

**SISTEM DETEKSI KEBAKARAN DALAM
RUANGAN BERBASIS IOT MENGGUNAKAN
SENSOR API, SUHU, KELEMBAPAN DAN
ASAP**

LATAR BELAKANG



1

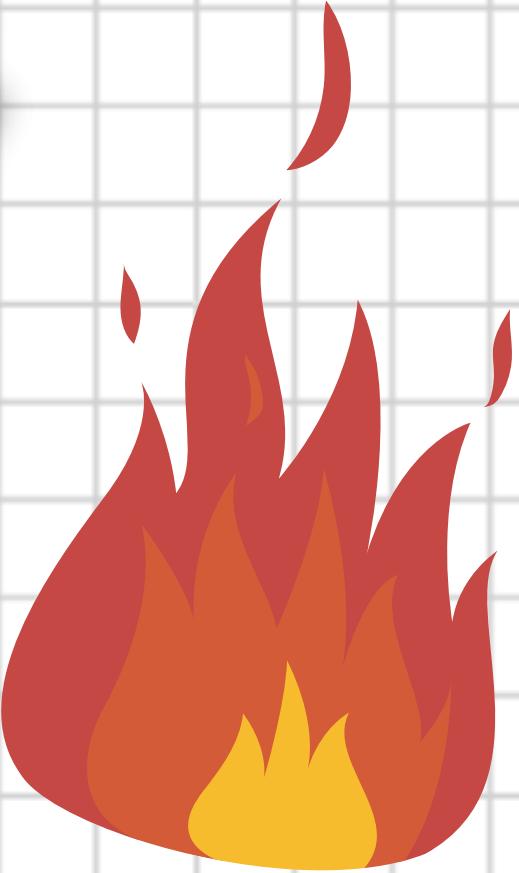
Kebakaran dalam ruangan berisiko tinggi terhadap keselamatan jiwa dan kerugian properti besar.

2

Sistem deteksi konvensional kurang efektif memberi peringatan dini dan respons cepat.

3

Teknologi IoT dimanfaatkan untuk membuat deteksi kebakaran real-time dengan respons otomatis yang efisien.



TUJUAN

Mengembangkan sistem deteksi dan penekanan kebakaran dalam ruangan berbasis IoT yang cerdas, responsif, dan dapat dipantau secara real-time, guna meningkatkan keselamatan dengan deteksi cepat, penanganan otomatis, serta pengurangan alarm palsu melalui logika fuzzy.



LANDASAN TEORI

1. Internet of Things (IoT)

Internet of Things (IoT) adalah teknologi yang menghubungkan perangkat fisik agar bisa saling berkomunikasi dan bertukar data melalui internet secara otomatis. Dalam sistem keamanan kebakaran, IoT digunakan untuk mendeteksi tanda-tanda awal kebakaran, merespons secara otomatis, dan memantau kondisi secara real-time lewat sensor dan alat pemadam yang terintegrasi.

LANDASAN TEORI



2. Sensor API (IR)

Mendeteksi nyala api melalui radiasi inframerah (760–1100 nm). Respon sangat cepat, mampu memberi peringatan sejak api mulai menyala, sebelum asap dan panas menyebar.

3. Sensor Suhu (DHT11) & Asap (MQ-2)

DHT11 mengukur suhu dan kelembaban sebagai indikator awal kebakaran. MQ-2 mendeteksi asap dan gas mudah terbakar untuk menguatkan deteksi kebakaran.

