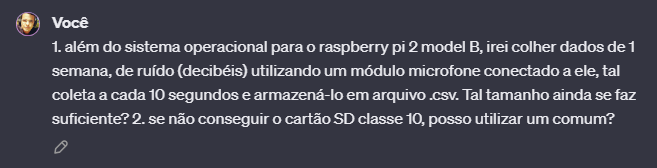
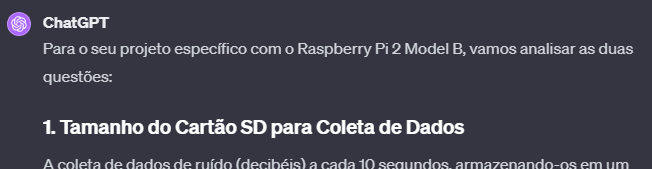
DecibelDataHarvester

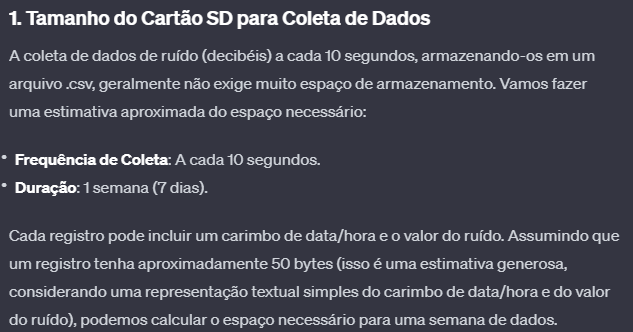


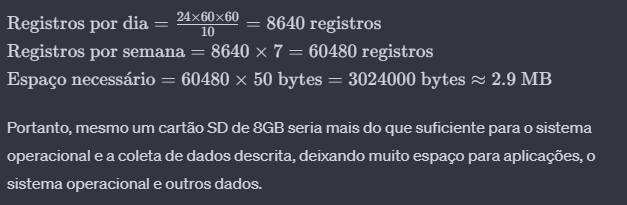
Consulta em ChatGPT, estimou-se:

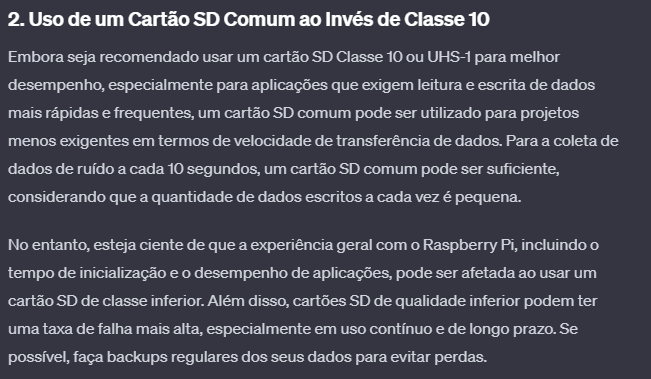


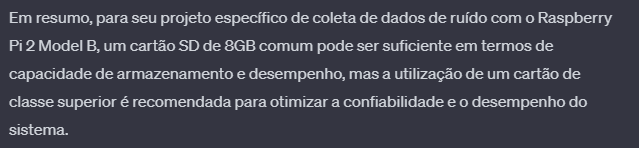
1. além do sistema operacional para o raspberry pi 2 model B, irei colher dados de 1 semana, de ruído (decibéis) utilizando um módulo microfone conectado a ele, tal coleta a cada 10 segundos e armazená-lo em arquivo .csv. Tal tamanho ainda se faz suficiente? 2. se não conseguir o cartão SD classe 10, posso utilizar um comum?



