

RANCANG BANGUN SISTEM PEMANTAUAN TANGKI AIR DAN  
KELMABABAN TANAH BERBASIS *INTERNET OF THING* DENGAN  
GOOGLE FIREBASE DAN MIT APP INVENTOR 2



PROPOSAL SKRIPSI

Diajukan untuk memenuhi salah satu syarat kelulusan program Strata Satu (S1)

ARIP JAMALUDIN

2022310095

**Program Studi Teknik Informatika Fakultas Informatika**

**Universitas Bina Insani**

**Bekasi**

**2023**

# **BAB I**

## **PENDAHULUAN**

### **1.1. Latar Belakang Masalah**

Pertumbuhan populasi dan urbanisasi yang semakin pesat menyebabkan peningkatan kebutuhan air bersih. Air adalah sumber daya yang sangat penting bagi kehidupan manusia, dan untuk memenuhi kebutuhan tersebut diperlukan pengelolaan sumber daya air yang efisien dan efektif. Salah satu cara untuk mengoptimalkan penggunaan air adalah dengan menggunakan sistem pemantauan tangki air.

Saat ini, sistem pemantauan tangki air yang umum digunakan masih menggunakan metode manual, yaitu dengan melakukan pengukuran secara langsung dengan menggunakan alat ukur. Metode ini memiliki beberapa kelemahan, seperti waktu yang dibutuhkan untuk melakukan pengukuran secara manual dan ketidakakuratan pengukuran. Oleh karena itu, diperlukan solusi yang lebih efisien dan efektif dalam memonitor penggunaan air.

Masalah utama yang sering dihadapi dalam penggunaan tangki sebagai sarana penampungan air adalah manajemen pengisiannya. Mengisi tangki air bisa saja menjadi hal yang sangat mudah jika air yang digunakan adalah air tanah yang diambil melalui pompa air biasa, dengan menggunakan sistem pelampung dan saklar ketinggian air, pompa dapat menyala dan mati secara otomatis. Namun hal berbeda terjadi jika sumber air untuk mengisi tangki adalah dalam pipa PDAM, sistem pelampung dan saklar ketinggian air tidak dapat berfungsi pada kasus ini diperlukan alat tambahan yang berfungsi sebagai pengontrol aliran air dari pipa PDAM.

Selain itu, kelembaban tanah juga merupakan faktor penting dalam pertanian dan irigasi. Tanah yang terlalu lembab atau terlalu kering dapat mempengaruhi pertumbuhan tanaman dan produksi hasil panen. Oleh karena

itu, pemantauan kelembaban tanah menjadi penting dalam pengelolaan pertanian. Terlebih saat setelah pandemi seperti sekarang ini bertani bukan lagi hal yang asing bagi warga perkotaan, pandemi telah mengubah cara hidup kita semua agar menjadi lebih sehat dan aktif, karena itu lahirlah istilah *urban farming* yang marak ditemui diperkotaan. Pengendalian dan pemanfaatan air bersih yang tepat guna tentu akan sangat membantu meringankan beban biaya PDAM bagi warga kota yang menggunakan tangki air sebagai penampungan dan memiliki hobi bertani di lahan sempit perkotaan.

Internet of Things (IoT) adalah salah satu teknologi yang dapat digunakan untuk memantau penggunaan air dan kelembaban tanah. Dengan menggunakan IoT, sistem pemantauan dapat terhubung dengan internet dan memberikan informasi secara *real-time*, sehingga dapat memantau penggunaan air dan kelembaban tanah secara lebih efisien dan efektif.

Dalam sebuah jurnal penelitian yang diterbitkan pada tahun 2022 oleh **Daulay, Ahmad Maqhribi** dkk. tentang “***Sistem Monitoring Air Pada Tangki Berbasis Internet of Things (IoT) Blynk App***” telah berhasil memonitor volume air yang ada dalam tangki secara *real-time* dan proses penyalan pompa air pun dapat dikendalikan melalui aplikasi berbasis *IoT* tersebut yaitu melalui aplikasi Blynk.

Berdasarkan latar belakang tersebut, penulis tertarik untuk merancang dan membangun sistem pemantauan tangki air dan kelembaban tanah berbasis *IoT*. Sistem ini diharapkan dapat membantu pengguna dalam memantau penggunaan air dan kelembaban tanah secara *real-time*, sehingga dapat mengoptimalkan penggunaan sumber daya air dan mempermudah pengelolaan pertanian.

Penulis ingin membuat sebuah “**Rancang Bangun Sistem Pemantauan Tangki Air Dan Kelembaban Tanah Berbasis Internet Of Thing (IoT) dengan Google Firebase dan Mit App Inventor 2**” agar penggunaan air bersih dapat lebih terkendali juga menghindari terjadinya

kekurangan air bersih saat pasokan air dari PDAM tidak mengalir karena kapasitas air yang ada di dalam tangki penampungan senantiasa dapat dipantau secara *real-time* melalui telepon pintar karena untuk saat ini alat atau sensor yang tersedia dipasaran adalah sensor untuk pengendalian aliran air dari pompa air tanah menuju ke tangki penampungan sedangkan untuk sensor pengendalian aliran air dari pipa PDAM ke tangki penampungan masih belum tersedia dipasaran karena memerlukan alat tambahan yaitu katup solenoid (*solenoid valve*) sebagai pengendalinya.

## **1.2. Identifikasi Masalah**

Berdasarkan pemaparan diatas berkaitan dengan pemantauan dan efisiensi penggunaan sumber daya air maka dapat diidentifikasi beberapa masalah yang saat ini sering dihadapi yaitu:

1. Memantau dan mengontrol penggunaan air dari PDAM masih dilakukan secara manual .
2. Tangki air meluap sehingga banyak air bersih yang terbuang sia-sia karena lupa menutup katup air dari PDAM menuju tangki penampungan.
3. Sering terjadi kekurangan air karena lupa mengisi tangki penampungan.
4. Tanaman banyak yang kering dan mati akibat lupa melakukan penyiraman.

## **1.3. Batasan Masalah**

Agar penelitian yang dilakukan lebih fokus dan tidak keluar dari pokok pembahasan maka penulis membuat batasan-batasan masalah sebagai berikut:

1. Penerapan teknologi *Internet Of Thing* fokus pada beberapa peralatan saja, yaitu: 1. Tangki penampungan air bersih yang bersumber dari pipa PDAM, 2. Pompa DC mini untuk melakukan penyiraman tanaman di dalam pot.
2. Penelitian ini hanya untuk pemantauan dan pengendalian manual melalui telepon pintar dengan system operasi android.

3. Pada penelitian pengendalian buka-tutup katup solenoid , menyalakan dan mematikan pompa mini untuk penyiraman tanaman hanya dilakukan secara manual melalui telepon pintar tidak dibuat secara otomatis Ketika sensor menunjukan angka tertentu..

#### **1.4. Perumusan Masalah**

1. Bagaimana memastikan pasokan air selalu cukup saat pasokan air dari PDAM terganggu ?
2. Bagaimana menghindari pemborosan air karena air pada tangki penampungan meluap ?
3. Bagaimana memantau ketersediaan air dan kelembaban tanah secara real-time ?
4. Bagaimana cara mengefesiensikan penggunaan air untuk kebutuhan sehari-hari dan penyiraman tanaman ?

