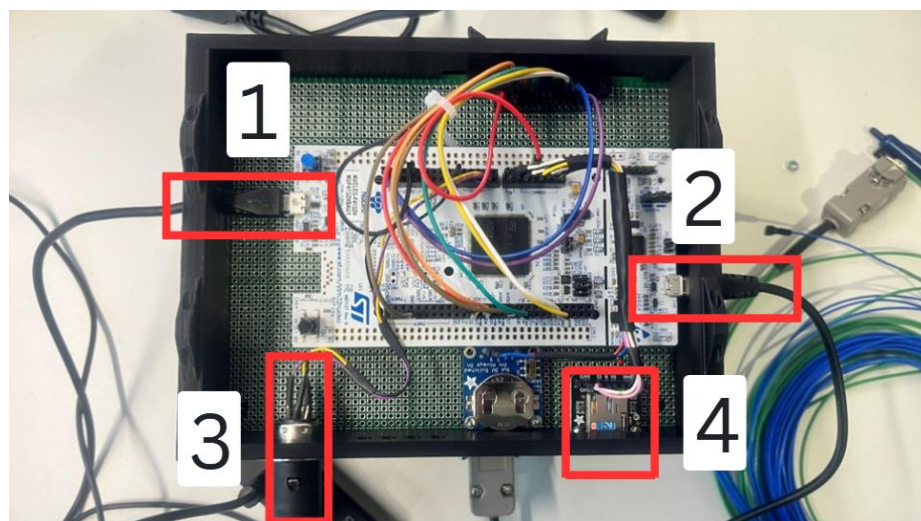


01 | Project Setup

1.1 MCU (Microcontroller Unit) Connections

CRITICAL: Make sure the USB Hub Figure 3 is not plugged into the laptop before doing this step!

Plug the 3 connections to the 3D printed box as shown below in Error! Not a valid bookmark self-



reference.:

Figure 1: MCU connections.

Label 1: USB connection to allow the E: drive to be seen.

Label 2: USB connection to power the MCU.

Label 3: DIN connector – this connection should feel intuitive.

Label 4: SD card connection – this connection should not be touched as the SD card should already be in.

IMPORTANT! – When connecting label 1, make sure the “teeth” are facing downwards.

IMPORTANT! – When connecting label 2, make sure the “teeth” are facing upwards.

2.1 Ribbon Cable Connection

Connect the 26 Pin IDC connectors up. An example of the connection is shown in

Commented [TT2R1]: Sweet thank you, will continue checking over the document.

Commented [SB3]: Can these steps be put in an SOP? I will email you an example to use as a template

Commented [TT4R3]: Yup can do

Figure 2.

IMPORTANT! Label 1 will always need to be connected in.

To enable **16 thermocouples**, labels **1 and 2** need to be connected.

To enable **32 thermocouples**, labels **1, 2 and 3** need to be connected.

IMPORTANT! The system will not work if label 3 and 1 are connected.



Figure 2: Ribbon cable connection.

3.1 USB Hub and Laptop Connections

Plug the USB connectors (labelled 1 and 2 from Figure 1) into the USB hub, as shown in Figure 3.

IMPORTANT! – Make sure that the USB hub's plug is not connected to the laptop before this step, if so unplug first.



Figure 3: USB hub connection

IMPORTANT – Ensure the USB connectors are plugged into the USB hub **first** (see Figure 3) and then continue.

IMPORTANT! - DO NOT connect the USB plug of connection 1 and 2 from **CRITICAL: Make sure the USB Hub Figure 3 is not plugged into the laptop before doing this step!**

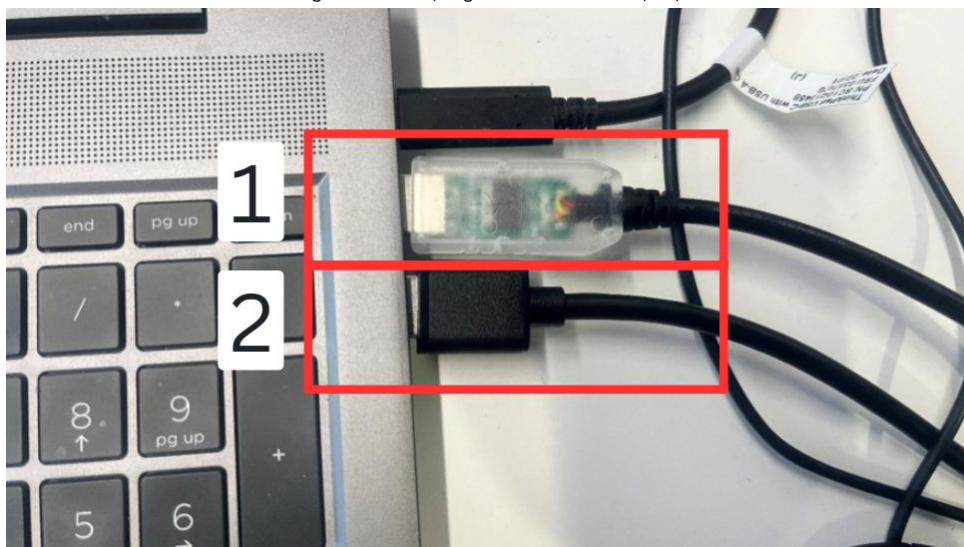
Plug the 3 connections to the 3D printed box as shown below in **Error! Not a valid bookmark self-reference.:**

Figure 1 **directly into the laptop but into to the USB hub first.**

Plug the USB hub's plug (label 2) and the DIN connector's USB plug (label 1) to your laptop as shown in

Figure 4.

Figure 4: USB plugs connected to laptop



Commented [IG5]: please note that the USB connectors must be first(!) plugged into the USB-hub, which should not be connected to the Laptop at that time, and *then* the USB-Hub can be connected to the laptop - that way the MCU board is powered correctly.

4.1 Thermocouple connection

After finishing the previous connections, the thermocouple D-Sub can now be plugged in (Figure 5).



Figure 5: Inserting the D-Sub connector in.

02 | Thonny Setup

Launch Thonny on the laptop

Check that the text in Thonny matches what is shown near label 2 in Figure 6.



Figure 6: Thonny IDE.

If the text does not match, click the text displayed near label 1 in

Figure 6.

When you do, all the COM ports (Communications Port) will appear, as shown in Figure 7.

Find and click on the line "MicroPython (generic) ST-Link VCP Ctrl @ COM11" or similar.

NOTE: The COM number (e.g., COM11) will vary between devices, so choose the line that begins with MicroPython (generic) ST-Link VCP Ctrl.

NOTE: This line will not appear if the connection labeled 2 in **CRITICAL: Make sure the USB Hub Figure 3 is not plugged into the laptop before doing this step!**

Plug the 3 connections to the 3D printed box as shown below in **Error! Not a valid bookmark self-reference.:**

Figure 1's is not plugged in.

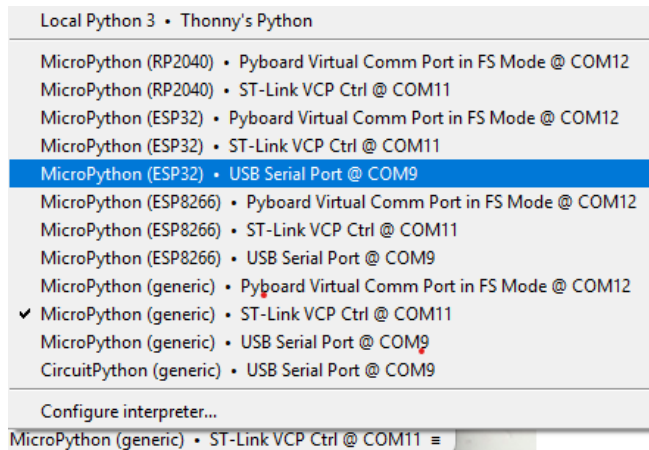


Figure 7: Thonny's COM port list.

IMPORTANT! The following steps are important.

Do not attempt to open the Python file directly from the E: drive through File Explorer.

- Click **File**, then **Open**. (Figure 8)
- The Thonny IDE should then appear as shown in Figure 9.
 - If the IDE does not look like Figure 9, check the USB connection.
- Click **MicroPython Device**, scroll down and the IDE should resemble Figure 10
- Select the `state_machine.py` file, click **OK** and the IDE should then appear as shown in Figure 11.

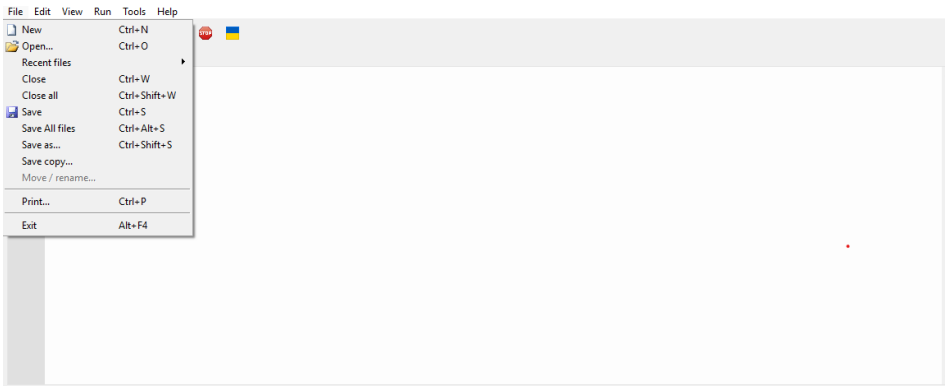


Figure 8: Clicking File.

