# GPT Multimedia Assistant

Student: Gul Farhad Ali Irinel

Grupa: 334CC

## Introducere

Pentru acest proiect, am ales să folosesc un Raspberry Pi Pico W, deoarece aveam nevoie de un microcontroler care să poată utiliza informațiile obținute într-un mod practic.

Rolul unui API este să expună funcționalitatea unui sistem într-un mod organizat.

Platforma care oferă informațiile trebuie să descrie modul în care se poate comunica cu ea prin API. Comunicare cu API-urile alese în cadrul dispozitivelor se face la nivel de HTTP Requests.

Proiectul se bazează pe comunicarea cu mai multe API-uri, printre care și cel de bază - API-ul pus la dispoziție de OpenAI pentru a comunica cu ChatGPT. În momentul de față, este cea mai bună metodă de a comunica cu un AI dintr-un dispozitiv remote cu putere de procesare limitată.

Ideea de la care am pornit a fost lansarea API-ului ChatGPT de către OpenAI care a deschis orizontul și spre dispozitivele mai slabe computațional.

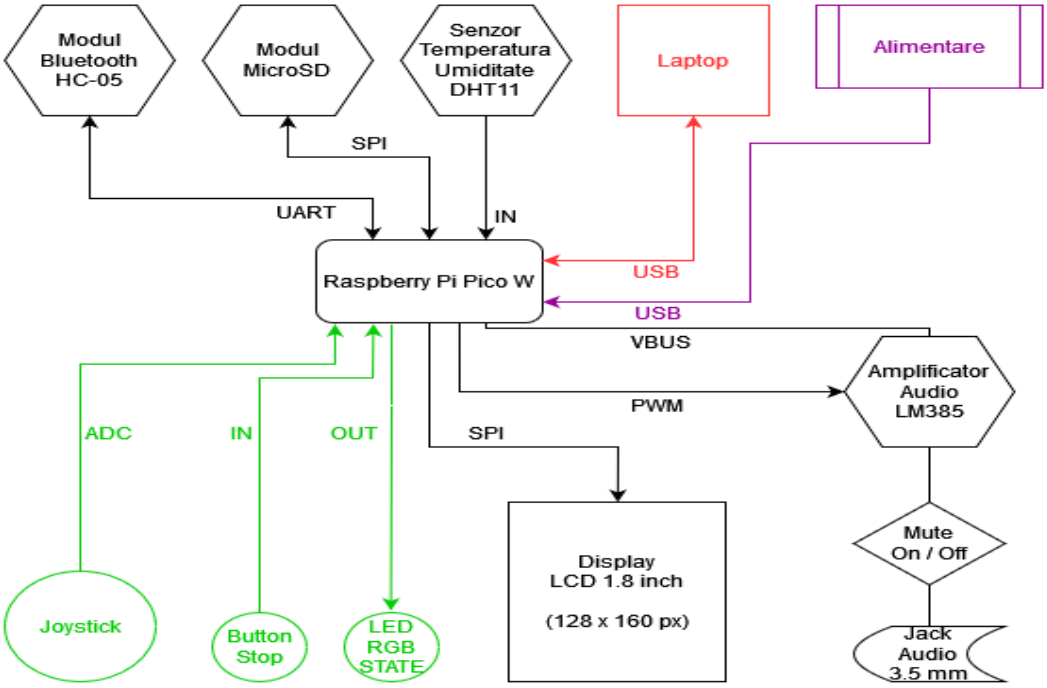
GPT Assistant Speaker este un dispozitiv portabil care preia comenzi din telefon, prin bluetooth și produce răspunsul vocal în 2 limbi posibile (engleză sau română, în funcție de modul selectat). Dispune de funcționalități precum: play la piese de pe youtube, vremea din orice oraș în momentul actual sau din următoarele zile, statistici locale de temperatură și umiditate. De asemenea, poate face request-uri la ChatGPT, să primească răspunsul și să-l transmită prin boxe.

Scopul acestui proiect este acela de a îndrepta atenția și spre această zonă de proiectare cu microcontrolere, zona de proiectare folosind Cloud, care are la bază API-uri.

In plus, acesta poate prelua date de la un modul ESP32 care transmite date in timp real de oriunde exista o conexiune la internet catre un server local. De la acel server local, GPTMA va prelua datele si le va procesa realizand un text mai cursiv.

