



# Chatbot Optimization

신한DS 금융 SW아카데미 3기

정민교

# Inspiration

## Generative Agents: Interactive Simulacra of Human Behavior

Joon Sung Park  
Stanford University  
Stanford, USA  
joonspk@stanford.edu

Joseph C. O'Brien  
Stanford University  
Stanford, USA  
jobrien3@stanford.edu

Carrie J. Cai  
Google Research  
Mountain View, CA, USA  
cjcai@google.com

Meredith Ringel Morris  
Google Research  
Seattle, WA, USA  
merrie@google.com

Percy Liang  
Stanford University  
Stanford, USA  
pliang@cs.stanford.edu

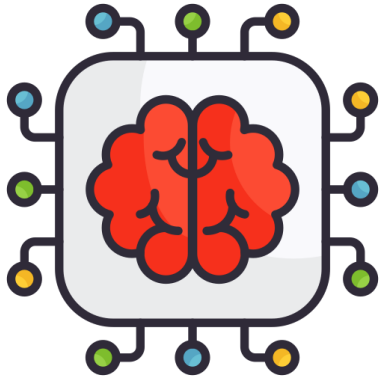
Michael S. Bernstein  
Stanford University  
Stanford, USA  
msb@cs.stanford.edu





**챗봇을 이용한 서비스를 만들 때 어떻게 하면 효과적으로 운영할 수 있을까?**

# Development Purpose



Chatbot Memory



Reducing tokens usage



Studying JDBC

# Chatbot Memory



채팅을 종료하시려면 q또는 quit를 입력해 주세요.

user> 앞으로 말 끝에 냥을 붙여서 대답해

GPT> 알겠어냥! 질문이 있으면 언제든지 물어보라냥!

user> 안녕?

GPT> 안녕하세요! 무엇을 도와드릴까요?

**Memory Off**



채팅을 종료하시려면 q또는 quit를 입력해 주세요.

user> 앞으로 말 끝에 냥을 붙여서 대답해

GPT> 알겠어냥! 무엇을 도와드릴까냥?

user> 안녕?

GPT> 안녕하냥! 오늘 기분은 어떠냥?

**Memory On**

# Reducing tokens usage : Token

Tokens	Characters
35	152

Many words map to one token, but some don't: indivisible.

Sequences of characters commonly found next to each other may be grouped together: 1234567890

Token ↑ = execution time ↓, price ↑

# Reducing tokens usage : Cache



채팅을 종료하시려면 q또는 quit를 입력해 주세요.

user> 점심 뭐먹으면 좋을까? 하나만 추천해줘

GPT> 점심으로는 닭가슴살 샐러드를 추천해드립니다. 영양이 균형 잡히고, 가볍지만 포만감을 줄 수 있어서 일상에서 활력을 유지하는 데 도움이 됩니다.

user> 점심 뭐먹으면 좋을까? 하나만 추천해줘

GPT> 점심으로 샐러드를 추천합니다. 건강에 좋고 가벼워서 오후 업무에 지장을 주지 않으면서도 다양한 재료를 활용해 영양을 골고루 섭취할 수 있습니다.

## Cache Off

Execution time : 2.76s



채팅을 종료하시려면 q또는 quit를 입력해 주세요.

user> 점심 뭐먹으면 좋을까? 하나만 추천해줘

GPT> 비빔밥을 추천드립니다! 건강에 좋은 재료들을 다양하게 섭취할 수 있어요.

user> 점심 뭐먹으면 좋을까? 하나만 추천해줘

GPT> 비빔밥을 추천드립니다! 건강에 좋은 재료들을 다양하게 섭취할 수 있어요.

## Cache On

Execution time : 0.04s

# Reducing tokens usage : Summary



```
user> 빛의 삼원색에 관하여 알려줘
GPT> 빛의 삼원색은 빨강, 초록, 파랑을 말합니다. 이 세 가지
색을 혼합하면 다양한 다른 색을 만들어낼 수 있습니다. 이러한
혼합 과정을 더 자세히 알고 싶으신가요?
user> 색의 삼원색과의 차이가 뭐야?
GPT> 색의 삼원색은 물감이나 프린터 잉크 등을 혼합하여 다양한
색을 만들어내는 기본적인 색상 모델을 말합니다. 빛의 삼원색은
빨강, 초록, 파랑을 조합하여 다양한 색을 만들어내는 것을 말합
니다. 물감이나 잉크는 빛을 흡수하는 반면, 빛의 삼원색은 빛을
반사하거나 흡수하여 다양한 색을 만들어내는데 사용됩니다.
user> 좀더 자세히 알려줘
GPT> 물감이나 프린터 잉크를 사용하는 색의 삼원색은 시안, 마
젠타, 옐로우로 구성되어 있습니다. 시안은 청록색에 가까운 파란
색, 마젠타는 붉은 보라색에 가까운 분홍색, 옐로우는 노란색에
가까운 주황색을 나타냅니다. 이 세 가지 색을 조합하여 다양한
색상을 만들어내는 것이 가능합니다. 빛의 삼원색과는 다르게 색
의 삼원색은 빛을 흡수하는 방식으로 색을 혼합하여 색을 만들어
내는 것이 특징입니다.
```

