

BAB IV

PEMBUATAN ALAT

4.1 Pembuatan Perangkat Keras

Pembuatan perangkat keras merupakan tahapan awal dalam proses pembangunan sistem pengering biji kopi berbasis mikrokontroler ESP32. Pada tahap ini dilakukan proses perakitan seluruh komponen elektronik dan mekanik sehingga membentuk suatu sistem yang mampu melakukan pengendalian suhu serta *monitoring* kondisi ruang pengering secara *real-time*. Perangkat keras yang digunakan terdiri atas oven listrik sebagai media pengering, ESP32 sebagai pengendali utama, sensor SHT30 sebagai pembaca suhu dan kelembapan relatif, *solid state relay* (SSR-25DAH) sebagai pengendali elemen pemanas, modul HLK-PM01 sebagai modul catu daya, serta OLED sebagai media penampil informasi. Seluruh komponen tersebut diintegrasikan menjadi satu sistem sehingga mampu menjalankan proses pengeringan secara otomatis sesuai dengan suhu *setpoint* yang telah ditentukan serta mengirimkan data hasil pemantauan ke aplikasi Blynk melalui jaringan Wi-Fi.

4.1.1 Alat dan Bahan

Tahap awal dalam pembuatan perangkat keras (*hardware*) adalah menyiapkan seluruh alat dan bahan yang diperlukan selama proses perakitan sistem. Alat yang digunakan berfungsi untuk membantu proses pemotongan, penyambungan, pemasangan, serta pengujian rangkaian. Sedangkan bahan merupakan komponen utama yang digunakan dalam membangun sistem pengering biji kopi. Daftar alat yang digunakan pada penelitian ini disajikan pada Tabel 4-1 berikut.

Tabel 4-1 Alat yang digunakan

No	Nama Alat	Jumlah
1	Multimeter	1 buah
2	Obeng set	1 buah
3	Tang Potong	1 buah

No	Nama Alat	Jumlah
4	Tang Crimping	1 buah
5	Solder	1 buah
6	Bor	1 buah
7	Gerinda	1 buah
8	Laptop	1 buah
9	Jangka Sorong	1 buah
10	Timbangan Digital	1 buah
11	Alat Tulis	1 set
12	Plastik	1 pak
13	Wadah	1 buah

Selain alat, penelitian ini juga memerlukan berbagai bahan dan komponen elektronika sebagai penyusun utama sistem pengering biji kopi. Bahan yang digunakan dipilih berdasarkan fungsi, spesifikasi, serta kesesuaiannya dengan kebutuhan sistem agar seluruh komponen dapat bekerja secara terintegrasi. Daftar bahan yang digunakan dalam penelitian ini disajikan pada Tabel 4-2.

Tabel 4-2 Bahan yang digunakan

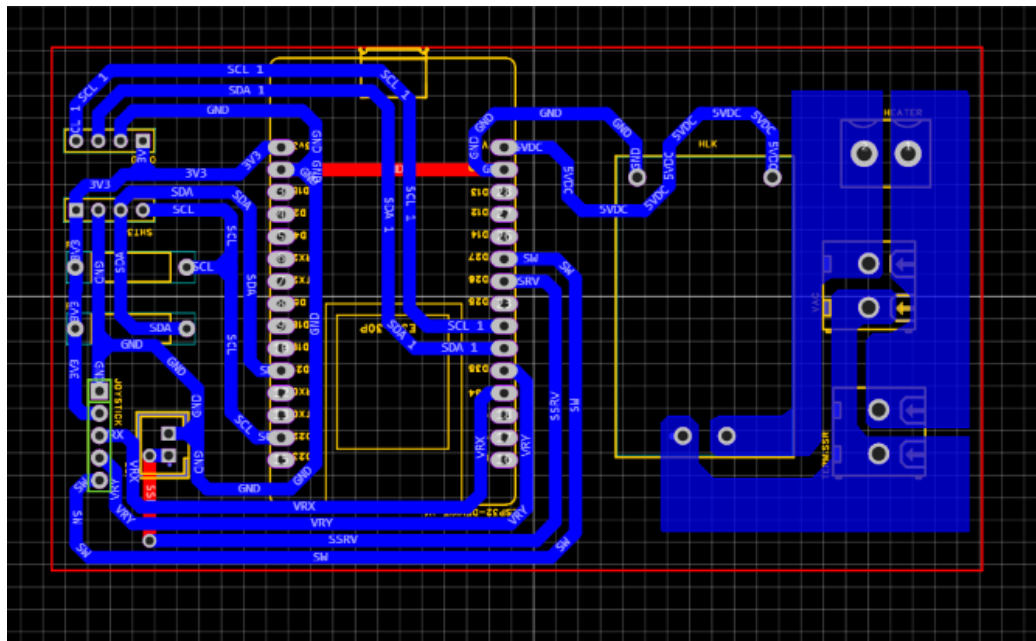
No	Nama Bahan	Jumlah
1	Gelondong Cherry Biji Kopi Robusta	± 3 Kg
2	Oven Listrik	1 buah
3	ESP32 DEVKIT V1	1 buah
4	SSR 25 DA	1 buah
5	Sensor SHT30	1 buah
6	HLKPM-01	1 buah
7	OLED	1 buah
8	Joystick	1 buah
9	Resistor 4,7K	2 buah
10	Pin Header 15 Pin	2 buah
11	JST 4 Pin	2 buah

No	Nama Bahan	Jumlah
12	JST 5 Pin	1 buah
13	Terminal Block 2 Pin	4 buah
14	PCB	1 buah
15	Kabel Jumper	2 buah
16	Box Hitam X4	1 buah
17	Kabel Micro to Type C	1 buah

4.1.2 Pembuatan Perangkat Elektrik

Pembuatan perangkat elektrik dilakukan dengan merangkai seluruh komponen elektronika sesuai dengan rancangan sistem yang telah dibuat. Komponen yang dirangkai meliputi mikrokontroler ESP32, sensor SHT30, modul HLK-PM01, SSR-25DAH, OLED, serta elemen pemanas oven. Seluruh komponen dihubungkan sesuai dengan fungsi masing-masing sehingga membentuk sistem pengendalian dan *monitoring* pada alat pengering biji kopi. Adapun langkah-langkah pembuatan perangkat elektrik adalah sebagai berikut:

1. Perancangan Skematik dan *Layout* PCB



Gambar 4.1 Design *Layout* PCB

Langkah awal dalam pembuatan perangkat elektrik adalah merancang skematik rangkaian dan *layout* PCB menggunakan *software* EasyEDA. Tahap perancangan ini bertujuan untuk menggambarkan hubungan antar komponen elektronika yang digunakan pada sistem pengering biji kopi sehingga setiap komponen dapat dihubungkan sesuai dengan fungsi dan rancangan sistem. Komponen utama yang dirancang meliputi mikrokontroler ESP32 sebagai pengendali utama sistem, sensor SHT30 sebagai pembaca suhu dan kelembapan relatif, modul HLK-PM01 sebagai catu daya, modul SSR-25DAH sebagai pengendali elemen pemanas, OLED sebagai media penampil informasi, serta joystick KY-023 sebagai perangkat masukan untuk memberikan perintah atau pengaturan pada sistem. Selain komponen utama adapun komponen pendukung lainnya seperti resistor, *terminal block*, JST *connector*, dan konektor daya. Hasil perancangan *layout* PCB ditunjukkan pada Gambar 4.1.

Setelah skematik selesai dibuat, dilakukan proses penyusunan *layout* PCB dengan menempatkan setiap komponen sesuai kebutuhan rangkaian, kemudian dilanjutkan dengan pembuatan jalur penghantar (*routing*) untuk menghubungkan seluruh komponen. Pada tahap ini juga dilakukan penyesuaian ukuran papan PCB, pemberian label pada jalur dan pin yang diperlukan, serta pengaturan posisi komponen agar proses pemasangan dan penyolderan menjadi lebih mudah. *Layout* PCB yang disusun secara sistematis menghasilkan rangkaian yang lebih ringkas dan rapi sehingga proses perakitan dapat dilakukan dengan lebih mudah serta risiko kesalahan dalam pengkabelan dapat diminimalkan.

2. Proses Pencetakan dan Penyolderan PCB

Setelah proses perancangan skematik dan *layout* PCB selesai dilakukan, tahap selanjutnya adalah pembuatan PCB berdasarkan desain yang telah dibuat menggunakan *software* EasyEDA.



Gambar 4.2 Pencetakan PCB dengan Mesin Laser

Pola jalur rangkaian dicetak secara langsung pada permukaan PCB menggunakan mesin laser sesuai dengan desain yang telah ditentukan seperti pada Gambar 4.2. Setelah pola rangkaian terbentuk, ukuran papan PCB disesuaikan dengan rancangan yang telah dibuat. Pemotongan dilakukan menggunakan gerinda hingga diperoleh ukuran PCB yang sesuai dengan kebutuhan.