LANDASAN TEORI

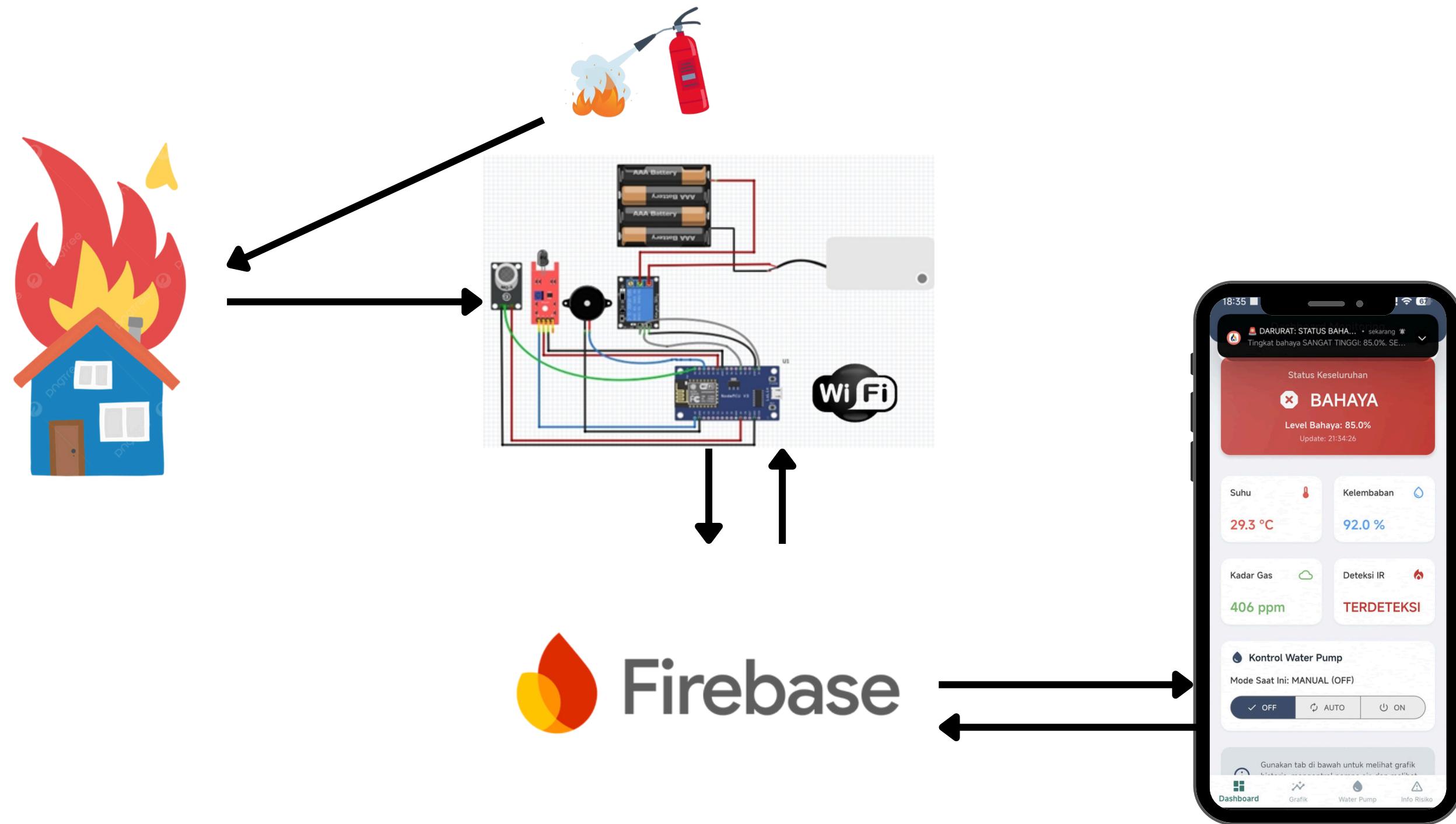
4. Logika Fuzzy

Memproses data sensor dalam bentuk linguistik (misal: suhu "tinggi") untuk menentukan tingkat risiko kebakaran (aman, peringatan, bahaya) secara bertahap, sehingga respons lebih tepat dan alarm palsu berkurang.

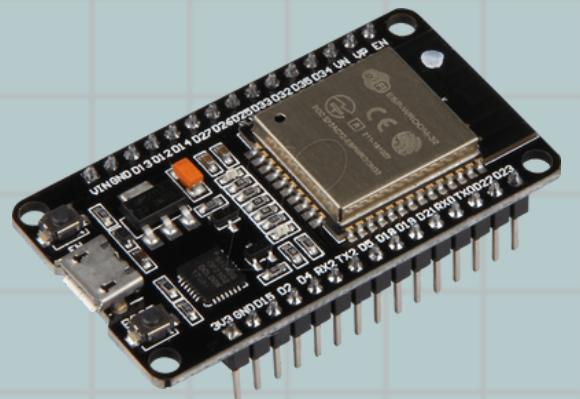
5. Firebase Realtime Database

Menyimpan dan menyinkronkan data sensor secara real-time. Digunakan untuk pencatatan data, pemantauan jarak jauh via aplikasi, dan pengiriman notifikasi saat terdeteksi kondisi bahaya.