**Original**

Total tokens : 573



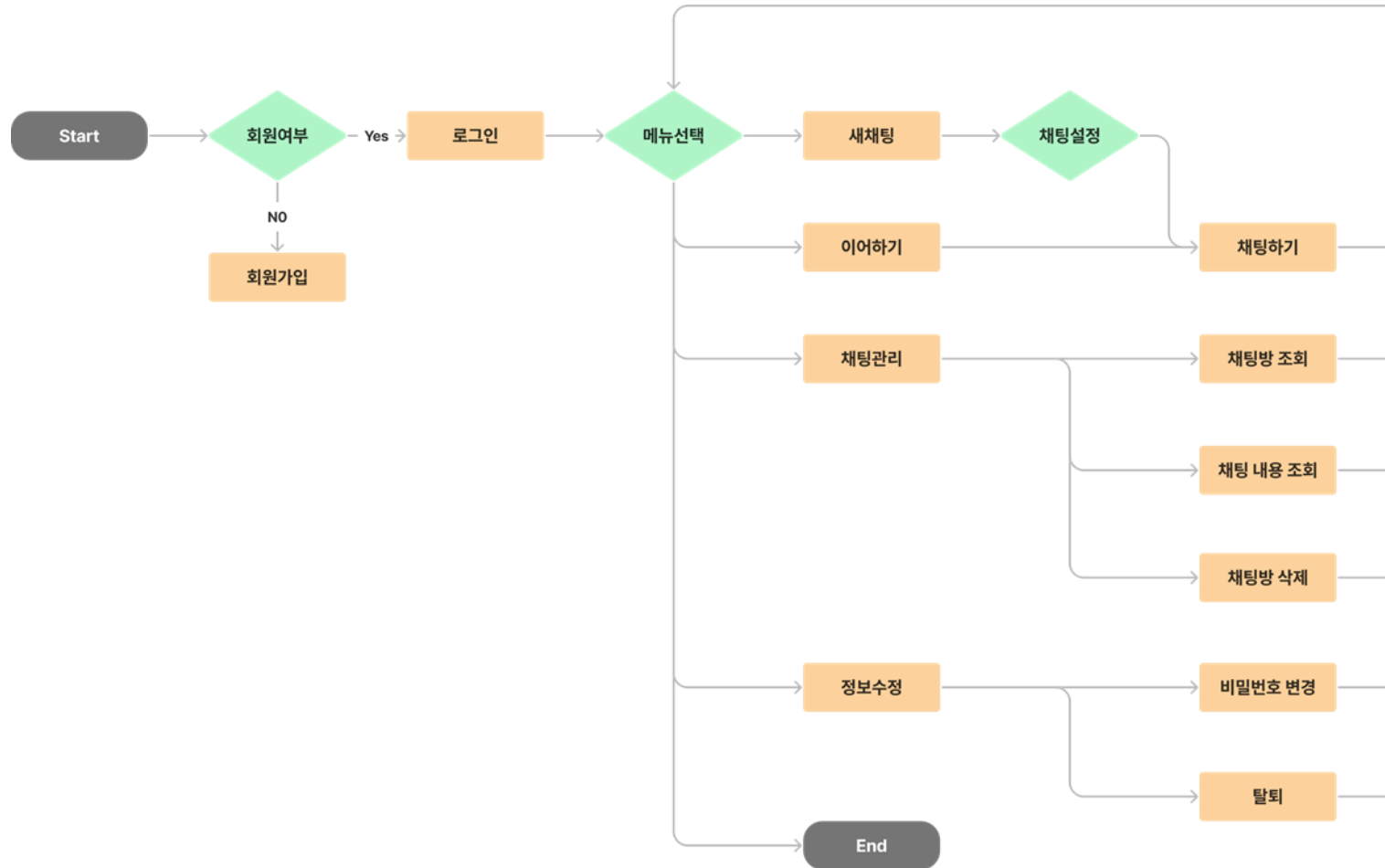
빛의 삼원색은 빨강, 초록, 파랑이며, 이들을 혼합하여 다양한  
색을 만듭니다. 색의 삼원색은 시안, 마젠타, 옐로우이며, 이들  
은 물감이나 잉크 형태로 색을 흡수하는 방식으로 색을 혼합합니  
다. 두 시스템의 주요 차이는 빛의 삼원색이 빛을 반사하거나 흡  
수하여 색을 생성하는 반면, 색의 삼원색은 물질이 빛을 흡수하여  
색을 혼합하는 방식에 있습니다.

