

LAMPIRAN

Lampiran 1 Kode C++

```
// =====  
// BLYNK IOT CONFIGURATION  
// =====  
  
// Ganti nilai di bawah ini dengan Template ID, Name, dan Auth Token Blynk milik  
// Anda dari Dashboard Blynk:  
#define BLYNK_TEMPLATE_ID "TMPL6ogUsFjMV"  
#define BLYNK_TEMPLATE_NAME "MONITORING KUALITAS AIR N  
TANDON AIR"  
#define BLYNK_AUTH_TOKEN "zEINSX5fLWD6Yi14qXGXhBFotBv7uMAV"  
  
#define BLYNK_PRINT Serial  
  
#include <Arduino.h>  
#include <Wire.h>  
#include <LiquidCrystal_I2C.h>  
#include <Preferences.h>  
#include <WiFi.h>  
#include <WiFiClient.h>  
#include <BlynkSimpleEsp32.h>  
  
// Kredensial WiFi (Sesuaikan dengan SSID & Password WiFi Anda)  
char ssid[] = "iPhone";  
char pass[] = "000000000";  
  
// Fallback Auth Token jika BLYNK_AUTH_TOKEN belum di-define di atas  
char auth[] = "YourAuthToken";  
#ifndef BLYNK_AUTH_TOKEN
```

```

#define CURRENT_BLYNK_AUTH BLYNK_AUTH_TOKEN
#else
#define CURRENT_BLYNK_AUTH auth
#endif

```

```

BlynkTimer blynkTimer;

```

```

// =====

```

```

// PIN CONFIGURATION

```

```

// =====

```

```

#define PIN_PH      32 // Sensor pH (ADC1_CH4)
#define PIN_TURBIDITY 33 // Sensor Turbidity (ADC1_CH5)
#define PIN_TDS      34 // Sensor TDS (ADC1_CH6 - Input Only)
#define PIN_TRIG      5 // HC-SR04 Trig
#define PIN_ECHO      18 // HC-SR04 Echo
#define PIN_RELAY      23 // Relay Pompa Air
#define PIN_SDA      21 // LCD I2C SDA
#define PIN_SCL      22 // LCD I2C SCL

```

```

// Set Relay Active Level (Sesuaikan dengan modul relay: LOW = Active LOW,
HIGH = Active HIGH)

```

```

#define RELAY_ON LOW

```

```

#define RELAY_OFF HIGH

```

```

// =====

```

```

// TANK CONSTANTS

```

```

// =====

```

```

const float TANK_HEIGHT_CM = 15.0; // Tinggi tandon air 15 cm
const float LEVEL_MIN_FILL = 10.0; // Pompa ON jika air <= 10%
const float LEVEL_MAX_FILL = 80.0; // Pompa OFF jika air >= 80%

```

```
// =====
// OBJECTS & GLOBALS
// =====

LiquidCrystal_I2C lcd(0x27, 16, 2); // Alamat I2C umum: 0x27 (atau 0x3F)
Preferences prefs;

// Calibration variables (Saved in NVS)
struct CalibrationData {
    float ph_v7;      // Tegangan saat pH 7.00 (Volt)
    float ph_v4;      // Tegangan saat pH 4.01 (Volt)

    // 2-Point Linear TDS Calibration
    float tds_vhigh;   // Tegangan saat larutan tinggi (misal 1382 EC / 691 PPM)
    float tds_ppm_high; // Target PPM tinggi (default 691 PPM)
    float tds_vlow;    // Tegangan saat air jernih (misal Le Minerale ~130 PPM)
    float tds_ppm_low;  // Target PPM rendah (default 130 PPM)

    float turb_high_v; // Ambang batas tegangan Jernih (Volt)
    float turb_low_v;  // Ambang batas tegangan Keruh (Volt)
} cal;

// State Variables
bool relayState = false;
unsigned long lastSensorRead = 0;
unsigned long lastLcdUpdate = 0;
int lcdScreenPage = 0;

// Sensor Readings
float currentDistanceCm = 0;
float currentWaterLevelPct = 0;
float currentpH = 7.0;
```

```

float currentTDS = 0;
float currentTurbVoltage = 0;
String statuspH = "Normal";
String statusTDS = "Aman";
String statusTurbidity = "Jernih";

// =====
// FUNCTION DECLARATIONS
// =====

void loadCalibration();
void saveCalibration();
void resetCalibrationDefaults();
void printCalibrationInfo();
void processSerialCommands();

float readVoltage(int pin, int samples = 10);
float getUltrasonicDistanceCm();
void updateSensors();
void controlWaterTank();
void updateLCD();
void sendBlynkData();

// =====
// SETUP
// =====

void setup() {
    Serial.begin(115200);
    delay(1000);
    Serial.println(F("\n=====
));
    Serial.println(F("  ESP32 WATER QUALITY & TANK SYSTEM  "));