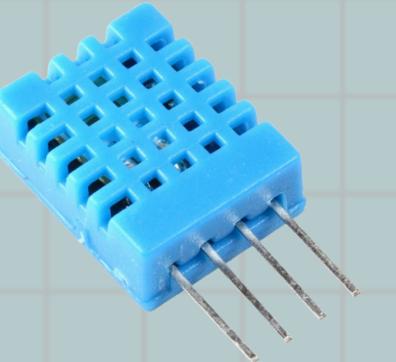
ALUR KERJA



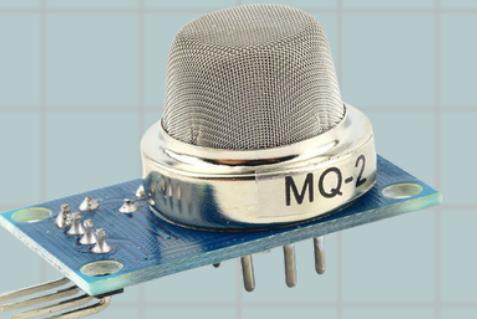
ALAT DAN BAHAN



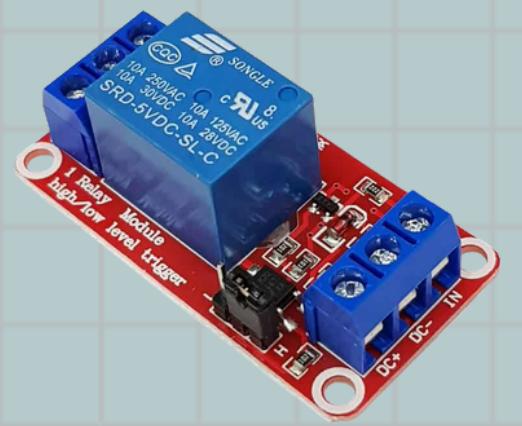
ESP32



DHT11



MQ-2



RELAY



BUZZER



SENSOR IR



POMPA



Firebase

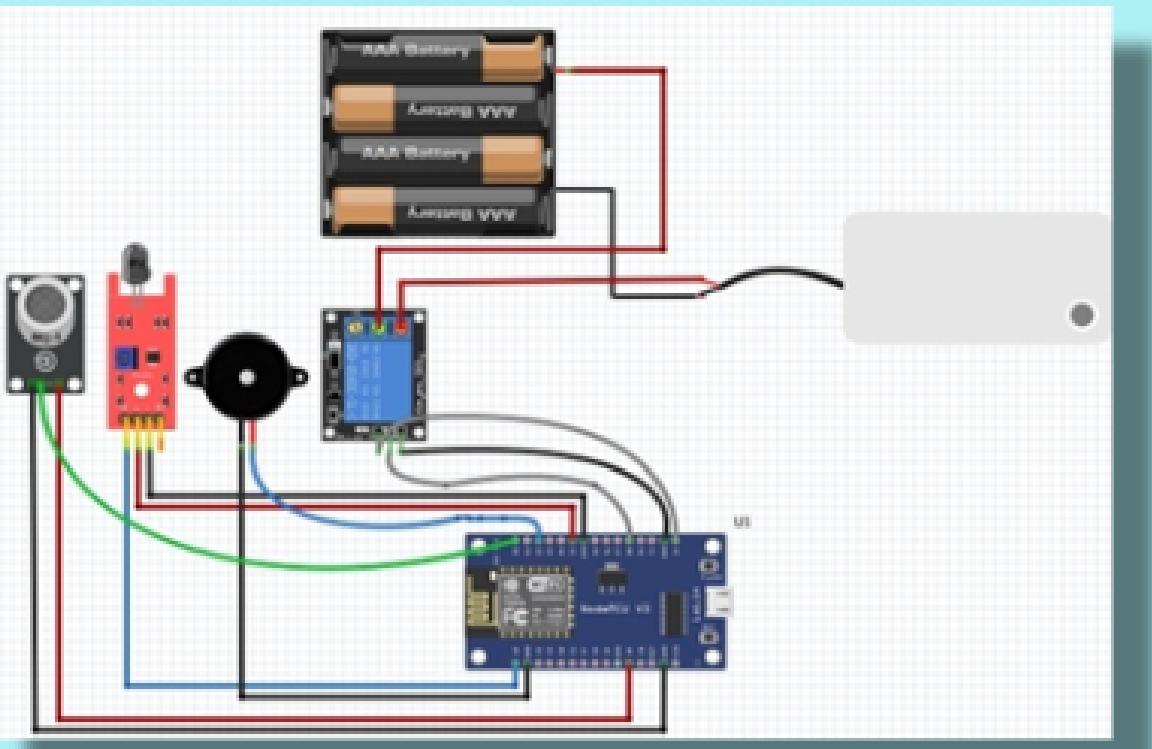


ARDUINO IDE

GAMBAR RANGKAIAN

Skema Koneksi Rangkaian

- Sensor IR: DO → GPIO 14
- DHT22: DATA → GPIO 15
- MQ-2: AO → GPIO 36
- Relay: IN → GPIO 23
- Buzzer: VCC 3.3V, GND
- Pompa Air: Terkoneksi ke relay, sumber eksternal 5–12V
- Semua VCC ke 3.3V, GND ke GND ESP32



PENJELASAN KODE **MAIN.INO**

Program ini memiliki dua fase utama: Fase Setup (yang berjalan sekali saat dinyalakan) dan Fase Loop (yang berjalan terus-menerus sesudahnya).

Fase 1: setup() – Inisialisasi dan Persiapan Sistem

Ketika Anda menyalakan ESP32, fungsi `setup()` akan dieksekusi baris per baris.

- **Serial.begin(115200);**

ESP32 mengaktifkan port komunikasi serialnya dengan kecepatan (baud rate) 115200 bit per detik.

- **dht.begin();**

Menginisialisasi komunikasi dengan sensor DHT11.

- **pinMode(...)**

Mengatur mode fungsionalitas untuk setiap pin GPIO.

`pinMode(MQ2_PIN, INPUT);`: Pin 34

`pinMode(IR_PIN, INPUT_PULLUP);`: Pin 26

`pinMode(BUZZER_PIN, OUTPUT);` dan `pinMode(WATERPUMP_PIN, OUTPUT);`: Pin 25 dan 23

- **digitalWrite(...)**