**Summary**

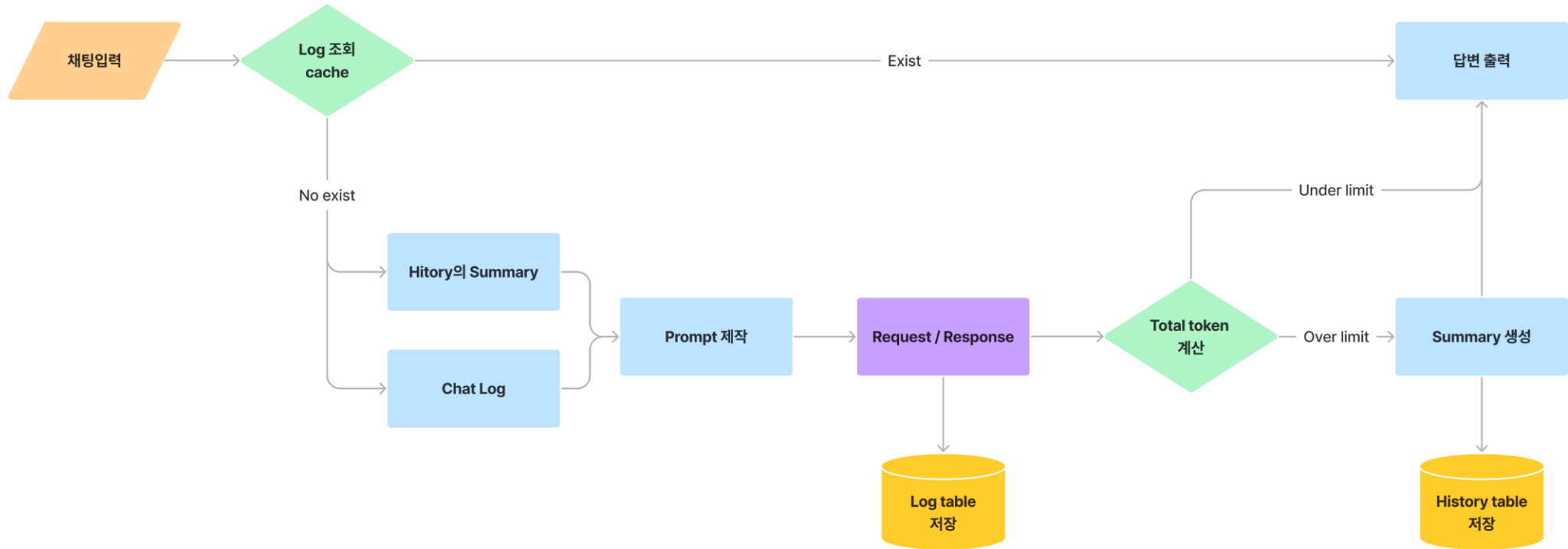
Total tokens : 201




# User flow








# Service logic







# ERD

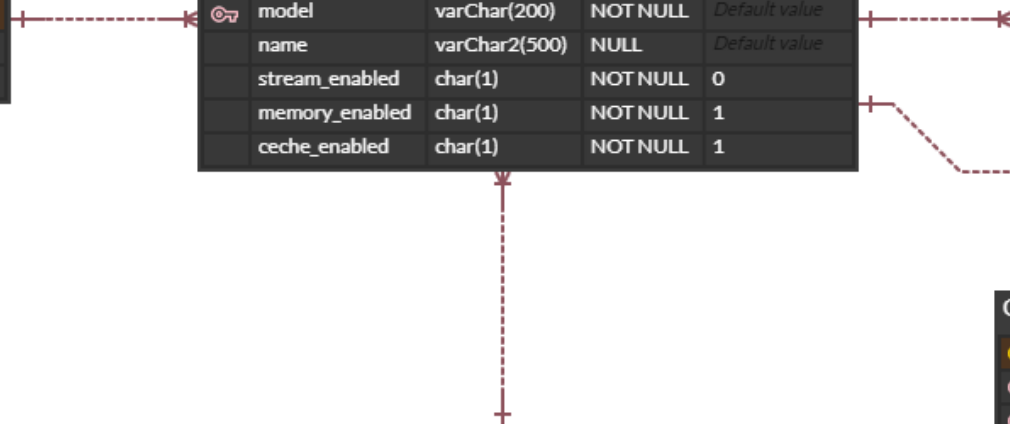
Users				
	id	varChar2(100)	NOT NULL	Default value
	password	varChar2(100)	NOT NULL	Default value
	is_active	char(1)	NOT NULL	1

