ChatGPT im Telefon – ein Retro-Sprachassistent

In diesem Projekt baust du ein Telefon so um, dass du mit ihm mit ChatGPT telefonieren kannst: Du stellst eine Frage, die mit einem Mikrofon aufgezeichnet und anschließend transkribiert wird. Danach wird der Text an die ChatGPT API gesendet. Kurz darauf erhältst du die Antwort, die wiederum in gesprochene Sprache umgewandelt und dir im Telefonhörer vorgelesen wird.

Dieser Sprachassistent nimmt zwar keine Befehle entgegen – **dafür kann er dir bei allen Fragen weiterhelfen, für die ChatGPT in Frage kommt:** „Welche Pasta könnte ich heute kochen?“, „Ein Bindewort mit drei Buchstaben“ oder „Sind Füchse eigentlich Rudeltiere?“

Zum Einsatz kommen hierbei diese Bauteile:

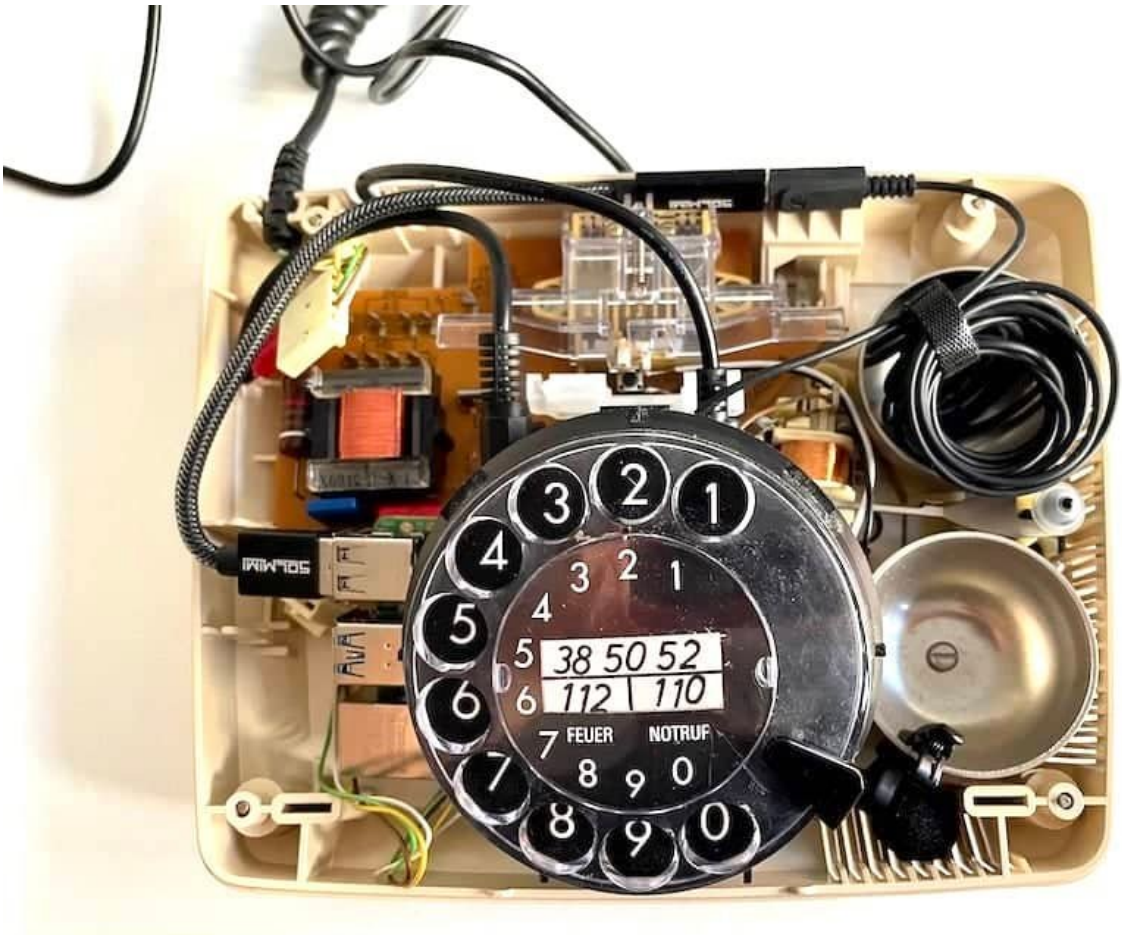
- Telefon
- Raspberry Pi 4
- Netzteil
- Lavalier-Mikrofon
- Flachstecker 2,8 mm
- Button
- Jumperkabel
- 3,5 mm Klinkenkabel

Auf dem Raspberry Pi läuft ein Python Script, das über einen

Mechanismus unter der Telefongabel gesteuert wird und sich um die Verarbeitung deiner Frage kümmert.

