BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Berdasarkan topik yang diangkat, terdapat beberapa referensi dari penelitian yang telah dilakukan oleh pihak sebelumnya.;

Pada penelitian yang dilakukan oleh M. Wildan Firdaus, M. Ary Murti dan kawan kawan dalam jurnal penelitianya yang berjudul SISTEM KONTROL DAN MONITORING GENSET MELALUI INTERNET. Dijelaskan bahwa sistem kontrol terdiri atas sekumpulan piranti-piranti dan peralatan elektronik yang mampu mengontrol dan keluaran sesuai yang diinginkan. Genset merupakan alat yang dapat merubah energi mekanik menjadi energi listrik dengan berbahan bakar solar. Umumnya sistem *monitoring* genset dilakukan secara manual, sehingga sangat tidak efektif karena operator harus melihat parameter yang ada pada suatu sistem dari dekat dan skala waktu pembacaan yang tidak tetap. Masalah di atas menjadi acuan dalam membuat sistem kontrol dan *monitoring* genset melalui internet. Sistem ini bekerja dengan mendeteksi ketinggian bahan bakar solar dan keadaan genset menyala atau mati. Bila genset menyala atau mati maka sistem akan menampilkan ketinggian bahan bakar solar secara terus menerus pada komputer [3].

Pada Penelitian yang dilakukan oleh Alamsyah, Ardi Amir dan Muhamad Nur Faisal. Dalam jurnal penelitianya yang berjudul

PERANCANGAN DAN PENERAPAN SISTEM KONTROL PERALATAN ELEKTRONIK JARAK JAUH BERBASIS WEB. Dijelaskan bahwa, Desain dan penerpan sistem kontrol peralatan elektronik jarak jauh berbassis web. Penelitian ini dirancang dan dibangun sebuah mini plant untuk memodelkan bangunan dan rumah yang terdiri dari peralatan rumah seperti kipas,AC, TV maupun lampu listrik. Sistem pengendalian berbasis web ini dirancang secara nirkabel (*Wireless*) dengan memanfaatkan teknologi internet. Data keluaran device (peralatan elektronik) di kendalikan oleh driver relay yang selanjutnya dikirim melalui komunikasi paralel (*port paralel*) yang dihubungkan ke komputer *server* sehingga dapat ditampilkan ke halaman *web* user. Pada pengujian perangkat keras diperoleh hasil pengukuran rangkaian *driver relay* pada saat port paralel baik pada logika 0 atau 1 adalah tegangan *Vcc* sebesar 11,93 volt. Untuk pengujian rangkaian *driver relay* didapatkan hasilnya bahwa pada saat saat transistor aktif (saturasi) nilai tegangan *relay* sebesar 11,61 *volt*, sedangkan pada saat saat transistor tidak aktif (*cut off*) nilai tegangan *relay* sebesar 0 *volt* [4].

Pada penelitian yang dilakukan oleh Zulkarnain Lubis, Adi Saputra, dan Kawan-kawan. Dalam jurnal penelitianya yang berjudul KONTROL MESIN AIR OTOMATIS BERBASIS ARDUINO DENGAN SMARTPHONE. Dijelaskan bahwa sistem pengontrolan saat ini yang masih memiliki banyak keterbatasan dalam melakukan kontrol terhadap mesin pompa air yang berada di rumah atau perusahaan yang menggunakan. Beragam keterbatasan yang ada menimbulkan banyak kekhawatiran dari tiap

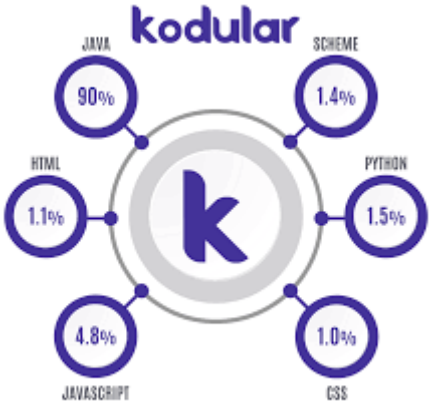
pemilik mesin air. Seperti disaat terlalu banyak pesan yang tersimpan pada SIM yang terdapat pada modul SMS *GATEWAY* maka modul SMS *GATEWAY* tidak akan bisa mengirimkan pesan kepada pemilik maka dengan itu pemilik harus terlebih dahulu untuk menghapus atau mengosongkan pesan yang masuk pada modul SMS *GATEWAY*, karena pesan yang tersimpan pada modul akan membuat modul SMS *GATEWAY* mengalami gangguan dalam mengirim pesan bak kosong atau penuh pada pemilik mesin air. Pada penelitian ini, dibuat dengan tujuan untuk memudahkan pemilik mesin air mengontrol mesin pompa dari kekosongan dan kepenuhan bak penampungan [5].

