

BAB IV

PEMBUATAN ALAT

4.1 Pembuatan Perangkat Keras (*Hardware*)

Pembuatan perangkat keras merupakan tahap realisasi dari rancangan yang telah disusun pada BAB III. Tahap ini mencakup pembuatan konstruksi mekanik, pemasangan media penyimpan panas, perakitan sistem pemanas, pemasangan sensor, serta penyusunan rangkaian monitoring. Proses pembuatan mengacu pada dimensi dan spesifikasi komponen yang telah ditetapkan agar prototipe *sand battery* dapat bekerja sesuai tujuan penelitian.

Prototipe menggunakan konstruksi dua tabung yang terdiri atas tabung dalam dan tabung luar. Tabung dalam berfungsi menampung 57 kg pasir silika berukuran 30 sampai 50 mesh, dua elemen pemanas, *coil* tembaga, dan dua thermocouple tipe K. Tabung luar berfungsi melindungi tabung dalam dan menjadi tempat pemasangan rockwool serta aluminium foil sebagai isolasi termal.

Sistem menggunakan dua panel surya 545 Wp yang terhubung secara seri sebagai sumber energi listrik. Kedua panel menyalurkan energi secara langsung menuju dua elemen pemanas 42 V dan 500 W yang juga terhubung secara seri. Elemen pemanas mengubah energi listrik menjadi energi panas, kemudian pasir silika menyerap dan menyimpan panas tersebut.

Sistem monitoring menggunakan sensor arus ACS712, sensor tegangan DC, dua modul MAX6675, ESP32 DevKit V1, Arduino Nano, LCD 20×4 , dan keypad 4×4 . Panel surya sekitar 10 Wp berfungsi sebagai panel referensi untuk mengestimasi iradiasi matahari. PSU 12 V 5 A mencatu rangkaian monitoring, sedangkan ESP32 mengirimkan data hasil pengukuran menuju *spreadsheet*.

4.1.1 Alat dan Bahan

Proses pembuatan prototipe *sand battery* memerlukan berbagai alat dan bahan yang saling mendukung. Peralatan digunakan dalam proses pengukuran, pemotongan, pembentukan, penyambungan, perakitan, serta pemeriksaan komponen. Bahan yang digunakan mencakup komponen mekanik, termal,

kelistrikan, elektronik, dan sistem monitoring yang membentuk satu kesatuan prototipe.

Pemilihan setiap alat dan bahan mengacu pada hasil perancangan yang telah dijelaskan pada BAB III. Komponen mekanik membentuk struktur utama prototipe yang terdiri atas tabung dalam, tabung luar, penutup tabung, dan *coil* tembaga. Pasir silika berukuran 30 sampai 50 mesh dengan massa 57 kg berfungsi sebagai media penyimpan energi termal, sedangkan rockwool dan aluminium foil berfungsi sebagai material isolasi.

Dua panel surya monokristalin berkapasitas masing-masing 545 Wp menyediakan energi listrik bagi dua elemen pemanas 42 V dan 500 W. Sistem monitoring menggunakan sensor arus ACS712, sensor tegangan DC, dua thermocouple tipe K, dua modul MAX6675, ESP32 DevKit V1, Arduino Nano, LCD 20×4 , dan keypad 4×4 . Panel surya berkapasitas sekitar 10 Wp berfungsi sebagai panel referensi untuk mengestimasi iradiasi matahari. PSU 12 V 5 A menyediakan catu daya bagi rangkaian monitoring, sedangkan *spreadsheet* menyimpan data hasil pengukuran.

Daftar alat yang digunakan selama proses pembuatan dan perakitan prototipe disajikan pada Tabel 4.1. Daftar bahan dan komponen yang membentuk prototipe *sand battery* disajikan pada Tabel 4.2.

Tabel 4 - 1 Daftar Alat Pembuatan Perangkat Keras (Hardware)

No.	Nama Alat	Fungsi
1	Mesin las	Menyambungkan bagian konstruksi logam
2	Mesin gerinda	Memotong dan merapikan permukaan pelat
3	Mesin bor	Membuat lubang pemasangan komponen
4	Pemotong pipa	Memotong pipa tembaga sesuai ukuran
5	Tang pembentuk pipa	Membantu pembentukan pipa tembaga
6	Solder	Menyambungkan komponen dan kabel elektronik
7	Multimeter	Mengukur tegangan, arus, resistansi, dan kontinuitas

No.	Nama Alat	Fungsi
8	Jangka sorong	Mengukur dimensi komponen secara presisi
9	Meteran	Mengukur dimensi tabung dan konstruksi
10	Timbangan	Mengukur massa pasir silika
11	Obeng	Memasang dan melepas baut atau terminal
12	Tang kombinasi	Memotong, menjepit, dan membentuk kabel
13	Komputer atau laptop	Memprogram mikrokontroler dan memantau data

Tabel 4 - 2 Bahan dan Komponen Prototipe

No.	Nama Bahan atau Komponen	Spesifikasi dan Fungsi
1	Tabung dalam	Diameter 32 cm dan tinggi 46 cm
2	Tabung luar	Diameter 50 cm dan tinggi 40 cm
3	Pasir silika	Massa 57 kg dan ukuran 30 sampai 50 mesh
4	Rockwool	Ketebalan 5 cm sebagai isolasi termal
5	Aluminium foil	Pelindung isolasi dan pengurang radiasi panas
6	Pipa tembaga	Material pembentuk <i>coil</i>
7	Elemen pemanas	2 unit, masing-masing 42 V dan 500 W
8	Panel surya utama	2 unit, masing-masing 545 Wp
9	Panel surya referensi	Sekitar 10 Wp untuk estimasi iradiasi
10	Thermocouple tipe K	2 unit untuk mengukur temperatur pasir
11	Modul MAX6675	2 unit untuk membaca thermocouple
12	Sensor arus ACS712	Mengukur arus pada rangkaian heater
13	Sensor tegangan DC	Mengukur tegangan pada rangkaian heater
14	ESP32 DevKit V1	Mengolah dan mengirimkan data monitoring
15	Arduino Nano	Mengelola LCD dan keypad
16	LCD	LCD 20 × 4 sebagai perangkat tampilan

No.	Nama Bahan atau Komponen	Spesifikasi dan Fungsi
17	Keypad	Keypad 4×4 sebagai perangkat masukan
18	PSU	12 V 5 A untuk catu daya sistem monitoring
19	Kabel listrik	Menghubungkan panel, heater, dan sensor
20	Kabel thermocouple	Menghubungkan thermocouple dengan MAX6675
21	Baut dan mur	Mengikat bagian konstruksi dan penutup tabung
22	Bahan penyekat	Menutup celah dan mengurangi kebocoran panas
23	<i>Spreadsheet</i>	Mencatat dan menyimpan data hasil pengukuran

4.1.2 Pembuatan Perangkat Mekanik

Perancangan mekanikal merupakan tahap penerapan rancangan konstruksi prototipe *sand battery* yang telah disusun pada BAB III. Tahap ini bertujuan membentuk struktur yang mampu menampung pasir silika, elemen pemanas, *coil* tembaga, sensor temperatur, dan material isolasi secara aman dan stabil.

Konstruksi mekanikal terdiri atas tabung dalam, tabung luar, penutup tabung, *coil* tembaga, elemen pemanas, thermocouple tipe K, rockwool, dan aluminium foil. Tabung dalam berfungsi sebagai wadah utama penyimpanan 57 kg pasir silika berukuran 30 sampai 50 mesh. Tabung tersebut memiliki diameter 32 cm dan tinggi 46 cm.

Tabung luar memiliki diameter 40 cm dan tinggi 50 cm. Komponen ini berfungsi melindungi tabung dalam dan material isolasi. Ruang antara kedua tabung menampung rockwool setebal 5 cm dan aluminium foil. Kedua material tersebut membantu mengurangi perpindahan panas dari tabung dalam menuju lingkungan.

Bagian dalam tabung memuat dua elemen pemanas dengan spesifikasi masing-masing sebesar 42 V dan 500 W. Kedua elemen pemanas ditempatkan

secara vertikal melalui penutup tabung. Sistem juga menggunakan *coil* tembaga sebagai jalur aliran udara dan penukar panas antara pasir silika dengan udara dari blower.

Dua thermocouple tipe K dipasang secara vertikal melalui penutup tabung. Kedua sensor menempati posisi horizontal yang berbeda untuk mengukur temperatur pasir pada dua lokasi. Penempatan sensor tidak bersentuhan langsung dengan heater, *coil* tembaga, atau dinding tabung agar hasil pengukuran mewakili temperatur pasir silika.

Penutup tabung menopang dua elemen pemanas, dua thermocouple, serta saluran masuk dan keluar *coil* tembaga. Baut pada bagian tepi menghubungkan penutup dengan badan tabung dan memudahkan proses pembukaan saat pemeriksaan atau pemeliharaan. Proses saat membuat dan merakit perangkat mekanik dapat dilihat pada Gambar 4. 1.



Gambar 4. 1 Proses Pengerjaan Mekanikal Prototipe Sand Battery

Tabel 4 - 3 Spesifikasi Pengerjaan Mekanikal

Bagian	Spesifikasi
Bentuk konstruksi	Silinder dengan dua lapisan tabung
Tabung dalam	Diameter 32 cm dan tinggi 46 cm
Tabung luar	Diameter 40 cm dan tinggi 50 cm
Media penyimpan	Pasir silika
Massa pasir	57 kg
Ukuran pasir	30 sampai 50 mesh
Material isolasi	Rockwool 5 cm dan aluminium foil
Jumlah heater	2 unit
Spesifikasi heater	42 V dan 500 W per unit
Orientasi heater	Vertikal
Material penukar panas	Pipa tembaga
Sensor temperatur	2 thermocouple tipe K

Hasil pengerjaan mekanikal dengan spesifikasi yang disebutkan pada Tabel 4 - 3 membentuk satu konstruksi yang mengintegrasikan media penyimpan panas, elemen pemanas, penukar panas, sensor temperatur, dan isolasi termal. Susunan tersebut menyediakan ruang yang memadai bagi pasir silika dan mendukung proses penyimpanan serta pengambilan energi termal.

4.1.2.1 Fabrikasi Komponen Mekanikal oleh Vendor



Gambar 4. 2 Pembuatan Tabung dan Tutup

Vendor melakukan fabrikasi komponen mekanikal berdasarkan gambar kerja yang telah penulis susun pada tahap perancangan. Proses fabrikasi menghasilkan tabung dalam, tabung luar, penutup tabung sebagai komponen utama prototipe seperti terlihat pada Gambar 4. 2.

Tabung dalam memiliki diameter 32 cm dan tinggi 46 cm. Tabung tersebut berfungsi sebagai wadah pasir silika, elemen pemanas, *coil* tembaga, dan thermocouple. Tabung luar memiliki diameter 50 cm dan tinggi 40 cm. Tabung luar berfungsi sebagai pelindung mekanis dan rumah bagi material isolasi termal.

Vendor juga membuat penutup tabung yang memuat lubang pemasangan dua elemen pemanas, dua thermocouple, serta saluran masuk dan keluar udara. Posisi lubang mengikuti tata letak komponen yang telah penulis tentukan pada gambar kerja. Spesifikasi keseluruhan tabung dapat dilihat pada Tabel 4 – 4.

Tabel 4 - 4 Spesifikasi Tabung

No.	Komponen	Spesifikasi utama
1	Tabung dalam	Diameter 32 cm dan tinggi 46 cm
2	Tabung luar	Diameter 40 cm dan tinggi 50 cm
3	Penutup tabung	Memuat lubang heater, thermocouple, dan saluran udara
4	<i>Coil</i> tembaga	Diameter 25,1 cm dan tinggi 36 cm

4.1.2.2 Pemasangan Elemen Pemanas

Penulis memasang dua elemen pemanas melalui lubang yang terdapat pada penutup tabung. Setiap elemen memiliki spesifikasi 42 V dan 500 W. Bagian terminal berada di atas penutup, sedangkan batang pemanas memanjang secara vertikal ke dalam tabung.

Penulis menempatkan kedua elemen pemanas pada dua posisi yang berbeda untuk memperluas daerah pemanasan. Penulis juga memastikan batang pemanas tidak menyentuh dinding, dasar tabung, atau *coil* tembaga.

Posisi terminal pada bagian atas memudahkan proses penyambungan kabel dan pemeriksaan komponen. Penulis menggunakan pengikat dan isolator yang sesuai untuk mempertahankan posisi heater serta mencegah kontak listrik dengan

penutup logam. Dokumentasi saat proses pemasangan dapat dilihat pada Gambar 4. 3.



Gambar 4. 3 Pemasangan Dua Elemen Pemanas pada Penutup Tabung

4.1.2.3 Pemasangan Thermocouple Tipe K

Dua thermocouple tipe K dipasang secara vertikal melalui lubang pada penutup tabung untuk mengukur temperatur pasir silika pada dua titik yang berbeda. Salah satu sensor ditempatkan pada bagian tengah, sedangkan sensor lainnya ditempatkan pada bagian tepi tabung.

Ujung kedua thermocouple dipastikan terendam di dalam pasir serta tidak bersentuhan langsung dengan elemen pemanas, *coil* tembaga, dinding, maupun dasar tabung. Penempatan tersebut bertujuan memperoleh data temperatur dari dua lokasi berbeda di dalam media penyimpanan panas.

Data dari kedua thermocouple dibaca melalui modul MAX6675 dan diolah oleh ESP32. Hasil pembacaan ditampilkan sebagai Suhu 1, Suhu 2, dan temperatur rata-rata pasir silika.

