BAB IV
PEMBUATAN ALAT

4.1 Pembuatan Perangkat Keras *(Hardware)*

Pembuatan perangkat keras merupakan tahap awal dalam proses pengembangan sistem kontrol presisi suhu dan kelembapan pada inkubator jamur berbasis algoritma PID menggunakan mikrokontroler ESP32. Pada tahap ini dilakukan penyusunan, perakitan, serta penyambungan seluruh komponen elektronika dan mekanik agar sistem dapat berfungsi sesuai dengan rancangan yang telah dibuat pada tahap perencanaan. Perangkat keras ini dirancang agar mampu menjaga suhu dan kelembapan di dalam inkubator secara otomatis dan real-time melalui kendali aktuator yang terintegrasi.

Perangkat keras sistem ini menggunakan beberapa komponen utama, yaitu mikrokontroler ESP32 sebagai pusat kendali yang mengolah data sensor sekaligus menjalankan algoritma PID. Sensor DHT22 digunakan untuk mendeteksi suhu dan kelembapan udara di dalam inkubator. Data dari sensor tersebut diproses oleh ESP32 untuk dibandingkan dengan nilai setpoint yang telah ditentukan, sehingga menghasilkan sinyal kendali yang sesuai.

Aktuator utama yang dikendalikan dalam sistem ini meliputi *Exhaust Fan* 12V DC untuk menurunkan suhu dengan sirkulasi udara, Pompa 12V DC yang terhubung ke nozzle spray kabut untuk meningkatkan kelembapan secara merata, serta Peltier Cooler 12V DC sebagai pendingin tambahan ketika suhu lingkungan terlalu tinggi. Semua aktuator dikendalikan melalui MOSFET yang berfungsi sebagai sakelar elektronik berbasis sinyal PWM dari ESP32.

Untuk monitoring dan kontrol jarak jauh, perangkat keras ini dilengkapi dengan konektivitas WiFi bawaan ESP32 yang terhubung ke aplikasi Blynk. Melalui aplikasi ini, data suhu dan kelembapan dapat ditampilkan secara real-time, sementara pengguna juga memiliki opsi untuk melakukan kontrol manual terhadap aktuator. Dengan demikian, sistem ini tidak hanya bekerja otomatis, tetapi juga

memberikan fleksibilitas dan kemudahan bagi pengguna dalam proses budidaya jamur.

Seluruh komponen mendapatkan daya dari adaptor AC to DC 12V 5A, yang kemudian didistribusikan ke aktuator serta diturunkan tegangannya menggunakan modul buck converter LM2596 agar sesuai dengan kebutuhan ESP32 dan sensor. Dengan desain ini, perangkat keras mampu bekerja stabil dan andal dalam mendukung terciptanya lingkungan inkubator yang optimal untuk pertumbuhan jamur.

4.1.1 Alat dan Bahan

Pada pelaksanaan proyek tugas akhir ini, terdapat berbagai alat-alat sebagai pendukung mempermudah perakitan pembuatan *prototype* inkubator jamur dapat dilihat pada tabel 4-1.

Tabel 4. 1 Alat – alat yang digunakan

	Nama Alat	Jumlah
1	Mesin Gerindra	1 buah
2	Mesin Bor	1 buah
3	Alat Solder	1 buah
4	Gunting	1 buah
5	Meteran	1 buah
6	Tespen	1 buah
7	Kunci pas	1 buah
8	Tang kombinasi	1 buah
9	Multimeter	1 buah
10	Obeng type plus	1 buah
11	Obeng type minus	1 buah
12	Alat tulis	1 set

Tabel 4. 2 Bahan-bahan yang digunakan

No	Nama Bahan	Jumlah
1	PC817	5
2	Socket IC 8P	3
3	R 100 0,5W	3
4	R 10K 0,5W	3
5	1N5819	3
6	R 220 0,5W	5
7	R 4K7 0,5W	3
8	LM2596	1
9	ESP32 DEVKIT V1	1
10	Pinhead Female 1x40	1
11	Heatsink TO-220	3
12	IRLZ44N	3
13	JST XH 3P	2
14	KF45	4
15	DHT22	1
16	Spacer M3 1 cm	6
17	PSU 12V 5A	1
18	Durabox 22x15 cm	1
19	Socket AC Cord 3P	1
20	Kabel AC Cord 3P	1
21	Fuse 10A	2
22	PCB FIBER 10x20cm	1
23	FeCl	1
24	Kabel Pita	1
25	Kabel NYAF RED 1,5mm	3
26	Kabel NYAF BLACK 1,5mm	3
27	Solasi Bakar 1,5mm	2
28	EXHAUST FAN 12VDC	1
29	Pompa 12VDC + selang	1
30	Kit Peltier Pendingin	1

4.1.2 Pembuatan Perangkat Elektrik

Pada pembuatan perangkat elektrik ini merupakan tahap utama dalam proses perakitan sistem pengendalian suhu dan kelembapan pada inkubator jamur berbasis PID dengan mikrokontroler ESP32. Tahap ini bertujuan untuk merancang dan menyusun hubungan antar komponen elektronika agar sistem dapat bekerja secara

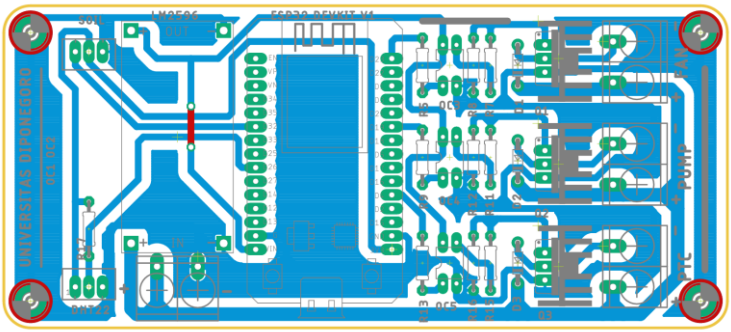
optimal sesuai dengan fungsinya. Adapun langkah-langkah pembuatan perangkat elektrik adalah sebagai berikut:

1. Perancangan skematik jalur dan layout PCB

Pembuatan skematik jalur dan layout PCB umumnya menggunakan beberapa software seperti Eagle, EasyEDA, atau KiCad. Pada perancangan kali ini digunakan software Eagle karena memiliki beberapa keunggulan, antara lain tampilan antarmuka yang cukup detail dan profesional, dukungan pustaka komponen yang luas, serta fleksibilitas tinggi dalam pengaturan skematik dan layout PCB. Eagle juga banyak digunakan dalam dunia industri maupun akademik sehingga hasil rancangan dapat dengan mudah dikembangkan lebih lanjut. Selain itu, software ini mendukung pembuatan file Gerber yang diperlukan untuk proses produksi PCB, serta menyediakan fitur autorouting yang membantu mempercepat proses perancangan jalur koneksi antar komponen secara akurat.