Tahap selanjutnya adalah proses pelarutan (*etching*) untuk membentuk jalur rangkaian pada permukaan PCB. Proses yang ditunjukkan pada Gambar 4.3 ini dilakukan dengan menggunakan larutan *ferric chloride* (FeCl_3) yang dicampurkan dengan air panas. Papan PCB dimasukkan ke dalam larutan tersebut hingga bagian tembaga yang tidak diperlukan terlarut, sedangkan bagian yang digunakan sebagai jalur rangkaian tetap dipertahankan. Setelah proses pelarutan selesai, PCB dikeluarkan dari larutan kemudian dibersihkan dari sisa larutan dan bagian tembaga yang telah terlarut. PCB selanjutnya dikeringkan sebelum dilakukan proses berikutnya.



Gambar 4.3 Proses Pelarutan PCB

Setelah PCB selesai melalui proses *etching*, dilakukan pengeboran pada titik-titik yang telah ditentukan sebagai tempat pemasangan kaki komponen dan konektor. Proses pengeboran disesuaikan dengan posisi komponen pada *layout* PCB agar pemasangan komponen dapat dilakukan sesuai dengan rancangan. PCB yang telah selesai diproses kemudian dirakit dengan memasang komponen elektronika yang meliputi mikrokontroler ESP32 DevKit V1, modul HLK-PM01, *solid state relay* (SSR-25DAH), konektor sensor SHT30, resistor, *terminal block*, serta konektor JST. Komponen seperti Modul OLED dan Joystick KY-023 tidak dipasang secara langsung pada PCB, tetapi dihubungkan melalui konektor JST sebagai perangkat masukan pada sistem seperti pada Gambar 4.4.



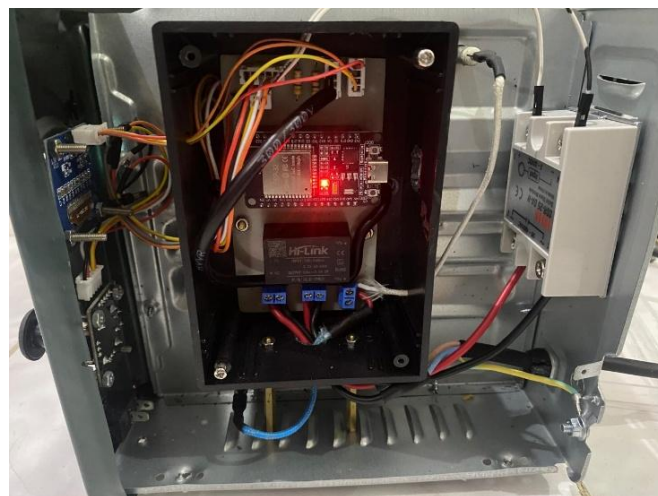
Gambar 4.4 Proses Pengeboran PCB

Setelah seluruh komponen yang terdapat pada PCB terpasang, dilakukan proses penyolderan pada setiap titik sambungan komponen dan jalur rangkaian. Penyolderan dilakukan secara bertahap untuk memastikan setiap sambungan memiliki kontak listrik yang baik. Selanjutnya, dilakukan pemeriksaan terhadap jalur rangkaian menggunakan multimeter untuk memastikan setiap jalur yang seharusnya terhubung memiliki kontinuitas dan tidak terdapat hubungan singkat (*short circuit*) antarjalur yang tidak seharusnya terhubung. Pemeriksaan ini dilakukan untuk memastikan kondisi PCB sebelum diintegrasikan ke dalam sistem pengering biji kopi. Dokumentasi proses penyolderan PCB ditunjukkan pada Gambar 4.5.



Gambar 4.5 Proses Penyolderan PCB

3. Pemasangan dan Penyambungan Kabel



Gambar 4.6 Pemasangan dan Penyambungan Kabel

Setelah proses perakitan dan penyolderan komponen pada PCB selesai dilakukan, tahap selanjutnya adalah integrasi sistem pada sisi bagian kontrol. Pada tahap ini, PCB dihubungkan dengan berbagai komponen dan perangkat pendukung yang digunakan pada alat pengering biji kopi sesuai dengan rancangan rangkaian. Komponen yang diintegrasikan meliputi sensor SHT30, modul OLED, *solid state relay* (SSR-25DAH), modul HLK-PM01 sebagai bagian dari catu daya, elemen pemanas pada oven, serta sumber tegangan. Joystick KY-023 juga dihubungkan sebagai perangkat masukan melalui konektor JST. Seluruh hubungan antarbagian sistem disesuaikan dengan rancangan kelistrikan yang telah dibuat.

Pemasangan komponen dan pengkabelan dilakukan dengan mempertimbangkan kerapian serta kemudahan dalam proses pemeriksaan. Kabel dan konektor digunakan pada bagian yang memerlukan hubungan antarperangkat sehingga pemasangan dapat dilakukan secara lebih terorganisasi dan memudahkan proses pelepasan maupun pemeriksaan apabila terjadi gangguan pada sistem. Komponen kemudian ditempatkan pada posisi yang telah ditentukan di bagian kontrol agar jalur pengkabelan lebih tertata dan tidak saling mengganggu.

Penataan komponen juga dilakukan dengan memperhatikan aspek keamanan instalasi, terutama pada pemisahan jalur tegangan AC yang digunakan untuk elemen pemanas dengan jalur tegangan DC yang digunakan pada rangkaian kendali dan komponen elektronika. Pemisahan tersebut dilakukan untuk mengurangi risiko gangguan pada rangkaian kendali serta meningkatkan keamanan selama sistem beroperasi. Setelah seluruh komponen dan koneksi terpasang, dilakukan pemeriksaan terhadap setiap sambungan untuk memastikan pengkabelan telah sesuai dengan rancangan dan tidak terdapat kesalahan koneksi.

Tahap terakhir dilakukan dengan mengaktifkan sistem untuk melakukan pemeriksaan awal terhadap fungsi setiap komponen. Pengujian dilakukan untuk memastikan perangkat dapat beroperasi sesuai dengan fungsi yang telah dirancang sebelum sistem digunakan pada proses pengeringan biji kopi. Setelah perangkat elektrik dinyatakan dapat beroperasi dengan baik, dilakukan proses pemrograman dan pengujian sistem secara keseluruhan. Hasil integrasi perangkat elektrik pada bagian kontrol ditunjukkan pada Gambar 4.6.

4.1.3 Pembuatan Perangkat Mekanik

Pembuatan perangkat mekanik pada sistem pengering biji kopi dilakukan melalui beberapa tahapan, mulai dari modifikasi oven listrik, pembuatan dan pemasangan *enclosure* panel kontrol, hingga proses perakitan seluruh komponen menjadi satu kesatuan sistem. Perancangan dilakukan terlebih dahulu untuk menentukan dimensi, posisi pemasangan, serta kebutuhan bukaan pada bagian oven dan *enclosure*. Dengan adanya desain awal, proses fabrikasi dan perakitan dapat dilakukan berdasarkan ukuran dan posisi komponen yang telah ditentukan.

1. Modifikasi Oven Listrik

Modifikasi oven dilakukan karena oven yang digunakan pada penelitian masih menggunakan sistem kendali bawaan. Sebelum dilakukan perubahan, bagian panel depan oven masih dilengkapi thermostat dan *timer* yang digunakan untuk mengatur pengoperasian oven. Kondisi awal oven sebelum dilakukan modifikasi ditunjukkan pada Gambar 4.7.



Gambar 4.7 Kondisi Oven Listrik Sebelum Modifikasi

Pada Gambar 4.7 dapat dilihat bagian panel depan oven yang masih dilengkapi dengan knop thermostat, *timer*, dan lampu *power*. Thermostat digunakan pada sistem bawaan untuk mengatur suhu oven, sedangkan *timer* digunakan untuk mengatur waktu pengoperasian oven. Lampu *power* berfungsi sebagai indikator kondisi oven saat sistem bawaan dioperasikan. Pada proses modifikasi, thermostat dan *timer* dilepas karena pengendalian elemen pemanas selanjutnya dilakukan menggunakan ESP32 dan *solid state relay* (SSR), sedangkan bagian lampu *power* digantikan dengan saklar yang digunakan untuk menyalakan dan mematikan sistem.

Proses modifikasi diawali dengan melepas thermostat, *timer*, dan lampu *power* dari panel depan oven. Pelepasan dilakukan beserta kabel yang terhubung dengan komponen tersebut agar jalur kelistrikan dan pengendalian elemen pemanas dapat disesuaikan dengan rangkaian baru. Kabel yang sebelumnya terhubung dengan sistem kendali bawaan kemudian diidentifikasi dan disesuaikan dengan kebutuhan rangkaian baru yang menggunakan *solid state relay* (SSR) sebagai pengendali elemen pemanas.

Pelepasan thermostat, *timer*, dan lampu *power* menyebabkan bagian panel depan yang sebelumnya digunakan untuk komponen tersebut menjadi kosong. Ruang yang tersedia kemudian dimanfaatkan untuk menempatkan komponen antarmuka dari sistem baru, yaitu display OLED, joystick KY-023, dan saklar. Saklar digunakan untuk menyalakan dan mematikan sistem, sedangkan OLED dan joystick digunakan sebagai antarmuka untuk menampilkan dan menavigasikan informasi sistem.