4.1.2.4 pemasangan rockwool dan aluminium foil

Pemasangan rockwool dan aluminium foil dilakukan untuk mengurangi perpindahan panas dari tabung penyimpanan menuju lingkungan. Rockwool dipasang pada bagian luar tabung sebagai lapisan isolasi termal dengan ketebalan sekitar 5 cm. Material rockwool dipotong dan disesuaikan dengan bentuk tabung, kemudian dipasang secara merata agar tidak terdapat celah yang dapat menjadi jalur keluarnya panas.

Setelah rockwool terpasang, bagian luarnya dilapisi menggunakan aluminium foil. Aluminium foil berfungsi sebagai lapisan pelindung rockwool sekaligus membantu mengurangi perpindahan panas melalui radiasi. Pada konfigurasi akhir, rockwool dan aluminium foil dipasang pada seluruh sisi tabung, termasuk bagian penutup. Pemasangan pada area di sekitar elemen pemanas, thermocouple, dan jalur komponen lainnya dilakukan secara hati-hati agar material isolasi tetap rapat tanpa mengganggu posisi maupun fungsi komponen.

Setelah proses pemasangan selesai, dilakukan pemeriksaan untuk memastikan rockwool dan aluminium foil telah menutupi permukaan tabung dengan baik. Pemasangan kedua material tersebut diharapkan dapat mengurangi kehilangan panas menuju lingkungan dan membantu pasir silika mempertahankan temperatur setelah proses pemanasan berakhir. Dokumentasi saat proses pemasangan rockwool dan aluminium foil dapat dilihat pada Gambar 4. 4.



Gambar 4. 4 Proses Pemasangan Rockwool dan Aluminium Foil

4.1.2.5 Hasil Akhir Perakitan Mekanikal



Gambar 4. 5 Hasil Akhir Perakitan Mekanikal Prototipe Sand Battery

Hasil perakitan mekanikal membentuk prototipe *sand battery* dengan konstruksi dua lapis seperti di perlihatkan pada Gambar 4. 5. Tabung dalam menampung pasir silika, dua elemen pemanas, *coil* tembaga, dan dua thermocouple. Tabung luar melindungi rockwool serta aluminium foil yang mengelilingi tabung dalam.

Penulis menempatkan terminal heater, konektor thermocouple, dan saluran udara pada bagian atas konstruksi. Susunan tersebut memudahkan proses penyambungan, pemeriksaan, dan pemeliharaan.

Perakitan mekanikal mengacu pada hasil fabrikasi vendor dan rancangan yang telah penulis tetapkan. Penulis melakukan pemeriksaan serta penyesuaian selama proses perakitan agar setiap komponen berada pada posisi yang sesuai. Spesifikasi keseluruhan dari perangkat mekanik yang dibuat dapat dilihat pada Tabel 4 – 5.

Tabel 4 - 5 Spesifikasi Keseluruhan Alat

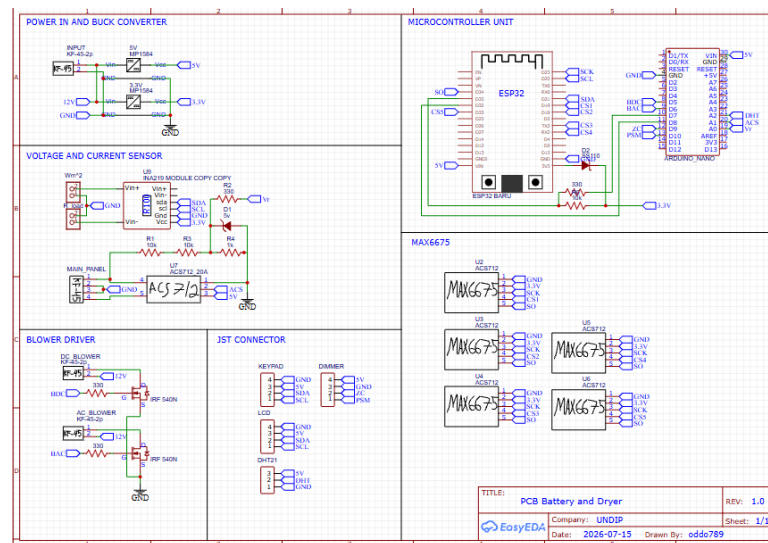
Bagian	Hasil perakitan
Tabung dalam	Diameter 32 cm dan tinggi 46 cm
Tabung luar	Diameter 40 cm dan tinggi 50 cm
Media penyimpan	Pasir silika 57 kg
Ukuran pasir	30 sampai 50 mesh
Isolasi	Rockwool 5 cm dan aluminium foil
Elemen pemanas	2 unit, 42 V dan 500 W
Orientasi heater	Vertikal
Thermocouple	2 unit tipe K
Pelaksana fabrikasi	Vendor
Pelaksana perakitan	Penulis

4.1.3 Pembuatan Elektrikal

Pembuatan elektrikal merupakan tahap realisasi sistem kelistrikan berdasarkan rancangan yang telah disusun pada BAB III. Tahap ini meliputi pemilihan dan pemasangan komponen, sensor, mikrokontroler, serta perangkat pendukung yang digunakan pada sistem monitoring.

Proses pembuatan elektrikal mencakup pembuatan *wiring diagram*, desain dan pembuatan PCB, pemasangan komponen pada PCB, penyambungan jalur kelistrikan, serta pemeriksaan hubungan antarperangkat. Seluruh proses tersebut dilakukan untuk memastikan sistem kelistrikan dapat membaca, mengolah, menampilkan, dan mencatat data pengujian dengan baik.

4.1.3.1 Pembuatan wiring diagram

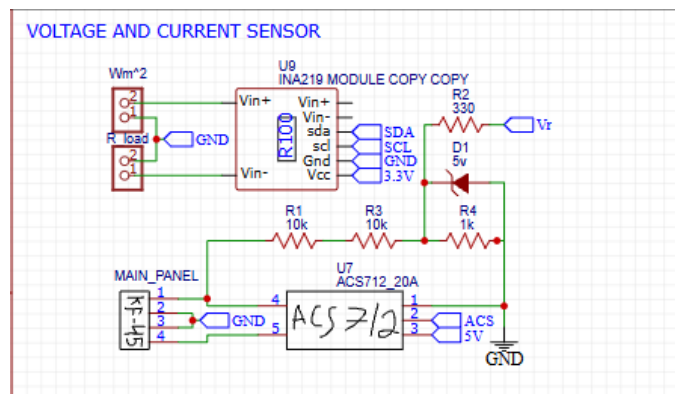


Gambar 4. 6 Wiring Diagram

Setiap komponen dihubungkan berdasarkan fungsi, kebutuhan tegangan, dan jalur komunikasi yang mengacu pada *datasheet* masing-masing perangkat. Pembuatan *wiring diagram* dimulai dari pemetaan jalur catu daya, jalur sensor, jalur komunikasi, dan hubungan antarmikrokontroler. Keseluruhan wiring diagram sistem dapat dilihat pada Gambar 4. 6.

Sumber tegangan rangkaian monitoring berasal dari *power supply*, kemudian diturunkan menggunakan *buck converter* MP1584EN sesuai dengan tegangan kerja masing-masing komponen. Tegangan **3,3 V** digunakan untuk mencatu ESP32, MAX6675, dan INA219, sedangkan tegangan **5 V** digunakan untuk mencatu Arduino Nano, LCD, keypad, dan ACS712 seperti dapat dilihat pada Gambar 4. 7.

dengan target pembacaan maksimal 105V yang diturunkan menjadi 5V sesuai dengan rating tegangan Arduino Nano, pada output sensor tegangan diberikan rangkaian pengaman berupa resistor 330 ohm sebagai pembatas arus dan dioda zener 5V untuk pengaman bila terjadi lonjakan tegangan diatas tegangan kerja. Sensor ACS712 terhubung dengan Arduino Nano dan dibaca menggunakan AnalogRead karena sensor ini mengkonversi besaran arus menjadi skala pembacaan berupa sinyal analog, pembacaan sinyal analog ini dilakukan dengan menggunakan library pada koding software Arduino Nano. INA219 terhubung dengan ESP32 dan berkomunikasi menggunakan protokol I2C sehingga pembacaan Arus, Tegangan dan Daya dapat dilakukan sekaligus oleh chip pada sensor lalu di konversi menjadi sinyal serial yang dibaca oleh library pada program ESP32. Wiring sensor tegangan dan arus dapat dilihat pada Gambar 4. 9.

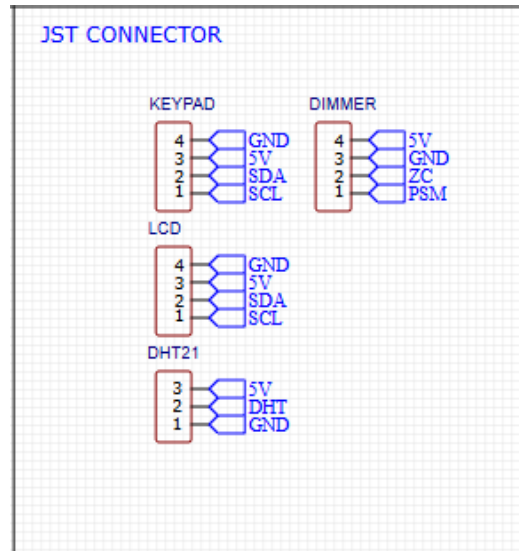


Gambar 4. 9 Wiring Voltage and Current Sensor

Pada bagian rangkaian aktuator, mikrokontroller mengatur kondisi pada aktuator berdasarkan data yang di dapatkan dari sensor dan masukan kondisi dari pengguna melalui interface.

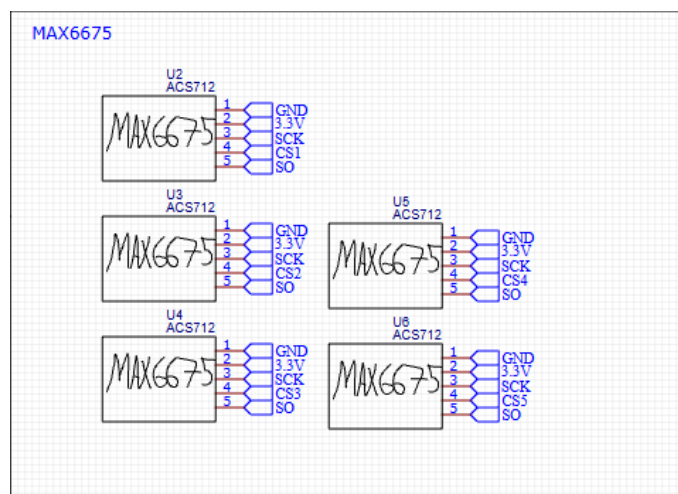
Jst connector digunakan sebagai jembatan penghubung antara kelistrikan PCB dan komponen yang tidak terpasang langsung di PCB supaya jalur kelistrikan terhubung dengan baik dan aman dari guncangan ataupun gangguan yang dapat menghambat penyaluran power untuk komponen maupun komunikasi antar komponen. Penggunaan jst juga mempermudah proses maintenance karena untuk melepaskan koneksi dari pcb dan komponen dapat dilakukan dengan melepas

konektor tersebut. Komponen yang terhubung menggunakan jst adalah Keypad, dan LCD. Wiring JST konektor dapat dilihat pada Gambar 4. 10.



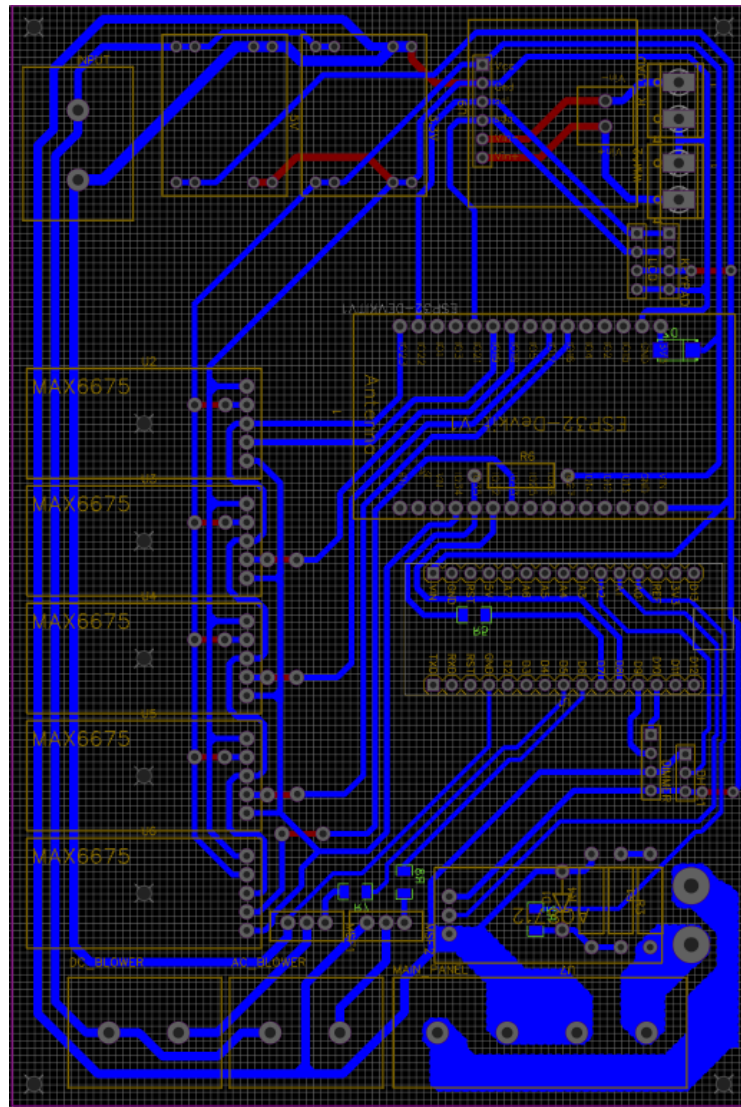
Gambar 4. 10 Wiring JST Connector

MAX6675 membaca suhu dari thermocouple untuk dikirimkan dan dibaca oleh ESP32, maka jalur kelistrikan antara sensor ini dan mikrokontroller harus dijaga supaya tetap stabil dan minin gangguan. Pada konektor pin header dimana sensor ini di pasang di PCB diberikan support berupa mur yang menjaga supaya sensor ini tidak bergerak jika sudah terpasang pada PCB. Wiring MAX6675 dapat dilihat pada Gambar 4. 11.