Mengatur kondisi awal untuk pin output.

`digitalWrite(BUZZER_PIN, LOW);`: Mengirim sinyal LOW ke buzzer,

`digitalWrite(WATERPUMP_PIN, LOW);`: Mengirim sinyal LOW ke pin pompa. Untuk menyalakan dan mematikan waterpump

```
connectToWiFi();  
if (WiFi.status() == WL_CONNECTED) {  
    initTime();  
  
    config.api_key = API_KEY;  
    config.database_url = DATABASE_URL;  
  
    // PENTING: Karena aturan RTDB Anda .read:  
    Firebase.begin(&config, nullptr);  
    Serial.println("Firebase initialized with
```

- **connectToWiFi();**

ESP32 memulai proses koneksi ke SSID "xrp" dengan password "mstahulhaq".

- **if (WiFi.status() == WL_CONNECTED)**

Program memeriksa apakah koneksi Wi-Fi berhasil.

- **initTime();**

Mengkonfigurasi ESP32 untuk mengambil data waktu dari server NTP (server waktu di internet)

- **Inisialisasi Firebase (Bagian Kritis)**

config.api_key = ...

config.database_url = ...

Menyalin kredensial proyek Firebase Anda ke dalam objek config.

- **Firebase.begin(&config, nullptr);**: Ini adalah inisialisasi pertama. nullptr berarti "tanpa otentikasi". Ini cocok jika aturan keamanan Firebase Anda adalah ".read": "true" dan ".write": "true".

Fase 2: loop() - Siklus Operasi Tiada Henti

Setelah setup() selesai, fungsi loop() akan berjalan berulang kali. Mari kita telusuri satu siklus.