#### **1.5. Tujuan dan Manfaat Penelitian**

1. Memanfaatkan teknologi *Internet of Thing* untuk meningkatkan efisiensi penggunaan sumber daya air di rumah.
2. Memudahkan pemantauan dan pengendalian penggunaan sumber daya air.

#### **1.6. Sistematika Penulisan**

Secara garis besar skripsi ini terdiri dari 5 (lima) bab dan beberapa lampiran. Adapun setiap bab dan sub-sub bab yang terdiri dari:

#### **BAB I            PENDAHULUAN**

Bab ini akan membahas Latar Belakang Masalah, Identifikasi Masalah, Batasan Masalah, Tujuan dan Manfaat Penelitian, Metode Penelitian, Tinjauan Pustaka, dan Sistematika Penulisan.

#### **BAB II           LANDASAN TEORI**

Pada Bab ini berisikan tinjauan pustaka yang merupakan tentang sumber acuan penulis. Penelitian terkait dengan permasalahan yang diangkat berasal dari jurnal lima tahun terakhir dari tahun penelitian.

#### **BAB III          METODOLOGI PENELITIAN**

Bab ini menjelaskan tentang metode pengumpulan data yang dilakukan, dan model pengembangan yang digunakan oleh penulis serta kerangka pemikiran untuk memecahkan masalah yang sedang diteliti.

#### **BAB IV        HASIL DAN PEMBAHASAN**

Dalam Bab ini berisikan rancangan sistem usulan yang dibuat oleh penulis dan juga berisi penjelasan secara global tentang interface dari sistem tersebut.

#### **BAB V        SIMPULAN DAN SARAN**

Bab ini berisi tentang kesimpulan dan saran yang bermanfaat dari hasil penulisan Skripsi.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1. Sistem Monitoring (*Monitoring System*)**

Sistem Monitoring adalah suatu sistem yang digunakan untuk mengumpulkan, menganalisis, dan memonitor data dalam suatu lingkungan atau sistem. Sistem ini dapat digunakan dalam berbagai bidang, seperti lingkungan, transportasi, kesehatan, keamanan, industri, dan banyak lagi. Tujuan dari Sistem Monitoring adalah untuk memberikan informasi yang akurat dan real-time mengenai kondisi sistem atau lingkungan yang dipantau.

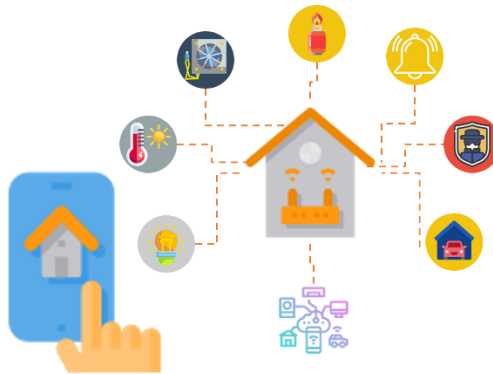
Dengan informasi ini, pengguna dapat membuat keputusan yang lebih baik dalam pengelolaan dan pengendalian sistem atau lingkungan tersebut. Sistem Monitoring dapat mengumpulkan data menggunakan berbagai jenis sensor, seperti sensor suhu, sensor kelembaban, sensor cahaya, sensor tekanan, sensor getaran, dan lain-lain. Data yang dikumpulkan oleh sensor kemudian diproses oleh sistem untuk menghasilkan informasi yang bermanfaat bagi pengguna.

Contoh penerapan Sistem Monitoring pada bidang lingkungan, Sistem Monitoring dapat digunakan untuk memantau kualitas udara, kualitas air, dan kondisi cuaca. Sistem ini akan mengumpulkan data dari sensor-sensor yang dipasang pada berbagai titik di wilayah yang dipantau, kemudian menganalisis data tersebut untuk memberikan informasi mengenai kualitas udara, kualitas air, atau kondisi cuaca pada wilayah tersebut.

Dalam industri, Sistem Monitoring dapat digunakan untuk memantau kondisi mesin dan peralatan. Sistem ini akan mengumpulkan data dari sensor-sensor pada mesin dan peralatan, kemudian menganalisis data tersebut untuk memberikan informasi mengenai performa mesin atau peralatan tersebut. Dengan informasi ini, pengguna dapat memprediksi kemungkinan kegagalan atau masalah pada mesin atau peralatan tersebut dan melakukan tindakan pencegahan sebelum terjadi kerusakan yang lebih serius.



Penelitian ini penulis akan mencoba membuat fungsi pemantauan kapasitas tangki air dan kelembaban tanah secara real-time namun kontrol bagi peralatan tersebut masih dilakukan secara manual melalui telepon pintar.



Sumber: [https:// icons8.com /](https://icons8.com/)

Gambar II.1

### Ilustrasi *Monitoring System*

## 2.2 Internet of Thing

*Internet of things* adalah sebuah teknologi yang memungkinkan kita untuk menghubungkan mesin, peralatan, dan benda fisik lainnya dengan sensor jaringan dan aktuator untuk memperoleh data dan memproses serta mampu mengelola kinerjanya sendiri, sehingga dimungkinkan adanya beberapa mesin untuk saling berkolaborasi. Sebuah publikasi mengenai *Internet of things* menjelaskan bahwa *internet of things* adalah suatu keadaan ketika benda memiliki sebuah identitas, bisa beroperasi secara intelijen, dan bisa berkomunikasi dengan sosial, lingkungan, serta dengan setiap penggunaanya.

Tujuan *Internet of things* adalah untuk membuat manusia berinteraksi dengan benda lebih mudah, bahkan dengan tujuan supaya benda juga bisa saling berkomunikasi antar satu benda dengan benda yang lainnya.



Sumber: [https:// icons8.com /](https://icons8.com/)

Gambar II.2  
Ilustrasi system *IOT*

### 2.3 Arduino IDE

Arduino IDE adalah *perangkat lunak* yang digunakan untuk membuat *sketch* pemrograman pada *board* mikrokontroler. Arduino IDE digunakan untuk membuat, mengubah dan meng-*upload kode program* ke dalam sebuah *board*. Arduino IDE menggunakan bahasa pemrograman JAVA, yang dilengkapi dengan *library C/C++(wiring)*, yang membuat operasi *input/output* lebih mudah.

Penulisan *Sketch* pada *Software* Arduino IDE

Sebuah *file pemrograman* pada *Arduino IDE* disebut dengan *file Sketch*. *Sketch* yang disimpan memiliki ekstensi file **.ino**.

#### A. Struktur Dasar Penulisan *Sketch*

Pada penulisan program Arduino IDE ada dua stuktur dasar yaitu *setup ()* dan *loop ()*..:



```

void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}

```

Sumber: Hasil Penelitian (2022)

Gambar II.3

### Sketch arduino IDE

#### 1. *Void setup () {}*

*Void setup* merupakan fungsi yang hanya akan dijalankan satu kali saja Ketika mikrokontroller pertama kali dinyalakan.