Pada skematik yang dirancang, seluruh komponen utama seperti ESP32, sensor DHT22, MOSFET driver, exhaust fan, pompa air dengan nozzle kabut, dan buck converter LM2596 disusun secara terstruktur agar koneksi antar komponen dapat dipahami dengan jelas. Perancangan ini juga memperhatikan jalur distribusi daya 12V dan 5V agar sesuai dengan kebutuhan setiap modul.

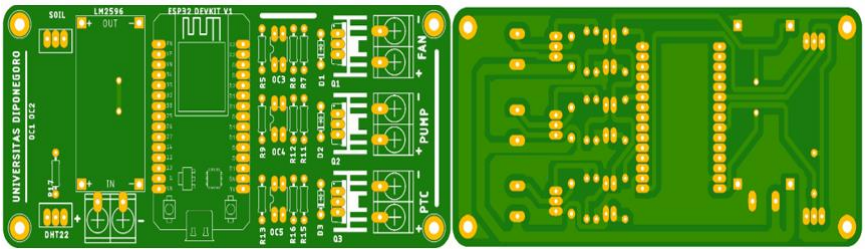
Gambar 4.1 menunjukkan hasil rancangan skematik jalur perangkat elektrik, di mana setiap komponen dihubungkan sesuai dengan fungsinya untuk mendukung sistem pengendalian suhu dan kelembapan inkubator jamur.



Gambar 4. 1 Skematik Jalur dan Layout PCB

2. Proses pecetakan dan Penyolderan PCB

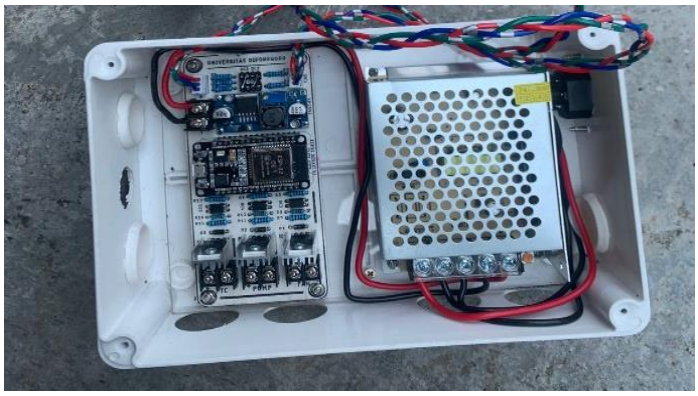
Proses pencetakan PCB pada penelitian ini menggunakan metode transfer setrika yang diawali dengan mencetak layout jalur rangkaian hasil desain dari software Eagle menggunakan printer laser pada media kertas glossy. Tahap awal dilakukan dengan membersihkan papan tembaga secara menyeluruh untuk menghilangkan lapisan oksidasi maupun kotoran yang dapat mengganggu proses transfer toner. Selanjutnya, kertas hasil cetakan ditempelkan pada permukaan papan tembaga, kemudian disetrika dengan suhu tinggi selama kurang lebih lima hingga sepuluh menit hingga toner berpindah dan menempel sempurna. Setelah proses penyetrikaan, papan direndam dalam air agar kertas mudah dilepaskan tanpa merusak pola jalur yang sudah terbentuk.



Gambar 4. 2 Pencetakan PCB Tampak Atas dan Bawah

3. Perakitan Komponen kedalam Panel Box

Pada perakitan komponen kedalam panel box dialaskan sebuah akrilik agar menghindari hubungan pendek (short), dan agar tersusun rapi pada akrilik di box panel dibutuhkan mur untuk memperkuat posisi komponen agar tetap stabil saat berjalannya sistem. Diapasangkanlah seluruh komponen seperti power supply, PCB, dan juga terminal block. Selanjutnya menyambungkan kabel pada setiap komponen ke komponen lainnya. Gambar 4.3 adalah hasil dari terpasangnya perakitan komponen kedalam box panel.



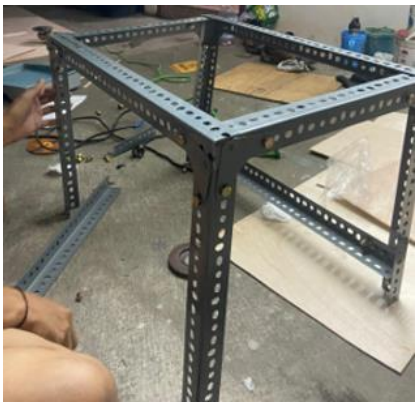
Gambar 4. 3 Perakitan Komponen kedalam Box

4.1.3 Pembuatan Perangkat Mekanik

Perangkat mekanik pada sistem kontrol dan monitoring suhu serta kelembapan pada inkubator jamur tiram, dirancang sebagai pendukung proses berjalannya sistem dan terintegrasi dengan elektronik. Seluruh bagian mekanik dibuat secara manual dengan bahan utama akrilik, besi siku, dan perintilan untuk menyokong berdirinya alat. Adapun langkah-langkah pembuatan perangkat mekanik adalah sebagai berikut:

1. Pemotongan dan Perakitan Besi Siku

Tahap awal dimulai dengan proses pemotongan sebuah besi siku menggunakan alat pemotong yaitu gerinda yang sudah diukur sesuai kebutuhan sisi masing-masing. Proses ini ditunjukan pada Gambar 4. 4 sebagai berikut:



Gambar 4. 4 Perakitan dan Pemasangan Rangka Inkubator Jamur

Kemudian proses perakitan rangka plat besi siku menjadi sebuah kubus persegi panjang yang membentuk kerangka inkubator jamur yang kokoh, rangka ini berguna sebagai penopang utama dari keseluruhan sistem perangkat. Setelah rangka selesai terpasang, Kemudian langkah berikutnya yaitu pemasangan akrilik yang sudah cutting sesuai sisi pada kerangka dan diberi lubang untuk penempatan komponen seperti Exhaust Fan, dan Peltier.