```

```

Serial.println(F("====="));

// Initialize Pins
pinMode(PIN_RELAY, OUTPUT);
digitalWrite(PIN_RELAY, RELAY_OFF);
pinMode(PIN_TRIG, OUTPUT);
pinMode(PIN_ECHO, INPUT);
digitalWrite(PIN_TRIG, LOW);

pinMode(PIN_PH, INPUT);
pinMode(PIN_TURBIDITY, INPUT);
pinMode(PIN_TDS, INPUT);

// Initialize I2C and LCD
Wire.begin(PIN_SDA, PIN_SCL);
lcd.init();
lcd.backlight();
lcd.setCursor(0, 0);
lcd.print(" IoT Monitoring ");
lcd.setCursor(0, 1);
lcd.print(" Kualitas Air ");
delay(2000);
lcd.clear();

// Load Calibration from NVS Preferences
loadCalibration();

// Initialize Blynk IoT & Timer
Serial.println(F("[BLYNK] Penghubungan ke WiFi & Blynk..."));
Blynk.begin(CURRENT_BLYNK_AUTH, ssid, pass);
blynkTimer.setInterval(1000L, sendBlynkData);

```

```

        Serial.println(F("Ketik 'HELP' pada Serial Monitor untuk daftar perintah
kalibrasi."));
    }

// =====
// MAIN LOOP
// =====

void loop() {
    Blynk.run();
    blynkTimer.run();

    // Process Serial commands for interactive calibration
    processSerialCommands();

    // Read sensors and run tank control logic every 500 ms
    if (millis() - lastSensorRead >= 500) {
        lastSensorRead = millis();
        updateSensors();
        controlWaterTank();
    }

    // Switch LCD screen page every 3 seconds
    if (millis() - lastLcdUpdate >= 3000) {
        lastLcdUpdate = millis();
        lcdScreenPage = (lcdScreenPage + 1) % 2;
        updateLCD();
    }
}

// =====

```

```

// BLYNK DATA TRANSMISSION (V0 - V5)
// =====
void sendBlynkData() {
    // V0: Nilai pH
    Blynk.virtualWrite(V0, currentpH);

    // V1: Nilai TDS (PPM)
    Blynk.virtualWrite(V1, currentTDS);

    // V2: Level Air Tandon (%)
    Blynk.virtualWrite(V2, currentWaterLevelPct);

    // V3: Turbidity Status (Huruf/String: Jernih / Agak Keruh / Keruh)
    Blynk.virtualWrite(V3, statusTurbidity);

    // V4: Status Kualitas Air (Ringkasan)
    String statusAir = "Normal";
    if (statuspH != "Normal" || statusTDS != "Aman" || statusTurbidity != "Jernih")
    {
        if (statuspH != "Normal") {
            statusAir = "pH " + statuspH;
        } else if (statusTDS != "Aman") {
            statusAir = "TDS Tinggi";
        } else {
            statusAir = statusTurbidity;
        }
    }
    Blynk.virtualWrite(V4, statusAir);

    // V5: Status Pompa (ON / OFF)
    Blynk.virtualWrite(V5, relayState ? "ON" : "OFF");

