응급 처치 AI 에이전트 앱

First Aid AI Agent App

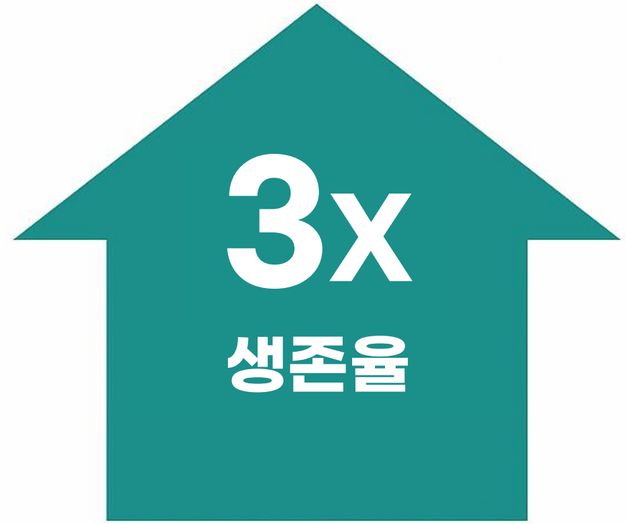
20191732 박준후
20227134 이창석

목차

프로젝트 개요

심폐소생술(CPR)의 효과

심정지 발생 시
즉각적인 심폐소생술

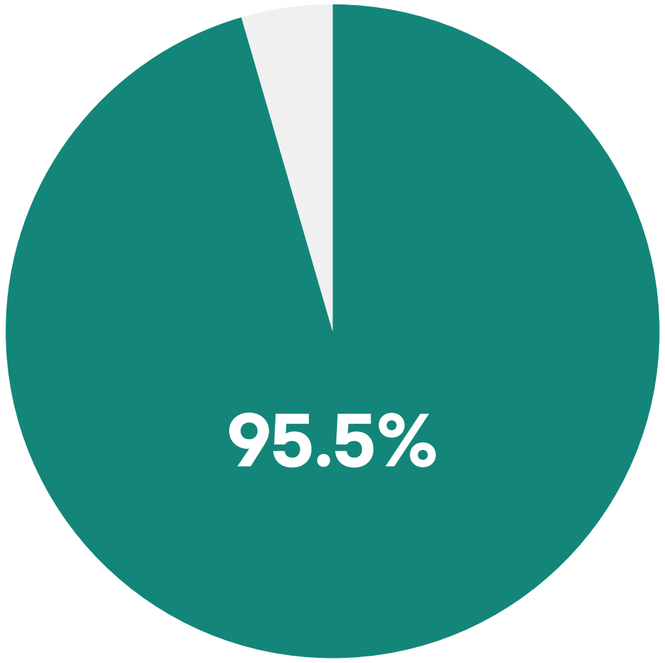


제세동 1분 지연

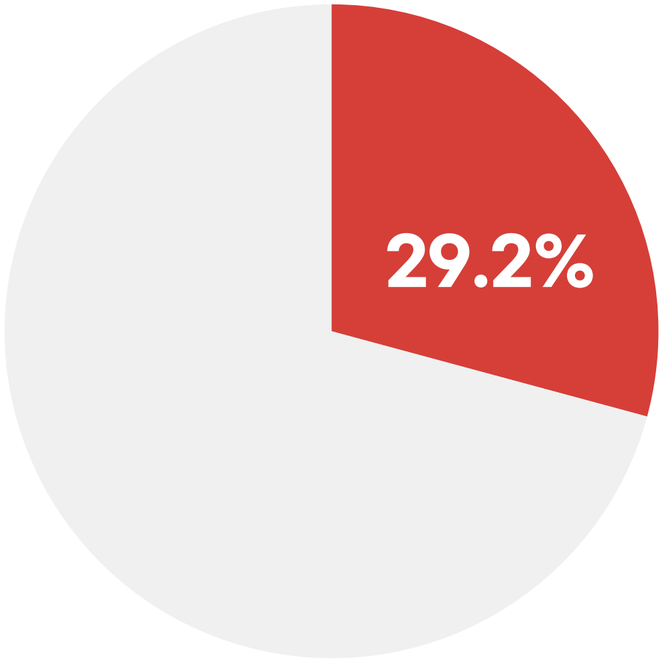
출처: 미국적십자사, After the lights and sirens: Patient access delay in cardiac arrest