Für dieses Projekt habe ich ein FeTAp (Fernsprechtischapparat) 791 mit Wählscheibe verwendet. Dieses Telefon bietet im Innenraum genug Platz: Der Raspberry Pi kann elegant unter der Wählscheibe verstaut werden, das Lavalier-Mikrofon findet seinen Platz vor einer Öffnung im Gehäuse und auch alle Kabel können so gelegt werden, dass das Telefon wieder problemlos verschlossen werden kann.

Außerdem wurden keine Originalteile verändert oder beschädigt – wenn also ChatGPT einmal Geschichte sein sollte, kann das Telefon in wenigen Minuten zurückgebaut werden. So sieht das Innenleben nach dem Einbau aus:



Der Aufbau der Hardware

Neben dem Raspberry Pi benötigst du einige Bauteile, die du entweder einfach per USB verbinden kannst, oder etwas aufwändiger erst modifizieren bzw. herstellen musst. Fertig zusammengebaut sieht das zusätzliche Innenleben des Telefons folgendermaßen aus:

Auf der rechten Seite siehst du das Lavalier-Mikrofon. Unten an der Pin-Leiste des Raspberry Pi befindet sich ein Button, der an zwei Kabel gelötet wurde. Am Line-Ausgang (oben) befindet sich ein Klinken-Stecker, dessen Kabel in zwei Flachsteckern endet. Doch eins nach dem anderen.

Das Mikrofon

Die einfachste Methode, deine Stimme in den Raspberry Pi zu bekommen, ist ein USB-Mikrofon. Auf dem Foto oben siehst du

das Lavalier-Mikrofon Sennheiser XS-Lav USB-C. Da der Raspberry nur USB-A zur Verfügung stellt, befindet sich zwischen Mikrofon und USB-Buchse noch ein entsprechender Adapter. Achte darauf, ein Adapter-Kabel zu verwenden. Ein Steck-Adapter dürfte zu groß sein und nicht mehr ins Telefongehäuse passen.

Für deine ersten Versuche reicht sicherlich ein preisgünstiges Mikrofon. Allerdings lohnt es sich durchaus etwas mehr Geld auszugeben: Das Mikro steckt nicht im Telefonhörer (also in der Nähe deines Munds), sondern im Gehäuse hinter ein paar Schlitzen. Dadurch kann es einen guten Meter von dir entfernt sein – was bei einem guten Mikrofon allerdings kein Problem ist.