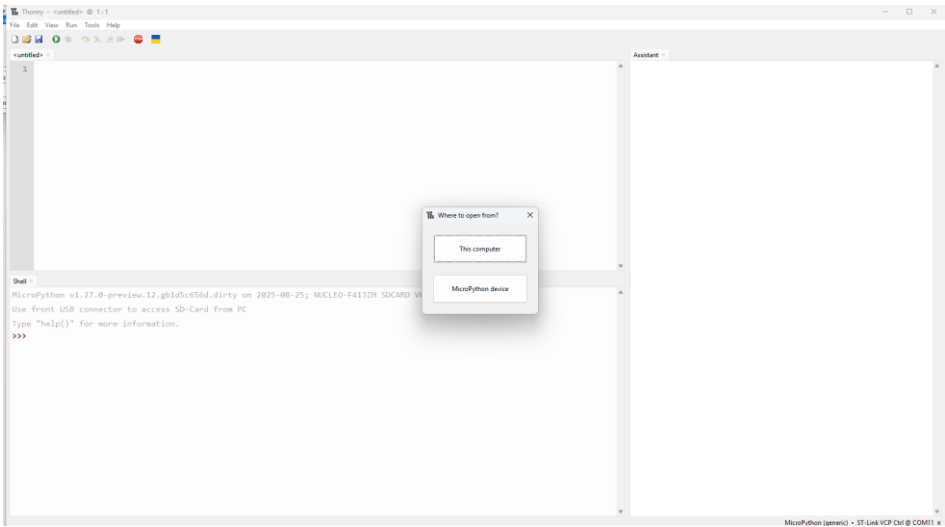


Figure 9: Clicking Open

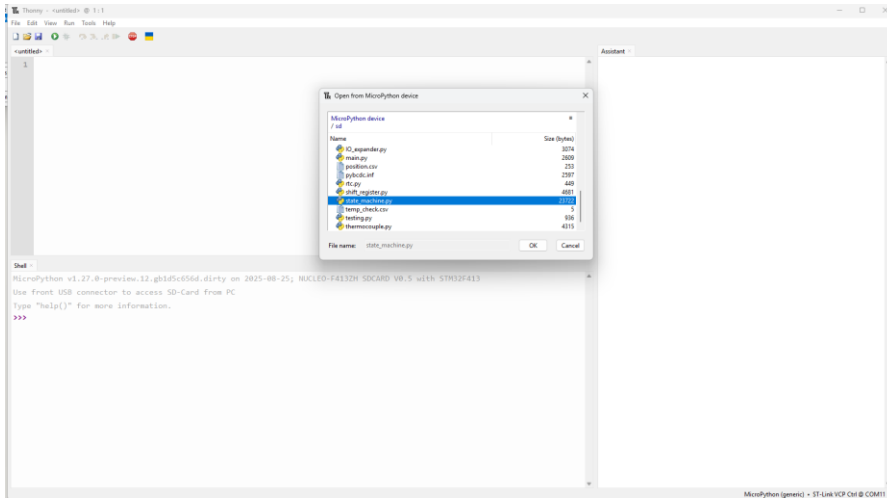


Figure 10: Choosing the state_machine.py file

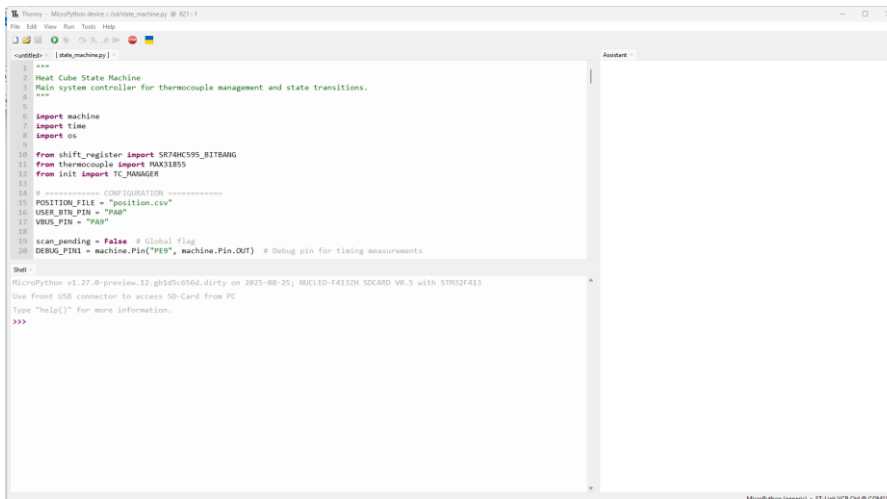


Figure 11: Clicking "OK" when selecting the state_machine.py file.

03 | VS code setup

Launch VS code:

Click **File**, and the menu shown in Figure 12 will appear. Select Open Folder.

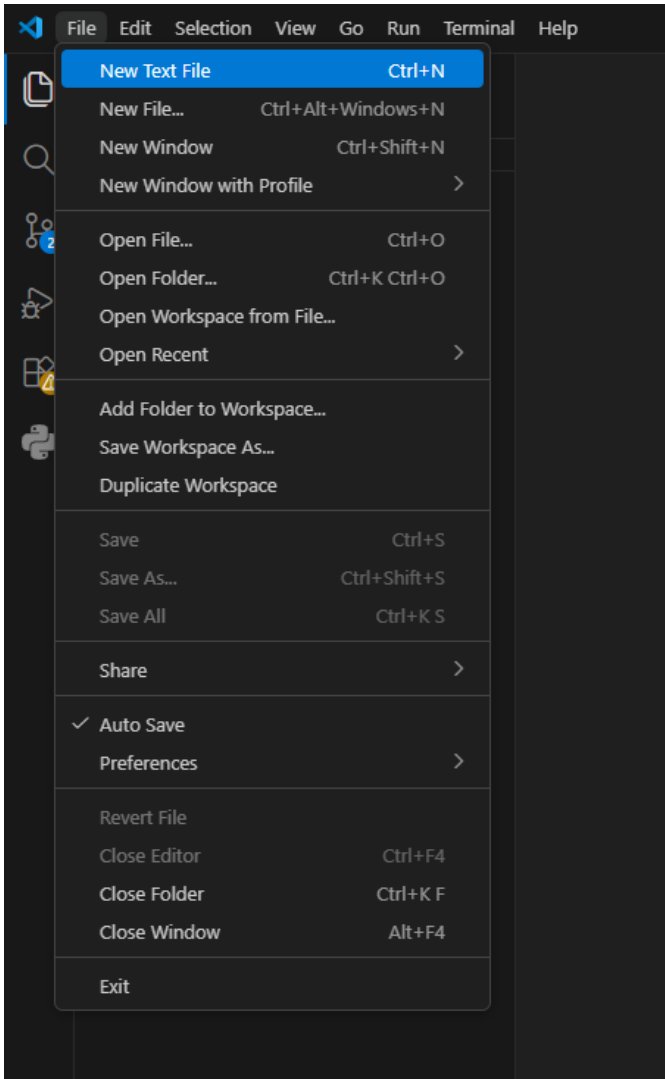


Figure 12: Clicking File on VS Code

After clicking **Open Folder**, locate and click on the **heat_cube** folder then click **Select Solder**. Figure 13 shows an example of this step.

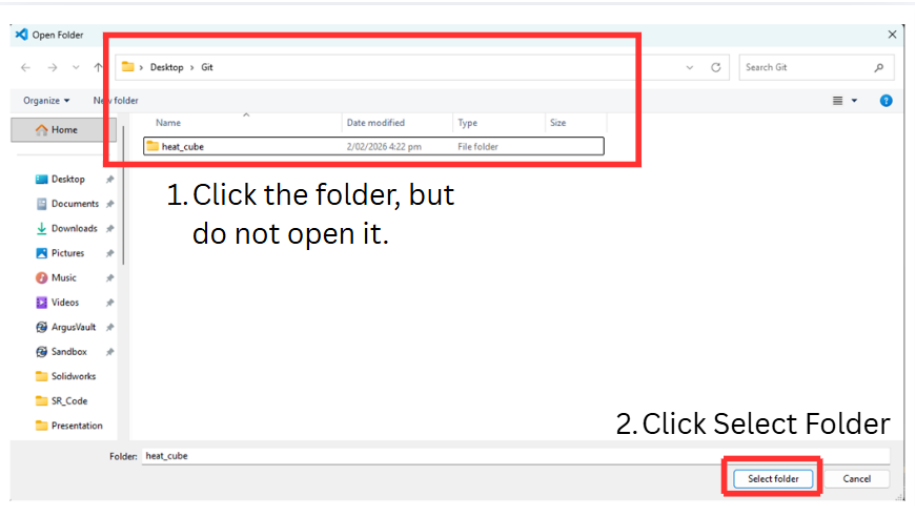


Figure 13: After clicking open folder on VS code.

After selecting the folder, the Explorer view should appear as shown in Figure 14.

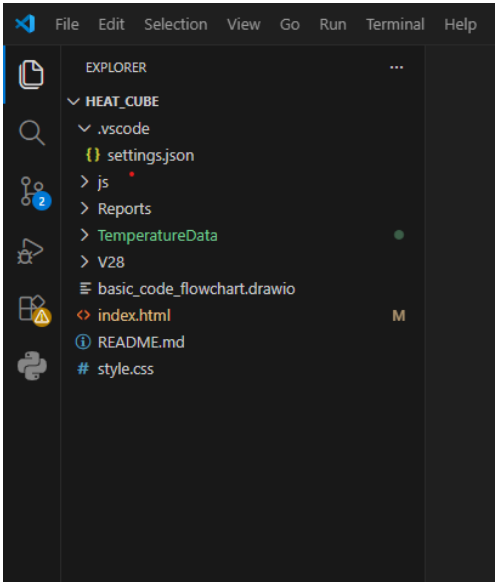


Figure 14: Files and folders within the heat_cube folder.

Right-click the file `index.html` and click **Open with Live Server**. (Figure 15).

