BAB II

TINJAUAN PUSTAKA

2.1 Teori Terkait

Penelitian ini merupakan pengembangan dari beberapa penelitian sebelumnya seperti pada penelitian[7] telah membahas topik serupa, namun masih memiliki beberapa kelemahan. Salah satu kelemahan adalah belum adanya aplikasi yang memudahkan masyarakat untuk *memonitoring* ketinggian debit air sungai. penggunaan *website* menurut penulis juga kurang efektif karena masih harus dikenakan biaya bulanan untuk *hosting* domain yang digunakan untuk sistem.

Pada penelitian[8] tentang Rancang Bangun Sistem *Monitoring* Pengukuran Volume Air Berbasis IoT Menggunakan *Arduino Wemos* ini sudah berhasil mengukur ketinggian air pada tangki, namun masih memiliki kekurangan yaitu masih belum adanya aplikasi yang digunakan untuk memudahkan proses *monitoring* ketinggian airnya.

Pada penelitian[9] tentang Rancang Bangun *Prototype* Sistem *Monitoring* Ketinggian Air Sungai Berbasis *Microcontroler Arduino* sudah berjalan dengan baik, namun masih belum menambahkan *Rain Drop Sensor* mengingat hujan menjadi salah satu pengaruh signifikan terhadap kenaikan air sungai.

2.2 Landasan Teoretis

2.2.1 ESP8266

ESP8266 adalah sebuah *microcontroler Wi-Fi* berbiaya rendah yang dilengkapi dengan perangkat lunak jaringan TCP/IP terintegrasi, diproduksi oleh *Espressif Systems* di Shanghai, China. *Chip* ini populer di komunitas pembuat (*maker community*) berbahasa Inggris pada Agustus 2014 melalui modul ESP-01, yang dibuat oleh pabrikan pihak ketiga *Ai-Thinker*. Modul kecil ini memungkinkan *microcontroler* untuk terhubung ke jaringan *Wi-Fi* dan membuat koneksi TCP/IP sederhana menggunakan perintah ala Hayes[9].

ESP8266 dapat diprogram langsung menggunakan *Arduino* IDE setelah *menginstal add-on* ESP8266[10]. Ini memungkinkan pengembangan berbagai proyek IoT dan otomasi rumah dengan biaya yang terjangkau.

Gambar 2.1 ESP8266

2.2.2 Sensor Ultrasonik

Sensor ultrasonik adalah alat yang menggunakan gelombang suara dengan frekuensi lebih dari 20.000 Hz untuk mengukur jarak dan mendeteksi objek di depannya. Sensor ini terdiri dari unit pemancar dan unit penerima yang bekerja bersama untuk

menghasilkan informasi. Sensor ultrasonik memiliki banyak aplikasi praktis dalam bidang industri, medis, otomotif, dan lainnya.

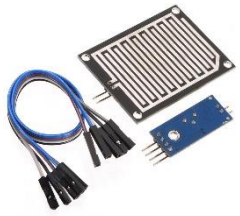
Prinsip kerja sensor ultrasonik adalah memanfaatkan pantulan gelombang suara yang dipancarkan oleh unit pemancar dan diterima oleh unit penerima. Dengan menghitung waktu yang dibutuhkan gelombang suara untuk mencapai objek dan kembali ke sensor, jarak objek dapat dihitung. Selain itu, sensor ultrasonik juga dapat mendeteksi jenis benda yang mampu memantulkan sinyal, seperti logam, plastik, atau kaca. Sensor ultrasonik beroperasi dengan rentang frekuensi antara 40 KHz hingga 400 KHz, yang berada di atas rentang pendengaran manusia[11].

Gambar 2.2 Sensor Ultrasonik

2.2.3 Rain Drop Sensor

Rain Drop Sensor adalah perangkat yang digunakan untuk mendeteksi hujan. Perangkat ini terdiri dari dua modul: papan pendeteksi hujan dan modul kontrol. Papan pendeteksi hujan berfungsi untuk mendeteksi keberadaan air hujan, sedangkan modul kontrol membandingkan nilai analog dan mengonversinya menjadi nilai digital.

Prinsip kerja sensor hujan ini berdasarkan resistansi. Ketika tidak ada tetesan hujan di papan sensor, resistansinya tinggi, sehingga tegangan yang dihasilkan juga tinggi (sesuai dengan hukum Ohm: V = IR). Namun, ketika ada tetesan hujan, resistansi berkurang karena air merupakan penghantar listrik. Kehadiran air menghubungkan jalur nikel secara paralel, sehingga resistansi berkurang dan tegangan turun. Sensor ini sangat berguna untuk mengotomatisasi berbagai sistem berdasarkan kondisi hujan.



