CompTech 10 - Weather Station Project

(Term 2, Feb 2024, By Evan Praël)

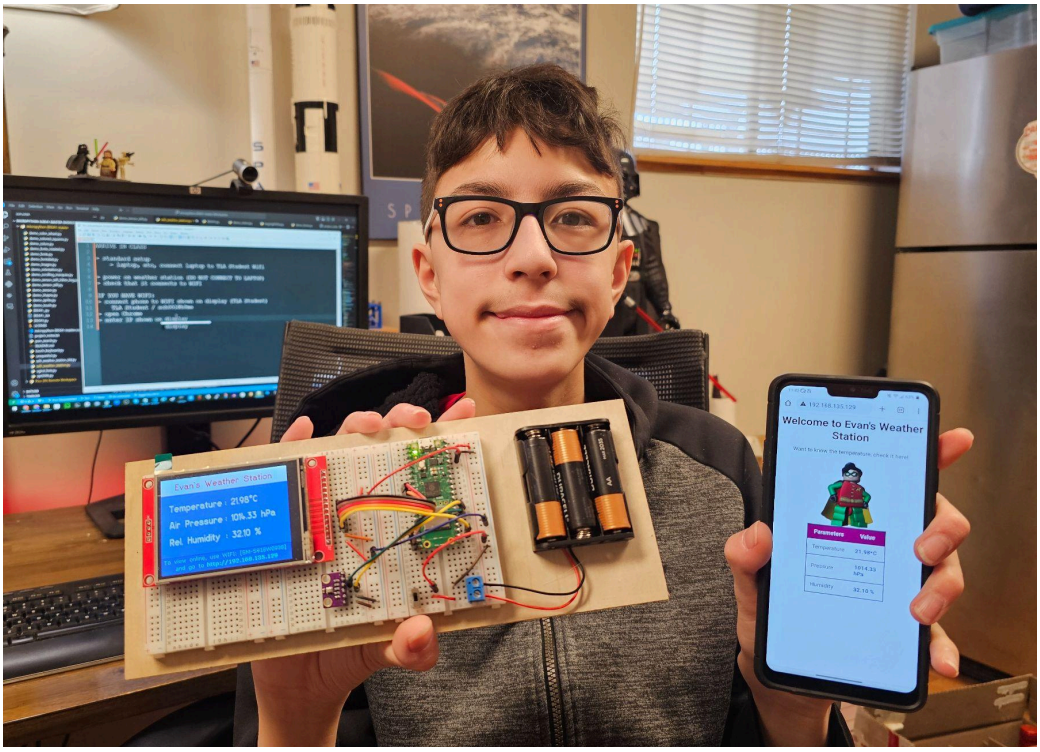
Introduction

For this project I built a simple weather station. It uses a Raspberry Pi Pico W, a weather sensor, and an LCD screen. The Pico also connects to wifi and can show the weather data on a web page.

The idea for this project came from another project I found online. That version only showed weather data on a web page. But my thinking was - what if I don't have wifi? Can I still see the weather data? So I added a LCD screen to the project. There I would show the weather data and the IP address users would need to view the web page.

Acknowledgements

Most of the project was done by myself, but I also had help from my dad who ordered the parts from Amazon and helped me sometimes when I got stuck on something. I also used Github Copilot a lot which is an AI tool that helps with coding. It's good at recognizing what's going on in the code and makes suggestions. It's also great for adding comments. I just have to start a comment and it automatically finishes it for me.



Finished project: Weather data showing on LCD screen and web page

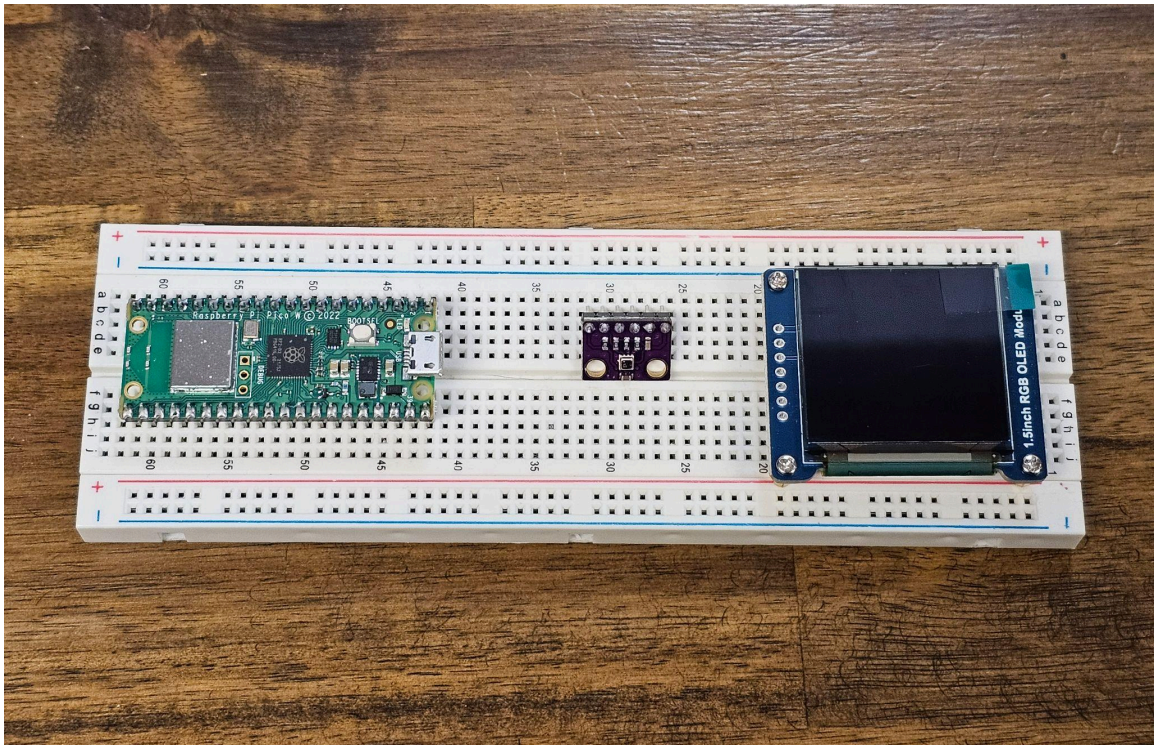
What I learned

I spent many hours on youtube and trying out examples. I started a collection of bookmarks that helped me finish the project:

- How to get started with the raspberry pi pico (and here)
- How to read data from a weather sensor (also this video, thx Mr Hewllet!)
- How to control a LCD display
- How to use fonts
- How to have the Pico be a web server (and here)
- How to have the Pico try to connect to different WIFIs
- How to create a simple web page with HTML (also in web server tutorials above)
- Why powering the Pico with a USB Battery Pack only works for a few seconds
- How to power the Raspberry Pi Pico with AA batteries.

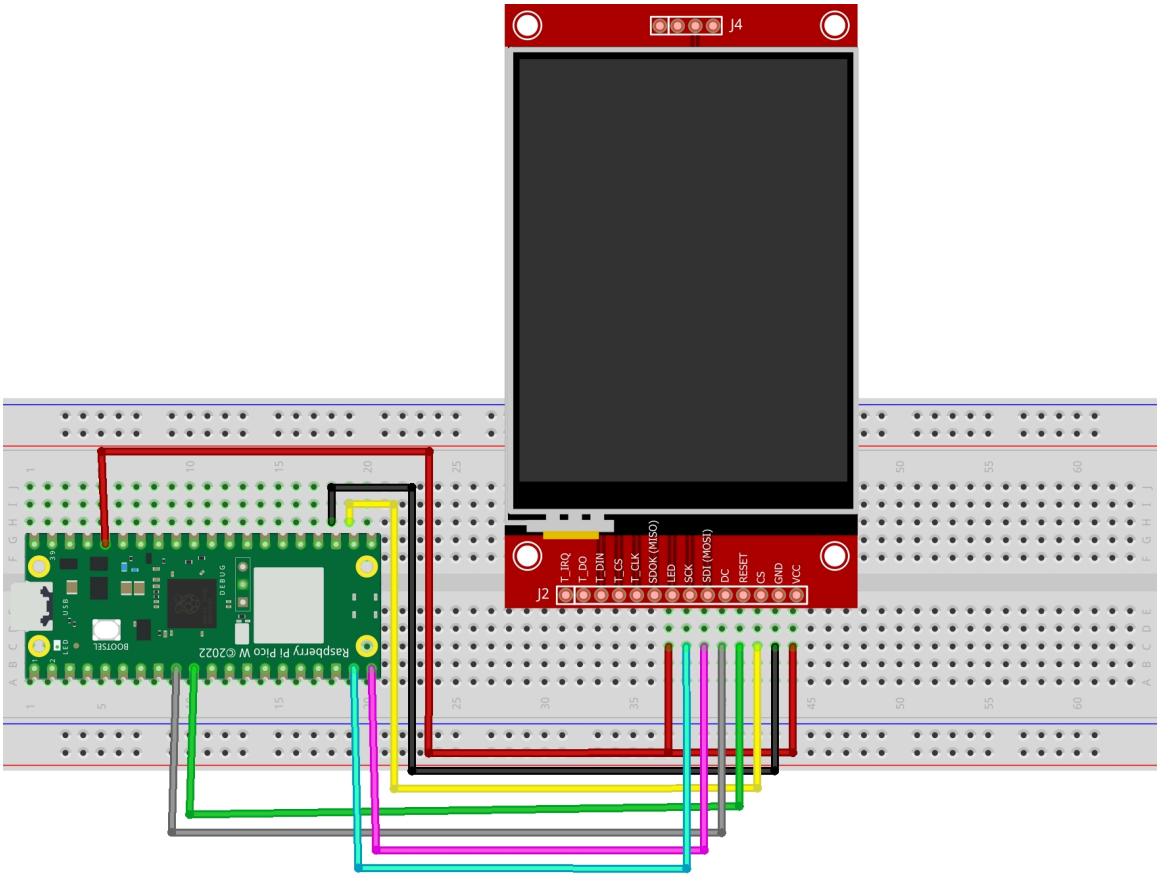
Getting Started

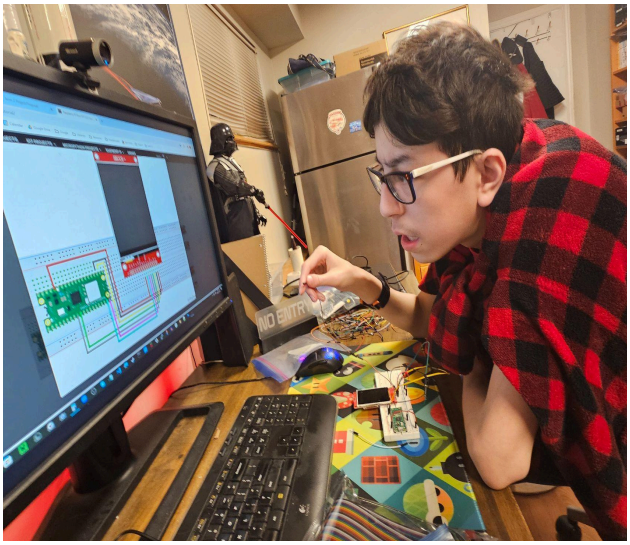
I started by ordering a Raspberry Pi Pico W, a weather sensor, and an OLED display. This is what I showed in my project proposal. I later switched to a LCD display because I couldn't find any libraries or tutorials for the OLED display.