#### 2. *Void loop () {}*

Fungsi ini akan dijalankan setelah *setup* (fungsi *void setup*) selesai, setelah dijalankan 1 kali, fungsi ini akan dijalankan secara terus menerus atau berulang sampai catu daya (*power*) dilepaskan dan *sketch* lain diupload pada *board* tersebut.

### B. *Syntak* dalam Penulisan Program

#### 1. *//* (komentar 1 baris)

Digunakan untuk memberi komentar atau catatan pada kode-kode yang dibuat.

#### 2. */\* \*/* (komentar 2 baris atau lebih)

Untuk menuliskan catatan pada beberapa baris sebagai komentar.

#### 3. *{ }* (kurung kurawal)

Digunakan untuk mendefinisikan kapan blok program mulai dan berakhir serta digunakan juga pada fungsi dan pengulangan.

#### 4. *;* (titik koma)

Setiap baris kode harus diakhiri dengan tanda; (titik koma) atau *semicolon*, jika ada titik koma yang hilang maka pesan *error* akan tampil Ketika sketch sedang *dicompile*.

### 5. Case Sensitive

Penulisan karakter menggunakan huruf kapital dan huruf biasa akan memiliki arti berbeda, sebagai contoh Buzzer (dengan B huruf kapital) dan buzzer adalah dua variable yang berbeda.

## C. Tombol fungsi pada *Software* Arduino IDE



Sumber: Hasil Penelitian (2022)

Gambar II.4

## Tombol fungsi pada Sketch Arduino IDE

### 1. Verify

*Verify* digunakan untuk memverifikasi kebenaran penulisan program pada *sketch* jika masih terdapat kesalahan akan muncul keterangan *error* serta detail kesalahan apa yang dimaksud.

### 2. Upload

*Upload* digunakan untuk mengirimkan atau memasukan program ke dalam *board* yang telah terhubung dengan *port* USB pada computer atau laptop.

### 3. *New*

*New* digunakan untuk membuat *sketch* baru atau membuka halaman *sketch* yang baru.

### 4. *Open*

*Open* digunakan untuk membuka *sketch* yang pernah dibuat dan disimpan.

### 5. *Save*

*Save* ditunjukan untuk menyimpan *sketch* atau program yang sudah dibuat.

### 6. Serial Monitor

Serial Monitor digunakan untuk menampilkan jendela komunikasi (pengiriman dan penerimaan data serial) antara board dan semua sensor atau modul yang dipasang dengan *programmer/user*. Serial monitor juga berfungsi untuk melihat masukan atau hasil pembacaan sensor apakah sudah sesuai atau belum juga bisa digunakan untuk menganalisa jalannya program apakah sudah sesuai dengan yang diharapkan atau belum.

## 2.5 MIT App Inventor 2

MIT App Inventor 2 adalah sebuah aplikasi berbasis web yang digunakan untuk membangun aplikasi Android secara visual dan intuitif, tanpa memerlukan pengetahuan pemrograman yang mendalam. Aplikasi ini dikembangkan oleh Massachusetts Institute of Technology (MIT) untuk membantu pengguna membangun aplikasi mobile dengan mudah dan cepat.



Sumber: <https://appinventor.mit.edu/>

## Gambar II.5

### *Logo MIT App Inventor 2*

Dengan MIT App Inventor 2, pengguna dapat membuat aplikasi mobile dengan cara menarik dan menjatuhkan elemen desain visual, seperti tombol, label, gambar, dan field input. Kemudian, pengguna dapat mengklik elemen tersebut untuk menentukan bagaimana aplikasi tersebut akan berfungsi. Aplikasi yang dibuat menggunakan MIT App Inventor 2 juga dapat diuji langsung pada emulator atau perangkat Android secara real-time.

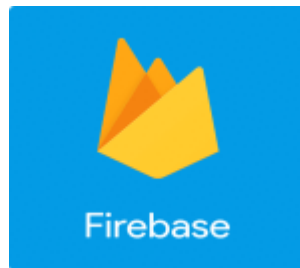
Selain itu, MIT App Inventor 2 juga memiliki fitur untuk mengakses berbagai jenis sensor pada perangkat Android, seperti GPS, kamera, dan sensor gerak. Hal ini memungkinkan pengguna untuk membangun aplikasi mobile yang lebih interaktif dan kaya fitur. MIT App Inventor 2 cocok untuk digunakan oleh pemula atau siapa saja yang ingin membangun aplikasi mobile tanpa perlu menguasai bahasa pemrograman yang kompleks. Dalam pembuatan aplikasi mobile menggunakan MIT App Inventor 2, pengguna akan dibantu dengan dokumentasi dan tutorial yang jelas dan mudah diikuti.

Dalam rangkaian pembuatan aplikasi, MIT App Inventor 2 juga menyediakan database untuk menyimpan data aplikasi. Dengan adanya database tersebut, pengguna dapat membuat aplikasi yang dapat mengakses data dari internet atau data yang tersimpan pada perangkat Android. MIT App Inventor 2 adalah aplikasi yang cukup populer dan banyak digunakan oleh pengguna Android. Aplikasi yang dibangun menggunakan MIT App Inventor 2 juga dapat diterbitkan pada Google Play Store, sehingga dapat diakses oleh pengguna Android di seluruh dunia.

## **2.6 Google Firebase**

*Google Firebase* adalah platform pengembangan aplikasi berbasis *cloud* yang menyediakan berbagai layanan untuk membangun, mengelola, dan mengembangkan aplikasi web dan mobile. *Firebase* menyediakan berbagai fitur seperti penyimpanan data, autentikasi pengguna, hosting aplikasi, analisis performa, dan masih banyak

lagi. *Firebase* menyediakan layanan *cloud storage* yang mudah digunakan dan dapat diakses dari berbagai *platform*. *Firebase* menyediakan layanan *Realtime Database*, yang merupakan basis data NoSQL yang menyediakan sinkronisasi *data real-time* antara aplikasi dan server, sehingga memungkinkan pengembang untuk membangun aplikasi web dan *mobile* yang responsif.



Gambar II.6

*Logo MIT App Inventor 2*

Selain itu, *Firebase* juga menyediakan layanan autentikasi pengguna yang mudah digunakan, sehingga pengguna dapat login ke aplikasi dengan akun Google, Facebook, Twitter, dan akun lainnya. *Firebase* juga menyediakan layanan Cloud Functions yang memungkinkan pengembang untuk menulis kode backend dengan mudah tanpa harus merakit dan mengelola infrastruktur backend mereka sendiri. *Firebase* juga menyediakan layanan hosting aplikasi, yang memungkinkan pengembang untuk mempublikasikan aplikasi web mereka dengan cepat dan mudah.

*Firebase Hosting* juga menyediakan layanan SSL gratis, sehingga pengguna dapat mengakses aplikasi web dengan aman. Selain itu, *Firebase* juga menyediakan layanan analisis performa yang dapat membantu pengembang memahami bagaimana pengguna berinteraksi dengan aplikasi mereka dan bagaimana aplikasi mereka dapat dioptimalkan untuk kinerja yang lebih baik. *Firebase* juga menyediakan layanan pengujian A/B dan layanan pengiriman notifikasi push.

