Setting up a pico to send data to a Flask web server running on a Raspberry pi

Pico code: main.py

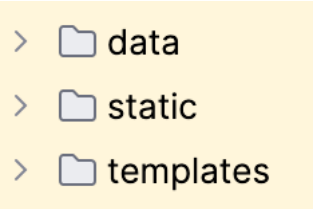
This program connects to local wifi and then sends data (posts) to the web server running on the raspberry pi.

```
import network
import time
from machine import Pin
import requests, json
from secrets import secrets
#import uasyncio as asyncio #test if not needed

ssid = secrets['ssid']
password = secrets['pw']


def connect():
    #Connect to WLAN
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    wlan.connect(ssid, password)
    while wlan.isconnected() == False:
        print('Waiting for connection...')
        time.sleep(1)

    ip = wlan.ifconfig()[0]
    print(f'Connected on {ip}')
    return f'Connected on {ip}'

def getTemp():
    #code for reading temp
    return temp

connect()

temp = getTemp()
data_from_pi = {'temp_1': temp}
response = requests.post('http://192.168.1.177:5000/pi', json=data_from_pi)
```

secrets.py

```
secrets = {
    'ssid': 'insert network name',
    'pw': 'insert network password'
}
```

Web APP



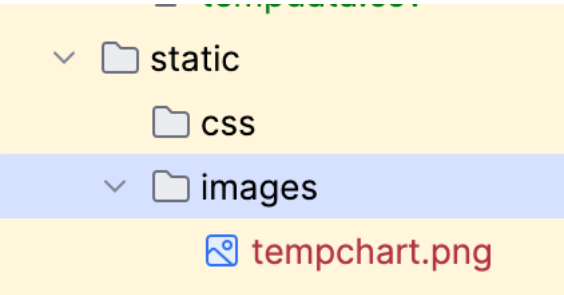
In the templates folder

----- index.html
----- chart.html

In the data folder

----- tempdata.csv

In the static folder



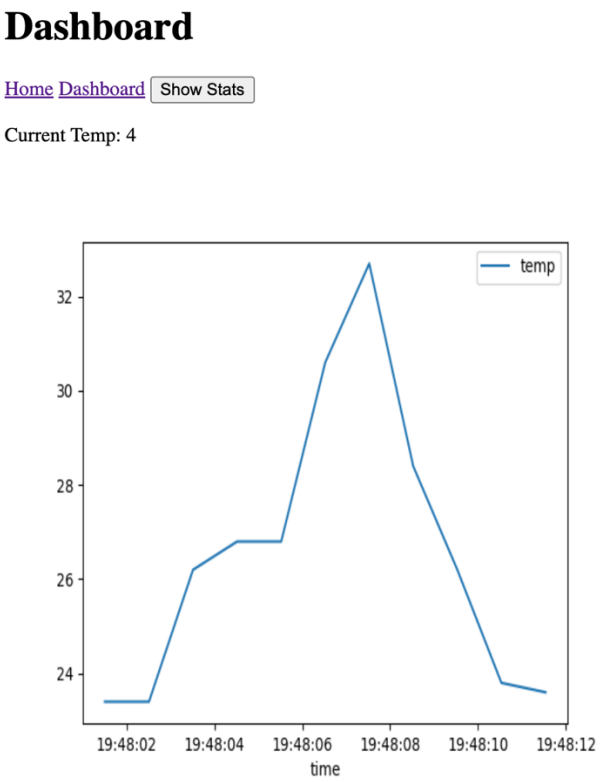


Chart.py

This code reads only today's temps and writes temp to a csv file when received from pico.

It also has the code to create a temp chart

```
import pandas as pd
import matplotlib.pyplot as plt
import csv
from datetime import datetime
def writeCSV(temp):
    with open('data/tempdata.csv', 'a', newline='') as file:
        writer = csv.writer(file)
        current_date = pd.Timestamp.now()
        writer.writerow([current_date, temp])
def readCSV():
    df = pd.read_csv("data/tempdata.csv")
    #Format the dataframe column
    df['date'] = pd.to_datetime(df['timestamp']).dt.date #create a
column for date
    df['time'] = pd.to_datetime(df['timestamp']).dt.time #create a
column for time

    #df['date'] = pd.to_datetime(df['date'], format='%Y-%m-%d')
    filtered_df = df[df['date'] == datetime.now().date()]
    print(filtered_df)
    return filtered_df

def tempChart(df):
    df.plot(kind='line', x='time', y='temp')
    plt.savefig('static/images/tempchart.png')



if __name__ == '__main__':
   pass
```

app.py

This is the code required in the flask app.py to get the pico data and display it on a html page, named ‘chart.html’

```
from flask import Flask, render_template, request,jsonify
from chart import *
from datetime import date

app = Flask(__name__)
currentTemp = ['4']

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/pi', methods=['POST'])
def pi():
    # getData
    currentTemp = request.json
    currentTemp[0] = pico_data['temp']
    writeCSV(currentTemp)
    return 'done'

@app.route('/chart')
def chart():
    df = readCSV()
    tempChart(df)

    return render_template('chart.html',
tempchart='static/images/tempchart.png',
                           temp=currentTemp[0])
@app.get('/update')
def update():
    df = readCSV()
    tempChart(df)
    return jsonify(temp=currentTemp[0])

if __name__ == '__main__':
    app.run(debug=True, port="80" ,host='0.0.0.0')
```

Chart.html looks like this:

```
<html>
<body>
<h1>Dashboard</h1>

<a href="{{url_for('index')}}">Home</a>
<a href="{{url_for('chart')}}">Dashboard</a>

<button onclick="start();">Show Stats</button>

<p id="temp"></p>

<img src={{tempchart}} height="500" width="500">

<script>
    no usages  new *
    function update(){
        //Fetch is a command to make a HTTP request to get json file
        fetch("/update").then(function(response){
            return response.json();

        }).then (function(data){
                console.log(data);
                document.getElementById("temp").innerText = "Current Temp: " + data['temp'];
            }).catch(function(err){
                console.log(err.message);
            });
      }
    no usages  new *
    function start(){
      setInterval(update, 1000); //updates every second
    }
    </script>

</body>
</html>
```