2. Pemasangan Komponen Pada Inkubator Jamur

Setelah rangka selesai, dilakukan pemasangan komponen elektronik dan aktuator komponen sesuai fungsinya. Terlihat Gambar 4.6 merupakan pada sisi bagian kiri dan kanan, yaitu pada sisi kiri telah dipasangkan Peltier dan terdapat panel box elektrikal yang didalamnya berupa rangkaian PCB, power supply, dan jalur kelistrikan. Pada sisi kanan telah terpasang *Exhaust Fan* sebagai keluaran udara.

Gambar 4. 5 Inkubator Jamur Tampak Samping Kiri dan Kanan

4.2 Pembuatan Perangkat Lunak *(Software)*

Perangkat lunak merupakan bagian penting dari sistem kontrol dan monitoring suhu serta kelembapan inkubator jamur, karena berfungsi sebagai pengendali utama dalam membaca data sensor, mengolah informasi, melakukan perhitungan kontrol PID, serta mengirimkan data secara real-time ke aplikasi Blynk.

Proses pembuatan perangkat lunak diawali dengan penyusunan algoritma dasar yang meliputi pengambilan data dari sensor DHT22 pengolahan data pada mikrokontroler ESP32, pengendalian aktuator berupa kipas exhaust dan pompa air, serta pengiriman data ke platform Blynk agar dapat dipantau melalui smartphone.

Program ditulis menggunakan bahasa C/C++ dengan dukungan Arduino IDE, karena mendukung pustaka yang luas, kemudahan dalam proses pemrograman, serta kompatibilitas dengan modul ESP32.

4.2.1 Pembuatan Program Mikrokontroler ESP32

```
// ================================================================
// Bagian 1: Inklusi Pustaka
// ================================================================
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <DHT.h>
#include "KREDWIFI.h"
```

Gambar 4. 6 Inklusi Pustaka

Gambar 4. 6 Pada tahap awal penulisan program dilakukan inklusi pustaka yang berfungsi untuk memanggil berbagai library serta file header yang dibutuhkan agar sistem dapat berjalan dengan baik. Pustaka WiFi.h digunakan untuk mengaktifkan fungsi komunikasi nirkabel pada ESP32 sehingga perangkat dapat terhubung ke jaringan internet. Selanjutnya, BlynkSimpleEsp32.h berperan penting dalam menghubungkan ESP32 dengan aplikasi Blynk agar data hasil pembacaan sensor dapat ditampilkan secara real-time pada smartphone pengguna. Untuk proses pembacaan suhu dan kelembapan, digunakan pustaka DHT.h yang mendukung sensor DHT22 sehingga nilai suhu dan kelembapan dapat diperoleh secara digital. Terakhir file KREDWIFI.h digunakan untuk menyimpan informasi kredensial jaringan seperti SSID dan password, sehingga keamanan lebih terjaga karena data sensitif tidak dituliskan langsung pada program utama. Dengan adanya pustaka-pustaka tersebut, program menjadi lebih modular, terstruktur, dan efisien dalam

mengintegrasikan fungsi komunikasi, pembacaan sensor, serta pengaturan perangkat keras.

```
// ============================================================
// Bagian 2: Konfigurasi Sistem
// ============================================================

// --- Kredensial Blynk ---
#define BLYNK_TEMPLATE_ID "TMPL6h8yI7PDF"
#define BLYNK_TEMPLATE_NAME "Sistem Monitoring Suhu dan Kelembapan Jamur"
#define BLYNK_AUTH_TOKEN "a4nD_krwSebXd6b7n_kjwF2s9HWG6HSK"

// --- Kredensial WiFi (dari KREDWIFI.h) ---
const char* ssid = SECRET_SSID;
const char* password = SECRET_PASS;

// --- Definisi Pin Hardware ---
const int DHT_PIN  = 26;      // Sensor suhu dan kelembapan udara (DHT22)
const int FAN_PIN  = 23;      // Kipas (Exhaust Fan)
const int PUMP_PIN = 19;      // Pompa air untuk nozzle kabut
const int PTC_PIN  = 4;       // Elemen pendingin PTC

// --- Konfigurasi Sensor dan Aktuator ---
#define DHT_TYPE DHT22
const float TEMP_OFFSET = -4.0; // Konstanta kalibrasi untuk sensor suhu
const int PUMP_PWM_POWER = 100; // Kekuatan pompa saat aktif (0-255)

// --- Konfigurasi PWM (LEDC) ---
const int FAN_CHANNEL    = 0;
const int PTC_CHANNEL    = 1;
const int PUMP_CHANNEL   = 2;
const int PWM_FREQ       = 5000;  // Frekuensi PWM dalam Hz
const int PWM_RESOLUTION = 8;     // Resolusi 8-bit (0-255)

// --- Parameter Kontrol PID (Hanya untuk Suhu) ---
double tempKp = 5.0;
double tempKi = 10.0;
double tempKd = 1.0;
```

Gambar 4. 7 Konfigurasi Sistem

Gambar 4. 7 bagian konfigurasi sistem, langkah pertama dilakukan pendefinisian kredensial Blynk yang terdiri dari Template ID, *Template Name*, dan *Authentication* Token. Parameter ini menjadi identitas unik yang menghubungkan perangkat ESP32 dengan server Blynk sehingga data dari sensor dapat ditransmisikan dan dipantau secara real-time melalui aplikasi. Selanjutnya, kredensial WiFi seperti SSID dan Password juga didefinisikan agar perangkat dapat terhubung ke jaringan internet.

Bagian berikutnya adalah pendefinisian pin input dan output untuk perangkat keras. Pin DHT_PIN dikhususkan untuk sensor DHT22 yang berfungsi membaca suhu dan kelembapan udara. Untuk aktuator, pin FAN_PIN dihubungkan ke kipas exhaust, pin PUMP_PIN digunakan untuk pompa air yang menggerakkan nozzle kabut, dan pin PTC_PIN disiapkan untuk elemen pendingin PTC. Dengan konfigurasi ini, masing-masing perangkat keras mendapatkan alamat pin yang jelas sehingga memudahkan dalam pengendalian melalui program.