Gambar 4. 11 Wiring MAX6675

4.1.3.2 Pembuatan Rangkaian PCB



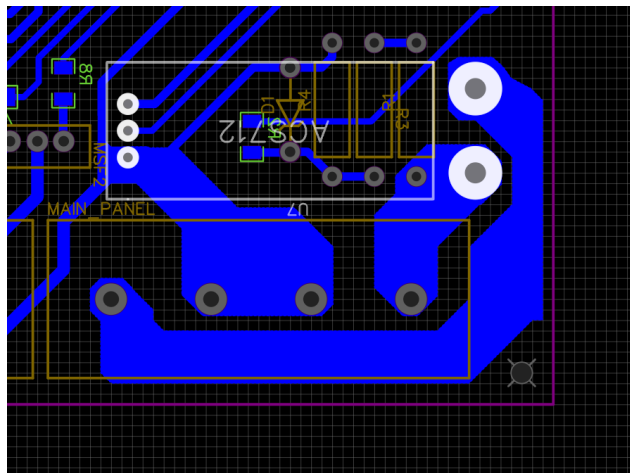
Gambar 4. 12 Layout PCB

Seperti pada Gambar 4. 12, rangkaian jalur PCB dibuat berdasarkan skematik/wiring diagram yang sudah dirancang sebelumnya. Pada pembuatan jalur pcb perlu diperhatikan seberapa besar daya yang akan disalurkan pada *wiring trace* dan penggunaan jalur tersebut sebagai jalur sinyal atau jalur power. Bila jalur digunakan sebagai jalur power maka ukuran dari jalur harus di pertebal jika dibandingkan jalur sinyal karena jalur power akan membawa arus yang lebih besar seperti pada jalur pembacaan tegangan dan arus panel utama yang terhubung ke

PCB menggunakan terminal block KF-45 lalu jalur yang menghubungkan terminal block tersebut ke sensor tegangan dan ACS712 dibuat dengan tebal minimal 3mm.

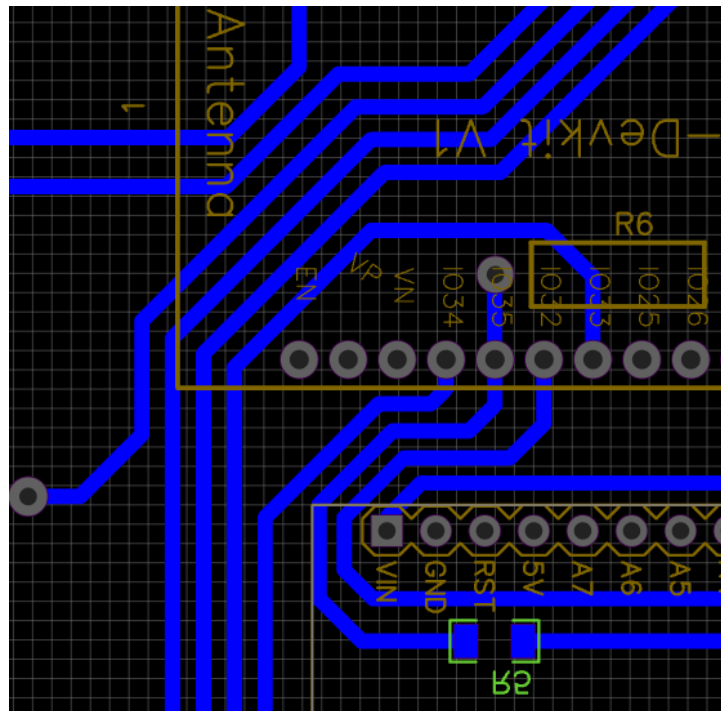
Lalu untuk mendapatkan lebar jalur PCB bagi Area dengan tebal tembaga pada PCB yang bisa didapatkan dari datasheet PCB yang digunakan atau keterangan dari penjual lalu hasil perhitungan dalam mils di konversi menjadi milimeter. Perhitungan untuk jalur bagian pembacaan tegangan adalah

Jika dikonversi menjadi mm maka didapat 11,98 mm² yang mana ini tidak memungkinkan karena kurangnya ruang untuk jalur pada PCB, maka alternatif yang digunakan adalah menebalkan jalur PCB menggunakan tenol solder sehingga jalur mampu menahan arus yang tinggi tanpa menghasilkan panas berlebih yang dapat merusak komponen lainnya dan PCB itu sendiri. Lebar jalur untuk power PCB dapat dilihat pada Gambar 4. 13.



Gambar 4. 13 Jalur Power PCB

Jalur sinyal pada PCB dapat dibuat lebih sempit dibandingkan jalur daya karena hanya membawa arus yang relatif kecil seperti diperlihatkan pada Gambar 4. 14. Perancangan jalur sinyal perlu memperhatikan jarak antarkomponen, panjang jalur, serta pemisahannya dari jalur daya untuk mengurangi kemungkinan gangguan dan *noise*. Meskipun komunikasi I²C, SPI, dan UART pada sistem bekerja pada frekuensi relatif rendah sehingga pengaruh impedansi akibat bentuk belokan tidak signifikan, jalur tetap dirancang menggunakan belokan 45° atau lengkung dan menghindari belokan tajam 90° agar tata letak PCB lebih rapi serta mudah difabrikasi.

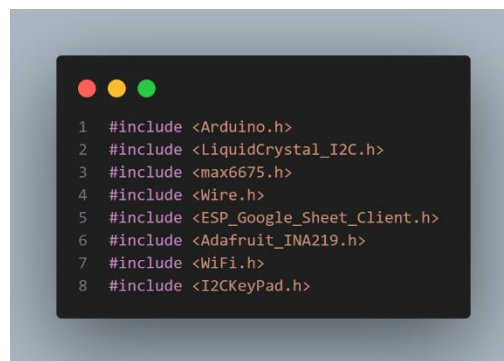


Gambar 4. 14 Jalur Sinyal PCB

4.2 Pembuatan Perangkat Lunak (Software)

4.2.1 Inisiasi dan Deklarasi

4.2.1.1 Library



Gambar 4. 15 Deklarasi Library ESP32

Deklarasi library yang digunakan pada keseluruhan program ESP32 dengan menggunakan format kode `#include <Nama_Library>`. Library yang digunakan pada ESP32 adalah `<Arduino.h>` sebagai library untuk framework arduino pada platform.io, `<LiquidCrystal_I2C.h>` sebagai library LCD 20x4 i2c, `<max6675.h>` sebagai library pembacaan thermocouple, `<Wire.h>` sebagai library untuk

komunikasi serial pada mikrokontroller dengan framework arduino, <ESP_Google_Sheet_Client.h> sebagai library untuk upload data menggunakan token google ke spreadsheet, <Adafruit_INA219.h> sebagai library INA219 untuk membaca radiasi matahari melalui panel 10W, <WiFi.h> sebagai library untuk menjalankan wifi ESP32, <I2CkeyPad.h> sebagai library untuk keypad i2c. Deklarasi library pada ESP32 dapat dilihat pada Gambar 4. 15.



Gambar 4. 16 Deklarasi Library Arduino Nano

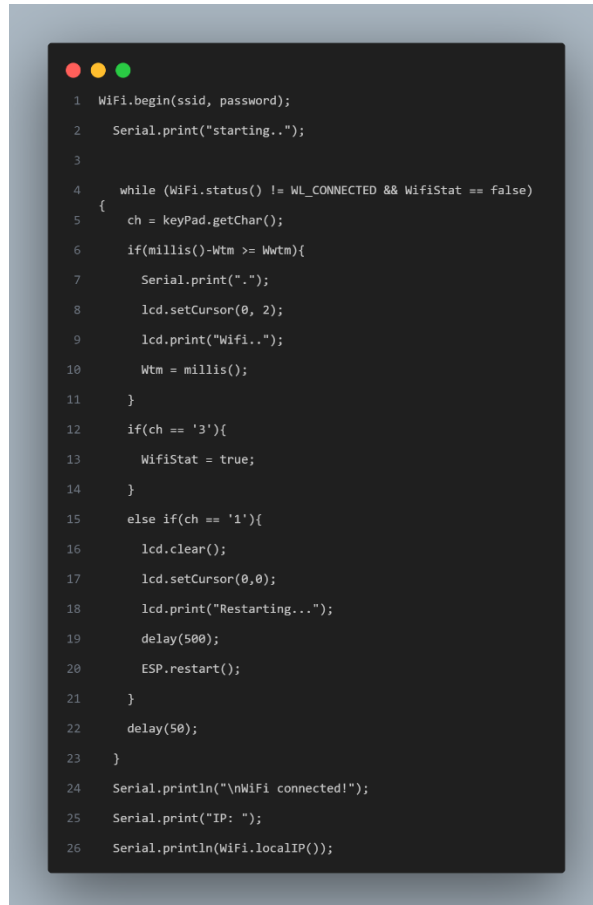
Deklarasi Library yang digunakan pada keseluruhan program Arduino Nano menggunakan format `#include <Nama_Library>`. Library yang digunakan pada Arduino Nano adalah <Arduino.h> sebagai library untuk framework arduino pada platform.io, <ACS712.h> sebagai library untuk pembacaan arus oleh sensor ACS712, <DHT.h> sebagai library untuk membaca suhu dan kelembapan pada oven dan ruang sekitar oven dan baterai, <SoftwareSerial.h> sebagai library untuk komunikasi uart tanpa menggunakan pin default uart Arduino Nano. Deklarasi library pada Arduino Nano dapat dilihat pada Gambar 4. 16.

4.2.1.2 Wifi



Gambar 4. 17 Deklarasi Wifi ESP32

Deklarasi wifi di mana SSID dan Password wifi dicantumkan supaya ESP32 dapat terhubung ke wifi yang disediakan seperti pada Gambar 4. 17.



```

1  WiFi.begin(ssid, password);
2  Serial.print("starting..");
3
4  while (WiFi.status() != WL_CONNECTED && WifiStat == false)
5  {
6      ch = keypad.getKey();
7      if(millis()-Wtm >= Wwtm){
8          Serial.print(".");
9          lcd.setCursor(0, 2);
10         lcd.print("Wifi..");
11         Wtm = millis();
12     }
13     if(ch == '3'){
14         WifiStat = true;
15     }
16     else if(ch == '1'){
17         lcd.clear();
18         lcd.setCursor(0,0);
19         lcd.print("Restarting...");
20         delay(500);
21         ESP.restart();
22     }
23     delay(50);
24 }
25 Serial.println("\nWifi connected!");
26 Serial.print("IP: ");
27 Serial.println(WiFi.localIP());

```

Gambar 4. 18 Eksekusi Wifi ESP32

Eksekusi wifi pada void setup() saat ESP32 pertama kali dihidupkan, sistem menunggu sampai wifi terhubung untuk melanjutkan ke proses selanjutnya. Percobaan hubung wifi bisa dilewati dengan menekan tombol '0' pada keypad. Kode untuk eksekusi wifi ESP32 dapat dilihat pada Gambar 4. 18.



```

1  void wifi(){
2      rssi = WiFi.RSSI();
3
4      if(WiFi.status() == WL_CONNECTED && !terhubung ){
5          configTime(7 * 3600,0 , ntpServer);
6          Serial.println("Connected!");
7          wipistat = 'O';
8
9          digitalWrite(2, HIGH);
10
11         terhubung = HIGH;
12     }
13
14     if(WiFi.status() != WL_CONNECTED){
15         if(millis() - Rctm > Reconnect_Interval){
16             Serial.println(".");
17             wipistat = 'X';
18             WiFi.disconnect();
19             WiFi.begin(ssid, password);
20             digitalWrite(2, LOW);
21             terhubung = LOW;
22             Rctm = millis();
23         }}
24 }

```

Gambar 4. 19 Pemantauan Kondisi Wifi

Void wifi() memiliki fungsi untuk memantau kondisi wifi disaat sistem sedang berjalan, void ini akan terus mencoba terhubung ke wifi apabila koneksi wifi ESP32 terputus dan akan menunjukkan kondisi wifi (terhubung atau terputus) pada LCD. Kode untuk pemantauan kondisi wifi dapat dilihat pada Gambar 4. 19.