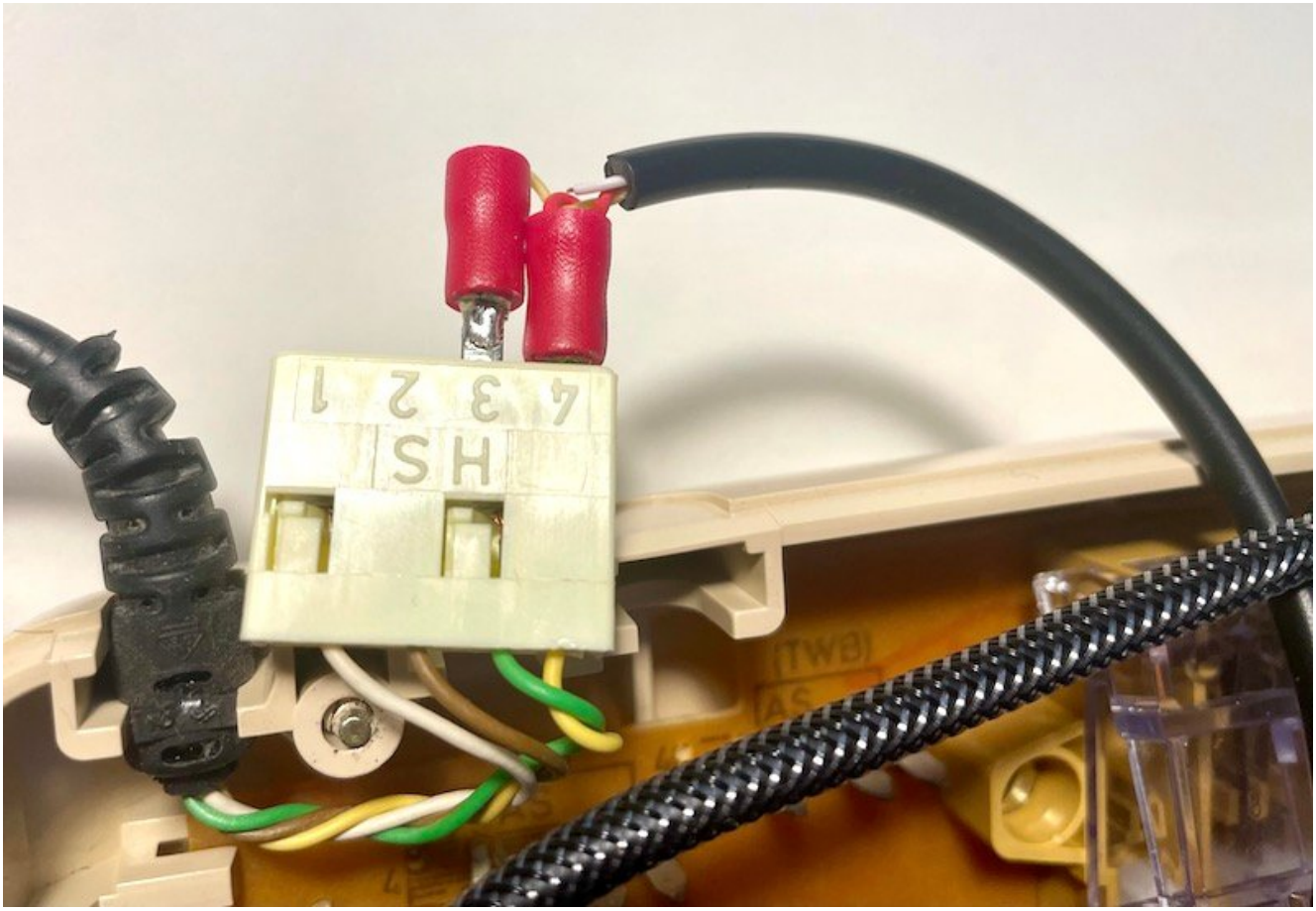
Der Lautsprecher im Telefonhörer

Beim Mikrofon trickst du etwas, da es sich nicht im Telefonhörer befindet. Anders beim Lautsprecher – hier kommt das Original zum Einsatz. Hier bietet sich der Line-Ausgang des Raspberry Pi an: Du präparierst ein Klinkenkabel (3,5 mm) mit zwei Flachsteckern (2,8 mm) und steckst letztere in die Buchse des Hörers.

Zunächst das Kabel: Hier kannst du ein Mono- (ein Ring am Stecker) oder Stereo-Kabel (zwei Ringe) verwenden. Schneide ein gut 10 cm langes Stück ab und isoliere die Enden ab:

Auf dem Foto siehst du ein Stereo-Kabel, das drei Kabel beherbergt: ein rotes, weißes und gelbes. Die ersten beiden übertragen den rechten und linken Kanal, das gelbe (kann bei dir z.B. auch schwarz sein) ist die Erde. **Für den Anschluss am Lautsprecher im Hörer benötigst du die Erde und entweder das rote oder weiße Kabel.**

Löte an diese beiden Enden je einen Flachstecker mit einer Breite von 2,8 mm an. Diese passen perfekt in die Anschluss-Buchse des Hörers, die du vorher vorsichtig von ihren Anschlüssen im Telefon ziehen kannst. Stecke nun die beiden Flachstecker zum gelben und grünen Kabel in die Buchse:



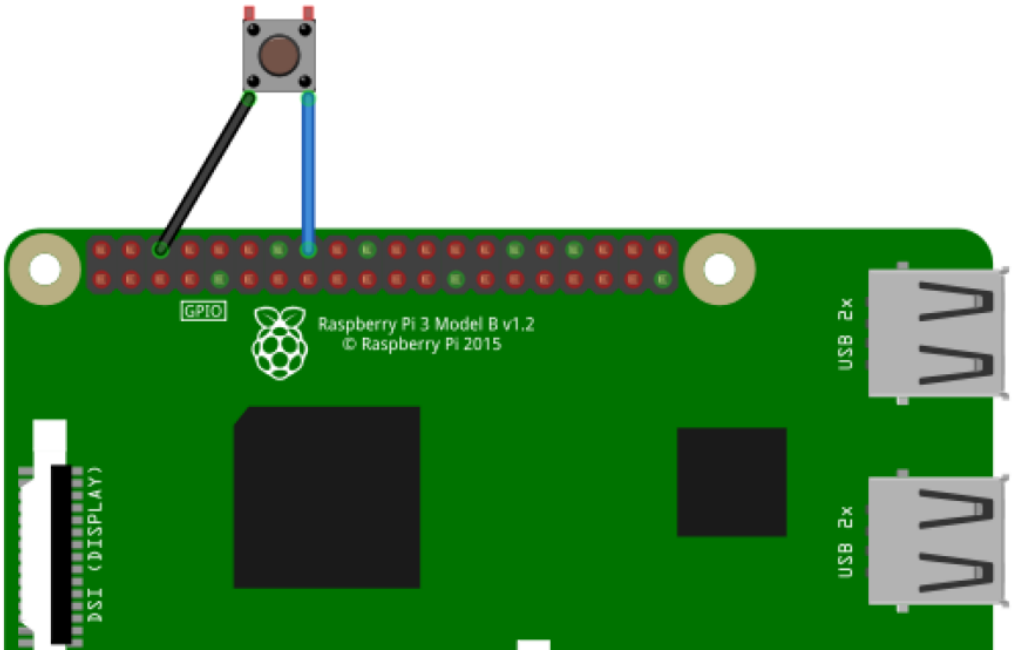
Hinweis: Je nachdem, welches Telefon du verwendest, können sich die Farben der Kabel, die zum Lautsprecher im Hörer führen, natürlich unterscheiden. Schraube in diesem Fall den Telefonhörer auf und schaue kurz nach, welche Kabel du mit dem Klinkenstecker verbinden musst.

Damit ist dein Hörer schon einsatzbereit. Wenn du ihn schon einmal vorab testen möchtest, schließe den Klinkenstecker am Raspberry Pi an und spiele eine Audio-Datei ab. Im Telefonhörer sollte der Lautsprecher diese nun abspielen.