*Firebase* dapat digunakan pada berbagai bahasa pemrograman, seperti Java, JavaScript, C++, dan banyak lagi. *Firebase* juga terintegrasi dengan layanan Google

Cloud Platform, sehingga pengguna dapat memanfaatkan fitur-fitur dan layanan yang disediakan oleh *Google Cloud Platform*. *Firebase* adalah salah satu *platform* pengembangan aplikasi berbasis *cloud* terbaik di pasaran saat ini, dan banyak digunakan oleh pengembang aplikasi web dan *mobile* di seluruh dunia.

## 2.7 Board NodeMCU

NodeMCU adalah sebuah *board* elektronik yang berbasis chip ESP8266 dengan kemampuan menjalankan fungsi mikrokontroler dan juga koneksi internet (*WiFi*). Terdapat beberapa pin *I/O* sehingga dapat dikembangkan menjadi sebuah aplikasi monitoring maupun *controlling* pada proyek *IOT*. NodeMCU ESP8266 dapat diprogram dengan *compiler*-nya Arduino, menggunakan Arduino IDE. Bentuk fisik dari NodeMCU ESP 8266, terdapat *port USB (mini-USB)* sehingga akan memudahkan dalam pemrogramannya.

NodeMCU ESP8266 merupakan modul turunan pengembangan dari modul platform IoT (Internet of Things) keluarga ESP8266 tipe ESP-12. Secara fungsi modul ini hampir menyerupai dengan platform modul arduino, tetapi yang membedakan yaitu dikhususkan untuk “*Connected to Internet*”.



Gambar II.7

NodeMCU dan Expansion Board

## 2.8 Modul wifi ESP8266



Modul wifi ESP8266 merupakan modul wifi yang berfungsi sebagai perangkat tambahan pada mikrokontroler seperti Arduino agar dapat terhubung langsung dengan wifi dan membuat koneksi TCP/IP. Modul ini memiliki tiga mode wifi yaitu *Station*, *Access Point* dan *Both*. Modul ini juga dilengkapi dengan prosesor, memori dan GPIO dimana jumlah pin bergantung dengan jenis ESP8266 yang kita gunakan.

Modul ini bisa berdiri sendiri tanpa menggunakan mikrokontroler apapun karena sudah memiliki perlengkapan layaknya mikrokontroler. *Firmware default* yang digunakan oleh perangkat ini menggunakan *AT Command*, selain itu ada beberapa *Firmware SDK* yang digunakan oleh perangkat ini berbasis *open-source* yang diantaranya adalah sebagai berikut:

- a) *Node MCU* dengan menggunakan *basic programming lua*.
- b) *MicroPython* dengan menggunakan *basic programming python*.
- c) *AT Command* dengan menggunakan perintah perintah *AT command*.



Gambar II.8

Chip ESP8266

## 2.9 Solenoid Valve

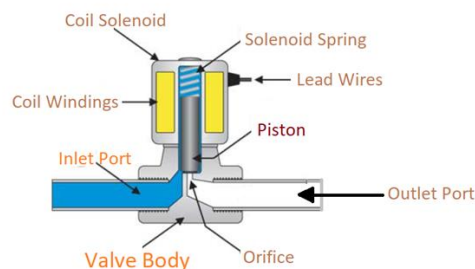
*Solenoid valve* adalah jenis *valve* yang digerakkan oleh medan magnet dari solenoid, yaitu koil kawat yang membentuk medan magnet saat dialiri arus listrik. Solenoid valve biasanya digunakan untuk mengontrol aliran cairan atau gas pada sistem hidrolik atau pneumatik.

*Solenoid valve* memiliki dua jenis konstruksi dasar: *normally open (NO)* dan *normally closed (NC)*. Pada *solenoid valve NO*, valve terbuka pada kondisi tanpa arus listrik dan tertutup saat medan magnet dibuat oleh arus listrik yang mengalir ke dalam solenoid. Sedangkan pada solenoid valve NC, valve tertutup pada kondisi tanpa arus listrik dan terbuka saat medan magnet dibuat oleh arus listrik yang mengalir ke dalam solenoid.

*Solenoid valve* dapat digunakan untuk berbagai aplikasi, seperti pada sistem pemanas, pendingin, pengaturan aliran bahan kimia, pengaturan aliran air, dan lain sebagainya. Selain itu, *solenoid valve* juga memiliki keuntungan dalam hal penghematan energi, kontrol presisi, dan kemampuan untuk bekerja secara otomatis.

*Solenoid valve* tersedia dalam berbagai ukuran dan jenis, tergantung pada jenis fluida yang akan diatur, tekanan dan suhu yang diinginkan, serta aplikasi khusus yang diperlukan. Beberapa jenis *solenoid valve* yang umum digunakan antara lain *solenoid valve 2/2-way* (dua jalur masuk/keluar), *solenoid valve 3/2-way* (tiga jalur masuk/keluar), dan *solenoid valve 5/2-way* (lima jalur masuk/keluar).

Secara sederhana konstruksi dari sebuah *solenoid valve* seperti gambar dibawah ini:



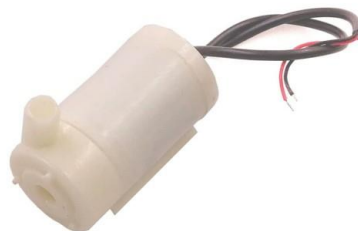
Sumber: <https://forumautomation.com/>

Gambar II.9  
*Komponen Solenoid Valve*

Meskipun *solenoid valve* memiliki banyak keuntungan, ada juga beberapa kekurangan yang harus diperhatikan, seperti risiko kebocoran jika terjadi kerusakan pada seal, risiko getaran yang dapat mempengaruhi kestabilan aliran cairan atau gas, dan risiko korosi pada material valve jika terjadi kontak dengan bahan kimia tertentu. Oleh karena itu, pemilihan *solenoid valve* yang tepat sangat penting untuk memastikan bahwa sistem hidrolik atau pneumatik dapat berjalan dengan efisien dan aman.

#### **2.10 Pompa DC Mini**

Pompa air DC mini adalah jenis pompa air yang menggunakan sumber daya listrik dari baterai DC. Ukuran pompa air DC mini relatif kecil dan biasanya digunakan untuk aplikasi yang membutuhkan aliran air yang rendah, seperti pada sistem irigasi, penyejuk udara, atau penyejuk mesin. Pompa air DC mini sangat populer karena ukurannya yang kecil dan mudah digunakan, serta konsumsi energi yang rendah.



Gambar II.10

### *Pompa air DC mini*

Pompa air DC mini biasanya menggunakan motor DC *brushless* yang efisien dan tahan lama. Motor DC brushless menggunakan sistem rotor dan stator yang tidak memiliki sikat (*brush*), sehingga menghasilkan kecepatan putar yang lebih stabil dan minim gesekan. Hal ini membuat pompa air DC mini lebih efisien dan memiliki umur yang lebih panjang dibandingkan dengan pompa air konvensional yang menggunakan motor dengan sikat.