Selanjutnya, dilakukan konfigurasi sensor dan aktuator dengan mendefinisikan tipe sensor DHT_TYPE sebagai DHT22, kemudian diberikan konstanta kalibrasi TEMP_OFFSET untuk menyesuaikan akurasi sensor suhu. Selain itu, ditentukan pula nilai PUMP_PWM_POWER yang mengatur kekuatan pompa saat aktif.

Agar aktuator dapat dikendalikan lebih presisi, digunakan sistem PWM (Pulse Width Modulation) dengan modul LEDC pada ESP32. Masing-masing perangkat seperti kipas, pompa, dan PTC memiliki channel tersendiri, dengan frekuensi operasi sebesar 5000 Hz dan resolusi 8-bit. Konfigurasi ini memungkinkan pengaturan kecepatan atau intensitas kerja aktuator secara lebih halus.

Terakhir, ditetapkan parameter kontrol PID yang hanya digunakan untuk pengaturan suhu. Parameter ini meliputi Kp (proportional gain), Ki (integral gain), dan Kd (derivative gain), yang masing-masing diberi nilai awal 5.0, 10.0, dan 1.0. Parameter inilah yang nantinya akan menentukan tingkat respons sistem dalam menjaga suhu inkubator jamur agar tetap stabil sesuai dengan set point yang telah ditentukan.

```
// ================================================================
// Bagian 3: Variabel Global & State
// ================================================================

// --- Objek Pustaka ---
DHT dht(DHT_PIN, DHT_TYPE);
BlynkTimer timer;

// --- Setpoint Kontrol (diatur melalui Blynk Dashboard) ---
double tempSetpoint = 22.0;          // Default untuk fase fruiting
double humidSetpointMax = 95.0;      // Batas atas kelembapan (pompa OFF)
double humidSetpointMin = 85.0;      // Batas bawah kelembapan (pompa ON)

// --- Variabel State Sistem ---
double currentTemp = 0;
double currentHumidity = 0;
int fanPower = 0;
int ptcPower = 0;
int pumpPower = 0;
bool isPumpOn = false;

// --- Variabel untuk Kalkulasi PID ---
double tempIntegralSum = 0;
double lastTempInput = 0;
unsigned long lastControlTime = 0;
```

Gambar 4. 8 Variabel Global dan State

Gambar 4. 8 variabel global dan state didefinisikan untuk mendukung proses pengendalian suhu dan kelembapan inkubator jamur tiram. Objek pustaka yang digunakan terdiri dari inisialisasi sensor DHT22 melalui perintah DHT dht(DHT_PIN, DHT_TYPE) untuk membaca suhu dan kelembapan udara, serta objek BlynkTimer timer yang berfungsi mengatur jadwal eksekusi fungsi secara periodik sehingga pembacaan sensor dan pengiriman data ke aplikasi Blynk dapat berjalan dengan interval yang konsisten. Selanjutnya, ditentukan variabel setpoint

kontrol yang dapat diatur langsung melalui dashboard Blynk, yaitu tempSetpoint sebagai acuan suhu inkubator dengan nilai default 22 °C untuk fase fruiting, humidSetpointMax sebagai batas atas kelembapan udara yang menandakan pompa harus berhenti menyemprotkan kabut, serta humidSetpointMin sebagai batas bawah kelembapan udara yang menandakan pompa harus kembali menyala.

Selain setpoint, terdapat variabel state sistem yang menyimpan data sensor dan kondisi aktuator. Variabel currentTemp dan currentHumidity digunakan untuk menyimpan hasil pembacaan sensor suhu dan kelembapan udara secara real-time. Sedangkan fanPower, ptcPower, dan pumpPower berfungsi untuk menyimpan nilai sinyal PWM yang dikirimkan ke aktuator berupa exhaust fan, modul Peltier, dan pompa air. Status pompa juga dipantau dengan variabel logika isPumpOn yang menunjukkan apakah pompa dalam keadaan aktif atau tidak.

Untuk mendukung algoritma kendali PID pada pengaturan suhu, ditambahkan variabel tempIntegralSum yang digunakan untuk menyimpan akumulasi perhitungan integral, lastTempInput yang menyimpan data suhu terakhir sebagai acuan perhitungan derivative, serta lastControlTime yang mencatat waktu eksekusi kontrol terakhir sehingga interval perhitungan dapat dihitung secara tepat. Dengan adanya struktur variabel global dan state ini, sistem mampu melakukan pemantauan, perbandingan terhadap setpoint, serta pengendalian aktuator secara terintegrasi dan adaptif, sehingga kestabilan suhu dan kelembapan pada inkubator jamur tiram dapat terjaga sesuai kondisi ideal untuk pertumbuhan jamur.

```
// ================================================================
// Bagian 4: Deklarasi Fungsi (Prototypes)
// ================================================================
void runControlLoop();
bool readSensors();
void controlHumidity();
int calculateTemperatureControl();
void applyActuatorOutputs(int tempControlOutput);
void sendDataToBlynk();
void printStatusToSerial();
```

Gambar 4. 9 Deklarasi Fungsi

Gambar 4. 9 ini didefinisikan prototipe fungsi-fungsi utama yang akan digunakan dalam program. Fungsi-fungsi ini diperlukan agar compiler mengetahui struktur dan tipe data yang akan dikembalikan sebelum implementasinya dituliskan.

Pertama, terdapat fungsi runControlLoop() yang berperan sebagai inti dari sistem kontrol, di mana seluruh proses pembacaan sensor, perhitungan kendali, hingga pengaturan aktuator akan dipanggil secara teratur. Selanjutnya, fungsi readSensors() bertugas membaca data dari sensor DHT22 lalu mengembalikan nilai logika true atau false untuk menandakan keberhasilan pembacaan.

Fungsi controlHumidity() digunakan untuk mengatur kelembapan dalam inkubator dengan cara menyalakan atau mematikan aktuator seperti pompa atau kipas sesuai hasil pembacaan sensor. Kemudian, terdapat fungsi calculateTemperatureControl() yang mengimplementasikan perhitungan kendali suhu berbasis PID, dan mengembalikan nilai integer sebagai output kontrol suhu yang nantinya akan digunakan untuk mengatur aktuator pemanas maupun pendingin.