NOTE: This step requires the Live Server extension to be installed in VS Code.

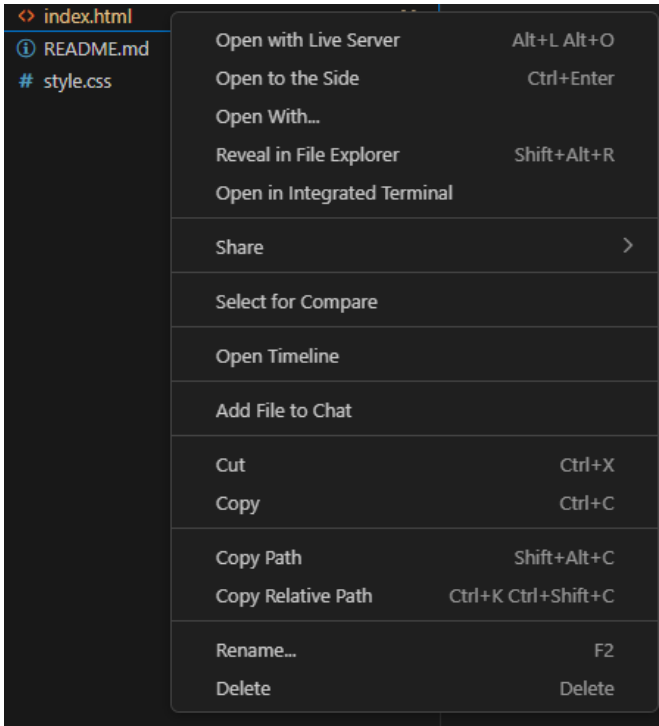


Figure 15: Right clicking index.html

The browser will open automatically. If this is your first time launching it, you may see a browser like Figure 16. Click **Enter Visualiser** to continue.

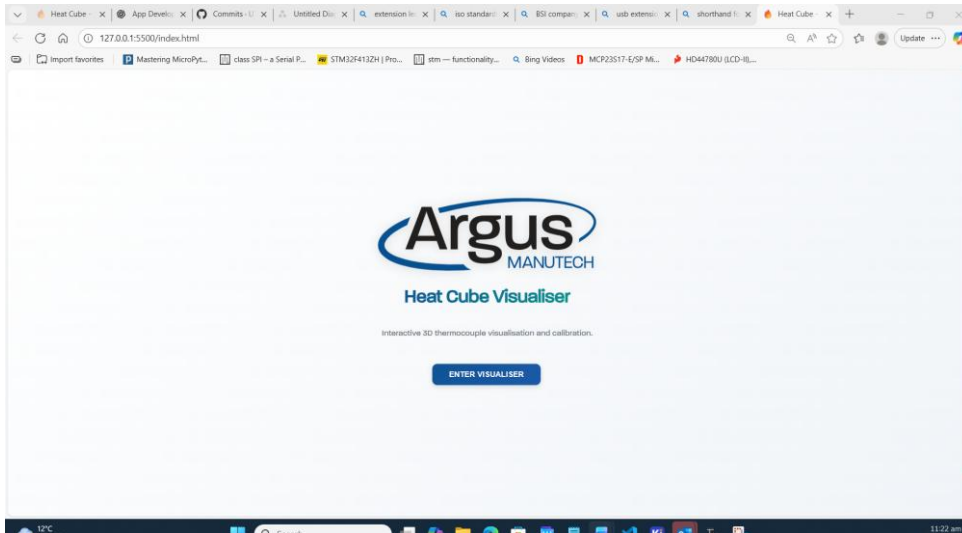


Figure 16: After right clicking Live Server.

Open the terminal. If the terminal is not visible, click the icon shown near label 1 marked with an "X" inside a circle and a warning triangle, then click **Terminal** as shown near label 2 in Figure 17.

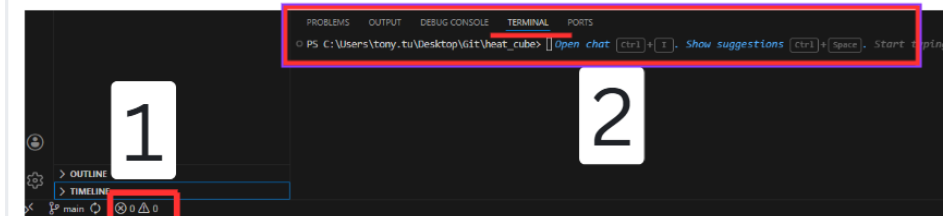


Figure 17: VS Code Terminal

In the terminal, you **must** be inside the *heat_cube/js* folder.

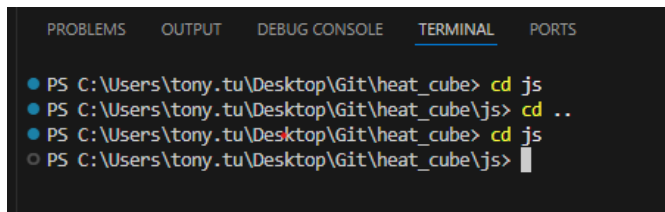
If the terminal appears as shown in Figure 18, navigate into the **js** folder.



Figure 18: VS code wrong folder.

To enter the **js** folder, type `cd js`. If you navigate into the wrong folder, type `cd ..` to move back one level.

Figure 19 illustrates this step if further clarification is needed.



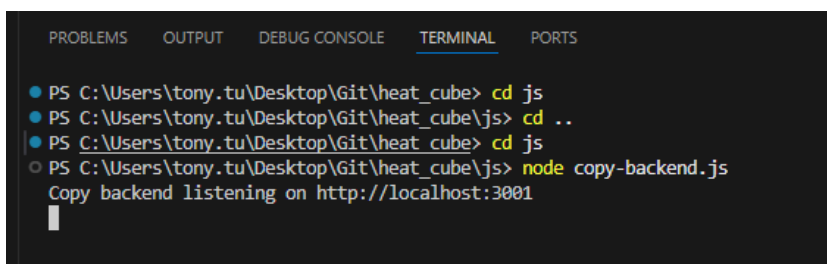
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

● PS C:\Users\tony.tu\Desktop\Git\heat_cube> cd js
● PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> cd ..
● PS C:\Users\tony.tu\Desktop\Git\heat_cube> cd js
○ PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> 
```

Figure 19: Terminal command.

When the terminal is in the correct folder, type `node c`

Once your terminal PS (Power Shell) is in the correct folder type **node copy-backend.js**. A message like the one shown in Figure 20 will appear.



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

● PS C:\Users\tony.tu\Desktop\Git\heat_cube> cd js
● PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> cd ..
● PS C:\Users\tony.tu\Desktop\Git\heat_cube> cd js
○ PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> node copy-backend.js
Copy backend listening on http://localhost:3001

```

Figure 20: Typing node command into terminal.

04 | Browser Setup

Once the Enter Visualiser button is clicked, a screen similar to Figure 21 will appear.

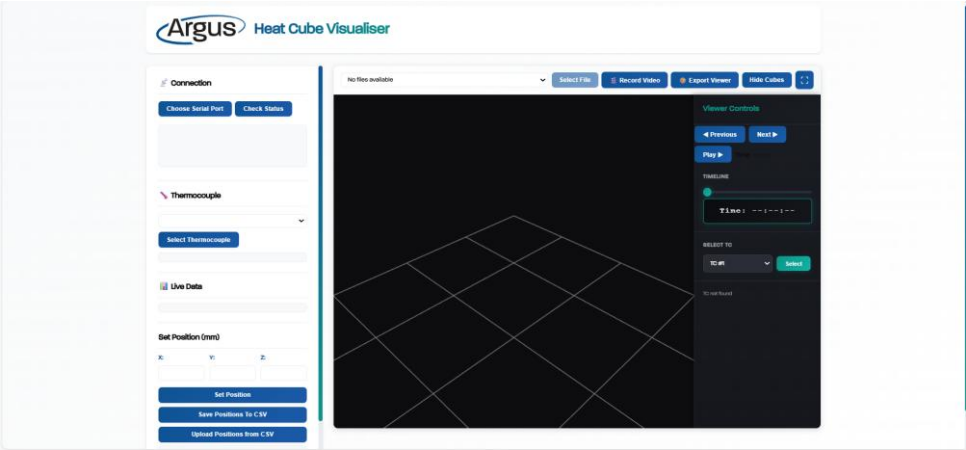


Figure 21: Initial browser screen.

In Thonny, click the green Run button to start the Python code. Text like that shown in Figure 22 will appear in the Python shell.

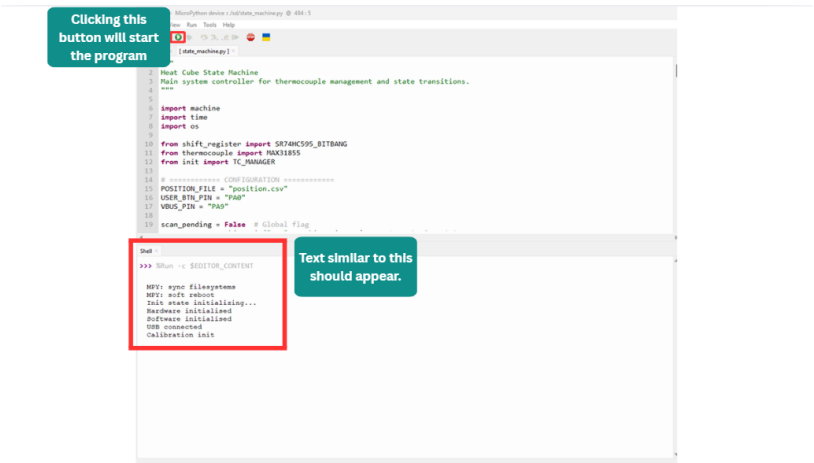


Figure 22: Clicking green play button.

