The good BOcTor

Ricchetti Nicolò, Ronzoni Davide, Caprari Filippo, Alberini Gianluca

- **Di cosa si occupa e come funziona**
- **Come crea le immagini**
- **Come funziona il comando “/database”**
- **Come traduce**

Di cosa si occupa e come funziona

```
else{
    ctx.reply("Thinking...")

    // Initialize the history of the conversation
    if (history[id] === undefined) {
        history[id] = [{
            role: "system",
            content: "Sei un medico di medicina generale che lavora nell'ospedale più pres
iente lo richiede genera una foto o un video."
        }]
    }

    try {
        history[id].push({
            role: "user",
            content: message
        })

        const completion = await openai.chat.completions.create({
            messages: history[id],
            model: 'gpt-3.5-turbo-0301'
        })
```

Immagini

```
bot.on("message", async(ctx) => {
    const message = ctx.message.text
    const id = ctx.message.chat.id


    const msg = message.toLowerCase()

    if(msg.includes("mi disegni") || msg.includes("disegna") || msg.includes("foto") || msg.includes("immagine")){
        ctx.reply("Drawing...");

        const ciao = await openai.images.generate({
        prompt:ctx.message.text,
        n:1,
        size:"1024x1024",
        });

        ctx.replyWithPhoto(ciao.data[0].url);
    }
```

Comando “/database”

```
const option = {
    method: 'GET',
    url: 'https://medical-articles-live.p.rapidapi.com/journals/diabetes',
    headers: {
      'X-RapidAPI-Key': process.env.RAPIDAPI_KEY,
      'X-RapidAPI-Host': 'medical-articles-live.p.rapidapi.com'
    }
  };
```

Comando “/database”

```
bot.command("database", async(ctx) => {
    try {
        const request1 = await axios.request(option);
        const dati = request1.data
        let flag = true
        let variabile = ctx.message.text.substring(10) //utente
        ctx.reply("Ecco tutto quello che ho trovato riguardante " + variabile)
       // console.log(dati)

        for (let i=0; i<dati.length; i++) {
            const dato = dati[i]

            if(dato.title.includes(variabile) && i!=0 ){

                ctx.reply(risposta.data.translatedText)
                ctx.reply(dato.url)
                flag = false

            }
        }

        if(flag){
            ctx.reply("info non presente nel database")
        }

    } catch (error) {
        ctx.reply(error)
    }
})
```

Processo di traduzione

```
        if(dato.title.includes(variabile) && i!=0 ){

            try {
                encodedParams.set("text", dato.title)

                const risposta = await axios.request(options);
                //const traduzione = risposta.data;

                console.log(risposta.data.translatedText);
                                    ctx.reply(risposta.data.translatedText)
                ctx.reply(dato.url)
                flag = false

            } catch (error) {
                console.error(error);
            }

        }
    }
```