

Detector de curgere a cafelei în moka pot



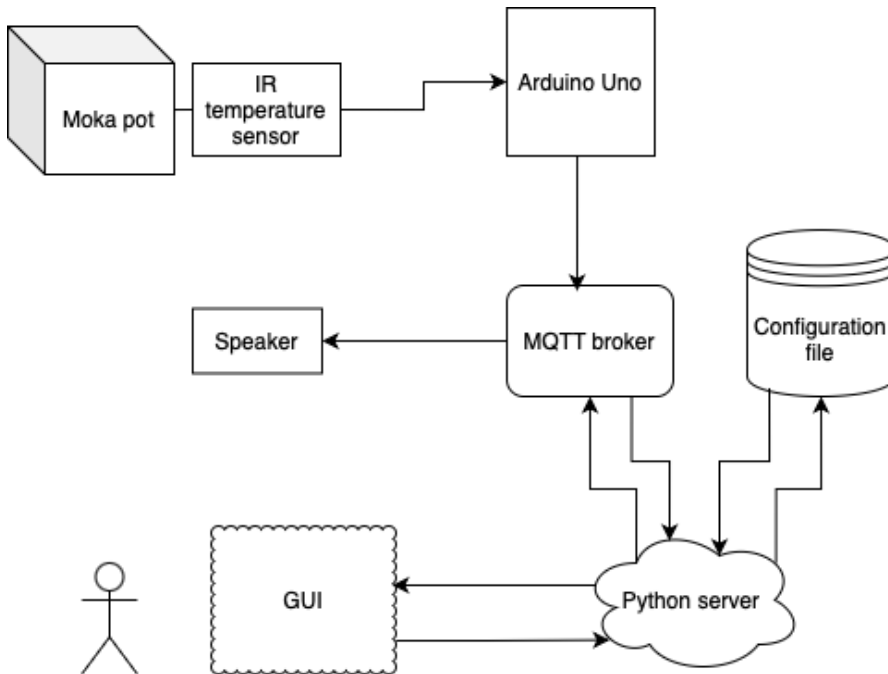
Introducere

Prietenul meu își face cafea la moka pot. După ce pune moka potul pe foc, pleacă din bucătărie și începe să facă altceva până începe cafeaua să curgă. Nu îl avertizează nimic atunci când curge cafeaua, iar el uită că a pus cafeaua pe foc. Așa că în multe dimineți fuge disperat de la birou în bucătărie ca să oprească focul. În multe dimineți face cafeaua de două ori.

Proiectul acesta presupune instalarea unui senzor lângă moka pot pentru a detecta momentul în care începe să curgă cafeaua și a îl notifica pe prietenul meu. Va primi o notificare pe laptop/telefon și un semnal sonor. La fiecare utilizare va putea introduce informații despre momentul când a fost notificat (prea devreme/prea târziu/la timp), astfel încât sistemul să se ajusteze în timp. De asemenea, va putea opri semnalul sonor de la distanță, folosind aceeași interfață grafică în care va vizualiza statisticile ajustării sistemului în timp.

Arhitectură

Diagrama topologiei rețelei:



Protocoalele de comunicație utilizate:

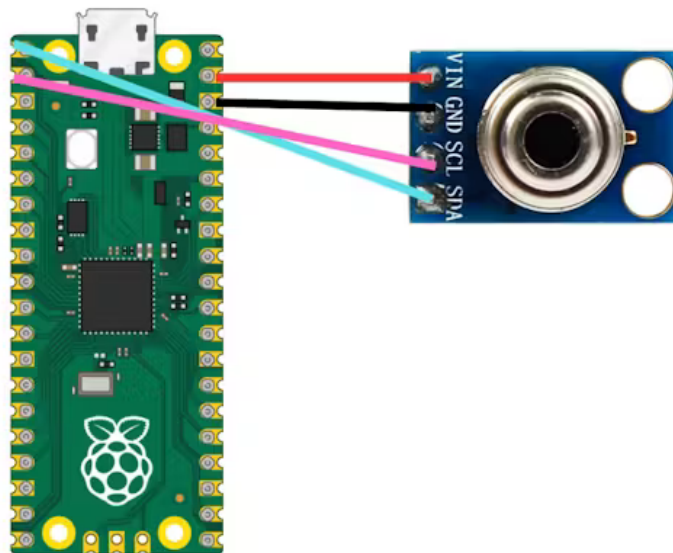
- HTTPS

Componente hardware:

