📘 DOKUMENTASI SISTEM KRAN WUDHU OTOMATIS

Praktik IoT 2 - Remaja Masjid Al-Ikhlas (IRMAL)

📋 DAFTAR ISI

🚰 A. PENGANTAR PROYEK

Smart Wudhu Faucet System adalah proyek IoT tingkat lanjut yang dirancang untuk mengotomatisasi kran wudhu di masjid menggunakan teknologi sensor jarak laser (ToF - Time of Flight). Sistem ini mendeteksi kehadiran tangan secara akurat dan mengontrol aliran air secara otomatis dengan fitur keamanan berlapis.

Mengapa Proyek Ini Penting?

- ✅ **Hemat Air**: Air hanya mengalir saat tangan terdeteksi
- ✅ **Higienis**: Tanpa sentuhan fisik (touchless)
- ✅ **Presisi Tinggi**: Sensor laser ToF akurasi ±3mm
- ✅ **Safety Features**: Timeout otomatis, error recovery
- ✅ **Production Ready**: Kode level industri dengan monitoring

Keunggulan Dibanding Project Sebelumnya:

Aspek	Project 1 (Tanaman)	Project 2 (Kran Wudhu)
Sensor	Analog (Capacitive)	Digital ToF (Laser)
Presisi	±5%	±3mm
Komunikasi	I2C (1 device)	I2C (2 devices)
Web Interface	Basic HTML	Dashboard dengan QR
Security	Password static	Dynamic password + auth
Error Handling	Basic	Advanced dengan recovery
Mode	3 mode	5 mode + degraded

🎯 B. TUJUAN PEMBELAJARAN

Melalui proyek ini, anggota IRMAL akan mempelajari:

1. Sensor ToF (Time of Flight)

- Prinsip kerja sensor laser VL53L0X
- Pengukuran jarak berbasis waktu tempuh cahaya
- Multiple sensor I2C dengan address switching
- Continuous reading mode vs single shot

2. Advanced I2C Management

- Multiple devices pada satu bus I2C
- Dynamic address assignment (XSHUT pins)
- Error detection dan timeout handling
- Clock speed optimization (400kHz)

3. Gesture Recognition

- Deteksi swipe (sapuan tangan)
- Debouncing untuk menghindari false trigger
- Multiple swipe counting untuk AP mode
- State machine untuk gesture tracking

4. Production-Level Code

- Watchdog timer untuk reliability
- CRC checksum untuk data integrity

- EEPROM management dengan struktur data
- Error logging dan statistics
- Memory management dan optimization

5. Web Dashboard

- Responsive HTML dengan template processing
- RESTful API (GET/POST endpoints)
- JSON data format
- Authentication system
- QR code generation untuk quick access

6. Safety & Reliability

- Maximum water time protection
- Sensor failure detection
- Automatic recovery mechanism
- Degraded mode operation
- Factory reset capability

🔧 C. KOMPONEN & SPESIFIKASI

Hardware yang Dibutuhkan:

No	Komponen	Spesifikasi	Jumlah	Fungsi
1	ESP32 WROOM 32D	Microcontroller dual-core	1	Otak sistem
2	VL53L0X ToF Sensor	Laser distance, I2C	2	Deteksi tangan & swipe
3	OLED Display	0.96″, 128x64, I2C	1	Status display
4	Relay Module	5V, 1-4 Channel (pakai 1)	1	Kontrol solenoid valve
5	Solenoid Valve	12V DC, normally closed	1	Valve air otomatis
6	LED Indicator	5mm, warna bebas	1	Status visual
7	Push Button	Momentary, normally open	1	Factory reset
8	Power Supply	12V/2A (atau 5V/2A + 12V/1A)	1	Sumber daya
9	Step-down converter	12V to 5V, 2A	1	Power untuk ESP32
10	Enclosure/Box	Waterproof IP65	1	Housing elektronik

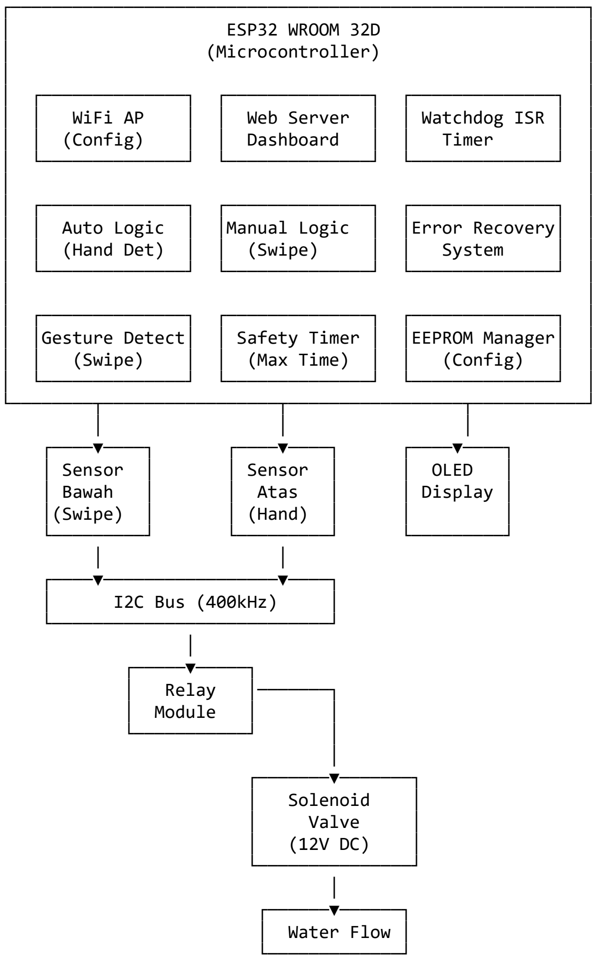
Software yang Dibutuhkan:

- Arduino IDE (v1.8.19 atau lebih baru)
- ESP32 Board Manager
- Library tambahan:
  - `Wire` (built-in)
  - `VL53L0X` (Pololu)
  - `Adafruit_GFX`
  - `Adafruit_SSD1306`
  - `WiFi` (ESP32 built-in)
  - `WebServer` (ESP32 built-in)
  - `EEPROM` (ESP32 built-in)
  - `qrcode` (Richard Moore)
  - `ArduinoJson` (Benoit Blanchon)

Spesifikasi Teknis:

```
ESP32:
- Clock: 240 MHz dual-core
- RAM: 520 KB
- Flash: 4 MB
- WiFi: 802.11 b/g/n
- I2C: Hardware support
- ADC: 12-bit (tidak dipakai)

VL53L0X:
- Range: 30mm - 2000mm
- Accuracy: ±3mm @ 50-1000mm
- Field of View: 25°
- Update Rate: Max 50Hz
- Interface: I2C (400kHz)
- Default Address: 0x29
- XSHUT: Active low shutdown

Solenoid Valve:
- Voltage: 12V DC
- Current: ~500mA
- Type: Normally closed (NC)
- Response: <100ms
- Pressure: 0-8 bar
```

🏗️ D. ARSITEKTUR SISTEM

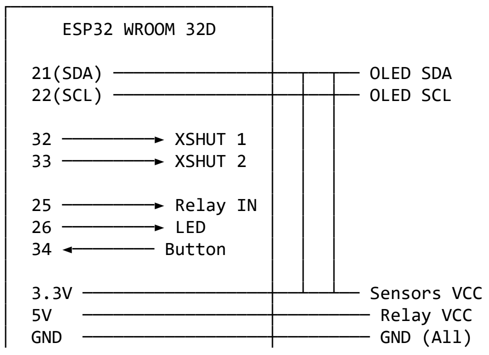


📌 E. SKEMA KONEKSI PIN

ESP32 Pin Assignment:

```
ESP32 GPIO → Komponen
────────────────────────────────────────────
GPIO 21 (SDA)     → OLED & Sensor 1 & 2 (SDA)
GPIO 22 (SCL)     → OLED & Sensor 1 & 2 (SCL)
GPIO 25           → Relay IN (Aktif LOW)
GPIO 26           → LED Indicator
GPIO 32           → Sensor 1 XSHUT (Address control)
GPIO 33           → Sensor 2 XSHUT (Address control)
GPIO 34           → Factory Reset Button (Pull-up)

3.3V  → VCC Sensor 1 & 2, OLED
5V    → VCC Relay Module
12V   → Solenoid Valve (via Relay)
GND   → GND semua komponen
```

Diagram Koneksi Detail:



```
└──────────────────────────┘

┌──────────────┐       ┌──────────────┐
│ VL53L0X #1   │       │ VL53L0X #2   │
│ (Sensor      │       │ (Sensor      │
│  Bawah)      │       │  Atas)       │
│              │       │              │
│XSHUT─GPIO32  │       │XSHUT─GPIO33  │
│SDA  ─GPIO21  │       │SDA  ─GPIO21  │
│SCL  ─GPIO22  │       │SCL  ─GPIO22  │
│VCC  ─3.3V    │       │VCC  ─3.3V    │
│GND  ─GND     │       │GND  ─GND     │
└──────────────┘       └──────────────┘

        ┌───────────────┐
        │ Relay Module  │
        │               │
        │ IN  ─ GPIO25  │
        │ VCC ─ 5V      │
        │ GND ─ GND     │
        │               │
        │ COM ─┐        │
        │ NO  ─┼───┐    │
        └──────┘   │────┘
                   │
            ┌──────▼──────┐
            │  Solenoid   │
            │   Valve     │
            │   12V DC    │
            └──────┬──────┘
                   │
            12V Power (+)

            GND ───┘
```

Penjelasan XSHUT (Address Switching):

VL53L0X memiliki default address `0x29`. Karena kita pakai 2 sensor, kita perlu ubah address salah satu:

```
// Sensor 1 setup
digitalWrite(XSHUT_1, HIGH);  // Aktifkan sensor 1
digitalWrite(XSHUT_2, LOW);   // Matikan sensor 2
sensor1.init();               // Init dengan address 0x29
sensor1.setAddress(0x30);     // Ubah ke 0x30

// Sensor 2 setup
digitalWrite(XSHUT_2, HIGH);  // Aktifkan sensor 2
sensor2.init();               // Init dengan address 0x29
sensor2.setAddress(0x31);     // Ubah ke 0x31
```

📚 F. KONSEP IOT YANG DIPELAJARI

1. Time of Flight (ToF) Technology

ToF adalah teknologi pengukuran jarak menggunakan waktu tempuh cahaya:

```
Prinsip Kerja:
1. Laser emitter memancarkan pulsa cahaya infrared
2. Cahaya memantul dari objek (tangan)
3. Sensor menerima cahaya pantul
4. Hitung waktu tempuh (time of flight)
5. Distance = (Speed of Light × Time) / 2

Formula:
d = (c × t) / 2
d = jarak (mm)
c = kecepatan cahaya (299,792 km/s)
t = waktu tempuh (nanoseconds)

Keunggulan vs Ultrasonic:
- Akurasi lebih tinggi (±3mm vs ±1cm)
- Tidak terpengaruh suhu/kelembaban
- Response time lebih cepat (<50ms vs ~100ms)
- Field of View lebih terfokus (25° vs 60°)
```

