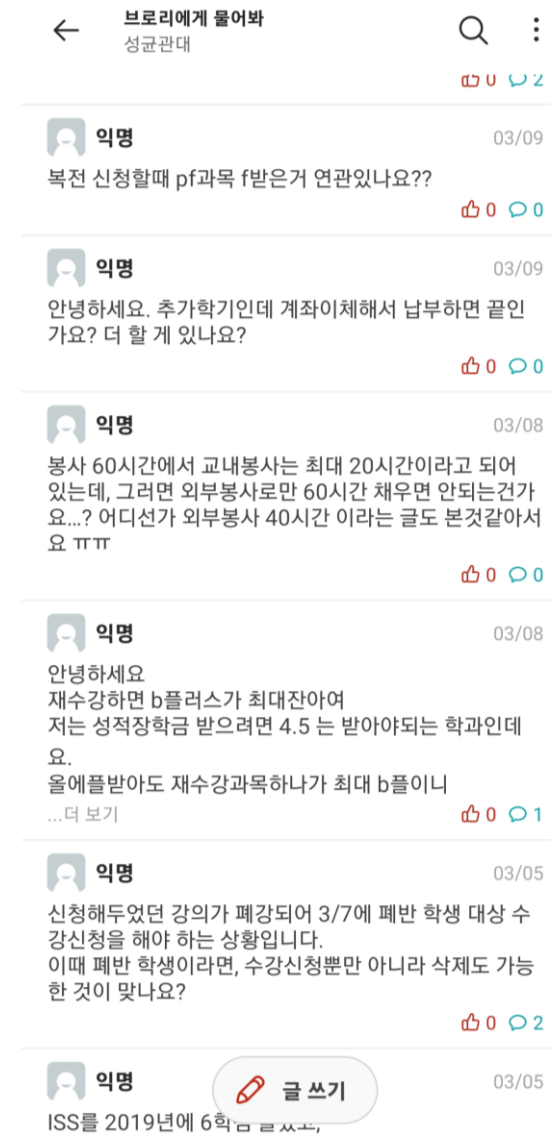
- Chatbot doesn't learn about new notice

Motivation

Students can get information at university community (e.g., Everytime)

- Students who have difficulty calling to administration ask question to community