OLED digunakan sebagai media untuk menampilkan informasi yang berkaitan dengan kondisi sistem, sedangkan joystick KY-023 digunakan sebagai tombol navigasi untuk berpindah atau menggeser tampilan pada OLED sehingga informasi suhu, kelembapan, dan *output* PID dapat dilihat secara bergantian. Saklar digunakan untuk mengendalikan kondisi nyala dan mati sistem. Ketiga komponen tersebut ditempatkan pada bagian depan oven agar proses pengoperasian dan pemantauan kondisi sistem dapat dilakukan secara langsung. Penempatan OLED, joystick, dan saklar juga disesuaikan dengan rancangan *enclosure* yang telah dibuat pada tahap desain tiga dimensi.

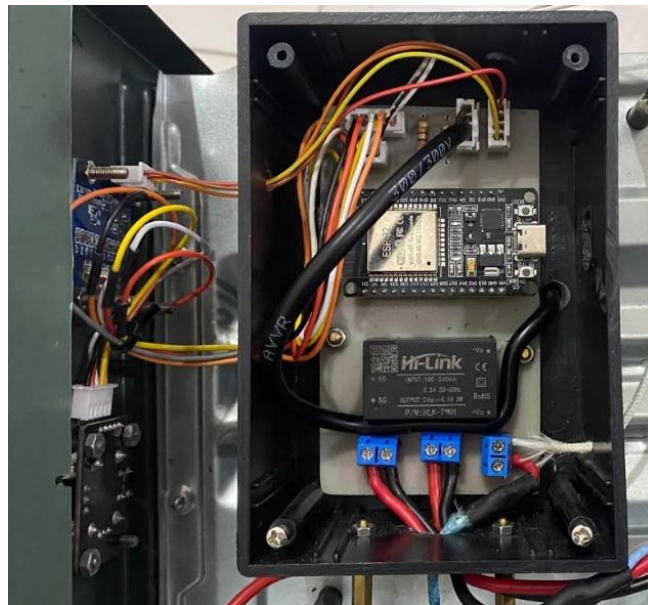
2. Pembuatan dan Pemasangan *Enclosure* Panel Kontrol serta Sensor SHT30

Enclosure panel kontrol dibuat sebagai tempat untuk menempatkan rangkaian kontrol dan komponen yang digunakan sebagai antarmuka sistem. Bentuk dan ukuran *enclosure* disesuaikan dengan rancangan tiga dimensi yang telah dibuat sebelumnya. *Enclosure* kemudian dipasang pada sisi kiri badan oven menggunakan titik pemasangan yang telah ditentukan. Penempatan tersebut memungkinkan komponen kontrol berada di luar ruang pengering sekaligus tetap mudah dijangkau ketika alat dioperasikan.

Bagian dalam *enclosure* digunakan untuk menempatkan rangkaian kontrol yang telah dirakit pada PCB. Komponen yang berada di dalam *enclosure* meliputi ESP32 DevKit V1, modul catu daya HLK-PM01, terminal block, serta konektor yang digunakan untuk menghubungkan bagian kontrol dengan komponen lainnya. Penempatan komponen dilakukan dengan memperhatikan kerapian jalur kabel sehingga setiap sambungan dapat dihubungkan sesuai dengan fungsi masing-masing.

Pada bagian depan *enclosure* disiapkan tempat untuk pemasangan OLED dan joystick KY-023. OLED digunakan untuk menampilkan informasi yang berkaitan dengan kondisi sistem, sedangkan joystick digunakan sebagai perangkat masukan. Kedua komponen tersebut ditempatkan pada bagian yang dapat dijangkau dari luar sehingga pengoperasian alat dapat dilakukan tanpa membuka *enclosure*.

Kondisi bagian dalam *enclosure* setelah komponen kontrol dan jalur kabel terpasang dapat dilihat pada Gambar 4.8. Pada gambar tersebut terlihat ESP32 dan modul HLK-PM01 ditempatkan di dalam *enclosure*, sedangkan beberapa kabel dan terminal digunakan sebagai penghubung menuju komponen yang berada di luar *enclosure*. Salah satu jalur kabel pada bagian sisi kiri *enclosure* digunakan untuk menghubungkan sensor SHT30 yang ditempatkan di dalam ruang pengering.



Gambar 4.8 Kondisi Bagian Dalam *Enclosure* Panel Kontrol

Kabel sensor SHT30 diarahkan keluar dari *enclosure* melalui lubang pada bagian samping yang telah disiapkan pada proses modifikasi oven. Kabel tersebut kemudian melewati dinding oven menuju ruang pengering. Jalur ini dibuat agar sensor yang berada di dalam oven tetap dapat terhubung dengan rangkaian kontrol yang berada di dalam *enclosure*

Sensor SHT30 dipasang pada bagian dalam ruang pengering dengan posisi yang telah ditentukan melalui rancangan. Penempatan sensor dilakukan pada bagian yang tidak mengganggu pergerakan *tray* dan tidak bersentuhan langsung dengan elemen pemanas. Posisi tersebut digunakan agar sensor dapat digunakan untuk memperoleh pembacaan suhu dan kelembapan relatif selama proses pengeringan.



Gambar 4.9 Posisi Penempatan Sensor SHT30 di Dalam Ruang Oven

Pada Gambar 4.9 dapat dilihat posisi sensor SHT30 yang telah ditempatkan di dalam ruang pengering. Kabel sensor diarahkan dari bagian dalam oven menuju *enclosure* melalui lubang yang terdapat pada badan oven. Setelah kabel terpasang, jalurnya ditata agar tidak menghalangi saat *tray* dimasukkan atau dikeluarkan dari oven dan tidak berada terlalu dekat dengan elemen pemanas.

Pemasangan *enclosure* dan sensor kemudian diperiksa kembali untuk memastikan keduanya berada pada posisi yang telah ditentukan. Pada bagian *enclosure* diperiksa kekuatan pemasangan dan kondisi jalur kabel, sedangkan pada sensor diperiksa posisi pemasangan serta jalur kabel yang melewati dinding oven.

Pemeriksaan juga dilakukan dengan mencoba pergerakan *tray* untuk memastikan kabel sensor tidak mengganggu proses penggunaannya.

Dengan selesainya tahap tersebut, bagian kontrol yang berada di dalam *enclosure* telah terhubung dengan sensor SHT30 yang ditempatkan di ruang pengering. *Enclosure* berfungsi sebagai tempat rangkaian kontrol dan antarmuka, sedangkan kabel SHT30 menjadi penghubung antara sensor di dalam oven dengan rangkaian kontrol di luar ruang pengering.

3. Pemasangan *Solid state relay*

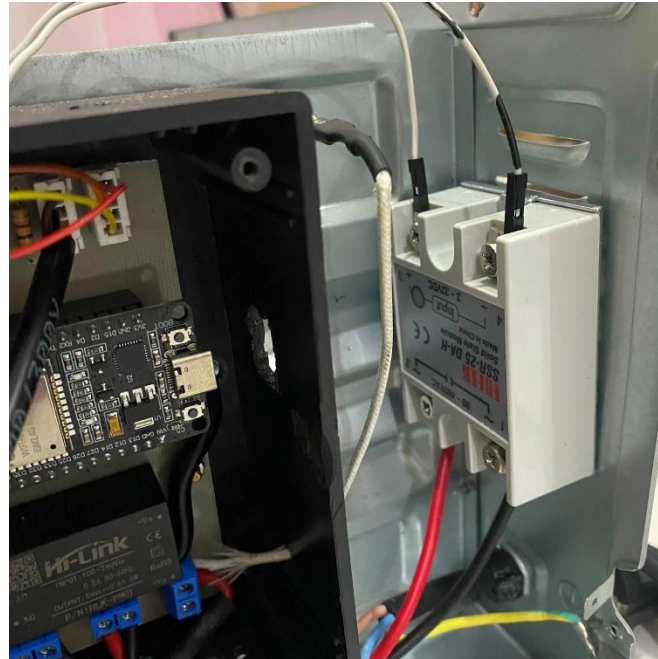
Solid state relay (SSR) FOTEK SSR-25DAH dipasang sebagai bagian penghubung antara rangkaian kontrol dengan elemen pemanas pada oven. Pemasangan dilakukan pada bagian luar badan oven, tepatnya di sisi kanan *enclosure* panel kontrol. Posisi tersebut dipilih agar SSR tidak ditempatkan di dalam *enclosure* yang berisi rangkaian kontrol bertegangan rendah, sekaligus memudahkan penghubungan jalur SSR dengan elemen pemanas oven.

SSR dipasang pada badan oven menggunakan baut dan mur pada titik yang telah ditentukan sebelumnya. Pemasangan pada bagian luar badan oven juga memungkinkan SSR berada pada permukaan logam yang dapat membantu pelepasan panas selama komponen bekerja. Hal tersebut diperlukan karena SSR akan melakukan proses penyaklaran terhadap elemen pemanas selama sistem pengering beroperasi.

Elemen pemanas yang digunakan pada oven memiliki daya sebesar 750 W dengan sumber tegangan 220V AC. Berdasarkan hubungan antara daya dan tegangan, arus yang mengalir pada elemen pemanas dapat dihitung menggunakan persamaan berikut.

$$I = \frac{P}{V} = \frac{750}{220} = 3,41 \text{ A}$$

Arus kerja sebesar 3,41 A tersebut masih berada di bawah kapasitas arus beban SSR-25DAH yang digunakan, yaitu sebesar 25 A. Dengan demikian, SSR yang dipasang memiliki kapasitas yang sesuai dengan kebutuhan pengendalian elemen pemanas pada sistem.