```

```

}

// =====
// SENSOR READING & LOGIC
// =====

// Helper: Read Average Voltage from ADC Pin (ESP32: 0 - 3.3V)
float readVoltage(int pin, int samples) {
    long sum = 0;
    for (int i = 0; i < samples; i++) {
        sum += analogRead(pin);
        delayMicroseconds(100);
    }
    float avgAdc = (float)sum / samples;
    // Konversi ADC (12-bit = 4095) ke Voltage (3.3V)
    return (avgAdc * 3.3) / 4095.0;
}

// Read HC-SR04 Ultrasonic Distance
float getUltrasonicDistanceCm() {
    digitalWrite(PIN_TRIG, LOW);
    delayMicroseconds(2);
    digitalWrite(PIN_TRIG, HIGH);
    delayMicroseconds(10);
    digitalWrite(PIN_TRIG, LOW);

    long duration = pulseIn(PIN_ECHO, HIGH, 30000); // Timeout 30ms (~5 meter)
    if (duration == 0) return TANK_HEIGHT_CM;      // Fallback jika tidak ada
    pantulan

    // Kecepatan suara = 0.0343 cm/us

```

```

float distance = (duration * 0.0343) / 2.0;
return distance;
}

// Read all sensors & process statuses
void updateSensors() {
    // 1. ULTRASONIC & WATER LEVEL
    currentDistanceCm = getUltrasonicDistanceCm();
    // Ketinggian air = Tinggi tandon (15cm) - Jarak sensor ke permukaan
    float waterHeight = TANK_HEIGHT_CM - currentDistanceCm;
    if (waterHeight < 0) waterHeight = 0;
    if (waterHeight > TANK_HEIGHT_CM) waterHeight = TANK_HEIGHT_CM;

    currentWaterLevelPct = (waterHeight / TANK_HEIGHT_CM) * 100.0;

    // 2. pH SENSOR (Linear 2-Point Calibration)
    float vPh = readVoltage(PIN_PH, 20);
    float slope = (7.00 - 4.01) / (cal.ph_v7 - cal.ph_v4);
    float intercept = 7.00 - (slope * cal.ph_v7);
    currentpH = (slope * vPh) + intercept;
    if (currentpH < 0.0) currentpH = 0.0;
    if (currentpH > 14.0) currentpH = 14.0;

    if (currentpH < 6.5) {
        statuspH = "Asam";
    } else if (currentpH > 8.5) {
        statuspH = "Basa";
    } else {
        statuspH = "Normal";
    }
}

```

```

// 3. TDS SENSOR (Linear 2-Point Calibration)
float vTds = readVoltage(PIN_PH == PIN_TDS ? PIN_TDS : PIN_TDS, 20);
float tdsSlope = (cal.tds_ppm_high - cal.tds_ppm_low) / (cal.tds_vhigh -
cal.tds_vlow);
if (cal.tds_vhigh == cal.tds_vlow) tdsSlope = 500.0; // Prevent div by 0
float tdsIntercept = cal.tds_ppm_low - (tdsSlope * cal.tds_vlow);

currentTDS = (tdsSlope * vTds) + tdsIntercept;
if (currentTDS < 0) currentTDS = 0;

if (currentTDS < 500.0) {
    statusTDS = "Aman";
} else {
    statusTDS = "Tinggi";
}

// 4. TURBIDITY SENSOR
// Ketentuan: Semakin tinggi nilai/tegangan => Semakin Jernih
// Semakin rendah nilai/tegangan => Semakin Keruh
currentTurbVoltage = readVoltage(PIN_TURBIDITY, 20);
if (currentTurbVoltage >= cal.turb_high_v) {
    statusTurbidity = "Jernih";
} else if (currentTurbVoltage >= cal.turb_low_v) {
    statusTurbidity = "Agak Keruh";
} else {
    statusTurbidity = "Keruh";
}

// Control Water Tank Pump Relay with Hysteresis
void controlWaterTank() {