4.2.1.3 Deklarasi Data Lain

```

1 #define R002 35
2 #define T002 32
3
4 //POTANS
5 #define therm000 34
6 #define thermC01 19
7 #define thermC02 18
8 #define thermC03 17
9 #define thermC04 16
10 #define thermC05 11
11 #define thermC06 23
12 #define thermCouple01(thermC01, thermC02, therm000);
13 #define thermCouple02(thermC01, thermC02, therm000);
14 #define thermCouple03(thermC01, thermC03, therm000);
15 #define thermCouple04(thermC01, thermC04, therm000);
16 #define thermCouple05(thermC01, thermC05, therm000);
17
18 float C1 = 0;
19 float C2 = 0;
20 float C3 = 0;
21 float C4 = 0;
22 float C5 = 0;
23 float Avg_Bat = 0;
24 float Avg_Ov = 0;
25 float Delta_Bat = 0;
26 float DeltaOidBat = 0;
27 float Delta_Ov = 0;
28 float DeltaOidOv = 0;
29 bool FirstDataBat = true;
30 bool FirstDataOv = true;
31
32 //I2C
33 #define I2CAddress 0x48;
34 char ch = 'N';
35 char ch = 'N';
36 bool KeyIn = false;
37 unsigned long lastPressTime = 0;
38 const unsigned long debounceDelay = 50;
39 char keypad[10] = "1234567890*";
40
41 /*
42  * D I B *
43  * C 9 8 7
44  * 6 5 4
45  * A 3 2 1
46  */
47
48 LiquidCrystal_I2C lcd(0x27,20,4);
49
50 Adafruit_DHT11 dht11;
51 float Solar_Radiation = 0;
52
53 //BATT
54 float voltage = 0;
55 float current = 0;
56 float Kalman_Batt_Oven = 0;
57 float Suhu_DHT_Oven = 0;
58 float Kalman_Batt_Battery = 0;
59 float Suhu_DHT_Battery = 0;
60
61 //INA219
62 float BusVoltage = 0;
63 float Current_mA = 0;
64 float Power_mW = 0;
65
66 //Epoch Time
67 const char* ntpServer = "pool.ntp.org";
68 String waktuStr = "";
69 String getFormattedTime() {
70   struct tm timeinfo;
71   if (!getLocalTime(&timeinfo)) {
72     return "Time error";
73   }
74   char buf[25];
75   sprintf(buf, "%d %d %d %d %d %d", timeinfo.tm_Mon, timeinfo.tm_May, timeinfo.tm_Day, timeinfo.tm_Hour, timeinfo.tm_Min, timeinfo.tm_Sec);
76   return String(buf);
77 }
78
79 //LCD Display
80 enum Layer{Home, Changing, Suhu_Battery, Suhu_Oven, DHT21_OVEN, DHT21_BATTERY, Panel_Ulana, Radiasi, Oven_Run, Oven_Stop, Oven_Running, Oven_Stopping, Restart, LCD_Restart, id};
81 Layer layer = Home;
82 Layer lastlayer = Home;
83 unsigned long lastIdSwitch = 0;
84 const unsigned long lcdInterval = 3000;
85 int lcdPage = 0;
86 const int totalOfPages = 6;
87 int lcdRow = 1;
88
89 //Run Oven
90 bool OvenMenu = false;
91 bool OvenDate = false;
92 bool OvenRunning = false;
93 unsigned long overTime = 0;
94 const unsigned long OverTime = 2000;

```

Gambar 4. 20 Deklarasi Library dan Data Lain ESP32

```

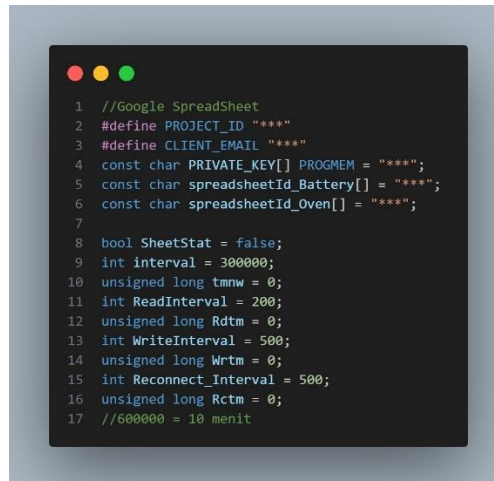
1  #define Vin A0
2  #define ACS_Pin A1
3  #define DHTPIN1 A2
4  #define DHTPIN2 A3
5  #define DHTTYPE DHT21
6  #define BDC 5
7  #define BAC 6
8  #define RX 8
9  #define TX 7
10
11
12  ACS712 ACS(ACS_Pin, 5.0, 1023, 100);
13
14  SoftwareSerial Serial2(RX, TX);
15
16  DHT dht(DHTPIN1, DHTTYPE);
17  DHT dht1(DHTPIN2, DHTTYPE);
18  //UART
19  float C1 = 0;
20  float C2 = 0;
21  float C3 = 0;
22
23  //Read Sens
24  int WriteInterval = 500;
25  unsigned long Wrtm = 0;
26  int ReadInterval = 200;
27  unsigned long Rdtm = 0;
28
29  //Read avg Voltage
30  int Voltage_Interval = 2;
31  unsigned long Vltm = 0;
32  long voltTotal = 0;
33  int voltSampleCount = 0;
34  const int VoltSamples = 100; //
35
36  //Read avg Current
37  int Current_Interval = 1;
38  unsigned long Cltm = 0;
39  long CurrentTotal = 0;
40  int CurrentSampleCount = 0;
41  const int CurrentSamples = 200; //
42
43  float mA = 0;
44  float h = 0;
45  float t = 0;
46  float h1 = 0;
47  float t1 = 0;
48  float avgAdc = 0;
49  double Vp = 0;
50  double Vreal = 0;
51
52  //Kontrol
53  float SetPoint = 45;
54  float Hysteresis = 2;
55  int ControlInterval = 500;
56  unsigned long Cttm = 0;
57  bool BlowerAC = false;
58  float HighHyst;
59  float LowHyst;
60  int OvenSerial = 0;
61  bool OvenState = 0;
62
63  //BLOWER
64  unsigned long Idop = 60000; // 1 menit = 60000
65  float TimerIdop = 3600000; //1 jam = 3600000
66  unsigned long LagiAktip = 0;
67  unsigned long LagiAktipJugo = 0;
68  bool BoolBombong = true;
69  bool Pakum = false;
70

```

Gambar 4. 21 Deklarasi Library dan Data Lain Arduino Nano

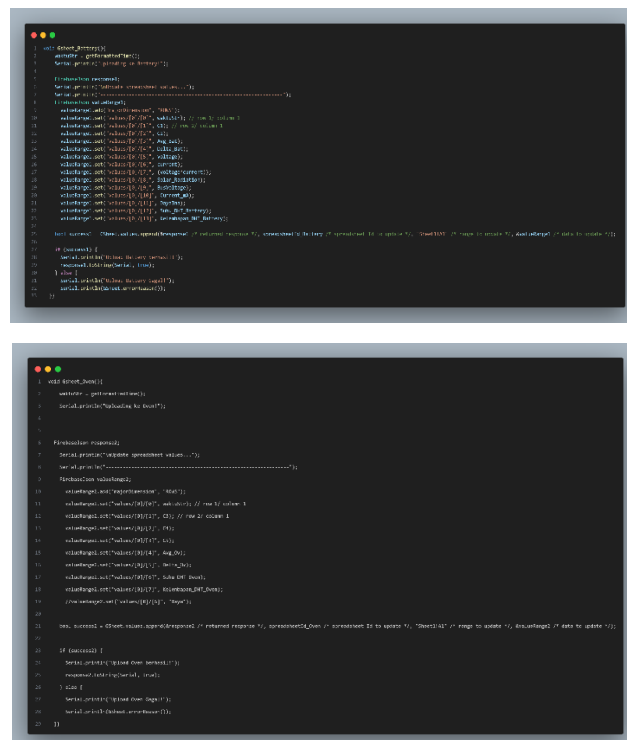
Deklarasi library, tipe data dan lain-lain untuk menjalankan program pada ESP32 dan Arduino Nano dapat dilihat pada Gambar 4. 20 dan Gambar 4. 21.

4.2.1.4 Spreadsheet



Gambar 4. 22 Deklarasi Spreadsheet

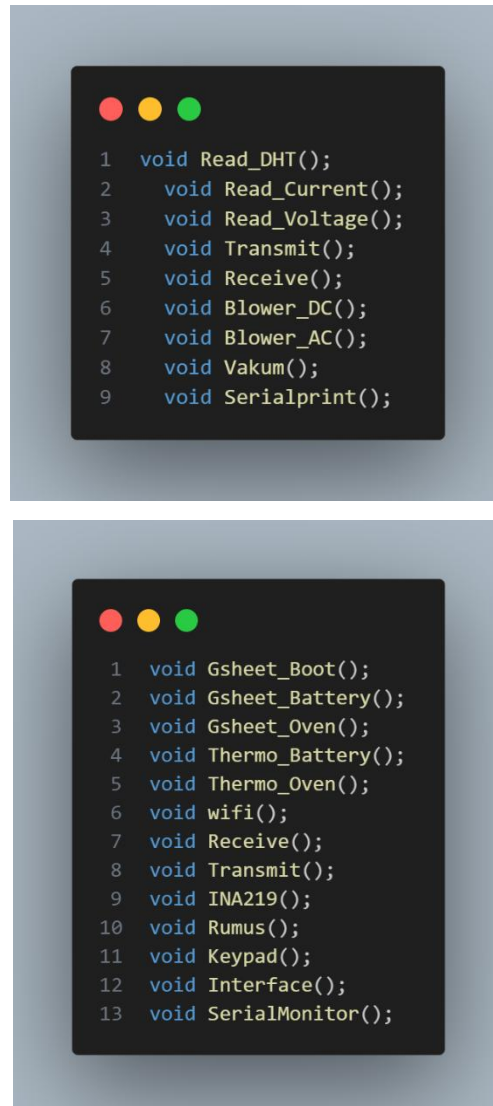
Deklarasi program spreadsheet yang menentukan spreadsheet id dan project id untuk mengupload data ke spreadsheet dan interval upload data ke spreadsheet seperti diperlihatkan pada Gambar 4. 22.



Program pengiriman data menuju *spreadsheet* terdiri atas fungsi Gsheet_Boot dan Gsheet_Battery. Fungsi Gsheet_Boot mengirimkan informasi awal ketika sistem diaktifkan, sedangkan Gsheet_Battery mengirimkan data sensor dan hasil perhitungan yang berkaitan dengan proses pengisian serta penyimpanan energi termal pada pasir silika. Pemisahan kedua fungsi tersebut memudahkan

pengelompokan dan analisis data pengujian sistem. Program keseluruhan pengiriman data menuju spreadsheet dapat dilihat pada Gambar 4. 23.

4.2.1.5 Prototype



Gambar 4. 24 Deklarasi fungsi software

Merupakan bagian yang mendeklarasikan fungsi yang digunakan di dalam software. Deklarasi fungsi software dapat dilihat pada Gambar 4. 24.

4.2.1.6 Interface

```

1 void Interface(){
2     char Nch = keyPad.getChar();
3
4     if(Nch != 0ch) {
5         if(Nch != 'N' && Nch != 'F') { ch = Nch; }
6         else { Keyin = false; ch = 'N'; }
7         Och = Nch;
8     } else {
9         ch = 'N';
10    }
11
12    if(ch != 'N') {
13        Serial.print("Key Pressed : ");
14        Serial.println(ch);
15    }
16
17    if(layar != zzz){
18        static unsigned long Memek = 0;
19        if(millis()-Memek>=500){
20            Memek = millis();
21            Serial.print("Layar : ");
22            Serial.println(layar);
23            Serial.print("Bool : ");
24            Serial.print(OvenMenu);
25            Serial.print("/");
26            Serial.print(OvenState);
27            Serial.print("/");
28            Serial.println(OvenRunning);
29        }
30    }

```

```

1 if(ch == '3' && !OvenMenu) { layar = Home; }
2 else if(ch == '1' && !OvenMenu) { layar = LastLayar; }
3 else if(ch == '+') { layar = Changing; }
4 else if(ch == 'D') { layar = Suhu_Battery; }
5 else if(ch == 'B') { layar = Suhu_Oven; }
6 else if(ch == 'O') { layar = DHT21_OVEN; }
7 else if(ch == 'C') { layar = DHT21_BATTERY; }
8 else if(ch == 'P') { layar = Panel_Utama; }
9 else if(ch == 'R') { layar = Radiasi; }
10 else if(ch == '7' && !OvenMenu && !OvenState) { LastLayar = layar; layar = Oven_Run; }
11 else if(ch == '7' && !OvenMenu && OvenState) { LastLayar = layar; layar = Oven_Stop; }
12 else if(ch == '1') { layar = Restart; }
13 else if(ch == 'A' && layar != zzz) { LastLayar = layar; layar = zzz; }
14 else if(ch == 'A' && layar == zzz) { layar = LastLayar; }
15 else if(ch == '2') { LastLayar = layar; layar = LCD_Restart; }
16
17 if(layar == Oven_Run || layar == Oven_Stop) { OvenMenu = true; }
18 else { OvenMenu = false; }
19
20 if(OvenRunning && OvenMenu && ch == '4') { layar = Oven_Running; OvnPup = millis(); }
21 if(!OvenRunning && OvenMenu && ch == '4') { layar = Oven_Stopping; OvnPup = millis(); }
22
23 if(layar == Oven_Running){OvenState = true;}
24 else if(layar == Oven_Stopping){OvenState = false;}
25
26 if((layar == Oven_Running || layar == Oven_Stopping) && millis() - OvnPup >= OvnPupInv) {
27     layar = LastLayar;
28 }
29

```

```

1 static unsigned long lastLcdDraw = 0;
2 if(millis() - lastLcdDraw >= 250 || layer != LcdReset) {
3     lastLcdDraw = millis();
4
5     if(layer != LcdReset) { lcd.clear(); }
6     if(layer == zzz) { lcd.noBacklight(); }
7     else if(layer != zzz) { lcd.backlight(); }
8

