Step 1: Gather Supplies and Tools

Refer to the supplies list provided earlier. You'll also need tools like a soldering iron, solder wire, wire cutters, a multimeter for testing, and a 3D printer (or access to one) for the enclosure. Ensure your battery is a 3.7V Li-ion/LiPo (e.g., 18650 or flat pack) compatible with the TP4056.

Step 2: 3D Print the Enclosure

Design or download a custom enclosure to house the ESP32, LCD, Vero board, battery, and switch. The enclosure in your photos appears custom-printed—measure your components (ESP32 board ~38x25mm, 20x4 LCD ~98x60mm) and create a simple case using software like Tinkercad or Fusion 360.
Or you can use my design.

Print with PLA Plus filament at 0.2mm layer height for durability. Include cutouts for the Micro USB (charging), switch, and LCD visibility.

Step 3: Solder the Components on Vero Board

Use the Vero board to create a permanent circuit. This avoids loose wires and makes the setup compact.

Wiring Overview:

- ESP32 to LCD (I2C): Connect LCD VCC to ESP32 5V, GND to GND, SDA to ESP32 GPIO21 (default I2C SDA), SCL to GPIO22 (default I2C SCL).

- Power Management: Connect the TP4056 IN+ and IN- to the Micro USB breakout (for charging input). Battery positive to TP4056 B+, negative to B-. TP4056 OUT+ to one side of the switch, the other side of the switch to ESP32 VIN (5V input). TP4056 OUT- to ESP32 GND. This allows on/off control and charging.

- Use wires for connections. Solder components securely, bridging strips on the Vero board as needed. Test continuity with a multi-meter to avoid shorts.

For the ESP32-LCD wiring diagram, refer to this image: [Wiring I2C LCD to ESP32](https://lastminuteengineers.com/wp-content/uploads/2019/06/Wiring-I2C-LCD-Display-to-ESP32.png).

For TP4056 battery charging setup with ESP32 (adapt by adding the switch in series on OUT+), use this circuit diagram: [ESP32 with TP4056 and Battery](https://i0.wp.com/randomnerdtutorials.com/wp-content/uploads/2019/04/esp32-solar-powered-circuit-1.png?quality=100&strip=all&ssl=1).

For soldering technique on Vero board: Plan your layout to minimize wire crossings. Place components, bend leads to hold them, solder from the copper side, and trim excess. An example of perfboard soldering can be found in tutorials like this one: [Using Perfboard Guide](https://www.instructables.com/Using-Perfboard/). (Focus on steps 5-10 for soldering tips.)

Step 4: Install Arduino IDE and ESP32 Support

Download and install the Arduino IDE from arduino.cc.

To add ESP32 support:

1. Open Arduino IDE and go to File > Preferences.

2. In the "Additional Boards Manager URLs" field, add: `https://dl.espressif.com/dl/package_esp32_index.json`.

3. Go to **Tools > Board > Boards Manager**, search for "ESP32", and install the package by Espressif.

4. Select your board: **Tools > Board > ESP32 Arduino > ESP32 Dev Module** (or similar).

For visual steps, see this image: [Opening Preferences in Arduino IDE](https://randomnerdtutorials.com/wp-content/uploads/2018/08/installing-esp32-arduino-ide-windows-mac-os-x-linux-open-preferences.png).

Step 5: Install Required Libraries

In Arduino IDE, go to **Sketch > Include Library > Manage Libraries**. Install:

- LiquidCrystal_I2C

- WiFi

- WiFiManager

- HTTPClient

- ArduinoJson

These are used in the code for LCD control, WiFi setup, API fetches, and JSON parsing.

Step 6: Modify and Prepare the Code

Copy the provided code into a new Arduino sketch. Make these changes:

- Update `weatherURL` to your city, e.g., `"https://wttr.in/YourCity?format=%t+%C"`.

- Update `calendarURL` to your actual endpoint (e.g., a Render.com hosted script for Google Calendar events).

- Adjust `gmtOffset_sec` for your timezone (e.g., 3600 * hours offset from GMT).

- If using a different LCD address, change `LiquidCrystal_I2C lcd(0x27, 20, 4);` (scan with an I2C scanner sketch if needed).

Save the sketch.

Step 7: Upload the Code to ESP32

Connect the ESP32 to your computer via Micro USB cable (without battery connected yet to avoid issues).

1. Select the correct port: **Tools > Port** (e.g., COM3 on Windows).

2. Press and hold the BOOT button on ESP32 if it doesn't upload automatically.

3. Click the Upload button (right arrow) in Arduino IDE.

For the upload process image: [Uploading Sketch to ESP32](https://randomnerdtutorials.com/wp-content/uploads/2018/08/arduino-ide-uploading-wifiscan-example-to-esp32.png).

Once uploaded, open the Serial Monitor (115200 baud) to check for errors.

Step 8: Assemble the Device

1. Mount the Vero board, ESP32, LCD, battery, TP4056, and switch into the 3D-printed enclosure.

2. Secure with screws or hot glue if needed. Ensure the LCD is visible and the switch/Micro USB are accessible.

3. Connect the battery last.

Step 9: Power On and Test

Flip the switch to power on. The device will create a WiFi access point ("ESP32-Clock"). Connect to it via phone/PC, enter your home WiFi credentials in the portal.

- The LCD should show time/date from NTP.

- Weather will fetch from wttr.in (e.g., "7 C Light Rain Shower" in your photos).

- Events from your calendar API (handle "API Error" if endpoint fails).

- Charge via Micro USB; the TP4056 LEDs indicate charging status.

If issues arise: Check wiring, Serial Monitor for debug, or restart. Enjoy your custom weather clock!