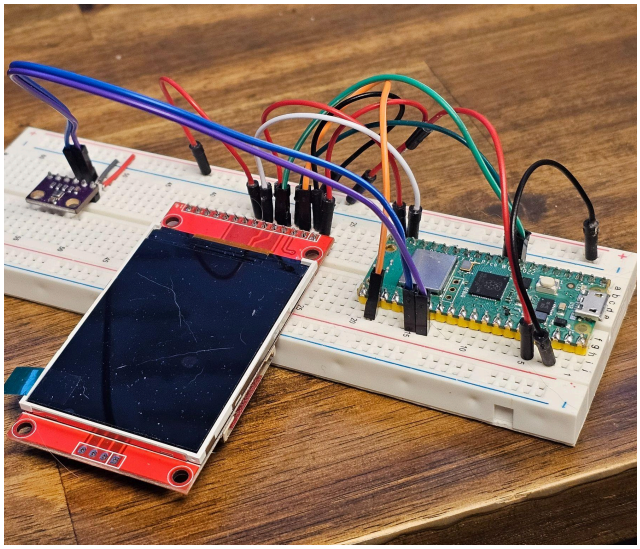


Connecting the LCD Display

This website had everything I needed to connect and program the LCD Display. It had the wiring diagram, the pins to connect to on the Pico, and the python library with lots of samples and demo code.

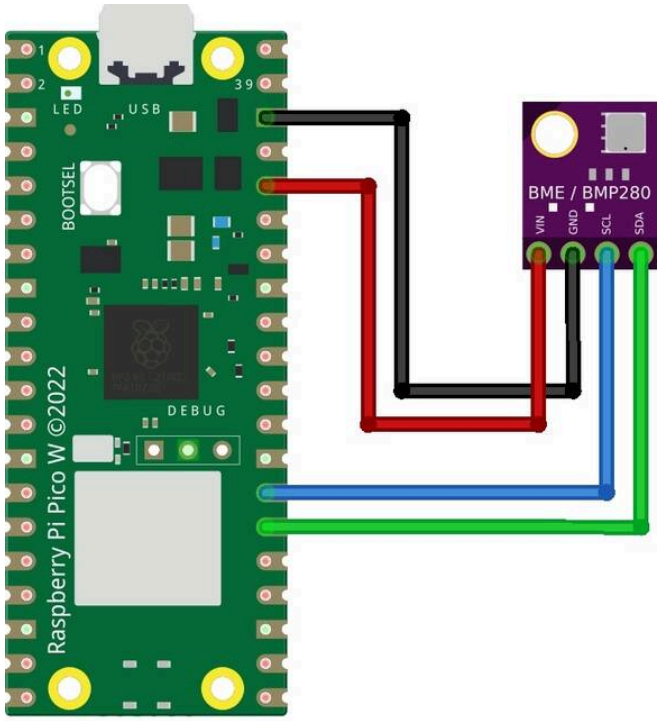


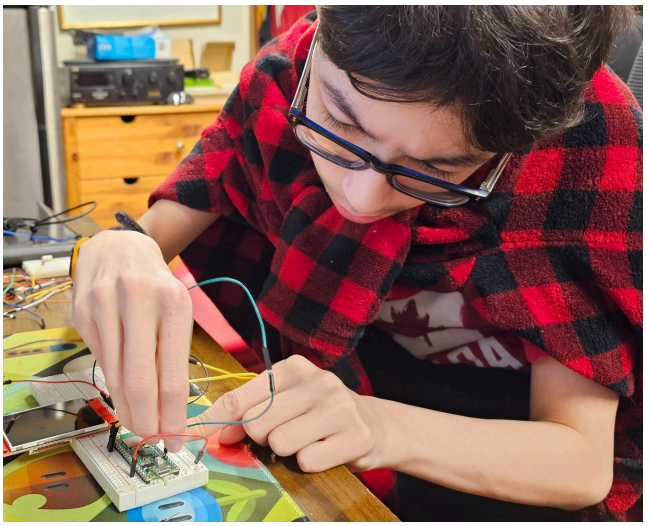


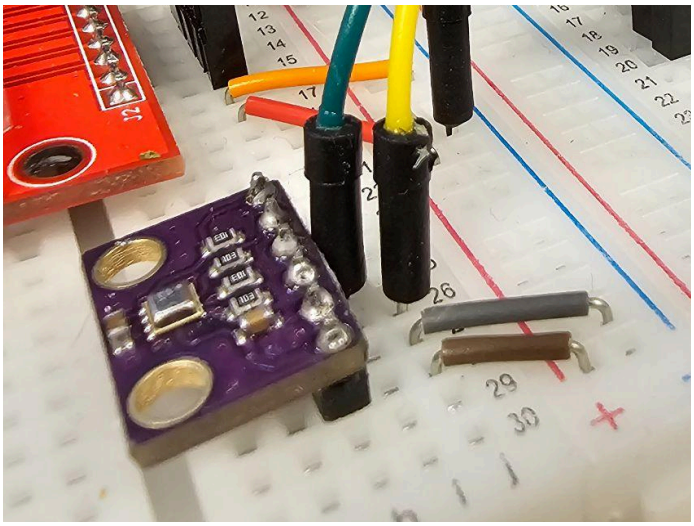


Connecting the Weather Sensor

This website had everything I needed to connect and program the weather sensor. Fortunately there isn't much to it - just 4 wires - power, ground, clock, and data. The website had the wiring diagram, the pins to connect to on the Pico, and the python library with demo code.





