HI-Chat – 2학기

: A Chatbot for Hongik University Information Guidance

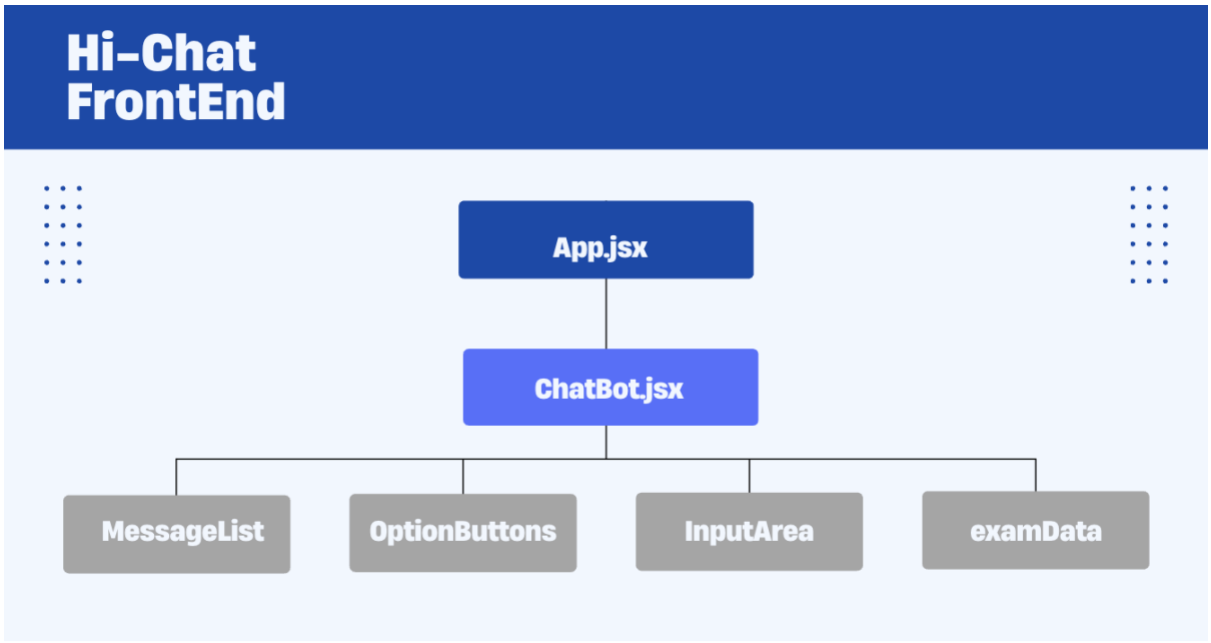
지도교수	팀원	학번	C011147	이름	이승연
이장호			C111038		김지윤

Contents

- **Hi- Chat의 컴포넌트 구조도 및 설명**
- **논리적 구조 설계 – 하이브리드 인터페이스**
- **주요 기술 스택**
- **사용자 시나리오의 예시**
- **향후 개선 방향**
- **결론**

Hi- Chat 컴포넌트 구조도 및 설명

프론트엔드 파트에서는 챗봇의 전체 UI/UX 설계, 상태 관리 로직 구현, 데이터 구조 설계 및 화면 렌더링을 담당하였다.



1. App.jsx

HI-Chat의 전체 화면을 구성하는 루트 컴포넌트이다. 이는 ChatBot.jsx를 불러와 실제 챗봇 인터페이스를 렌더링하고, 앱의 진입점 역할을 한다.

2. ChatBot.jsx

사용자와의 대화 흐름을 관리하고 메시지, 버튼, 입력창 등을 제어하는 핵심 컴포넌트이다. 사용자의 선택이나 입력에 따라 단계별로 학사 관련 정보를 안내한다.

2.1 MessageList

사용자가 보낸 메시지와 챗봇의 응답 메시지를 화면에 차례대로 보여주는 영역이다. messages 상태값을 기반으로 대화가 생성될 때마다 새로운 메시지가 아래에 추가된다. 각 메시지는 user 또는 bot 타입으로 구분되어 다른 스타일로 표시되어 실제 채팅처럼 보인다. 메시지 내용이 누적되어도 스크롤로 자동 이동되도록 설계해서 사용성이 불편하지 않게 했다.

2.2 OptionButtons

OptionButtons는 챗봇이 제시하는 선택지 버튼들을 모아둔 부분이다. 예를 들어 "시험 일정 조회"나 "장학금 안내" 와 같은 4개의 주요 기능 옵션들이 해당된다. 사용자가 버튼을 클릭하면 handleOptionClick()이 실행되어 챗봇이 그 선택에 맞는 다음 질문이나 결과를 보여준다. 버튼식 인터페이스로 구현해 사용자가 오입력할 가능성을 줄이고, 직관적으로 기능을 고를 수 있도록 하였다.