실제 CPR 시행률 저조

2022년 대한민국 국민의 95.5% CPR 인지
BUT, 일반인 CPR 시행률 29.2%



2022 CPR 인지율

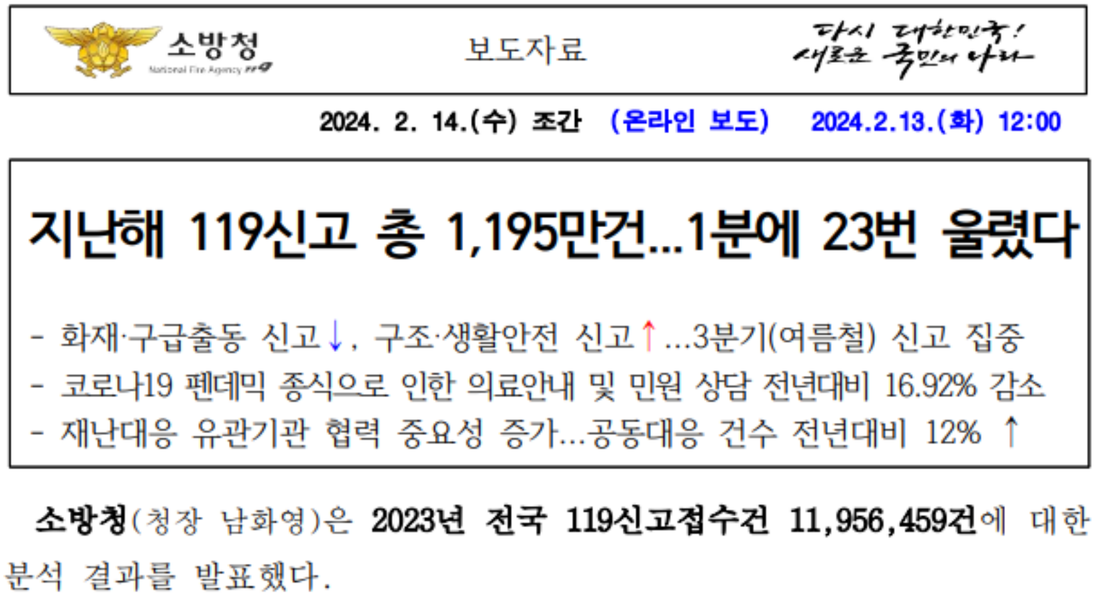


2022 일반인 CPR 시행률

출처: KOSIS 국가통계포털

현재의 문제점

1. 많은 양의 119 신고



소방청 National Fire Agency 보도자료 다시 대한민국! 새로운 국민의 나라

2024. 2. 14.(수) 조간 (온라인 보도) 2024.2.13.(화) 12:00

지난해 119신고 총 1,195만건...1분에 23번 울렸다

- 화재·구급출동 신고↓, 구조·생활안전 신고↑...3분기(여름철) 신고 집중
- 코로나19 펜데믹 종식으로 인한 의료안내 및 민원 상담 전년대비 16.92% 감소
- 재난대응 유관기관 협력 중요성 증가...공동대응 건수 전년대비 12% ↑

소방청(청장 남화영)은 **2023년 전국 119신고접수건 11,956,459건**에 대한 분석 결과를 발표했다.

응급상황에 대한 신속한 대응 불가능

출처: 소방청

현재의 문제점

2. 대다수의 119 신고가 응급전화가 아님

연도	총계	현장 출동관련 신고				유관기관이첩	의료안내 및 민원상담	무응답	오접속	기타
		화재	구조	구급	생활안전(대민출동)					
'23년	11,956,459	439,151	855,537	3,202,268	885,789	108,087	3,289,991	1,441,890	969,121	764,625