```

```

1 switch(layer) {
2     case Home:
3         lcd.setCursor(0,0); lcd.print("Home");
4         lcd.setCursor(19,0); lcd.print(wipistat);
5         lcd.setCursor(0, 1); lcd.print("Pencet Apa kek ");
6         break;

```

```

1 case Changing:
2     #pragma region changing
3     if (millis() - lastLcdSwitch >= lcdInterval || layer != LcdReset) {
4         lastLcdSwitch = millis();
5         lcd.clear();
6
7         switch (lcdPage) {
8             case 0:
9                 lcd.setCursor(0,0); lcd.print("Battery"); lcd.setCursor(19,0); lcd.print(wipistat);
10                lcd.setCursor(0, 1); lcd.print("Suhu 1: "); lcd.print(C1); lcd.print(" C");
11                lcd.setCursor(0, 2); lcd.print("Suhu 2: "); lcd.print(C2); lcd.print(" C");
12                break;
13            case 1:
14                lcd.setCursor(0,0); lcd.print("Oven"); lcd.setCursor(19,0); lcd.print(wipistat);
15                if(OvenState){
16                    lcd.setCursor(0, 1); lcd.print("Suhu 1: "); lcd.print(C3); lcd.print(" C");
17                    lcd.setCursor(0, 2); lcd.print("Suhu 2: "); lcd.print(C4); lcd.print(" C");
18                    lcd.setCursor(0, 3); lcd.print("Suhu 3: "); lcd.print(C5); lcd.print(" C");
19                    }else{lcd.setCursor(0, 1); lcd.print("Oven isn't Running");}
20                break;
21            case 2:
22                lcd.setCursor(0, 0); lcd.print("DHT21_OVEN "); lcd.setCursor(19,0); lcd.print(wipistat);
23                if(OvenState){
24                    lcd.setCursor(0, 1); lcd.print("Suhu DHT : "); lcd.print(Suhu_DHT_Oven);
25                    lcd.setCursor(0, 2); lcd.print("Humid DHT : "); lcd.print(Kelompokan_DHT_Oven);
26                    }else{lcd.setCursor(0, 1); lcd.print("Oven isn't Running");}
27                break;
28            case 3:
29                lcd.setCursor(0, 0); lcd.print("DHT21_BATTERY "); lcd.setCursor(19,0); lcd.print(wipistat);
30                lcd.setCursor(0, 1); lcd.print("Suhu DHT : "); lcd.print(Suhu_DHT_Battery);
31                lcd.setCursor(0, 2); lcd.print("Humid DHT : "); lcd.print(Kelompokan_DHT_Battery);
32                break;
33            case 4:
34                lcd.setCursor(0, 0); lcd.print("Panel Utama"); lcd.setCursor(19,0); lcd.print(wipistat);
35                lcd.setCursor(0, 1); lcd.print("Volt : "); lcd.print(voltage); lcd.print(" V ");
36                lcd.setCursor(0, 2); lcd.print("Current : "); lcd.print(current); lcd.print(" A ");
37                break;
38            case 5:
39                lcd.setCursor(0, 0); lcd.print("Radiasi"); lcd.setCursor(19,0); lcd.print(wipistat);
40                lcd.setCursor(0,1); lcd.print("Daya : "); lcd.print(DayaRasa); lcd.print(" W ");
41                lcd.setCursor(0,2); lcd.print("BusVoltage"); lcd.print("V"); lcd.print(Current_mA); lcd.print("A");
42                lcd.setCursor(0,3); lcd.print("P/A : "); lcd.print(Solar_Radiation); lcd.print(" M/W*2");
43                break;
44            }
45            lcdPage++;
46            if (lcdPage >= totalLcdPages) { lcdPage = 0; }
47        }
48        #pragma endregion
49        break;

```

```

1 case Suhu_Battery:
2     lcd.setCursor(0,0); lcd.print("Battery"); lcd.setCursor(19,0); lcd.print(wipistat);
3     lcd.setCursor(0, 1); lcd.print("Suhu 1: "); lcd.print(C1); lcd.print(" C");
4     lcd.setCursor(0, 2); lcd.print("Suhu 2: "); lcd.print(C2); lcd.print(" C");
5     break;
6
7 case Suhu_Oven:
8     lcd.setCursor(0,0); lcd.print("Oven"); lcd.setCursor(19,0); lcd.print(wipistat);
9     if(OvenState){
10         lcd.setCursor(0, 1); lcd.print("Suhu 1: "); lcd.print(C3); lcd.print(" C");
11         lcd.setCursor(0, 2); lcd.print("Suhu 2: "); lcd.print(C4); lcd.print(" C");
12         lcd.setCursor(0, 3); lcd.print("Suhu 3: "); lcd.print(C5); lcd.print(" C");
13     }else{lcd.setCursor(0, 1); lcd.print("Oven isn't Running!");}
14     break;
15
16 case DHT21_OVEN:
17     lcd.setCursor(0, 0); lcd.print("DHT21_OVEN "); lcd.setCursor(19,0); lcd.print(wipistat);
18     if(OvenState){
19         lcd.setCursor(0, 1); lcd.print("Suhu DHT : "); lcd.print(Suhu_DHT_Oven);
20         lcd.setCursor(0, 2); lcd.print("Humid DHT : "); lcd.print(Kelambapan_DHT_Oven);
21     }else{lcd.setCursor(0, 1); lcd.print("Oven isn't Running!");}
22     break;
23
24 case DHT21_BATTERY:
25     lcd.setCursor(0, 0); lcd.print("DHT21_BATTERY "); lcd.setCursor(19,0); lcd.print(wipistat);
26     lcd.setCursor(0, 1); lcd.print("Suhu DHT : "); lcd.print(Suhu_DHT_Battery);
27     lcd.setCursor(0, 2); lcd.print("Humid DHT : "); lcd.print(Kelambapan_DHT_Battery);
28     break;
29
30 case Panel_Utama:
31     lcd.setCursor(0, 0); lcd.print("Panel Utama"); lcd.setCursor(19,0); lcd.print(wipistat);
32     lcd.setCursor(0, 1); lcd.print("Volt : "); lcd.print(voltage); lcd.print(" V ");
33     lcd.setCursor(0, 2); lcd.print("Current : "); lcd.print(current); lcd.print(" A ");
34     break;
35
36 case Radiasi:
37     lcd.setCursor(0, 0); lcd.print("Radiasi"); lcd.setCursor(19,0); lcd.print(wipistat);
38     lcd.setCursor(0,1); lcd.print("Daya : "); lcd.print(Dayalna); lcd.print(" W ");
39     lcd.setCursor(0,2); lcd.print("BusVoltage"); lcd.print("V"); lcd.print(Current_mA); lcd.print("A");
40     lcd.setCursor(0,3); lcd.print("P/A : "); lcd.print(Solar_Radiation); lcd.print(" W/m^2");
41     break;
42
43 case Oven_Run:
44     lcd.setCursor(0,0); lcd.print("Run Oven?");
45     lcd.setCursor(0,1); lcd.print("C to Run");
46     lcd.setCursor(0,2); lcd.print("0 Back");
47     OvenRunning = true;
48     break;
49
50 case Oven_Stop:
51     lcd.setCursor(0,0); lcd.print("Stop Oven?");
52     lcd.setCursor(0,1); lcd.print("C to Stop");
53     lcd.setCursor(0,2); lcd.print("0 Back");
54     OvenRunning = false;
55     break;
56
57 case Oven_Running:
58     lcd.setCursor(0,0); lcd.print("Oven is Running!");
59     break;
60
61 case Oven_Stopping:
62     lcd.setCursor(0,0); lcd.print("Oven Stop Running!");
63     break;
64
65 case Restart:
66     lcd.setCursor(0,0); lcd.print("Restarting..");
67     delay(1000);
68     ESP.restart();
69     break;
70
71 case LCD_Restart:
72     lcd.init();
73     lcd.backlight();
74     lcd.clear();
75     layer = LastLayer;
76     break;
77
78 case zzz:
79     break;
80 }
81 lcdReset = layer;
82 }
83 }

```

Gambar 4. 25 Program Interface

Bagian dari program yang menampilkan interface kepada user melalui LCD 20x4 dan Keypad matrix 4x4 sebagai perangkat interaktif. Terdapat menu Home sebagai menu utama yang bisa diakses dengan menekan tombol '0' pada keypad, tekan tombol '1' untuk mengakses menu Suhu_Battery, tekan tombol '2' untuk mengakses menu Suhu_Oven, tekan tombol '3' untuk mengakses menu DHT21_OVEN, tekan tombol '4' untuk mengakses menu DHT21_BATTERY, tekan tombol '5' untuk mengakses menu Panel Utama, tekan tombol '6' untuk mengakses menu Radiasi, tekan tombol 'A' untuk mengakses menu Changing, tekan tombol 'B' untuk mengakses menu Run atau stop pengering, tekan tombol '*' untuk masuk mode sleep interface, tekan tombol '#' untuk reset LCD, tekan tombol 'D' untuk restart ESP32. Penjelasan dari masing-masing menu antara lain. Program untuk interface pada LCD dapat dilihat pada Gambar 4. 25.

4.2.1.6.1 Home

Menu ini merupakan menu utama yang ditampilkan saat sistem pertamakali booting. Pada saat sedang berada di menu lain user dapat kembali ke menu Home dengan menekan tombol '0' pada keypad.

4.2.1.6.2 Changing

Menu ini akan menampilkan semua menu pembacaan sensor secara bergantian setiap 1 detik dengan urutan Suhu Baterai – Suhu Pengering – DHT21 Pengering – DHT21 Ruangan – Pembacaan arus, tegangan dan daya Panel utama – Radiasi matahari. Menu ini dapat diakses dengan menekan tombol 'A' pada keypad.

4.2.1.6.3 Suhu_Battery



Gambar 4. 26 Tampilan LCD pada Menu Suhu Battery

Menu ini menampilkan hasil pembacaan suhu pada baterai yang di ukur oleh 2 thermocouple. Menu ini dapat diakses dengan menekan tombol '1' pada keypad. Gambar tampilan menu suhu battery dapat dilihat pada Gambar 4. 26.

4.2.1.6.4 Panel Utama



Gambar 4. 27 Tampilan LCD pada Menu Panel Utama

Menu ini menampilkan hasil pembacaan dari Arus, Tegangan dan perhitungan daya yang di ukur oleh sensor tegangan dan ACS712 yang terhubung dengan panel utama. Menu ini dapat diakses dengan menekan tombol '5' pada keypad. Gambar tampilan menu panel utama dapat dilihat pada Gambar 4. 27.

4.2.1.6.5 Radiasi



Gambar 4. 28 Tampilan LCD pada Menu Radiasi

Menu ini menampilkan hasil pembacaan daya dan perhitungan radiasi yang di ukur oleh sensor INA219 yang terhubung dengan panel 10w. Menu ini dapat diakses dengan menekan tombol '6' pada keypad. Gambar tampilan menu radiasi dapat dilihat pada Gambar 4. 28.

4.2.1.6.6 Restart

Menu ini adalah menu untuk merestart ESP32. Menu ini dapat diakses dengan menekan tombol 'D' pada keypad.

4.2.1.6.7 LCD Restart

Menu ini adalah menu untuk mereset koneksi LCD ke ESP32 apabila terjadi error komunikasi. Menu ini dapat diakses dengan menekan tombol '#' pada keypad.

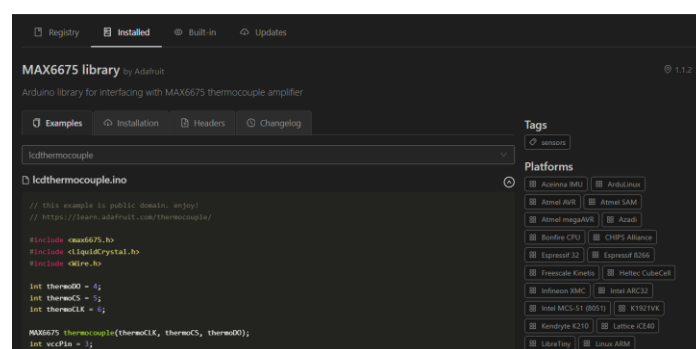
4.2.1.6.8 LCD Sleep

Menu ini adalah menu untuk mode sleep LCD. Menu ini dapat diakses dengan menekan tombol '*' pada keypad.

4.2.2 Pembacaan Sensor

4.2.2.1 Thermocouple & Max6675

Pembacaan suhu pada thermocouple yang di baca oleh MAX6675 menggunakan MAX6675 library oleh Adafruit untuk mempermudah proses pembuatan program. Library ini dipanggil dengan menuliskan #Include <max6675.h> pada program. Library untuk MAX6675 dapat dilihat pada Gambar 4. 29.