Der Button

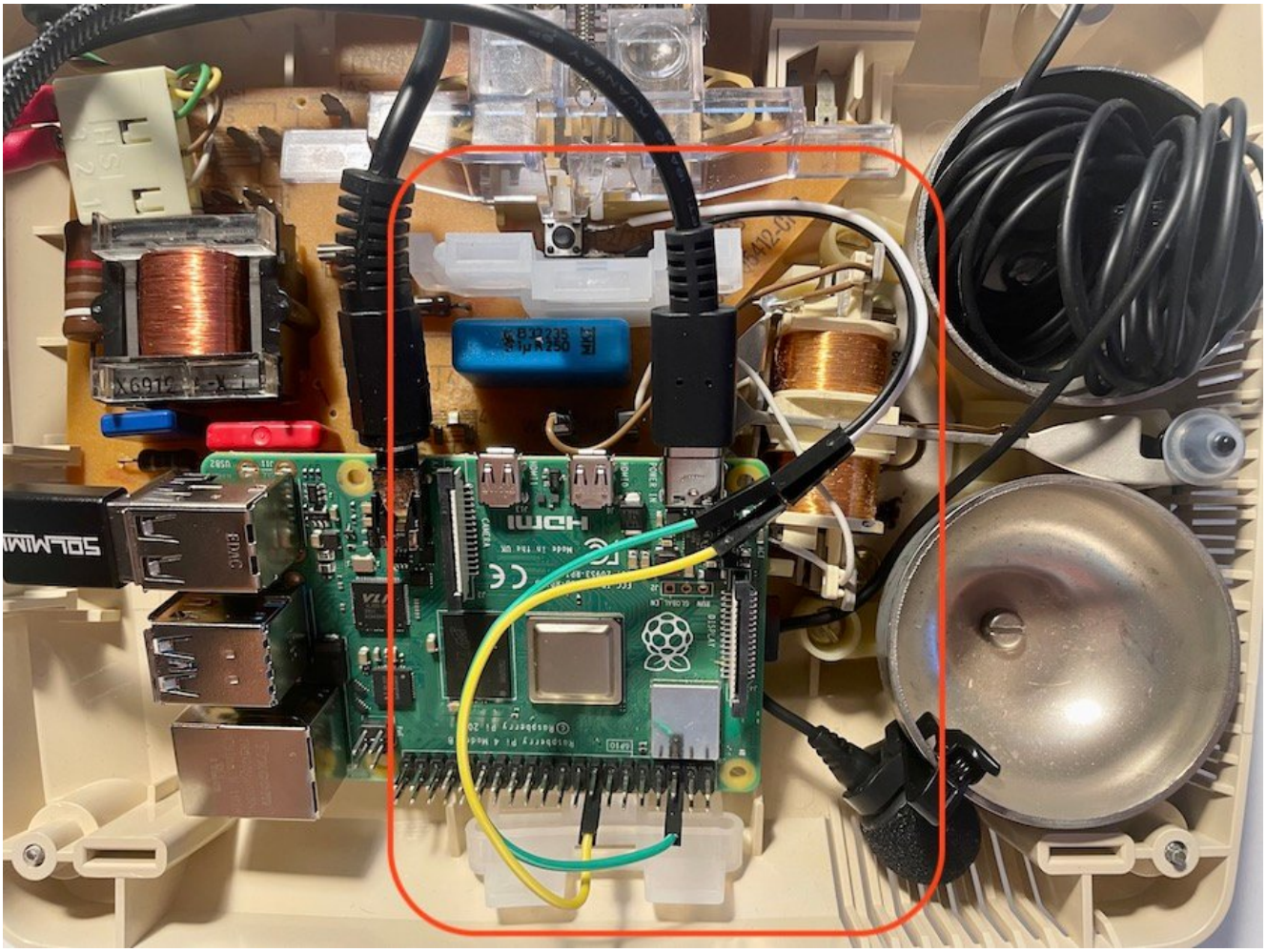
Kommen wir zum mechanischen Teil der Hardware – einem Button, mit dem du das Gespräch mit ChatGPT starten kannst. Sobald der Button gedrückt (oder losgelassen wird) wartet der Raspberry Pi auf deine Frage und das weitere Programm nimmt seinen Lauf.

Für den Anschluss am Raspberry Pi benötigst du neben dem Button zwei Jumper-Kabel. Diese sollten circa 20 cm lang sein, damit du den Button an einer geeigneten Stelle im Gehäuse platzieren kannst – dazu gleich mehr. Solltest du nur kürzere Kabel zur Hand haben, kannst du diese auch zusammenstecken. Schließe den Button wie folgt an:



Ein Pin des Button ist hierbei mit Erde (GND) und der andere am Raspberry Pi mit Pin 16 (GPIO23) verbunden. **Einen Pull-up- bzw. Pull-down-Widerstand benötigst du nicht, da du den internen Widerstand verwenden wirst.**

Jetzt stellt sich die Frage, wohin mit dem Button im Gehäuse? Besonders praktisch wäre es, wenn er nicht extra gedrückt werden müsste, sondern betätigt wird, sobald der Hörer abgenommen wird. Hierfür kannst du den Button unter dem Gabelmechanismus platzieren, wie auf dem folgenden Bild zu sehen:



Wenn der Hörer aufliegt, wird die Telefongabel heruntergedrückt – der Mechanismus drückt dann den Button herunter. Sobald du den Hörer abnimmst, springt der Mechanismus hoch und lässt dadurch auch den Button los – und das Python Script wird gestartet bzw. das Gespräch kann losgehen. **Das funktioniert aber nur, wenn der Button leichtgängig genug ist, damit das Gewicht des Hörers ausreicht, um ihn herunterzudrücken.** Hier ist etwas Ausprobieren deinerseits gefragt. Solltest du jedoch keinen passenden Button finden, muss es doch andersherum funktionieren: Du nimmst zuerst den Hörer ab und drückst die Telefongabel manuell herunter, um das Script zu starten.

Und das war es auch schon auf der Hardware-Seite! **Warte allerdings noch mit dem Einbau des Raspberry Pi und seiner Peripherie bis du das Projekt zum Laufen gebracht hast.** So

kannst leichter Maus, Tastatur und Bildschirm anschließen und dich um die Software kümmern.