2. I2C Multiple Devices

Mengelola 2 sensor ToF + OLED pada satu I2C bus:

```
I2C Bus Topology:
       ESP32
      /  |  \
     /   |   \
  OLED  S1   S2
 (0x3C)(0x30)(0x31)

Problem: Kedua sensor punya address sama (0x29)
Solution: XSHUT pin untuk sequential initialization

Steps:
1. Matikan semua sensor (XSHUT = LOW)
2. Nyalakan sensor 1 (XSHUT_1 = HIGH)
3. Init sensor 1 dengan address 0x29
4. Ubah address sensor 1 ke 0x30
5. Nyalakan sensor 2 (XSHUT_2 = HIGH)
6. Init sensor 2 dengan address 0x29
7. Ubah address sensor 2 ke 0x31
8. Sekarang keduanya bisa berjalan bersamaan
```

3. Gesture Recognition (Swipe Detection)

Algoritma mendeteksi sapuan tangan:

```
Swipe Detection Logic:
┌─────────────────────────────────────┐
│ 1. Tangan masuk zona (< 50mm)       │
│    → swipe_start_time = now         │
│    → in_swipe_zone = true           │
│                                     │
│ 2. Track minimum distance           │
│    → min_distance = min(readings)   │
│                                     │
│ 3. Tangan keluar zona (> 80mm)      │
│    → swipe_duration = now - start   │
│                                     │
│ 4. Validasi swipe:                  │
│    ✓ Duration: 100-1000ms           │
│    ✓ Min distance: < 25mm           │
│    ✓ Clear exit (> swipe + 30mm)    │
│                                     │
│ 5. Action: Toggle mode / AP mode    │
└─────────────────────────────────────┘

Multiple Swipe untuk AP Mode:
- Swipe 1x: Toggle AUTO ↔ MANUAL
- Swipe 5x dalam 3 detik: Activate AP Mode
```

4. Watchdog Timer

Safety mechanism untuk mencegah sistem hang:

```
Watchdog Concept:
┌───────────────────────────────────────┐
│ System "feeds" watchdog setiap loop   │
│ Watchdog timer = 8 detik              │
│                                       │
│ Normal operation:                     │
│ Loop → Feed → Timer reset → Loop      │
│   ↑__________________________|        │
│                                       │
│ System hang:                          │
│ Loop → HANG → Timer expired → RESET   │
└───────────────────────────────────────┘

Implementation:
void feedWatchdog() {
  timing.last_watchdog_feed = millis();
  watchdog_triggered = false;
}

// Di setiap loop
void loop() {
  feedWatchdog();  // Reset watchdog
  // ... kode lainnya
}

// ISR check timeout
void watchdogISR() {
  if (millis() - last_feed > TIMEOUT) {
    ESP.restart();  // Force restart
  }
}
```

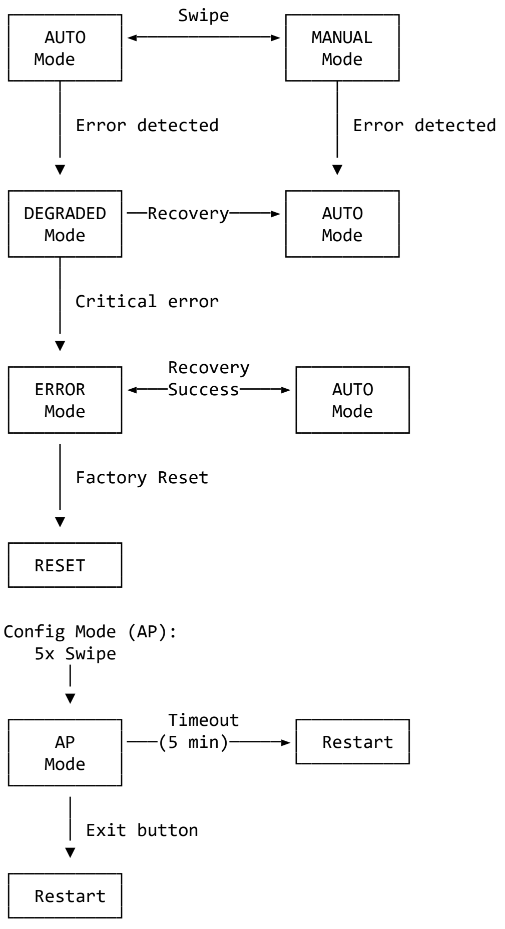
5. EEPROM Struktur Data

Persistent storage dengan integrity check:

```
EEPROM Layout (1024 bytes):
┌────────┬───────────────────┬──────┐
│ Offset │ Field             │ Size │
├────────┼───────────────────┼──────┤
│ 0      │ Signature (0xAA55) │ 2    │
│ 2      │ Version           │ 2    │
│ 4      │ Total Activations │ 4    │
│ 8      │ Water Duration    │ 4    │
│ 12     │ Hand Min Distance │ 2    │
│ 14     │ Hand Max Distance │ 2    │
│ 16     │ Swipe Distance    │ 2    │
│ 18     │ Water Off Delay   │ 4    │
│ 22     │ Max Water Time    │ 4    │
│ 26     │ Admin Password    │ 32   │
│ 58     │ Operation Hours   │ 4    │
│ 62     │ Error Count       │ 2    │
│ 64     │ CRC Checksum      │ 2    │
└────────┴───────────────────┴──────┘

CRC (Cyclic Redundancy Check):
- Deteksi data corruption
- Polynomial: 0x1021 (CRC-16-CCITT)
- Calculate saat save, verify saat load
```

6. State Machine Design

Sistem mode menggunakan state machine:



⚙️ G. CARA KERJA SISTEM

1. Pembacaan Sensor (Continuous Mode)

```
// Sensor dijalankan dalam continuous mode
sensor1.startContinuous(50);  // Read setiap 50ms
sensor2.startContinuous(50);

// Pembacaan non-blocking
void readSensors() {
  // Throttle reading (50ms interval)
  if (millis() - timing.last_sensor_read < SENSOR_READ_INTERVAL) {
    return;  // Skip jika belum waktunya
  }

  timing.last_sensor_read = millis();

  // Baca jarak dari kedua sensor
  sensors.distance1 = sensor1.readRangeContinuousMillimeters();
  sensors.distance2 = sensor2.readRangeContinuousMillimeters();

  // Error detection
  bool sensor1_error = sensor1.timeoutOccurred() ||
                       sensors.distance1 > 8000;
  bool sensor2_error = sensor2.timeoutOccurred() ||
                       sensors.distance2 > 8000;

  // Update health status
  sensors.sensor1_healthy = !sensor1_error;
  sensors.sensor2_healthy = !sensor2_error;

  // Track consecutive errors
  if (sensor1_error || sensor2_error) {
    sensors.consecutive_errors++;
  } else {
    sensors.consecutive_errors = 0;  // Reset on success
  }

  // Enter degraded mode jika terlalu banyak error
  if (sensors.consecutive_errors >= 10) {
    system_state.mode = MODE_DEGRADED;
  }
}
```

Kenapa Continuous Mode?

- Latency rendah (<50ms)
- CPU efficient (no blocking wait)
- Smooth gesture detection
- Real-time response

2. Kontrol Relay (Aktif LOW)

```
void turnWaterOn() {
  if (system_state.water_state != WATER_ON) {
    digitalWrite(RELAY_PIN, LOW);  // Relay aktif LOW
    system_state.water_state = WATER_ON;
    timing.water_start_time = millis();

    // Statistics
    config.total_activations++;
    markConfigDirty();  // Flag untuk save

    logEvent("WATER_TURNED_ON");
  }
}

void turnWaterOff() {
  if (system_state.water_state != WATER_OFF) {
    digitalWrite(RELAY_PIN, HIGH);  // Relay non-aktif

    // Hitung durasi
    unsigned long duration = millis() - timing.water_start_time;
    config.total_water_duration += duration / 1000;  // dalam detik
    markConfigDirty();

    system_state.water_state = WATER_OFF;
    logEvent("WATER_TURNED_OFF");
  }
}
```

Relay Wiring:

```
Relay Module:
COM  ──────► 12V Power Supply (+)
NO   ──────► Solenoid Valve (+)
Valve (-) ──► Ground

GPIO 25:
  LOW  → Relay ON  → Valve OPEN  → Water Flow
  HIGH → Relay OFF → Valve CLOSED → No Water
```

3. Safety Features

A. Maximum Water Time Protection

```
const unsigned long MAX_WATER_TIME = 600000;  // 10 menit

void handleWaterControl() {
  if (system_state.water_state == WATER_ON) {
    unsigned long duration = millis() - timing.water_start_time;

    // Warning 1 menit sebelum timeout
    if (duration >= (MAX_WATER_TIME - 60000) &&
        duration < (MAX_WATER_TIME - 59000)) {
      displayWarning("SISA 1 MENIT!");
      logEvent("TIMEOUT_WARNING");
    }

    // Force shutoff pada max time
    if (duration >= MAX_WATER_TIME) {
      turnWaterOff();
      system_state.mode = MODE_AUTO;
      stats.max_water_time_hits++;

      logEvent("SAFETY_TIMEOUT_TRIGGERED");
      displayWarning("TIMEOUT!");
      delay(2000);
    }
  }
}
```

B. Sensor Error Handling

```
// Emergency shutoff jika semua sensor mati
if (!sensors.sensor1_healthy && !sensors.sensor2_healthy) {
  if (system_state.water_state != WATER_OFF) {
    turnWaterOff();
    system_state.mode = MODE_ERROR;
    logError("EMERGENCY_SHUTOFF_BOTH_SENSORS_FAILED");
    displayError("SENSOR ERROR");
  }
  return;
}
```

C. Watchdog Protection

```
volatile bool watchdog_triggered = false;

void IRAM_ATTR watchdogISR() {
  watchdog_triggered = true;
}

void setupWatchdog() {
  watchdogTicker.attach_ms(WATCHDOG_INTERVAL, []() {
    if (millis() - timing.last_watchdog_feed > WATCHDOG_INTERVAL * 2) {
      Serial.println("Watchdog timeout! Restarting...");
      ESP.restart();
    }
  });
}
```

D. Water Off Delay (Hysteresis)

```
// Mencegah air ON-OFF flickering
const unsigned long WATER_OFF_DELAY = 1500;  // 1.5 detik

if (tangan_tidak_terdeteksi) {
  if (system_state.water_state != WATER_TURNING_OFF) {
    system_state.water_state = WATER_TURNING_OFF;
    timing.hand_detection_time = millis();
  }

  // Off setelah delay
  if (millis() - timing.hand_detection_time >= WATER_OFF_DELAY) {
    turnWaterOff();
  }
}
```

🎮 H. MODE OPERASI

MODE 1: AUTO (Default) 🤖

Cara Kerja:

- Sensor atas (Sensor 2) mendeteksi kehadiran tangan
- Jika tangan dalam range (80-300mm) → Air ON
- Jika tangan keluar range → Tunggu 1.5 detik → Air OFF
- Hysteresis mencegah flickering