- 비응급 또는 단순 민원성 신고 다수
- 불필요한 출동 및 신고로 인해 진짜 응급환자 대응 지연

출처: 소방청

본 프로젝트의 목표

I. 구급 신고를 전담하는 앱을 제공하여 불필요한 신고 방지

II. AI 에이전트 대응을 통한 인력 낭비 방지

III. 응급도에 따른 신속 · 정확한 신고 처리

IV. 상황에 맞는 응급처치 안내

I. 응급의료 접근성 향상

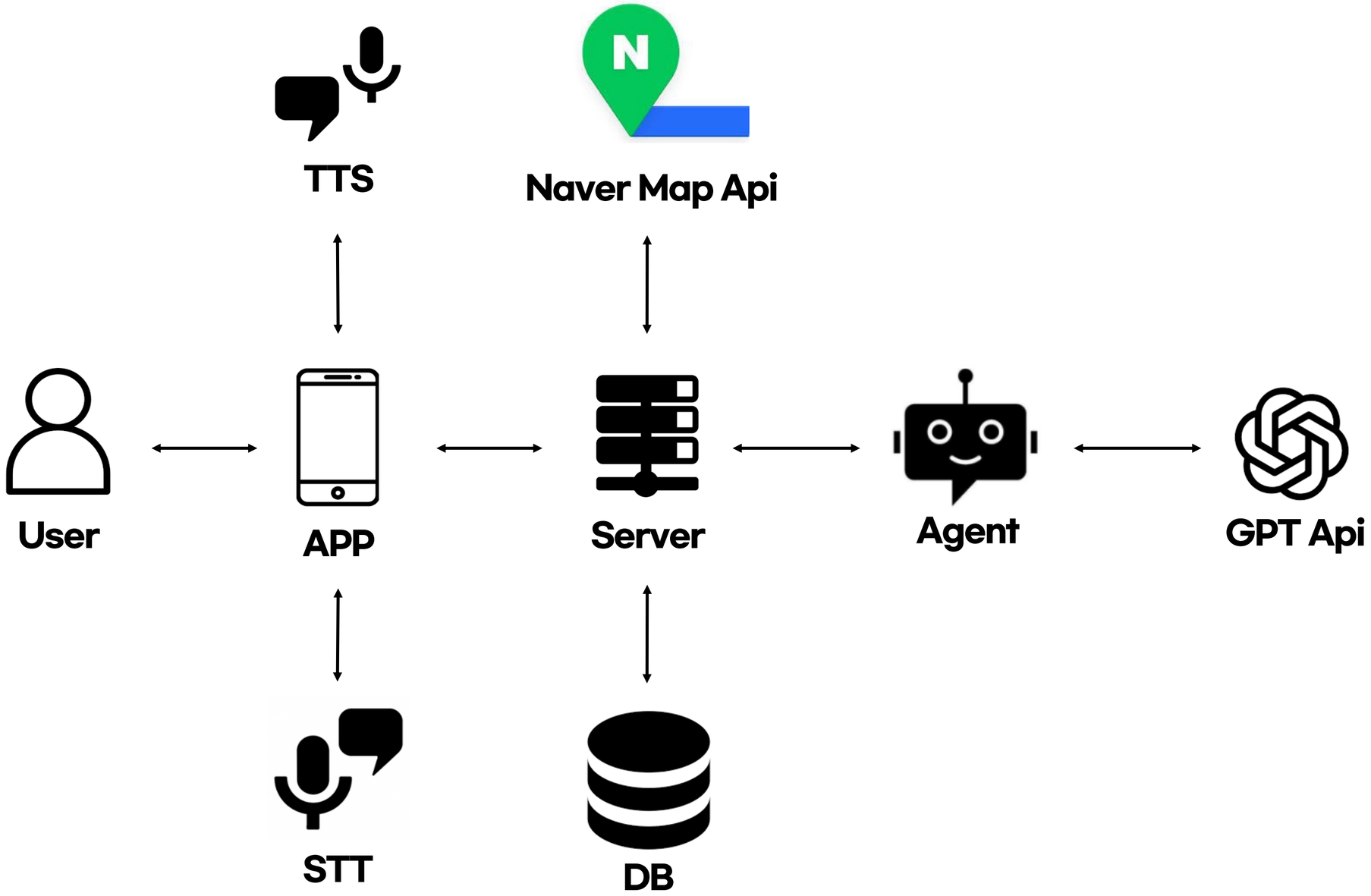
II. 시민 구조 역량 강화

III. 실제 구조 효율성 향상

※ 환자의 생존율 & 회복 가능성 향상

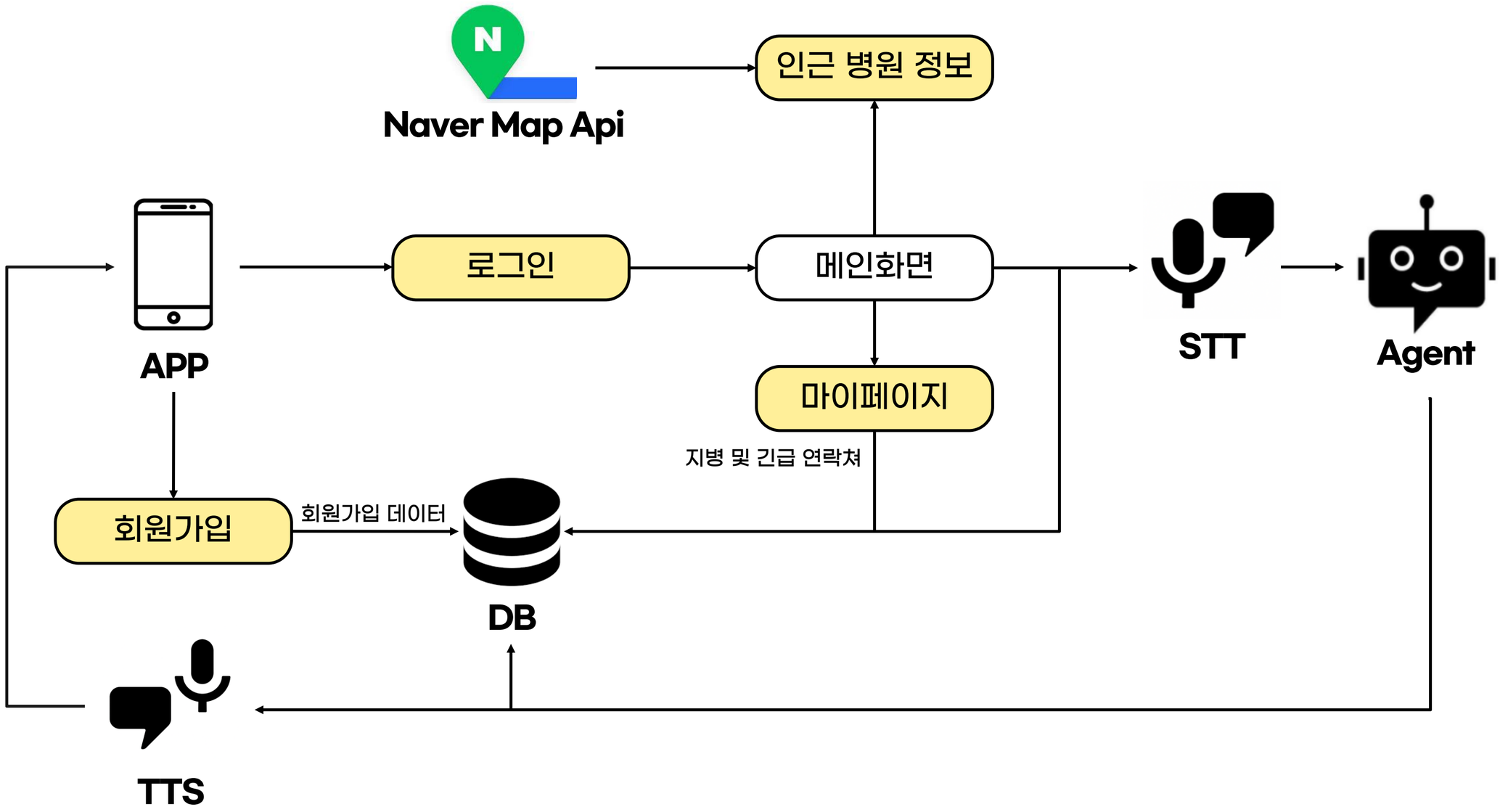
시스템 아키텍처

시스템 아키텍쳐

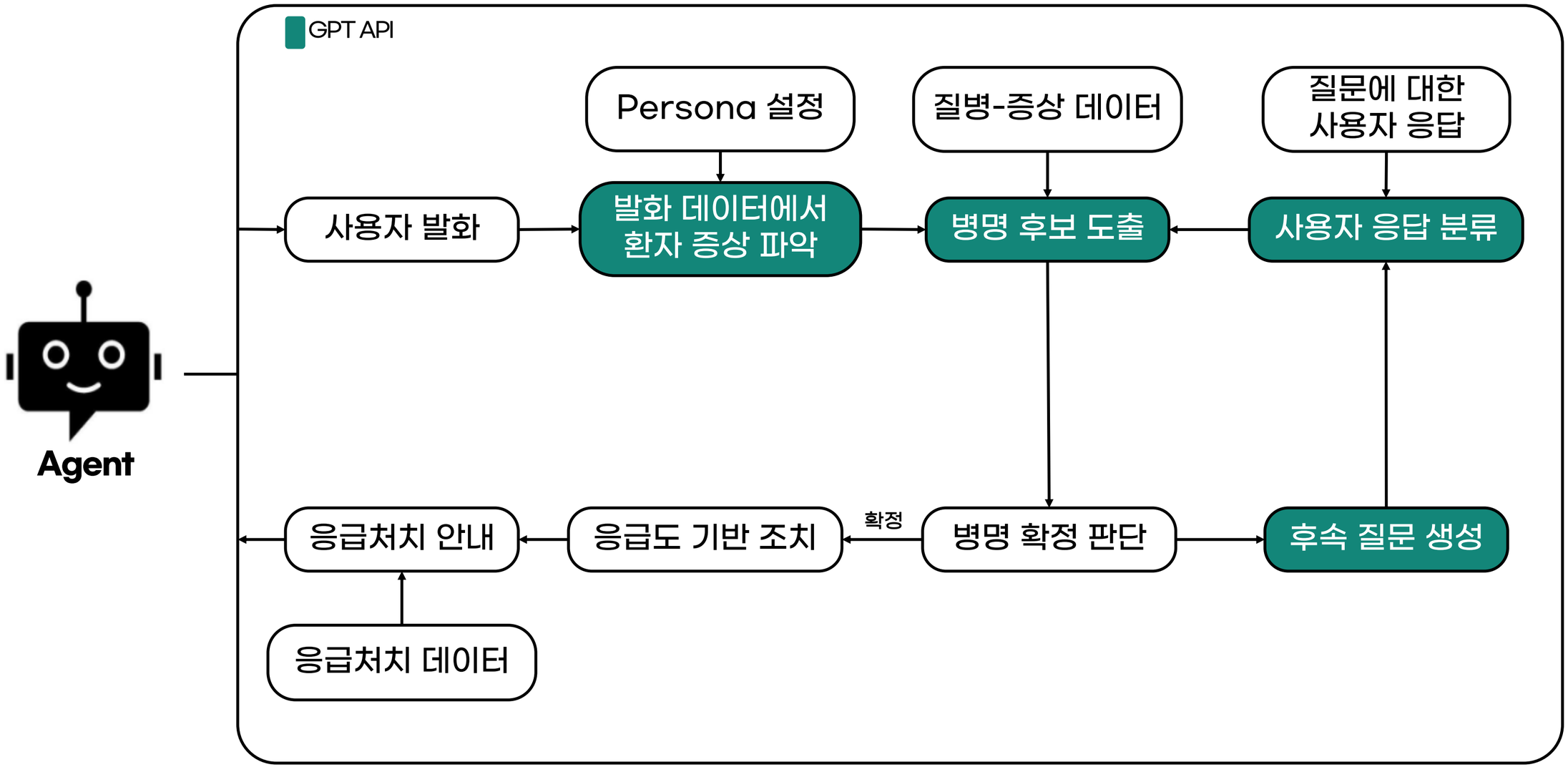


주요 기능

주요 기능 - APP, 백엔드