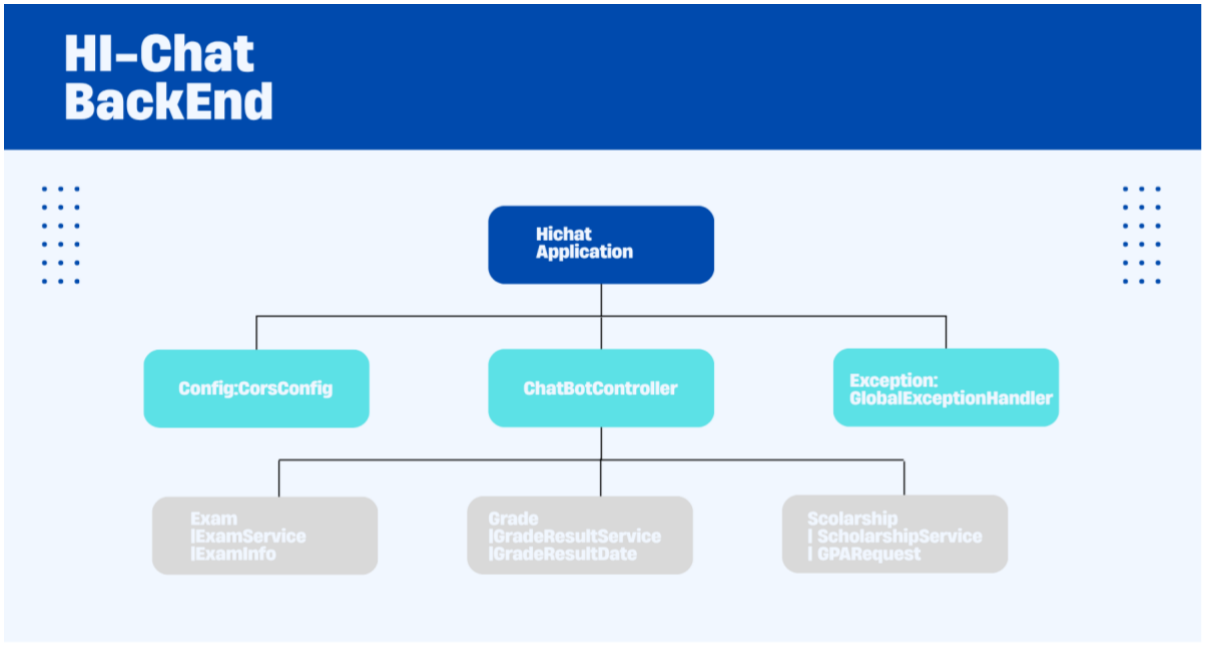
2.3 InputArea

사용자가 직접 텍스트를 입력할 수 있는 영역이다. 현재는 "장학금 안내" 기능에서 평점을 입력할 때만 활성화된다. 사용자가 평점을 입력하고 전송 버튼을 누르면 handleInputSubmit() 함수가 실행되고, 입력값을 검증한 후 장학금 정보가 출력된다. 이 구조는 이후 NLP 기능을 추가해서 자연어로도 질의할 수 있도록 설계되었다.

2.4 examData.js

examData.js는 챗봇이 참고하는 데이터 파일이다. 현재는 2025학년도 2학기 중간고사 일정과 장학금 정보 정도만 담고 있고, 데이터가 많지 않기 때문에 따로 데이터베이스를 연결하지 않았다. 구조는 JSON 형태로 되어 있어서 학기 → 학년 → 과목 → 교수 순으로 탐색할 수 있다. 나중에 기능이 확장되면 이 데이터를 API나 DB로 전환하기 쉽게 설계해둔 상태이다.

백엔드 파트에서는 REST API 설계와 비즈니스 로직 구현, 요청값 검증/예외 처리, CORS 설정, 그리고 시험·성적·장학금 등의 인메모리 데이터 관리 및 응답 포맷화를 담당하였다.



1. HichatApplication

스프링 부트 진입점. 내장 톰캣을 기동하고 하위 패키지들을 컴포넌트 스캔한다. 실행 포트는 application.properties 의 server.port(현재 8080).