- Many students ask questions via community, however many of those questions can't get answer
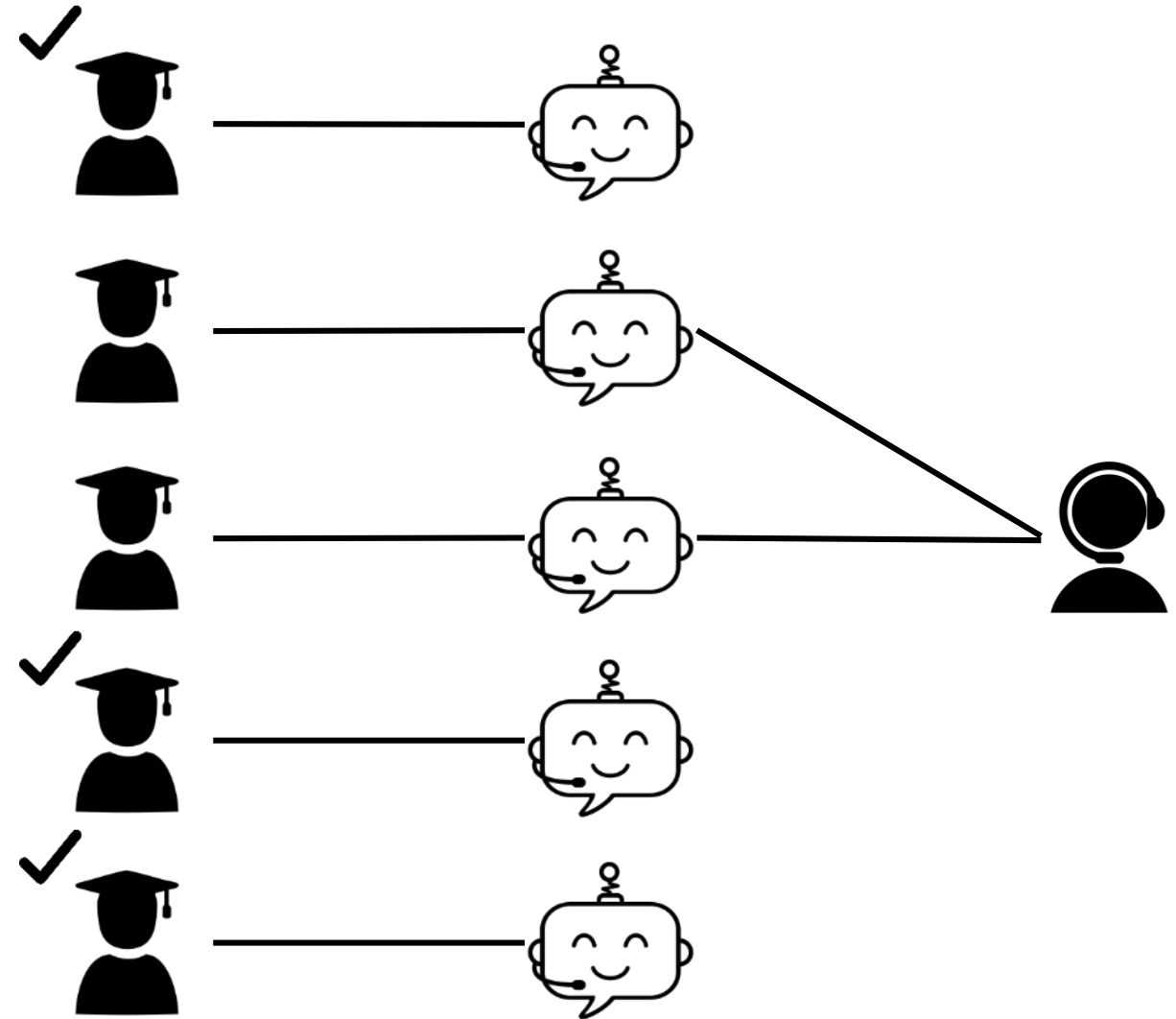
- Reliability of answer at community is not ensured

Part 2
Project's Goal

Project's Goal

- By improving the shortcomings of existing chatbot, students can use SKKU-GPT more conveniently

- Students can get administrative answers faster

- Students can get more accurate answers than the community

- Students can receive relevant answers outside of administrative office hours

- Reduce the burden on administrative offices

Project's Goal

Large language models, such as ChatGPT, tend to tell untrue content as if it were true (Hallucination)

- Therefore, in order to improve the reliability of the answer, documents such as announcements, school rules, etc. are included with questions
- Suggest a way to provide it to ChatGPT, allowing models to view documents and answer questions
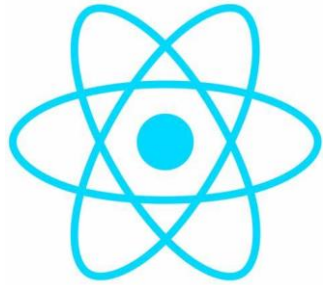 - At this time, it is necessary to find a document related to the question (=search engine)