Naver Map Api
인근 병원 정보
APP
로그인
메인화면
STT
Agent
마이페이지
지병 및 긴급 연락쳐
회원가입
회원가입 데이터
DB
TTS

주요 기능 - AI 에이전트


GPT API
Persona 설정
질병-증상 데이터
질문에 대한
사용자 응답
사용자 발화
발화 데이터에서
환자 증상 파악
병명 후보 도출
사용자 응답 분류
Agent
응급처치 안내
응급도 기반 조치
확정
병명 확정 판단
후속 질문 생성
응급처치 데이터

성능 테스트 결과

성능 테스트 결과 - APP, 백엔드

- **테스트 방식:**
  - 모듈 단위 테스트 (Unit Test)
    - 각 기능이 독립적으로 정상 작동하는지 개별 확인함
  - 통합 테스트 (Integration Test)
    - 전체 시스템을 통합한 상태에서 실제 시나리오 기반 흐름을 점검함
    - 회원 가입부터 응급 전화까지 모든 UI페이지 연결이 정상 작동하는지 테스트 함
    - 앱과의 연계가 자연스럽게 정상 작동하는지 테스트함

성능 테스트 결과 - APP, 백엔드

항목	결과	비고
마이페이지 생성	✅ 성공	입력, 버튼 정상 작동
회원가입 페이지	✅ 성공	다른 UI 와 연결 양호, 정보 입력 양호
메인페이지	✅ 성공	버튼 정상 작동
지도 페이지	✅ 성공	주변 병원 위치 표시 정상적
회원정보 수정	✅ 성공	정보 입력시 수정 확인
응급전화 페이지	✅ 성공	외부 API(ChatGPT)와 정상적인 소통 확인
정보 입력 DB	✅ 성공	정보 저장 정상 작동
정보 호출 DB	✅ 성공	정상적으로 정보 호출 작동
NAVER MAP API 호환	✅ 성공	Naver MAP 정상 작동
TTS/STT 호환	✅ 성공	TTS/TTS 정상 작동