Einrichten der Software

Kommen wir zum Kern dieses Projekts – der Software. Auf dem Raspberry Pi wirst du ein Python Script erstellen und speichern, das in einem Endlos-Loop läuft und auf dein Signal (den Button) wartet. Außerdem benötigst du einige MP3s, die du als Ansagen und für Fehlermeldungen verwendest. **Das Telefon hat kein Display und keine Kontrollleuchten, deshalb läuft die „Benutzeroberfläche“ über den Lautsprecher im Hörer.**

Doch zunächst musst du das Betriebssystem für den Raspberry Pi vorbereiten. **Hier benötigst du die Möglichkeit, per SSH auf den kleinen Rechner zugreifen zu können,** da du später keinen Monitor mehr zur Verfügung haben wirst, über den du das Script starten kannst.

Falls auf deinem Raspberry Pi schon ein Betriebssystem läuft, SSH aber noch nicht aktiviert ist, hole das über die Einstellungen nach. Rufe hierfür in der Kommandozeile die Konfiguration auf:

```
sudo raspi-config
```

Anschließend wählst du den Menüpunkt **Interfacing Options / SSH** und aktivierst SSH. Falls du ein frisches Betriebssystem verwenden möchtest – in diesem Tutorial erkläre ich, wie du SSH direkt beim Erstellen der SD-Karte aktivieren kannst. Dort erfährst du auch, wie du per SSH von einem anderen Computer auf den Raspberry PI zugreifen kannst.

Die benötigten Python-Bibliotheken

Bevor du mit dem Python Script loslegen kannst, musst du ein paar Bibliotheken installieren, die du später brauchen wirst. Rufe hierfür auf dem Raspberry Pi die Kommandozeile auf und

gib **nacheinander** die folgenden Befehle ein:

```
pip install speechRecognition==3.10.0
pip install openai==0.28.0
pip install gtts
pip install pygame

sudo apt install python3-pip flac ffmpeg -y
sudo apt install python3-pyaudio

sudo apt-get install rpi.gpio
```

Hierbei handelt es sich um Bibliotheken für die folgenden Funktionen:

- **speechRecognition:** Stellt Funktionen bereit, um deine Stimme in Text umwandeln zu können. Achte auf die Version 3.10.0
- **openai:** Stellt die Verbindung zu ChatGPT her
- **gtts:** Wandelt die Antwort von ChatGPT wieder in gesprochene Sprache um
- **pygame, flac ffmpeg, pyaudio:** Benötigst du für die Verarbeitung der Audio-Dateien
- **rpi.gpio:** Sorgt dafür, dass du die Pins des Raspberry Pi ansteuern kannst

Erstelle ein Konto bei OpenAI und einen API-Key

Um die API von ChatGPT nutzen zu können, benötigst du ein Konto bei OpenAI. Die API ist kostenpflichtig – **aber keine Sorge, hierfür fallen keine horrenden Beträge an.** Eine Antwort auf eine Frage zu bekommen, die du über das Telefon stellt, dürfte dich in der Regel nur einen Bruchteil eines Cents kosten. Selbst wenn du GPT-4 verwendest (wie, erfährst du gleich), dürften sich die Kosten in Grenzen halten.

In diesem Tutorial bei Pollux Labs erfährst du, wie du ein Konto bei OpenAI anlegst und dir einen API-Key erstellst. Dort findest du auch weitere Informationen zu den Preisen sowie einen Link zur aktuellen Preisliste. Sobald du einen API-Key und etwas Guthaben bei OpenAI besitzt, kann es direkt mit dem Python-Script weitergehen.