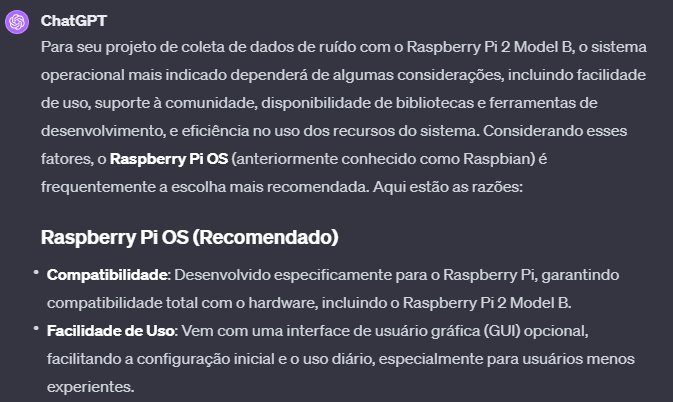


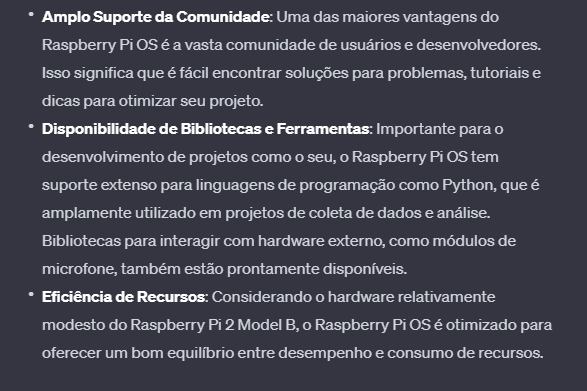


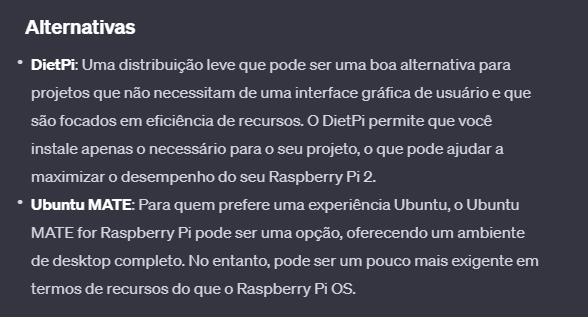


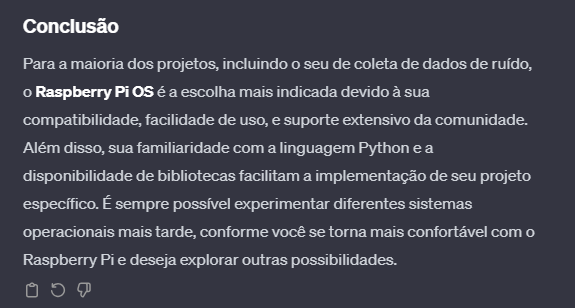


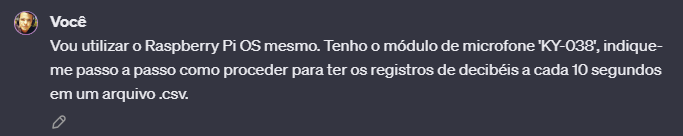


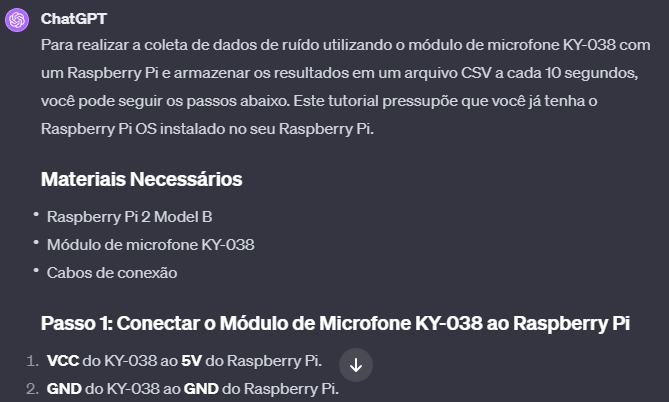


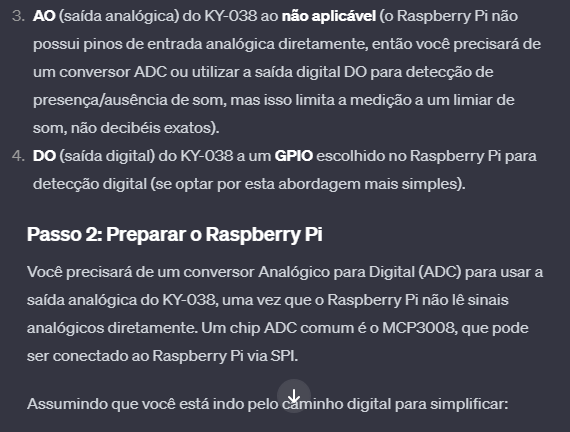


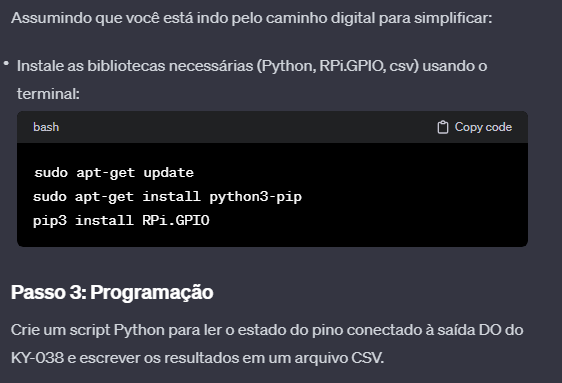










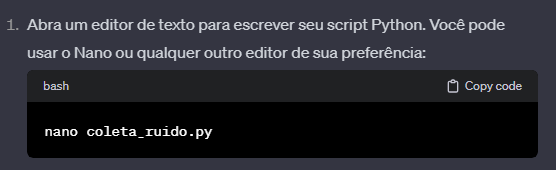


Code:

sudo apt-get update

sudo apt-get install python3-pip

pip3 install RPi.GPIO



nano coleta\_ruido.py



import RPi.GPIO as GPIO

import time

import csv

import datetime

# Configura o modo GPIO

GPIO.setmode(GPIO.BCM)

# Defina aqui o número do pino GPIO ao qual o DO do KY-038 está conectado

pin\_microfone = 17 # Exemplo: GPIO 17

GPIO.setup(pin\_microfone, GPIO.IN)