Chat				
	chat_id	number	NOT NULL	Default value
	user_id	varChar2(100)	NOT NULL	Default value
	model	varChar2(200)	NOT NULL	Default value
	name	varChar2(500)	NULL	Default value
	stream_enabled	char(1)	NOT NULL	0
	memory_enabled	char(1)	NOT NULL	1
	ceche_enabled	char(1)	NOT NULL	1


History				
	history_id	number	NOT NULL	Default value
	chat_id	number	NOT NULL	Default value
	deps	number	NOT NULL	0
	summary	varChar2(4000)	NULL	Default value
	prompt_tokens	number	NOT NULL	Default value
	completion_tokens	number	NOT NULL	Default value

Models				
	name	varChar(200)	NOT NULL	Default value
	company	varChar2(1000)	NULL	Default value
	price_per_ptoken	number	NOT NULL	Default value
	price_per_ctoken	number	NOT NULL	Default value

Chat_log				
	log_id	number	NOT NULL	Default value
	chat_id	number	NOT NULL	Default value
	history_id	number	NULL	Default value
	request	varChar2(4000)	NULL	Default value
	response	varChar2(4000)	NULL	Default value
	prompt_tokens	number	NOT NULL	Default value
	completion_tokens	number	NOT NULL	Default value
	create_at	timestamp	NOT NULL	systemtimestamp



# Program views : Home



```
=====
                        Home
=====

1.로그인
2.등록
3.종료
입력>
```

Home



```
-----
                        Registration
-----


id> qwer
password> 1111
회원가입을 성공하였습니다.
```



```
-----
                        Registration
-----

id> qwer
password> 1
비밀번호는 영문과 숫자로만 이루어져야하며
4자리 이상이어야 합니다.
```

Registration



```
-----
                        Login
-----

id> qwer
password> 1111
로그인 성공
```



```
-----
                        Login
-----

id> qwer
password> 2222
id 또는 password가 잘못되었습니다.
```

Login

# Program views : Menu

```
=====
Menu
=====
1. 새 채팅
2. 이어하기
3. 채팅관리
4. 정보수정
5. 로그아웃
입력>
```

Menu

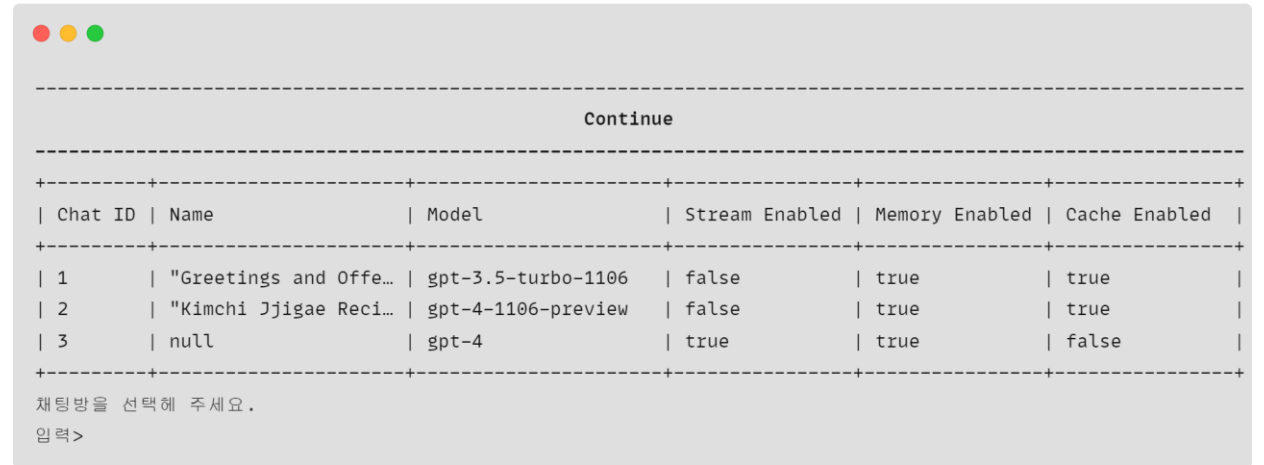
```
-----
New Chat
-----
+-----+-----+-----+-----+
| No | Name                | Company | Price/PToken | Price/CToken |
+-----+-----+-----+-----+
| 1  | gpt-3.5-turbo-1106  | openAI  | 0.0010/1k    | 0.0020/1k    |
| 2  | gpt-4               | openAI  | 0.0300/1k    | 0.0600/1k    |
| 3  | gpt-4-1106-preview  | openAI  | 0.0100/1k    | 0.0300/1k    |
+-----+-----+-----+-----+
사용하실 모델을 골라주세요. [defalut: gpt-3.5-turbo-1106]
입력> 3
stream 기능을 사용하시겠습니까? [defalut: false] (y/n)
입력>
cache 기능을 사용하시겠습니까? [defalut: true] (y/n)
입력>
memory 기능을 사용하시겠습니까? [defalut: true] (y/n)
입력>
```