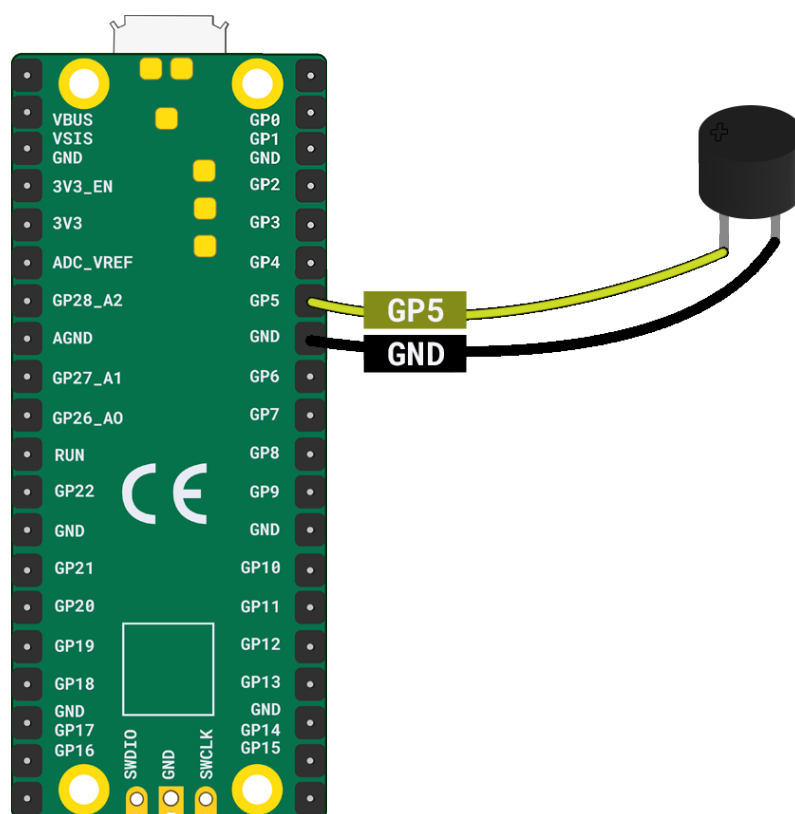
- Raspberry Pi Pico W
- Senzor de temperatură IR MLX90614ESF
- Difuzor
- Breadboard
- Fire pentru breadboard

Implementare

Configurarea hardware-ului:



1. Senzorul de temperatură IR
2. Buzzer pasiv



Configurarea software-ului:

1. Server care primește cereri HTTPS de la plăcuță și de la browser

```
python3 app.py
```

2. Client instalat pe plăcuță care trimite datele de la senzori
3. Client HTTPS (browser) care primește informațiile despre temperatură și despre activitatea buzzerului și afișează grafic datele

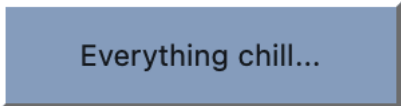
Sistemul de alertare:

1. Alertare sonoră prin buzzer

Buzzerul transmite un semnal sonor continuu începând din momentul în care cafeaua începe să curgă în moka pot.

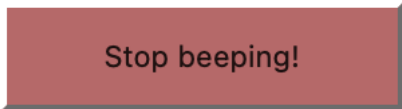
2. Alertare vizuală în interfața grafică

Atunci când este activat buzzerul, în interfața grafică se schimbă culoarea unui buton din albastru:



Everything chill...

în roșu:



Stop beeping!