For the initial setup, click the **Choose Serial Port** button. A popup like the one shown in Figure 23 will appear.

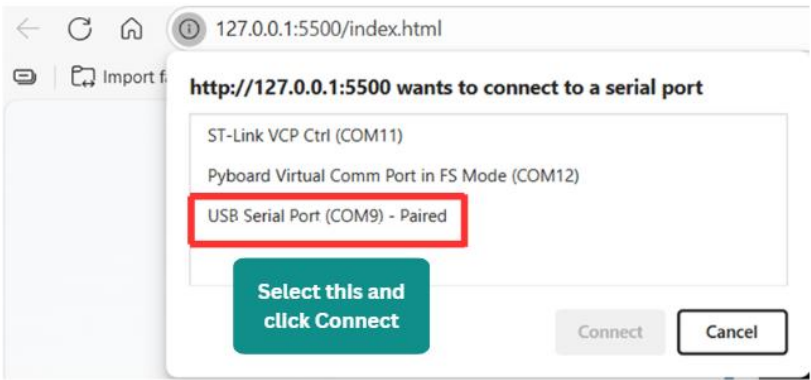


Figure 23: Serial port pop up selection.

Select the appropriate serial port, which in this example (shown in Figure 23) is **USB Serial Port (COM9)** and click **Connect**. The serial port only needs to be paired once, every other time it will automatically connect.

Figure 24 and Figure 25 show the browser and the Thonny terminal after the serial port has been connected.

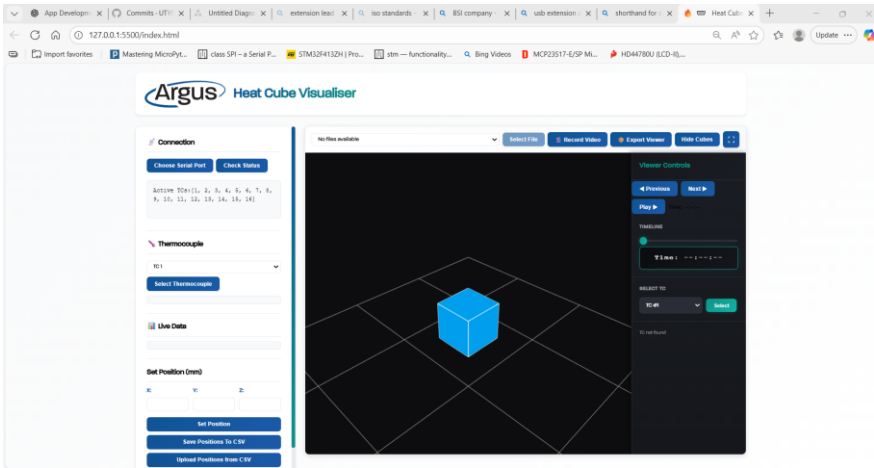
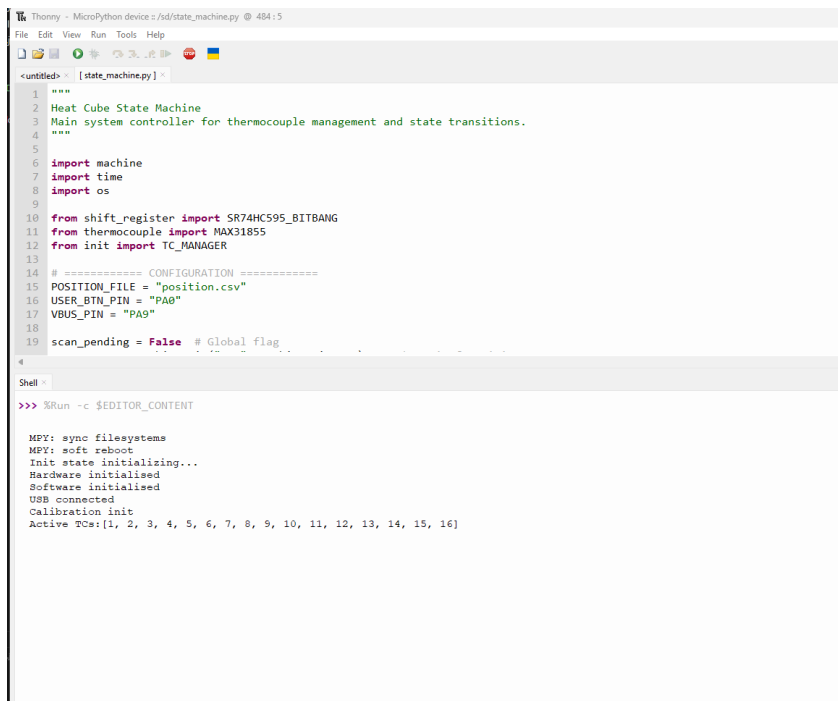


Figure 24: What browser looks like after choosing Serial Port.



The screenshot shows the Thonny IDE interface. The top pane displays a Python script named `state_machine.py`. The script includes a docstring, imports for `machine`, `time`, `os`, `SR74HC595_BITBANG`, `MAX31855`, and `TC_MANAGER`, and a configuration section with variables like `POSITION_FILE`, `USER_BTN_PIN`, `VBUS_PIN`, and a `scan_pending` flag. The bottom pane shows the shell output after running the script, indicating successful initialization of hardware and software, and listing active thermocouples.

```
1 """
2 Heat Cube State Machine
3 Main system controller for thermocouple management and state transitions.
4 """
5
6 import machine
7 import time
8 import os
9
10 from shift_register import SR74HC595_BITBANG
11 from thermocouple import MAX31855
12 from init import TC_MANAGER
13
14 # ===== CONFIGURATION =====
15 POSITION_FILE = "position.csv"
16 USER_BTN_PIN = "PA0"
17 VBUS_PIN = "PA9"
18
19 scan_pending = False # Global flag
```

```
>>> %Run -c $EDITOR_CONTENT
MPY: sync filesystems
MPY: soft reboot
Init state initializing...
Hardware initialised
Software initialised
USB connected
Calibration init
Active TCs: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]
```

Figure 25: What Thonny IDE shows after connecting to serial port.

05 | Thermocouple Selection and Position Setting

This section explains how to select thermocouples and set their positions.

To select a thermocouple, you can:

- Click the drop down shown near label 1 in Figure 26 and click **Select Thermocouple**.
- Hover and click a thermocouple cube shown in near label 7 Figure 26.
 - Note if multiple cubes are overlapping each other this cannot be done.

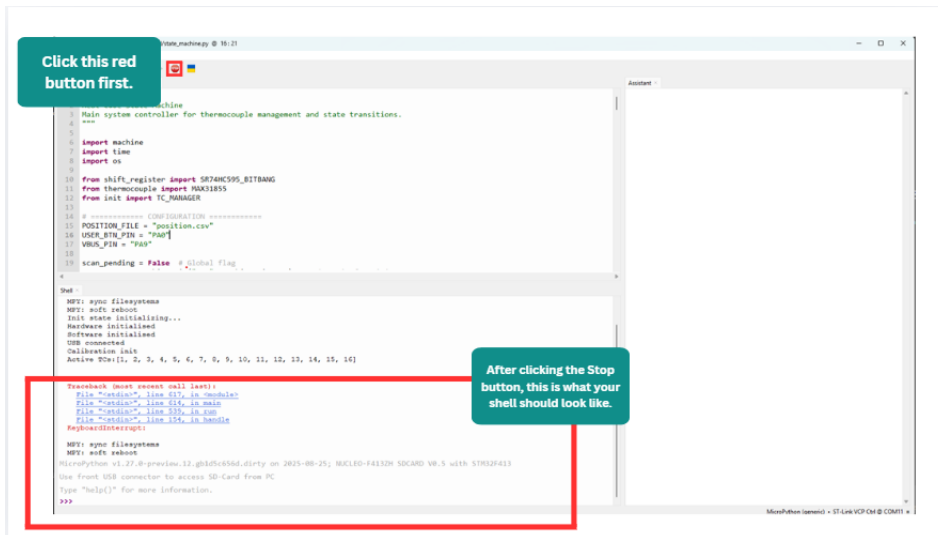
After selecting a thermocouple, the respective LED corresponding to that thermocouple should start flashing. If the selected thermocouple probe is touched, its temperature will update in the **Live Data** section (label 2), and the colour of the corresponding cube should change (label 7).

To set the positions of the thermocouple, you can:

- Type in the X, Y and Z values in the fields shown near label 3, click **Set Position** button (label 4). The current X, Y and Z positions of the selected cube will automatically load in the text fields.
- Click the **Upload Positions from CSV** button near label 6.
 - This assumes that the correct number of thermocouples is connected and that the **position.csv** file has been filled out correctly. See 5.1 for instructions.

After setting the positions of all the thermocouples, click **Save Positions To CSV** (label 5) to save the positions into the **position.csv** file.

NOTE: If you want to stop the flashing light click on the **Set Position (label 4)**.



Once the code has been stopped, locate and open the **position.csv** file on the E: Drive using File Explorer (see Figure 29).

NOTE: The E: Drive will not appear if the connected labelled 1 in Figure 1 is not connected.

The **first column** represents the **thermocouple number**, the **second column** represents the **X position**, the **third column** represents the **Y position**, and the fourth column represents the **Z position**. These columns are labelled in Figure 29 if unclear.