Langkah 1: Baca Semua Sensor (Input)

Sensor yang dibaca:

- Suhu (**temperature**)
- Kelembaban (**humidity**)
- GasMQ2 (**mq2RawValue**)
- Api via IR (**irDetectedFire**)

Langkah 2: Proses Berpikir (Fuzzy Logic)

Fungsi utama: **calculateFuzzyLogic(...)**

- **Fuzzifikasi:** Ubah nilai sensor jadi derajat keanggotaan (rendah/sedang/tinggi)
- **Evaluasi Aturan:** Terapkan logika IF-THEN untuk menghitung seberapa "aman/waspada/bahaya"
- **Rule khusus:** Jika **irDetectedFire == true**, langsung dianggap "bahaya"
- **Defuzzifikasi:** Gabungkan hasil jadi 1 angka: **fuzzyDangerLevelOutput** (misal: 88.5)
- **Hasil akhir:** **fuzzyStatusOutput** → "AMAN" / "WASPADA" / "BAHAYA"

Langkah 3: Ambil Tindakan (Output)

- Jika status **BAHAYA** atau **WASPADA (>60)**:
 - Buzzer **MENYALA**
 - Pompa air **MENYALA**
- Jika tidak:
 - Buzzer & pompa **mati**

Fase 2: loop() - Siklus Operasi Tiada Henti

Setelah setup() selesai, fungsi loop() akan berjalan berulang kali. Mari kita telusuri satu siklus.

Langkah 4: Laporan (Komunikasi)

- **Debugging:** Semua data dicetak ke Serial Monitor
- **Firebase:**
 - Data sensor dikemas dalam JSON
 - Dikirim ke path **/fire_detection**
 - Hanya jika **Firebase.ready()**

Langkah 5: Istirahat

- `delay(500);`: Jeda 0.5 detik agar tidak mengirim data terlalu sering
- Setelah itu, loop dimulai kembali dari awal

PENJELASAN KODE DATA.JSON

```
{  
  "fire_detection": {  
    "alarm_active": false,  
    "fuzzy_danger_level": "0.00",  
    "fuzzy_status": "AMAN",  
    "gas": 718,  
    "humidity": "71.0",  
    "ir_detected_fire": false,  
    "temperature": "30.2",  
    "timestamp": "2025-06-25 22:25:25"  
  },  
  "water_pump_control": {  
    "state": "AUTO",  
    "current_status": "AUTO (OFF)"  
  }  
}
```

Ini adalah representasi data dari Firebase Realtime Database. Struktur ini terbagi menjadi dua bagian utama (disebut "node" atau "simpul"), yaitu **fire_detection** dan **water_pump_control**

Bagian 1: fire_detection

Node ini sepenuhnya dihasilkan oleh kode ESP32 yang ada di dalam fungsi loop(), khususnya pada bagian FirebaseJson json;. Setiap "key" di sini sesuai dengan perintah json.set("key", value);

Bagian 2: water_pump_control

Node ini dihasilkan oleh kode ESP32 dan Kode tersebut juga bisa mengirim data ke fire_detection. Dengan cara dikelola oleh aplikasi antarmuka seperti aplikasi HP untuk memberikan kontrol manual / otomatis.

PENJELASAN KODE **MAIN.DART**

```
void main() async {
    WidgetsFlutterBinding.ensureInitialized();
    await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);
    await flutterLocalNotificationsPlugin.initialize(...); // Konfigurasi notifikasi
    runApp(const MyApp());
}
```

1. main.dart

Fungsi:

- Inisialisasi Firebase
- Konfigurasi notifikasi
- Menjalankan aplikasi utama

```
class MyApp extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            theme: ThemeData(fontFamily: 'Poppins', ...),
            home: const SplashScreen(),
        );
    }
}
< 1/1 > Accept Tab Accept Word Ctrl + RightArrow ...
```

2. MyApp

Fungsi:

- Menentukan tema dan halaman awal (**SplashScreen**)

```
Future.delayed(Duration(seconds: 5), () {
    Navigator.pushReplacement(context,
        MaterialPageRoute(builder: (_) => MainScreen()));
});
```

3. SplashScreen

Fungsi:

- Tampilkan animasi awal (**Lottie.asset(...)**)
- Setelah 5 detik, lanjut ke **MainScreen**

PENJELASAN KODE MAIN.DART

```
factory FireDetectionData.fromMap(Map<String, dynamic> map)
| => FireDetectionData(
  temperature: double.tryParse(map['temperature']) ?? 0.0,
  fuzzyStatus: map['fuzzy_status'] ?? 'AMAN',
  ...
);
```

4. FireDetectionData Model

Fungsi:

- Mengubah data dari Firebase menjadi objek Dart

```
class _MainScreenState extends State<MainScreen> {
  FireDetectionData? _latestData;
  List<FireDetectionData> _historicalData = [];
  WaterPumpControl? _waterPumpControl;
  int _selectedIndex = 0;

  @override
  void initState() {
    super.initState();
    _listenToFirebase();
  }
}
```