Make new chat

# Program views : Menu



Menu



Continue

# Program views : Management

=====

Management

=====

1. 채팅방 조회하기

2. 채팅방 내용보기

3. 채팅방 삭제하기

4. 돌아가기

입력>

## Management

-----

Fetch Chat

-----

Chat ID	Name	Model	Stream Enabled	Memory Enabled	Cache Enabled
1	"Greetings and Offe...	gpt-3.5-turbo-1106	false	true	true
2	"Kimchi Jjigae Reci...	gpt-4-1106-preview	false	true	true
3	null	gpt-4	true	true	false

-----

-----

Fetch Chat Log

-----

조회할 채팅방을 선택해 주세요.  
입력> 2

Log ID	Request	Response	Prompt Tokens	Completion Tokens
3	김치찌개 만드는 법	김치찌개는 한국의 대표적인 김치를 이용한 찌개 요리입니다. 매콤하고 구...	29	
499				
4	아하 고마워	천만에요! 김치찌개 레시피가 도움이 되었으면 좋겠습니다. 요리하실 때 ...	542	
81				

-----

-----

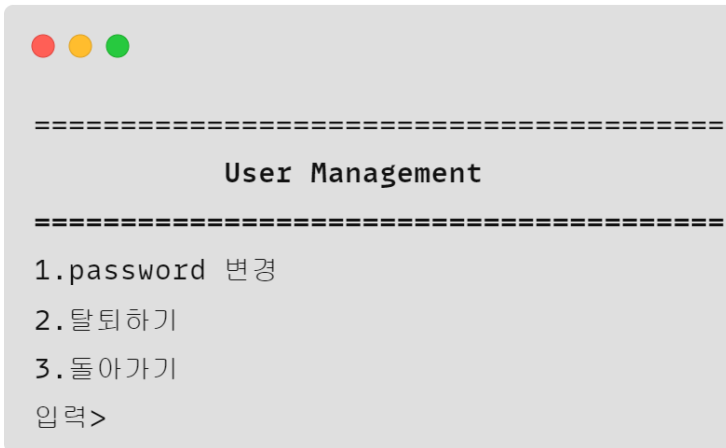
Delete Chat

-----

삭제할 채팅방을 선택해 주세요.  
입력> 3

채팅방이 성공적으로 삭제되었습니다.

# Program views : Management



A terminal window with a light gray background and three colored window control buttons (red, yellow, green) in the top-left corner. The title "User Management" is centered between two dashed horizontal lines. Below the title, a list of three menu items is displayed: "1.password 변경", "2. 탈퇴하기", and "3. 돌아가기". At the bottom, the prompt "입력>" is shown.

```
=====
User Management
=====
1.password 변경
2. 탈퇴하기
3. 돌아가기
입력>
```

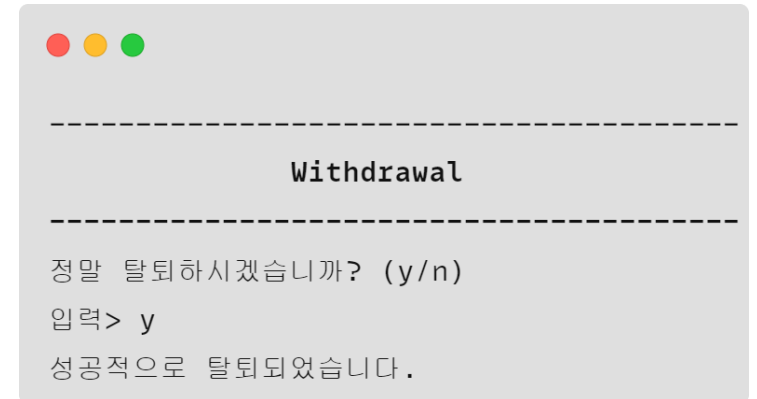
User management



A terminal window with a light gray background and three colored window control buttons (red, yellow, green) in the top-left corner. The title "Update Password" is centered between two dashed horizontal lines. Below the title, a prompt "새로운 비밀번호를 입력해 주세요." is shown. The user input "입력> 2222" is displayed on the next line. The final line shows the confirmation message "성공적으로 비밀번호가 변경되었습니다."

```
-----
Update Password
-----
새로운 비밀번호를 입력해 주세요.
입력> 2222
성공적으로 비밀번호가 변경되었습니다.
```