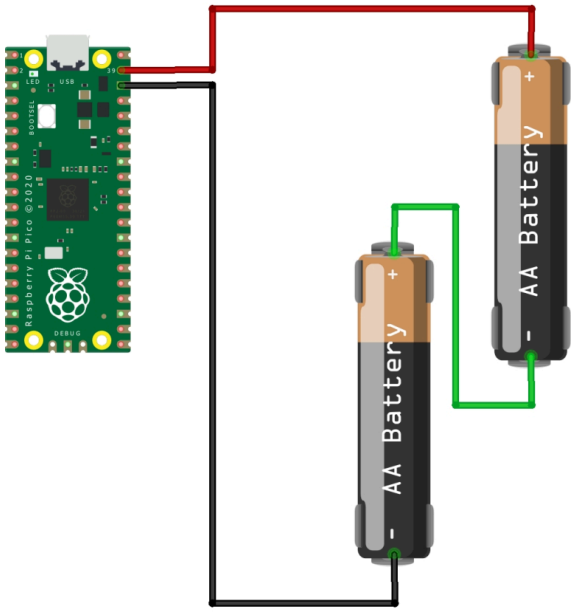


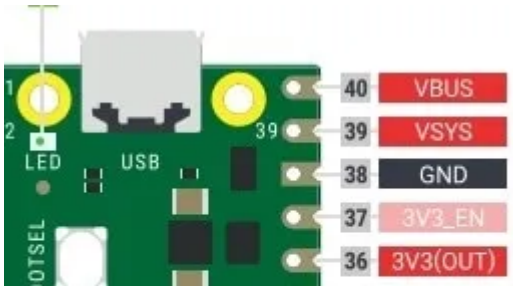
How it looked on my breadboard

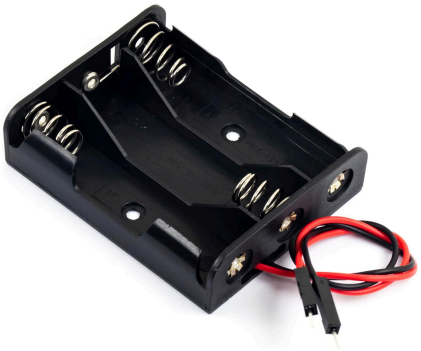
Powering the Pico With AA Batteries and a Switch

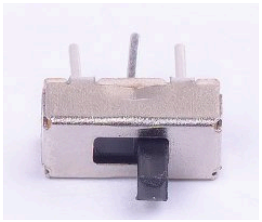
I really wanted to see the Pico running on its own - with no connection to a computer. I first tried using my USB power bank to power the Pico, but it would always shut off after 30 seconds. I found out the Pico uses so little power that the power bank thinks nothing is connected and just shuts off.

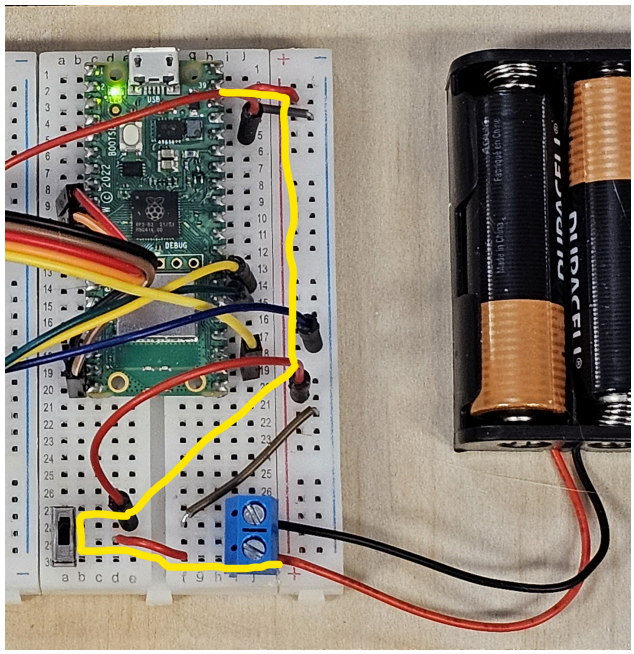
But AA batteries work well as this website explains. The Pico needs between 1.8V - 5.5V and the power needs to come in on pin 39 (VSYS).











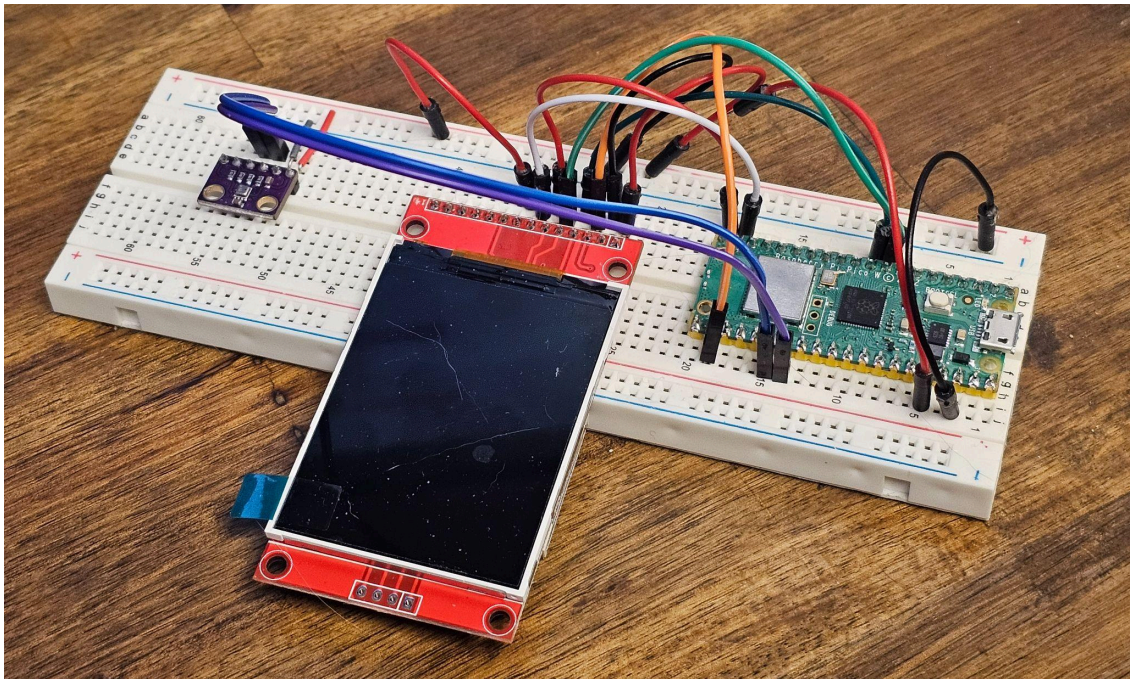
I decided to use 3 batteries and put them in a battery holder. I ran the power through a switch so I could easily turn the Pico on and off.