Selanjutnya, fungsi applyActuatorOutputs(int tempControlOutput) bertugas menerima hasil keluaran kendali suhu dari fungsi PID lalu menerapkannya ke perangkat keras seperti kipas, PTC, atau pompa sesuai kondisi. Fungsi sendDataToBlynk() digunakan untuk mengirimkan data sensor maupun status perangkat secara real-time ke aplikasi Blynk, sehingga pengguna dapat memantau kondisi inkubator dari jarak jauh. Terakhir, fungsi printStatusToSerial() berfungsi untuk menampilkan data sensor, setpoint, serta status perangkat ke Serial Monitor sebagai alat debugging maupun evaluasi sistem secara langsung.

```
// ============================================================
// Bagian 5: Fungsi Setup (Inisialisasi)
// ============================================================

void setup() {
  Serial.begin(115200);
  Serial.println("\n[INFO] Sistem Monitoring Jamur Tiram v5.0
  sedang dimulai...");

  // Konfigurasi channel PWM untuk semua aktuator
  ledcSetup(FAN_CHANNEL, PWM_FREQ, PWM_RESOLUTION);
  ledcAttachPin(FAN_PIN, FAN_CHANNEL);

  ledcSetup(PTC_CHANNEL, PWM_FREQ, PWM_RESOLUTION);
  ledcAttachPin(PTC_PIN, PTC_CHANNEL);

  ledcSetup(PUMP_CHANNEL, PWM_FREQ, PWM_RESOLUTION);
  ledcAttachPin(PUMP_PIN, PUMP_CHANNEL);

  // Pastikan semua aktuator mati saat startup
  ledcWrite(FAN_CHANNEL, 0);
  ledcWrite(PTC_CHANNEL, 0);
  ledcWrite(PUMP_CHANNEL, 0);

  dht.begin();
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);

  lastControlTime = millis();
  lastTempInput = dht.readTemperature(); // Bacaan awal untuk turunan PID

  // Menjadwalkan loop kontrol utama untuk berjalan setiap 1 detik
  timer.setInterval(1000L, runControlLoop);

  Serial.println("[INFO] Setup selesai. Sistem berjalan.");
  Serial.println("[INFO] Setpoint dapat diatur melalui Blynk Dashboard:");
  Serial.println("       V1: Setpoint Suhu");
  Serial.println("       V2: Setpoint Kelembapan Minimum");
  Serial.println("       V3: Setpoint Kelembapan Maksimum");
}
```

Gambar 4. 10 Inisialisasi

Gambar 4. 10 Fungsi setup() pada program ESP32 bertujuan untuk melakukan proses inisialisasi awal sistem sebelum masuk ke loop utama. Pertama, komunikasi serial diaktifkan menggunakan Serial.begin(115200) untuk menampilkan informasi status sistem pada Serial Monitor. Setelah itu, konfigurasi PWM dilakukan untuk semua aktuator, yaitu exhaust fan, elemen pendingin Peltier, dan pompa kabut. Setiap aktuator diberikan channel khusus dengan memanfaatkan fungsi ledcSetup() dan ledcAttachPin(), sehingga intensitas kerjanya dapat dikendalikan secara bertingkat melalui sinyal PWM, bukan hanya dalam mode ON/OFF. Untuk memastikan keamanan, semua aktuator diatur dalam kondisi mati dengan memberikan nilai duty cycle 0 menggunakan ledcWrite(), sehingga tidak ada perangkat yang langsung aktif ketika sistem pertama kali dijalankan.

Selanjutnya, sensor DHT22 diinisialisasi menggunakan dht.begin(), sedangkan koneksi ke aplikasi Blynk dilakukan melalui Blynk.begin (BLYNK_AUTH_TOKEN, ssid, password). Dengan langkah ini, sistem dapat langsung membaca data suhu dan kelembapan udara sekaligus mengirimkan hasil pengukuran maupun menerima perintah kendali dari aplikasi Blynk secara real-time melalui jaringan WiFi. Variabel lastControlTime diisi dengan nilai millis() untuk mencatat waktu awal eksekusi kontrol, sedangkan lastTempInput menyimpan hasil pembacaan suhu pertama kali dari sensor sebagai acuan awal perhitungan PID.

Untuk menjamin pengendalian berjalan terjadwal, fungsi utama kontrol runControlLoop dijadwalkan agar dieksekusi setiap 1 detik dengan memanfaatkan timer.setInterval(1000L, runControlLoop). Terakhir, sistem memberikan beberapa pesan informasi melalui Serial Monitor, seperti konfirmasi bahwa setup berhasil,

daftar parameter setpoint yang dapat diatur melalui aplikasi Blynk, serta status kesiapan sistem. Dengan demikian, fungsi setup() tidak hanya menyiapkan perangkat keras dan perangkat lunak, tetapi juga memastikan bahwa seluruh komponen sistem dalam keadaan aman, stabil, dan siap menjalankan proses monitoring serta pengendalian inkubator jamur secara otomatis.

```
// ====================================================================
// Bagian 6: Loop Utama
// ====================================================================

void loop() {
  Blynk.run();
  timer.run();
}
```

Gambar 4. 11 Loop Utama

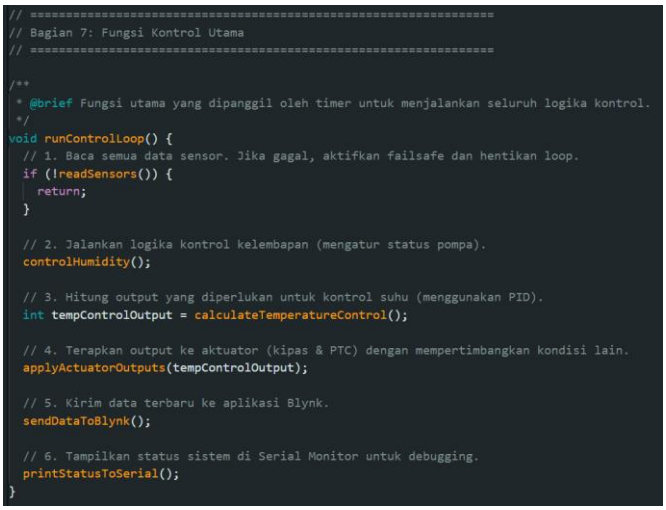
Gambar 4. 11 bagian Loop Utama, kode yang ditampilkan terlihat sederhana, namun sebenarnya memiliki peran yang sangat penting dalam menjaga kestabilan sistem kontrol dan monitoring suhu serta kelembapan inkubator jamur tiram. Fungsi loop() pada Arduino maupun ESP32 akan selalu berjalan secara berulang tanpa henti setelah proses setup selesai. Hal ini membuat semua instruksi di dalamnya akan terus dieksekusi selama perangkat aktif, sehingga sistem dapat bekerja secara real-time.