Gambar 4.10 Pemasangan SSR FOTEK SSR-25DAH pada Badan Oven

Posisi SSR setelah dipasang pada badan oven dapat dilihat pada Gambar 4.10. Pada gambar tersebut terlihat SSR ditempatkan di sisi luar oven dan berada di dekat *enclosure* panel kontrol. Penempatan ini membuat jalur kabel antara bagian kontrol, SSR, dan elemen pemanas dapat disusun dengan lebih terarah.

Pemasangan kabel dilakukan pada dua sisi SSR, yaitu sisi kontrol dan sisi beban. Pada sisi kontrol, terminal SSR dihubungkan dengan rangkaian ESP32 melalui jalur yang berasal dari *enclosure*. Jalur kontrol tersebut terhubung ke GPIO26 dan GND pada ESP32. Sementara itu, sisi beban SSR dihubungkan dengan jalur tegangan 220 V AC dan terminal elemen pemanas oven.

Kabel pada sisi beban ditempatkan melalui jalur yang telah disiapkan pada proses modifikasi oven. Sambungan tersebut dilakukan menggunakan terminal sekrup pada SSR agar kabel dapat terpasang dengan kuat. Setelah seluruh kabel terhubung, posisi kabel ditata kembali agar tidak mengganggu bagian lain pada oven dan tidak berada pada jalur pergerakan pintu maupun *tray*.

Pemeriksaan dilakukan setelah pemasangan SSR dan seluruh kabel selesai dilakukan. Pemeriksaan meliputi kondisi pemasangan SSR pada badan oven, kekencangan baut dan terminal sekrup, serta kesesuaian jalur kabel antara ESP32,

SSR, dan elemen pemanas. Pemeriksaan pada bagian sambungan juga dilakukan untuk memastikan tidak terdapat kabel yang longgar sebelum sistem dioperasikan.

Dengan selesainya pemasangan SSR, jalur pengendalian elemen pemanas telah terhubung dengan rangkaian kontrol yang berada di dalam *enclosure*. SSR menjadi bagian penghubung antara sinyal kendali dari ESP32 dengan sumber daya elemen pemanas, sehingga proses penyalaan dan penghentian pemanas dapat dilakukan oleh sistem kendali yang telah dirancang.

4. Pemeriksaan Akhir dan Hasil Pembuatan Perangkat Mekanik

Pemeriksaan akhir dilakukan setelah seluruh proses pemasangan dan modifikasi pada oven selesai. Pemeriksaan ini dilakukan untuk memastikan setiap bagian yang telah dibuat dan dipasang berada pada posisi yang sesuai dengan rancangan. Bagian yang diperiksa meliputi *enclosure* panel kontrol, posisi OLED dan joystick, sensor SHT30, SSR, jalur kabel, serta bagian oven yang mengalami perubahan selama proses pembuatan.

Pada bagian luar oven, pemeriksaan dilakukan terhadap pemasangan *enclosure* dan SSR. Baut dan mur yang digunakan sebagai pengikat diperiksa kembali untuk memastikan kedua komponen terpasang dengan kuat. Posisi *enclosure* juga diperiksa agar tetap berada pada sisi kiri badan oven sesuai dengan rancangan, sedangkan SSR diperiksa pada bagian pemasangannya di sisi luar oven.

Pemeriksaan dilanjutkan pada bagian antarmuka yang terdapat pada *enclosure*. Posisi OLED dan joystick diperiksa untuk memastikan keduanya terpasang dengan baik dan dapat digunakan dari luar panel. Bagian dalam *enclosure* juga diperiksa untuk melihat kondisi pemasangan komponen kontrol dan jalur kabel yang telah dirangkai. Jalur kabel yang keluar dari *enclosure* menuju sensor SHT30 dan SSR diperhatikan agar tidak tertarik atau terjepit pada saat oven digunakan.

Pada bagian dalam ruang pengering, posisi sensor SHT30 diperiksa kembali. Sensor dipastikan tetap berada pada posisi yang telah ditentukan dan kabelnya tidak mengganggu pergerakan *tray*. Pintu oven kemudian dibuka dan ditutup untuk memastikan tidak terdapat kabel atau komponen tambahan yang menghambat pergerakannya. *Tray* juga dimasukkan dan dikeluarkan kembali untuk memastikan ruang pengering tetap dapat digunakan sebagaimana mestinya.

Pemeriksaan terhadap sambungan dilakukan menggunakan multimeter untuk memastikan tidak terdapat hubungan singkat pada bagian rangkaian yang seharusnya terpisah. Selain itu, kondisi sambungan kabel dan terminal diperiksa kembali secara visual. Pemeriksaan tersebut dilakukan sebelum alat digunakan pada tahap pemrograman dan pengujian sistem. Bagian antarmuka berupa OLED dan joystick ditempatkan pada panel kontrol, sementara sensor SHT30 telah dihubungkan dari bagian kontrol menuju ruang pengering.

Hasil akhir pembuatan perangkat mekanik berupa satu unit oven listrik yang telah disesuaikan dengan kebutuhan sistem pengering biji kopi. Sistem kendali bawaan berupa thermostat dan *timer* telah dilepas dari oven, kemudian bagian panelnya dimanfaatkan untuk penempatan antarmuka sistem yang baru. Elemen pemanas dan *tray* bawaan tetap digunakan, sedangkan komponen tambahan berupa *enclosure*, sensor SHT30, dan SSR telah dipasang sesuai dengan rancangan.

Dengan selesainya tahap pemeriksaan tersebut, bagian mekanik alat telah terpasang dan terintegrasi dengan perangkat elektrik yang dibuat pada tahap sebelumnya. Kondisi fisik alat selanjutnya digunakan sebagai dasar untuk melanjutkan proses pembuatan perangkat lunak dan pengujian sistem pengering biji kopi.

4.2 Pembuatan Perangkat Lunak

Pembuatan perangkat lunak merupakan tahapan yang dilakukan untuk mengatur cara kerja sistem pengering biji kopi agar seluruh komponen perangkat keras dapat beroperasi sesuai dengan fungsi yang dirancang. Pada tahap ini dilakukan pengembangan program pada mikrokontroler ESP32 menggunakan Visual Studio Code dengan ekstensi PlatformIO sebagai lingkungan pengembangan. Program yang dibuat meliputi proses pembacaan data dari sensor SHT30, implementasi algoritma kontrol *Proportional–Integral–Derivative* (PID) untuk mengendalikan suhu ruang pengering melalui *Solid state relay* (SSR), serta pengiriman data suhu dan kelembapan ke aplikasi Blynk melalui jaringan Wi-Fi. Tahapan pembuatan perangkat lunak pada penelitian ini meliputi pemrograman ESP32, penyusunan alur kerja sistem (flowchart), implementasi algoritma PID,

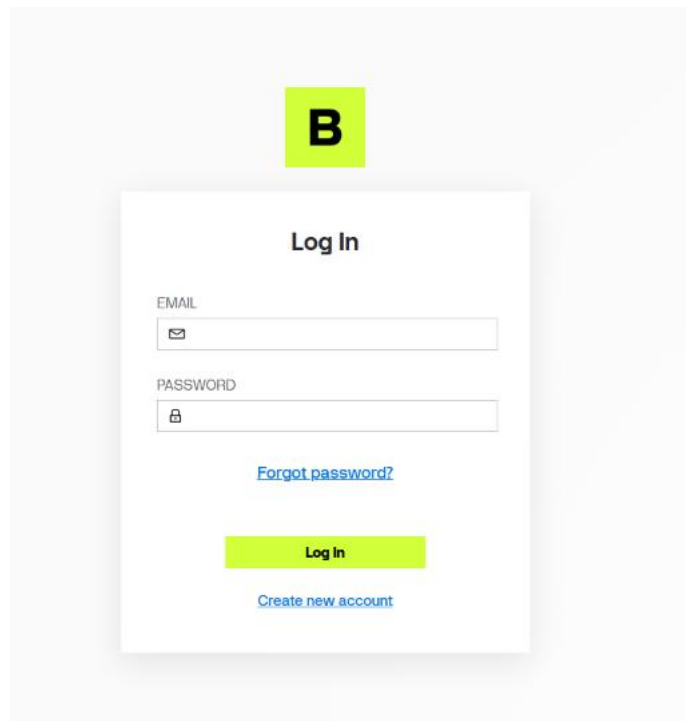
konfigurasi koneksi Wi-Fi dan Blynk, serta pengujian pengiriman data ke aplikasi Blynk.

4.2.1 Pembuatan *Template* dan Konfigurasi Blynk

Platform Blynk digunakan sebagai media pemantauan jarak jauh pada penelitian ini karena menyediakan antarmuka yang sederhana untuk menampilkan data sensor secara real-time dan dapat diakses melalui aplikasi pada perangkat mobile. Sebelum data dari ESP32 dapat dipantau, beberapa tahap konfigurasi perlu dilakukan terlebih dahulu melalui Blynk Web Console.