Flow Logic:

```
┌──────────────────┐
│ Baca Sensor Atas │
└────────┬─────────┘
         │
         ▼
    ┌──────────┐  NO
    │  Tangan  ├──────────┐
    │Terdeteksi│          │
    └────┬─────┘          │
      YES│                │
         ▼                │
    ┌──────────┐          │
    │  Air ON  │          │
    └────┬─────┘          │
         │                │
         ▼                │
    ┌──────────┐          ▼
    │ Monitor  │     ┌──────────┐
    │  Tangan  │     │ Air OFF  │
    └────┬─────┘     │  (delay) │
         │           └──────────┘
         ▼
  Tangan hilang?
         │ YES
         ▼
    ┌──────────┐
    │Start OFF │
    │  Timer   │
    └────┬─────┘
         │
         ▼ 1.5s elapsed
    ┌──────────┐
    │ Air OFF  │
    └──────────┘
```

Code Implementation:

```
void handleAutoMode() {
  if (sensors.sensor2_healthy) {
    // Deteksi tangan dalam range
    if (sensors.distance2 >= config.hand_min_distance &&
        sensors.distance2 <= config.hand_max_distance) {

      if (system_state.water_state != WATER_ON) {
        turnWaterOn();
      }
    }
    // Tangan keluar range
    else if (system_state.water_state == WATER_ON) {
      // Mulai countdown untuk OFF
      if (system_state.water_state != WATER_TURNING_OFF) {
        system_state.water_state = WATER_TURNING_OFF;
        timing.hand_detection_time = millis();
      }

      // OFF setelah delay
      if (millis() - timing.hand_detection_time >= config.water_off_delay) {
        turnWaterOff();
      }
    }
  }
}
```

Parameter Default:

- **Hand Min Distance**: 80mm (jarak minimum deteksi)
- **Hand Max Distance**: 300mm (jarak maksimum deteksi)
- **Water Off Delay**: 1500ms (delay sebelum OFF)

Contoh Skenario:

```
Waktu | Jarak Sensor | Status Air | Keterangan
------|--------------|------------|------------------
00:00 | 1500mm       | OFF        | Standby
00:01 | 150mm        | ON         | Tangan terdeteksi
00:05 | 120mm        | ON         | Tangan masih ada
00:10 | 450mm        | ON (delay) | Tangan hilang, mulai countdown
00:11 | 600mm        | ON (delay) | Countdown 0.5s
00:12 | 800mm        | OFF        | Delay selesai, air OFF
```

MODE 2: MANUAL 🎯

Cara Kerja:

- Air tetap mengalir tanpa deteksi otomatis
- Hanya bisa dimatikan dengan swipe lagi atau timeout
- Cocok untuk keperluan khusus (isi ember, cuci kaki)

Aktivasi Manual Mode:

1. Swipe tangan di sensor bawah (< 50mm)
2. Sistem toggle dari AUTO ke MANUAL
3. Air langsung ON dan tetap ON
4. Swipe lagi untuk kembali ke AUTO

Code Logic:

```
void handleManualMode() {
  // Air tetap ON, tidak ada automatic control
  // Hanya bisa OFF via:
  // 1. Swipe lagi (toggle ke AUTO)
  // 2. Safety timeout (10 menit)
}

void handleValidSwipe() {
  if (millis() - timing.last_swipe_time > DEBOUNCE_TIME) {
    timing.last_swipe_time = millis();

    if (system_state.mode == MODE_AUTO) {
      system_state.mode = MODE_MANUAL;
      turnWaterOn();
      logEvent("MANUAL_MODE_ACTIVATED");
    }
    else if (system_state.mode == MODE_MANUAL) {
      system_state.mode = MODE_AUTO;
      turnWaterOff();
      logEvent("AUTO_MODE_ACTIVATED");
    }
  }
}
```

Indikator LED:

- AUTO mode: LED OFF (kecuali air ON)
- MANUAL mode: LED berkedip 1Hz (500ms ON, 500ms OFF)

MODE 3: DEGRADED ⚠️

Cara Kerja:

- Mode otomatis dengan sensor terbatas
- Aktif ketika salah satu sensor error
- Sistem tetap berjalan dengan sensor yang sehat

Skenario:

A. Sensor Bawah Error, Atas OK:

```
Status: Degraded Mode
- Swipe detection: DISABLED
- Auto detection: ENABLED (sensor atas)
- Manual mode: DISABLED (tidak bisa toggle)
```

B. Sensor Atas Error, Bawah OK:

```
Status: Degraded Mode
- Swipe detection: ENABLED (sensor bawah)
- Auto detection: DISABLED
- Manual mode: ENABLED (toggle via swipe)
```

Recovery Attempt:

```
void handleDegradedMode() {
  // Coba gunakan sensor yang sehat
  if (sensors.sensor2_healthy && !sensors.sensor1_healthy) {
    handleAutoMode();  // Pakai sensor atas untuk auto
  }
  else if (sensors.sensor1_healthy && !sensors.sensor2_healthy) {
    // Manual mode only via swipe
  }

  // Coba recovery setiap 10 detik
  if (millis() - timing.last_error_time > ERROR_RECOVERY_INTERVAL) {
    attemptSensorRecovery();
  }
}

bool attemptSensorRecovery() {
  stats.recovery_attempts++;

  Serial.println("Attempting sensor recovery...");

  // Reset hardware
  digitalWrite(XSHUT_1, LOW);
  digitalWrite(XSHUT_2, LOW);
  delay(100);

  // Re-initialize sensors
  if (initializeSensors()) {
    sensors.consecutive_errors = 0;
    sensors.sensor1_healthy = true;
    sensors.sensor2_healthy = true;
    stats.successful_recoveries++;

    logEvent("SENSOR_RECOVERY_SUCCESS");
    return true;
  }

  logError("SENSOR_RECOVERY_FAILED");
  return false;
}
```

MODE 4: ERROR ❌

Trigger Conditions:

- Kedua sensor error bersamaan
- Consecutive errors ≥ 10
- Critical system failure

Behavior:

```
1. Air FORCE OFF (emergency shutoff)
2. LED berkedip cepat (500ms)
3. Display: "SYSTEM ERROR"
4. Automatic recovery attempt setiap 30 detik
5. Watchdog tetap aktif (restart jika hang)
```

Code:

```
void handleErrorState() {
  digitalWrite(RELAY_PIN, HIGH);  // Force OFF

  // LED blink fast
  static unsigned long last_blink = 0;
  if (millis() - last_blink > 500) {
    digitalWrite(LED_PIN, !digitalRead(LED_PIN));
    last_blink = millis();
  }

  // Auto recovery setiap 30 detik
  if (millis() - timing.last_error_time > 30000) {
    if (attemptSensorRecovery()) {
      system_state.mode = MODE_AUTO;
      logEvent("ERROR_RECOVERY_SUCCESS");
    }
    timing.last_error_time = millis();
  }

  updateErrorDisplay();
}
```

MODE 5: AP (Config Mode) 🌐

Aktivasi:

Swipe 5x dalam 3 detik di sensor bawah

Features:

```
✓ WiFi Access Point aktif
✓ Web dashboard untuk konfigurasi
✓ QR code di OLED untuk quick access
✓ Auto-exit setelah 5 menit timeout
✓ Manual exit via web button
```

AP Configuration:

```
SSID: "WudhuConfig"
Password: "Wudhu_" + device_id (dynamic)
IP Address: 192.168.4.1
Timeout: 5 menit
```

Multiple Swipe Detection:

```
void handleValidSwipe() {
  if (swipe.swipe_count == 0) {
    swipe.first_swipe_time = millis();
    swipe.swipe_count = 1;
  }
  else if (millis() - swipe.first_swipe_time < 3000) {
    swipe.swipe_count++;

    // 5x swipe = AP mode
    if (swipe.swipe_count >= 5) {
      activateAPMode();
      return;
    }
  }
  else {
    // Reset jika > 3 detik
    swipe.swipe_count = 1;
    swipe.first_swipe_time = millis();
  }

  // Swipe biasa (toggle mode)
  // ...
}
```

🚀 I. PANDUAN INSTALASI

Langkah 1: Persiapan Arduino IDE

1. Download & install Arduino IDE dari arduino.cc
2. Tambahkan ESP32 Board Manager:
   - File → Preferences
   - Additional Board Manager URLs:

   ```
   https://dl.espressif.com/dl/package_esp32_index.json
   ```

   - Klik OK
3. Install ESP32 Board:
   - Tools → Board → Board Manager
   - Cari "ESP32"
   - Install "esp32 by Espressif Systems"

Langkah 2: Install Library

Via Library Manager (Sketch → Include Library → Manage Libraries):

Library	Author	Version
VL53L0X	Pololu	Latest
Adafruit GFX Library	Adafruit	Latest
Adafruit SSD1306	Adafruit	Latest

Library	Author	Version
qrcode	Richard Moore	Latest
ArduinoJson	Benoit Blanchon	v6.x

Library built-in ESP32:

- Wire (I2C)
- WiFi
- WebServer
- EEPROM
- Ticker

Langkah 3: Konfigurasi Board

```
Tools → Board: "ESP32 Dev Module"
Tools → Upload Speed: "115200"
Tools → Flash Frequency: "80MHz"
Tools → Flash Mode: "QIO"
Tools → Flash Size: "4MB"
Tools → Partition Scheme: "Default 4MB with spiffs"
Tools → Core Debug Level: "None" (atau "Info" untuk debug)
Tools → Port: [Pilih COM port ESP32]
```

Langkah 4: Upload Code

1. Buka file `kran_wudhu_otomatis.ino`
2. Verify (✓) untuk compile
3. Jika berhasil, Upload (→)
4. Tunggu hingga "Done uploading"

Langkah 5: Monitoring

1. Tools → Serial Monitor
2. Set baud rate: **115200**
3. Output yang diharapkan:

```
=== KRAN WUDHU PRODUCTION SYSTEM ESP32 ===
Model: KW-2024-PRO-ESP32
Firmware: 3.1.0-ESP32
Serial: KW12345678

Sensor init attempt 1/3
Sensor 1 init success
Sensor 2 init success
Sensors initialized successfully

OLED initialized
Device ID: 12345678
AP Password: Wudhu_123456
Admin Password: Admin_12345678

Free RAM: 290000 bytes
=== SYSTEM INITIALIZATION COMPLETE ===
```

📖 J. PANDUAN PENGGUNAAN

A. Penggunaan Normal (AUTO Mode)

1. **Standby State:**
   - OLED menampilkan "WUDHU OTOMATIS"
   - LED OFF
   - Sistem siap mendeteksi
2. **Wudhu:**
   - Dekatkan tangan ke sensor atas (jarak 10-20cm)
   - Air otomatis mengalir
   - OLED menampilkan durasi dan progress bar
   - Tarik tangan, tunggu 1.5 detik, air otomatis berhenti
3. **Tips:**
   - Jarak ideal: 10-15cm dari sensor
   - Jangan terlalu dekat (<5cm) untuk menghindari percikan
   - Jika air tidak berhenti, gerakkan tangan menjauh (>50cm)

B. Manual Mode (via Swipe)

1. **Aktivasi:**
   - Sapukan tangan cepat di depan sensor bawah
   - Gerakan: masuk (<5cm) → keluar (>8cm)
   - Durasi: 100-1000ms (tidak terlalu cepat/lambat)

2. **Indikator Manual Mode:**
    - OLED: “MANUAL” di pojok
    - LED berkedip 1Hz
    - Air tetap ON
3. **Kembali ke AUTO:**
    - Swipe lagi di sensor bawah
    - Atau tunggu timeout 10 menit