성능 테스트 결과 - AI 에이전트

- **테스트 방식:**
  - 모듈 단위 테스트 (Unit Test)
    - 각 기능이 독립적으로 정상 작동하는지 개별 확인함
  - 통합 테스트 (Integration Test)
    - 전체 시스템을 통합한 상태에서 실제 시나리오 기반 흐름을 점검함
    - 총 12개의 질병 케이스(심정지, 골절, 화상 등)를 설정하여 실제 대화 흐름을 시뮬레이션함
    - 예외 상황(Fallback) 및 종료 흐름도 포함하여 테스트 진행

성능 테스트 결과 - AI 에이전트

항목	결과	비고
증상 추출 정확도	✅ 성공	발화 데이터에서 GPT가 정확히 증상 인식
병명 후보군 생성	✅ 성공	질병-증상 매핑 기반 정상 작동
follow-up 질문 생성	✅ 성공	중복 없이 조건에 맞게 정상 생성
사용자 응답 분류	✅ 성공	예/아니오/모르겠음 정확히 분류
병명 확정 판단	✅ 성공	결정적 증상에서 확정
응급도에 따른 119 분기	✅ 성공	긴급, 응급, 비응급 분기 흐름 정상
신고 메시지 생성 및 출력	✅ 성공	JSON 형식으로 정확히 출력
응급처치 안내 출력	✅ 성공	병명 기반 응급처치 안내 정상
질문 상한 초과 fallback 처리	✅ 성공	질문 15회 초과 시 자동 중단
중복 질문 fallback 처리	✅ 성공	동일 질문 3회 시 진단 종료 정상 작동
API 오류 대비 예외 처리	✅ 성공	GPT timeout 시 안내 출력
강제 종료 처리	✅ 성공	'종료' 입력 시 흐름 종료

주요 알고리즘 및 설계 방법

주요 알고리즘 및 설계 방법

1. 기능 실행 시, 초기 증상 분석 및 초기 병명 추론

- 기능 실행 시, 에이전트가 “환자의 상태를 말씀해주세요. 어떤 증상이 있나요?” 질문
- 사용자 대답 시, 에이전트 - 사용자 대화 히스토리를 chat_history에 저장
- chat_history를 분석해 환자의 증상 파악
- GPT가 해당 증상들과 질병-증상 매핑 정보를 이용해 상태(확정/진행중), 누적 증상, 병명 후보 파악
- GPT 응답을 파싱하여 상태(확정/진행중), 누적 증상, 병명 후보 추출
- 병명이 확정되지 않은 경우 ‘진행중’ 상태로 follow-up 질문 단계로 이동

주요 알고리즘 및 설계 방법

2. 후속 질문(follow-up) 생성

- '진행중' 상태일 때 후속 질문 생성
- 병명 후보들을 응급도 순으로 정렬 (긴급 > 응급 > 비응급)
- 병명 후보들의 증상들을 질병-증상 매핑 정보에서 가져옴
- 병명 후보들의 증상들 중 '누적 증상 + 스킵증상'에 포함된 증상들은 제거
- 응급도 높은 질병의 아직 확인되지 않은 증상 하나의 존재 여부를 묻는 질문 생성
  - 좋은 예시: "의식을 잃었나요?", "가슴 통증이 있나요?"
  - 나쁜 예시: "의식이 있나요?", "가슴 통증이 없나요?"
- GPT는 아래 포멧을 지킴:
  - 다음 질문 대상 증상: 팔에 통증
  - 추가 질문: 팔에 통증이 있나요? 추가적인 증상이 있다면 편하게 말씀해주세요.

주요 알고리즘 및 설계 방법

3. 후속 질문(follow-up) 처리

- 후속 질문이 중복되었는지 확인 → 중복이면 최대 3회까지 후속 질문 재생성 시도
  - 재생성 시도 3회 초과 시: 질문 생성 실패 안내 → 119 연결 권유
- 후속 질문 수가 10회, 15회인 경우:
  - 10회 초과 → 119 연결 권유
  - 15회 초과 → 병명 확정 실패 및 응급도 기반 안내
- 질문 중복 여부 확인 후, 중복이 아닐 시 사용자에게 질문
- 자연어 사용자 응답을 '예/아니오/모르겠음'으로 분류
  - "예" → 다음 질문 대상 증상 '누적 증상'에 추가
  - "아니오/모르겠음" → 다음 질문 대상 증상 '스킵 증상'에 추가
- GPT가 누적 증상과 병명 후보들의 질병-증상 매핑 정보를 이용해 병명 추론