Instruksi pertama adalah Blynk.run();, yang berfungsi untuk menjaga koneksi antara mikrokontroler dengan server Blynk. Perintah ini memastikan komunikasi data dua arah berjalan dengan lancar, baik untuk mengirimkan data sensor (suhu dan kelembapan) ke aplikasi smartphone maupun menerima perintah dari pengguna, seperti perubahan setpoint atau pengendalian manual perangkat. Tanpa adanya pemanggilan rutin fungsi ini di dalam loop, aplikasi Blynk tidak akan menerima pembaruan data secara konsisten dan sistem pemantauan real-time akan terganggu.

Instruksi kedua adalah timer.run();, yang digunakan untuk mengeksekusi fungsi-fungsi yang telah dijadwalkan sebelumnya menggunakan objek BlynkTimer. Dalam program ini, salah satu fungsi yang dijadwalkan adalah

runControlLoop yang dieksekusi setiap 1 detik sekali. Fungsi inilah yang berisi logika utama sistem kontrol PID, yaitu membaca sensor, menghitung error, menentukan output kontrol, dan mengatur kerja aktuator (kipas, pompa kabut, atau pemanas). Dengan mekanisme timer ini, sistem dapat menjaga ritme kontrol agar berjalan konsisten, tanpa tergantung kecepatan loop utama.

Secara keseluruhan, meskipun loop utama hanya berisi dua baris kode, fungsinya sangat krusial sebagai "jantung sistem". Ia menjaga kestabilan koneksi dengan Blynk sekaligus memastikan semua proses terjadwal (seperti kontrol PID, pembacaan sensor, serta update data) berjalan tepat waktu. Desain loop yang minimalis ini juga memberikan keuntungan berupa efisiensi pemrosesan, karena sebagian besar tugas berat dialihkan ke timer dan event handler, sehingga mikrokontroler tidak terbebani dengan instruksi berulang yang tidak perlu.



```
// ============================================================
// Bagian 7: Fungsi Kontrol Utama
// ============================================================

/**
 * @brief Fungsi utama yang dipanggil oleh timer untuk menjalankan seluruh logika kontrol.
 */
void runControlLoop() {
  // 1. Baca semua data sensor. Jika gagal, aktifkan failsafe dan hentikan loop.
  if (!readSensors()) {
    return;
  }

  // 2. Jalankan logika kontrol kelembapan (mengatur status pompa).
  controlHumidity();

  // 3. Hitung output yang diperlukan untuk kontrol suhu (menggunakan PID).
  int tempControlOutput = calculateTemperatureControl();

  // 4. Terapkan output ke aktuator (kipas & PTC) dengan mempertimbangkan kondisi lain.
  applyActuatorOutputs(tempControlOutput);

  // 5. Kirim data terbaru ke aplikasi Blynk.
  sendDataToBlynk();

  // 6. Tampilkan status sistem di Serial Monitor untuk debugging.
  printStatusToSerial();
}
```

Gambar 4. 12 Fungsi Kontrol Utama

Gambar 4. 12 fungsi kontrol utama runControlLoop() ini adalah inti dari keseluruhan sistem pengendalian suhu dan kelembapan pada inkubator jamur. Fungsi ini dipanggil secara berkala oleh timer sehingga mampu menjaga kestabilan sistem dalam interval waktu tertentu. Berikut langkah-langkahnya secara detail:

1. Pembacaan Sensor (*readSensors()*)

Pada tahap pertama, sistem membaca semua data dari sensor, baik suhu maupun kelembapan. Jika proses pembacaan gagal (misalnya sensor tidak merespons atau menghasilkan nilai tidak valid), maka sistem akan langsung menghentikan eksekusi fungsi dengan failsafe menggunakan return. Mekanisme ini penting agar sistem tidak mengambil keputusan berdasarkan data yang salah, sehingga keamanan inkubator tetap terjaga.

2. Kontrol Kelembapan (*controlHumidity()*)

Setelah data berhasil dibaca, sistem menjalankan logika untuk mengendalikan kelembapan. Fungsi ini bertanggung jawab menyalakan atau mematikan pompa kabut sesuai kebutuhan. Misalnya, jika nilai kelembapan turun di bawah setpoint yang ditentukan, pompa akan aktif sehingga kabut air dilepaskan ke dalam inkubator. Sebaliknya, jika kelembapan sudah sesuai atau melebihi setpoint, pompa akan dimatikan.

3. Perhitungan Kontrol Suhu dengan PID (*calculateTemperatureControl()*)

Pada tahap ini, sistem menghitung output kontrol suhu dengan metode PID (Proportional-Integral-Derivative). Algoritma PID bekerja dengan membandingkan nilai suhu aktual dari sensor dengan suhu target (setpoint). Selisih keduanya disebut error. PID kemudian mengolah error ini dengan tiga komponen utama:

- Proportional (P): Merespons sebanding dengan besar error.
- Integral (I): Mengakumulasi error dari waktu ke waktu untuk mengurangi offset.
- Derivative (D): Memprediksi tren perubahan error agar respon lebih stabil.

Hasil perhitungan ini berupa nilai kontrol yang akan dipakai untuk mengatur kecepatan kipas atau intensitas pemanas.