C. Configuration Mode (AP Mode)

1. Aktivasi AP Mode:

```
Cara 1: Swipe 5x berturut-turut (dalam 3 detik)
├── Swipe 1
├── Swipe 2
├── Swipe 3
├── Swipe 4
└── Swipe 5 → AP Mode aktif!

Cara 2: Factory reset button (hold 10 detik)
```

2. Koneksi ke Dashboard:

Via QR Code:

1. OLED menampilkan QR code
2. Scan dengan smartphone
3. Connect ke WiFi “WudhuConfig”
4. Browser otomatis buka dashboard

Via Manual:

1. Buka WiFi settings di smartphone/laptop
2. Connect ke: **WudhuConfig**
3. Password: **Wudhu_xxxxxx** (lihat di OLED atau serial monitor)
4. Buka browser: `http://192.168.4.1`

3. Login Dashboard:

```
┌─────────────────────────────────┐
│     KRAN WUDHU PRO              │
│   Production System ESP32       │
│                                 │
│  Username: [admin          ]    │
│  Password: [************** ]    │
│                                 │
│        [     LOGIN     ]        │
│                                 │
│  Device ID: 12345678            │
│  IP: 192.168.4.1                │
└─────────────────────────────────┘

Credentials:
- Username: admin
- Password: Admin_[device_id] (lihat serial monitor)
```

4. Dashboard Features:

A. System Status:

```
┌─────────────────────────────────┐
│ SENSOR STATUS                   │
├─────────────────────────────────┤
│                                 │
│ ┌──────────┐    ┌──────────┐    │
│ │ Bawah    │    │  Atas    │    │
│ │  85mm    │    │ 150mm    │    │
│ │ HEALTHY  │    │ HEALTHY  │    │
│ └──────────┘    └──────────┘    │
│                                 │
└─────────────────────────────────┘

┌─────────────────────────────────┐
│ STATISTICS                      │
├─────────────────────────────────┤
│  Total Aktivasi:    1,234       │
│  Total Durasi:      567 menit   │
│  Jam Operasi:       48 jam      │
│  Error Count:       3           │
└─────────────────────────────────┘
```

B. Configuration:

```
┌─────────────────────────────────┐
│ PARAMETER SETTINGS              │
├─────────────────────────────────┤
│                                 │
│ Sensor Atas Min:   [80   ] mm     │
│ Sensor Atas Max:   [300  ] mm     │
│ Swipe Distance:    [50   ] mm     │
│ Water Off Delay:   [1500 ] ms     │
│ Max Water Time:    [10   ] min    │
│                                   │
│      [💾 SAVE CHANGES]             │
└───────────────────────────────────┘
```

C. Maintenance:

```
┌───────────────────────────────────┐
│ MAINTENANCE OPTIONS               │
├───────────────────────────────────┤
│ [🔄 Reset Statistics]              │
│ [🏭 Factory Reset]                 │
│ [🚪 Exit AP Mode]                  │
└───────────────────────────────────┘
```

5. Setting Parameters:

Sensor Atas Min (Hand Min Distance):

- Range: 50-200mm
- Default: 80mm
- Fungsi: Jarak minimum untuk deteksi tangan
- Tips: Naikkan jika terlalu sensitif

Sensor Atas Max (Hand Max Distance):

- Range: 100-1000mm
- Default: 300mm
- Fungsi: Jarak maksimum untuk deteksi tangan
- Tips: Turunkan untuk save range lebih ketat

Swipe Distance:

- Range: 20-100mm
- Default: 50mm
- Fungsi: Jarak threshold untuk swipe gesture
- Tips: Turunkan jika swipe sulit terdeteksi

Water Off Delay:

- Range: 500-10000ms
- Default: 1500ms
- Fungsi: Delay sebelum air OFF setelah tangan hilang
- Tips: Naikkan untuk mencegah air cepat mati

Max Water Time:

- Range: 1-30 menit
- Default: 10 menit
- Fungsi: Safety timeout untuk prevent flooding
- Tips: Sesuaikan dengan kebutuhan (misal: 5 menit untuk wudhu normal)

6. Exit AP Mode:

Auto-exit:

- Timeout 5 menit tanpa aktivitas
- Sistem auto restart dan kembali normal

Manual exit:

- Klik tombol "Exit AP Mode" di dashboard
- Atau swipe 5x lagi di sensor bawah

D. Monitoring via Serial Monitor

Serial monitor memberikan log real-time:

```
[EVENT] WATER_TURNED_ON | Uptime: 125 s
[EVENT] Sensor - S1:150mm S2:120mm
[EVENT] Mode: AUTO, Water: ON, Duration: 5s
[EVENT] WATER_TURNED_OFF | Uptime: 130 s
[EVENT] MANUAL_MODE_ACTIVATED | Uptime: 145 s
[EVENT] AP_MODE_ACTIVATED | Uptime: 200 s
[ERROR] SENSOR_TIMEOUT | S1:1(1500mm) S2:0(8192mm) | Errors:1
[EVENT] SENSOR_RECOVERY_SUCCESS | Uptime: 230 s
```

Log Format:

```
[LEVEL] Message | Context
```

Log Levels:

- `[EVENT]`: Normal operations

- `[ERROR]`: Errors dan exceptions
- `[WARNING]`: Warnings (tidak di implement code ini)

🔧 K. TROUBLESHOOTING

Problem 1: Sensor Tidak Mendeteksi

Gejala:

- Air tidak mengalir saat tangan didekatkan
- OLED menampilkan jarak > 2000mm
- Serial: "SENSOR_TIMEOUT"

Solusi:

1. **Cek koneksi I2C:**

```
Sensor 1:
- SDA → GPIO 21
- SCL → GPIO 22
- VCC → 3.3V (PENTING: Bukan 5V!)
- GND → GND
- XSHUT → GPIO 32

Sensor 2:
- SDA → GPIO 21 (sharing dengan Sensor 1)
- SCL → GPIO 22 (sharing dengan Sensor 1)
- VCC → 3.3V
- GND → GND
- XSHUT → GPIO 33
```

2. **Cek power supply:**
   - VL53L0X butuh 3.3V, TIDAK 5V!
   - Arus minimum: 20mA per sensor
   - Gunakan pin 3.3V ESP32, bukan 5V
3. **Test dengan I2C Scanner:**

```
#include <Wire.h>

void setup() {
  Serial.begin(115200);
  Wire.begin(21, 22);

  Serial.println("I2C Scanner");

  for (byte addr = 1; addr < 127; addr++) {
    Wire.beginTransmission(addr);
    if (Wire.endTransmission() == 0) {
      Serial.printf("Device found at 0x%02X\n", addr);
    }
  }
}

void loop() {}
```

Expected output:

```
Device found at 0x30  (Sensor 1)
Device found at 0x31  (Sensor 2)
Device found at 0x3C  (OLED)
```

4. **Cek XSHUT pins:**

```
// Di serial monitor, cek log:
"Sensor 1 XSHUT: HIGH"
"Sensor 2 XSHUT: HIGH"
```

5. **Replace sensor jika rusak:**
   - Test sensor satu per satu
   - Ganti jika tetap error setelah semua cek

Problem 2: Air Tidak Mau Berhenti

Gejala:

- Air terus mengalir meski tangan sudah ditarik
- Atau air tidak mau OFF di manual mode

Solusi:

1. **Force OFF via Serial:**

```
// Tambahkan di loop() untuk testing:
if (Serial.available()) {
  char cmd = Serial.read();
  if (cmd == 'f') {  // 'f' = force off
    turnWaterOff();
    Serial.println("FORCE OFF");
  }
}
```

2. **Cek relay:**

```
Test relay manual:
- Cabut jumper dari GPIO 25
- Hubungkan GPIO 25 ke GND → Relay ON (air ON)
- Hubungkan GPIO 25 ke 3.3V → Relay OFF (air OFF)

Jika relay tidak respond:
- Ganti relay module
- Atau cek power relay (butuh 5V)
```

3. **Cek sensor atas (jarak detection):**

```
Jika sensor membaca jarak terus < 300mm:
- Ada objek di depan sensor (lap, dinding, dll)
- Arahkan sensor ke arah yang bebas obstacle
- Atau turunkan hand_max_distance di config
```

4. **Emergency restart:**

```
- Tekan tombol RST/EN di ESP32
- Atau cabut-colok power
- Sistem akan restart dan air force OFF
```

Problem 3: OLED Tidak Menyala

Gejala:

- Display hitam / tidak ada tampilan
- Serial: “OLED init failed” (jika ada log)

Solusi:

1. **Cek koneksi I2C OLED:**

```
OLED:
- SDA → GPIO 21 (sharing dengan sensors)
- SCL → GPIO 22 (sharing dengan sensors)
- VCC → 3.3V atau 5V (tergantung module)
- GND → GND
```

2. **Cek alamat I2C:**

```
// Alamat OLED biasanya 0x3C atau 0x3D
// Coba ubah di code:
display.begin(SSD1306_SWITCHCAPVCC, 0x3D);  // Ganti ke 0x3D
```

3. **Test OLED terpisah:**

```
// Upload code test OLED saja
#include <Adafruit_SSD1306.h>

Adafruit_SSD1306 display(128, 64, &Wire, -1);

void setup() {
  Wire.begin(21, 22);

  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("OLED gagal!");
  } else {
    display.clearDisplay();
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(0, 0);
    display.println("OLED OK!");
    display.display();
  }
}

void loop() {}
```

4. **Ganti OLED jika rusak**

Problem 4: WiFi AP Tidak Muncul

Gejala:

- Swipe 5x tapi SSID "WudhuConfig" tidak muncul
- Atau SSID muncul tapi tidak bisa connect

Solusi:

1. **Cek aktivasi AP mode:**

```
Serial monitor harus menampilkan:
"AP Mode activated"
"SSID: WudhuConfig"
"Password: Wudhu_xxxxxx"
"IP: 192.168.4.1"
```

2. **Cek swipe detection:**

```
Swipe harus:
- Jarak < 50mm (sensor bawah)
- Durasi 100-1000ms
- 5x dalam 3 detik

Tips:
- Sapukan tangan dengan jarak 3-5cm dari sensor
- Gerakan seperti "usap" cepat
- Jangan terlalu pelan (>1 detik)
```

3. **Force AP mode via serial:**

```
// Tambahkan di loop():
if (Serial.available()) {
  char cmd = Serial.read();
  if (cmd == 'a') {  // 'a' = activate AP
    activateAPMode();
  }
}
```

4. **Restart ESP32:**

```
- Tekan RST/EN
- Coba swipe 5x lagi
```

5. **Cek WiFi channel:**

```
// Ubah channel jika bentrok
WiFi.softAP(ap_ssid, ap_password.c_str(), 6);  // Channel 6
```

Problem 5: Sistem Sering Restart

Gejala:

- ESP32 restart terus-menerus
- Serial: "Brownout detector" atau "Watchdog timeout"

Solusi:

1. **Upgrade power supply:**

```
Requirement:
- ESP32: ~500mA (saat WiFi aktif)
- Sensors: ~40mA (2x VL53L0X)
- OLED: ~20mA
- Relay: ~70mA (saat aktif)
- Total: ~630mA

Recommendation:
- 5V/2A minimum
- 5V/3A ideal
- Jangan pakai USB laptop (arus terbatas!)
```

2. **Pisahkan power solenoid:**

```
ESP32 + Electronics: 5V/2A
Solenoid Valve: 12V/1A (separate supply)

JANGAN power solenoid dari ESP32!
```