```

```

// Logika pengisian tandon:
// Air <= 10% -> Relay ON (isi tandon)
// Air >= 80% -> Relay OFF (hentikan pengisian)
// Di antara 10% dan 80% -> Pertahankan status relay sebelumnya
if (currentWaterLevelPct <= LEVEL_MIN_FILL) {
    relayState = true;
} else if (currentWaterLevelPct >= LEVEL_MAX_FILL) {
    relayState = false;
}

digitalWrite(PIN_RELAY, relayState ? RELAY_ON : RELAY_OFF);
}

// Display Info on LCD
void updateLCD() {
    lcd.clear();
    if (lcdScreenPage == 0) {
        // SCREEN 1: MONITORING TANDON
        // Line 1: Air: 75.5% [ON/OFF]
        lcd.setCursor(0, 0);
        lcd.print("Air:");
        lcd.print(currentWaterLevelPct, 1);
        lcd.print("% ");
        lcd.print(relayState ? "[ON]" : "[OFF]");

        // Line 2: Tinggi: 11.3cm
        lcd.setCursor(0, 1);
        float waterHeight = TANK_HEIGHT_CM - currentDistanceCm;
        if (waterHeight < 0) waterHeight = 0;
        lcd.print("Tinggi: ");
        lcd.print(waterHeight, 1);
    }
}

```

```

    lcd.print(" cm");

} else {
    // SCREEN 2: KUALITAS AIR
    // Line 1: pH:7.2(Norm)
    lcd.setCursor(0, 0);
    lcd.print("pH:");
    lcd.print(currentpH, 1);
    lcd.print("(");
    lcd.print(statuspH);
    lcd.print(")");

    // Line 2: TDS:230 Turb:Jernih/Agak/Keruh
    lcd.setCursor(0, 1);
    lcd.print("TDS:");
    lcd.print((int)currentTDS);
    lcd.print(" ");
    if (statusTurbidity == "Jernih") lcd.print("Jernih");
    else if (statusTurbidity == "Agak Keruh") lcd.print("AgakK");
    else lcd.print("Keruh");
}
}

// =====
// PREFERENCES & CALIBRATION STORAGE (NVS)
// =====

void loadCalibration() {
    prefs.begin("water_iot", true); // Read-only mode
    cal.ph_v7    = prefs.getFloat("ph_v7", 1.65);    // Default ~1.65V
    cal.ph_v4    = prefs.getFloat("ph_v4", 2.03);    // Default ~2.03V

```

```

cal.tds_vhigh = prefs.getFloat("tds_vh", 1.40);    // Default ~1.40V
cal.tds_ppm_high = prefs.getFloat("tds_ph", 691.0);    // Default 691 PPM
(1382 EC)
cal.tds_vlow = prefs.getFloat("tds_vl", 0.30);    // Default ~0.30V
cal.tds_ppm_low = prefs.getFloat("tds_pl", 130.0);    // Default 130 PPM (Le
Minerale)
cal.turb_high_v = prefs.getFloat("turb_high", 2.50);    // Tegangan batas Jernih
(>2.5V)
cal.turb_low_v = prefs.getFloat("turb_low", 1.50);    // Tegangan batas Agak
Keruh (1.5V - 2.5V)
prefs.end();

Serial.println(F("[NVS] Data kalibrasi berhasil dimuat dari NVS Flash."));
printCalibrationInfo();
}

void saveCalibration() {
  prefs.begin("water_iot", false); // Read-write mode
  prefs.putFloat("ph_v7", cal.ph_v7);
  prefs.putFloat("ph_v4", cal.ph_v4);
  prefs.putFloat("tds_vh", cal.tds_vhigh);
  prefs.putFloat("tds_ph", cal.tds_ppm_high);
  prefs.putFloat("tds_vl", cal.tds_vlow);
  prefs.putFloat("tds_pl", cal.tds_ppm_low);
  prefs.putFloat("turb_high", cal.turb_high_v);
  prefs.putFloat("turb_low", cal.turb_low_v);
  prefs.end();

  Serial.println(F("[NVS] Data kalibrasi BERHASIL DISIMPAN ke NVS
Flash!"));
  printCalibrationInfo();
}