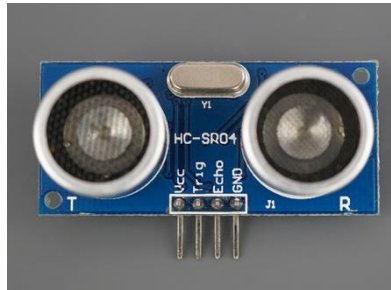
Pompa air DC mini biasanya dapat menghasilkan aliran air dengan kapasitas sekitar 2 hingga 8 liter per menit, tergantung pada ukuran dan jenisnya. Selain itu, pompa air DC mini juga memiliki keunggulan dalam hal kontrol kecepatan putar, sehingga aliran air dapat diatur sesuai dengan kebutuhan aplikasi.

Pompa air DC mini biasanya mudah dipasang dan dapat dihubungkan langsung dengan baterai DC atau sumber daya listrik lainnya yang sesuai dengan kebutuhan. Beberapa jenis pompa air DC mini dilengkapi dengan sensor otomatis yang dapat mendeteksi keberadaan air dan mematikan pompa secara otomatis jika air tidak ditemukan, sehingga mencegah kerusakan pada pompa.

Meskipun pompa air DC mini memiliki banyak keunggulan, ada beberapa hal yang perlu diperhatikan dalam penggunaannya. Salah satu hal yang perlu diperhatikan adalah kebersihan air yang digunakan, karena partikel-partikel padat dapat merusak impeller pada pompa. Selain itu, kestabilan tegangan listrik pada baterai DC atau sumber daya listrik yang digunakan juga perlu diperhatikan, karena fluktuasi tegangan dapat mempengaruhi performa pompa.

#### **2.11 Sensor ultrasonic HC-SR04**

Sensor ultrasonic HC-SR04 adalah sensor jarak yang digunakan untuk mengukur jarak antara sensor dengan objek di sekitarnya menggunakan gelombang suara ultrasonik. Sensor ini dapat mendeteksi objek yang berada di depannya dalam jangkauan 2 cm hingga 4 meter.



Gambar II.11

#### Modul Ultrasonic HC-SR04

Prinsip kerja dari sensor ultrasonic HC-SR04 adalah dengan mengirimkan sinyal gelombang suara ultrasonik ke objek di depannya dan kemudian menerima pantulan gelombang suara tersebut. Waktu tempuh pantulan gelombang suara dari objek dikalikan dengan kecepatan suara untuk menghitung jarak antara sensor dan objek.

Sensor ultrasonic HC-SR04 terdiri dari dua modul utama, yaitu modul transmitter dan modul receiver. Modul transmitter berfungsi untuk mengirimkan gelombang suara ultrasonik, sedangkan modul receiver berfungsi untuk menerima pantulan gelombang suara dan mengubahnya menjadi sinyal listrik yang dapat diolah oleh mikrokontroler atau komputer.

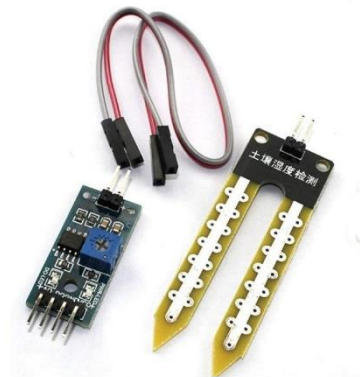
Sensor ultrasonic HC-SR04 sangat berguna untuk aplikasi yang memerlukan pengukuran jarak, seperti pada robotika, kendaraan otonom, pengukuran air pada tangki, dan lain sebagainya. Sensor ini juga cukup mudah digunakan dan dihubungkan dengan mikrokontroler atau komputer melalui koneksi digital seperti GPIO atau I2C.

Namun, sensor ultrasonic HC-SR04 juga memiliki beberapa kekurangan. Sensor ini tidak dapat mendeteksi objek yang tidak memiliki permukaan yang datar, seperti objek berbentuk bola atau objek yang sangat kecil. Selain itu, sensor ini juga rentan terhadap gangguan suara atau benda-benda yang ada di sekitarnya yang dapat memantulkan gelombang suara ultrasonik.

## 2.12 Soil Moisture Sensor

Soil moisture sensor adalah alat yang digunakan untuk mengukur kelembaban tanah pada suatu lokasi. Sensor ini bekerja dengan cara mengukur tingkat konduktivitas listrik dari tanah yang berkaitan dengan kandungan air dalam tanah tersebut. Semakin tinggi kandungan air dalam tanah, semakin tinggi pula konduktivitas listriknya.

Ada dua jenis soil moisture sensor yang umum digunakan, yaitu jenis resistive dan jenis capacitive. Sensor tipe resistive bekerja dengan memanfaatkan perbedaan resistansi antara tanah yang kering dan basah, sedangkan sensor tipe capacitive bekerja dengan cara mengukur perubahan kapasitansi permukaan elektroda pada sensor ketika bertemu dengan air.



Gambar II.12

### *Soil Moisture Sensor*

*Soil moisture sensor* umumnya terdiri dari dua bagian utama, yaitu probe atau elektroda yang ditanamkan di dalam tanah dan sebuah modul elektronik yang mengolah data dari probe tersebut. Beberapa sensor juga dilengkapi dengan antarmuka seperti *USB* atau *Bluetooth* untuk mengirimkan data ke perangkat lain.

Kelebihan dari *soil moisture sensor* adalah kemampuannya untuk membantu pengguna dalam mengatur frekuensi penyiraman tanaman dengan tepat, sehingga

penggunaan air dapat dioptimalkan dan efisiensi dapat ditingkatkan. Selain itu, sensor ini juga dapat membantu pengguna dalam memantau kesehatan tanaman dan mencegah risiko *overwatering* atau *underwatering*.

Namun, *soil moisture sensor* juga memiliki beberapa kekurangan. Salah satunya adalah sensor ini dapat memberikan hasil yang tidak akurat jika dipasang pada lokasi yang tidak mewakili kelembaban tanah secara keseluruhan. Selain itu, sensor ini juga memerlukan kalibrasi secara berkala untuk memastikan keakuratannya.

*Soil moisture sensor* dapat dihubungkan dengan board Arduino untuk membantu pengguna dalam mengukur kelembaban tanah pada suatu lokasi secara *real-time*. Pada umumnya, *soil moisture sensor* dihubungkan dengan papan Arduino menggunakan kabel jumper dan ditempatkan di dalam pot atau di tanah di dekat akar tanaman yang ingin diukur kelembabannya.

Untuk membaca data dari *soil moisture sensor*, pengguna dapat menggunakan board Arduino dengan salah satu dari dua jenis input analog, yaitu ADC (*analog to digital converter*) atau pin analog. Setelah itu, pengguna dapat memprogram board Arduino untuk mengukur nilai analog dari *soil moisture sensor* dan menampilkan data kelembaban tanah pada layar atau dapat diakses melalui koneksi internet.