Part 3
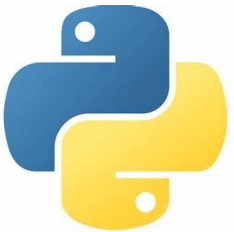# Implementation / Demo

Environment

Frontend

JavaScript

React Native

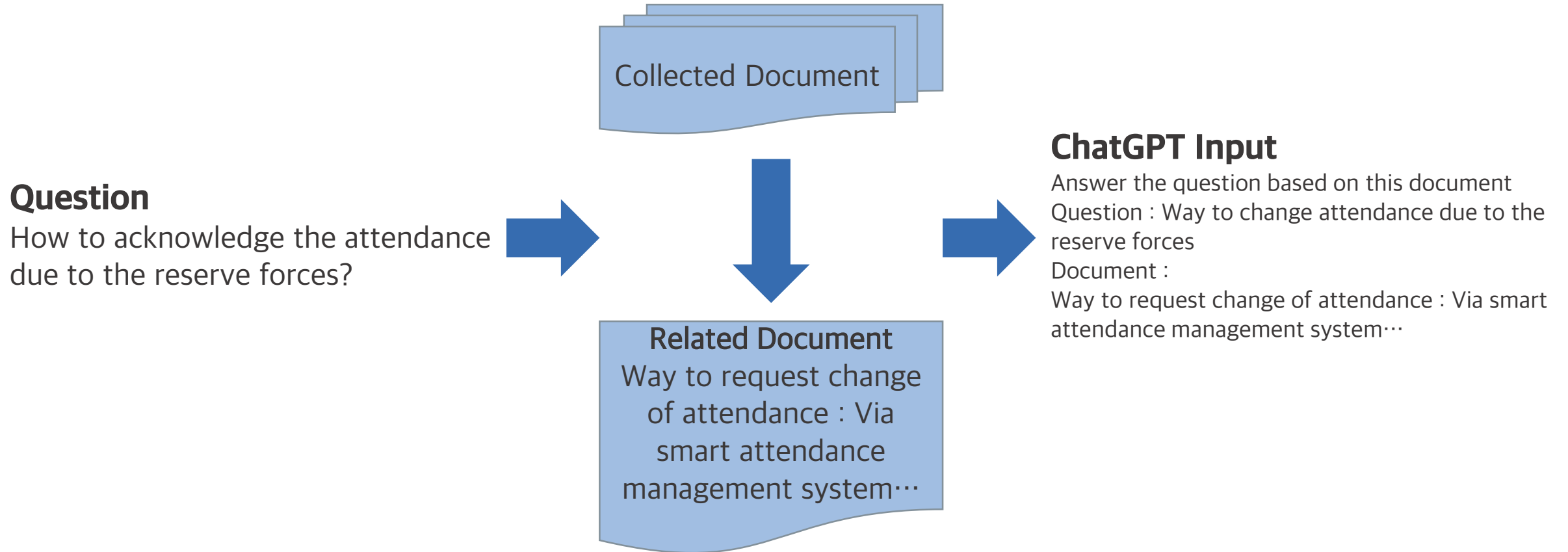## Backend

python™

Flask
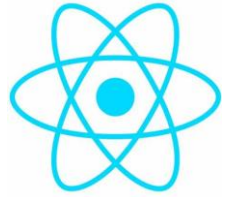
Document Search

Elasticsearch

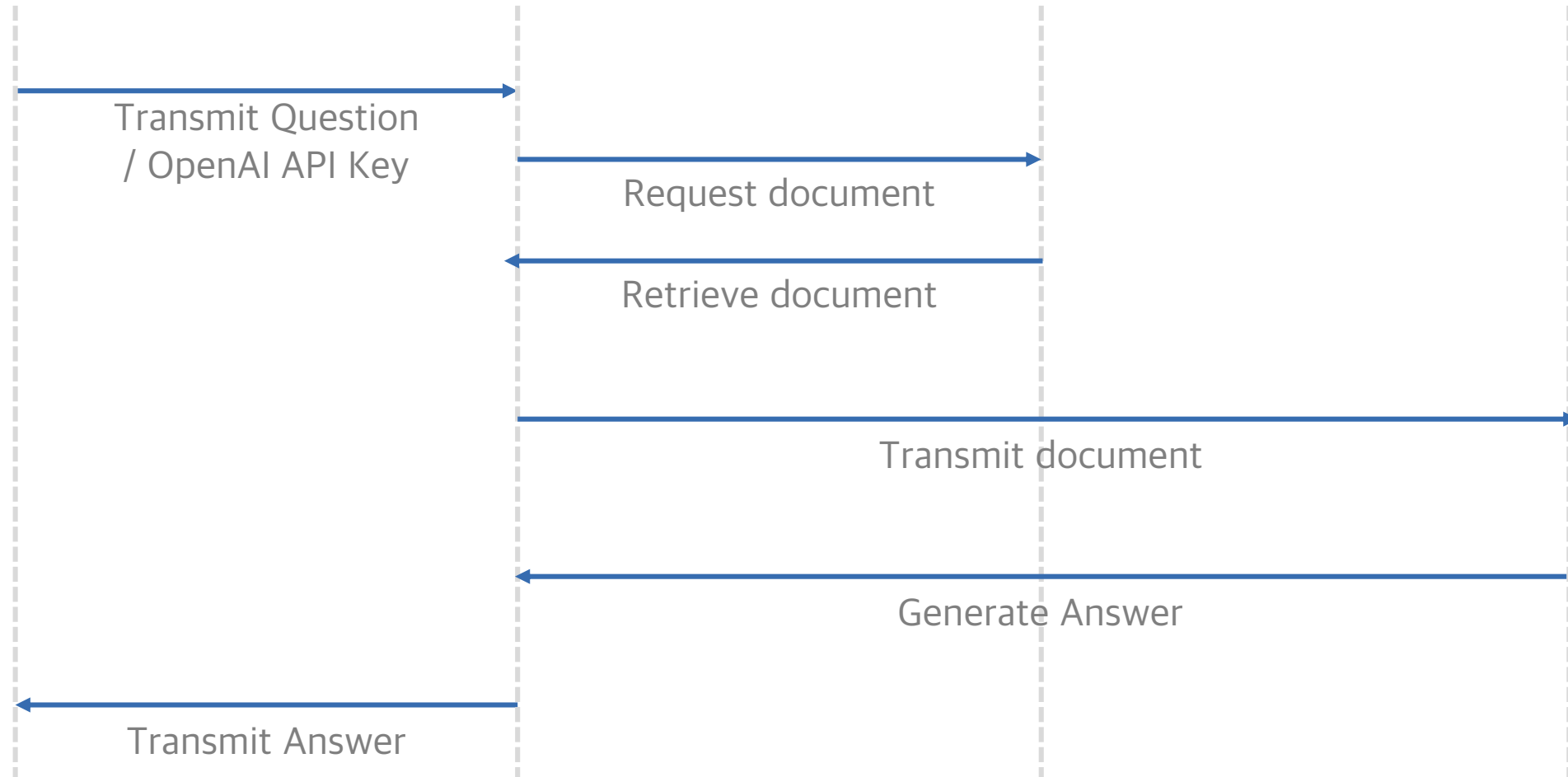## Generating Answer

ChatGPT API

Implementation Plan

Question
How to acknowledge the attendance due to the reserve forces?

Collected Document

Related Document
Way to request change of attendance : Via smart attendance management system…

ChatGPT Input
Answer the question based on this document
Question : Way to change attendance due to the reserve forces
Document :
Way to request change of attendance : Via smart attendance management system…

Implementation Plan

Implementation Plan

- We have generated documents on department notices, academic policies for the past five years in the form of JSON

- Elasticsearch generates index automatically using JSON document
  - By generated index, we can get document based on question's words

Demo

Part 4
Test Plan

Test Plan - Login

- **Login Success**

  : Checking that login is executed when ID and password is registered

- **Login Failed (Password)**

  : Checking that login is failed when password is different with registered user's password

- **Login Failed (Unregistered user)**

  : Checking that login is failed when ID and password is not registered

- **Logout**

  : Checking that logout is executed successfully when logged-in user tries to logout.

Test Plan - Chat

- **Answer Generation**

  : Checking that system generate valid answer related to academic policies

- **Answer Generation Time**

  : User should be able to receive answer within 5 seconds

- **Answer Generation Repeatability**

  : Application should be able to generate answer when user transmit question repeatedly

- **Support Multi-Language**

  : Application should be able to generate answer in Korean and English

Test Plan - Interface

- **Elasticsearch**

  : Checking that Elasticsearch can return valid document related to user's question

  : Checking the case that there is no appropriate document related to user's question

- **OpenAI API**

  : Checking that system can communicate with OpenAI server successfully

  : User's question must not longer than 120 characters

  : User's question must follow terms of use

  : When OpenAI server doesn't make response, exception handling should be executed

More details in Test Plan Specification

Thank You