4. Penerapan Output ke Aktuator (*applyActuatorOutputs()*)

Nilai keluaran dari PID kemudian diterapkan pada perangkat aktuator, yaitu kipas exhaust dan peltier (PTC). Fungsi ini memastikan bahwa aktuator bekerja

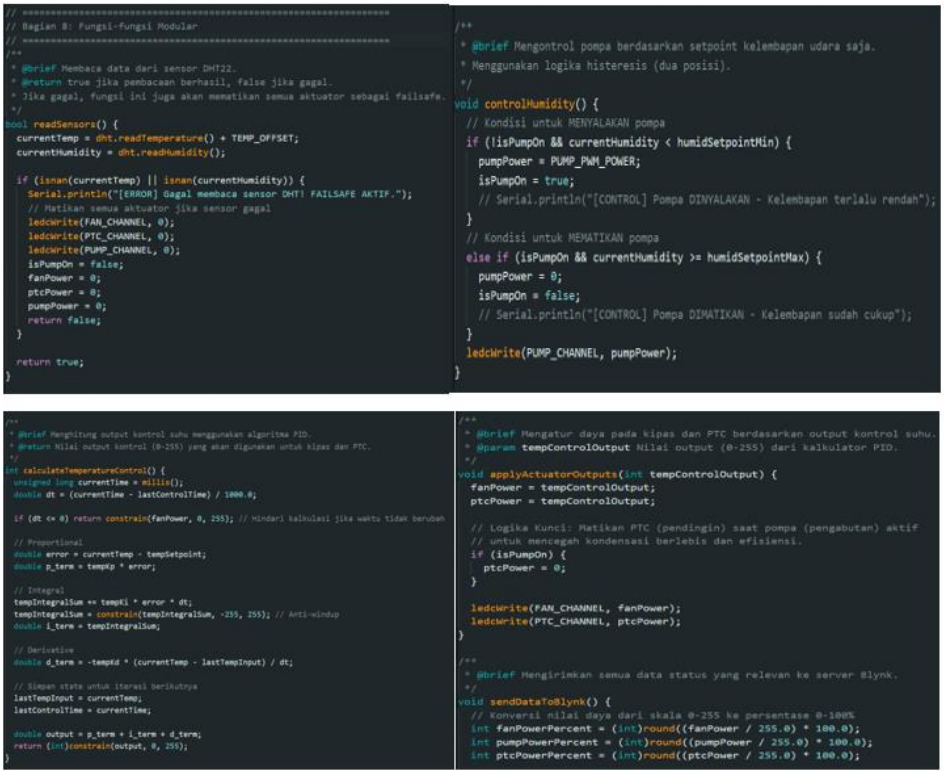
sesuai kebutuhan, dengan tetap mempertimbangkan kondisi lain seperti batas aman suhu maksimum. Misalnya, jika suhu terlalu tinggi, kipas akan diaktifkan lebih lama untuk mempercepat pendinginan.

5. Pengiriman Data ke Aplikasi Blynk (*sendDataToBlynk()*)

Setelah kontrol dilakukan, sistem mengirimkan data terbaru berupa nilai suhu, kelembapan, serta status aktuator ke aplikasi Blynk. Hal ini memungkinkan pengguna untuk memantau kondisi inkubator secara real-time melalui smartphone, sehingga kontrol dan pengawasan dapat dilakukan dari jarak jauh.

6. Menampilkan Status ke Serial Monitor (*printStatusToSerial()*)

Tahap terakhir adalah menampilkan informasi status sistem ke Serial Monitor. Data ini bermanfaat untuk proses debugging maupun pengujian, karena pengguna dapat melihat log aktivitas sistem, nilai sensor, serta perintah yang dijalankan oleh aktuator.



Gambar 4. 13 Fungsi Modular

Gambar 4. 13 merupakan bagian dari fungsi modular, Fungsi readSensors() bertugas membaca data suhu dan kelembapan dari sensor DHT22. Nilai suhu diambil melalui perintah dht.readTemperature() dengan penambahan offset tertentu, sedangkan nilai kelembapan udara dibaca menggunakan dht.readHumidity(). Untuk menjaga keandalan data, sistem melakukan pengecekan menggunakan fungsi isnan(). Jika pembacaan gagal atau menghasilkan data tidak valid, maka sistem akan masuk ke mode failsafe. Pada kondisi ini, seluruh aktuator seperti kipas, pompa, dan PTC otomatis dimatikan melalui perintah ledcWrite(), serta pesan kesalahan akan ditampilkan pada serial monitor agar pengguna mengetahui adanya gangguan. Dengan demikian, fungsi ini memastikan sensor bekerja secara andal dan sistem tetap aman apabila terjadi kegagalan pembacaan.

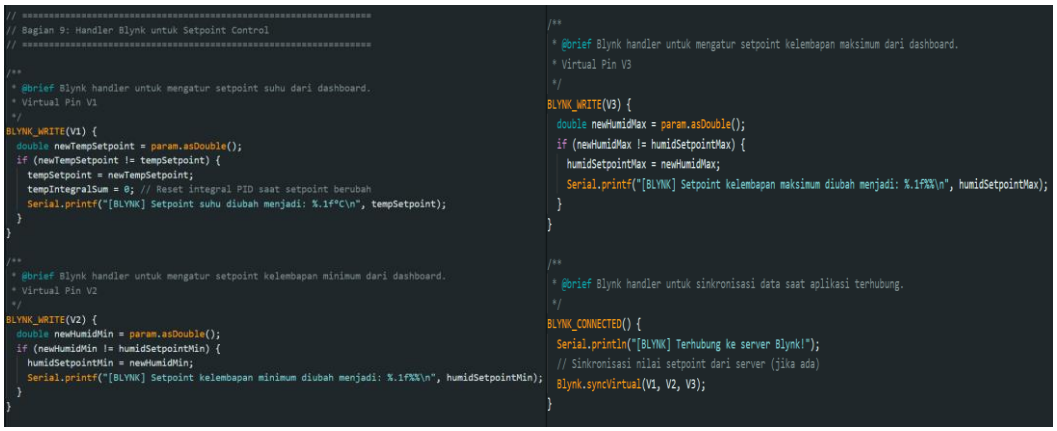
Fungsi controlHumidity() berfungsi mengatur pompa berdasarkan setpoint kelembapan udara dengan logika histeresis (dua posisi). Pompa akan menyala jika sedang dalam keadaan mati (!isPumpOn) dan nilai kelembapan udara lebih rendah dari batas bawah (humidSetpointMin). Ketika kondisi ini terpenuhi, variabel pumpPower diberi nilai PWM tertentu dan status pompa (isPumpOn) diubah menjadi true. Sebaliknya, pompa dimatikan apabila sedang dalam keadaan menyala (isPumpOn == true) dan kelembapan udara sudah melebihi batas atas (humidSetpointMax). Dalam hal ini, pumpPower diatur ke nol dan status pompa kembali false. Proses kendali ini menggunakan perintah ledcWrite() untuk mengirim sinyal ke aktuator, sehingga kelembapan tetap stabil sesuai setpoint.

