Settings and Streaming

Semantic Kernel

גלעד מרקמן

פרויקט ב- Github

Settings

- מודלי הבינה המלאכותית מאפשרים להעביר אליהם הגדרות שונות הקובעות כיצד יתנהג המודל.
- הגדרות אילו משתנות ממודל למודל. כך לדוגמה ההגדרות של מודל GPT שונות מההגדרות של מודל Gemini. יתר על כן, אפילו ההגדרות של GPT-4.1 שונות מההגדרות של GPT-5.
- על כן, אנחנו רק נציג את הדרך בה אנו קובעים את ההגדרות וניתן דוגמה. יש להיעזר בתיעוד של כל מודל באתרי החברות.
- לדוגמה: למודלים של GPT עד למודל GPT-5.1 היתה הגדרה של Tempratute הקובעת את מידת האקראיות של התשובה. ככל שהטמפרטורה גבוהה התשובה היתה יותר אקראית וגמישה. מאפיין זה בוטל במודלים החדשים.

Settings

- השימוש ב Settings נעשה בשני שלבים:
  - הגדרת אובייקט ה Settings.
  - העברת ה Settings למודל יחד עם ה Prompts.
- הגדרת ה Setting :

```
var settings = new OpenAIPromptExecutionSettings {
    Temperature = 2,// creative max = 2
    MaxTokens = 100,
};
```

- העברת ההגדרות למודל:

```
var result = await chatService.GetChatMessageContentAsync(history, settings, kernel);
```

Streaning

- התשובה של מודלי הבינה המלאכותית עשויה להיות ארוכה מאוד.
- במקרה הרגיל התשובה מוחזרת אלינו רק בסיומה כמקשה אחת. בדרך זו עלינו לחכות עד לסיום הכנת התשובה כדי להציגה למשתמש.

- ה Streaming מאפשר לנו להציג את התשובה כזרם תוך כדי הכנתה על ידי המודל. חווית המשתמש במקרה זה היא טובה יותר.

- ב Streaming אנחנו מדפיסים את התשובה של המודל עוד לפני שמתקבל התשובה הסופית.

- נדגים כיצד אנחנו יכולים לבצע Streaming בקוד שלנו.

Streaning

```
while (true)
{
    Console.Write(">> ");
    string userMessage = Console.ReadLine();
    if (string.IsNullOrWhiteSpace(userMessage)) {break;}
    history.AddUserMessage(userMessage);

    string string_builder = "";

    var stream = chatService.GetStreamingChatMessageContentsAsync(
        chatHistory: history);

    await foreach (var chunk in stream)
    {
        Console.Write(chunk);
        string_builder += chunk.Content;
    }

    Console.WriteLine();
    history.AddAssistantMessage(string_builder);
}
```

- מחרוזת לשרשור התשובה המלאה.
- קריאה מיוחדת המאפשרת Streaming.
- הדפסת התשובה תוך כדי קבלתה, ושרשור התשובה המלאה.
- עדכון ההיסטוריה עם התשובה המלאה

Streaming

- הקריאה למודל עם Streaming מחזירה לנו אובייקט Stream שממתין לקבלת חלקים מהתשובה.

```
var stream = chatService.GetStreamingChatMessageContentsAsync(
    chatHistory: history);
```

- הלולאה שלנו `await foreach (var chunk in stream)` מתבצעת בכל פעם שמתקבל חלק מהתשובה, עד אשר מתקבלת התשובה המלאה.

- למעשה בלולאה אנחנו מחכים בכל פעם לקבלת החלק הבא של התשובה, מדפיסים אותו ומשרשרים אותו לתשובה המלאה.