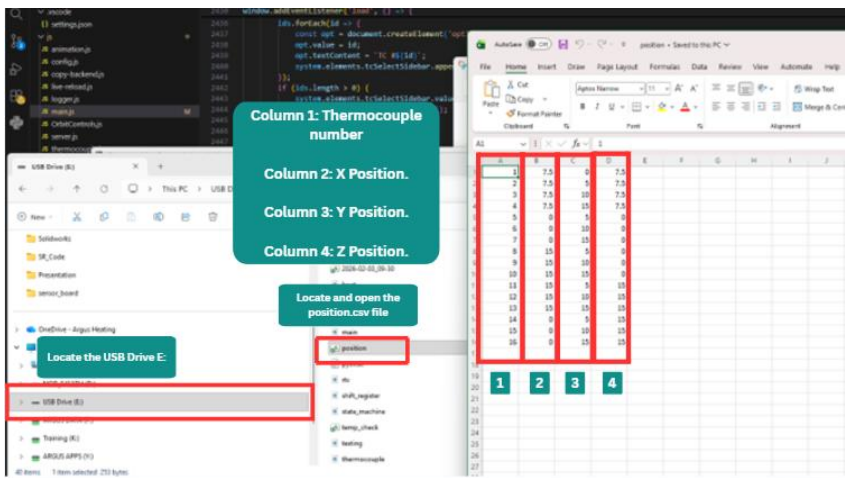


Figure 29: File format for position.csv file

After editing the position.csv file, save the file and click the Reset button on the MCU again (Figure 28).

IMPORTANT: Once pressed, the Thonny IDE should show messages as seen in Figure 25.

Click the **Upload Positions from CSV** button, and the cubes will update to the positions specified in the **positions.csv** file. An example of a configuration is shown in Figure 30.

NOTE: The white grid and axes may appear misaligned, as they are set using fixed values in the code.

If scene is frozen then refresh the page.

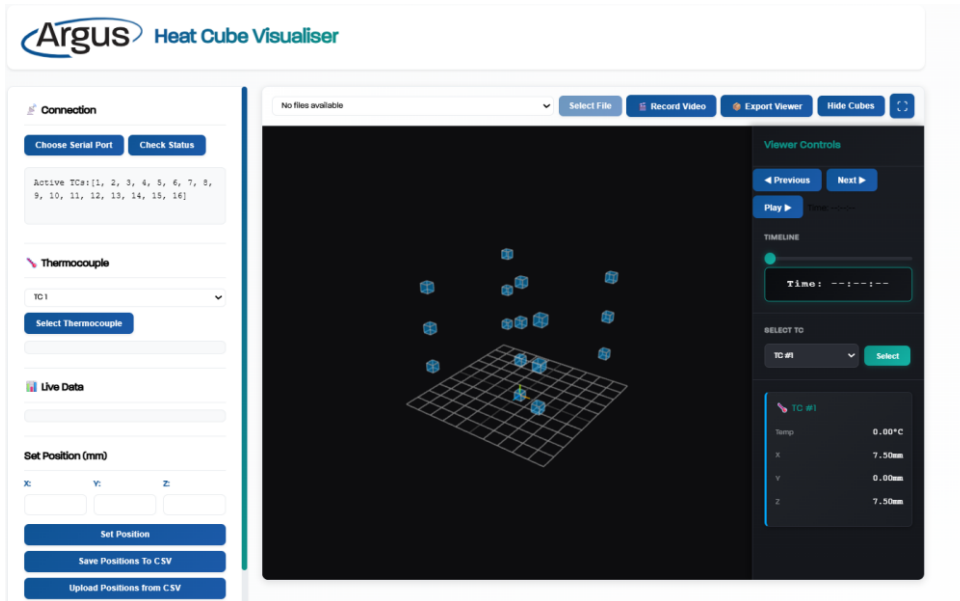


Figure 30: After clicking Uploading Positions from CSV button.

06 | Thermocouple Data Measurement

To start measuring data from the thermocouples, click the **Finish Calibration** button, as shown in Figure 31.

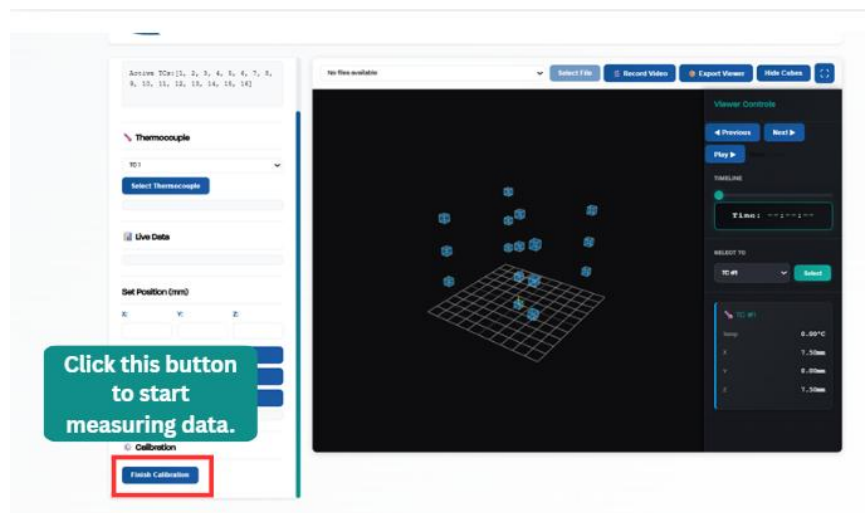


Figure 31: Button to press to start measuring.

When the Finish Calibration button is clicked, the LEDs on the PCB will flash.

The cubes on the screen will appear red if no thermocouples are connected, as shown in Figure 32.

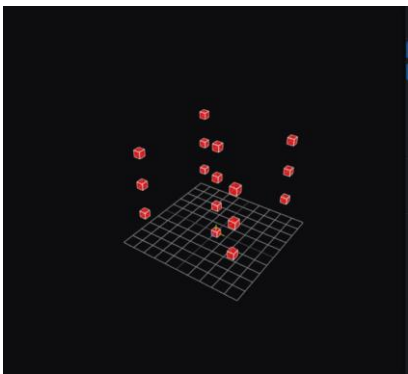


Figure 32: Red cubes which represents that no thermocouple is connected.

To stop measuring, click the same button used to start the measurement (see Figure 33).

The button will typically display **Enter Calibration Mode**, but in some cases, it may still show **Finish Calibration**. Regardless of the label, clicking the button will stop the measurement process if measurement is currently active. If the LEDs do not flash, measurement has stopped successfully.

NOTE: Do not click this button repeatedly rapidly; wait approximately 2 seconds for changes to take effect.

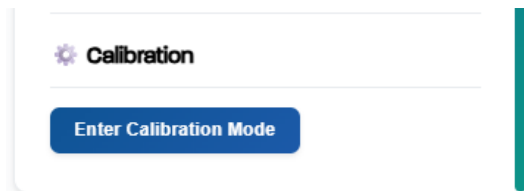


Figure 33: Enter Calibration Mode Button.

To confirm whether the system is in calibration mode or measurement mode, check the last message in the Thonny terminal (see Figure 34).

If the last message is '**Calibration init**', the system is in calibration mode.

If the last message is '**Measure init**', the system is in measurement mode.

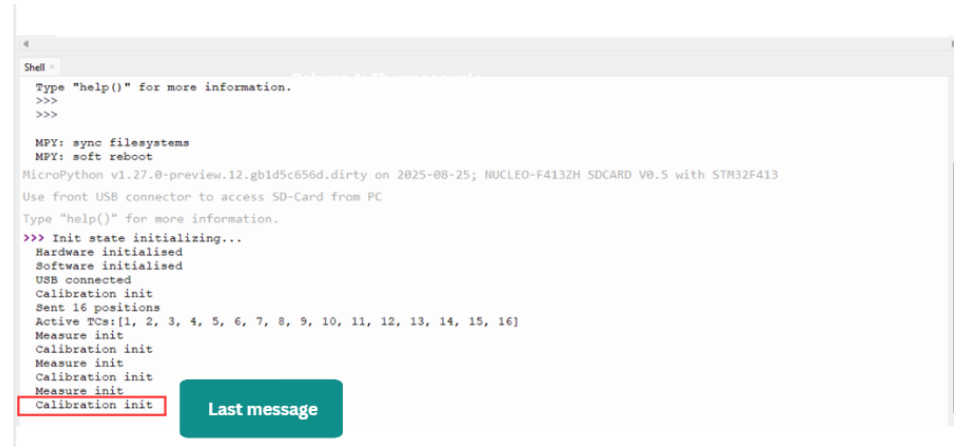


Figure 34: Thonny shell upon entering measurement mode and calibration mode.

07 | Reviewing data from previous files.

After thermocouple data has been measured and a file needs to be reviewed, follow the steps in this section.

Steps to check:

- Ensure the VS code terminal appears as shown in Figure 20.
- Ensure the Python shell appears as shown in Figure 34.

In the browser click the **Check Status** button, as shown in Figure 35.

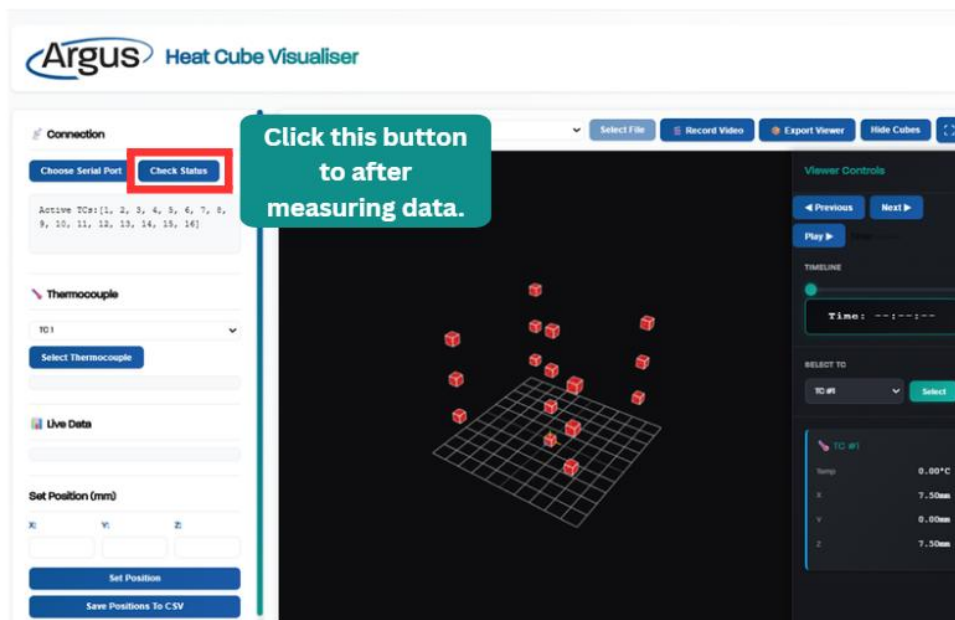


Figure 35: Check Status button.