2. ChatBotContorller : 프론트에서 호출하는 모든 엔드포인트가 모여있는 컨트롤러

GET /(grades – 학기별 학년 목록, subjects – 학기/학년별 과목 목록, professors – 과목별 교수 목록, exam-info – 과목.교수별 시험 일정(일시/강의실), grade-result-date – 성적 공개 일정),
POST /scholarship – GPA,봉사여부로 장학금 적격 목록 반환

3. Exam : 시험 일정
- ExamService: 메모리 데이터에서 학기→학년→과목→교수→분반 시험정보를 탐색해 응답.
- ExamInfo (DTO): date/time/room 등 시험 세부정보를 담는 응답 모델

4. Grade : 성적 일정

- GradeResultService: 현재 학기의 성적 열람 시작일·시간을 제공
- GradeResultDate (DTO): semester/date/time 필드로 단순 반환

5. Scholarship : 장학금

- ScholarshipService: 표 규칙에 따라 필터링.

  홍익인간: GPA ≥ 4.0 (봉사 무관, 전액)
  자주/창조: GPA ≥ 3.5 & 봉사 필요
  협동: GPA ≥ 3.3 & 봉사 필요
  정진: GPA ≥ 3.0 (봉사 불필요, 90 만원)

- GPARequest (DTO, 입력): gpa, volunteer 검증(@DecimalMin/Max)
- Scholarship (DTO, 출력): name/minGpa/amount/description/requireVolunteer

6. Config

- CorsConfig: 프론트 개발 도메인(예: http://localhost:3001)을 허용하고 GET/POST/OPTIONS 를 열어준다.

7. Exception

- GlobalExceptionHandler: 검증 실패(400)와 서버 오류(500)를 통일된 JSON 형태로 응답.

8. 데이터 흐름

- 프론트(React) → /api/chat/* 호출 → Controller → Service(인메모리 데이터/규칙) → DTO 로 JSON 응답 → 프론트가 채팅풍 메시지로 렌더링

논리적 구조 설계 – 하이브리드 인터페이스

HI-Chat은 버튼식 인터페이스와 대화식 인터페이스를 혼합한 하이브리드형 인터페이스로 작동되기로 계획되었다. 현재는 버튼식 인터페이스만 구현된 상태이다. 따라서 사용자가 명확하게 선택지를 고를 수 있도록 설계하였다. 향후, 사용자의 자연어 입력을 인식할 수 있는 간단한 NLP 인터페이스(IBM Watson)을 추가할 예정이다. 또한, 추후 DB 도입 시 service 아래 repository 인터페이스로 확장 용이하게 설계할 예정이다. 이에 사용자가 "시험 일정 알려줘"와 같이 직접 입력하더라도 의도를 파악할 수 있도록 한다. 현재는 그 기반이 되는 버튼 선택 로직을 안정화하는 데 초점을 두고 있다

frontend part 주요 기술 스택

- **언어:** JavaScript (React + JSX)
- **UI 라이브러리:** lucide-react (아이콘 이용 위함)
- **스타일링:** CSS

: 프로젝트의 유지 보수의 편이성과 구조의 명확성을 위해 스타일링을 별도 파일인 ChatBot.css로 분리하였다.

- **데이터 관리:** Local JSON (examData.js)

: 현재 데이터 관리 파일은 examData.js로 2025학년도 2학기 중간고사 일정과 장학금 안내 등 비교적 단순하고 고정적인 정보를 포함하고 있다. 이러한 데이터는 변경 주기가 길고 실시간 갱신이 필요하지 않다. 따라서, 지금까지는 별도의 데이터베이스를 구축하지 않고 로컬 JSON 구조로 관리할 수 있도록 설계되었다.

backend part 주요 기술 스택

- **언어 & 런타임:** Java 17(OpenJDK), Spring Boot 3.5
- **프레임워크/모듈:** Spring Web(Spring MVC) -REST API 제공,
  Spring Validation(Jakarta.validation)- @Valid, @DecimalMin/Max로 요청값 검증

- **빌드 & 프로젝트 관리:** Gradle(Wrapper포함),
  표준 디렉터리 구조(src/main/java, src/test/java),
  실행 포트 설정: application.properties의 server.port (현재 8080)

- **구성(패키지) & 컴포넌트:**

1) controller

ChatBotController – /api/chat 하위 엔드포인트 집약

GET /grades, GET /subjects, GET /professors, GET /exam-info

GET /grade-result-date

POST /scholarship (GPA/봉사여부로 장학금 적격 조회)

2) service

ExamService – 시험 일정/교수/과목 조회 (메모리 데이터)

GradeResultService – 성적 공개일 제공 (메모리 데이터)

ScholarshipService – 표 규칙 기반 장학금 필터링 로직(GPA/봉사 조건)