```

```
}

```

```
void resetCalibrationDefaults() {
    cal.ph_v7      = 1.65;
    cal.ph_v4      = 2.03;
    cal.tds_vhigh   = 1.40;
    cal.tds_ppm_high = 691.0;
    cal.tds_vlow    = 0.30;
    cal.tds_ppm_low  = 130.0;
    cal.turb_high_v  = 2.50;
    cal.turb_low_v   = 1.50;
    saveCalibration();
    Serial.println(F("[NVS] Kalibrasi di-reset ke nilai default pabrik.));
}

```

```
void printCalibrationInfo() {
    Serial.println(F("-----"));
    Serial.println(F("    NILAI KALIBRASI AKTIF (NVS)    "));
    Serial.println(F("-----"));
    Serial.print(F("pH 7.00 Voltage      : ")); Serial.print(cal.ph_v7, 3);
    Serial.println(F(" V"));
    Serial.print(F("pH 4.01 Voltage      : ")); Serial.print(cal.ph_v4, 3);
    Serial.println(F(" V"));
    Serial.print(F("TDS High (Larutan)   : ")); Serial.print(cal.tds_ppm_high, 1);
    Serial.print(F(" PPM @ ")); Serial.print(cal.tds_vhigh, 3); Serial.println(F(" V"));
    Serial.print(F("TDS Low (Air Jernih) : ")); Serial.print(cal.tds_ppm_low, 1);
    Serial.print(F(" PPM @ ")); Serial.print(cal.tds_vlow, 3); Serial.println(F(" V"));
    Serial.print(F("Turbidity High (Jernih): > ")); Serial.print(cal.turb_high_v, 2);
    Serial.println(F(" V"));
    Serial.print(F("Turbidity Low (Keruh) : < ")); Serial.print(cal.turb_low_v, 2);
    Serial.println(F(" V"));
}

```

```

    Serial.println(F("-----"));
}

// =====
// SERIAL CALIBRATION MENU INTERFACE
// =====

void processSerialCommands() {
    if (!Serial.available()) return;

    String input = Serial.readStringUntil('\n');
    input.trim();
    input.toUpperCase();

    if (input == "HELP") {
        Serial.println(F("\n==== MENU PERINTAH SERIAL KALIBRASI ===="));
        Serial.println(F("SHOW          : Tampilkan nilai kalibrasi & sensor realtime"));
        Serial.println(F("CAL_PH7       : Kalibrasi pH 7.00 (Celupkan di larutan pH 7)"));
        Serial.println(F("CAL_PH4       : Kalibrasi pH 4.01 (Celupkan di larutan pH 4)"));
        Serial.println(F("CAL_TDS <ppm>  : Kalibrasi TDS tinggi (misal: CAL_TDS 691 pada larutan 1382)"));
        Serial.println(F("CAL_TDS_LOW <ppm>: Kalibrasi TDS rendah (misal: CAL_TDS_LOW 130 pada Le Minerale)"));
        Serial.println(F("CAL_TURB_HIGH  : Set tegangan saat ini sebagai batas air JERNIH"));
        Serial.println(F("CAL_TURB_LOW   : Set tegangan saat ini sebagai batas air KERUH"));
        Serial.println(F("RESET          : Reset semua kalibrasi ke default"));
        Serial.println(F("=====\n"));
    }
}