Update password



A terminal window with a light gray background and three colored window control buttons (red, yellow, green) in the top-left corner. The title "Withdrawal" is centered between two dashed horizontal lines. Below the title, a prompt "정말 탈퇴하시겠습니까? (y/n)" is shown. The user input "입력> y" is displayed on the next line. The final line shows the confirmation message "성공적으로 탈퇴되었습니다."

```
-----
Withdrawal
-----
정말 탈퇴하시겠습니까? (y/n)
입력> y
성공적으로 탈퇴되었습니다.
```

Withdrawal



# Program views : Chat

```
=====
                                Chat Start
=====
+-----+-----+-----+-----+-----+
| Chat ID | Name           | Model           | Stream Enabled | Memory Enabled | Cache Enabled |
+-----+-----+-----+-----+-----+
| 1       | "Greetings and Offe... | gpt-3.5-turbo-1106 | false         | true           | true           |
+-----+-----+-----+-----+-----+

채팅을 종료하시려면 q또는 quit를 입력해 주세요.
user> 안녕
GPT> 안녕하세요! 어떻게 도와드릴까요?
user> 반가워
GPT> 반가워요! 무엇을 도와드릴까요?
user>
```

# Additional features : generate name

```
=====
Chat Start
=====
+-----+-----+-----+-----+-----+-----+
| Chat ID | Name           | Model           | Stream Enabled | Memory Enabled | Cache Enabled |
+-----+-----+-----+-----+-----+-----+
| 1       | null           | gpt-3.5-turbo-1106 | false         | true          | true          |
+-----+-----+-----+-----+-----+-----+

채팅을 종료하시려면 q또는 quit를 입력해 주세요.
user> 안녕
GPT> 안녕하세요! 무엇을 도와드릴까요?
user> 배고파
GPT> 뭘 먹고 싶으세요? 추천해 드릴까요?
user> 점심 추천점
GPT> 어떤 음식을 좋아하세요? 한식, 중식, 양식 중에서 어떤 음식을 원하시나요?
user> 카레 만드는 법 알려줘
GPT> 카레를 만드는 법은 다양하지만, 간단한 레시피를 알려드리겠습니다. 먼저 양파, 당근, 감자 등을 잘게 썰어 준비합니다. 팬에 식용유를 두르고 양파를 볶다가 다진 마늘과 생강을 넣고 볶습니다. ....
즐거운 식사 되세요!
user> q
채팅을 종료중입니다. 잠시만 기다려 주세요.
```

null

```
=====
Chat Start
=====
+-----+-----+-----+-----+-----+-----+
| Chat ID | Name           | Model           | Stream Enabled | Memory Enabled | Cache Enabled |
+-----+-----+-----+-----+-----+-----+
| 1       | "Request for Lunch ... | gpt-3.5-turbo-1106 | false         | true          | true          |
+-----+-----+-----+-----+-----+-----+

채팅을 종료하시려면 q또는 quit를 입력해 주세요.
user> 배고파
GPT> 뭘 먹고 싶으세요? 추천해 드릴까요?
user> 점심 추천점
GPT> 어떤 음식을 좋아하세요? 한식, 중식, 양식 중에서 어떤 음식을 원하시나요?
user> 카레 만드는 법 알려줘
GPT> 카레를 만드는 법은 다양하지만, 간단한 레시피를 알려드리겠습니다. 먼저 양파, 당근, 감자 등을 잘게 썰어 준비합니다....
즐거운 식사 되세요!
user>
```

"Request for Lunch  
Recommendation and Curry  
Recipe Instructions"

# Additional features : Stream

```
-----
| No | Name | Company | Price/PToken | Price/CToken |
|-----|-----|-----|-----|-----|
| 1 | gpt-3.5-turbo-1106 | openAI | 0.0010/1k | 0.0020/1k |
| 2 | gpt-4 | openAI | 0.0300/1k | 0.0600/1k |
| 3 | gpt-4-1106-preview | openAI | 0.0100/1k | 0.0300/1k |
|-----|-----|-----|-----|-----|
사용하실 모델을 골라주세요. [defalut: gpt-3.5-turbo-1106]
입력> 1
stream 기능을 사용하시겠습니까? [defalut: false] (y/n)
입력>
:cache 기능을 사용하시겠습니까? [defalut: true] (y/n)
입력>
memory 기능을 사용하시겠습니까? [defalut: true] (y/n)
입력>

=====
Chat Start
=====
| Chat ID | Name | Model | Stream Enabled | Memory Enabled | Cache Enabled |
|-----|-----|-----|-----|-----|-----|
| 3 | null | gpt-3.5-turbo-1106 | false | true | true |
|-----|-----|-----|-----|-----|
채팅을 종료하시려면 q 또는 quit을 입력해 주세요.
user> |
```