### **2.13 Modul Relay 2 Channel**

Relay 2 channel adalah perangkat elektronik yang digunakan untuk mengontrol dan menghubungkan dua sirkuit listrik terpisah. Relay bekerja dengan cara mengoperasikan sebuah saklar atau switch dengan menggunakan sinyal listrik. Dalam hal ini, relay 2 channel memiliki dua switch yang dapat digunakan untuk mengontrol dua sirkuit listrik yang berbeda secara terpisah..

Salah satu contoh penggunaan relay 2 channel adalah dalam sistem otomatisasi rumah. Misalnya, relay 2 channel dapat digunakan untuk mengontrol lampu dan kipas angin pada ruangan yang sama. Dengan menggunakan relay 2

channel, kedua perangkat dapat dikendalikan secara independen, meskipun keduanya terhubung ke sirkuit yang sama.



Gambar II.13

#### *Relay 2 Channel*

Dalam relay 2 channel, terdapat dua jenis kontak atau switch, yaitu kontak Normally Open (NO) dan kontak Normally Closed (NC). Kontak NO membuka sirkuit ketika relay diaktifkan, sedangkan kontak NC menutup sirkuit ketika relay diaktifkan. Penggunaan kontak NO atau NC tergantung pada kebutuhan pengguna.

Secara umum, relay 2 channel sangat berguna dalam mengontrol perangkat listrik yang terpisah, terutama dalam sistem otomatisasi dan kontrol. Relay ini dapat diprogram dan dikendalikan dengan menggunakan mikrokontroler atau komputer, sehingga sangat cocok digunakan dalam aplikasi yang memerlukan kontrol dan monitoring yang presisi dan andal.

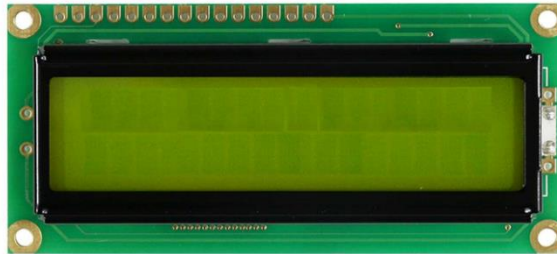
#### **2.14 Modul LCD 16X2**

LCD 16x2 adalah jenis layar kristal cair (liquid crystal display) yang memiliki 16 karakter pada baris pertama dan 16 karakter pada baris kedua, sehingga disebut juga sebagai "16x2" atau "16 kolom x 2 baris". LCD 16x2 umumnya digunakan sebagai output dalam berbagai proyek elektronik, terutama dalam mikrokontroler dan Arduino.

LCD 16x2 terdiri dari dua baris karakter, masing-masing dengan 16 karakter. Setiap karakter terdiri dari 5x8 dot matrix, yang berarti karakter tersebut terdiri dari 5



kolom dan 8 baris titik. Layar ini dapat menampilkan karakter ASCII, angka, huruf, dan simbol lainnya.



Gambar II.14

#### *Modul LCD 16x2*

Untuk mengontrol LCD 16x2, biasanya digunakan sebuah driver atau modul pengendali LCD, seperti HD 44780 atau sejenisnya. Modul pengendali ini memungkinkan LCD 16x2 untuk dihubungkan dengan mikrokontroler atau Arduino melalui jalur data dan kontrol.

Untuk menampilkan karakter pada LCD 16x2, kita dapat menggunakan library dan fungsi khusus pada bahasa pemrograman Arduino, seperti LiquidCrystal library. Dengan menggunakan fungsi ini, kita dapat mengatur posisi karakter pada layar, menampilkan karakter, membersihkan layar, dan melakukan berbagai operasi lainnya.

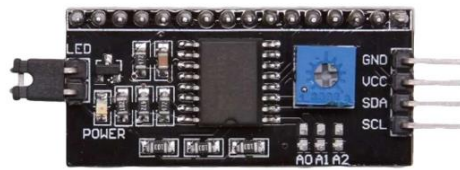
Keuntungan dari penggunaan LCD 16x2 adalah mudah dalam penggunaan dan pemasangan, dapat menampilkan informasi dengan jelas dan mudah dibaca, serta cocok digunakan pada berbagai proyek elektronik dan robotika. Namun, perlu diingat bahwa penggunaan LCD 16x2 juga memerlukan pengaturan dan konfigurasi yang sesuai dengan driver atau modul pengendalinya agar dapat berfungsi dengan baik.

### **2.15 Modul I2C untuk LCD 16X2**

Modul I2C untuk LCD 16x2 adalah sebuah perangkat elektronik yang digunakan untuk memudahkan penggunaan LCD 16x2 pada mikrokontroler atau Arduino dengan menghubungkannya melalui protokol komunikasi I2C (Inter-

Integrated Circuit). Modul I2C untuk LCD 16x2 memiliki dua komponen utama, yaitu driver atau pengendali LCD dan konverter I2C.

Driver atau pengendali LCD pada modul I2C adalah chip HD44780 atau sejenisnya yang digunakan untuk mengontrol LCD 16x2. Sedangkan konverter I2C berfungsi sebagai penerjemah sinyal antara mikrokontroler atau Arduino dan driver LCD.



Gambar II.15

#### *Modul I2C untuk LCD 16x2*

Dalam penggunaannya, modul I2C untuk LCD 16x2 dapat memudahkan penggunaan LCD 16x2 dengan menghilangkan kebutuhan untuk menghubungkan setiap pin pada LCD secara manual. Penggunaan modul I2C untuk LCD 16x2 juga memungkinkan pengguna untuk mengontrol LCD 16x2 dengan menggunakan sedikit pin I/O pada mikrokontroler atau Arduino.

Untuk menghubungkan modul I2C untuk LCD 16x2 pada mikrokontroler atau Arduino, dibutuhkan koneksi ke jalur data dan kontrol I2C pada mikrokontroler atau Arduino. Jalur data dan kontrol I2C pada mikrokontroler atau Arduino terdiri dari SDA (Serial Data Line) dan SCL (Serial Clock Line), yang digunakan untuk mentransmisikan data dan sinyal kontrol antara mikrokontroler atau Arduino dengan perangkat I2C, termasuk modul I2C untuk LCD 16x2.

Untuk menggunakan modul I2C untuk LCD 16x2 pada mikrokontroler atau Arduino, diperlukan pustaka khusus seperti `LiquidCrystal_I2C.h`. Dengan menggunakan pustaka ini, pengguna dapat mengontrol tampilan LCD 16x2 pada modul I2C dengan mudah dan cepat, tanpa perlu melakukan konfigurasi atau pengaturan khusus. Keuntungan dari penggunaan modul I2C untuk LCD 16x2 adalah memudahkan penggunaan LCD 16x2 pada mikrokontroler atau Arduino, menghemat

jumlah pin I/O yang digunakan, dan memudahkan pembacaan dan penulisan data pada LCD. Namun, perlu diingat bahwa penggunaan modul I2C juga memerlukan penyesuaian konfigurasi dan pengaturan yang sesuai agar dapat berfungsi dengan baik.

