BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Penelitian yang disajikan ini berlandaskan pada beberapa studi terdahulu yang relevan, khususnya dalam konteks pengembangan sistem pemantauan berbasis Internet of Things (IoT). Referensi utama mencakup pemanfaatan Arduino IDE sebagai platform untuk pemrograman mikrokontroler, serta integrasi dengan *Firebase* sebagai basis data yang beroperasi secara *real-time*.

Sebagai contoh, riset Akalily Mardhiyya dan rekan-rekan pada tahun 2024 berjudul "Pengembangan Sistem Deteksi Kebocoran Tabung Gas LPG Berbasis ESP8266" sangat relevan dengan bahasan ini. Studi tersebut mengimplementasikan sensor MQ-2 yang terhubung dengan ESP8266, serta memanfaatkan notifikasi *real-time* untuk memberikan peringatan dini akan adanya kebocoran gas. Hasil penelitian ini mengindikasikan efektivitas sistem berbasis IoT dalam mendeteksi kebocoran gas secara cepat dan akurat[3].

Studi lain yang patut diperhatikan adalah dari Ade Novit Syahputra dan P. Simanjuntak pada tahun 2025, yang berfokus pada pengembangan sistem deteksi kebocoran gas berbasis IoT menggunakan kombinasi Arduino dan sensor MQ-2. Sistem yang mereka kembangkan mampu mengidentifikasi keberadaan gas berbahaya dan menyampaikan peringatan melalui notifikasi. Hal ini menunjukkan bahwa paduan antara komponen perangkat keras dan perangkat lunak berpotensi meningkatkan tingkat keselamatan bagi pengguna[4].

2.2 Landasan Teori

Landasan teori merupakan dasar konseptual yang digunakan dalam pengembangan Sistem Monitoring *Smart Gas Leak Detector* Berbasis Aplikasi Android. Berikut adalah teori-teori yang mendasari sistem ini:

2.2.1 Arduino Ide

Lingkungan Pengembangan Terpadu (IDE) Arduino berfungsi sebagai perangkat lunak utama untuk menyusun sketsa program. Aplikasi ini memungkinkan pengguna dalam menulis, memodifikasi, mengunggah kode ke papan mikrokontroler spesifik, serta melakukan pengembangan program. Diciptakan dengan basis bahasa pemrograman Java, Arduino IDE diperkaya dengan pustaka C/C++ (dikenal sebagai Wiring) yang dirancang untuk menyederhanakan proses operasi input dan output[5].

Gambar 2. 1 Arduino Ide

2.2.2 Android Studio

Android Studio, sebagai sarang kreativitas para pengembang aplikasi, merupakan Lingkungan Pengembangan Terpadu (IDE) berteknologi canggih, digagas oleh para ahli dari Google. IDE ini memberikan banyak macam fitur yang sangat berguna untuk pengembang. Salah satunya adalah sistem build berbasis *Gradle* yang fleksibel, memungkinkan para pengembang untuk mengelola dan menyusun proyek aplikasi dengan efisien[6].

Gambar 2. 2 Android Studio

2.2.3 Firebase Realtime Database

Firebase Realtime Database adalah layanan *cloud-hosted NoSQL* dari Google yang memungkinkan penyimpanan dan sinkronisasi data secara *real-time* antar pengguna. Dalam konteks sistem monitoring, *Firebase* digunakan sebagai media penyimpanan data dari sensor gas yang dikirim oleh mikrokontroler, lalu ditampilkan secara langsung di aplikasi Android[7].

Gambar 2. 3 Google *Firebase*

2.2.4 Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging adalah layanan dari *Firebase* yang memungkinkan pengiriman pesan atau notifikasi *push* ke perangkat Android. Pada sistem monitoring kebocoran gas, FCM berfungsi untuk mengirimkan peringatan atau notifikasi saat status gas terdeteksi dalam kondisi "bahaya”[8].

Gambar 2. 4 Firebase Cloud Messaging

2.2.5 Internet Of Things (IOT)

Internet of Things merupakan sebuah konsep yang di mana perangkat-perangkat fisik terhubung ke internet dan saling bertukar data. Dalam sistem ini, sensor gas (seperti MQ-2) terhubung dengan mikrokontroler (ESP8266/ESP32), yang kemudian mengirimkan data ke *Firebase* untuk dipantau melalui aplikasi Android[9].

Gambar 2. 5 Internet of Thing

2.2.6 User Interface (UI) dan User Experience (UX)

User Interface merupakan sebuah tampilan visual dari sebuah aplikasi, sedangkan *User Experience* (UX) adalah keseluruhan pengalaman pengguna dalam menggunakan aplikasi tersebut. Dalam perancangan aplikasi monitoring gas, UI dan UX menjadi penting untuk memastikan bahwa informasi kondisi gas dapat diakses dan dipahami dengan mudah oleh pengguna[10].

Gambar 2. 6 User Interface

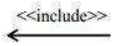
2.2.7 UML (*Unifield Modeling Language*)

Unfield Modeling Language Merupakan Bahasa permodelan standar yang di gunakan untuk mendesain dan mendokumentasikan sistem perangkat lunak. UML memvisualisasikan struktur dan perilaku sistem melalui berbagai jenis diagram dengan UML, pengembang, analis, dan pemangku kepentingan dapat memahami alur sistem[11]. Terdapat beberapa uml yang di gunakan di antaranya:

1. Use Case Diagram

Use Case Diagram menyajikan representasi visual dari fungsionalitas yang diharapkan dari suatu sistem, sekaligus menggambarkan interaksi antara aktor dan sistem tersebut. Di dalam konteks *real*, aktor didefinisikan sebagai entitas, baik itu individu maupun sistem lain, yang menjalankan peran atau tugas dalam sistem.

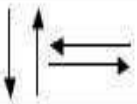
Tabel 2. 1 Use Case Diagram

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	*Generalisasi* : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

2. Activity Diagram

Activity Diagram merupakan gambaran alur aktivitas atau alur kerja dalam suatu sistem yang sedang beroperasi. Diagram ini juga berfungsi untuk mengorganisir atau menentukan urutan tampilan dari sistem terkait. Di dalamnya, terdapat berbagai komponen grafis yang saling terhubung melalui tanda panah. Panah-panah tersebut mengarahkan pada urutan langkah-langkah yang dieksekusi, mulai dari titik awal hingga penyelesaian.

Tabel 2. 2 Activity Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan diakhiri
5		*Decision*	Diguanakan untuk menggambarkan suatu keputusan / tindakan yang harus diambil pada kondisi tertentu
6		*Line Connector*	Digunakan untuk menghubungkan satu simbol dengan simbol lainnya