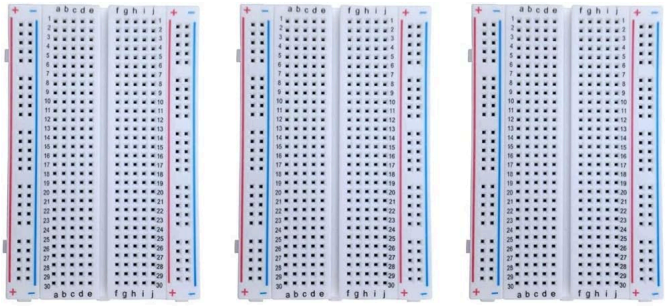
The yellow line to the left shows the path the power takes: into the breadboard, past the switch, and into pin 39.

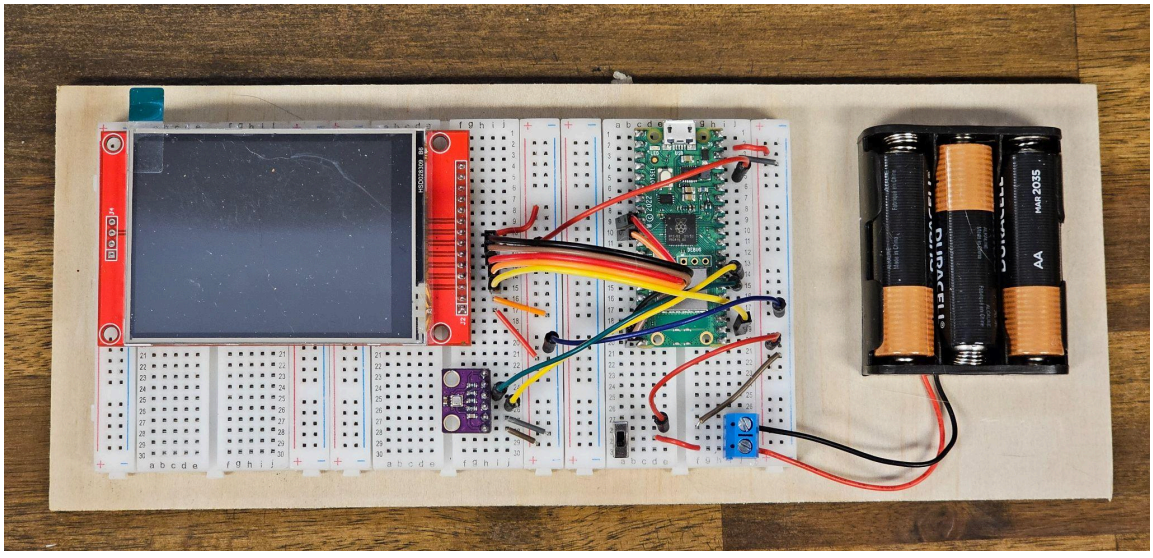
The battery pack only powers the Pico. The weather sensor and LCD display are powered by the pico from pin 36 (3.3V).

Final Changes

I wanted to clean things up a bit. The LCD was hanging off the side and the wires were very long. I found I could get a nicer layout by switching to three shorter breadboards side-by-side. Now the LCD could rest on something without hanging. Then I glued the breadboards and battery pack to a board.