Das Python-Script

Um das Script zu erstellen, öffne auf deinem Raspberry Pi einen Editor (z.B. Thonny), erstelle ein neues Projekt und kopiere den folgenden Code hinein:

```
import speech_recognition as sr
from openai import OpenAI
import json
from gtts import gTTS
import pygame
import random
import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)
buttonPin = 16
GPIO.setup(buttonPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)

client = OpenAI(
  api_key="DEIN API KEY",
)

def callGPT():
    #Zufällige Ansagen erzeugen
    randQuestion = random.randrange(1,5)
    randWait = random.randrange(1,4)

    #Ansage abspielen
     time.sleep(2) #2 Sekunden warten, bis die Frage kommt
(Zeit, den Hörer abzunehmen)
    pygame.mixer.init()
pygame.mixer.music.load("Question{}.mp3".format(randQuestion))
    pygame.mixer.music.play()
```

```
    while pygame.mixer.music.get_busy():
        pass
    #Gesprochene Frage vom Mikrofon empfangen
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print("Stelle deine Frage.")
        audio = r.listen(source)

    #Frage transkribieren
    try:
         recognizedText = r.recognize_google(audio, language =
"de_DE")
          print("Google Speech Recognition hat folgendes
verstanden: " + recognizedText)
    except sr.UnknownValueError:
         print("Google Speech Recognition konnte dich nicht
verstehen")
        pygame.mixer.init()
        pygame.mixer.music.load("unintelligible.mp3")
        pygame.mixer.music.play()
        while pygame.mixer.music.get_busy():
            pass
        return
    except sr.RequestError as e:
          print("Konnte kein Ergebnis von Google Speech
Recognition empfangen; {0}".format(e))
        pygame.mixer.init()
        pygame.mixer.music.load("error.mp3")
        pygame.mixer.music.play()
        while pygame.mixer.music.get_busy():
            pass
        return


    #Ansage "Bitte warten" abspielen
    pygame.mixer.init()
    pygame.mixer.music.load("Wait{}.mp3".format(randQuestion))
    pygame.mixer.music.play()
    while pygame.mixer.music.get_busy():
        pass
```

```
    #ChatGPT
    print("Schreibe die Antwort...")
    completion = client.chat.completions.create(
    #model="gpt-3.5-turbo",
    model="gpt-4",
    messages=[
    {"role": "system", "content": "Du beantwortest Fragen von
Nutzern."},
     {"role": "user", "content": "Beantworte die folgende
Frage: {}".format(recognizedText)}]
    )

    text = completion.choices[0].message.content

    #Erstellen des gTTS-Objekts
    tts = gTTS(text, lang='de')

     #Speichern der Antwort als MP3 (alte Antworten werden
überschrieben)
    tts.save("output.mp3")

    #Antwort abspielen
    pygame.mixer.init()
    pygame.mixer.music.load("output.mp3")
    pygame.mixer.music.play()
    while pygame.mixer.music.get_busy():
        pass


while True:
    buttonState = GPIO.input(buttonPin)
     if buttonState == GPIO.LOW: #LOW = Funktionsstart bei
gedrücktem Button, HIGH = Start bei Loslassen
        callGPT()
    else:
        print(".")
```

Speichere das Script gleich ab mit einem Namen deiner Wahl – meines heißt **runcallgpt.py**

Trage nun deinen API-Key von OpenAI in das Script ein:

```
API_KEY = "DEIN API-KEY VON OPENAI"
```

Für einen ersten Tests des Python-Scripts benötigst du noch ein paar MP3-Dateien, die als Ansagen und Fehlermeldungen dienen. Du kannst hierfür dieses ZIP-Archiv von mir herunterladen und verwenden. Darin findest du verschiedene MP3s sowie ein kleines Script, mit dem du deine eigenen Ansagen erstellen kannst. **Entpacke die Dateien in dasselbe Verzeichnis, in dem auch dein Python-Script liegt.** Falls du eigene Ansagen verwendest, achte auf die Dateinamen – diese müssen mit den Dateinamen im Script übereinstimmen.

Dein erster Test

Wenn du die obigen Vorbereitungen abgeschlossen hast und alle Kabel an ihrem Platz sind, kann es losgehen! Starte das Script, nimm den Hörer ans Ohr und drücke den Button – nach 2 Sekunden sollte dich eine nette Computer-Stimme begrüßen und dich um deine Frage bitten.

Nachdem du gesprochen hast, sollte deine Frage als Text in der Konsole von Thonny erscheinen. Nach einer weiteren Ansage sollte dir die Antwort von ChatGPT vorgelesen werden.

Wie du siehst, sind das eine Menge „sollte“ – falls nichts passiert oder das Script an einer bestimmten Stelle abbricht, gehe die einzelnen Stationen darin durch:

- Ist der Button richtig angeschlossen?
- Funktioniert das Mikrofon?

Im Script wird das „Default-Mikrofon“ verwendet, der Raspberry Pi hat aber kein solches. Oft findet er es trotzdem – falls nicht, lasse dir mit folgendem Befehl die Liste der erkannten Geräte anzeigen:

```
sr.Microphone.list_microphone_names()
```

Wenn dort dein Mikro zum Beispiel an dritter Stelle auftaucht, ersetze im obigen Script die Zeile

```
with sr.Microphone() as source:
```

durch die folgende:

```
with sr.Microphone(device_index=3) as source:
```

- Stimmt der API-KEY von OpenAI?
- Stimmen die Dateinamen der Ansage-MP3s mit dem Script überein?