## **2.16 Android**

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware*, dan aplikasi. Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk membuat aplikasi mereka sendiri. Pada awalnya dikembangkan oleh Android Inc, sebuah perusahaan pendatang baru yang membuat perangkat lunak untuk ponsel yang kemudian dibeli oleh Google Inc. Untuk pengembangannya, dibentuklah *Open Handset Alliance* (OHA), konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Antarmuka pengguna pada sistem operasi Android berupa gerakan manipulasi langsung, menggunakan gerakan sentuh, misalnya menggeser, mengetuk, dan mencubit untuk memanipulasi objek di layar, serta papan ketik virtual untuk menulis teks. Selain perangkat layar sentuh, Google juga telah mengembangkan Android TV untuk televisi, Android Auto untuk mobil, dan Android Wear untuk jam tangan, masing-masingnya memiliki antarmuka pengguna yang berbeda. Varian Android juga digunakan pada Laptop, konsol permainan, kamera digital, dan peralatan elektronik lainnya.

Android adalah sistem operasi dengan sumber terbuka (*open source*), dan Google merilis kodenya di bawah Lisensi Apache dengan begitu memungkinkan perangkat lunak android untuk dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat dan pengembang aplikasi. Selain itu, Android memiliki banyak komunitas pengembang aplikasi (*apps*) yang memperluas fungsionalitas perangkat, umumnya ditulis dalam versi kustomisasi bahasa pemrograman Java.

Sifat Android yang terbuka telah mendorong munculnya sejumlah besar komunitas pengembang aplikasi untuk menggunakan kode sumber terbuka sebagai dasar proyek pembuatan aplikasi, dengan menambahkan fitur-fitur baru bagi pengguna tingkat lanjut atau mengoperasikan Android pada perangkat yang secara resmi dirilis dengan menggunakan sistem operasi lain. Versi dan distribusi pengguna system operasi android dapat dilihat pada table berikut:

Versi	Nama kode	Tanggal rilis	Level API	Distribusi
<b>11</b>	<i>11</i>	8 September 2020	30	
<b>10</b>	<i>10</i>	3 September 2019	29	
<b>9.0</b>	<i>Pie</i>	6 Agustus 2018	28	
<b>8.0</b>	<i>Oreo</i>	21 Agustus 2017	26	
<b>7.0</b>	<i>Nougat</i>	22 Agustus 2016	24	Kurang dari 0.1%
<b>6.0</b>	Marshmallow	19 Agustus 2015	23	
<b>5.1</b>	<i>Lollipop</i>	9 Maret 2015	22	
<b>5.0</b>	<i>Lollipop</i>	15 Oktober 2014	21	
<b>4.4.x</b>	<i>KitKat</i>	31 Oktober 2013	19	24,5%
<b>4.3.x</b>	Jellybean	24 Juli 2013	18	8%
<b>4.2.x</b>	Jellybean	13 November 2012	17	20,7%
<b>4.1.x</b>	Jellybean	9 Juli 2012	16	25,1%
<b>4.0.3–4.0.4</b>	<i>Ice Cream Sandwich</i>	16 Desember 2011	15	9,6%
<b>3.2</b>	<i>Honeycomb</i>	15 Juli 2011	13	
<b>3.1</b>	<i>Honeycomb</i>	10 Mei 2011	12	
<b>2.3.3–2.3.7</b>	<i>Gingerbread</i>	9 Februari 2011	10	11,7%

<b>2.3–2.3.2</b>	<i>Gingerbread</i>	6 Desember 2010	9	
<b>2.2</b>	<i>Froyo</i>	20 Mei 2010	8	0,7%
<b>2.0–2.1</b>	<i>Eclair</i>	26 Oktober 2009	7	
<b>1.6</b>	<i>Donut</i>	15 September 2009	4	
<b>1.5</b>	<i>Cupcake</i>	30 April 2009	3	

Tabel III.2

Versi dan distribusi android

### 2.17 Fritzing

Fritzing adalah aplikasi open-source yang digunakan untuk mendesain rangkaian elektronik dan skematik, serta merancang circuit board (PCB) secara visual dan mudah. Aplikasi ini dapat membantu pengguna yang tidak memiliki latar belakang teknis mendesain rangkaian elektronik dengan mudah dan cepat.

Fritzing menyediakan library komponen elektronik standar yang mencakup berbagai jenis komponen seperti resistor, kapasitor, transistor, dan sensor, yang dapat digunakan pengguna untuk merancang rangkaian mereka. Selain itu, pengguna juga dapat membuat library komponen kustom yang dapat digunakan di desain mereka selanjutnya.



Gambar II.16

*Logo aplikasi fritzing*

Fritzing memungkinkan pengguna untuk merancang skematik dengan cara menarik dan menjatuhkan komponen ke canvas, dan kemudian menghubungkan komponen tersebut menggunakan kabel. Setelah rangkaian skematik dirancang,

pengguna dapat merancang PCB dengan menempatkan komponen pada board dan menentukan jalur koneksi. Fritzing juga menyediakan fitur visualisasi 3D yang memungkinkan pengguna melihat tampilan realistis dari circuit board mereka.

Salah satu fitur unggulan Fritzing adalah kemampuan untuk membuat breadboard view dari skematik. Breadboard view adalah tampilan dari rangkaian elektronik pada breadboard, yang memungkinkan pengguna untuk menguji rangkaian elektronik mereka secara fisik sebelum merancang PCB. Dengan menggunakan breadboard view, pengguna dapat memastikan bahwa rangkaian elektronik mereka bekerja dengan baik sebelum merancang circuit board-nya.

Fritzing juga mendukung ekspor file desain dalam berbagai format, termasuk PDF, SVG, dan PNG, sehingga pengguna dapat membagikan desain mereka dengan mudah. Aplikasi ini dapat digunakan oleh pemula maupun pengguna yang berpengalaman dalam merancang rangkaian elektronik, dan telah menjadi alat yang populer di kalangan DIY electronics and maker community.

## **2.18 UML**

*Unified Modelling Language* (UML) adalah Bahasa penggambaran/visualisasi yang didefinisikan oleh *Object Management Group* (OMG) untuk memvisualisasikan, menspesifikasikan, membangun dan mendokumentasikan penelitian dari sebuah sistem perangkat lunak (Jonatan Simpson, 2017)

*Unified Modelling Language* (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk

menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan.