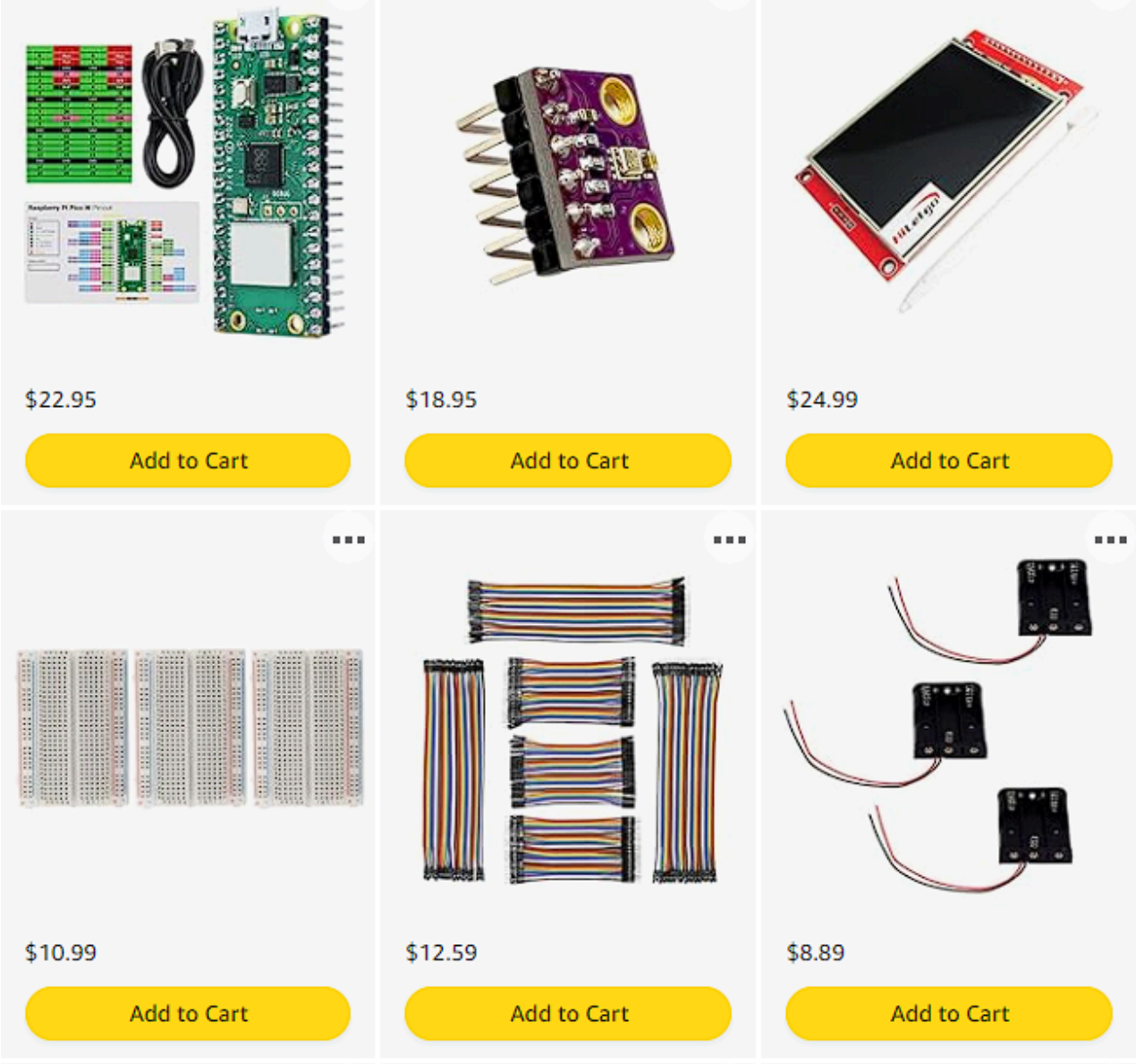


Parts List

I made a shopping list on Amazon that makes it easy to shop for the parts. Some items come in multiples like battery packs or wires.

Required Parts

- 1x Raspberry Pi Pico WH
- 1x Weather Sensor
- 1x LCD Display
- 1x Set of 3 short breadboards
- 1x Set of jumper wires
- 1x Battery Pack for 3 AA batteries
- 1x Screw Terminal (connect battery pack wires to breadboard)
- 1x Slide Switch
- 3x AA batteries

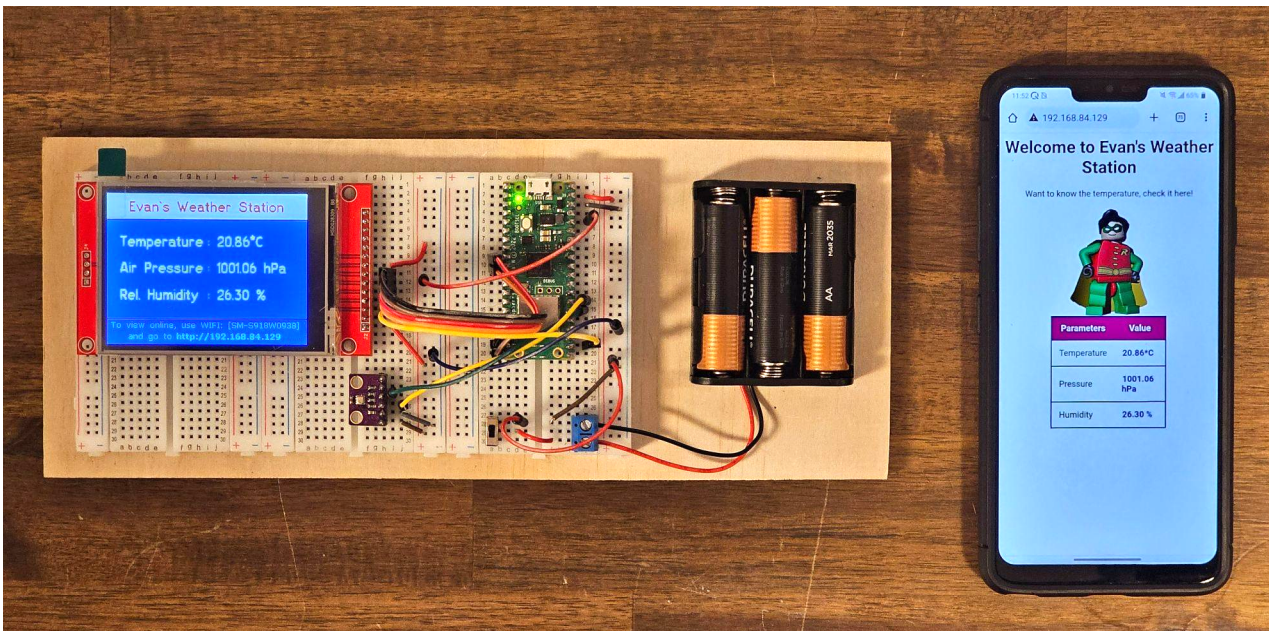


The Code - High Level Flow

These are the steps that take place inside the Pico when it gets turned on.

- Turn on LED to show there's power
- Initialize all the components (fonts, screen, sensor, etc)
- Connect to WIFI and let user know this is happening
- If connected:
    - Show weather data and IP Address that browsers have to connect to

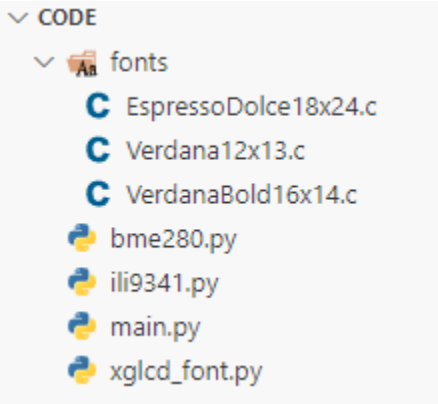
  Otherwise
    - Show just the weather data with message "WIFI not available"
- Enter a forever loop:
    - Wait up to 30 seconds or until someone connects from the web
    - Read the latest weather data from the sensor
    - If someone is connecting from the web
        - Build a web page with the weather data and send it back
    - Update the display with the latest weather data



Final Output: weather data on LCD screen and web page.
Bottom of LCD screen also shows how to view on the web.