## Descriere generală

#### Schema componentelor hardware




**Schema de comunicare în cloud**

# Hardware Design

Lista pieselor folosite:

- Raspberry Pico W
- Modul Bluetooth HC-05
- Modul MicroSD
- Senzor temperatură DHT11
- Display LCD cu ST7735
- Modul ridicator tensiune DC-DC Boost XL6009
- Amplificator audio LM385
- Jack Audio 3.5mm
- LED RGB
- Switch ON / OFF
- Buton STOP
- Joystick
- Fire de legatură
- Breadboard
- Cablaj de test

# Software Design

Proiectarea software a fost realizata in CircuitPython, care dispune de biblioteci si suport pentru ce am realizat in cadrul proiectului.

Pentru a realiza un API Request avem în primul rând nevoie de conexiune la internet.

După conectarea Pico la internet prin WIFI, avem nevoie de o bibliotecă care să poată face request-uri. Am ales biblioteca adafruit_requests, capabilă de requesturi simple.

Tot codul este complex și lung, vom pune accentul pe baza acestuia și anume requestul la ChatGPT.

Pentru a face acest request, avem nevoie de preconfigurare de socket, ssl, pentru ca mai apoi să deschidem o sesiune de request.

Formatul prompt-ului trimis este de forma: full_prompt = [{“role”: “user”, “content”: payload},], payload fiind textul primit ca input. În cadrul request-ului, trebuie precizat ce model vrem să folosim, am folosit gpt-3.5-turbo.

De asemenea, am precizat și authorization key-ul, cel obținut de pe pagina celor de la OpenAI.

Pentru a trimite request-ul, am folosit requests.post cu parametrii: URL, modelul dorit și mesajul ca JSON și Authorization Key ca header.

Răspunsul primit poate fi extras sub formă de JSON din response.content.

Fiecare API poate avea un model de împachetare a datelor diferit, la fel și tipul de cerere care trebuie făcută pentru a afla informațiile.

Modelul de request de mai sus se aplică mai departe și pentru alte API-uri pe care le-am folosit în proiect cu modificările necesare aferente.

API - uri folosite:

ChatGPT

Vreme - https://openweathermap.org/api

Text-to-speech - https://rapidapi.com/voicerss/api/text-to-speech-1

Youtube API - https://github.com/FarhadGUL06/YoutubeAPI

ESP_32_STATS_API - https://github.com/FarhadGUL06/esp32_stats_api

Folosesc proiectul de la IC - ESP32 pe post de tracker pentru statistici de temperature, umiditate, calitatea aerului, acceleratia (In cazul deplasarii) din orice loc cu o conexiune la internet, care trimite date din 10 in 10 secunde catre API-ul ESP_32_STATS_API, date care pot fi preluate ulterior de catre GPT Multimedia Assistant.

Astfel, de oriunde in lume, cu acces la internet, se pot afla date despre un alt loc unde se afla acest tracker.

Schema de comunicare cu API-urile de mai sus:

Pe caile de tranzitie sunt desentate simboluri reprezentant calea prin care se transmit datele.

Un caz special de request pe care vreau să-l tratez este cel către API-ul de text-to-speech, cel care dă voce și viață dispozitivului. Cererea se face asemănător, doar că în loc de post am folosit requests.get.

Din cauza memoriei RAM destul de mici pe care o are un Pico, răspunsul trebuie parcurs pe iterații, neputând procesa tot răspunsul deodată, fiind vorba de un fișier audio. Astfel, iterez în blocuri de 2048 și salvez rezultatul pe cardul SD.

Aici intervine limitarea platformei de redare sunet, care nu poate procesa audio-ul pe bucăți, încercasem chiar și cu fișier wav, din cauza buffer-ului intern audio, nefolosind eficient memoria RAM de care dispune Pico.

Întrucât pot apărea probleme de memorie ocupată din cauza lucrului cu fișiere mari, ca metodă de siguranță și fiabilitate, la fiecare eroare pe care o întâlnește, dispozitivul repornește. Pentru cazurile de bază această situație nu este prezentă, dar în cazuri speciale poate apărea și de aceea este importantă o metodă care să asigure viața prelungită a dispozitivului.