2.2 Landasan Teori

2.2.1. Kodullar

Kodular adalah salah suatu aplikasi atau *tools* IDE *open source* seperti *App Inventor*. Kodular ini memiliki fitur-fitur *widget* yang paling banyak dari *tools* IDE sejenisnya. Kodular adalah situs *web* yang menyediakan alat untuk membangun aplikasi Android dengan konsep pemrograman *drag and drop block*. Pemrograman blok adalah fitur inti dari kodular, dengan fitur ini tidak perlu lagi memasukkan kode program secara manual untuk membuat aplikasi Android. Kodular juga menyediakan *Base mini* dan fungsi penyimpanan sehingga kita dapat menyimpan dan mengunduh data sesuai keinginan. Dari segi antarmuka/GUI, kode dapat disesuaikan dengan

tema untuk membuat aplikasi yang kita buat lebih *modern* dan profesional [6].



Gambar 2. 1. Logo Kodullar

2.2.2. Android

Android adalah sebuah sistem operasi perangkat *mobile* berbasis linux yang mencangkup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google *Inc*. membeli Android *Inc*. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel atau *smartphone*. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, *konsorsium* dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-*Mobile*, dan Nvidia. Pada saat perilisan perdana Android, 5

November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan *open source* pada perangkat mobile. Di lain pihak, Google merilis kode-kode Android di bawah

lisensi *Apache*, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler [7].

Gambar 2. 2. Logo Android

2.2.3. Bluetooth

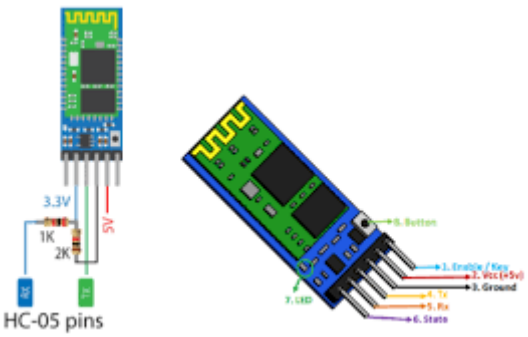
Bluetooth adalah spesifikasi industri untuk jaringan kawasan pribadi (personal area *networks* atau PAN) tanpa kabel. *Bluetooth* menghubungkan dan dapat dipakai untuk melakukan tukar-menukar informasi di antara peralatan-peralatan. Spesifiksi dari peralatan *Bluetooth* ini dikembangkan dan didistribusikan oleh kelompok *Bluetooth Special Interest Group*. *Bluetooth* beroperasi dalam pita *frekuensi* 2,4 *Ghz* dengan menggunakan sebuah *frequency hopping traceiver* yang mampu menyediakan layanan komunikasi data dan suara secara *real time* antara *hosts bluetooth* dengan jarak terbatas. Kelemahan teknologi ini adalah jangkauannya yang pendek dan kemampuan transfer data yang rendah. Teknologi *Bluetooth* menghubungkan begitu banyak perangkat digital, teknologi ini sangat

memudahkan dalam melakukan aktifitas sehari-hari. *Bluetooth* membantu *user* mendengarkan musik, berbicara di telepon, dan bermain *video game*, semua menjadi lebih nyaman tanpa adanya kabel yang semrawut.