Wait until the PowerShell (PS) terminal in VS Code displays text like that shown in Figure 36.

NOTE: Check the terminal messages to confirm that the recently recorded file has been copied over correctly.

NOTE: Wait until the last line displays **Backup complete**.

```

PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> node copy-backend.js
Copy backend listening on http://localhost:3001
Backing up files from E:\ to C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData
Copied E:\2026-01-14.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-14.csv
Copied E:\2026-01-16.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-16.csv
Copied E:\2026-01-19.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-19.csv
Copied E:\2026-01-20.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-20.csv
Copied E:\2026-01-21.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-21.csv
Copied E:\2026-01-26.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-26.csv
Copied E:\2026-01-27.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-27.csv
Copied E:\2026-01-28.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28.csv
Copied E:\2026-01-27.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-27.csv
Copied E:\2026-01-28_10-45.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_10-45.csv
Copied E:\2026-01-28_11-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_11-00.csv
Copied E:\2026-01-28_12-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_12-00.csv
Copied E:\2026-01-28_12-15.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_12-15.csv
Copied E:\2026-01-28_12-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_12-30.csv
Copied E:\2026-01-28_12-45.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_12-45.csv
Copied E:\2026-01-28_13-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_13-00.csv
Copied E:\2026-01-28_14-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_14-00.csv
Copied E:\2026-01-28_14-15.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_14-15.csv
Copied E:\2026-01-28_15-45.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_15-45.csv
Copied E:\2026-01-28_16-15.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_16-15.csv
Copied E:\2026-01-28_16-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-28_16-30.csv
Copied E:\2026-01-29_09-45.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_09-45.csv
Copied E:\2026-01-29_10-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_10-00.csv
Copied E:\2026-01-29_10-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_10-30.csv
Copied E:\2026-01-29_15-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_15-30.csv
Copied E:\2026-01-29_16-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_16-00.csv
Copied E:\2026-01-30_08-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-30_08-30.csv
Copied E:\2026-02-03_09-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-02-03_09-30.csv
Copied E:\2026-02-03_12-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-02-03_12-00.csv
Copied E:\2026-02-03_12-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-02-03_12-30.csv
Backup complete.

```

Figure 36: Power shell after launching node and clicking "Check Status".

Click the dropdown labelled "Select a file..." and select a file to view. Then click **Select File**. If the file that was just recorded does not appear, refresh the page.

Further explanation is shown in Figure 37.

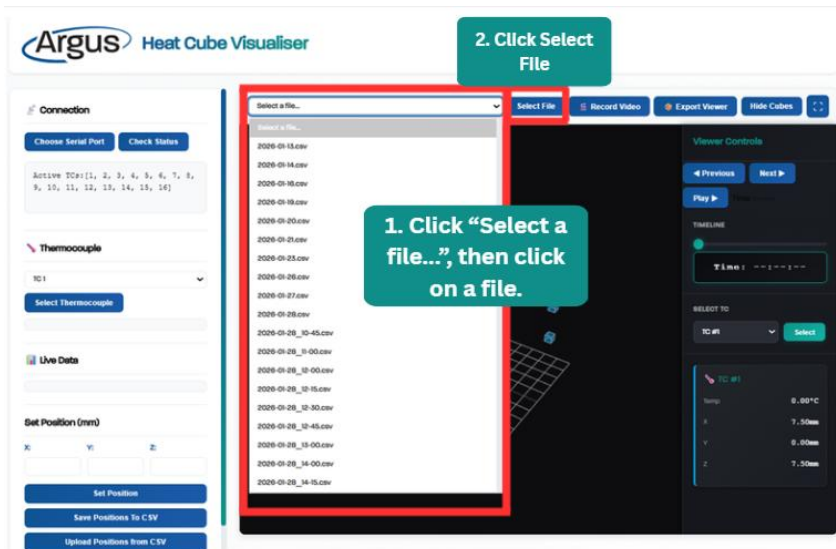


Figure 37: After clicking "Select a file..."

After this, the textbox will display information like that shown in Figure 38.

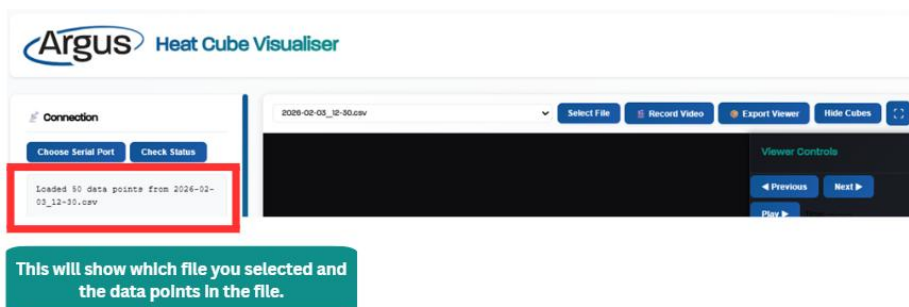


Figure 38: After clicking Select File.

08 | Features of viewing a file

This section covers the available features after a file has been selected for review.

Figure 39 shows the features available when reviewing a file.

The Feature Buttons section includes the following features:

- **Record Video** button – Click to record a video of the scene.
 - o This recording will capture any scene movement, including rotation, panning or zooming.
- **Export Viewer** button – Click to generate a standalone HTML file containing the same data points as the selected file.
 - o The HTML file is downloaded to the **Downloads** folder.

NOTE: The **Record Video** feature is not recommended for large files, as recording may take a significant amount of time. The **Export Viewer** feature is recommended instead.

The Viewer Controls section includes the following features:

- **Previous, Next and Play** buttons:
 - o **Previous** – Displays the previous data.
 - o **Next** – Displays the next data.
 - o **Play** – Plays through all the data, starting at the time displayed in the **Time Box**.
- When **Play** is clicked, the button text will change to **Pause**.
 - o **Pause** – Stops the playback of data.
- **Timeline Scroller** – Scroll the timeline forward or backward to view the data associated with the time displayed in the **Time Box**.
- **Select TC Section:**
 - o NOTE: This control is used for viewing the data of a specific thermocouple from a file.
 - o Selecting a thermocouple in this section does not allow configuration of its position.
- **Hide Cubes** button – Hides all cubes in the scene.

- **Full screen icon** – Displays the scene in full-screen mode.

NOTE: Drag the Timeline scroller back if the scene isn't changing when clicking the **Play** button.

NOTE: Remember to click **Pause** if you want to stop the playback.

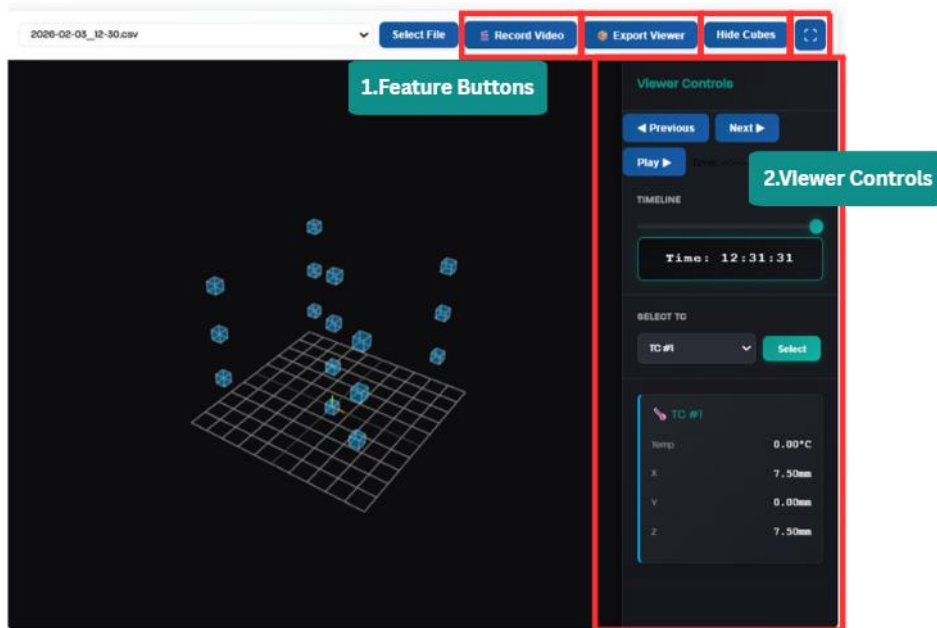


Figure 39: Features on the browser

Figure 40 illustrates what occurs when the **Export Viewer** button is clicked.

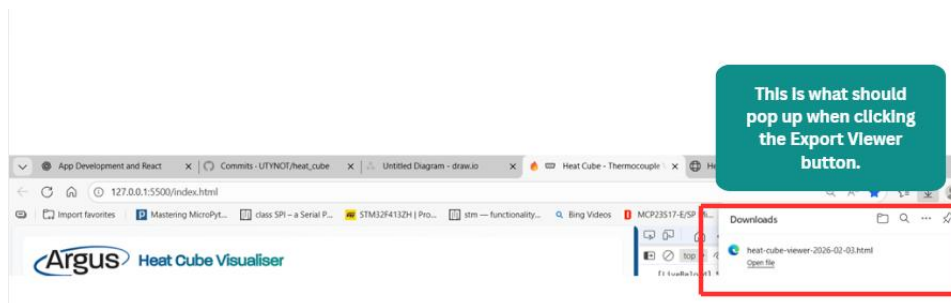


Figure 40: Clicking export viewer.