주요 알고리즘 및 설계 방법

4. 병명 확정 조건 및 처리

- 병명 확정 성공 조건:
  - 질문 횟수 15회 이하 && 병명 후보가 1개 남은 경우, 해당 병명 후보를 확정 지음
    - 병명 확정 성공 시, '확정' 상태로 업데이트 후 응급도 기반 조치 실행

- 병명 확정 실패 조건
  - 질문 횟수 15회 초과 또는 질문 가능한 증상이 더 이상 없는 경우
    - 지금까지 확인된 누적 증상과 질병-증상 매핑 데이터를 기반으로 GPT에 병명 추론 요청
    - 병명이 하나로 확정되지 않을 경우 '병명 확정 실패'로 처리
  - 추가 질문 생성 실패한 경우 (3회 연속 중복 질문 생성 또는 GPT 오류 등), '병명 확정 실패'로 처리
  - 병명 확정 실패 시 병명 후보, 후보의 최고 응급도 출력 후 119 연결 or 처음 부터 재시작 선택지 제공

주요 알고리즘 및 설계 방법

5. 응급도 기반 후속 조치

- '확정' 상태일 때, 응급도에 따라 조치를 취한다
  - 긴급 상황:
    1. 119 신고,
    2. 상세 위치 파악 후 119 전달
    3. 응급처치 안내
  - 응급 상황: 119 신고 사용자 의사 확인
    - 수락 시: 상세 위치 파악 후 119신고 + 응급처치 안내
    - 거절 시: 바로 응급처치 안내
  - 비응급 상황: 바로 응급처치 안내
- 119 신고는 백엔드에서 진행, 에이전트에서 백엔드로 환자의 증상, 병명, 상세 위치 데이터 전달

주요 알고리즘 및 설계 방법

6. 응급처치 안내

- 병명 확정 후 해당 병명.json 파일에서 응급처치 데이터 로딩
  - 내부 구조: intro, precaution, checklist, instruction, closing
- 응급처치 단계는 다음 순서로 안내 됨
  1. instruction에 checklist가 존재하는 경우:
     - checklist를 사용자에게 질문 후,
       - 응답이 “예”일 경우 yes_instructions에 해당 instruction 저장
       - 응답이 “아니요” 또는 “모르겠어요”일 경우: 해당 instruction은 무시
  2. checklist가 끝나면 intro 출력
  3. 주의사항 안내
  4. checklist가 없는 instruction 및 yes_instructions에 있는 instruction 출력
  5. closing멘트 출력

주요 알고리즘 및 설계 방법

disease_symptom.json:

```
"심장마비": {
  "emergency_level": "긴급",
  "symptoms": [
    "가슴 통증",
    "어깨, 팔, 등, 목, 턱, 치아 통증",
    "상복부로 퍼지는 통증",
    "식은땀",
    "피로",
    "속쓰림",
    "소화불량",
    "어지럼증",
    "갑작스러운 현기증",
    "메스꺼움",
    "숨 가쁨"
  ]
},
```

골절.json:

```
{
  "intro": "상황에 맞는 응급처치를 안내하겠습니다.",
  "precaution": "환자를 가능한 한 움직이지 마세요. 꼭 필요한 경우 외에는 안정된 자세로 두고 의료 지원을 기다리세요.",
  "checklist": [
    {
      "question": "출혈이 있나요?",
      "instruction": "멸균 붕대, 깨끗한 천 또는 옷으로 상처 부위를 압박하여 출혈을 멈추세요."
    },
    {
      "question": "부목을 사용할 수 있나요?",
      "instruction": "뼈를 정렬하지 말고, 골절 부위 위아래에 부목을 댄 뒤, 패딩을 추가해 통증을 줄이세요."
    },
    {
      "question": "얼음찜질이 가능한가요?",
      "instruction": "얼음은 수건이나 천에 싸서 피부에 직접 닿지 않도록 하여 얼음찜질을 하세요."
    },
    {
      "question": "환자가 실신하거나 짧고 빠른 호흡을 하나요?",
      "instruction": "머리를 몸통보다 낮게 눕히고 다리를 살짝 들어 올려 쇼크를 완화하세요."
    }
  ],
  "closing": "이상으로 응급처치 안내를 마치겠습니다. 빠른 회복을 바랍니다."
}
```