Gambar 2. 3. Logo Bluetooth

2.2.4. Bluetooth HC-5

Modul Bluetooth HC-05 dengan *supply* tegangan sebesar 3,3 V ke pin 12 modul *Bluetooth* sebagai VCC. Pin 1 pada modul *Bluetooth* sebagai *transmitter*. Kemudian pin 2 pada modul *Bluetooth* sebagai *receiver*. Tabel 6 berikut ini menunjukkan jumlah pin pada modul *Bluetooth HC-05* beserta tipe dan kegunaannya [8].



Gambar 2. 4. Modul Bluetooth HC-5

2.2.5. Aplikasi

Menurut Pamama, 2012 aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti sistem perniagaan, game, pelayanan masyarakat, peiklanan dan hampir semua proses kegiatan.

Yang dimaksud perangkat lunak aplikasi satu unit perangkat lunak aplikasi adalah suatu sub kelas perangkat lunak komputer yang memanfaatkan kemampuan langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media [9].

Gambar 2. 5. Aplikasi

2.2.6. Unifield 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.

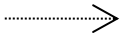
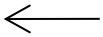
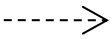
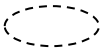
Beberapa diagram yang digunakan di UML (Unifed Modeling Language) :

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. 1 Use Case Diagram

No	Gambar	Nama	Keterangan
1.		Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.

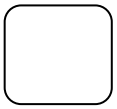
No	Gambar	Nama	Keterangan
2.		Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3.		Generalization	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur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

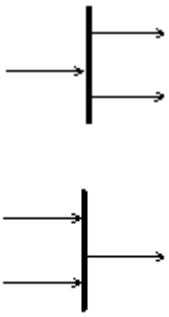
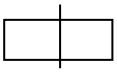
2. *Diagram Aktivitas (Activity Diagram)*

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. 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 suatu aksi.

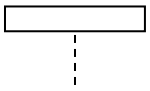
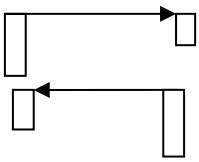
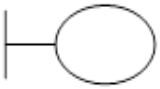
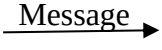
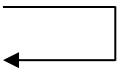
No	Gambar	Nama	Keterangan
3.		Initial Node	Bagaimana objek dibentuk atau diawali.
4.		Final Node	Bagaimana objek dibentuk dan dihancurkan.
5.		Fork Node	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.
6.		Decision	Pilihan untuk mengambil keputusan
7		Fork/Join	Digunakan untuk menunjukkan 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

3. *Diagram Urutan* (*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:

Tabel 2. 3 Sequence Diagram

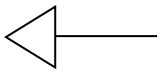
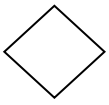
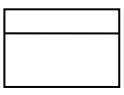
No	Gambar	Nama	Keterangan
1.		LifeLine	Objek *entity*, antar muka yang saling berinteraksi.
2.		Message	Spesifikasi dari komunikasi antar objek yang memuat informasi – informasi tentang aktifitas yang terjadi.
3		Actor	Menggambarkan orang yang sedang berinteraksi dengan sistem
4		Boundary Class	Menggambarkan 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

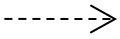
4. *Diagram* Kelas (*Class Diagram)*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class diagram menggambarkan struktur dan deskripsi class, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. Class memiliki tiga area pokok : nama (*stereotipe*), atribut, dan metoda. Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
b. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak anak yang mewarisinya

Tabel 2. 4 Class Diagram

No	Gambar	Nama	Keterangan
1.		Generalization	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*)
2.		Nary Association	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3.		Class	Himpunan dari objek - objek yang berbagi atribut serta operasi yang sama.

No	Gambar	Nama	Keterangan
4.		Collaboration	Deskripsi dari urutan aksi - aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
5.		Dependency	Operasi yang benar - benar dilakukan oleh suatu objek.
6.		Dependency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
7.		Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.

5. *Deployment Diagram*

Deployment *Diagram* digunakan untuk menggambarkan detail bagaimana komponen disusun di infrastruktur sistem.

Tabel 2. 5 Deployment Diagram

Gambar	Keterangan
1	Satu dan hanya satu.
0..*	Boleh tidak ada satu atau 1 atau lebih
1..*	1 atau lebih
0...1	Boleh tidak ada, maksimal 1.
n...n	Batasan antara contoh 2..4 mempunyai arti minimal 2 maksimal 4.