After opening the exported HTML file, a black scene may appear initially, as shown in Figure 41.

Zoom out to view the scene.

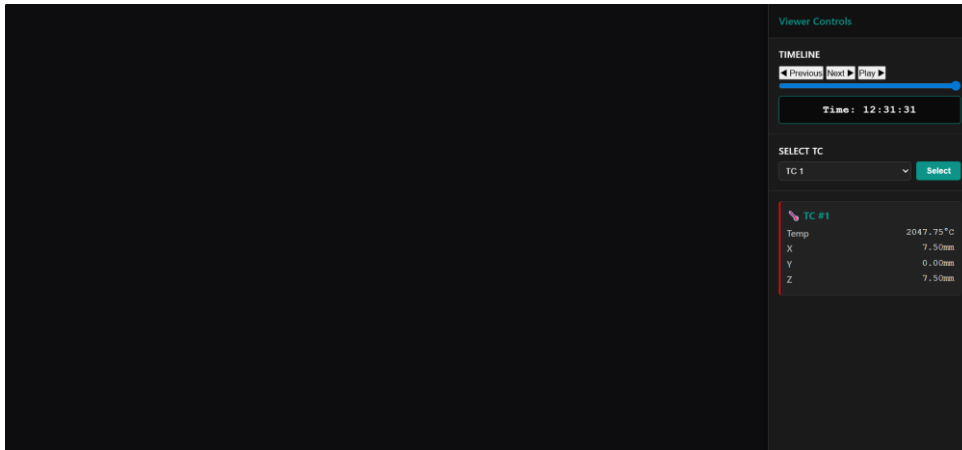


Figure 41: Clicking the downloaded html link.

After zooming out, the scene will appear similar to Figure 42. This exported HTML file also includes additional viewing features. These features are the same as those in the **Viewer Controls** section on the original page.

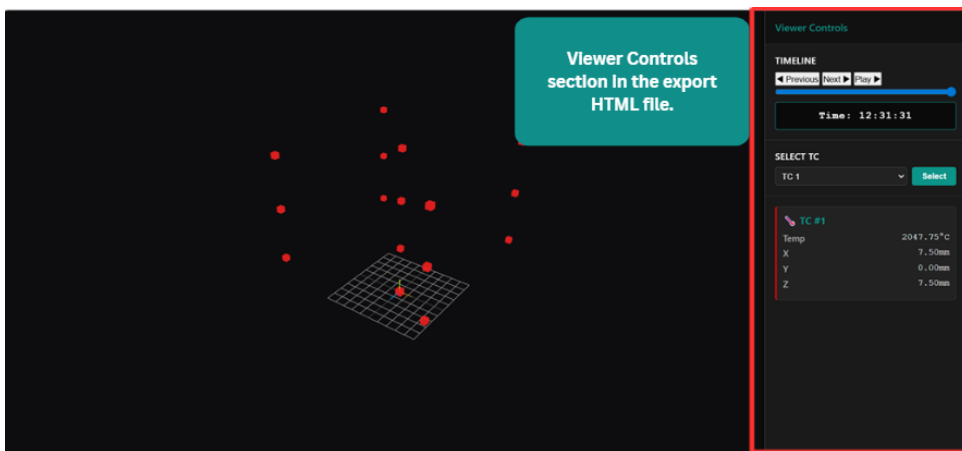


Figure 42: Zooming out in downloaded HTML file.

09 | Debugging

This section covers ways to debug when the system is faulty.

6.1 Solution to most issues

Remove a blank line or make any small change in the **HTML**, **CSS**, or **JavaScript** files, then press **Ctrl + Z** to revert the change. This will refresh and reload the page.

Refresh the page and/or click the **Check Status** button. (See Figure 43).

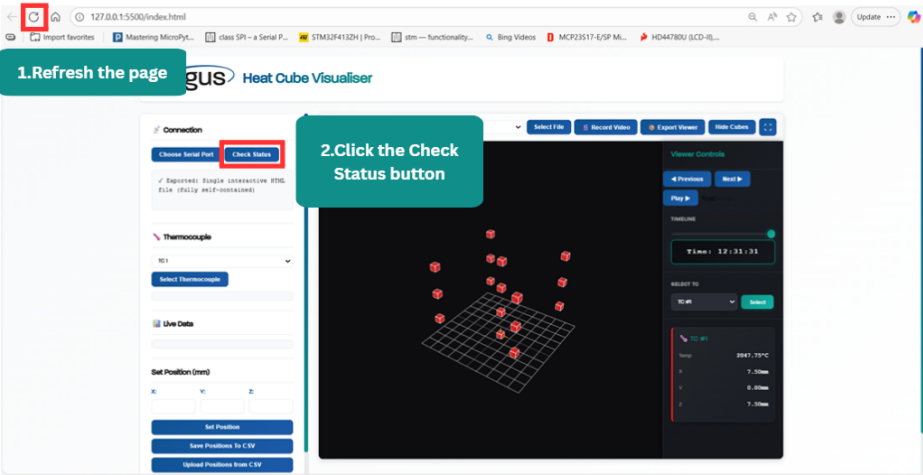


Figure 43: Refresh and Check Status.

Restart the Thonny code by clicking the red **Stop** button, then click the green **Run** button to start it again.

Check the PowerShell terminal in VS Code to see if it has stopped. If it has stopped, activate the terminal input preparing to type. Then press **Ctrl + C** to clear it and then type **node copy-backend.js**. (See Figure 44 for an example).

```
Copied E:\2026-01-27.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-27.csv
Copied E:\2026-01-29_09-45.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_09-45.csv
Copied E:\2026-01-29_10-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_10-00.csv
Copied E:\2026-01-29_10-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_10-30.csv
Copied E:\2026-01-29_15-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_15-30.csv
Copied E:\2026-01-29_16-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-29_16-00.csv
Copied E:\2026-01-30_08-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-01-30_08-30.csv
Copied E:\2026-02-03_09-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-02-03_09-30.csv
Copied E:\2026-02-03_12-00.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-02-03_12-00.csv
Copied E:\2026-02-03_12-30.csv -> C:\Users\tony.tu\Desktop\Git\heat_cube\TemperatureData\2026-02-03_12-30.csv
Backup complete.
PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> node copy-backend.js
Copy backend listening on http://localhost:3001
PS C:\Users\tony.tu\Desktop\Git\heat_cube\js> node copy-backend.js
Copy backend listening on http://localhost:3001
```

Ctrl + C was pressed between these two lines.

Figure 44: Debug JavaScript terminal.

If the terminal needs to be viewed, right-click on an empty region in the browser and then click **Inspect**. (See Figure 45).

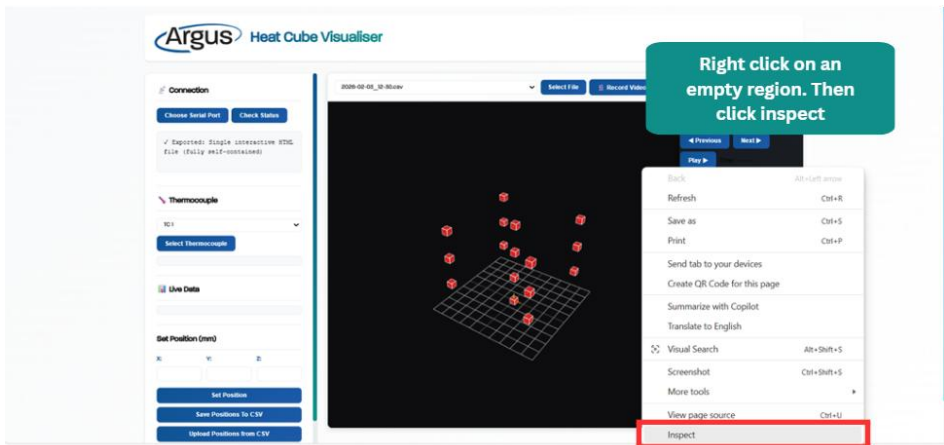


Figure 45: Where to find Inspect.

Click the **Console** icon as shown in Figure 46. This displays the console log, where debug messages can be viewed.

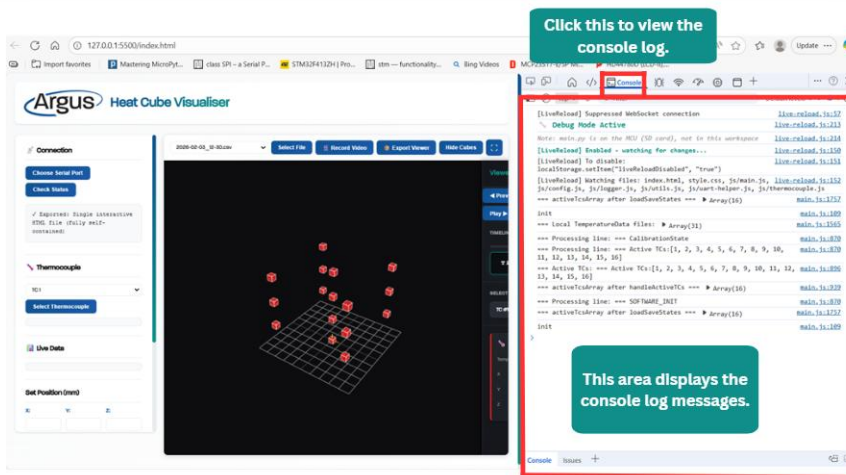


Figure 46: Console log messages in the browser.