5. MainScreen

Fungsi:

- Mengatur state utama dan navigasi antar halaman

PENJELASAN KODE MAIN.DART

```
_databaseRef.onValue.listen((event) {
  final data = FireDetectionData.fromMap(event.snapshot.value);
  setState(() {
    _latestData = data;
    _historicalData.add(data);
    ...
  });
  _checkAndShowNotification(data.fuzzyStatus);
});
```

```
BottomNavigationBar(
  currentIndex: _selectedIndex,
  onTap: (index) => setState(
    () => _selectedIndex = index),
)
```

6. Koneksi Firebase

Fungsi:

- Mendengarkan perubahan realtime dari node **/fire_detection** dan **/water_pump_control**

7. Navigasi

Fungsi:

- Pindah antar halaman bawah (dasbor, grafik, kontrol pompa, info risiko)

PENJELASAN KODE MAIN.DART

```
Future<void> _updateWaterPumpState(String newState) async {
  await _pumpControlRef.set({
    "state": newState,
    "current_status": newState == "ON" ? true : false
  });
}
```

```
void _checkAndShowNotification(String fuzzyStatus) {
  if (fuzzyStatus == "WASPADA" || fuzzyStatus == "BAHAYA") {
    _showNotification(fuzzyStatus);
  }
}
void _showNotification(String status) {
  flutterLocalNotificationsPlugin.show(
    0,
    "Peringatan Kebakaran",
    "Status: $status",
    NotificationDetails(
      android: AndroidNotificationDetails(
        'channel_id', 'FireAlert',
        sound: RawResourceAndroidNotificationSound('alarm_kebakaran'),
        ...
      )
    );
}
```

8. Kontrol Pompa

Fungsi:

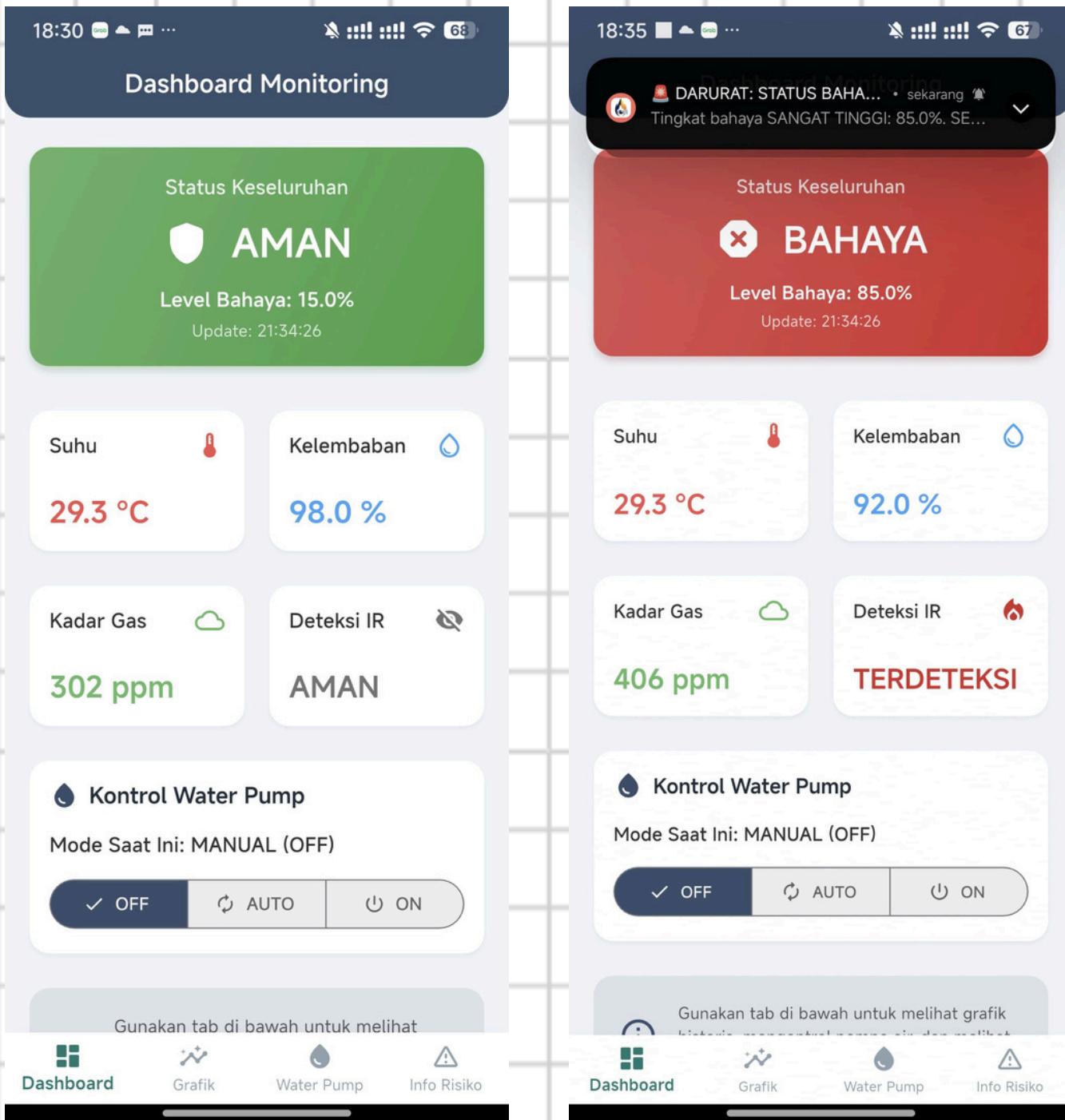
- Kirim perubahan kontrol pompa ke Firebase