Gambar 4. 29 Library MAX6675

Karena MAX6675 berkomunikasi menggunakan protokol SPI maka dibutuhkan pin Digital Out, pin Clock dan pin Chip Select yang di program sedemikian rupa menyesuaikan dengan ketersediaan pin pada ESP32. Deklarasi data untuk pembacaan dan perhitungan yang berkaitan dengan suhu dari thermocouple juga di tulis di bagian ini. Kode pembacaan hasil data thermocouple dapat dilihat pada Gambar 4. 30.

```

1 //MAX6675
2 #define thermoD0 34
3 #define thermoCS1 19
4 #define thermoCS2 18
5 #define thermoCS3 17
6 #define thermoCS4 16
7 #define thermoCS5 33
8 #define thermoCLK 23
9 MAX6675 thermocouple1(thermoCLK, thermoCS1, thermoD0);
10 MAX6675 thermocouple2(thermoCLK, thermoCS2, thermoD0);
11 MAX6675 thermocouple3(thermoCLK, thermoCS3, thermoD0);
12 MAX6675 thermocouple4(thermoCLK, thermoCS4, thermoD0);
13 MAX6675 thermocouple5(thermoCLK, thermoCS5, thermoD0);
14 float C1 = 0;
15 float C2 = 0;
16 float C3 = 0;
17 float C4 = 0;
18 float C5 = 0;
19 float Avg_Bat = 0;
20 float Avg_Ov = 0;
21 float Delta_Bat = 0;
22 float DeltaOldBat = 0;
23 float Delta_Ov = 0;
24 float DeltaOldOv = 0;
25 bool FirstDataBat = true;
26 bool FirstDataOv = true;

```

Gambar 4. 30 Kode Pembacaan Hasil Data Thermocouple

Pembacaan temperatur dilakukan secara berkala pada bagian void loop() dengan memanggil masing-masing modul MAX6675 menggunakan fungsi yang tersedia pada *library*. ESP32 membaca data dari dua thermocouple tipe K sesuai dengan urutan program, kemudian menghitung temperatur rata-rata pasir berdasarkan hasil pembacaan kedua sensor. Data temperatur tersebut selanjutnya ditampilkan pada LCD dan dikirimkan menuju *Google Spreadsheet* untuk keperluan pencatatan dan analisis. Program untuk perhitungan rata-rata suhu thermocouple dapat dilihat pada Gambar 4. 31.

```

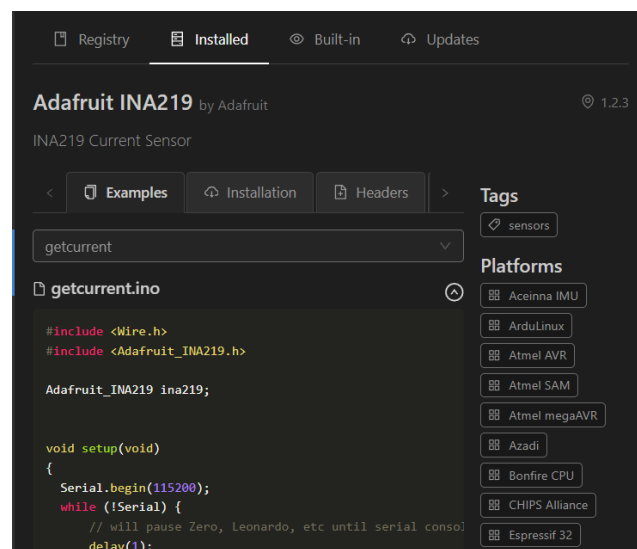
1 void Thermo_Battery(){
2     C1 = thermocouple5.readCelsius();
3     C2 = thermocouple4.readCelsius();
4     Avg_Bat = ((C1+C2)/2);
5 }
6 void Thermo_Oven(){
7     C3 = thermocouple3.readCelsius();
8     C4 = thermocouple2.readCelsius();
9     C5 = thermocouple1.readCelsius();
10    Avg_Ov = ((C3+C4+C5)/3);
11 }

```

Gambar 4. 31 Kode Perhitungan Rata-Rata Suhu Thermocouple

4.2.2.2 INA219

Pembuatan program untuk membaca Tegangan dan Arus pada INA219 menggunakan library Adafruit INA219 untuk mempermudah proses pemrograman. Library yang digunakan untuk INA219 dapat dilihat pada Gambar 4. 32.



Gambar 4. 32 Library INA219

Pendeklarasian data yang akan digunakan dalam proses pembacaan dan perhitungan tegangan, arus dan daya dari sensor INA219 dengan kode program yang dapat dilihat pada Gambar 4. 33.



Gambar 4. 33 Deklarasi Data Sensor INA219

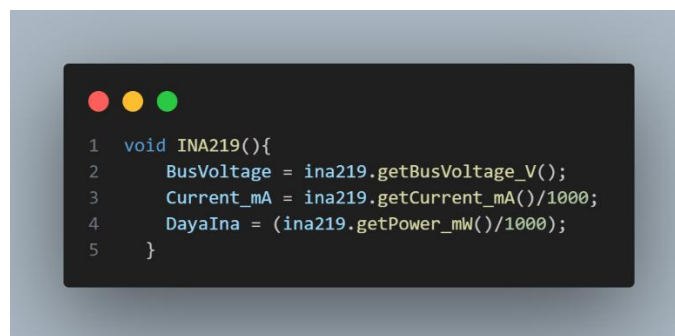
Selanjutnya library perlu memanggil dan memastikan bahwa sensor terhubung dan terdeteksi oleh mikrokontroler pada void setup untuk memastikan

pembacaan data berhasil dengan kode program yang dapat dilihat pada Gambar 4. 34.



Gambar 4. 34 Program Pengecekan Koneksi INA219

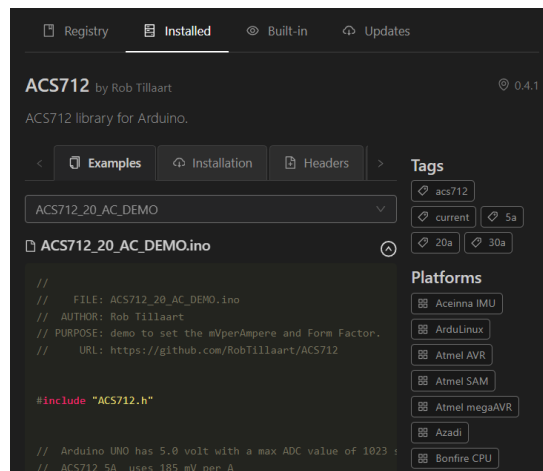
Pembacaan sensor ini dilakukan dengan memanggil perintah baca sensor dari library dengan kode program yang dapat dilihat pada Gambar 4. 35



Gambar 4. 35 Program Memanggil Perintah Baca Semsor dari Library INA219

4.2.2.3 ACS712

Pembuatan program untuk membaca arus oleh sensor AC712 menggunakan library ACS712 oleh Rob Tillaart untuk mempermudah proses pembuatan program. Library yang digunakan untuk ACS712 dapat dilihat pada Gambar 4. 36.



Gambar 4. 36 Library ACS712

Library perlu mengetahui Pin yang digunakan untuk membaca sinyal analog dari sensor, tegangan Vcc pada sensor, resolusi ADC mikrokontroller dan rating arus dari sensor tersebut. Kode untuk define pin untuk sensor ACS712 dapat dilihat pada Gambar 4. 37.



Gambar 4. 37 Define Pin ACS712

Karena pembacaan langsung dari ADC menghasilkan data yang selalu berfluktuasi karena sifat sinyal analog rentan terhadap gangguan (noise). Data pembacaan dibaca sampai dengan 200 sampel lalu dirata-ratakan untuk mendapatkan pembacaan yang stabil. Pembacaan pada sensor dilakukan dengan menggunakan perintah baca sensor dari library. Program pembacaan sensor ACS712 dapat dilihat pada Gambar 4. 38.



Gambar 4. 38 Kode Pembacaan Sensor ACS712

4.2.2.4 Voltage Sensor

Pembacaan sensor tegangan dilakukan dengan membaca tegangan pada keluaran voltage divider dengan range 0V-5V yang mewakili tegangan tinggi 0V-105V. Pin yang digunakan untuk pembacaan adalah A0. Define pin untuk pembacaan tegangan dapat dilihat pada Gambar 4. 39.



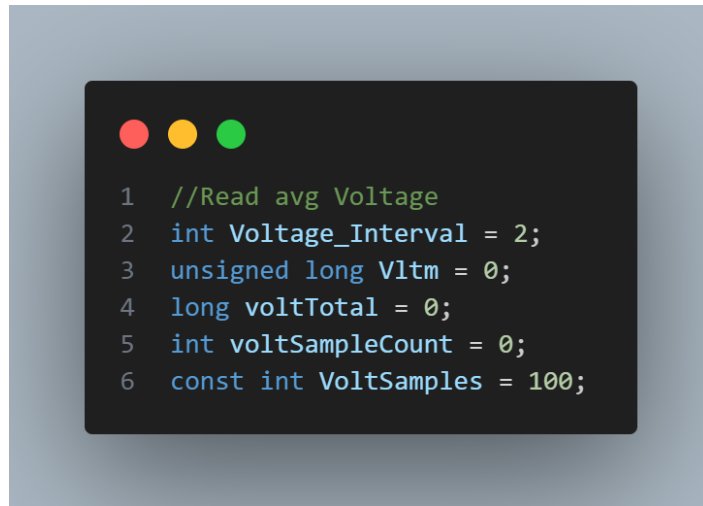
Gambar 4. 39 Define Pin A0

Lalu pin di inisiasi pada void loop supaya pin menjadi mode input dan dapat membaca sinyal analog dari sensor dengan kode program yang dapat dilihat pada Gambar 4. 40.



Gambar 4. 40 Inisiasi Pin pada Void Loop

Karena pembacaan langsung dari ADC menghasilkan data yang selalu berfluktuasi karena sifat sinyal analog rentan terhadap gangguan (noise). Data pembacaan dibaca sampai dengan 200 sampel lalu dirata-ratakan untuk mendapatkan pembacaan yang stabil. Program untuk pembacaan tegangan dapat dilihat pada Gambar 4. 41.



Gambar 4. 41 Kode Pembacaan Sensor

Pembacaan sensor tegangan dilakukan menggunakan fungsi `analogRead()` pada pin A0 Arduino Nano. Untuk mengurangi fluktuasi sinyal, pembacaan dilakukan sebanyak 100 sampel, kemudian dihitung nilai rata-ratanya. Nilai ADC tersebut dikonversi menjadi tegangan pada pin menggunakan persamaan:

$$V_{\text{pin}} = \frac{\text{ADC} \times V_{\text{referensi}}}{1023}$$

Keterangan:

V_{pin} = tegangan pada pin analog Arduino Nano (V);

ADC = nilai rata-rata hasil pembacaan ADC;

$V_{\text{referensi}}$ = tegangan referensi ADC, yaitu sekitar 5 V; dan

1023 = nilai maksimum ADC 10-bit Arduino Nano.

Karena rangkaian pembagi tegangan memiliki rasio 1:21, tegangan panel dihitung menggunakan:

$$V_{\text{panel}} = V_{\text{pin}} \times 21$$

Hasil pembacaan sensor kemudian dibandingkan dengan multimeter sebagai alat ukur referensi. Selisih hasil pengukuran tersebut digunakan untuk menentukan faktor koreksi pada program agar pembacaan sensor mendekati nilai alat ukur referensi. Kode program untuk konversi tegangan dapat dilihat pada Gambar 4. 42.

```

1 void Read_Voltage(){
2   if (millis() - Vltm >= Voltage_Interval) {
3     voltTotal += analogRead(Vin);
4     voltSampleCount++;
5     Vltm = millis();
6
7     if (voltSampleCount >= VoltSamples) {
8       avgAdc = voltTotal / (float)VoltSamples;
9       Vp = (avgAdc * 4.98) / 1024.0;
10      Vreal = Vp / 0.0484;
11
12      // reset for the next batch
13      voltTotal = 0;
14      voltSampleCount = 0;
15    }
  }
}

```

Gambar 4. 42 Kode konversi tegangan pada pin ke tegangan real

4.2.3 Program

4.2.3.1 Komunikasi Mikrokontroller

Komunikasi antara ESP32 dan Arudino Nano bertujuan untuk saling mengirim data sehingga masing-masing mikrokontroller dapat memproses data yang di terima dan memberikan keluaran berdasarkan data tersebut. Komunikasi dilakukan dengan menggunakan protokol komunikasi UART (Universal Asynchronous Receiver-Transmitter).

4,2.3.2 Program Arduino Nano

Arduino Nano digunakan untuk membaca tegangan panel melalui rangkaian pembagi tegangan dan arus panel melalui sensor ACS712. Hasil pembacaan kedua sensor kemudian dikirimkan menuju ESP32 melalui komunikasi UART menggunakan *library* SoftwareSerial.

Pengiriman data dilakukan secara berkala setiap 500 milidetik. Data tegangan dan arus dikirimkan dalam satu rangkaian data dengan karakter pemisah agar dapat dipisahkan dan diolah oleh ESP32. Indikator LED_BUILTIN digunakan untuk menunjukkan berlangsungnya proses komunikasi antara Arduino Nano dan ESP32. Kode program untuk pengiriman data oleh Arduino Nano dapat dilihat pada Gambar 4. 43.

```

1 void Transmit(){
2   // Serial.print("Vp/Vreal/I : ");
3   // Serial.print(Vp);
4   // Serial.print("/");
5   Serial2.print(Vreal);
6   Serial2.print("/");
7   Serial2.print(mA);
8   Serial2.print("/");
9   Serial2.print(h);
10  Serial2.print("/");
11  Serial2.print(t);
12  Serial2.print("/");
13  Serial2.print(h1);
14  Serial2.print("/");
15  Serial2.println(t1);
16 }
17 void Receive(){
18   if (Serial2.available()) {
19     digitalWrite(LED_BUILTIN, HIGH);
20     Serial.println("Mashooookk");
21     String data = Serial2.readStringUntil('\n');
22     data.trim();
23
24     int c1 = data.indexOf('/');
25     int c2 = data.indexOf('/', c1 + 1);
26     int c3 = data.indexOf('/', c2 + 1);
27
28     if (c1 > 0 && c2 > c1 && c3 > c2) {
29       Serial.println("Jadeee");
30       C1 = data.substring(0, c1).toFloat();
31       C2 = data.substring(c1 + 1, c2).toFloat();
32       C3 = data.substring(c2 + 1, c3).toFloat();
33       OvenSerial = data.substring(c3 + 1).toInt();
34       // Serial.print(C1);
35       // Serial.print("/");
36       // Serial.print(C2);
37       // Serial.print("/");
38       // Serial.print(C3);
39     }
40   }
41   else{digitalWrite(LED_BUILTIN, LOW);}
42   if(OvenSerial == 0){OvenState = false;}
43   else{OvenState = true;}
44 }

```

Gambar 4. 43 Kode Pengiriman Data Arduino Nano

4.2.3.4 Program ESP32

Program komunikasi UART pada ESP32 digunakan untuk menerima data tegangan dan arus panel surya dari Arduino Nano. Data diterima dalam bentuk rangkaian karakter dengan tanda garis miring sebagai pemisah, kemudian dipisahkan dan dikonversi menjadi data numerik untuk proses pengolahan.

