Good afternoon dear examiner. Today, I would like to introduce our temperature control system. The main functionality of our system is to apply highly controlled temperature change on a small surface. Ranging from 5 to 45 Celsius degree. During practical experiment, our system can reach its maximum temperature within 40 seconds from room temperature, and cool down from that to 5 degrees within 2 mins. The accuracy of the temperature controller is within 0.01 degree. The system’s accuracy and efficiency makes it extremely suitable to be applied to polymerase chain reaction, a biomedical reaction for generating DNA strand replica, which requires multiple heating and cooling cycle. Our system contains of 3 main components: user interface, current amplifier, signal conditioner and the processing unit. Now, please allow me to get into the detail of our solution, starting from the user interaction module.

The user interface of our system is consisted of local input, local output, and cloud output. The local input is an 10K potential meter nob connected to an analog pin of the Arduino, the center processing unit. The function AnalogRead() was used to read the analog pin connecting to the potential meter and function Map() is used to map the raw analog value from AnalogRead() to our desired temperature range 5-45 degree. The the LCD display, the official LCD lib from Arduino is used which makes printing string on the LCD as easy as printing lines in serial monitor. For more information of the system’s status, we used to an Arduino Wifi Shield to obtain http connectivity. The Arduino acts as an http client in this case and constantly assessing API from dweet.io, an IOT data record platform. Thanks to dweet.io, pushing data through their API is as easy as making request to a certain domain name.

A thermistor is used for temperature sensing of the system. Knowing that the resistance of the thermistor will change according to the temperature, we adopted the method of using a NE555 timer in a stable mode to detect the change. We exploit the fact that NE555’s output frequency is changing sensitively according to its configuration of resistor combination, given by this equation from the official specification [], and the typical change of frequency in our temperature range is from 4000 to 6000 hz which is wide enough for accurate record of temperature data mapping unlike the voltage divider method. The relationship between temperature and frequency can be obtained in the collaboration by record a curve contain temperature data from AD590 and frequency of the NE555 output by using an Arduino digital pin. An Arduino digital pin is capable of reading frequency input ranging from 0 – 10K Hz in an accurate manner (proven by website). After obtaining the curve, temperature value can be regenerated easily using the curve.