1. Pembuatan akun dan *Login*

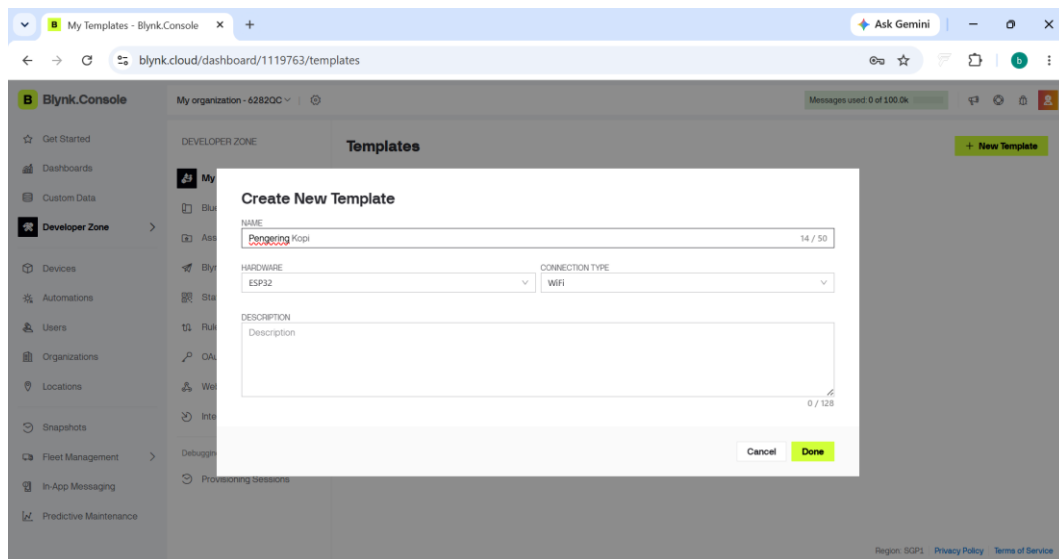
Tahap awal dilakukan dengan mengakses laman resmi Blynk melalui browser pada alamat `blynk.cloud`. Bagi pengguna yang belum memiliki akun, pendaftaran dilakukan dengan mengisi alamat email dan kata sandi pada menu *Sign up*. Setelah akun berhasil dibuat, proses masuk dilakukan dengan memasukkan email dan kata sandi yang telah didaftarkan pada halaman *Login*. Tampilan halaman *login* ditunjukkan pada



Gambar 4.11 Halaman *Login*

2. Pembuatan *Template*

Setelah berhasil masuk ke Blynk Console, tahap berikutnya adalah pembuatan *template* baru melalui menu *New Template*. Pada formulir yang muncul, nama *template* diisi sesuai dengan nama perangkat yang dirancang, yaitu Pengereng Kopi. Jenis perangkat keras dipilih sebagai ESP32, sedangkan jenis koneksi ditetapkan sebagai Wi-Fi, sesuai dengan modul komunikasi yang tersedia pada mikrokontroler yang digunakan.



Gambar 4.12 Pembuatan *Template*

Setelah seluruh isian dilengkapi, *template* disimpan dengan menekan tombol Done. Proses pembuatan *template* ditunjukkan pada Gambar 4.12.

3. Penambahan *Datastream*

Datastream merupakan saluran komunikasi data antara perangkat keras dan aplikasi Blynk, di mana setiap parameter yang dipantau direpresentasikan melalui satu *virtual pin*. Penambahan *datastream* dilakukan melalui menu *Datastreams* pada halaman *template* yang telah dibuat, kemudian dipilih jenis *Virtual pin* untuk tiap parameter yang akan dikirim.

SUHU

General

Semantic

Units

History

Expiration

Sync

Automations

Visibility

Notes

API reference

General

DATASTREAM NAME
SUHU → DISPLAY NAME
SUHU
For your reference and for MQTT topic naming Shown in widgets, dashboards and reports.

VIRTUAL PIN
V0
COLOR
 ICON


DATA TYPE
Integer Double String Enum Location

MIN VALUE
0
MAX VALUE
100
DECIMALS

DEFAULT VALUE
0

Cancel Save

Gambar 4.13 Penambahan *Datastream*

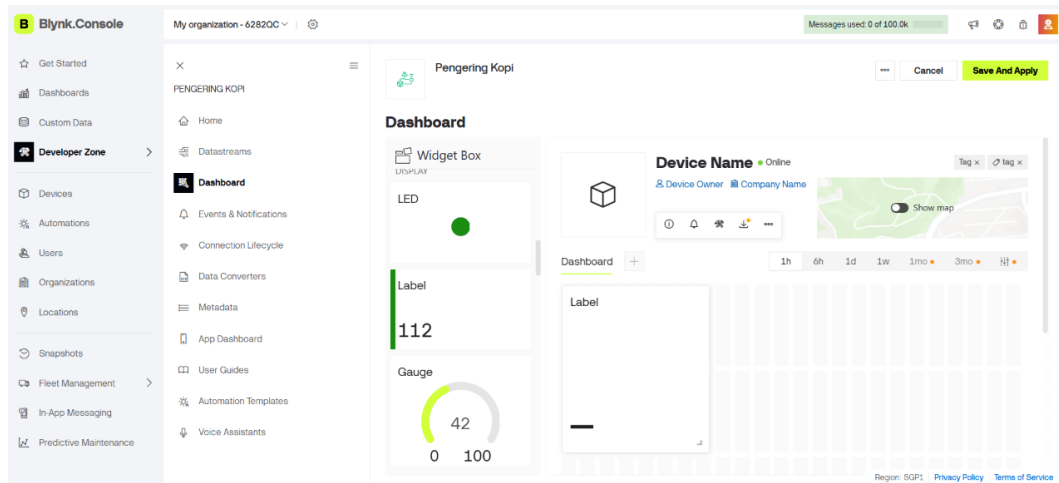
Pada penelitian ini, tiga *datastream* ditambahkan untuk menampung data suhu, kelembapan, dan *output* PID. Setiap *datastream* diberi nama, tipe data, satuan, serta rentang nilai minimum dan maksimum sesuai dengan karakteristik parameter yang diwakilinya. Rincian konfigurasi *datastream* disajikan pada Tabel 4-3.

Tabel 4-3 Konfigurasi *Datastream*

Virtuan Pin	Nama	Tipe Data	Satuan	Nilai Minimum	Nilai Maksimum
V0	Suhu	Double	°C	0	100
V1	Kelembapan	Double	%	0	100
V2	PID	Double	%	0	100

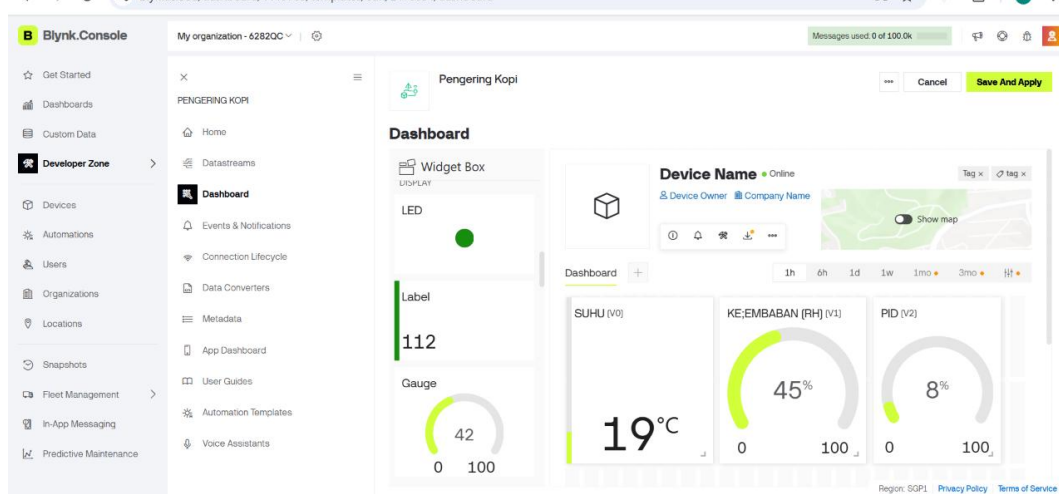
4. Perancangan *Dashboard*

Tahap terakhir adalah perancangan tampilan *dashboard* yang berfungsi sebagai antarmuka pemantauan bagi pengguna. *Widget* ditambahkan melalui menu *Web Dashboard* dengan cara menarik (*drag and drop*) jenis *widget* yang diinginkan ke area kerja. *Widget* yang digunakan pada penelitian ini terdiri atas *labeled value* dan *gauge*, yang masing-masing dihubungkan dengan *datastream* terkait melalui pengaturan pada setiap *widget*.



Gambar 4.14 Perancangan *Dashboard*

Setelah seluruh *widget* ditempatkan dan dihubungkan, tampilan *dashboard* disimpan dan dapat langsung diakses melalui aplikasi Blynk pada perangkat mobile. Hasil akhir tampilan *dashboard* ditunjukkan pada Gambar 4.15



Gambar 4.15 Tampilan Akhir *Dashboard*

Dengan selesainya tahapan *login*, pembuatan *template*, konfigurasi *datastream*, dan perancangan *dashboard*, platform Blynk telah siap digunakan untuk menerima data yang dikirimkan oleh ESP32.