ESP32 menghitung daya panel berdasarkan hasil perkalian tegangan dan arus. Data tersebut digabungkan dengan hasil pembacaan dua thermocouple pasir

dan estimasi iradiasi matahari. Seluruh hasil pengukuran ditampilkan melalui LCD 20×4 dan dikirimkan menuju *Google Spreadsheet* setiap lima menit untuk keperluan pencatatan dan analisis.

Indikator LED_BUILTIN pada ESP32 digunakan untuk menunjukkan berlangsungnya penerimaan data dari Arduino Nano melalui komunikasi UART. Kode program untuk penerimaan data oleh ESP32 dapat dilihat pada Gambar 4. 44.



```

1 void Receive(){
2   if (Serial2.available()) {
3     digitalWrite(LED_BUILTIN, HIGH);
4     String data = Serial2.readStringUntil('\n');
5     data.trim();
6
7     int c1 = data.indexOf('/');
8     int c2 = data.indexOf('/', c1 + 1);
9     int c3 = data.indexOf('/', c2 + 1);
10    int c4 = data.indexOf('/', c3 + 1);
11    int c5 = data.indexOf('/', c4 + 1);
12
13    if (c1 > 0 && c2 > c1 && c3 > c2 && c4 > c3 && c5 > c4) {
14      voltage      = data.substring(0, c1).toFloat();
15      current      = data.substring(c1 + 1, c2).toFloat();
16      Kelembapan_DHT_Oven = data.substring(c2 + 1, c3).toFloat();
17      Suhu_DHT_Oven    = data.substring(c3 + 1, c4).toFloat();
18      Kelembapan_DHT_Battery = data.substring(c4 + 1, c5).toFloat();
19      Suhu_DHT_Battery  = data.substring(c5 + 1).toFloat();
20    } else {
21      Serial.print("Data format tidak valid: ");
22      Serial.println(data);
23    }
24    digitalWrite(LED_BUILTIN, LOW);
25  }
26  void Transmit(){
27    //Transmit
28    Serial2.print(C3);
29    Serial2.print("/");
30    Serial2.print(C4);
31    Serial2.print("/");
32    Serial2.print(C5);
33    Serial2.print("/");
34    Serial2.println(OvenState);
35  }

```

Gambar 4. 44 Kode Penerimaan Data ESP32

4.2.4 Perhitungan

4.2.4.1 Delta Suhu

Laju perubahan temperatur menunjukkan perubahan temperatur rata-rata pasir pada setiap interval pencatatan data. Nilai tersebut dihitung dari selisih temperatur rata-rata saat ini dan temperatur rata-rata sebelumnya, kemudian dibagi dengan interval waktu pengukuran.

$$r_T = \frac{T_i - T_{i-1}}{t_i - t_{i-1}}$$

Keterangan:

r_T = laju perubahan temperatur (°C/menit);

T_i = temperatur rata-rata pada pengukuran saat ini (°C);

T_{i-1} = temperatur rata-rata pada pengukuran sebelumnya (°C); dan

$t_i - t_{i-1}$ = interval waktu pencatatan data (menit).

Karena data dikirimkan menuju *Google Spreadsheet* setiap lima menit, interval waktu yang digunakan adalah lima menit. Nilai positif menunjukkan kenaikan temperatur, sedangkan nilai negatif menunjukkan penurunan temperatur. Program untuk perhitungan delta suhu dapat dilihat pada Gambar 4. 45.



```

1  //Battery
2  if(FirstDataBat){
3      DeltaOldBat = Avg_Bat;
4      FirstDataBat = false;
5  }
6  else{
7      Delta_Bat = ((Avg_Bat - DeltaOldBat)/(interval/60000));
8  }
9  DeltaOldBat = Avg_Bat;
10
11 //Oven
12 if(FirstDataOv){
13     DeltaOldOv = Avg_Ov;
14     FirstDataOv = false;
15 }
16 else{
17     Delta_Ov = ((Avg_Ov - DeltaOldOv)/(interval/60000));
18 }
19 DeltaOldOv = Avg_Ov;

```

Gambar 4. 45 Kode Perhitungan Delta Suhu

4.2.4.2 Radiasi Matahari

Perhitungan Radiasi Matahari didapatkan dari data mentah hasil pembacaan arus, tegangan dan daya dari panel 10W yang dibaca oleh sensor INA219. Konversi dari daya panel menjadi Radiasi Matahari menggunakan rumus

Daya Panel = pembacaan daya oleh INA219

Luas panel = 0.061425

Efisiensi panel= 17.46

Panel surya referensi digunakan untuk mengamati kecenderungan perubahan radiasi matahari selama pengujian. Luas permukaan panel diperoleh berdasarkan hasil pengukuran panjang dan lebar panel. Berdasarkan ukuran 0,19 m dan 0,32 m, luas permukaan panel adalah:

$$A = 0,19 \times 0,32 = 0,0608 \text{ m}^2$$

Daya keluaran panel referensi dihitung berdasarkan hasil pembacaan tegangan dan arus oleh INA219. Estimasi iradiasi matahari dihitung menggunakan:

$$G = \frac{P}{A\eta}$$

Keterangan:

G = estimasi iradiasi matahari (W/m^2);

P = daya keluaran panel referensi (W);

A = luas permukaan panel referensi (m^2); dan

η = efisiensi panel dalam bentuk desimal.

Jika efisiensi panel sebesar 17,46%, nilai yang digunakan dalam perhitungan adalah:

$$\eta = 0,1746$$

Metode ini hanya menghasilkan estimasi karena keluaran panel referensi juga dipengaruhi oleh temperatur modul, karakteristik beban resistor, sudut pemasangan, kondisi permukaan panel, dan titik operasi panel. Pengukuran tidak dikalibrasi menggunakan piranometer atau *solar power meter*. Oleh karena itu, hasil perhitungan digunakan untuk menunjukkan kecenderungan perubahan radiasi, bukan sebagai nilai iradiasi absolut.

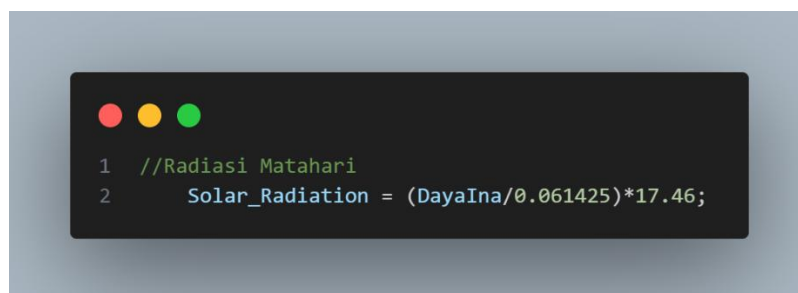
Gambar pengukuran perhitungan luas panel dapat dilihat pada Gambar 4.

46.



Gambar 4. 46 Pengukuran Luas Panel

Lalu efisiensi panel didapatkan dari datasheet panel. Namun menggunakan besaran efisiensi dari datasheet masih belum menghasilkan pembacaan yang akurat. Untuk mendapatkan nilai efisiensi yang real diperlukan nilai radiasi matahari pada saat pembacaan daya panel dilakukan, perhitungan menggunakan rumus pada program perhitungan dapat dilihat pada Gambar 4. 47..



Gambar 4. 47 Kode Perhitungan Radiasi Sinar Matahari

4.2.5 Data Logging

Upload data logging ke spreadsheet terbagi menjadi data pada baterai pasir dan data pada pengering. Spreadsheet dibedakan menjadi 2 untuk memudahkan proses analisa data. Spreadsheet baterai pasir akan selalu mengupload data setiap 5 menit apabila sistem sudah di hidupkan dan terhubung dengan wifi sedangkan spreadsheet pengering hanya akan melakukan proses upload apabila perintah untuk

menjalankan pengering diberikan oleh user. Proses data login juga mengupload header data ke masing-masing spreadsheet setiap kali ESP32 di hidupkan atau di restart.

4.2.5.1 SpreadSheet Boot

Spreadsheet Boot akan mengupload header yang menampilkan data apa saja yang di upload ke masing-masing spreadsheet. Pada spreadsheet baterai header data bertuliskan Jam, Suhu 1, Suhu 2, Suhu rerata, Delta T, Tegangan, Arus, Daya, Radiasi, V INA, A INA, P INA, Suhu DHT, Kelembaban.

Jam	Suhu 1	Suhu 2	Suhu rerata	Delta T	Tegangan	Arus	Daya	Radiasi	V INA	A INA	P INA	Suhu DHT	Kelembaban
03 Sep 2026, 17:58	19.25	27.5	23.375	0	0.02	-24.2	-0.484	0	0	0	0	29.1	71.3

Pada Spreadsheet pengering header data berisikan Jam, Suhu 1, Suhu 2, Suhu 3, Suhu rerata, Delta T, Suhu DHT, Kelembaban.

Jam	Suhu 1	Suhu 2	Suhu 3	Suhu Rerata	Delta T	Suhu DHT	Kelembaban
03 Sep 2026, 17:58	26.5	26.25	26.75	26.5	0	28.5	95.4

Header diupload setiap booting bertujuan untuk memberikan batas pada spreadsheet dan juga menjadi indikasi jika terjadi gangguan pada sistem yang mengakibatkan sistem restart tanpa ada perintah dari user.