Gambar 2.3 Rain Drop Sensor

2.2.4 FireBase

Firebase adalah suatu layanan dari Google yang digunakan untuk mempermudah para pengembang aplikasi dalam mengembangkan aplikasi. Dengan adanya Firebase, pengembang aplikasi bisa fokus mengembangkan aplikasi tanpa harus memberikan usaha yang besar. Dua fitur yang menarik dari Firebase yaitu Firebase *Remote Config* dan *Firebase Realtime Database*. Selain itu terdapat fitur pendukung untuk aplikasi yang membutuhkan pemberitahuan yaitu *Firebase Notification*[12]. Firebase menawarkan fitur seperti

database real-time, autentikasi pengguna, analitik, pengujian A/B, dan kampanye pesan untuk meningkatkan keterlibatan pengguna.

Firebase sangat berguna untuk pengembang yang ingin mempercepat proses pengembangan dan fokus pada pengalaman pengguna daripada manajemen infrastruktur. Dengan integrasi yang mendalam dengan layanan Google lainnya, Firebase memungkinkan pengembang untuk memanfaatkan kekuatan *cloud* dengan cara yang mudah dan efisien.

Gambar 2.4 Firebase

2.2.5 MiT App Inventor

MIT App Inventor adalah platform pengembangan aplikasi yang intuitif dan berbasis visual yang memungkinkan pengguna dari segala usia dan latar belakang untuk menciptakan aplikasi seluler untuk sistem operasi Android tanpa perlu menulis kode dalam bahasa pemrograman tradisional[13]. Dikembangkan oleh MIT *(Massachusetts Institute of Technology)*, App Inventor menggunakan pendekatan *drag-and-drop* untuk merancang antarmuka pengguna dan logika aplikasi, membuat proses pembuatan aplikasi menjadi lebih mudah dan dapat diakses oleh pemula sekalipun.

App Inventor sangat cocok untuk proyek-proyek pendidikan, *prototyping* cepat, dan bagi siapa saja yang ingin memulai belajar

tentang pengembangan aplikasi seluler. Platform ini juga sering digunakan dalam konteks pendidikan STEM *(Science, Technology, Engineering, and Mathematics)* untuk mengajarkan konsep-konsep pemrograman dan pemikiran komputasional[13].

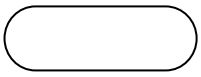
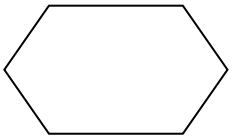
Gambar 2.5 MiT App Inventor

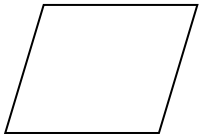
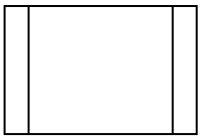
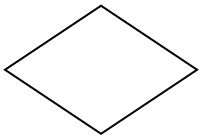
2.2.6 Flowchart

Flowchart adalah bagan alir yang menggambarkan tentang urutan langkah jalannya suatu program dalam sebuah bagan dengan simbol-simbol bagan yang sudah ditentukan.

Adapun simbol-simbol flowchart program adalah sebagai berikut :

Tabel 2.1 Keterangan simbol *flowchart*

Simbol	Keterangan
	**Terminator / Terminal** Merupakan simbol yang digunakan untuk menentukan state awal dan state akhir suatu flowchart program.
	**Preparation / Persiapan** Merupakan simbol yang digunakan untuk mengidentifikasi variabel-variabel yang akan digunakan dalam program. Bisa berupa pemberian harga awal, yang ditandai dengan nama variabel sama dengan (‘’) untuk tipe string, (0) untuk tipe numeric, (.F./.T.) untuk tipe Boolean dan ({//}) untuk tiper tanggal.

Simbol	Keterangan
	**Input output / Masukan keluaran** Merupakan simbol yang digunakan untuk memasukkan nilai dan untuk menampilkan nilai dari suatu variabel. Ciri dari simbol ini adalah tidak ada operator baik operator aritmatika hingga operator perbandingan. Yang membedakan antara masukan dan keluaran adalah jika Masukan cirinya adalah variabel yang ada didalamnya belum mendapatkan operasi dari operator tertentu, apakah pemberian nilai tertentu atau penambahan nilai tertentu. Adapun ciri untuk keluaran adalah biasanya variabelnya sudah pernah dilakukan pemberian nilai atau sudah dilakukan operasi dengan menggunakan operator terntentu.
	**Process / Proses** Merupakan simbol yang digunakan untuk memberikan nilai tertentu, apakah berupa rumus, perhitungna counter atau hanya pemrian nilai tertentu terhadap suatu variabell.
	**Predefined Process / Proses Terdefinisi** Merupakan simbol yang penggunaannya seperti link atau menu. Jadi proses yang ada di dalam simbol ini harus di buatkan penjelasan flowchart programnya secara tersendiri yang terdiri dari terminator dan diakhiri dengan terminator.
	**Decision / simbol Keputusan** Digunakan untuk menentukan pilihan suatu kondisi (Ya atau tidak). Ciri simbol ini dibandingkan dengan simbol-simbol flowchart program yang lain adalah simbol keputusan ini minimal keluaran arusnya 2 (dua), jadi Jika hanya satu keluaran maka penulisan simbol ini adalah salah, jadi diberikan pilihan jika kondisi bernilai benar (true) atau salah (false). Sehingga jika nanti keluaran dari simbol ini adalah lebih dari dua bisa dituliskan. Khusus untuk yang keluarannya dua, harus diberikan keterangan Ya dan Tidaknya pada arus yang keluar.