4.2.2 Konfigurasi *Library*

Seluruh pustaka (*library*) yang dibutuhkan pada penelitian ini didaftarkan pada bagian `lib_deps` di dalam berkas `platformio.ini`, sehingga proses pemasangan dilakukan secara otomatis saat proyek dibangun (*build*). Pustaka yang didaftarkan

meliputi Adafruit SHT31 *Library* untuk pembacaan sensor suhu dan kelembapan, Adafruit GFX *Library* sebagai pustaka grafis dasar, Adafruit SH110X sebagai pustaka pengendali layar OLED, serta Blynk untuk keperluan komunikasi dengan platform pemantauan. Rincian pustaka yang digunakan disajikan pada Tabel 4-4. Tampilan konfigurasi pustaka pada berkas platformio.

Tabel 4-4 Tampilan Konfigurasi Pustaka Pada Berkas Platform IO

No	<i>Library</i>	Fungsi
1	Adafruit SHT31 <i>Library</i>	Pembacaan data suhu dan kelembapan dari sensor SHT30
2	Adafruit GFX <i>Library</i>	Penyediaan fungsi grafis dasar untuk tampilan OLED
3	Adafruit SH110X	Pengendalian layar OLED SH1106
4	Blynk	Komunikasi data antara ESP32 dan platform Blynk

Setelah seluruh pustaka terpasang, pemanggilan dilakukan pada bagian awal berkas main.cpp menggunakan perintah `#include`. Selain pustaka utama yang telah disebutkan, turut dipanggil pula beberapa pustaka bawaan dari framework Arduino dan ESP32, yaitu `WiFi.h` untuk keperluan koneksi jaringan nirkabel, `Wire.h` untuk komunikasi I2C, `time.h` untuk pengelolaan waktu, serta `HTTPClient.h` yang digunakan pada proses pengiriman data ke Google Sheets. Potongan kode pemanggilan pustaka tersebut ditampilkan berikut.

```

17
18  #include "Arduino.h"
19  #include <WiFi.h>
20  #include <Wire.h>
21  #include <time.h>
22  #include <HTTPClient.h>
23  #include <Adafruit_SHT31.h>
24  #include <Adafruit_GFX.h>
25  #include <Adafruit_SH110X.h>
26  #include <BlynkSimpleEsp32.h>
27

```

Gambar 4.16 Pemanggilan *Library* pada main.cpp

4.2.3 Pemrograman Pembacaan Sensor SHT30 dan PID

Pembacaan data sensor SHT30 dan perhitungan algoritma PID diprogram dalam satu alur fungsi yang sama, mengingat nilai suhu hasil pembacaan sensor digunakan secara langsung sebagai variabel masukan bagi perhitungan PID pada iterasi berikutnya.

Pembacaan sensor dilakukan secara berkala dengan periode pembacaan yang ditetapkan setiap 1000 milidetik. Nilai suhu dan kelembapan diperoleh melalui fungsi `readTemperature()` dan `readHumidity()` dari pustaka Adafruit SHT31. Sebelum digunakan lebih lanjut, hasil pembacaan diperiksa validitasnya menggunakan fungsi `isnan()`. Apabila nilai yang terbaca tidak valid, sistem menandai kondisi sensor sebagai bermasalah, keluaran kontrol diatur menjadi nol, dan pin SSR dinonaktifkan sebagai langkah pengamanan. Potongan kode pembacaan dan validasi sensor ditampilkan pada Gambar 4.17.

```

255     if (now - tSensor >= SENSOR_INTERVAL) {
256         tSensor = now;
257
258         float t = sht30.readTemperature();
259         float h = sht30.readHumidity();
260
261         if (isnan(t) || isnan(h)) {
262             localSensorOK = false;
263             localDutyCycle = 0;
264             digitalWrite(SSR_PIN, LOW);
265             Serial.println("[SENSOR] ERROR!");
266         } else {
267             localSuhu = t;
268             localKelembapan = h;
269             localSensorOK = true;

```

Gambar 4.17 Kode Pembacaan dan Validasi Sensor SHT30

Apabila pembacaan sensor dinyatakan valid, nilai suhu tersebut selanjutnya digunakan dalam perhitungan PID. Sebelum perhitungan dilakukan, beberapa kondisi diperiksa terlebih dahulu sebagai bentuk pengamanan sistem. Perhitungan PID hanya dijalankan apabila sistem dalam kondisi aktif, suhu belum melewati batas aman sebesar 75 °C, dan nilai kelembapan belum mencapai target yang ditentukan sebesar 20%. Apabila salah satu dari kondisi tersebut tidak terpenuhi, keluaran kontrol diatur menjadi nol dan nilai akumulasi *error* pada komponen integral turut direset.

Perhitungan PID dilakukan dengan terlebih dahulu menentukan nilai *error*, yaitu selisih antara nilai *setpoint* dan suhu aktual yang terbaca. Nilai *error* tersebut kemudian diolah melalui tiga komponen, yaitu komponen proporsional yang diperoleh dari perkalian *error* dengan konstanta K_p , komponen integral yang diperoleh dari akumulasi *error* dikalikan dengan konstanta K_i , serta komponen derivatif yang diperoleh dari selisih *error* saat ini dengan *error* sebelumnya dikalikan dengan konstanta K_d . Konstanta yang digunakan pada penelitian ini yaitu K_p sebesar 1,5, K_i sebesar 0, dan K_d sebesar 0,7. Potongan kode perhitungan PID ditampilkan pada Gambar 4.18 Kode Perhitungan Algoritma PID.

```

271     if (!localRunning) {
272         // sistem berhenti -> PID tidak aktif
273         localDutyCycle = 0;
274         pidIntegral    = 0;
275         pidLastError   = 0;
276     } else if (localSuhu >= SAFETY_MAX) {
277         // proteksi dasar hardware (suhu)
278         localDutyCycle = 0;
279         pidIntegral    = 0;
280         Serial.printf("[SAFETY] suhu %.1f >= %.1f -> OFF\n", localSuhu, SAFETY_MAX);
281     } else if (localKelembapan <= KELEMBAPAN_TARGET) {
282         // target kelembapan tercapai -> proses pengeringan selesai, PID mati
283         localDutyCycle = 0;
284         pidIntegral    = 0;
285         Serial.printf("[SELESAI] RH %.1f%% <= %.1f%% -> PID OFF\n",
286                       localKelembapan, KELEMBAPAN_TARGET);
287     } else {
288         // ---- PID standar ----
289
290         float error = SETPOINT - localSuhu;
291
292         pidIntegral += error;
293         float P = Kp * error;
294         float I = Ki * pidIntegral;
295         float D = Kd * (error - pidLastError);
296         pidLastError = error;
297
298         float out = P + I + D;
299         if (out > DAYA_MAX) out = DAYA_MAX;
300         if (out < 0)      out = 0;
301
302         localDutyCycle = out;
303     }
304

```

Gambar 4.18 Kode Perhitungan Algoritma PID

Nilai keluaran PID yang diperoleh selanjutnya digunakan untuk menentukan lama waktu aktifnya SSR dalam satu periode waktu tertentu. Pendekatan ini dilakukan dengan membandingkan proporsi nilai keluaran PID terhadap durasi satu periode kontrol yang ditetapkan, sehingga semakin besar nilai

keluaran PID, semakin lama pula durasi SSR dalam kondisi aktif pada periode tersebut. Metode ini digunakan karena SSR bekerja sebagai saklar dengan dua kondisi (hidup atau mati), sehingga tidak dapat diberikan sinyal analog secara langsung sebagaimana aktuator berjenis kontinu. Potongan kode pengendalian SSR berdasarkan keluaran PID ditampilkan pada Gambar 4.19.

```

319
320 // ----- 3. SSR slow-PWM, dicek tiap tick 100ms -----
321 {
322     if (now - windowStart >= WINDOW_SIZE) windowStart = now;
323     unsigned long onTime = (unsigned long)((localDutyCycle / 100.0) * WINDOW_SIZE);
324     bool enable = localRunning && localSensorOK;
325     bool on = enable && ((now - windowStart) < onTime);
326     digitalWrite(SSR_PIN, on ? HIGH : LOW);
327 }

```

Gambar 4.19 Kode Pengendalian SSR Berbasis Keluaran PID

4.2.4 Pemrograman Antarmuka Lokal (OLED dan Joystick)

Antarmuka lokal pada alat terdiri atas layar OLED sebagai media penampil informasi dan joystick sebagai media masukan bagi pengguna. Kedua komponen ini memungkinkan pengguna memantau kondisi alat serta mengendalikan sistem secara langsung tanpa harus mengakses aplikasi Blynk.