3. **Tambah kapasitor:**

```
- 1000µF di VIN ESP32
- 100µF di VCC relay module
- Untuk stabilize voltage saat relay switch
```

4. **Cek watchdog:**

```
Jika restart karena watchdog:
- Ada infinite loop di code
- I2C timeout terlalu lama
- Check code untuk blocking operations
```

5. **Disable brownout (temporary):**

```
// Sudah ada di code:
WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
```

Problem 6: Swipe Tidak Terdeteksi

Gejala:

- Sapukan tangan berkali-kali tapi tidak ada respon
- Atau swipe kadang detect, kadang tidak

Solusi:

1. **Cek sensor bawah (Sensor 1):**

```
Serial monitor harus menampilkan:
"S1: XXXmm"  (jarak real-time)

Saat tangan dekat (< 50mm):
"Swipe zone entered"
```

2. **Adjust swipe distance:**

```
Via AP Mode Dashboard:
- Turunkan "Swipe Distance" dari 50mm ke 30mm
- Save dan test lagi
```

3. **Cek timing swipe:**

```
Swipe valid:
✓ Duration: 100-1000ms
✓ Min distance: < 25mm
✓ Clear exit: > 80mm

Swipe invalid:
✗ Terlalu cepat (<100ms)
✗ Terlalu lambat (>1s)
✗ Tidak cukup dekat (>50mm)
✗ Tidak clear exit (stuck di zona)
```

4. **Test swipe di serial:**

```
// Tambahkan debug log:
void detectSwipeGesture() {
  if (sensors.distance1 < config.swipe_distance && !swipe.in_swipe_zone) {
    Serial.println(">>> SWIPE START");
    // ...
  }

  if (sensors.distance1 >= config.swipe_distance + 30 && swipe.in_swipe_zone) {
    Serial.printf(">>> SWIPE END: duration=%lu, min=%u\n",
                  swipe_duration, swipe.min_distance_in_swipe);
    // ...
  }
}
```

Problem 7: Web Dashboard Tidak Bisa Dibuka

Gejala:

- Connect ke WiFi berhasil
- Tapi browser tidak bisa buka `192.168.4.1`

Solusi:

1. **Cek IP smartphone:**

```
WiFi Settings → WudhuConfig → Details
IP: 192.168.4.X  (harus 4.X, bukan yang lain)
Gateway: 192.168.4.1
```

2. **Gunakan IP langsung:**

```
Jangan: http://wudhuconfig
Gunakan: http://192.168.4.1
```

3. **Clear browser cache:**

```
Chrome: Settings → Privacy → Clear cache
Safari: Settings → Clear history
Atau gunakan incognito/private mode
```

4. **Test dengan curl:**

```
# Di laptop yang terconnect
curl http://192.168.4.1

# Harus return HTML content
```

5. **Restart web server:**

```
// Exit AP mode (via swipe 5x atau timeout)
// Re-enter AP mode
// Web server akan restart
```

Problem 8: Factory Reset Tidak Berfungsi

Gejala:

- Tekan button factory reset tidak ada respon
- Atau button ditekan 10 detik tapi tidak reset

Solusi:

1. **Cek koneksi button:**

```
Factory Reset Button:
- Pin 1 → GPIO 34
- Pin 2 → GND
- Internal pull-up: ENABLED

Test:
Serial.println(digitalRead(34));
// Normal: 1 (HIGH)
// Pressed: 0 (LOW)
```

2. **Cek di serial monitor:**

```
Saat button ditekan 10 detik:
"PERFORMING FACTORY RESET"
"Factory reset complete. Restarting..."
```

3. **Force factory reset via AP mode:**

```
1. Masuk AP mode (swipe 5x)
2. Login dashboard
3. Klik "Factory Reset"
4. Konfirmasi
5. Sistem akan restart dengan config default
```

4. **Manual factory reset via serial:**

```
// Tambahkan di loop():
if (Serial.available()) {
  char cmd = Serial.read();
  if (cmd == 'r') {  // 'r' = reset
    performFactoryReset();
  }
}
```

Problem 9: Error “I2C Communication Failed”

Gejala:

- Serial: “WARNING: I2C communication failed”
- Sensors atau OLED kadang tidak respond

Solusi:

1. **Cek pull-up resistor:**

```
I2C butuh pull-up 4.7kΩ di SDA & SCL
- Biasanya sudah ada di module
- Jika tidak, tambahkan eksternal:
  * 4.7kΩ dari SDA ke 3.3V
  * 4.7kΩ dari SCL ke 3.3V
```

2. **Kurangi I2C speed:**

```
// Di initializeHardware(), ubah:
Wire.setClock(400000);  // 400kHz
// Menjadi:
Wire.setClock(100000);  // 100kHz (lebih stabil)
```

3. **Perpendek kabel I2C:**

```
Panjang maksimal kabel I2C:
- 100kHz: ~1 meter
- 400kHz: ~20cm

Solusi:
- Gunakan kabel sependk mungkin
- Hindari kabel jumper panjang
- PCB custom lebih baik
```

4. **Cek ground connection:**

```
Semua GND harus terhubung:
ESP32 GND ─┬─ Sensor 1 GND
           ├─ Sensor 2 GND
           ├─ OLED GND
           ├─ Relay GND
           └─ Power Supply GND
```

5. **Reset I2C bus:**

```
// Tambahkan fungsi recovery:
void resetI2C() {
  Wire.end();
  delay(100);
  Wire.begin(SDA_PIN, SCL_PIN);
  Wire.setClock(400000);
}
```

Problem 10: Statistics Tidak Tersimpan

Gejala:

- Setelah restart, total aktivasi kembali ke 0
- Setting threshold berubah ke default

Solusi:

1. **Cek EEPROM signature:**

```
// Di serial monitor saat boot:
"Configuration loaded"  // OK
"Invalid EEPROM signature"  // ERROR
```

2. **Force save configuration:**

```
// Tambahkan di loop():
if (Serial.available()) {
  char cmd = Serial.read();
  if (cmd == 's') {  // 's' = save
    saveConfiguration();
    Serial.println("Config saved!");
  }
}
```

3. **Verify EEPROM data:**

```
void dumpEEPROM() {
  Serial.println("EEPROM Contents:");
  Serial.printf("Signature: 0x%04X (expect 0xAA55)\n", config.signature);
  Serial.printf("Activations: %lu\n", config.total_activations);
  Serial.printf("Duration: %lu seconds\n", config.total_water_duration);
  Serial.printf("Hand Min: %u mm\n", config.hand_min_distance);
  Serial.printf("Hand Max: %u mm\n", config.hand_max_distance);
}
```

4. **Manual EEPROM clear (jika corrupt):**

```
void clearEEPROM() {
  for (int i = 0; i < EEPROM_SIZE; i++) {
    EEPROM.write(i, 0);
  }
  EEPROM.commit();
  Serial.println("EEPROM cleared");
}
```

Problem 11: Solenoid Valve Tidak Kerja

Gejala:

- Relay “click” tapi tidak ada air keluar
- Atau valve stuck (tidak bisa buka/tutup)

Solusi:

1. **Cek power valve:**

```
Solenoid Valve butuh:
- Voltage: 12V DC
- Current: ~500mA

Wiring:
Power 12V (+) → Relay COM
Relay NO → Valve (+)
Valve (-) → Ground

Test: Ukur voltage di valve saat relay ON
Harus: ~12V
```

2. **Cek relay wiring:**

```
Relay Module (Aktif LOW):
GPIO 25 LOW  → Relay LED ON → Valve OPEN
GPIO 25 HIGH → Relay LED OFF → Valve CLOSED

Test relay:
digitalWrite(RELAY_PIN, LOW);   // Harus ada "click"
delay(2000);
digitalWrite(RELAY_PIN, HIGH);  // Harus ada "click" lagi
```

3. **Test valve manual:**

```
1. Lepaskan koneksi relay
2. Hubungkan valve langsung ke 12V
3. Jika valve buka → relay bermasalah
4. Jika valve tetap tidak buka → valve rusak
```

4. **Cek flow direction valve:**

```
Solenoid valve punya direction:
- IN: dari sumber air
- OUT: ke keran

Arrow di body valve menunjukkan arah flow
Pastikan terpasang dengan benar!
```

5. **Ganti valve jika rusak:**

```
Tanda valve rusak:
- Tidak ada bunyi "click" saat powered
- Air bocor saat valve closed
- Stuck open atau stuck closed
```

🎓 L. PENGEMBANGAN LEBIH LANJUT

Ide 1: Integrasi Sensor Flow Meter

Tujuan: Ukur volume air yang digunakan

Hardware Tambahan:

- YF-S201 Flow Sensor (1/2 inch)
- Interrupt-based counting

Implementasi:

```
#define FLOW_SENSOR_PIN 35
volatile uint16_t pulseCount = 0;
float flowRate = 0.0;
float totalVolume = 0.0;

void IRAM_ATTR flowPulseCounter() {
  pulseCount++;
}

void setup() {
  pinMode(FLOW_SENSOR_PIN, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(FLOW_SENSOR_PIN),
                  flowPulseCounter, FALLING);
}

void calculateFlowRate() {
  // YF-S201: 450 pulses/liter
  flowRate = (pulseCount / 450.0) * 60;  // L/min
  totalVolume += (pulseCount / 450.0);   // Total liter
  pulseCount = 0;
}
```

Dashboard Update:

- Total air terpakai: X liter
- Flow rate real-time: X L/min
- Efisiensi penggunaan air
- Alert jika penggunaan abnormal

Ide 2: MQTT Cloud Integration

Tujuan: Monitoring jarak jauh via cloud

Setup:

```
#include <PubSubClient.h>

WiFiClient espClient;
PubSubClient mqtt(espClient);

const char* mqtt_server = "broker.hivemq.com";
const char* topic_status = "masjid/wudhu/status";
const char* topic_stats = "masjid/wudhu/stats";

void publishStatus() {
  StaticJsonDocument<256> doc;
  doc["device_id"] = device_id;
  doc["water_state"] = system_state.water_state;
  doc["mode"] = system_state.mode;
  doc["sensor1_healthy"] = sensors.sensor1_healthy;
  doc["sensor2_healthy"] = sensors.sensor2_healthy;

  char buffer[256];
  serializeJson(doc, buffer);
  mqtt.publish(topic_status, buffer);
}

void loop() {
  mqtt.loop();

  // Publish every 5 seconds
  if (millis() - lastPublish > 5000) {
    publishStatus();
    lastPublish = millis();
  }
}
```

Mobile App (Node-RED):

- Dashboard real-time status
- Historical data charts
- Push notification untuk error
- Remote configuration

Ide 3: Multi-Kran System

Tujuan: Kontrol 4 kran wudhu dalam satu sistem

Hardware:

- ESP32 Master (dashboard)
- 4x ESP32 Slave (setiap kran)
- ESP-NOW untuk komunikasi

Arsitektur:

```
            ┌─────────────┐
            │ ESP32 Master│
            │ (Dashboard) │
            └──────┬──────┘
                   │
        ┌──────────┼──────────┐
        │          │          │
   ┌────▼───┐ ┌────▼───┐ ┌────▼───┐ ┌────────┐
```

```
│Kran 1│  │Kran 2│  │Kran 3│  │Kran 4│
└──────┘  └──────┘  └──────┘  └──────┘
```