Simbol	Keterangan
	**Connector** Konektor dalam satu halaman merupakan penghubung dari simbol yang satu ke simbol yang lain. Tampa harus menuliskan arus yang panjang. Sehingga akan lebih menyederhanakan dalam penggambaran aliran programnya, simbol konektornya adalah lingkaran, sedangkan Konektor untuk menghubungkan antara simbol yang satu dengan simbol yang lainnya yang berbeda halaman, maka menggunakan simbol konektor yang segi lima, dengan deberikan identitasnya, bisa berupa charater alpabet A – Z atau a – z atau angka 1 sampai dengan 9.
	Arrow / Arus Merupakan simbol yang digunakan untuk menentukan aliran dari sebuah flowchart program. Karena berupa arus, maka dalam menggambarkan arus data harus diberi simbol panah.

2.2.7 *Unified Modeling Language* (UML)

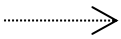
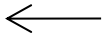
UML (*Unified Modeling Language*) adalah metode pemodelan secara visual sebagai sarana untuk merancang dan atau membuat software berorientasi objek. Karena UML ini merupakan bahasa visual untuk pemodelan bahasa berorientasi objek, maka semua elemen dan diagram berbasiskan pada paradigma object oriented. UML sendiri juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas - kelas dalam bahasa program yang spesifik.

Beberapa diagram yang digunakan di UML (Unifed Modeling Language) :

1. Use case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah use case mempresentasikan sebuah interaksi antara aktor dengan sistem. Use case merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-create sebuah daftar belanja, dan sebagainya. Seorang atau sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan - pekerjaan tertentu.

Tabel 2.2 Keterangan simbol *Use Case* Diagram

No	Gambar	Nama	Keterangan
1.		Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2.		Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3.		Generalizatio n	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada diatasnya objek induk (*ancertor*).
4.		Include	Menspesifikasikan bahwa use case sumber secara eksplisit.

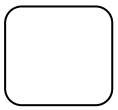
No	Gambar	Nama	Keterangan
5.		Extend	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6.		Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7.		System	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8.		Use Case	Deskripsi dari urutan aksi – aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
9.		Collaboration	Interaksi aturan – aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen – elemennya (sinergi).
10.		Note	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

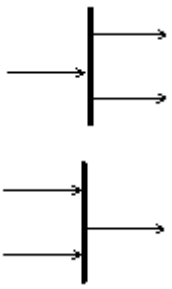
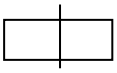
2. Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing - masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Activity diagram merupakan state diagram khusus, dimana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing). Oleh karena itu

activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses - proses dan jalur - jalur aktivitas dari level atas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktifitas. Decision digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses - proses paralel (fork dan join) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

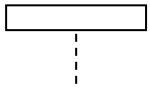
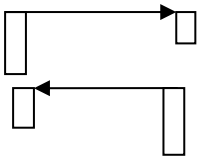
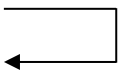
Tabel 2.3 Keterangan simbol *activity diagram*

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 suatu aksi.
3.		Initial Node	Bagaimana objek dibentuk atau diawali.
4.		Final Node	Bagaimana objek dibentuk dan dihancurkan.
5.		Fork Node	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.
6.		Decision	Pilihan untuk mengambil keputusan

No	Gambar	Nama	Keterangan
7.		Fork/Join	Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu
8.		Rake	Menunjukkan adanya dekomposisi
9.		Time	Tanda waktu
10.		Send	Tanda pengiriman

1. Sequence Diagram

Sequence diagram menggambarkan interaksi antar di sekitar (pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek - objek yang terkait). Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah - langkah yang dilakukan sebagai respon dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang men-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan. Masing - masing objek, termasuk aktor, memiliki lifeline vertikal.Keterangan simbol *Sequence* Diagram

No	Gambar	Nama	Keterangan
1.		LifeLine	Objek *entity*, antar muka yang saling berinteraksi.
2.		Message	Spesifikasi dari komunikasi antar objek yang memuat informasi – informasi tentang aktifitas yang terjadi.
3		Actor	Menggambarkan orang yang sedang berinteraksi dengan sistem
4		Boundary Class	Menggambarkan penggambaran dari form
5		Entity Class	Mengambarkan hubungan kegiatan yang akan dilakukan
6.		Control Class	Menggambarkan penghubung antara Boundary dengan tabel
7		Activation	Sebagai sebuah objek yang akan melakukan sebuah aksi
8		Message	Mengindikasikan komunikasi antara objek dengan objek
9		Self Message	Menginndikasikan komunikasi kembali kedalam sebuah objek itu sendiri