Stream off

```
사용하실 모델을 골라주세요. [defalut: gpt-3.5-turbo-1106]
입력> 2
stream 기능을 사용하시겠습니까? [defalut: false] (y/n)
입력> y
stream 기능을 사용하시면 summary와 cache 기능을 사용할 수 없습니다.
memory 기능을 사용하시겠습니까? [defalut: true] (y/n)
입력>
```

```
=====
Chat Start
=====
| Chat ID | Name | Model | Stream Enabled | Memory Enabled | Cache Enabled |
|-----|-----|-----|-----|-----|-----|
| 9 | null | gpt-4 | true | true | false |
|-----|-----|-----|-----|-----|
채팅을 종료하시려면 q 또는 quit을 입력해 주세요.
user> 안녕
GPT> 안녕하세요! 도와드릴 일이 있나요?
user> 반가워!
GPT> 제도 반갑습니다! 무엇을 도와드릴까요?
user> 대한민국의 수도가 어디야?
GPT> 대한민국의 수도는 서울입니다.
user>
```

Stream on

# Source code : OpenAIRequest

```
private static final String OPENAI_URL = "https://api.openai.com/v1/chat/completions";

private static String sendHttpRequest(String jsonString, String urlString) throws Exception {
    URL url = new URL(urlString);
    HttpURLConnection connection = (HttpURLConnection) url.openConnection();
    connection.setRequestMethod("POST");
    connection.setRequestProperty("Content-Type", "application/json");
    connection.setRequestProperty("Authorization", "Bearer " + DotEnv.getEnv("OPENAI_KEY"));
    connection.setDoOutput(true);

    try (OutputStream os = connection.getOutputStream()) {
        byte[] input = jsonString.getBytes("utf-8");
        os.write(input, 0, input.length);
    }

    StringBuilder response = new StringBuilder();
    try (BufferedReader br = new BufferedReader(new InputStreamReader(connection.getInputStream(), "utf-8"))) {
        String responseLine;
        while ((responseLine = br.readLine()) != null) {
            response.append(responseLine.trim());
        }
    }
    connection.disconnect();
    return response.toString();
}

public JSONObject chatBot(String modelName, String prompt) {
    String jsonString = String.format(
        "{\n\"model\": \"%s\", \"messages\": [{\n\"role\": \"user\", \"content\": \"%s\"}], \"max_tokens\": 500,\n\"temperature\": 0.7}",
        modelName, prompt);

    try {
        String response = sendHttpRequest(jsonString, OPENAI_URL);
        JSONObject jsonResponse = new JSONObject(response);
        return jsonResponse;
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
}
```

chatbot : not stream

```
public String chatBotStream(String modelName, String prompt) {
    String jsonString = String.format(
        "{\n\"model\": \"%s\", \"messages\": [{\n\"role\": \"system\", \"content\": \"You are a helpful assistant.\"}, {\n\"role\": \"user\", \"content\": \"%s\"}], \"stream\": true",
        modelName, prompt);

    try {
        URL url = new URL(OPENAI_URL);
        HttpURLConnection connection = (HttpURLConnection) url.openConnection();
        connection.setRequestMethod("POST");
        connection.setRequestProperty("Content-Type", "application/json");
        connection.setRequestProperty("Authorization", "Bearer " + DotEnv.getEnv("OPENAI_KEY"));
        connection.setDoOutput(true);
        connection.setChunkedStreamingMode(0); // 스트리밍 모드 활성화

        try (OutputStream os = connection.getOutputStream()) {
            byte[] input = jsonString.getBytes("utf-8");
            os.write(input, 0, input.length);
        }

        String contents = "";
        // 서버로부터 스트리밍 응답 처리
        try (BufferedReader br = new BufferedReader(new InputStreamReader(connection.getInputStream(), "utf-8"))) {
            String responseLine;
            while ((responseLine = br.readLine()) != null) {
                if (responseLine.startsWith("data: ")) {
                    String jsonData = responseLine.substring(6);
                    try {
                        JSONObject jsonObject = new JSONObject(jsonData);
                        String content = jsonObject.getJSONArray("choices").getJSONObject(0).getJSONObject("delta").getString("content");
                        System.out.println(content);
                        contents += content;
                    } catch (Exception e) {
                        //
                    }
                }
            }
        }
        System.out.println();
        connection.disconnect();
        return contents;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}
```

chatbot : stream

# Source code : LogService

```
public class ChatLogService {

    OpenAIRequest openAIRequest = new OpenAIRequest();
    HistoryDAO historyDAO = new HistoryDAO();
    ChatLogDAO chatLogDAO = new ChatLogDAO();
    JSONObject jr;

    public HashMap<String, String> getChatbotResponse(int chatid, ChatDTO chat, String input)
    {
        HashMap<String, String> map = new HashMap<>();
        if (input.equals("")) {
            map.put("contents", "내용을 입력해 주세요.");
            return map;
        }

        // cache 조회
        if (chat.isCache_enabled()) {
            LogDTO log = checkSameContents(chatid, input);
            if (log != null) {
                map.put("contents", log.getResponse());
                return map;
            }
        }

        String prompt = input;
        // 프롬프트 제작
        if (chat.isMemory_enabled()) {
            prompt = makePrompt(chatid, input);
        }
    }
}
```