The Code - Python Source

This is the collection of files on the Pico that runs everything. These files can be found on my [github page]



3 fonts used on LCD display.

bme280.py - library for weather sensor
ili9341.py - library for LCD Display
main.py - main program (code below)
xglcd_font.py - library for loading and displaying fonts

This is the program that runs on startup, loads all the libraries, and kicks everything off.

main.py

```
# CompTech 10 Term 2 Project - Raspberry Pi Pico Weather Station with LCD Display & WIFI
# By Evan Prael, Feb 15, 2024
#
# This program runs on a Raspberry Pi Pico W, reads a weather sensor and displays that data
# on a LCD Screen. It also connects to WIFI and displays it on a web page.
#
# Code and libraries used from these websites:
#
# web server, weather sensor:
# https://how2electronics.com/bme280-raspberry-pi-pico-w-web-server-weather-station/
# https://datasheets.raspberrypi.com/picow/connecting-to-the-internet-with-pico-w.pdf#page=22
#
# LCD display & fonts
# https://diyprojectslab.com/raspberry-pi-pico-tft-lcd-touch-screen-tutorial/
# https://github.com/rdagger/micropython-ili9341/tree/master
#
# All project files on Github
# https://github.com/eprael/weatherstation
#
# Demo Video on Youtube:
# https://youtu.be/lZpsUI2AZsA
#
# Parts list on Amazon
# https://www.amazon.ca/hz/wishlist/ls/19ZVP4QBNUOJP?viewType=grid
```

```
# import libraries
from machine import Pin, I2C, SPI       # pico board
import time, sys                        # time delays
import socket, network                  # wifi and web server
import bme280                           # weather sensor
from ili9341 import Display, color565   # LCD display
from xglcd_font import XglcdFont        # fonts


# turn on LED first thing and use as a power light
led = machine.Pin("LED", machine.Pin.OUT)
led.on()

print('initializing devices...')

# setup weather sensor and pins
i2c=I2C(0,sda=Pin(20), scl=Pin(21), freq=400000)

# setup LCD display and pins
spi = SPI(1, baudrate=40000000, sck=Pin(14), mosi=Pin(15))
display = Display(spi, dc=Pin(6), cs=Pin(17), rst=Pin(7))


# setup fonts for display
print('loading fonts...')
smallFont = XglcdFont('fonts/Verdana12x13.c', 12, 13)
smallFontBold = XglcdFont('fonts/VerdanaBold16x14.c', 16, 14)
largeFont = XglcdFont('fonts/EspressoDolce18x24.c', 18, 24)

# variables for screen colors
backgroundColor = color565(0, 0, 255)  # blue
headerColor = color565(255, 0, 0) # red
headerBackground = color565(255, 255, 255) # white
footerColor = color565(0, 255, 255) # cyan
footerBackground = color565(0, 0, 180) # darkblue
textColor = color565(255, 255, 0) # yellow
shadowColor = color565(0, 0, 0) # black

# main weather variables
temperature = ''
pressure = ''
humidity = ''

# wifi networks to try (SSID & password)
wifiNetworks = {
   'HomeWifi' : 'xxxxxxxx',
   'SchoolWifi' : 'xxxxxxxx'
}

# setup wifi
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
```

```
#disable power saving for wifi
wlan.config(pm = 0xa11140)

# ------------------------------------ functions ---------------------------------------

# refresh weather data from sensor
def refresh_weatherData():
    global temperature, pressure, humidity
    try:
        bme = bme280.BME280(i2c=i2c)
        temperature = bme.values[0]
        pressure = bme.values[1]
        humidity = bme.values[2]
    except:
        # sensor not connected or not working
        temperature = "n/a"
        pressure =  "n/a"
        humidity =  "n/a"


# create a web page with the weather data in it
# this gets sent back to the browser
# html borrowed and modified from
#   https://how2electronics.com/bme280-raspberry-pi-pico-w-web-server-weather-station/

def web_page():
    html = """
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="icon" href="data:,">
    <style>
        body { text-align: center; font-family: "Helvetica", Arial;}
        table { border-collapse: collapse; width:55%; margin-left:auto; margin-right:auto; }
        th { padding: 12px; background-color: #87034F; color: white; }
        tr { border: 2px solid #000556; padding: 12px; }
        tr:hover { background-color: #bcbcbc; }
        td { border: none; padding: 14px; }
        .sensor { color:DarkBlue; font-weight: bold; background-color: #ffffff; padding: 1px;

        img{display: block; margin-left: auto; margin-right: auto;}
    </style>
    <meta http-equiv="refresh" content="60">
    <meta name="viewport" content="width=device-width, initial-scale=1">
</head>
<body>
    <h1>
        Welcome to Evan's Weather Station
    </h1>
    Want to know the temperature, check it here! <br>
    <br>
```

```
    <img src="https://static.wikia.nocookie.net/legobatman/images/a/af/Batman1_Robin.png"
height="200" />

    <table>
    <tr>
        <th>Parameters</th><th>Value</th>
    </tr>
    <tr>
        <td>Temperature</td>
        <td><span class="sensor">""" + temperature + """</span></td>
    </tr>
    <tr>
        <td>
            Pressure</td><td><span class="sensor">""" + pressure + """</span>
        </td>
    </tr>
    <tr>
        <td>Humidity</td>
        <td><span class="sensor">""" + humidity + """</span></td>
    </tr>
    </table>
</body>
</html>"""
    return html


# Try each wifi network until connected. return True if connected, False if not

def connect_to_wifi():
    wlan.disconnect()
    time.sleep(1)

    # try different wifis until connected
    for ssid, password in wifiNetworks.items():
        # sometimes the pico doesn't connect on the first try, so try twice
        for i in range(2):
            print('connecting to network', ssid, ',', password, '...')
            wlan.connect(ssid, password)
            wait = 0
            #while not wlan.isconnected() and wait < 10:
            while wlan.status() >= 0 and wlan.status() < 3 and wait < 10:
                print ('waiting for connection... status = ', wlan.status())
                time.sleep(1)
                wait += 1

            if wlan.isconnected():
                print(f"connected. IP: {wlan.ifconfig()[0]}")
                return True
            else:
                print(f"connection to {ssid} failed. status = {wlan.status()}")
    return False
```

```
# display-print: writes text to the display using display.draw_text.
def dprint (x,y,foreColor,backColor,font,msg):
    display.draw_text(y, 320-x, msg, font, foreColor, landscape=True, background=backColor)


# initialize the display
# draw title and show "connecting to wifi" message
def initialize_display():
    print('initializing display...')
    display.clear(backgroundColor)
    display.fill_rectangle(0,0,40,320,headerBackground)
    display.fill_rectangle(41,0,1,320,shadowColor)
    dprint (35,10, headerColor, headerBackground, largeFont, "Evan`s Weather Station")
    dprint (54,96, textColor, backgroundColor, largeFont, 'Connecting to WIFI')
    dprint (90,144, textColor, backgroundColor,largeFont, 'Please wait...')

# show wifi info at bottom of screen
def setup_display():
    # clear all but the header
    display.fill_rectangle(43,0,197,320,backgroundColor)

    # create footer bar
    display.fill_rectangle(200,0,40,320,footerBackground)
    display.fill_rectangle(199,0,1,320,shadowColor)

    # show what to do to view weather data online
    if (wlan.isconnected()):
        dprint (5,204,footerColor,footerBackground, smallFont,
            f"To view online, use WIFI: [{wlan.config('essid')}]")
        dprint (33,222,footerColor,footerBackground, smallFont, f"and go to ")
        dprint (110,221,footerColor,footerBackground, smallFontBold,
            f"http://{wlan.ifconfig()[0]}")
    else:
        dprint (96,215,footerColor,footerBackground, smallFont, f"WIFI not available")

    # display weather headings
    dprint (20, 65, textColor, backgroundColor, largeFont, 'Temperature')
    dprint (20, 107, textColor, backgroundColor, largeFont, 'Air Pressure')
    dprint (20, 150, textColor, backgroundColor, largeFont, 'Rel. Humidity')
    # display colons separately so they can be aligned
    dprint (160, 63, textColor, backgroundColor, largeFont, ':')
    dprint (160, 104, textColor, backgroundColor, largeFont, ':')
    dprint (160, 148, textColor, backgroundColor, largeFont, ':')


# refresh and display weather data
def display_weatherData():
    print('updating display...')
    refresh_weatherData()
    dprint (175, 65, textColor, backgroundColor, largeFont, f"{temperature}  ")
    dprint (175, 107, textColor, backgroundColor, largeFont, f"{pressure}  ")
    dprint (175, 150, textColor, backgroundColor, largeFont, f"{humidity}  ")
```

```
# with wifi connected, run both the display and web server
def run_display_and_webserver():

    # first update the display, then update it every 30s while waiting for a web connection
    display_weatherData()

    print('launching web server...')

    # setup web server on port 80
    addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    server.bind(('', 80))
    server.listen(5)
    print('listening on', addr)

    stay_in_loop=True

    # while waiting for a web connection, break out every 30s to update the display
    server.settimeout(30)
    while stay_in_loop:
        try:
            print('waiting for connection...')
            conn, addr = server.accept()

            print('connection from %s' % str(addr))
            conn.settimeout(3.0)
            try:
                print('receiving request...')
                request = conn.recv(1024)
                print(str(request))
            except:
                print('receiving request timed out')

            print('getting web page')
            returnHTML = web_page()
            print(returnHTML[0:100])
            print('sending header...')
            conn.send('HTTP/1.1 200 OK\r\nContent-type: text/html\r\n\r\n')
            print('sending html...')
            conn.sendall(returnHTML)
            print('closing connection')
            conn.close()

        except OSError as e:
            # error 110 is the timeout that happens every 30s when waiting for a web
            # connection so just update the display and continue
            if e.errno == 110:
                display_weatherData()
            else:
                print('OSError', e)
        except Exception as e:
```

```
            print ('Exception', e)
            conn.close()
            print('connection closed')
        except KeyboardInterrupt:
            # if the user presses ctrl-c or presses stop in vscode,
            print('KeyboardInterrupt')
            keepListening = False
            print('connection closed')
            stay_in_loop = False

    print('closing server...')
    server.close()
    time.sleep(1)
    print('disconnecting wifi...')
    wlan.disconnect()
    time.sleep(1)
    print('cleaning up...')
    wlan.active(False)
    print('web server ended.')

def run_display_only():
    print('launching display loop...')
    stay_in_loop=True
    while stay_in_loop:
        try:
            display_weatherData()
            time.sleep(30)
        except Exception as e:
            print ('Exception', e)
        except KeyboardInterrupt:
            print ('Keyboard interrupt')
            stay_in_loop = False
    print('display loop ended.')


# ----------------- main code -----------------

initialize_display()
connect_to_wifi()
setup_display()

if wlan.isconnected():
    run_display_and_webserver()
else:
    run_display_only()

# cleanup
display.clear()
led.off()

print('done.')
```