ESP-NOW Implementation:

```
#include <esp_now.h>

typedef struct {
  uint8_t kran_id;
  bool water_state;
  uint8_t mode;
  uint16_t distance1;
  uint16_t distance2;
} KranData;

void onDataReceive(const uint8_t *mac, const uint8_t *data, int len) {
  KranData *kran = (KranData*)data;
  Serial.printf("Kran %d: Water=%d, Mode=%d\n",
                kran->kran_id, kran->water_state, kran->mode);

  // Update dashboard
  updateKranStatus(kran->kran_id, kran);
}
```

Dashboard Fitur:

- Status 4 kran sekaligus
- Total usage per kran
- Maintenance alert
- Centralized configuration

Ide 4: Voice Announcement

Tujuan: Audio feedback untuk pengguna

Hardware:

- DFPlayer Mini MP3 Module
- Speaker 3W

Implementasi:

```
#include <DFRobotDFPlayerMini.h>

DFRobotDFPlayerMini dfPlayer;

void setup() {
  Serial2.begin(9600);
  dfPlayer.begin(Serial2);
  dfPlayer.volume(20);
}

void playAnnouncement(uint8_t track) {
  dfPlayer.play(track);
}

// Di code:
void turnWaterOn() {
  digitalWrite(RELAY_PIN, LOW);
  playAnnouncement(1);  // "Silakan berwudhu"
}

void turnWaterOff() {
  digitalWrite(RELAY_PIN, HIGH);
  playAnnouncement(2);  // "Terima kasih"
}

void safetyTimeout() {
  playAnnouncement(3);  // "Waktu habis, mohon selesaikan"
}
```

Audio Files (SD Card):

1. 001.mp3: “Silakan berwudhu”
2. 002.mp3: “Terima kasih”
3. 003.mp3: “Waktu segera habis”
4. 004.mp3: “Sistem dalam pemeliharaan”
5. 005.mp3: “Mohon tunggu”

Ide 5: LCD Touchscreen Interface

Tujuan: Interface lokal tanpa perlu smartphone

Hardware:

- TFT LCD 2.8″ Touchscreen (ILI9341)
- SPI interface

UI Screens:

Home Screen:

```
┌─────────────────────────────┐
│   KRAN WUDHU OTOMATIS       │
├─────────────────────────────┤
│                             │
│   [   AUTO MODE   ]         │
│                             │
│   Status: READY             │
│   Sensor: OK                │
│                             │
│   [SETTINGS] [STATS]        │
└─────────────────────────────┘
```

Settings Screen:

```
┌─────────────────────────────┐
│   PENGATURAN                │
├─────────────────────────────┤
│                             │
│ Jarak Min:  [80 ] mm        │
│             [-] [+]         │
│                             │
│ Jarak Max:  [300] mm        │
│             [-] [+]         │
│                             │
│ Max Time:   [10 ] min       │
│             [-] [+]         │
│                             │
│      [SIMPAN] [BATAL]       │
└─────────────────────────────┘
```

Implementation:

```
#include <TFT_eSPI.h>

TFT_eSPI tft = TFT_eSPI();

void drawHomeScreen() {
  tft.fillScreen(TFT_BLACK);
  tft.setTextColor(TFT_WHITE, TFT_BLACK);
  tft.setTextSize(2);

  tft.setCursor(20, 20);
  tft.println("KRAN WUDHU");

  // Draw mode button
  tft.fillRoundRect(40, 60, 160, 50, 10, TFT_GREEN);
  tft.setTextColor(TFT_BLACK);
  tft.setCursor(60, 80);
  tft.println("AUTO MODE");

  // Draw status
  tft.setTextColor(TFT_WHITE);
  tft.setCursor(20, 130);
  tft.printf("Status: %s", system_state.water_state == WATER_ON ? "AKTIF" : "SIAP");
}

void handleTouch() {
  uint16_t x, y;
  if (tft.getTouch(&x, &y)) {
    // Detect button press
    if (y > 60 && y < 110 && x > 40 && x < 200) {
      // Mode button pressed
      toggleMode();
    }
  }
}
```

Ide 6: Solar Power System

Tujuan: Sistem mandiri untuk area tanpa listrik

Komponen:

- Solar Panel 20W
- Battery LiPo 12V 5000mAh
- Solar Charge Controller
- Step-down 12V→5V (untuk ESP32)

Power Budget:

```
ESP32:           ~200mA (average)
Sensors (2x):    ~40mA
OLED:            ~20mA
Relay (standby): ~5mA
Total:           ~265mA

Battery Life:
5000mAh / 265mA = ~19 jam

Solar Panel:
20W @ 12V = 1.67A (ideal)
Charging time: ~3 jam (sunny day)
```

Implementation:

```
#define BATTERY_PIN 36
#define SOLAR_PIN 39

float getBatteryVoltage() {
  int raw = analogRead(BATTERY_PIN);
  // Voltage divider 12V → 3.3V (R1=10k, R2=3.3k)
  float voltage = (raw / 4095.0) * 3.3 * (10.0 + 3.3) / 3.3;
  return voltage;
}

int getBatteryPercent() {
  float voltage = getBatteryVoltage();
  // LiPo 12V: 12.6V (full) - 10.0V (empty)
  int percent = map(voltage * 10, 100, 126, 0, 100);
  return constrain(percent, 0, 100);
}

void checkLowBattery() {
  int battery = getBatteryPercent();

  if (battery < 20) {
    // Low power mode
    system_state.mode = MODE_DEGRADED;
    display.ssd1306_command(SSD1306_DISPLAYOFF);  // OFF display

    // Reduce sensor read frequency
    SENSOR_READ_INTERVAL = 200;  // 200ms instead of 50ms

    logEvent("LOW_BATTERY_MODE");
  }

  if (battery < 10) {
    // Critical: force shutdown
    turnWaterOff();
    logEvent("CRITICAL_BATTERY_SHUTDOWN");
    esp_deep_sleep_start();
  }
}
```

Ide 7: Preventive Maintenance Alert

Tujuan: Notifikasi otomatis untuk maintenance

Features:

```
typedef struct {
  uint32_t valve_cycles;          // Jumlah ON/OFF valve
  uint32_t operation_hours;       // Total jam operasi
  uint32_t sensor_error_count;    // Total error sensor
  unsigned long last_maintenance; // Timestamp maintenance terakhir
} MaintenanceData;

MaintenanceData maintenance;

void checkMaintenanceNeeded() {
  bool needs_maintenance = false;
  String reason = "";

  // Check 1: Valve cycles (valve lifespan ~100k cycles)
  if (maintenance.valve_cycles > 50000) {
    needs_maintenance = true;
    reason += "Valve perlu service (>50k cycles)\n";
  }

  // Check 2: Operation hours (maintenance every 1000 hours)
  if (maintenance.operation_hours > 1000) {
    needs_maintenance = true;
    reason += "Service rutin 1000 jam\n";
  }

  // Check 3: High error rate
  if (maintenance.sensor_error_count > 100) {
    needs_maintenance = true;
    reason += "Sensor error rate tinggi\n";
  }

  // Check 4: Last maintenance >6 months
  if (millis() - maintenance.last_maintenance > 15552000000) {  // 6 months
    needs_maintenance = true;
    reason += "Maintenance terakhir >6 bulan\n";
  }

  if (needs_maintenance) {
    // Send notification
    displayWarning("MAINTENANCE DIPERLUKAN");
    sendMaintenanceAlert(reason);
    logEvent("MAINTENANCE_ALERT_TRIGGERED");
  }
}

void sendMaintenanceAlert(String reason) {
  // Via MQTT/Telegram/Email
  StaticJsonDocument<512> doc;
  doc["device_id"] = device_id;
  doc["alert_type"] = "maintenance";
  doc["reason"] = reason;
  doc["valve_cycles"] = maintenance.valve_cycles;
  doc["operation_hours"] = maintenance.operation_hours;
  doc["error_count"] = maintenance.sensor_error_count;

  // Publish to cloud
  // ...
}
```

Ide 8: Water Quality Monitoring

Tujuan: Monitor kualitas air wudhu

Sensor Tambahan:

- TDS Sensor (Total Dissolved Solids)
- pH Sensor
- Temperature Sensor (DS18B20)

Implementation:

```
#define TDS_PIN 34
#define PH_PIN 35

struct WaterQuality {
  float tds;          // ppm (parts per million)
  float ph;           // pH value (0-14)
  float temperature;  // Celsius
  bool is_safe;       // Safe for wudhu
};

WaterQuality water_quality;

void readWaterQuality() {
  // Read TDS
  int tdsRaw = analogRead(TDS_PIN);
  water_quality.tds = (133.42 * tdsRaw * tdsRaw * tdsRaw
                      - 255.86 * tdsRaw * tdsRaw
                      + 857.39 * tdsRaw) * 0.5;

  // Read pH
  int phRaw = analogRead(PH_PIN);
  water_quality.ph = 3.5 * phRaw / 1024.0 + 0.0;  // Calibration needed

  // Read temperature (DS18B20)
  water_quality.temperature = sensors.getTempCByIndex(0);

  // Check if safe
  water_quality.is_safe = (water_quality.tds < 500 &&      // TDS < 500 ppm
                          water_quality.ph > 6.5 &&        // pH 6.5-8.5
                          water_quality.ph < 8.5 &&
                          water_quality.temperature < 40); // < 40°C

  if (!water_quality.is_safe) {
    displayWarning("KUALITAS AIR TIDAK NORMAL");
    logError("WATER_QUALITY_UNSAFE");
  }
}
```

Dashboard Display:

```
┌─────────────────────────────┐
│  KUALITAS AIR               │
├─────────────────────────────┤
│  TDS:  245 ppm  ✓ BAIK      │
│  pH:   7.2      ✓ BAIK      │
│  Suhu: 28°C     ✓ BAIK      │
│                             │
│  Status: AMAN UNTUK WUDHU   │
└─────────────────────────────┘
```

📚 M. LAMPIRAN

Lampiran A: Pin Reference Card

```
╔═════════════════════════════════════════╗
║  ESP32 PIN QUICK REFERENCE              ║
╠═════════════════════════════════════════╣
║  I2C Communication:                     ║
║  ├─ GPIO 21 (SDA)   : I2C Data          ║
║  └─ GPIO 22 (SCL)   : I2C Clock         ║
║                                         ║
║  Sensor Control:                        ║
║  ├─ GPIO 32 (XSHUT): Sensor 1 Enable    ║
║  └─ GPIO 33 (XSHUT): Sensor 2 Enable    ║
║                                         ║
║  Output Control:                        ║
║  ├─ GPIO 25         : Relay (Aktif LOW) ║
║  └─ GPIO 26         : LED Indicator     ║
║                                         ║
║  Input:                                 ║
║  └─ GPIO 34         : Factory Reset Btn ║
║                                         ║
║  Power:                                 ║
║  ├─ 3.3V  : Sensors, OLED (max 600mA)   ║
║  ├─ 5V    : Relay Module                ║
║  ├─ 12V   : Solenoid Valve              ║
║  └─ GND   : Common Ground               ║
╚═════════════════════════════════════════╝
```

