**a) Describe the process you used to develop the custom code for the given task.**

I began with an examination of Circuit #7 instructions from USK Guide Manual together with LiquidCrystal library information from online resources. The examination revealed information about how to set up sensors and operate LCD modules. Following the manual I constructed the circuit while making sure each connection (the sensor with LCD wires and potentiometer) was properly established. A complete verification of every connection happened before powering the application.

I checked the temperature sensor functionality by viewing its output through the serial monitor of the Arduino system. The sensor temperature changed when I breathed for observation through the monitor readings. I proceeded to link the LCD display. The LCD received its connections through pins 12, 11, 5, 4, 3, 2 on the Arduino Uno while the contrast adjustment component used a potentiometer at Pin 3. The Arduino.cc platform provided basic code examples for me to confirm static text functionality of the LCD device.

I added the functioning sensor platform with the operating LCD display after confirming their individual performance. The program obtains sensor measurements to compute temperatures which are shown through serial output and an LCD screen display simultaneously. The device underwent different tests that validated sensor readings with room temperature and confirmed the LCD screen displayed correct text and matched the readings from other temperature measurement devices. The integrated system worked correctly after my verification tests.

**b) Provide the complete (final) software code with sufficient comments.**

#include <LiquidCrystal.h>

// Initialize LCD with (RS, E, D4, D5, D6, D7)

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int potPin = A0; // Potentiometer connected to A0

float potValue = 0.0; // Store potentiometer reading

float tempValue = 0.0; // Simulated temperature

void setup() {

lcd.begin(16, 2); // Initialize LCD (16x2)

lcd.print("Temp (deg. F):"); // Static text

Serial.begin(9600); // Start Serial Monitor

delay(1000); // Allow LCD to initialize

}

void loop() {

potValue = analogRead(potPin); // Read potentiometer value (0 - 1023)

float voltage = (potValue / 1023.0) \* 5.0; // Convert reading to voltage (0V - 5V)

// Map voltage to temperature range (Simulating 32°F to 100°F)

tempValue = map(potValue, 0, 1023, 32, 100);

Serial.print("Potentiometer Value: ");

Serial.print(potValue);

Serial.print(" | Voltage: ");

Serial.print(voltage);

Serial.print("V | Simulated Temp: ");

Serial.print(tempValue);

Serial.println(" F");

lcd.setCursor(0, 1); // Move cursor to second row

lcd.print(" "); // Erase old value (overwrite)

lcd.setCursor(0, 1); // Move cursor again

lcd.print(tempValue); // Display updated temperature

delay(1000); // Update every second

}

**c) Provide snapshots confirming successful download of the code.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**d) Provide pictures with labels showing your setup and properly working display.**

**A circuit board with wires

AI-generated content may be incorrect.**

**e) Develop a strategy to verify proper functionality of the system.**

* Verify ambient temperature readings by using either known measurement devices like “Data Monitor” smartphone apps or direct access to the lab's temperature sensor.
* Ensure that the serial monitor presents temperature data near the surrounding room temperature measurement while the sensor maintains its rest position.
* Blow the sensor while observing the monitoring devices to verify that the readings are enhanced equally between the serial monitor and LCD screen.
* The LCD device updates its data every half-second while showing correct readings with no issues or display artifacts.

**f) Provide details of how you have verified the functionality of the system.**

The sensor measurement results were tested against the current environment temperatures measured within the room. The sensor underwent multiple tests which demonstrated its ability to register elevated readings during manual heating operations through breathing or contact actions. The code executed a reset operation on the LCD display before each new measurement to eliminate remaining text on the screen. A serial monitor output created another verification method to cross-check the system. Different tests in changing conditions with human contact showed that the sensor functioned correctly while the LCD screen updated in real time.