7.1 Full reset of local storage on browser

To fully reset the browser and clear the local storage, type the command `localStorage.clear()`; at the end of `main.js` (see Figure 47).

In the example shown in Figure 47, this command would be typed on line 2448. After the browser refreshes, delete the line.

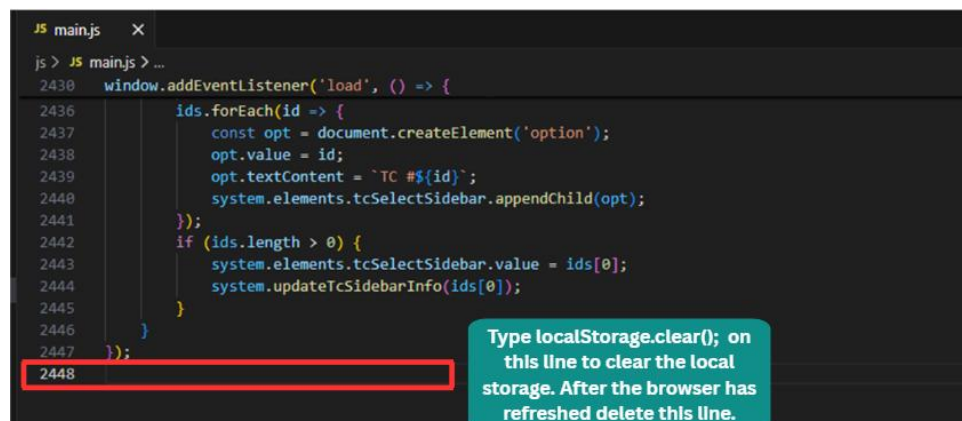


Figure 47: Where to type to clear and reset local storage.

10 | Common Issues

Below are several common issues that may arise, along with their possible solutions.

Issue: Cannot view thermocouple data, or buttons are unresponsive

Solutions:

Verify that the Python code is running and you are hosting the server on VS code. To verify this click **Check Status** the Python shell should restart and display text like Figure 25.

Refer to Section 6.1 for additional troubleshooting steps.

Issue: Files are corrupted.

Solution:

Safety delete all files from the SD card, then copy the latest version of **SR_Code** back onto the SD card.

- The folder containing this code is shown in Figure 48.

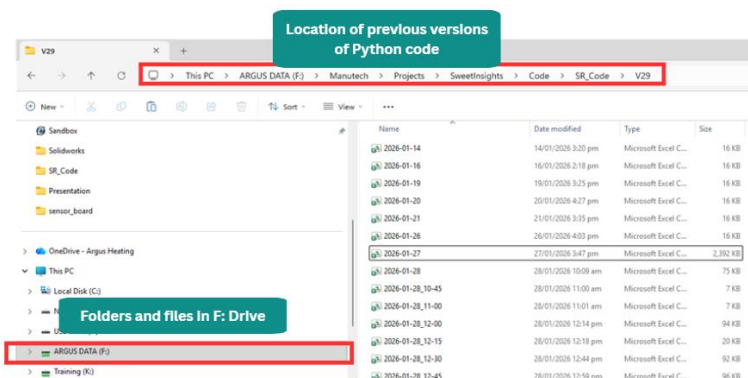


Figure 48: SR Code Location.

Issue: Just cannot get things to work.

Solution: Check all connections, verify the code is running and inspect the wiring to ensure nothing has been unplugged. If further help is needed, ask.

Issue: A newly recorded data file does not appear in the dropdown list

Solution: Confirm that **node copy-backend.js** is typed and the server is running in VS Code's

terminal. If so, check the terminal messages to see that the file was copied over. Refresh the page afterward.

Issue: The 3D Scene is frozen

Solution: Refresh the page to reload

Issue: All the cubes on the browser scene are red.

Solution: No thermocouples are connected.

Issue: The scene is black with no visible cubes.

Solution: The camera view is likely positioned either too close to or too far from the cubes. Zoom in or pan the scene to relocate the cubes. If the issues persist, preform a full browser reset by following section 7.1.

Issue: JavaScript code was modified accidentally as cannot be restored manually.

Solution: Revert to a previous working version in Git, which will be labelled something like **"Working with SR Code V29"**. Then copy the commit ID from the highlighted section shown in Figure 49.

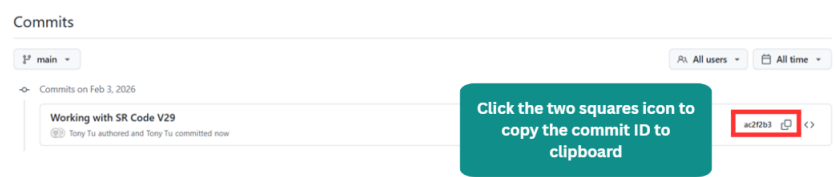


Figure 49: Previous commits

Then right-click the project folder on your laptop and click **Open in Terminal**. (See Figure 50)

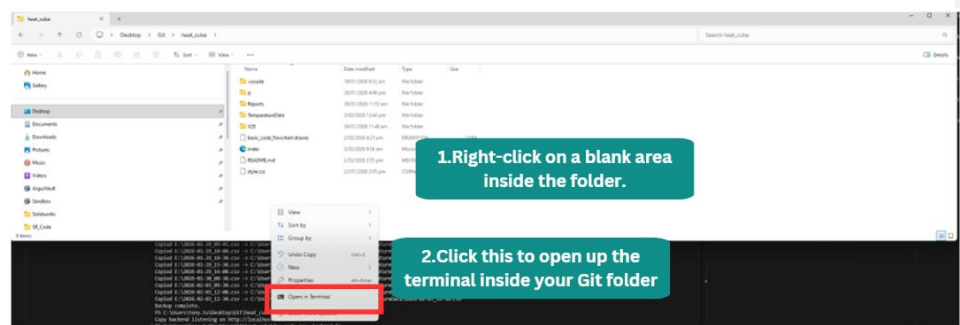


Figure 50; Right clicking inside the folder to open the terminal.

Type the command `git reset --hard <commit_ID>` using the commit ID you want to revert to (e.g., `git reset --hard ac2f2b345d15403049c703a3175b1caf4e8c9e55`) into the terminal.

Then type `git status` to confirm that the repository has returned to the desired version.

This step is shown in Figure 51. The terminal output will appear similar to the example displayed.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PowerShellLatest

PS C:\Users\tony.tu\Desktop\Git\heat_cube> git reset --hard ac2f2b345d15403049c703a3175b1caf4e8c9e55
HEAD is now at ac2f2b3 Working with SR Code V29
PS C:\Users\tony.tu\Desktop\Git\heat_cube> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\tony.tu\Desktop\Git\heat_cube> |
```

1. Typing `git reset ...` should show this message

2. Typing `git status` should show these messages

Figure 51: Git commands.

11 | Requirements

Thonny IDE for the MCU code – This has python installed on it.

VS Code IDE (optional) for working with JavaScript code

Required extensions for VS Code:

- Live Server
- GitHub Copilot Chat (optional, if AI assistance is desired)

These extensions are highlighted in Figure 52.

Node JS – For hosting a server to copy files over

Git (Optional) – Version control.

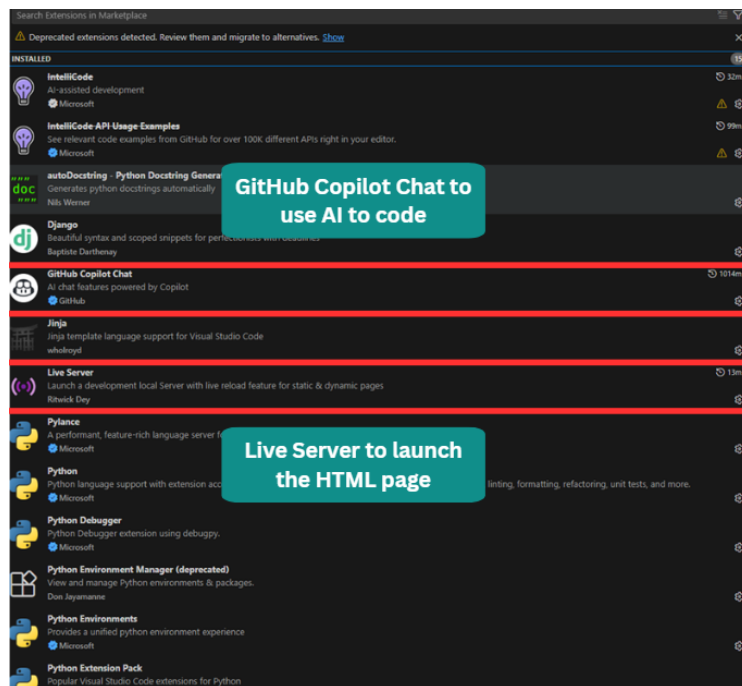


Figure 52: Extensions for VS Code.

12 | How to debug and add new features easily: AI

Using AI to debug code and add new features can be helpful, but it is a double-edged sword.

You need to be more specific about what you want AI to do: an example of this is like:

"I want you to add a feature of a timeline where the user is able to first select a file to view and then is able to scroll a timeline and view the datapoints stored in that file with the first column representing the time that data was recorded and equal column after that represents the temperature data for that cube so second column is the temperature data for the first cube. Then alongside this the colour of that cube will change depending on the designated temperature and be updated on the scene..."

Instead of just:

"Add a timeline scroller thing"

Since AI requires context to understand the intended task, it will otherwise attempt to do what it **thinks** you want.

If an issue is encountered, it is often better to first consider how you would approach the solution yourself, then explain to Copilot the steps needed to identify and fix the problem.

Otherwise, the AI may make the issue worse, though not always.