# Função para coletar dados

def coletar\_dados():

with open('dados\_ruido.csv', 'a', newline='') as arquivo:

escritor\_csv = csv.writer(arquivo)

while True:

# Lê o pino digital

valor = GPIO.input(pin\_microfone)

# Grava no CSV: timestamp e valor

escritor\_csv.writerow([datetime.datetime.now(), valor])

time.sleep(10) # Espera 10 segundos para a próxima coleta

if \_\_name\_\_ == '\_\_main\_\_':

try:

# Cabeçalho do CSV

with open('dados\_ruido.csv', 'w', newline='') as arquivo:

escritor\_csv = csv.writer(arquivo)

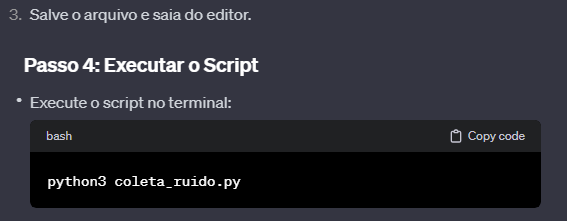
escritor\_csv.writerow(['Timestamp', 'Ruido'])

# Inicia a coleta

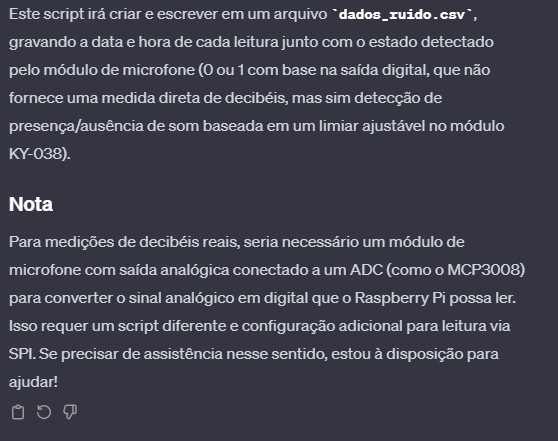
coletar\_dados()

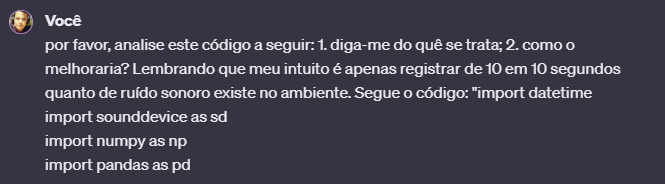
except KeyboardInterrupt:

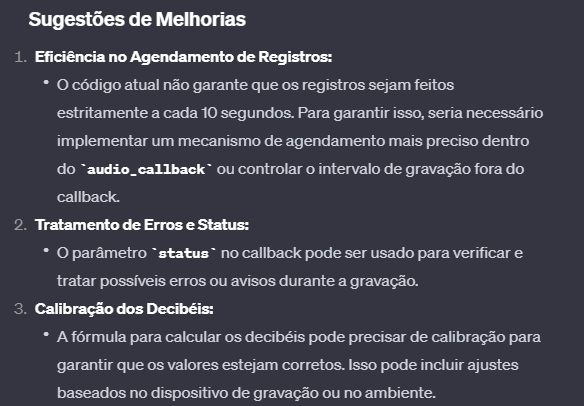
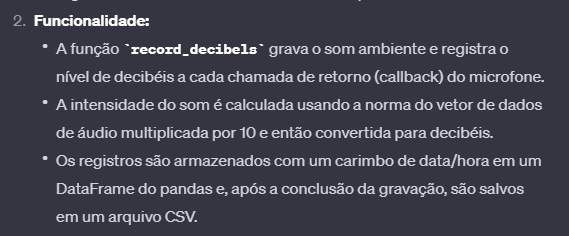
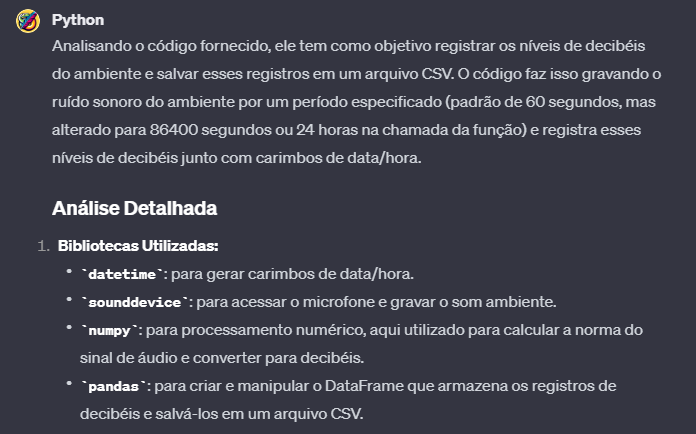
GPIO.cleanup()

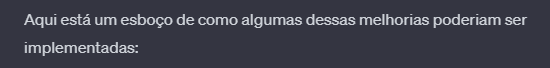
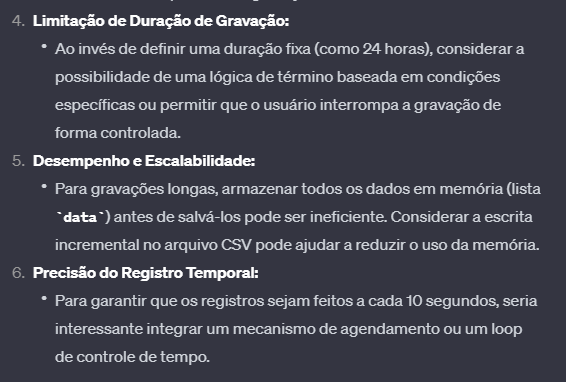


python3 coleta\_ruido.py









Código da sugestão de melhoria:

import datetime

import sounddevice as sd

import numpy as np

import pandas as pd

import time

def record\_decibels(duration=60, interval=10, filename='decibel\_log.csv'):

start\_time = time.time()

next\_record\_time = start\_time + interval

data = []

def audio\_callback(indata, frames, time, status):

if status:

print(status)

current\_time = datetime.datetime.now()

if current\_time.timestamp() >= next\_record\_time:

volume\_norm = np.linalg.norm(indata) \* 10

decibels = 20 \* np.log10(volume\_norm)

data.append([current\_time.strftime("%Y-%m-%d %H:%M:%S"), decibels])

next\_record\_time += interval

with sd.InputStream(callback=audio\_callback):

while time.time() - start\_time < duration:

sd.sleep(interval \* 1000 - (time.time() - start\_time) % (interval \* 1000))

df = pd.DataFrame(data, columns=['Timestamp', 'Decibels'])

df.to\_csv(filename, index=False)

# Ajuste conforme necessário para gravar por um período específico ou até que uma condição seja atendida

