CS 350 Final Project Smart Thermostat

This project was focused on building and testing a prototype smart thermostat for SysTec. The goal was to create a working demo that reads temperature data, displays it on an LCD, and simulates communication with a cloud server. The system was built using a Raspberry Pi 4B, which connects to an AHT20 temperature and humidity sensor over I2C, controls LEDs and buttons through GPIO, displays information on an LCD, and sends simulated thermostat data over UART. Together, these components show the core functions that would later be expanded into a Wi-Fi connected thermostat.

Peripheral Support

The Raspberry Pi 4B includes all the hardware interfaces needed for this project:

* I2C: Used to connect to the AHT20 sensor for reading temperature and humidity data.
* GPIO: Controls three buttons (for adjusting settings and states) and two LEDs (for heating and cooling status).
* UART: Simulates sending thermostat data to SysTec’s cloud system.
* LCD (via I2C): Displays system status, temperature readings, and time/date information.

These peripherals were handled through Python using gpiozero, adafruit-circuitpython-ahtx0, and adafruit-circuitpython-charlcd. Each device worked as expected, showing that the Raspberry Pi can handle multiple concurrent I/O operations reliably.

For the final product, the thermostat will need to connect to the SysTec cloud using Wi-Fi. The Raspberry Pi already has built-in Wi-Fi, which makes it ideal for testing cloud features early on using HTTP or MQTT communication in Python.

For actual production hardware, Microchip and NXP microcontrollers are better suited because they use less power, have dedicated Wi-Fi modules, and run embedded code more efficiently.

* Raspberry Pi: Great for prototyping; full Linux OS and easy connectivity.
* Microchip (SAMW25): Integrated Wi-Fi, lower power use, but limited memory.
* NXP (i.MX RT): Fast, reliable, and more industrial-grade, though more complex to program.

Overall, the Raspberry Pi is best for early development and testing, while Microchip or NXP devices are better for long-term, low-power IoT deployment.

Architecture Capability Comparison

| Device | Flash / RAM | Wi-Fi | Peripheral Support | Summary |
| --- | --- | --- | --- | --- |
| Raspberry Pi 4B | microSD (8–32 GB) / 2–8 GB | Built-in | I2C, GPIO, UART, LCD | Excellent for development and testing |
| Microchip SAMW25 | 256 KB / 32 KB | Built-in | I2C, UART, GPIO | Ideal for low-power IoT |
| NXP i.MX RT1060 | 4 MB / 1 MB | External module | I2C, UART, GPIO | High performance and reliable for production |

All three platforms can support the required peripherals, but the Raspberry Pi’s flexibility and full OS make it the easiest to build and test on. When moving toward a commercial product, the NXP or Microchip platforms would offer stronger embedded performance and lower power consumption.

The prototype successfully showed all key thermostat features, reading and displaying temperature, reacting to user input, controlling indicators, and sending simulated data. The Raspberry Pi’s built-in interfaces made development simple and effective. For the next step, porting this design to a Microchip or NXP microcontroller would make it more efficient, compact, and ready for Wi-Fi-enabled production use.

References:  
Adafruit Industries. (n.d.). *CircuitPython Libraries for AHT20 and LCD Displays*. <https://learn.adafruit.com/>