Kode program untuk spreadsheet boot dapat dilihat pada Gambar 4. 48

```

1  #!/usr/bin/perl
2
3  use strict;
4  use warnings;
5
6  use Getopt::Std;
7  use IO::Socket;
8  use IO::Socket::SSL;
9  use IO::Socket::INET;
10
11 use vars qw($opt_h $opt_v $opt_p $opt_s $opt_t $opt_u $opt_l $opt_f $opt_m $opt_n $opt_o $opt_e $opt_c $opt_d $opt_r $opt_b $opt_k $opt_g $opt_j $opt_i $opt_a $opt_x $opt_y $opt_z $opt_0 $opt_1 $opt_2 $opt_3 $opt_4 $opt_5 $opt_6 $opt_7 $opt_8 $opt_9 $opt_10 $opt_11 $opt_12 $opt_13 $opt_14 $opt_15 $opt_16 $opt_17 $opt_18 $opt_19 $opt_20 $opt_21 $opt_22 $opt_23 $opt_24 $opt_25 $opt_26 $opt_27 $opt_28 $opt_29 $opt_30 $opt_31 $opt_32 $opt_33 $opt_34 $opt_35 $opt_36 $opt_37 $opt_38 $opt_39 $opt_40 $opt_41 $opt_42 $opt_43 $opt_44 $opt_45 $opt_46 $opt_47 $opt_48 $opt_49 $opt_50 $opt_51 $opt_52 $opt_53 $opt_54 $opt_55 $opt_56 $opt_57 $opt_58 $opt_59 $opt_60 $opt_61 $opt_62 $opt_63 $opt_64 $opt_65 $opt_66 $opt_67 $opt_68 $opt_69 $opt_70 $opt_71 $opt_72 $opt_73 $opt_74 $opt_75 $opt_76 $opt_77 $opt_78 $opt_79 $opt_80 $opt_81 $opt_82 $opt_83 $opt_84 $opt_85 $opt_86 $opt_87 $opt_88 $opt_89 $opt_90 $opt_91 $opt_92 $opt_93 $opt_94 $opt_95 $opt_96 $opt_97 $opt_98 $opt_99);
12
13 my $host = $opt_h;
14 my $port = $opt_p;
15 my $ssl = $opt_s;
16 my $server = $opt_t;
17 my $user = $opt_u;
18 my $pass = $opt_p;
19 my $key = $opt_k;
20 my $cert = $opt_c;
21 my $ca = $opt_a;
22 my $mode = $opt_m;
23 my $format = $opt_f;
24 my $verbose = $opt_v;
25 my $quiet = $opt_q;
26 my $help = $opt_h;
27 my $version = $opt_v;
28 my $debug = $opt_d;
29 my $retry = $opt_r;
30 my $backlog = $opt_b;
31 my $bind = $opt_g;
32 my $listen = $opt_l;
33 my $accept = $opt_a;
34 my $connect = $opt_c;
35 my $read = $opt_r;
36 my $write = $opt_w;
37 my $close = $opt_c;
38 my $shutdown = $opt_s;
39 my $wait = $opt_w;
40 my $error = $opt_e;
41 my $alarm = $opt_a;
42 my $poll = $opt_p;
43 my $select = $opt_s;
44 my $epoll = $opt_e;
45 my $kqueue = $opt_k;
46 my $pollfd = $opt_p;
47 my $pollfd2 = $opt_p2;
48 my $pollfd3 = $opt_p3;
49 my $pollfd4 = $opt_p4;
50 my $pollfd5 = $opt_p5;
51 my $pollfd6 = $opt_p6;
52 my $pollfd7 = $opt_p7;
53 my $pollfd8 = $opt_p8;
54 my $pollfd9 = $opt_p9;
55 my $pollfd10 = $opt_p10;
56 my $pollfd11 = $opt_p11;
57 my $pollfd12 = $opt_p12;
58 my $pollfd13 = $opt_p13;
59 my $pollfd14 = $opt_p14;
60 my $pollfd15 = $opt_p15;
61 my $pollfd16 = $opt_p16;
62 my $pollfd17 = $opt_p17;
63 my $pollfd18 = $opt_p18;
64 my $pollfd19 = $opt_p19;
65 my $pollfd20 = $opt_p20;
66 my $pollfd21 = $opt_p21;
67 my $pollfd22 = $opt_p22;
68 my $pollfd23 = $opt_p23;
69 my $pollfd24 = $opt_p24;
70 my $pollfd25 = $opt_p25;
71 my $pollfd26 = $opt_p26;
72 my $pollfd27 = $opt_p27;
73 my $pollfd28 = $opt_p28;
74 my $pollfd29 = $opt_p29;
75 my $pollfd30 = $opt_p30;
76 my $pollfd31 = $opt_p31;
77 my $pollfd32 = $opt_p32;
78 my $pollfd33 = $opt_p33;
79 my $pollfd34 = $opt_p34;
80 my $pollfd35 = $opt_p35;
81 my $pollfd36 = $opt_p36;
82 my $pollfd37 = $opt_p37;
83 my $pollfd38 = $opt_p38;
84 my $pollfd39 = $opt_p39;
85 my $pollfd40 = $opt_p40;
86 my $pollfd41 = $opt_p41;
87 my $pollfd42 = $opt_p42;
88 my $pollfd43 = $opt_p43;
89 my $pollfd44 = $opt_p44;
90 my $pollfd45 = $opt_p45;
91 my $pollfd46 = $opt_p46;
92 my $pollfd47 = $opt_p47;
93 my $pollfd48 = $opt_p48;
94 my $pollfd49 = $opt_p49;
95 my $pollfd50 = $opt_p50;
96 my $pollfd51 = $opt_p51;
97 my $pollfd52 = $opt_p52;
98 my $pollfd53 = $opt_p53;
99 my $pollfd54 = $opt_p54;
100 my $pollfd55 = $opt_p55;
101 my $pollfd56 = $opt_p56;
102 my $pollfd57 = $opt_p57;
103 my $pollfd58 = $opt_p58;
104 my $pollfd59 = $opt_p59;
105 my $pollfd60 = $opt_p60;
106 my $pollfd61 = $opt_p61;
107 my $pollfd62 = $opt_p62;
108 my $pollfd63 = $opt_p63;
109 my $pollfd64 = $opt_p64;
110 my $pollfd65 = $opt_p65;
111 my $pollfd66 = $opt_p66;
112 my $pollfd67 = $opt_p67;
113 my $pollfd68 = $opt_p68;
114 my $pollfd69 = $opt_p69;
115 my $pollfd70 = $opt_p70;
116 my $pollfd71 = $opt_p71;
117 my $pollfd72 = $opt_p72;
118 my $pollfd73 = $opt_p73;
119 my $pollfd74 = $opt_p74;
120 my $pollfd75 = $opt_p75;
121 my $pollfd76 = $opt_p76;
122 my $pollfd77 = $opt_p77;
123 my $pollfd78 = $opt_p78;
124 my $pollfd79 = $opt_p79;
125 my $pollfd80 = $opt_p80;
126 my $pollfd81 = $opt_p81;
127 my $pollfd82 = $opt_p82;
128 my $pollfd83 = $opt_p83;
129 my $pollfd84 = $opt_p84;
130 my $pollfd85 = $opt_p85;
131 my $pollfd86 = $opt_p86;
132 my $pollfd87 = $opt_p87;
133 my $pollfd88 = $opt_p88;
134 my $pollfd89 = $opt_p89;
135 my $pollfd90 = $opt_p90;
136 my $pollfd91 = $opt_p91;
137 my $pollfd92 = $opt_p92;
138 my $pollfd93 = $opt_p93;
139 my $pollfd94 = $opt_p94;
140 my $pollfd95 = $opt_p95;
141 my $pollfd96 = $opt_p96;
142 my $pollfd97 = $opt_p97;
143 my $pollfd98 = $opt_p98;
144 my $pollfd99 = $opt_p99;
145 my $pollfd100 = $opt_p100;
146 my $pollfd101 = $opt_p101;
147 my $pollfd102 = $opt_p102;
148 my $pollfd103 = $opt_p103;
149 my $pollfd104 = $opt_p104;
150 my $pollfd105 = $opt_p105;
151 my $pollfd106 = $opt_p106;
152 my $pollfd107 = $opt_p107;
153 my $pollfd108 = $opt_p108;
154 my $pollfd109 = $opt_p109;
155 my $pollfd110 = $opt_p110;
156 my $pollfd111 = $opt_p111;
157 my $pollfd112 = $opt_p112;
158 my $pollfd113 = $opt_p113;
159 my $pollfd114 = $opt_p114;
160 my $pollfd115 = $opt_p115;
161 my $pollfd116 = $opt_p116;
162 my $pollfd117 = $opt_p117;
163 my $pollfd118 = $opt_p118;
164 my $pollfd119 = $opt_p119;
165 my $pollfd120 = $opt_p120;
166 my $pollfd121 = $opt_p121;
167 my $pollfd122 = $opt_p122;
168 my $pollfd123 = $opt_p123;
169 my $pollfd124 = $opt_p124;
170 my $pollfd125 = $opt_p125;
171 my $pollfd126 = $opt_p126;
172 my $pollfd127 = $opt_p127;
173 my $pollfd128 = $opt_p128;
174 my $pollfd129 = $opt_p129;
175 my $pollfd130 = $opt_p130;
176 my $pollfd131 = $opt_p131;
177 my $pollfd132 = $opt_p132;
178 my $pollfd133 = $opt_p133;
179 my $pollfd134 = $opt_p134;
180 my $pollfd135 = $opt_p135;
181 my $pollfd136 = $opt_p136;
182 my $pollfd137 = $opt_p137;
183 my $pollfd138 = $opt_p138;
184 my $pollfd139 = $opt_p139;
185 my $pollfd140 = $opt_p140;
186 my $pollfd141 = $opt_p141;
187 my $pollfd142 = $opt_p142;
188 my $pollfd143 = $opt_p143;
189 my $pollfd144 = $opt_p144;
190 my $pollfd145 = $opt_p145;
191 my $pollfd146 = $opt_p146;
192 my $pollfd147 = $opt_p147;
193 my $pollfd148 = $opt_p148;
194 my $pollfd149 = $opt_p149;
195 my $pollfd150 = $opt_p150;
196 my $pollfd151 = $opt_p151;
197 my $pollfd152 = $opt_p152;
198 my $pollfd153 = $opt_p153;
199 my $pollfd154 = $opt_p154;
200 my $pollfd155 = $opt_p155;
201 my $pollfd156 = $opt_p156;
202 my $pollfd157 = $opt_p157;
203 my $pollfd158 = $opt_p158;
204 my $pollfd159 = $opt_p159;
205 my $pollfd160 = $opt_p160;
206 my $pollfd161 = $opt_p161;
207 my $pollfd162 = $opt_p162;
208 my $pollfd163 = $opt_p163;
209 my $pollfd164 = $opt_p164;
210 my $pollfd165 = $opt_p165;
211 my $pollfd166 = $opt_p166;
212 my $pollfd167 = $opt_p167;
213 my $pollfd168 = $opt_p168;
214 my $pollfd169 = $opt_p169;
215 my $pollfd170 = $opt_p170;
216 my $pollfd171 = $opt_p171;
217 my $pollfd172 = $opt_p172;
218 my $pollfd173 = $opt_p173;
219 my $pollfd174 = $opt_p174;
220 my $pollfd175 = $opt_p175;
221 my $pollfd176 = $opt_p176;
222 my $pollfd177 = $opt_p177;

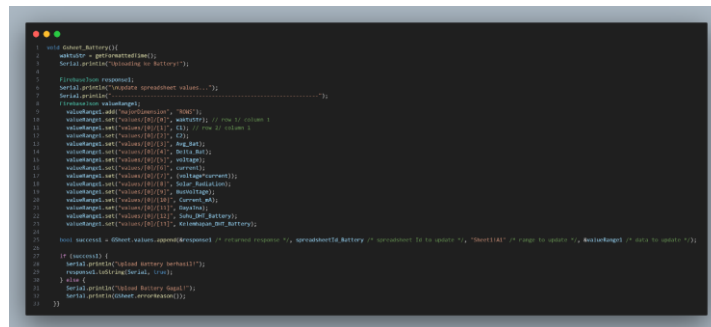
```

Gambar 4. 48 Kode SpreadSheet Boot

4.2.5.2 SpreadSheet Baterai Pasir

Proses pengunggahan data baterai pasir dilakukan setiap lima menit selama sistem beroperasi dan terhubung dengan jaringan Wi-Fi. Data yang diunggah ke *Google Spreadsheet* meliputi waktu aktual yang diperoleh dari *epoch time*, temperatur dari dua thermocouple pada baterai pasir, temperatur rata-rata, laju perubahan temperatur, tegangan, arus, dan daya panel utama, indeks estimasi penyinaran, serta tegangan, arus, dan daya panel referensi 10 Wp.

Kode program spreadsheet baterai pasir dapat dilihat pada gambar 4. 49.



Gambar 4. 49 Kode SpreadSheet Baterai Pasir

4.2.6 Monitoring Menggunakan Blynk

Blynk digunakan sebagai antarmuka monitoring berbasis *Internet of Things* untuk menampilkan kondisi prototipe *Thermal Energy Storage* secara langsung melalui perangkat seluler. Sistem ini memanfaatkan konektivitas Wi-Fi pada ESP32 untuk mengirimkan data hasil pembacaan sensor menuju platform Blynk.

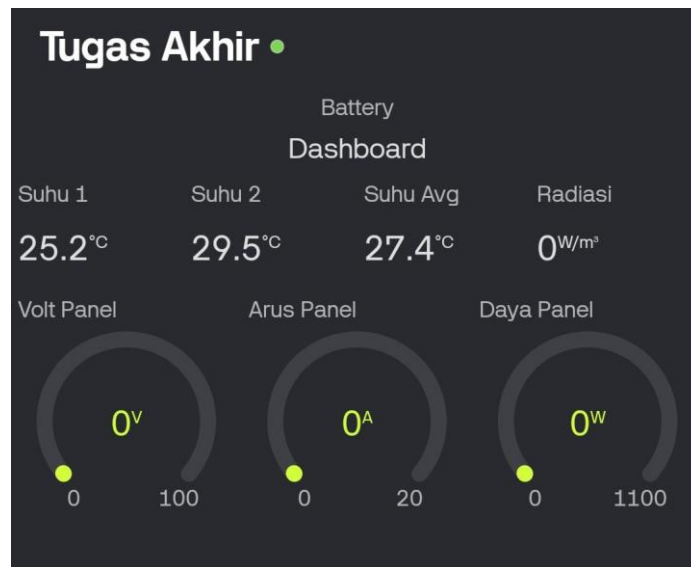
Implementasi Blynk diawali dengan pembuatan *template* dan *datastream* sesuai dengan parameter yang akan ditampilkan. Setiap parameter dihubungkan dengan *virtual pin* yang berbeda agar data dapat dikirimkan dan ditampilkan secara terpisah. *Virtual pin* merupakan jalur pertukaran data antara program pada ESP32 dengan aplikasi Blynk dan tidak mewakili pin fisik pada mikrokontroler. Data dari ESP32 dapat dikirim menuju Blynk menggunakan fungsi `Blynk.virtualWrite()`.

Parameter yang ditampilkan pada *dashboard* Blynk meliputi Suhu 1, Suhu 2, temperatur rata-rata pasir, tegangan panel surya, arus panel surya, daya listrik, dan radiasi matahari. Data tersebut ditampilkan menggunakan *value display*, *gauge*, dan grafik agar perubahan kondisi sistem dapat dipantau berdasarkan waktu.

Pengiriman data menuju Blynk dilakukan secara berkala menggunakan pengatur waktu pada program ESP32. Pengaturan interval diperlukan agar proses pengiriman tidak dilakukan terus-menerus pada setiap perulangan program karena kondisi tersebut dapat mengganggu kestabilan komunikasi. Dokumentasi Blynk merekomendasikan penggunaan pengatur waktu ketika fungsi `Blynk.virtualWrite()` digunakan untuk mengirim data secara berkala.

Blynk dan *Google Spreadsheet* memiliki fungsi yang berbeda. Blynk digunakan untuk memantau kondisi alat secara langsung, sedangkan *Google*

Spreadsheet digunakan untuk menyimpan data pengujian sebagai bahan pengolahan dan analisis. Penggunaan kedua platform tersebut memungkinkan pengguna memantau kondisi prototipe sekaligus memperoleh rekaman data selama pengujian berlangsung. Tampilan interface pada dashboard Blynk dapat dilihat pada Gambar 4. 50.



Gambar 4. 50 Tampilan Blynk