9. Notifikasi

Fungsi:

- Kirim notifikasi lokal saat status = "**WASPADA**" atau "**BAHAYA**"

APLIKASI FIRE MONITOR



◆ Enhanced Dashboard Page

- Menampilkan data sensor secara real-time
- Menampilkan status Aman/Waspada/Bahaya
- Kontrol cepat pompa

UI APLIKASI FIRE MONITOR



◆ Enhanced Charts Page

- Menampilkan grafik data historis dari masing masing sensor (line, bar, pie chart)
- Fitur AI Prediksi berdasarkan status dominan
- Tampilan masuk 20 data terakhir

UI APLIKASI FIRE MONITOR



◆ EnhancedRiskInfoPage

- Informasi tentang makna status fuzzy dan saran tindakan

HASIL

◆ Responsivitas Sensor

Sensor MQ-2 menunjukkan respons lebih cepat dibanding DHT11, mencapai ambang batas terlebih dahulu dalam 92% percobaan. Kombinasi keduanya mampu mendeteksi dengan latensi di bawah 6 detik, termasuk untuk api membara.

◆ Durasi Pengiriman Notifikasi

Notifikasi dari Firebase tiba di aplikasi rata-rata dalam $1,6 \pm 0,2$ detik pada Wi-Fi 20 Mbps. Total waktu dari deteksi sensor hingga notifikasi ke pengguna rata-rata 4,3 detik.

◆ Kinerja Respons Pompa

Pompa yang dikendalikan oleh relay dapat aktif dalam waktu kurang dari 0,8 detik setelah deteksi. Pada skenario dengan api (S2 dan S3), semprotan air mampu memadamkan api sepenuhnya dalam waktu kurang dari 15 detik.

◆ Durasi Pengiriman Notifikasi

Aplikasi seluler yang dikembangkan berhasil menampilkan data suhu, kelembaban, kadar gas, dan level bahaya secara real-time.

KESIMPULAN

Sistem ini berhasil mengembangkan sistem deteksi dan pemadam kebakaran dalam ruangan berbasis IoT yang efisien. Sistem ini dapat mendeteksi tanda-tanda awal kebakaran dengan cepat dan mengaktifkan pompa air untuk pemadaman dalam hitungan detik.

Integrasi logika fuzzy dan pemantauan seluler membuat sistem ini lebih cerdas dalam mendeteksi kebakaran dan memungkinkan pemantauan real-time yang fleksibel bagi pengguna. Fitur ini meningkatkan peluang intervensi dini dan meminimalkan alarm palsu, sehingga sistem lebih adaptif terhadap lingkungan nyata yang dinamis. Pendekatan IoT dalam manajemen bahaya kebakaran ini terbukti menjanjikan untuk mengurangi waktu respons dan meningkatkan keselamatan di dalam ruangan.





**THANKS FOR
LISTENING!**