3) dto

ExamInfo, GradeResultDate – 응답 DTO

GPARequest – 장학금 조회 요청 DTO

Scholarship – 장학금 응답 DTO

4) config

CorsConfig – 프론트 도메인(예: http://localhost:3001) 허용, GET/POST/OPTIONS 허용

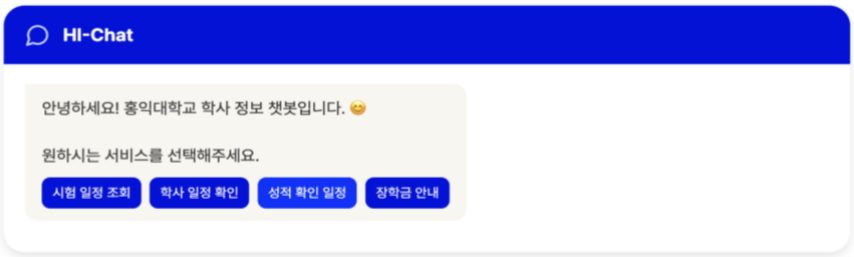
5) exception

GlobalExceptionHandler – 검증 오류(400)·서버 오류(500) JSON 포맷 통일

사용자 시나리오의 예시 - ①

사용자가 2025학년도 2학기 중간고사 2학년 과목인 멀티미디어응용수학의 시험일정에 대해

조회하는 시나리오에 대한 예시이다



<첫 화면>

<시험 일정 조회 → 2학년 클릭>

<시험 일정 조회 → 2학년 클릭>

① 첫 화면

사용자가 사이트에 접속하면 챗봇이 인사 메시지와 함께
시험 일정 조회 / 학사 일정 확인 / 성적 확인 일정 / 장학금 안내 의 네 가지 메뉴를 보여준다.

② '시험 일정 조회' 선택 후 → 학기/학년 선택

[시험 일정 조회]를 클릭하면 챗봇은 "조회할 학기와 시험을 선택해주세요."라는 메시지를 보여주고,
2025학년도 학기별 시험 옵션을 제시한다.
사용자가 [2025-2학기 중간고사] → [2학년] 을 순서대로 선택하면 해당 학년의 과목 목록을 챗봇이 불러온다.

③ 과목 및 교수 선택 단계

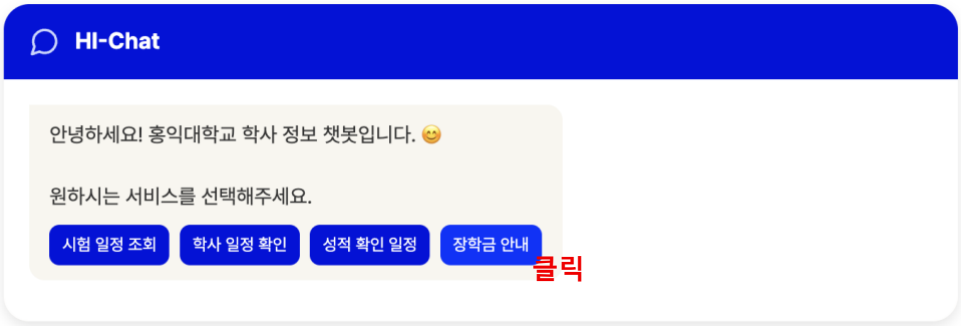
이후, 사용자는 [멀티미디어응용수학] 을 선택한다.
그 후 챗봇이 "교수님을 선택해주세요."라고 안내하며,
등록된 교수명(김태형 교수)을 버튼 형태로 표시한다.
이 단계에서의 선택은 examData.js 내부 데이터를 기반으로 이루어고, 선택된 교수명에 따라 세부 시험 일정이 즉시 출력된다.

④ 결과 출력 및 다음 단계 안내

챗봇은 선택된 과목의 시험 일시와 강의실 정보를 메시지 형태로 보여준다.
그리고, "다른 서비스를 이용하시겠습니까?"라는 질문을 출력해 다시 초기 메뉴로 돌아갈 수 있도록 설계되어 있다.

사용자 시나리오의 예시 - ②

사용자가 자신의 학점을 입력했을 때 챗봇이 수령할 수 있는 장학금에 대한 정보를 제공하는 시나리오



<첫 화면>

<[장학금 안내] 클릭 -> 사회봉사 시간 이수 여부 체크 >

<관련 정보 제공>

① 첫 화면

사용자가 사이트에 접속하면 챗봇이 인사 메시지와 함께 시험 일정 조회 / 학사 일정 확인 / 성적 확인 일정 / 장학금 안내 의 네 가지 메뉴를 제시한다.
사용자가 [장학금 안내] 버튼을 클릭하면 장학금 관련 안내 대화가 시작된다.