Pentru acest dispozitiv, până în prezent, am ales să implementez următoarele: 1. Vremea în diverse orașe, în prezent sau în viitorul apropiat. Rezultatul este sub formă de json, care este parsat la ChatGPT iar acesta compune un text cu datele obținute 2. Discutat cu ChatGPT, compus povești cu input-ul utilizatorului ca punct de start și cam orice se poate face cu ChatGPT prin GUI. Toate răspunsurile primite de la ChatGPT vor fi transpuse în mp3 audio și redate prin boxă

3. Pot fi redate piese muzicale, cardul microSD fiind folosit ca zonă buffer unde se stochează piesa descărcată. Fișierul mp3 este prelucrat de către API-ul de youtube pentru a putea fi redat de Pico fără probleme (mono, 44.1kHz ca sample rate). Mai departe, piesa este stocată pe cardul SD și redată. Următoarea dată când va fi cerută aceeași piesă, aceasta nu va mai fi descărcată, ci va fi direct redată direct din cardul SD. Utilizatorul se conectează prin bluetooth la modulul HC-05. Acesta va folosi o aplicație de comunicare pe care o poate găsi pe magazin play. Un exemplu de aplicație compatibilă ar fi Arduino Bluetooth Control. Conectarea se face prin terminalul aplicației. Am ales să prefixez comenzile cu „!”. Utilizatorul poate să execute următoarele comenzi în terminal:

```
!help: afișează comenzile disponibile

!ping: se poate verifica conexiunea între telefon și Pico. Se va afișa un
„Pong!” pe ecran.

!restart / !reload: se rulează de la început fișierul code.py

!p <payload>: se poate reda piesa cu titlul specificat în <payload>

!tell <payload> / !say <payload>: dispozitivul va transmite audio inputul
primit

!mem: memoria RAM disponibilă

!wf <payload>: vremea în viitorul apropiat din orașul dat ca input, dacă
<payload> este gol,
atunci se va analiza vremea în viitorul apropiat din București
```

```
!w <payload>: vremea în acest moment din orașul dat ca input, dacă <payload>
este gol,
atunci se va analiza vremea în acest moment din București

!last: se va reda ultimul răspuns primit de la ChatGPT

!lan ro / en: se va seta limba pe ro / en, în funcție de cum este precizat.
Limba este utilizată și pentru API-ul text-to-speech, selectându-se vocea
potrivită limbii selectate

!stats: statistici de temperatură și umiditate din încăperea in care se afla
GPT Multimedia Assistant

!esp - statistici de temperatura, umiditate si altele din locul in care se
afla ESP32, cu afisare pe ecran

!espv - statistici transmise sub forma audio, prin boxa

<payload>: scriind direct textul, neprefixat de „!”, acesta va fi prelucrat
de ChatGPT și se va oferi un
răspuns
```

Spre exemplu, pentru o cerere de vreme, se trimite un API Request către API-ul de vreme, se așteaptă răspunsul.

După ce a fost primit, se trimite json-ul mai departe la ChatGPT pentru a fi prelucrat într-un text cursiv.

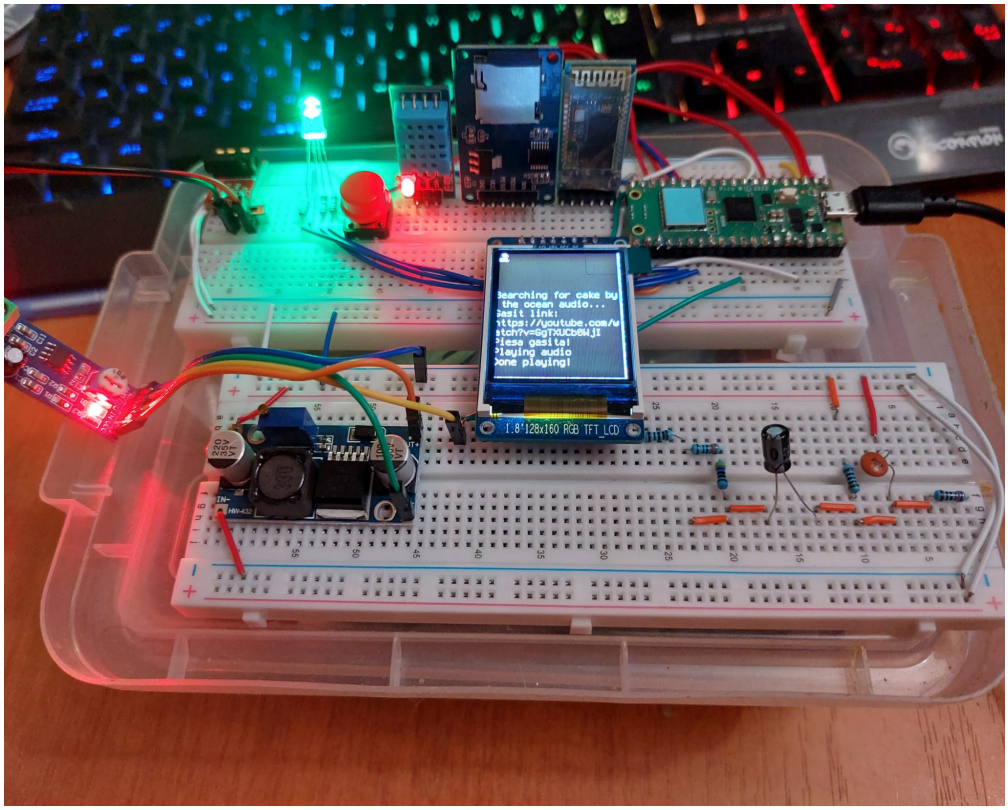
La final, răspunsul primit de la ChatGPT este trimis la API-ul de text-to-speech care prelucrează textul și trimite răspuns fișierul binar mp3 care este salvat pe cardul SD și redat ulterior.