```

```

    } else if (input == "SHOW") {
        printCalibrationInfo();
        Serial.print(F("Raw Tegangan pH      : ")); Serial.print(readVoltage(PIN_PH),
3); Serial.println(F(" V"));
        Serial.print(F("Raw      Tegangan      TDS              : "));
Serial.print(readVoltage(PIN_TDS), 3); Serial.println(F(" V"));
        Serial.print(F("Raw      Tegangan      Turbidity:      "));
Serial.print(readVoltage(PIN_TURBIDITY), 3); Serial.println(F(" V"));
        Serial.print(F("Level Tandon      : ")); Serial.print(currentWaterLevelPct,
1); Serial.println(F(" %"));

    } else if (input == "CAL_PH7") {
        float v = readVoltage(PIN_PH, 30);
        cal.ph_v7 = v;
        saveCalibration();
        Serial.print(F("[CAL] Kalibrasi pH 7.00 tersimpan pada tegangan: "));
        Serial.print(v, 3); Serial.println(F(" V"));

    } else if (input == "CAL_PH4") {
        float v = readVoltage(PIN_PH, 30);
        cal.ph_v4 = v;
        saveCalibration();
        Serial.print(F("[CAL] Kalibrasi pH 4.01 tersimpan pada tegangan: "));
        Serial.print(v, 3); Serial.println(F(" V"));

    } else if (input.startsWith("CAL_TDS_LOW")) {
        int spaceIndex = input.indexOf(' ');
        float targetPpm = 130.0;
        if (spaceIndex > 0) {
            targetPpm = input.substring(spaceIndex + 1).toFloat();

```

```

    }
    float v = readVoltage(PIN_TDS, 30);
    cal.tds_vlow = v;
    cal.tds_ppm_low = targetPpm;
    saveCalibration();
    Serial.print(F("[CAL] Kalibrasi TDS LOW ("));
    Serial.print(targetPpm, 1);
    Serial.print(F(" PPM) tersimpan pada tegangan: "));
    Serial.print(v, 3); Serial.println(F(" V"));

} else if (input.startsWith("CAL_TDS")) {
    int spaceIndex = input.indexOf(' ');
    float targetPpm = 691.0;
    if (spaceIndex > 0) {
        targetPpm = input.substring(spaceIndex + 1).toFloat();
    }
    float v = readVoltage(PIN_TDS, 30);
    cal.tds_vhigh = v;
    cal.tds_ppm_high = targetPpm;
    saveCalibration();
    Serial.print(F("[CAL] Kalibrasi TDS HIGH ("));
    Serial.print(targetPpm, 1);
    Serial.print(F(" PPM) tersimpan pada tegangan: "));
    Serial.print(v, 3); Serial.println(F(" V"));

} else if (input == "CAL_TURB_HIGH") {
    float v = readVoltage(PIN_TURBIDITY, 30);
    cal.turb_high_v = v;
    saveCalibration();
    Serial.print(F("[CAL] Ambang batas JERNIH tersimpan pada tegangan: "));
    Serial.print(v, 3); Serial.println(F(" V"));
}

```

```
} else if (input == "CAL_TURB_LOW") {  
    float v = readVoltage(PIN_TURBIDITY, 30);  
    cal.turb_low_v = v;  
    saveCalibration();  
    Serial.print(F("[CAL] Ambang batas KERUH tersimpan pada tegangan: "));  
    Serial.print(v, 3); Serial.println(F(" V"));  
  
} else if (input == "RESET") {  
    resetCalibrationDefaults();  
} else {  
    Serial.println(F("Perintah tidak dikenali. Ketik 'HELP' untuk daftar  
perintah."));  
}  
}
```



Lampiran 2 Kartu Bimbingan



UNIVERSITAS PUTRA BANGSA

Kampus Pusat : Jl. Ronggowarsito 18 Pejagoan Kebumen, Telp. 0287-384011
Kampus Dua : Jl. Raya Buntu - Gombong KM 05 Kemranjen Banyumas, Telp. 0287-5296662