각각의 역할

각각의 역할

개발자

박준후 (팀장)
AI 에이전트 개발

이창석 (팀원)
APP 개발, 백엔드 개발

조력자

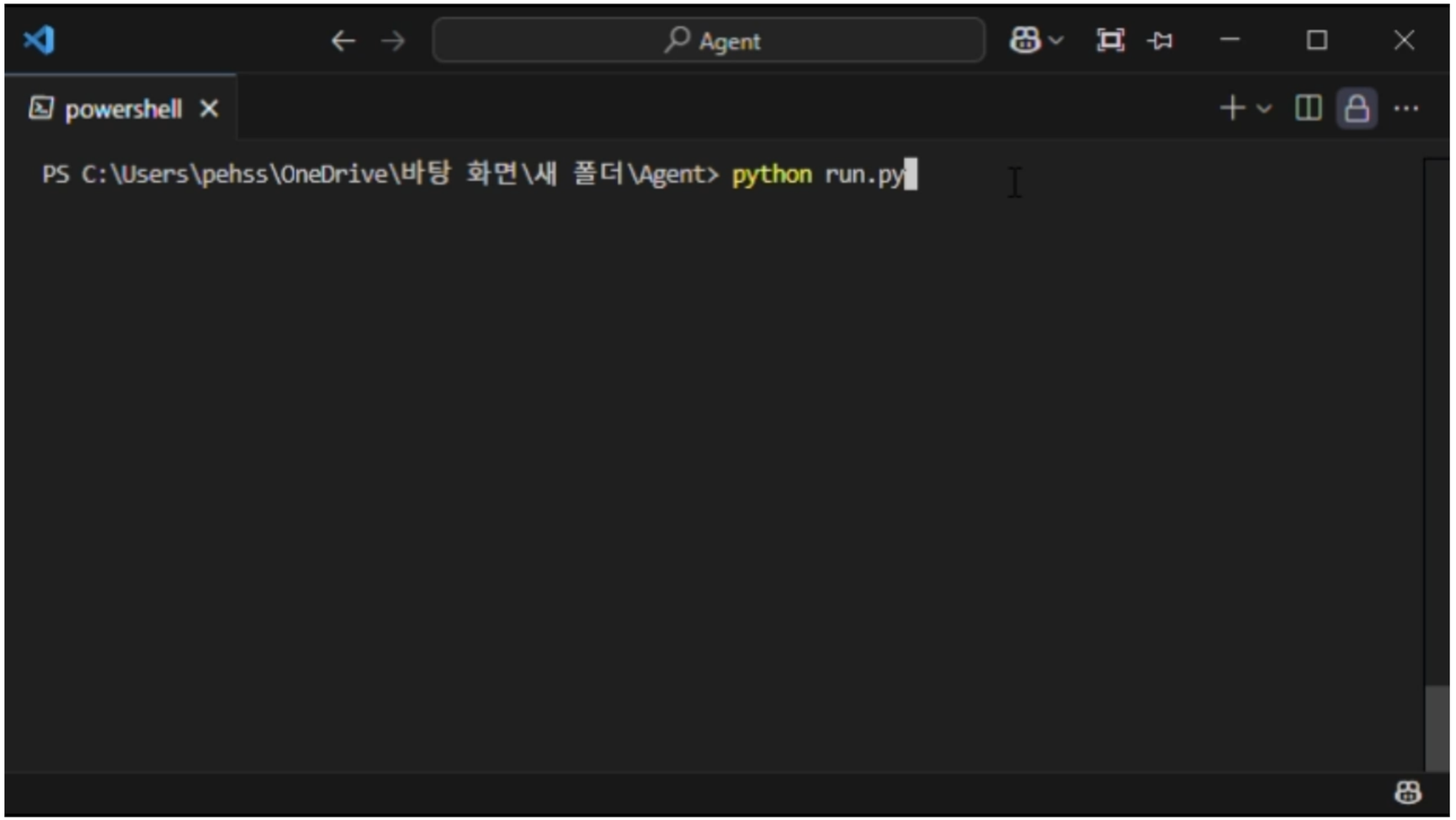
박천음 교수님
피드백

김민상 멘토님
피드백

데모

데모 - APP

데모 - AI 에이전트(CLI)



향후 계획

향후 계획

- **RAG 기반 응급처치 검색 시스템 구현:**
  - 방학 중 지도 교수님 랩실과 협력하여 응급처치 검색 시스템 구축 예정

- **앱, 백엔드, AI 에이전트 연동:**
  - 방학 중 실제 모바일 환경에서 AI 에이전트가 작동하도록 앱·서버 연동 예정

- **사용자 인터페이스 고도화 및 다중 세션 구조 개발:**
  - 2학기 중 실제 서비스화를 고려한 UI 개선 및 다중 사용자 대응 구조 설계 예정

- **장난 전화 탐지 기능 개발:**
  - 2학기 중 장난/오인 신고 여부를 판별하는 탐지 기능 개발 예정

질의 응답

감사합니다