Lampiran B: VL53L0X Datasheet Summary

```
╔═════════════════════════════════════════╗
║  VL53L0X ToF SENSOR SPECIFICATIONS      ║
╠═════════════════════════════════════════╣
║  Measurement:                           ║
║  ├─ Range: 30mm - 2000mm                ║
║  ├─ Accuracy: ±3mm @ 50-1000mm          ║
║  ├─ Field of View: 25°                  ║
║  └─ Update Rate: Max 50Hz               ║
║                                         ║
║  Interface:                             ║
║  ├─ Protocol: I2C                       ║
║  ├─ Default Address: 0x29               ║
║  ├─ Clock Speed: Up to 400kHz           ║
║  └─ XSHUT: Active-low shutdown          ║
║                                         ║
║  Power:                                 ║
║  ├─ Voltage: 2.6V - 3.5V (3.3V typ)     ║
║  ├─ Current: ~20mA (measuring)          ║
║  └─ Shutdown: <1µA                      ║
║                                         ║
║  Operating Modes:                       ║
║  ├─ Single Shot: One measurement        ║
║  ├─ Continuous: Repeated reads          ║
║  └─ Timing Budget: 20-200ms             ║
╚═════════════════════════════════════════╝
```

Lampiran C: Error Codes Reference

```
╔═════════════════════════════════════════╗
║  ERROR CODES & SOLUTIONS                ║
╠═════════════════════════════════════════╣
║  E01: SENSOR_INIT_FAILED                ║
║  └─ Cek I2C, power 3.3V, XSHUT pins     ║
║                                         ║
║  E02: OLED_INIT_FAILED                  ║
║  └─ Cek I2C address (0x3C/0x3D)         ║
║                                         ║
║  E03: SENSOR_TIMEOUT                   ║
║  └─ Sensor tidak respond, cek kabel    ║
║                                        ║
║  E04: I2C_COMMUNICATION_ERROR          ║
║  └─ Cek pull-up, panjang kabel, speed  ║
║                                        ║
║  E05: EMERGENCY_SHUTOFF                ║
║  └─ Kedua sensor error, air force OFF  ║
║                                        ║
║  E06: SAFETY_TIMEOUT                   ║
║  └─ Max water time reached (normal)    ║
║                                        ║
║  E07: WATCHDOG_TIMEOUT                 ║
║  └─ System hang, auto restart          ║
║                                        ║
║  E08: LOW_BATTERY                      ║
║  └─ Battery <20%, degraded mode        ║
║                                        ║
║  E09: WATER_QUALITY_UNSAFE             ║
║  └─ TDS/pH abnormal, check water       ║
║                                        ║
║  E10: MAINTENANCE_ALERT                ║
║  └─ Service needed (info only)         ║
╚════════════════════════════════════════╝
```

Lampiran D: Konfigurasi Default

```
// EEPROM Default Values
config.signature = 0xAA55;
config.version = 0x0310;  // v3.1.0
config.total_activations = 0;
config.total_water_duration = 0;
config.hand_min_distance = 80;      // mm
config.hand_max_distance = 300;     // mm
config.swipe_distance = 50;         // mm
config.water_off_delay = 1500;      // ms
config.max_water_time = 600000;     // ms (10 menit)
config.operation_hours = 0;
config.error_count = 0;

// Timing Constants
DEBOUNCE_TIME = 300;               // ms
AP_TIMEOUT = 300000;               // ms (5 menit)
ERROR_RECOVERY_INTERVAL = 10000;   // ms (10 detik)
MAINTENANCE_INTERVAL = 3600000;    // ms (1 jam)
WATCHDOG_INTERVAL = 8000;          // ms (8 detik)
SENSOR_READ_INTERVAL = 50;         // ms

// WiFi AP
SSID: "WudhuConfig"
Password: "Wudhu_[device_id]"  // Dynamic
IP: 192.168.4.1
```

Lampiran E: Wiring Diagram (Full)

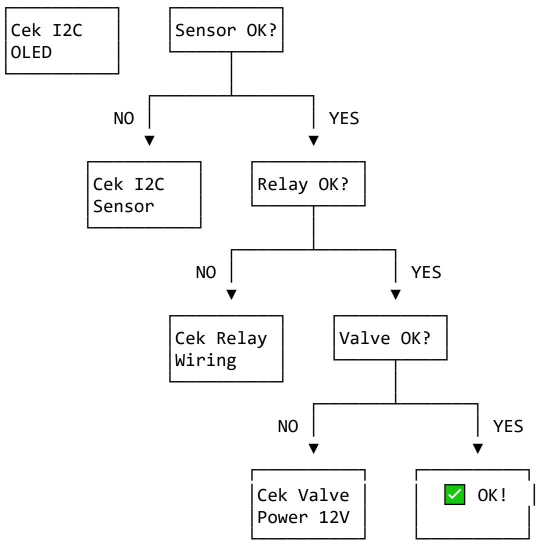
```
┌──────────────────────────────────────────────────────────────┐
│                      WIRING DIAGRAM                          │
└──────────────────────────────────────────────────────────────┘

Power Supply (12V/2A):
    12V ──────┬──── Solenoid Valve (+)
              │
              └──── Step-down Converter (12V→5V)
                             │
                            5V ──────┬──── ESP32 VIN
                                     ├──── Relay Module VCC
                                     └──── (Power distribution)

ESP32 WROOM 32D:
    3.3V ──────┬──── VL53L0X Sensor 1 VCC
               ├──── VL53L0X Sensor 2 VCC
               └──── OLED VCC

    GPIO 21 (SDA) ─────┬──── Sensor 1 SDA
                       ├──── Sensor 2 SDA
                       └──── OLED SDA

    GPIO 22 (SCL) ─────┬──── Sensor 1 SCL
                       ├──── Sensor 2 SCL
                       └──── OLED SCL

    GPIO 32 ───> Sensor 1 XSHUT
    GPIO 33 ───> Sensor 2 XSHUT
    GPIO 25 ───> Relay IN
    GPIO 26 ───> LED (+)
    GPIO 34 <─── Factory Reset Button

    GPIO ────┬─── Sensor 1 GND
             ├─── Sensor 2 GND
             ├─── OLED GND
             ├─── Relay GND
             ├─── LED GND (via resistor 330Ω)
             ├─── Button GND
             └─── Power Supply GND

Relay Module:
    COM ───> Power 12V (+)
    NO  ───> Solenoid Valve (+)

Solenoid Valve:
    (+) ───< Relay NO
    (-) ───> Ground

LED:
    (+) ───< GPIO 26
    (-) ───[330Ω]─── GND
```

Lampiran F: Maintenance Checklist

```
╔════════════════════════════════════════╗
║ JADWAL MAINTENANCE                     ║
╠════════════════════════════════════════╣
║ HARIAN:                                ║
║ □ Cek status sistem via OLED           ║
║ □ Test manual/auto mode                ║
║ □ Cek tidak ada kebocoran air          ║
║ □ Monitor serial log untuk error       ║
║                                        ║
║ MINGGUAN:                              ║
║ □ Bersihkan sensor (lap kering)        ║
║ □ Cek koneksi kabel                    ║
║ □ Test swipe gesture                   ║
║ □ Review statistics (AP mode)          ║
║ □ Cek LED indicator                    ║
║                                        ║
║ BULANAN:                               ║
║ □ Backup configuration (AP mode)       ║
║ □ Test emergency shutoff               ║
║ □ Cek kondisi relay (bunyi click?)     ║
║ □ Test factory reset                   ║
║ □ Kalibrasi sensor jika perlu          ║
║ □ Update firmware jika ada versi baru  ║
║                                        ║
║ SEMESTERAN (6 BULAN):                  ║
║ □ Service solenoid valve               ║
║ □ Ganti filter air (jika ada)          ║
║ □ Deep cleaning semua komponen         ║
║ □ Cek kondisi PCB (korosi?)            ║
║ □ Test semua mode operasi              ║
║ □ Verifikasi accuracy sensor           ║
║                                        ║
║ TAHUNAN:                               ║
║ □ Replace sensor jika performance ↓    ║
║ □ Ganti solenoid valve preventive      ║
║ □ Full system audit                    ║
║ □ Update dokumentasi                   ║
║ □ Training user baru                   ║
╚════════════════════════════════════════╝
```

Lampiran G: Troubleshooting Flowchart

```
          System Tidak Bekerja?
                  │
                  ▼
          ┌───────────────┐
          │  Power ON?    │
          └───────────────┘
                  │
          ┌───────┴───────┐
      NO  │               │  YES
          ▼               ▼
    ┌──────────┐    ┌──────────┐
    │ Cek PSU  │    │OLED OK?  │
    │ & Kabel  │    └──────────┘
    └──────────┘          │
                  ┌───────┴───────┐
              NO  │               │  YES
                  ▼               ▼
```



Lampiran H: Serial Commands (Debug)

Untuk debugging, tambahkan di `loop()`:

```
void handleSerialCommands() {
  if (Serial.available()) {
    char cmd = Serial.read();

    switch(cmd) {
      case 'h':  // Help
        printHelp();
        break;

      case 's':  // Status
        printStatus();
        break;

      case 'o':  // Water ON
        turnWaterOn();
        Serial.println("Water forced ON");
        break;

      case 'f':  // Water OFF
        turnWaterOff();
        Serial.println("Water forced OFF");
        break;

      case 'a':  // Activate AP
        activateAPMode();
        break;

      case 'r':  // Factory Reset
        performFactoryReset();
        break;

      case 'd':  // Dump EEPROM
        dumpEEPROM();
        break;

      case 'c':  // Save Config
        saveConfiguration();
        Serial.println("Config saved");
        break;

      case 't':  // Test sensors
        testSensors();
        break;

      case 'm':  // Toggle mode
        system_state.mode = (system_state.mode == MODE_AUTO) ?
                            MODE_MANUAL : MODE_AUTO;
        Serial.printf("Mode: %d\n", system_state.mode);
        break;

      case 'i':  // System info
        printSystemInfo();
        break;

      default:
        Serial.println("Unknown command. Press 'h' for help.");
    }
  }
}

void printHelp() {
  Serial.println("\n=== DEBUG COMMANDS ===");
  Serial.println("h - Help (this menu)");
  Serial.println("s - Status");
  Serial.println("o - Water ON");
  Serial.println("f - Water OFF");
  Serial.println("a - Activate AP");
  Serial.println("r - Factory Reset");
  Serial.println("d - Dump EEPROM");
  Serial.println("c - Save Config");
  Serial.println("t - Test Sensors");
  Serial.println("m - Toggle Mode");
  Serial.println("i - System Info");
  Serial.println("===================\n");
}

void printStatus() {
  Serial.println("\n=== SYSTEM STATUS ===");
  Serial.printf("Mode: %d\n", system_state.mode);
  Serial.printf("Water: %s\n", system_state.water_state == WATER_ON ? "ON" : "OFF");
  Serial.printf("Sensor 1: %umm (%s)\n", sensors.distance1,
                sensors.sensor1_healthy ? "OK" : "ERROR");
  Serial.printf("Sensor 2: %umm (%s)\n", sensors.distance2,
                sensors.sensor2_healthy ? "OK" : "ERROR");
  Serial.printf("Errors: %u\n", sensors.consecutive_errors);
  Serial.printf("Uptime: %lu s\n", millis()/1000);
  Serial.println("===================\n");
}

void testSensors() {
  Serial.println("\n=== SENSOR TEST ===");

  for (int i = 0; i < 10; i++) {
    uint16_t d1 = sensor1.readRangeContinuousMillimeters();
    uint16_t d2 = sensor2.readRangeContinuousMillimeters();

    Serial.printf("Test %d: S1=%umm S2=%umm\n", i+1, d1, d2);
    delay(100);
  }

  Serial.println("Test complete\n");
}

void printSystemInfo() {
  Serial.println("\n=== SYSTEM INFO ===");
  Serial.println("Model: " + String(MODEL_NUMBER));
  Serial.println("Firmware: " + String(FIRMWARE_VERSION));
  Serial.println("Serial: " + String(SERIAL_NUMBER));
  Serial.println("Device ID: " + device_id);
  Serial.printf("Free RAM: %u bytes\n", ESP.getFreeHeap());
  Serial.printf("Total Activations: %lu\n", config.total_activations);
  Serial.printf("Total Duration: %lu seconds\n", config.total_water_duration);
  Serial.printf("Operation Hours: %lu\n", config.operation_hours);
  Serial.printf("Error Count: %u\n", config.error_count);
  Serial.println("==================\n");
}

void dumpEEPROM() {
  Serial.println("\n=== EEPROM DUMP ===");
  Serial.printf("Signature: 0x%04X\n", config.signature);
  Serial.printf("Version: 0x%04X\n", config.version);
  Serial.printf("Total Activations: %lu\n", config.total_activations);
  Serial.printf("Water Duration: %lu s\n", config.total_water_duration);
  Serial.printf("Hand Min: %u mm\n", config.hand_min_distance);
  Serial.printf("Hand Max: %u mm\n", config.hand_max_distance);
  Serial.printf("Swipe Dist: %u mm\n", config.swipe_distance);
  Serial.printf("Water Delay: %lu ms\n", config.water_off_delay);
  Serial.printf("Max Time: %lu ms\n", config.max_water_time);
  Serial.printf("Op Hours: %lu\n", config.operation_hours);
  Serial.printf("Errors: %u\n", config.error_count);
  Serial.printf("CRC: 0x%04X\n", config.crc);
  Serial.println("==================\n");
}
```