Falls dein Fehler nicht dabei ist, kopiere dir die Fehlermeldung und starte eine Google-Suche. Oft wirst du damit am schnellsten eine Lösung für dein spezifisches Problem finden.

ChatGPT-3.5 vs. ChatGPT-4

Im Script gibt es eine Stelle, an der du entscheiden kannst, ob du das Sprachmodell mit der Version 3.5 oder lieber die aktuellere Version 4 verwenden möchtest. Die beiden unterscheiden sich in ihrer Leistung und damit in der Qualität der Antworten – **dafür ist die Version 4 aber auch gut 20 Mal so teuer wie ihre Vorgängerin.** Genauere Zahlen findest du in der oben erwähnten Preisliste von OpenAI.

Hier kannst du einstellen, welche Version du verwendest. Kommentiere einfach die Zeile mit der nicht zu verwendenden Version aus:

```
#model="gpt-3.5-turbo",
model="gpt-4",
```

Funktioniert? DAnn baue das Telefon zusammen

Wenn du alles funktioniert wie es soll, kannst du die Peripherie vom Raspberry Pi trennen (bis auf das Mikrofon und das Kabel zum Telefonhörer) und dich um dein Einbau kümmern. Nimm hierfür am besten die Wählscheibe heraus und verstaue alles, **ohne Kabel zu knicken oder Steckverbindungen zu lösen.** Achte darauf, den Button unter den Mechanismus der Telefongabel zu verstauen, ohne das er wegrutschen kann.

Wenn alles sicher untergebracht ist, kannst du die Wählscheibe wieder in die Halterung setzen, das Netzteil des Raspberry Pi aus dem Gehäuse führen und den Deckel des Telefons aufsetzen.

Das Python-Script aus der Ferne starten

Wenn du das Telefon zusammengebaut hast, hast du natürlich keine Maus, Tastatur und keinen Bildschirm mehr für den Raspberry Pi zur Verfügung. Hier kommt nun SSH ins Spiel. In diesem Projekt bei Pollux Labs wird beschrieben, wie du per SSH von einem anderen Computer auf den Raspberry Pi zugreifen kannst.

Sobald die Verbindung steht, steuere den Ordner an, in dem das Script und die MP3 liegen. In meinem Beispiel liegen die Dateien im Ordner Desktop/callGPT:

```
cd Desktop/callGPT
```

Starte anschließend das Script, in meinem Fall:

```
python runcallgpt.py
```

Nun sollte das Script laufen. Löse am Telefon den Button aus und lausche der Stimme aus dem Hörer.

Autostart

Um das Script mit dem Boot des Raspberry Pi zu starten, lege zunächst einen CronJob an. Gib hierfür in die Kommandozeile folgenden Befehl ein:

```
sudo nano /etc/rc.local
```

Ergänze anschließend ganz unten die folgenden Zeilen. Wobei du natürlich den Pfad zu deinem Script anpassen musst.

```
sleep 5
su - pi -c 'python /home/pi/Desktop/callGPT/runcallgpt.py' &
```

Eine weitere Anpassung musst du im Script nun noch machen: Ergänze zu jeder MP3, die abgespielt werden soll, noch den vollständigen Pfad – ansonsten werden sie nicht gefunden. In meinem Beispiel also:

```
pygame.mixer.music.load("/home/pi/Desktop/callGPT/unintelligible.mp3")
```

So geht es weiter

Wenn dein KI-Telefon funktioniert – herzlichen Glückwunsch! Allerdings gibt es noch eine Vielzahl von Optimierungen, die es es noch besser machen würden. Wie wäre es z.B. mit einer „besseren“ Stimme? In diesem Tutorial lernst du eine weitere Methode für Text to Speech kennen.

Aktuell kannst du das Vorlesen einer Antwort mit dem obigen

Script nicht unterbrechen, indem du z.B. die Gabel herunterdrückst. Auch Folgefragen zu stellen, ist noch nicht möglich. Und sicherlich gibt es noch weitere Ideen, die dieses Gadget besser machen würden. Ich werde dieses Tutorial um weitere Versionen ergänzen – falls du Lösungen und Ideen gefunden hast, schreibe sie gerne in die Kommentare.