Fungsi calculateTemperatureControl() digunakan untuk menghitung sinyal kendali suhu dengan algoritma PID (Proportional–Integral–Derivative). Fungsi ini pertama-tama menghitung selang waktu antar iterasi dengan memanfaatkan millis(), sehingga nilai kendali tetap proporsional terhadap waktu. Komponen proportional diperoleh dari selisih antara suhu aktual dengan setpoint, sedangkan integral dihitung dari akumulasi error terhadap waktu dengan pembatasan (anti-windup) menggunakan fungsi constrain(). Komponen derivative dihitung berdasarkan perubahan suhu terakhir terhadap waktu untuk meredam perubahan yang terlalu cepat. Hasil akhir PID berupa penjumlahan dari ketiga komponen (P +

I + D), kemudian dibatasi ke dalam rentang 0–255 agar sesuai dengan skala PWM. Nilai keluaran inilah yang akan digunakan untuk mengatur kipas dan PTC.

Fungsi applyActuatorOutputs() bertugas mengatur daya kipas (fanPower) dan PTC (ptcPower) berdasarkan nilai keluaran PID suhu. Nilai tersebut ditulis ke masing-masing kanal PWM menggunakan ledcWrite(). Terdapat logika tambahan yang mematikan PTC apabila pompa dalam kondisi aktif, untuk mencegah timbulnya kondensasi berlebih dan meningkatkan efisiensi energi. Dengan cara ini, sistem dapat menyeimbangkan antara proses pendinginan dan pengabutan.

Terakhir, fungsi sendDataToBlynk() digunakan untuk mengirimkan data kondisi sistem ke server Blynk agar dapat dipantau secara real-time melalui aplikasi. Data yang dikirim mencakup suhu, kelembapan udara, status pompa, serta daya kipas dan PTC. Sebelum dikirim, data daya dikonversi dari skala PWM 0–255 ke persentase 0–100 agar lebih mudah dibaca pada dashboard. Dengan adanya fungsi ini, pengguna dapat memantau kinerja sistem dari jarak jauh sekaligus memastikan bahwa seluruh proses pengendalian berjalan sesuai rancangan.



```
// ============================================================
// Bagian 9: Handler Blynk untuk Setpoint Control
// ============================================================

/**
 * @brief Blynk handler untuk mengatur setpoint suhu dari dashboard.
 * Virtual Pin V1
 */
BLYNK_WRITE(V1) {
  double newTempSetpoint = param.asDouble();
  if (newTempSetpoint != tempSetpoint) {
    tempSetpoint = newTempSetpoint;
    tempIntegralSum = 0; // Reset integral PID saat setpoint berubah
    Serial.printf("[BLYNK] Setpoint suhu diubah menjadi: %.1f°C\n", tempSetpoint);
  }
}

/**
 * @brief Blynk handler untuk mengatur setpoint kelembapan minimum dari dashboard.
 * Virtual Pin V2
 */
BLYNK_WRITE(V2) {
  double newHumidMin = param.asDouble();
  if (newHumidMin != humidSetpointMin) {
    humidSetpointMin = newHumidMin;
    Serial.printf("[BLYNK] Setpoint kelembapan minimum diubah menjadi: %.1f%%\n", humidSetpointMin);
  }
}
```

```
/**
 * @brief Blynk handler untuk mengatur setpoint kelembapan maksimum dari dashboard.
 * Virtual Pin V3
 */
BLYNK_WRITE(V3) {
  double newHumidMax = param.asDouble();
  if (newHumidMax != humidSetpointMax) {
    humidSetpointMax = newHumidMax;
    Serial.printf("[BLYNK] Setpoint kelembapan maksimum diubah menjadi: %.1f%%\n", humidSetpointMax);
  }
}

/**
 * @brief Blynk handler untuk sinkronisasi data saat aplikasi terhubung.
 */
BLYNK_CONNECTED() {
  Serial.println("[BLYNK] Terhubung ke server Blynk!");
  // Sinkronisasi nilai setpoint dari server (jika ada)
  Blynk.syncVirtual(V1, V2, V3);
}
```

Gambar 4. 14 Fungsi Utilitas dan Handler Blynk

Gambar 4. 14 Pada bagian ini, sistem dilengkapi dengan fungsi-fungsi yang memungkinkan pengguna untuk mengatur nilai setpoint suhu maupun kelembapan secara langsung dari aplikasi Blynk melalui virtual pin. Handler pertama dijalankan ketika terjadi perubahan pada Virtual Pin V1. Data masukan dari aplikasi dibaca dengan param.asDouble() dan disimpan pada variabel newTempSetpoint. Jika nilai

setpoint baru berbeda dengan setpoint lama, maka variabel tempSetpoint diperbarui dan komponen integral PID direset dengan cara mengosongkan nilai akumulasi integral (tempIntegralSum = 0). Hal ini bertujuan untuk mencegah osilasi kendali akibat adanya perubahan mendadak pada titik acuan. Informasi perubahan nilai setpoint suhu juga ditampilkan melalui serial monitor.

Selanjutnya, handler untuk Virtual Pin V2 digunakan untuk memperbarui batas kelembapan minimum yang diizinkan. Nilai baru disimpan ke dalam humiSetpointMin jika terjadi perubahan, kemudian ditampilkan pada serial monitor agar pengguna dapat memantau pembaruan secara langsung. Dengan mekanisme yang sama, handler pada Virtual Pin V3 mengatur batas kelembapan maksimum yang diterapkan dalam sistem. Selain itu, tersedia fungsi BLYNK_CONNECTED() yang dipanggil ketika perangkat berhasil terhubung ke server Blynk. Pada fungsi ini, dilakukan proses sinkronisasi nilai setpoint dari server ke perangkat melalui perintah Blynk.syncVirtual(V1, V2, V3). Dengan cara ini, perangkat akan selalu memperbarui dan menyesuaikan nilai setpoint suhu maupun kelembapan sesuai kondisi terakhir yang tersimpan di aplikasi, bahkan setelah perangkat melakukan restart atau kehilangan koneksi sementara. Dengan implementasi handler-handler ini, sistem kontrol lingkungan untuk budidaya jamur menjadi lebih fleksibel karena pengguna dapat secara langsung mengatur dan memantau parameter kendali dari aplikasi Blynk. Nilai setpoint suhu maupun batas kelembapan minimum dan maksimum dapat disesuaikan kapan saja tanpa perlu memodifikasi kode program secara manual, sehingga mendukung proses pertumbuhan jamur sesuai kebutuhan fase inkubasi maupun fruktifikasi.