Cara Pakai:

1. Buka Serial Monitor (115200 baud)
2. Ketik huruf command (tanpa Enter di beberapa terminal)
3. Tekan Enter
4. Lihat output

Contoh Session:

```
>>> h
=== DEBUG COMMANDS ===
h - Help (this menu)
s - Status
...

>>> s
=== SYSTEM STATUS ===
Mode: 0 (AUTO)
Water: OFF
Sensor 1: 150mm (OK)
Sensor 2: 200mm (OK)
...

>>> o
Water forced ON

>>> f
Water forced OFF
```

Lampiran I: Upgrade Path

Version History:

```
v3.1.0 (Current) - 2024
├─ Production-ready code
├─ Advanced error handling
├─ Watchdog protection
├─ Web dashboard with auth
├─ QR code support
└─ EEPROM with CRC

v3.0.0 - 2024
├─ Multiple sensor support
├─ Gesture detection (swipe)
├─ AP configuration mode
└─ Basic web interface

v2.0.0 - 2023
├─ Single ToF sensor
├─ Basic auto detection
└─ OLED display

v1.0.0 - 2023
├─ Ultrasonic sensor
└─ Manual control only
```

Upgrade ke v4.0.0 (Planned):

```
New Features:
□ MQTT cloud integration
□ Flow meter support
□ Water quality monitoring
□ Multi-kran system (ESP-NOW)
□ LCD touchscreen option
□ Voice announcement
□ Solar power support
□ Mobile app (Flutter)

Breaking Changes:
- EEPROM structure updated (auto migrate)
- API endpoint changes (/api/v2/)
- New pin assignments for additional sensors

Migration Steps:
1. Backup current config (AP mode → download)
2. Upload new firmware
3. Config auto-migrate
4. Verify settings in dashboard
5. Test all modes
```

Lampiran J: Glossary (Istilah Teknis)

```
ADC (Analog-to-Digital Converter):
  Komponen yang mengubah sinyal analog ke digital
  ESP32 punya 18 channel ADC 12-bit (0-4095)

AP (Access Point):
  Mode WiFi dimana ESP32 bertindak sebagai hotspot
  Devices lain connect ke ESP32 untuk konfigurasi

CRC (Cyclic Redundancy Check):
  Algoritma deteksi error pada data
  Digunakan untuk validasi EEPROM integrity

Debouncing:
  Teknik eliminasi noise/bouncing pada button/sensor
  Mencegah multiple trigger dalam waktu singkat

EEPROM (Electrically Erasable Programmable ROM):
  Memory non-volatile untuk menyimpan konfigurasi
  Data tidak hilang saat power OFF

ESP-NOW:
  Protokol komunikasi peer-to-peer ESP32
  Low latency, tidak butuh WiFi router

Field of View (FoV):
  Sudut pandang sensor
  VL53L0X: 25° (cone shape)

Gesture Recognition:
  Deteksi gerakan tangan untuk input
  Swipe, tap, hover, dll

Hysteresis:
  Gap antara threshold ON dan OFF
  Mencegah oscillation/flickering

I2C (Inter-Integrated Circuit):
  Protokol komunikasi serial 2-wire
  SDA (data) + SCL (clock)

ISR (Interrupt Service Routine):
  Fungsi yang dipanggil saat interrupt trigger
  Harus singkat dan cepat (tidak boleh delay)

JSON (JavaScript Object Notation):
  Format data text untuk pertukaran data
  Lightweight dan human-readable

MQTT (Message Queuing Telemetry Transport):
  Protokol messaging untuk IoT
  Publish-subscribe model

Pull-up Resistor:
  Resistor yang menarik line ke HIGH (3.3V)
  I2C butuh pull-up 4.7kΩ di SDA & SCL

PWM (Pulse Width Modulation):
  Teknik kontrol power dengan mengatur duty cycle
  Digunakan untuk dimming LED, motor speed, dll

RESTful API:
  Web service architecture
  HTTP methods: GET, POST, PUT, DELETE

Solenoid Valve:
  Valve elektrik yang dibuka/tutup dengan elektromagnet
  Normally Closed (NC) atau Normally Open (NO)

State Machine:
  Model komputasi dengan states dan transitions
  Used untuk manage system modes

Time of Flight (ToF):
  Metode pengukuran jarak dengan waktu tempuh cahaya
  Laser dipacarkan → pantul → diterima → hitung waktu

Watchdog Timer:
  Safety mechanism untuk detect system hang
  Auto restart jika tidak di-feed dalam timeout period

XSHUT (External Shutdown):
  Pin untuk shutdown sensor (aktif LOW)
  Digunakan untuk change I2C address
```

🙏 N. PENUTUP

Pesan untuk Anggota IRMAL:

```
╔══════════════════════════════════════╗
║                                      ║
║  "Teknologi adalah amanah,           ║
║   gunakan untuk kebaikan bersama"    ║
║                                      ║
║  Proyek Kran Wudhu Otomatis ini      ║
║  adalah langkah lanjutan dalam       ║
║  pembelajaran IoT. Lebih kompleks    ║
║  dari project sebelumnya, dengan     ║
║  sensor presisi tinggi, error        ║
║  handling yang robust, dan production-
║  ready code.                             ║
║                                          ║
║  Kalian telah belajar:                   ║
║  ✓ Time of Flight sensor technology      ║
║  ✓ Advanced I2C multi-device             ║
║  ✓ Gesture recognition                   ║
║  ✓ Production-level error handling       ║
║  ✓ Web dashboard dengan security         ║
║  ✓ EEPROM data management                ║
║  ✓ Watchdog & safety features            ║
║                                          ║
║  Terus belajar, terus berkembang,        ║
║  dan selalu ingat: teknologi adalah      ║
║  untuk kemudahan ibadah dan manfaat      ║
║  umat.                                   ║
║                                          ║
║  Semoga proyek ini membawa berkah.       ║
║                                          ║
║  Barakallahu fiikum!                     ║
║                                          ║
║  - Irnity                                ║
║    Ikhlas Research Network for           ║
║    Innovation, Technology & Youth        ║
║    2025                                  ║
╚══════════════════════════════════════════╝
```

Next Steps:

```
UNTUK PEMULA:
1. Pahami konsep dasar ToF sensor
2. Praktik rakit hardware step-by-step
3. Upload code dan monitor serial log
4. Test setiap mode operasi
5. Pahami troubleshooting dasar

UNTUK INTERMEDIATE:
1. Modifikasi parameter di code
2. Tambahkan serial debug commands
3. Eksperimen dengan gesture detection
4. Customize web dashboard
5. Implementasi flow meter

UNTUK ADVANCED:
1. Integrasi MQTT cloud
2. Develop multi-kran system
3. Implementasi ML prediction
4. Buat mobile app
5. Kontribusi improvement code
6. Jadi mentor untuk junior

PROJECT SELANJUTNYA:
□ Smart Lighting Masjid (PIR + LDR)
□ Attendance System (RFID/Fingerprint)
□ Prayer Time Display (RTC + 7-segment)
□ Temperature Monitor (DHT22 + Telegram)
□ Security System (Camera + Motion)
```

Feedback & Support:

📧 Email: irnity.ikhlas@gmail.com
📱 WhatsApp:
🌐 Website:
💻 GitHub:
📺 YouTube:

Doa Penutup:

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

رَبِّ زِدْنِي عِلْمًا وَارْزُقْنِي فَهْمًا

"Ya Allah, tambahkanlah ilmu bagiku
dan anugerahkan kepadaku pemahaman"

اَللَّهُمَّ انْفَعْنِيْ بِمَا عَلَّمْتَنِيْ
وَعَلِّمْنِيْ مَا يَنْفَعُنِيْ

"Ya Allah, manfaatkanlah ilmu yang telah
Engkau ajarkan kepadaku, dan ajarkanlah
kepadaku ilmu yang bermanfaat bagiku"

آمِينَ يَا رَبَّ الْعَالَمِينَ

Keep Innovating, Keep Learning, Keep Contributing!

```
╔══════════════════════════════════╗
║     Smart Wudhu Faucet System    ║
║       Version 3.1.0-ESP32        ║
║      Praktik IoT 2 - IRMAL       ║
║     Masjid Jami Al-Ikhlas        ║
║              2025                ║
╚══════════════════════════════════╝

Developed with ❤️ by Irnity
For the benefit of ummah
Production-ready • Reliable • Efficient

Barakallahu fiikum!
```

[END OF DOCUMENTATION]

📄 **Total Pages:** 60+ halaman
📊 **Total Words:** ~18,000 kata
⏱️ **Reading Time:** ~3-4 jam
🎯 **Skill Level:** Intermediate to Advanced
📅 **Last Updated:** Desember 2025
🏷️ **Version:** 1.0.0

Catatan: Dokumentasi ini adalah living document yang akan terus diupdate seiring development dan feedback dari anggota IRMAL. Jangan ragu untuk berkontribusi, memberikan saran, atau melaporkan bug.

Semoga Allah SWT memberikan keberkahan dalam setiap ilmu yang kita pelajari dan amalkan. Semoga project ini menjadi amal jariyah yang terus memberikan manfaat. Aamiin Ya Rabbal 'Aalamiin.