Kartu Bimbingan Skripsi

Nama : Faiz Fadhlurrohmah
NIM : 220202665
Program Studi : Sarjana Ilmu Komputer
Fakultas : Sains dan Teknologi
Dosen Pembimbing : Awaludin Abid, S.Kom, M.Kom

No	Tanggal	Materi Bimbingan	Status
1	07 November 2025	kosnsultasi judul	Valid
2	24 November 2025	konsultasi terkait topik permasalahan	Valid
3	11 Desember 2025	judul baru	Valid
4	14 Januari 2026	BAB 1	Valid
5	11 Februari 2026	BAB 1 DAN BAB 2	Valid
6	25 Februari 2026	BAB 2 DAN BAB 3	Valid
7	04 Maret 2026	BAB 3	Valid
8	07 Maret 2026	BAB 3	Valid
9	10 Maret 2026	BAB 3	Valid
10	22 April 2026	BAB 3	Valid
11	05 Mei 2026	Flowchar dan DFD	Valid
12	18 Mei 2026	BAB 3	ACC Seminar
13	13 Juni 2026	Revisian seminar proposal	Valid
14	17 Juli 2026	BAB 4	Valid
15	22 Juli 2026	BAB 4	Valid
16	29 Juli 2026	Bab 4 dan 5	Valid
17	03 Agustus 2026	BAB 4 dan 5	ACC Sidang

Kebumen, 04 Agustus 2026
Dosen Pembimbing



Awaludin Abid, S.Kom, M.Kom
NIDN. 0619129003

Lampiran 3 Kartu Tanda Peserta Seminar Proposal



UNIVERSITAS PUTRA BANGSA
PROGRAM STUDI ILMU KOMPUTER (SI)
KARTU TANDA PESERTA SEMINAR PROPOSAL SKRIPSI

Nama : Faiz Fadhilurrahman
NIM : 220202665

NO	PEMAKALAH			TANGGAL	PARAF MODERATOR
	NAMA MAHASISWA	NIM	NAMA MODERATOR		
1.	Yulia Dewi	210202589	Awalyudin Abid, S.Kom, M.Kom Miftahul Huda, S.Pd, M.Kom	3 Juni 2025	
2.	Hikmal Adma	210202599	Awalyudin Abid, S.Kom, M.Kom Miftahul Huda, S.Pd, M.Kom	3 Juni 2025	
3.	Kharisma Khoerunnisa	210202573	Lolanda Hamim Annisa, S.Kom, M.Kom Miftahul Huda, S.Pd, M.Kom	9 Juli 2025	
4.	Maeta Ayuning Tias	210202603	Miftahul Huda, S.Pd, M.Kom Lolanda Hamim Annisa, S.Kom, M.Kom	9 Juli 2025	
5.	Andy Irawan Saputra	210202557	Lolanda Hamim Annisa, S.Kom, M.Kom Miftahul Huda, S.Pd, M.Kom	9 Juli 2025	
6.	Rahman Zul Iman	210202605	Yulianto, S.Kom, M.Kom Sargjimin, S.Kom, M.Kom	9 Juli 2025	
7.	Sukriso Pamungkas	210202585	Yulianto, S.Kom, M.Kom Sargjimin, S.Kom, M.Kom	9 Juli 2025	

Catatan :

- Kartu ini dibawa pada waktu mengikuti kegiatan seminar
- Kartu ini harus diserahkan kepada moderator untuk diparat sebagai tanda bukti
- Sebelum Mahasiswa membawakan seminar diharuskan mengikuti kegiatan seminar minimal 6 kali
- Mahasiswa yang dapat mengikuti kegiatan seminar ialah semester V keatas
- Kartu ini sebagai salah satu syarat ujian skripsi

Ketypen,
Ketua Prodi Ilmu Komputer

Awalyudin Abid, S.Kom, M.Kom
NIDN: 0619129003