② 평점 입력 단계

챗봇은 "교내 장학금 수혜 가능 여부를 확인해드리겠습니다. 본인의 평점을 입력해주세요."라는 메시지를 출력한다.
사용자는 예시(예: 3.75)에 맞춰 평점을 입력한 뒤 전송 버튼을 누른다.

**** 로직 변경 내용**

이 부분은 기존 ChatBot.jsx에서 단순히 클라이언트 내부 조건으로 처리되던 로직이었으나,
백엔드 파트를 추가하면서 서버와 연동되도록 구조가 변경되었다.

평점 입력 시, 프론트엔드에서는 사용자의 입력값을 백엔드로 전달하고 서버에서는 실제 장학금 기준(평점 컷, 필수 조건 등)에 따라 수혜 가능 여부를 판별한다.

이 방식을 통해 규정이 변경되더라도 프론트 코드를 수정할 필요 없이,
백엔드 로직만 수정하면 자동으로 반영되도록 개선되었다.

③ 사회봉사 이수 여부 확인

이전 버전에서는 평점만으로 단순한 장학금 추천 결과가 출력되었으나,
팀원이 백엔드 파트를 맡으면서 '사회봉사 시간 이수 여부' 체크 단계가 새로 추가되었다.
챗봇은 "사회봉사 시간을 이수하셨나요?"라는 질문을 출력하고,
사용자는 [예] / [아니요] 중 하나를 선택한다.

이 선택 결과는 백엔드로 함께 전송되어 장학금별 조건 판정에 반영된다.

④ 결과 출력 및 다음 단계 안내

챗봇은 사용자의 평점과 봉사 여부를 바탕으로 신청 가능한 장학금 목록(예: 자유, 향준, 협동, 정진)을 제시한다.
또, 각 장학금의 지원비율과 필요 조건(예: 평점, 봉사시간 등)을 상세히 안내한다.
마지막으로 "다른 서비스를 이용하시겠습니까?"라는 메시지를 출력해
사용자가 초기 메뉴로 돌아가거나 다른 기능을 선택할 수 있도록 한다.

향후 개선 방향

1. **자연어 처리 기능(NLP) 추가:** IBM Watson API를 활용해 사용자의 키워드 입력을 인식하여 버튼식 인터페이스 외에도 자유로운 질의응답이 가능하도록 확장할 예정이다.

2. **데이터 관리 고도화:** 현재는 로컬 JSON 형태로 관리 중이지만, 이후 서버와의 연동을 통해 실시간 학사 일정 변경에도 자동으로 반영될 수 있도록 개선할 계획이다.

3. **반응형 UI 적용:** 모바일 및 태블릿 환경에서도 동일한 사용자 경험을 제공하기 위해

반응형 디자인을 적용할 예정이다.

4. **컴포넌트 세분화:** MessageList, OptionButtons, InputArea를 별도 파일로 분리하여 코드 가독성과 유지보수성을 높일 예정이다.

결론

현재까지 HI-Chat의 프론트엔드 파트는 버튼식 인터페이스를 기반으로 한 기본 챗봇 구조를 완성하였다. 또, 사용자가 직관적으로 서비스를 이용할 수 있도록 UI 설계에 중점을 두었다. 이번 단계에서는 로컬 데이터를 활용한 기능 구현과 대화 흐름 제어를 안정화하는 데 집중하였다. 이후에는 NLP 기능과 서버 연동을 통해 보다 확장성 있는 학사 정보 챗봇으로 발전시킬 계획이다

백엔드는 인메모리 데이터로 동작하는 가벼운 REST API를 구축해 검증·예외처리·CORS 등 기본 인프라를 안정화했다. 다음 단계에서는 JDBC 기반 DB 연동(예: MySQL/H2 → 추후 PostgreSQL 확장)으로 전환하여 시험·성적·장학금 스키마를 설계하고, repository 계층과 트랜잭션 관리, 마이그레이션(Flyway) 등을 도입해 데이터 소스 다양화와 영속성을 확보할 예정이다. 필요 시 API 버저닝과 인증을 추가해 확장성과 보안을 강화할 것이다.

*참고문헌

1. 이창희, 이해영, 김인택. (2018). 기록정보서비스를 위한 메신저 기반의 챗봇 프로토타입 개발 연구: 명지대학교 대학사료실을 중심으로. 정보관리학회지 , 35(3), 215-244.

2. 임지수, 김다영, 조수민, 유견아.(2018). 챗봇과 규칙기반 전문가시스템을 이용한 애견 건강관리 시스템의 개발, 디지털콘텐츠학회논문지 ,19(11), 2059-2066