## Rezultate Obţinute

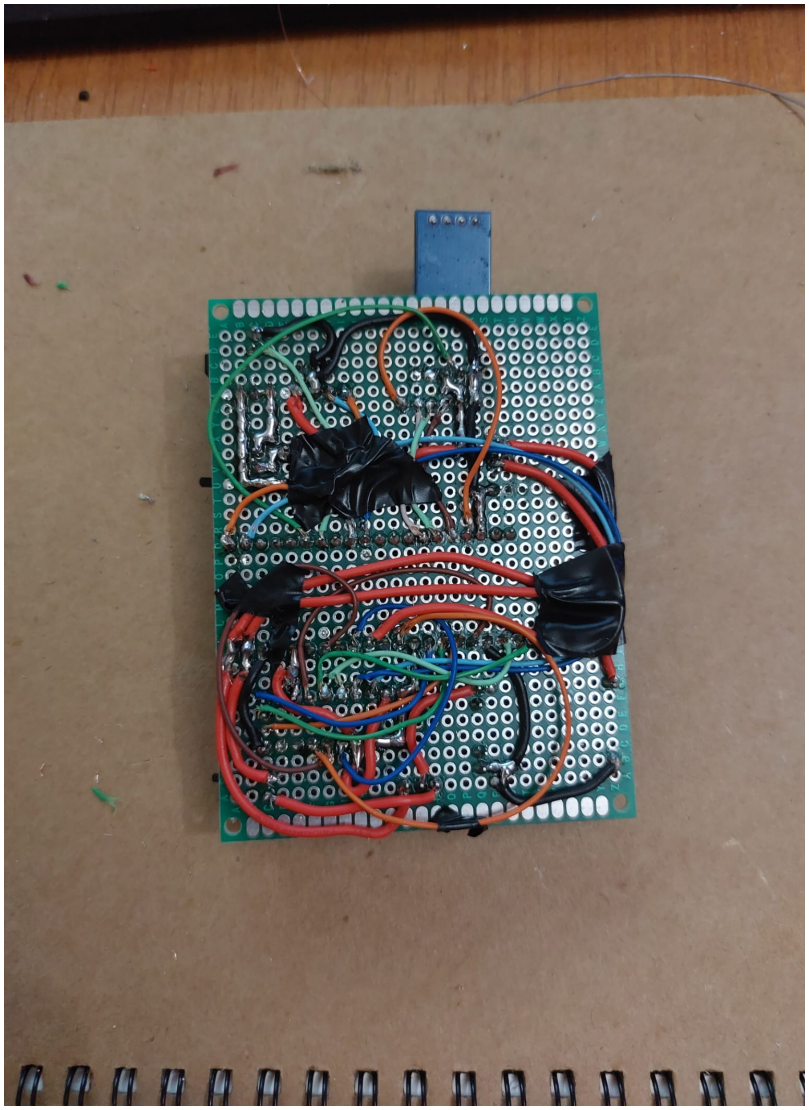
Un mic demo cu cateva din functionalitatile GPTMA in faza incipienta:

Prezentare GPT Multimedia Assistant

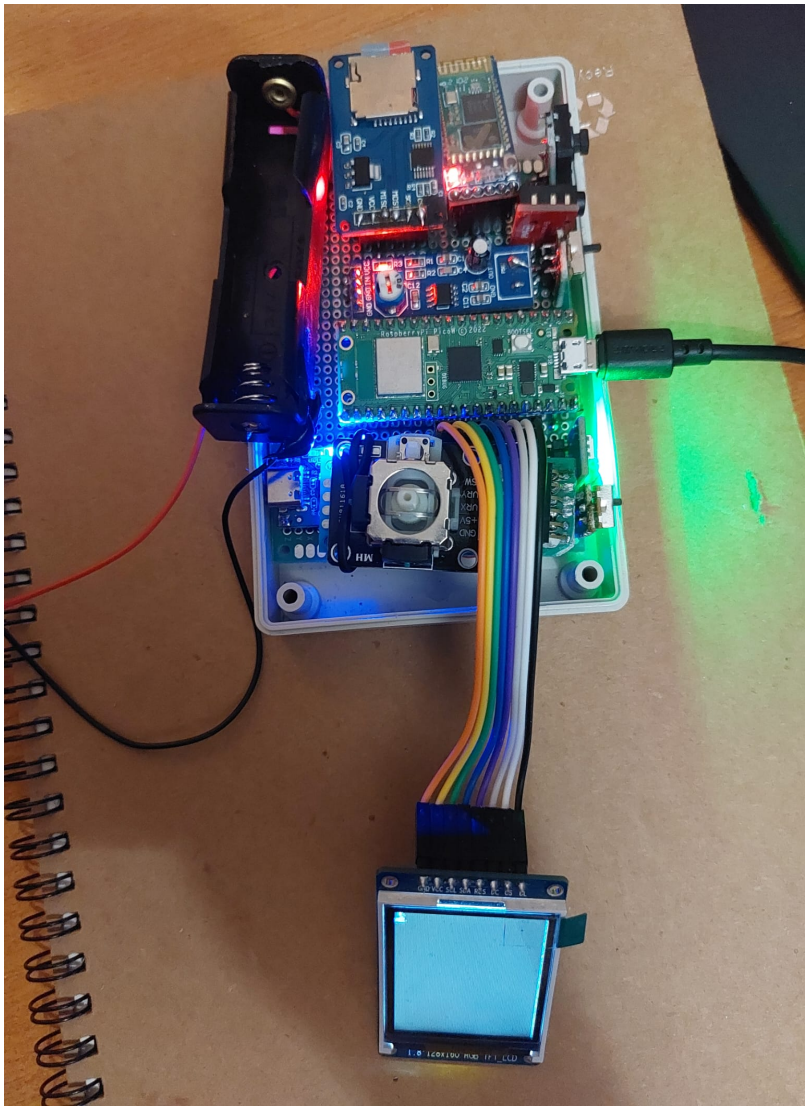
Poza cu proiectul la inceputul proiectarii:




Lipiturile de pe spatele placutei:



Poza cu proiectul lipit pe placaj de test:



Proiectul in stadiu final:


## Concluzii

Lucrul asupra acestui proiect a fost unul destul de indelungat.

De la inceput am plecat de la premisa ca daca nu iese o singura parte din proiect (comunicarea cu GPT de exemplu), se duce toata ideea si sensul proiectului.

Pana la urma, in prima faza reusisem sa proiectez software proiectul intr-o faza incipienta, pe care am prezentat-o si la SCSS.

Ulterior, am decis sa fac portabil proiectul (am incercat sa-l aduc o data la facultate montat pe breadboard si s-a facut praf).

Aici am mai consumat cateva zile cu lipiturile care se pot vedea in poza, au fost si altele nereusite.

Intr-un final a iesit un proiect complet functional, deschizator de drumuri pentru orientarea asupra proiectarii programelor mai sofisticate pe microcontrolere.

# Download

PicoAssistant - fisierul cod si biblioteci pt GPT Multimedia Assistant

YoutubeAPI - API-ul care proceseaza audio si trimite catre GPTMA

ESP32_STATS_API - API-ul care preia si trimite date de la ESP32

# Jurnal

03.05.2023 - generare pagina + introducere

07.05.2023 - adaugare componente folosite

19.05.2023 - prezentare proiectare hardware in laborator

20.05.2023 - adaugare implementare hardware

26.05.2023 - prezentare demo functionare in laborator

28.05.2023 - adaugare implementare software + concluzii

# Bibliografie/Resurse

[1] https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

[2] https://docs.circuitpython.org/projects/requests/en/latest/api.html

[3] https://platform.openai.com

[4] https://openweathermap.org/api

[5] https://rapidapi.com/voicerss/api/text-to-speech-1

[6] https://play.google.com/store/apps/details?id=com.broxcode.arduinobluetoothfree

Export to PDF