Tampilan pada layar OLED dibagi menjadi tiga halaman, yaitu halaman suhu, halaman kelembapan, dan halaman duty cycle keluaran PID, yang dapat dipilih secara bergantian oleh pengguna. Setiap kali tampilan diperbarui, bagian atas layar menampilkan status sistem berupa kondisi berjalan atau berhenti, nomor halaman yang sedang aktif, serta indikator status koneksi jaringan. Apabila kondisi sensor dinyatakan tidak valid, layar akan menampilkan peringatan kesalahan sensor menggantikan tampilan data pada halaman manapun yang sedang dipilih. Potongan kode pembaruan tampilan header dan penanganan kondisi sensor bermasalah ditampilkan pada Gambar 4.20.

```

361 oled.setTextSize(1);
362 oled.setCursor(0, 0);
363 oled.print(localRunning ? "RUN " : "STOP");
364 oled.setCursor(48, 0);
365 oled.printf("[%d/%d]", localPage + 1, TOTAL_PAGE);
366 oled.setCursor(110, 0);
367 oled.print(localWifiOK ? "W" : "-");
368 oled.drawFastHLine(0, 11, 128, SH110X_WHITE);
369
370 if (!localSensorOK) {
371     oled.setTextSize(1);
372     oled.setCursor(0, 28);
373     oled.print("SENSOR ERROR!");
374 }
375

```

Gambar 4.20 Kode Tampilan Status dan Peringatan Sensor pada OLED

Pada kondisi sensor terbaca dengan baik, tampilan halaman utama menyesuaikan dengan halaman yang sedang dipilih oleh pengguna. Halaman suhu menampilkan nilai suhu aktual dalam ukuran huruf yang lebih besar agar mudah dibaca, sementara halaman kelembapan dan halaman *duty cycle* menampilkan informasi sejenis untuk parameter masing-masing. Potongan kode tampilan salah satu halaman, yaitu halaman suhu, ditampilkan pada Gambar 4.21 Kode Tampilan Halaman Suhu pada OLED.

```

376 oled.setTextSize(1);
377 oled.setCursor(0, 16); oled.print("SUHU");
378 oled.setTextSize(3);
379 oled.setCursor(0, 28); oled.print(localSuhu, 1);
380 oled.setTextSize(1);
381 oled.setCursor(98, 36); oled.print("C");
382 oled.setCursor(0, 56);

```

Gambar 4.21 Kode Tampilan Halaman Suhu pada OLED

Masukan dari pengguna diterima melalui joystick, yang terdiri atas satu sumbu analog untuk navigasi halaman dan satu tombol tekan untuk mengubah status berjalan atau berhentinya sistem. Pembacaan sumbu analog dilakukan dengan membandingkan nilai bacaan terhadap ambang batas tertentu untuk menentukan arah pergerakan, yaitu ke kiri atau ke kanan, disertai mekanisme yang memastikan perubahan halaman hanya terjadi satu kali untuk setiap pergerakan joystick hingga posisi kembali netral. Potongan kode pembacaan arah joystick ditampilkan pada Gambar 4.22.

```

553 // ----- 1. Baca arah joystick (horizontal) -----
554 int h = analogRead(JOY_HORIZONTAL);
555 int arah = 0;
556
557 if (center) {
558     if (h < 100) { arah = -1; center = false; } // ke kiri
559     else if (h > (4095 - 100)) { arah = 1; center = false; } // ke kanan
560 } else {
561     if (h > 1300 && h < 2700) center = true; // kembali netral
562 }
563
564 if (arah != 0) {
565     if (lockData()) {
566         shared.page = (shared.page + arah + TOTAL_PAGE) % TOTAL_PAGE;
567         unlockData();
568     }
569     Serial.printf("[JOY] arah=%d -> page berubah\n", arah);
570 }
571

```

Gambar 4.22 Kode Pembacaan Arah Joystick

Pembacaan tombol tekan dilakukan dengan memeriksa perubahan kondisi pin dari kondisi tinggi menuju kondisi rendah, disertai penerapan tundaan waktu (*debounce*) selama 250 milidetik untuk mencegah pembacaan ganda akibat ketidakstabilan sinyal mekanis pada saat tombol ditekan. Setiap penekanan tombol yang valid menyebabkan status berjalannya sistem berubah dari aktif menjadi tidak aktif, atau sebaliknya. Potongan kode pembacaan tombol joystick ditampilkan pada Gambar 4.23.

```

572 // ----- 2. Baca tombol tekan (toggle running) -----
573 bool nowButton = digitalRead(JOY_SW);
574 if (lastButton == HIGH && nowButton == LOW &&
575     millis() - lastDebounce > DEBOUNCE_MS) {
576     lastDebounce = millis();
577
578     if (lockData()) {
579         shared.running = !shared.running;
580         Serial.printf("[JOY] Tombol ditekan -> running = %s\n",
581             shared.running ? "ON" : "OFF");
582         unlockData();
583     }
584 }
585 lastButton = nowButton;
586 vTaskDelayUntil(&lastWake, tickPeriod);
587 }
588
589 }

```

Gambar 4.23 Kode Pembacaan Tombol Joystick

4.2.5 Konfigurasi Konektivitas WiFi dan Blynk

Koneksi antara ESP32 dan platform Blynk dilakukan melalui jaringan Wi-Fi. Sebelum proses koneksi dapat berjalan, beberapa parameter perlu didefinisikan terlebih dahulu di dalam program, yaitu nama jaringan (SSID) dan kata sandi Wi-Fi,

serta kredensial autentikasi Blynk yang terdiri atas *Template ID*, *Template Name*, dan *Auth token*. Kredensial Blynk ini diperoleh dari *template* yang telah dibuat sebelumnya pada Blynk Console sebagaimana dijelaskan pada subbab 4.2.1. Rincian parameter koneksi yang digunakan disajikan pada Tabel 4-5.

Tabel 4-5 Parameter Koneksi

No	Parameter	Nilai
1	<i>Template ID</i>	TMPL6tZlgvhCL
2	<i>Template Name</i>	Pengering Kopi
3	Batas waktu koneksi WiFi	15 detik
4	Jeda percobaan ulang WiFi	10 detik

Potongan kode pendefinisian kredensial ditampilkan pada Gambar 4.24.

```

13 #define BLYNK_TEMPLATE_ID "TMPL6tZlgvhCL"
14 #define BLYNK_TEMPLATE_NAME "Pengering Kopi"
15 #define BLYNK_AUTH_TOKEN "_9wQ62MX-P3Udw8aU5MC0Dp4Wdf_3K1m"
16 #define BLYNK_PRINT Serial
17
45 // WIFI
46 // =====
47 char ssid[] = "nopi";
48 char password[] = "gorengikan";
49

```

Gambar 4.24 Kode Pendefinisian Kredensial WiFi dan Blynk

Pengelolaan koneksi WiFi pada program ini menggunakan pendekatan *state machine* yang terdiri atas tiga kondisi, yaitu `WIFI_IDLE`, `WIFI_CONNECTING`, dan `WIFI_CONNECTED_STATE`. Pada kondisi `WIFI_IDLE`, program memeriksa apakah jeda waktu sejak percobaan terakhir telah melewati batas yang ditentukan, kemudian memulai proses sambungan dengan memanggil fungsi `WiFi.begin()`. Setelah pemanggilan tersebut, kondisi berpindah ke `WIFI_CONNECTING` untuk menunggu hasil sambungan. Potongan kode proses koneksi WiFi ditampilkan pada Gambar 4.25.

```

440     case WIFI_IDLE:
441         // belum konek / baru saja terputus -> coba mulai connect
442         if (now - wifilastTry >= WIFI_RETRY_INTERVAL) {
443             Serial.printf("[WIFI] Mencoba sambung ke '%s'...\n", ssid);
444             WiFi.disconnect();
445             WiFi.begin(ssid, password);
446             wifiAttemptStart = now;
447             wifilastTry      = now;
448             wifiState        = WIFI_CONNECTING;
449         }
450         break;
451

```

Gambar 4.25 Kode Proses Koneksi WiFi

Pada kondisi `WIFI_CONNECTING`, program memeriksa apakah status WiFi telah berubah menjadi `WL_CONNECTED`. Apabila koneksi berhasil, konfigurasi Blynk dilakukan dengan memanggil fungsi `Blynk.config()` yang memuat *Auth token* sebagai parameter autentikasi. Pada tahap yang sama, sinkronisasi waktu melalui NTP (Network Time Protocol) turut dimulai dengan memanggil fungsi `configTime()` menggunakan server `pool.ntp.org` dan offset zona waktu sebesar `+25200` detik (UTC+7, WIB). Sinkronisasi waktu ini diperlukan untuk pemberian stempel waktu (*timestamp*) pada data yang dikirimkan ke Google Sheets. Apabila koneksi tidak berhasil dalam batas waktu 15 detik, kondisi dikembalikan ke `WIFI_IDLE` untuk melakukan percobaan ulang. Potongan kode tersebut ditampilkan pada Gambar 4.26.

```

452     case WIFI_CONNECTING:
453         if (WiFi.status() == WL_CONNECTED) {
454             Serial.print("[WIFI] Tersambung. IP: ");
455             Serial.println(WiFi.localIP());
456
457             Blynk.config(BLYNK_AUTH_TOKEN);
458             blynkCfg = true;
459
460             if (!ntpStarted) {
461                 configTime(GMT_OFFSET, DST_OFFSET, NTP_SERVER);
462                 ntpStarted = true;
463                 Serial.println("[NTP] Sinkronisasi waktu dimulai...");
464             }
465
466             wifiState = WIFI_CONNECTED_STATE;
467         } else if (now - wifiAttemptStart >= WIFI_CONNECT_TIMEOUT) {
468             Serial.println("[WIFI] Gagal connect, akan dicoba lagi...");
469             wifiState = WIFI_IDLE; // kembali coba nanti, BUKAN menyerah permanen
470         }
471         break;
472

```

Gambar 4.26 Kode Proses Koneksi WiFi (State `WIFI_CONNECTING`)