Cache, prompt 제작

```
// 답변 생성
ChatLogDTO chatDTO = null;
if (chat.isStream_enabled()) {
    String contents = openAIRequest.chatBotStream(chat.getModel(), prompt);
    chatDTO = openAIRequest.makeChat(chatid, chat.getModel(), input, contents);
    map.put("prompt", prompt);
    map.put("contents", null);
} else {
    jr = openAIRequest.chatBot(chat.getModel(), prompt);
    chatDTO = openAIRequest.makeChat(chatid, chat.getModel(), input, jr);
    map.put("prompt", prompt);
    map.put("contents", chatDTO.getResponse());
}

// log 저장
int cResult = chatLogDAO.insertChatLog(chatDTO);
if (cResult == 0) {
    map.put("contents", "대화내용이 저장되지 않았습니다. DB error");
    return map;
}
}
```

Response 생성 및 저장

```
// total token 계산
if (chat.isMemory_enabled() && !chat.isStream_enabled()) {
    int totalTokens = chatLogDAO.calculateTotalTokens(chatid);

    if (totalTokens > Constants.RECORD_TOKEN_LIMIT) {
        // summary 생성
        String contents = prompt + chatDTO.getResponse() + "/위의 대화내용 단답형으로 요약";
        jr = openAIRequest.chatBot(Constants.MODEL_NAME_FOR_SUMMARY, contents);
        HistoryDTO historyDTO = openAIRequest.makeHistory(jr, chatid);
        // 대화 내용 저장
        int hResult = historyDAO.insertHistory(historyDTO, chatid);
        if (hResult == 0) {
            map.put("contents", "대화요약이 저장되지 않았습니다. DB error");
            return map;
        }
    }
}
return map;
}
```

Total token 계산,  
Summary 생성

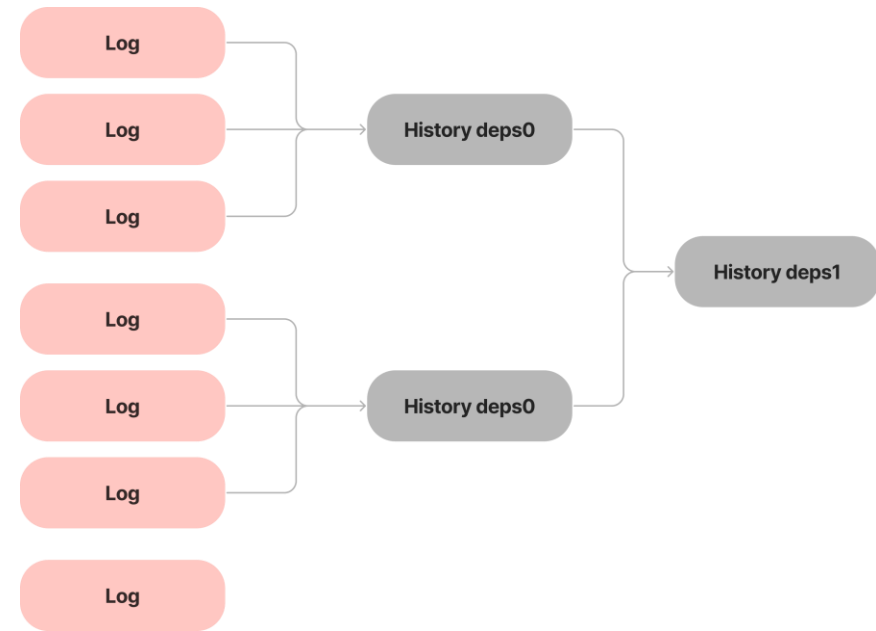
# Improvements

"제품 **가격**은 얼마입니까?"



귀하의 제품 **가격**은 얼마입니까?  
제품 **가격**을 알려 주실 수 있나요?  
귀하의 제품 에 대해 **얼마**를 지불 해야 합니까?  
귀하의 제품에 붙은 **가격**표는 무엇입니까?  
제품 **가격**을 어떻게 알 수 있나요?

1. Cache 기능의 낮은 유연성



2. History deps 구현



**Thank You**