Mesajul afișat pe buton este de asemenea schimbat astfel încât să sugereze activarea/dezactivarea buzzerului.

Buzzerul poate fi dezactivat apăsând butonul reprezentat mai sus. Click-ul va face butonul să se schimbe din roșu înapoi în albastru.

Dacă buzzerul a fost dezactivat, dar moka pot-ul nu a fost verificat (dispozitivul IoT nu a fost îndepărtat de moka pot), atunci buzzerul poate fi reactivat în mod automat. Acest lucru se întâmplă dacă moka pot-ul nu este verificat într-un interval de 7 secunde de la dezactivarea buzzerului.

Buzzerul poate fi dezactivat fie prin apăsarea butonului, fie prin îndepărtarea dispozitivului de moka pot (mai exact, prin scăderea temperaturii înregistrare de senzorul de temperatură IR).

Vizualizare și procesare de date

Vizualizarea datelor

Datele recepționate de senzorul de temperatură IR sunt afișate într-o interfață grafică intuitivă în formă de grafic al temperaturii în funcție de timp (ora la care a fost recepționată temperatura). Pe același grafic sunt reprezentate și momentele de activare, respectiv dezactivare a buzzerului.



Graficul poate fi analizat, mărit, micșorat, extins pe întreaga pagină, fotografiat sau salvat. Aceste funcționalități sunt disponibile prin intermediul butoanelor așezate în colțul din dreapta sus al paginii.

Procesarea datelor

Dispozitivul IoT recepționează informații legate de temperatura la care a ajuns moka pot-ul.

```
34 def get_temperature():
35     # TODO: get temperature from sensor
36     timestamp = time.time()
37
38     temp = 50 * (sin(timestamp * 0.1) + 1)
39
40     return temp
```

Temperatura este transmisă către server și salvată în memorie. De asemenea, plăcuța transmite și informații cu privire la starea buzzerului (activ/inactiv).

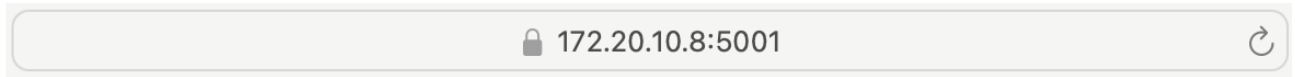
```
88 @app.post('/board')
89 def board_post():
90     global beeps, should_beep, board_data
91
92     beeps = request.json['beeps']
93     should_beep_board = request.json['should_beep']
94     temp = request.json['temp']
95
96     board_data.append((datetime.now(), temp, beeps))
97
98     if should_beep_board:
99         should_beep = True
100
101     return {'should_beep': should_beep}
```


Clientul HTTPS din browser se folosește de aceste date pentru a afișa graficul cu evoluția sistemului în timp.

```
46 @app.get('/graph')
47 def graph():
48     times = [t[0] for t in board_data]
49     temps = [t[1] for t in board_data]
50     beeps = [int(t[2]) for t in board_data]
51
52     plt = plot(Figure(
53         data=[
54             Scatter(x=times, y=temps, name='Temperatures', yaxis='y1'),
55             Scatter(x=times, y=beeps, name='Beeps', yaxis='y2')
56         ],
57         layout=Layout(
58             title='Temperature and Beeps Over Time',
59             xaxis=dict(
60                 title='Time',
61                 tickformat='%H:%M:%S',
62                 type='date'
63             ),
64             yaxis=dict(
65                 title='Temperature',
66                 range=[-5, 105],
67             ),
68             yaxis2=dict(
69                 title='Beeps',
70                 range = [-0.05, 1.05],
71                 overlaying='y',
72                 side='right'
73             )
74         )
75     ), output_type='div', include_plotlyjs='cdn')
76
77     return plt
```

Securitate

Securitatea transmisiei cererilor este asigurată prin protocolul HTTPS.



Certificatul SSL a fost generat folosind comanda:

```
openssl req -x509 -newkey rsa:4096 -keyout key.pem -out cert.pem  
-days 365 -nodes
```