Pada kondisi `WIFI_CONNECTED_STATE`, program memantau status koneksi secara terus-menerus. Apabila koneksi terdeteksi terputus, kondisi

dikembalikan ke WIFI_IDLE sehingga proses sambungan ulang dilakukan secara otomatis tanpa memerlukan intervensi dari pengguna. Untuk menjaga keberlangsungan komunikasi dengan server Blynk, fungsi Blynk.run() dipanggil secara berkala selama koneksi dalam keadaan aktif. Potongan kode tersebut ditampilkan pada Gambar 4.27.

```

472
473     case WIFI_CONNECTED_STATE:
474         if (WiFi.status() != WL_CONNECTED) {
475             Serial.println("[WIFI] Koneksi terputus! Akan reconnect...");
476             blynkCfg = false;
477             wifiState = WIFI_IDLE;
478         }
479         break;
480     }
481
482     bool wifiOK = (wifiState == WIFI_CONNECTED_STATE);
483
484     // ----- Jalankan Blynk kalau sudah terhubung -----
485     if (wifiOK && blynkCfg) {
486         Blynk.run();
487     }
488

```

Gambar 4.27 Kode Pemeliharaan Koneksi dan Deteksi Pemutusan WiFi

4.2.6 Pemrograman Pengiriman Data ke Blynk dan Google Sheets

Data suhu, kelembapan, dan duty cycle yang diperoleh dari proses pembacaan sensor dan perhitungan PID dikirimkan ke dua platform secara bersamaan, yaitu Blynk untuk keperluan pemantauan secara real-time dan Google Sheets untuk keperluan pencatatan data historis. Kedua pengiriman ini dilakukan dengan interval yang berbeda sesuai dengan fungsinya masing-masing, sebagaimana disajikan pada Tabel 4-6.

Tabel 4-6 Konfigurasi Interval Pengiriman Data

No	Platform	Interval	Metode
1	Blynk	2000 ms (2 detik)	Blynk.virtualWrite()
2	Google Sheets	60000 ms (60 detik)	HTTP GET ke Google Apps Script

Pengiriman data ke Blynk dilakukan dengan memanggil fungsi Blynk.virtualWrite() untuk setiap *virtual pin* yang telah dikonfigurasi pada subbab 4.2.1. Pengiriman hanya dijalankan apabila koneksi WiFi dalam keadaan aktif dan

status Blynk terdeteksi terhubung. Setiap kali interval pengiriman tercapai, tiga nilai dikirimkan secara berurutan, yaitu suhu pada *virtual pin* V0, kelembapan pada *virtual pin* V1, dan duty cycle pada *virtual pin* V2. Potongan kode pengiriman data ke Blynk ditampilkan pada Gambar 4.28.

```

498
499 // ----- Kirim ke Blynk, interval fleksibel -----
500 if (wifiOK && Blynk.connected() && (now - tBlynk >= BLYNK_INTERVAL)) {
501     tBlynk = now;
502     Blynk.virtualWrite(VPIN_SUHU, localSuhu);
503     Blynk.virtualWrite(VPIN_RH, localKelembapan);
504     Blynk.virtualWrite(VPIN_DUTY, localDuty);
505 }
506

```

Gambar 4.28 Kode Pengiriman Data ke Blynk

Pengiriman data ke Google Sheets tidak dilakukan secara langsung, melainkan melalui Google Apps Script yang berfungsi sebagai perantara antara ESP32 dan spreadsheet. Google Apps Script merupakan layanan dari Google yang memungkinkan pembuatan fungsi otomatis berbasis web, di mana pada penelitian ini digunakan untuk menerima data dari ESP32 dan menuliskannya ke dalam baris baru pada Google Sheets secara otomatis. Alamat URL Apps Script yang digunakan didefinisikan di dalam program sebagai berikut. Potongan kode pendefinisian URL ditampilkan pada Gambar 4.29 Kode Pendefinisian URL Google Apps Script.

```

51 // GOOGLE SHEETS
52 // =====
53 const char* SHEET_URL =
54     "https://script.google.com/macros/s/AKfycby-CbUpfLjqH86i5Gfr3XT3Dz6GmdAFv1quso0Gg_aUmD3H9bRzc2SMGEVgy38x04J2Yg/exec";
55

```

Gambar 4.29 Kode Pendefinisian URL Google Apps Script

Proses pengiriman dilakukan melalui HTTP GET request ke alamat URL tersebut. Sebelum data dikirimkan, stempel waktu berupa tanggal dan jam diperoleh terlebih dahulu dari hasil sinkronisasi NTP yang telah dikonfigurasi pada subbab 4.2.5. Data yang dikirimkan disusun sebagai parameter pada URL, yang terdiri atas tanggal, waktu, nilai suhu, dan nilai kelembapan. Batas waktu (*timeout*) untuk setiap permintaan HTTP ditetapkan selama 4000 milidetik. Apabila server memberikan respons dengan kode 200 atau 302, pengiriman dianggap berhasil. Potongan kode pengiriman data ke Google Sheets ditampilkan pada Gambar 4.30.

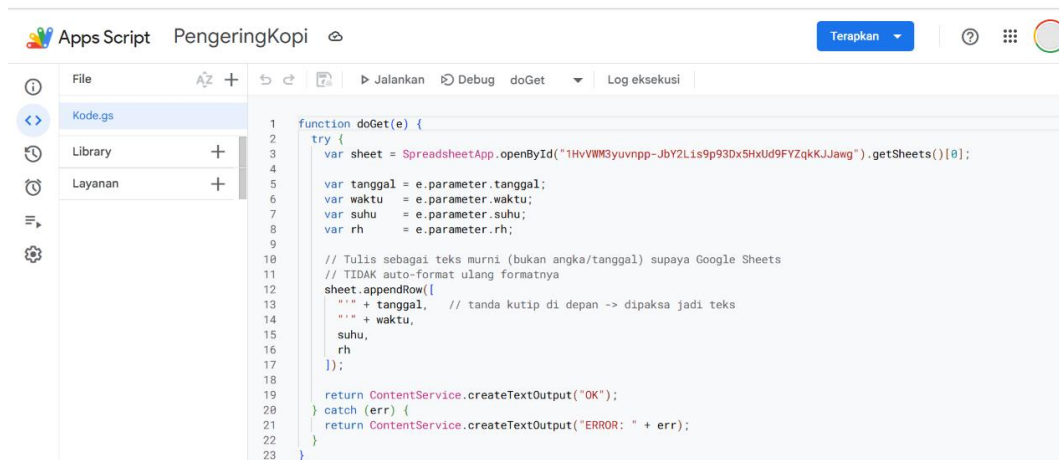
```

507 // ----- Kirim ke Google Sheets, interval fleksibel -----
508 if (wifiOK && (now - tSheet >= SHEET_INTERVAL)) {
509     tSheet = now;
510     struct tm timeinfo;
511     char tanggalStr[11] = "0000-00-00";
512     char waktuStr[9] = "00:00:00";
513     if (getLocalTime(&timeinfo, 100)) {
514         strftime(tanggalStr, sizeof(tanggalStr), "%Y-%m-%d", &timeinfo);
515         strftime(waktuStr, sizeof(waktuStr), "%H:%M:%S", &timeinfo);
516     }
517     HTTPClient http;
518     String url = String(SHEET_URL)
519         + "?tanggal=" + String(tanggalStr)
520         + "&waktu=" + String(waktuStr)
521         + "&suhu=" + String(localSuhu, 2)
522         + "&rh=" + String(localKelembapan, 1);
523     http.begin(url);
524     http.setTimeout(4000);
525     int code = http.GET();
526     http.end();
527     if (code == 200 || code == 302) {
528         Serial.printf("[SHEETS] OK (%s %s)\n", tanggalStr, waktuStr);
529     } else {
530         Serial.printf("[SHEETS] GAGAL code=%d\n", code);
531     }
532 }

```

Gambar 4.30 Kode Pengiriman Data ke Google Sheets.

Pada sisi Google Apps Script, fungsi doGet() digunakan untuk menerima parameter yang dikirimkan oleh ESP32, kemudian menuliskan data tersebut ke dalam baris baru pada spreadsheet. Tampilan kode Google Apps Script yang digunakan ditunjukkan pada Gambar 4.31.



Gambar 4.31 Kode Google Apps Script

Tabel 4-7 Parameter Data yang Dikirimkan ke Google Sheets

No	Parameter URL	Isi	Contoh
1	tanggal	Tanggal pengiriman (format YYYY-MM-DD)	2025-07-03
2	waktu	Waktu pengiriman (format HH:MM:SS, WIB)	14:30:05
3	suhu	Nilai suhu dalam °C (2 desimal)	64.25
4	rh	Nilai kelembapan relatif dalam % (1 desimal)	45.3

Keberhasilan pengiriman data ke kedua platform dapat diverifikasi secara langsung melalui tampilan masing-masing platform. Pada Blynk, data yang berhasil diterima ditampilkan pada *widget dashboard* sesuai dengan *virtual pin* yang telah dikonfigurasi. Tampilan *dashboard* Blynk saat menerima data ditunjukkan pada Gambar 4.31 Tampilan *Dashboard* Blynk Saat Menerima Data. Pada Google Sheets, data yang berhasil diterima tersimpan secara otomatis ke dalam baris baru pada spreadsheet sesuai dengan parameter yang dikirimkan. Tampilan Google Sheets yang telah terisi data ditunjukkan pada Gambar 4.32 Tampilan Data pada Google Sheets.