Melalui berbagai penjelasan rumit yang terdapat di dokumen dan buku-buku UML, sebenarnya konsepsi dasar UML dapat dirangkum seperti pada tabel II.2

<i>Major Area</i>	<i>View</i>	<i>Diagrams</i>	<i>Main Concepts</i>
<i>Structural</i>	<i>Static view</i>	<i>Class diagram</i>	<i>Class, association, generalization, dependency, realization, interface</i>
<i>Use case view</i>	<i>Use case diagram</i>	<i>Use case, actor, association, extend, include, use case generalization</i>	
<i>Implementation view</i>	<i>Component diagram</i>	<i>Component, interface, dependency, realization</i>	
<i>Deployment view</i>	<i>Deployment diagram</i>	<i>Node, component, dependency, location</i>	
<i>Dynamic</i>	<i>State machine view</i>	<i>Statechart diagram</i>	<i>State, event, transition, action</i>
<i>Activity view</i>	<i>Activity diagram</i>	<i>State, activity, completion transition, fork, join</i>	
<i>Interaction view</i>	<i>Sequence diagram</i>	<i>Interaction, object, message, activation</i>	
<i>Collaboration diagram</i>	<i>Collaboration, interaction,</i>		

	<i>collaboration role, message</i>		
<i>Model management</i>	<i>Model management view</i>	<i>Class diagram</i>	<i>Package, subsystem, model</i>
<i>Extensibility</i>	<i>All</i>	<i>All</i>	<i>Constraint, stereotype, tagged values</i>

Tabel II.3

### Konsep Dasar UML

Abstraksi konsep dasar UML yang terdiri dari *structural classification*, *dynamic behavior*, dan *model management*, dapat dipahami dengan mudah apabila melihat tabel 2.3. *Main concepts* dapat dipandang sebagai *term* yang akan muncul pada saat membuat diagram, sedangkan *view* adalah kategori dari diagram tersebut. Berikut merupakan beberapa diagram yang ada pada UML:

- a. *Use Case Diagram*
- b. *Class Diagram*
- c. *Activity Diagram*
- d. *Sequence Diagram*

#### 2.18.1 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* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, membuat sebuah daftar belanja, dan sebagainya. Seorang/sebuah actor adalah sebuah *entitas* manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

*Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan



merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat menambahkan fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang ditambahkan akan dipanggil setiap kali *use case* yang menambahkan dieksekusi secara normal.

Sebuah *use case* dapat ditambahkan oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat melakukan *extend use case* lain dengan behaviour yang dimiliki. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

### 2.18.2 Class Diagram

*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi) [8]. *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:

- a. Nama (dan *stereotype*)
- b. Atribut
- c. Metode

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.
- c. *Public*, dapat dipanggil oleh siapa saja.

*Class* dapat merupakan implementasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class*. Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time*. Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package*. Kita juga dapat membuat diagram yang terdiri atas *package*. Hubungan Antar *Class*:

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.
2. Agregasi, yaitu hubungan yang menyatakan bagian (“terdiri atas”). Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
3. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang di-*passing* dari satu *class* kepada *class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan *sequence* diagram yang akan dijelaskan kemudian.

### 2.18.3 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, di mana 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 aktivitas. Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour* pada kondisi tertentu. Untuk mengilustrasikan prosesproses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity* diagram dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

#### 2.18.4 Sequence Diagram

*Sequence* diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk 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 respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang menjadi *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.

*Message* digambarkan sebagai garis berpanah dari satu objek ke objek lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation bar* menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*. Untuk objekobjek

yang memiliki sifat khusus, standar UML mendefinisikan icon khusus untuk objek *boundary*, *controller* dan *persistent entity*.

## **2.19 Star UML**

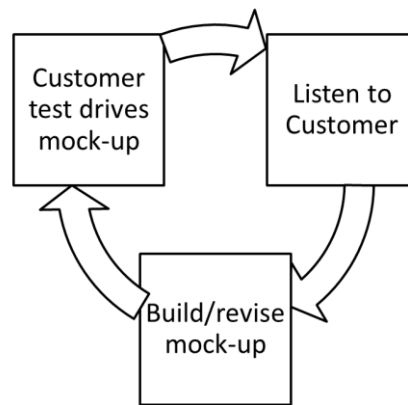
StarUML adalah sebuah proyek *open source* untuk pengembangan secara cepat, fleksibel, *extensible*, *featureful*, dan bebas-tersedia. UML / platform MDA berjalan pada platform Win32. Tujuan dari proyek StarUML adalah untuk membangun sebuah alat pemodelan perangkat lunak dan juga platform yang menarik adalah pengganti alat UML komersial seperti Rational Rose, together dan sebagainya. StarUML mendukung UML (Unified Modeling Language) dan dilengkapi 11 macam diagram yang berbeda, selanjutnya mendukung notasi UML 2.0 dan juga mendukung pendekatan MDA (*Model Driven Architecture*) dengan dukungan konsep UML. StarUML dapat memaksimalkan produktivitas dan kualitas dari suatu software project.

## **2.20 Prototyping**

*Prototyping* merupakan salah satu metode pengembangan perangkat lunak yang menggunakan pendekatan untuk membuat rancangan dengan cepat dan bertahap sehingga dapat segera dievaluasi oleh calon pengguna/klien. Dengan metode *prototyping* ini pengembang dan klien dapat saling berinteraksi selama proses pembuatan *prototype* sistem. Terkadang sering terjadi, klien hanya mendefinisikan secara umum apa yang dikehendaki tanpa menyebutkan proses masukan (*input*) dan keluaran (*output*) dari sistem yang akan dibuat. Untuk mengatasi ketidakselarasan tersebut maka harus dibutuhkan kerjasama yang baik di antara keduanya, sehingga pengembang akan mengetahui dengan benar apa yang dibutuhkan klien. Dengan demikian nantinya akan menghasilkan sebuah rancangan sistem yang interaktif sesuai dengan kebutuhan.

### **2.20.1 Metode Prototype**

Metode *Prototype* menurut Pressman (2002:40), dimulai dengan mengumpulkan kebutuhan. Pengembang dan klien bertemu guna mendefinisikan obyektif keseluruhan dari perangkat lunak, mengidentifikasi segala kebutuhan dari segi *input* dan format *output* serta gambaran *interface*, kemudian dilakukan perancangan cepat. Dari hasil perancangan cepat tersebut nantinya akan dilakukan pengujian dan evaluasi. Penjelasan lengkap pada metode *prototype* akan dijelaskan melalui gambar berikut.



Gambar II.17

Ilustrasi metode *prototype* (Pressman,2002)

Penjelasan untuk Gambar II.20 diatas adalah sebagai berikut:

- a. *Listen to Customer*, pada tahap ini merupakan identifikasi kebutuhan pengguna, permasalahan yang dialami oleh pengguna. Data yang diperoleh dari permasalahan dijadikan sebagai acuan pencarian solusi dan pengembangan pada tahap selanjutnya.
- b. *Build and Revise Mock-up* , yaitu proses perancangan *prototype* pada sistem yang diusulkan oleh pengguna, dengan tahapan sebagai berikut:
  - Perancangan proses masukan dan keluaran
  - Perancangan *Use-Case Diagram* dan *Activity Diagram*
  - Perancangan antarmuka pengguna

- c. *Customer Test Drives Mock-up*, Pada tahap ini akan dilakukan pengujian terhadap *prototype* pertama dari sistem yang telah dibuat. Jika *prototype* pertama belum sesuai dengan kebutuhan pengguna maka akan dilakukan proses perbaikan terus menerus hingga sistem yang dihasilkan benar-benar sesuai dengan kebutuhan pengguna. Proses pengujian *prototype* sistem nantinya menggunakan teknik pengujian *black box*. Hasil dari pengujian *black box* tersebut nantinya akan dibahas pada pembahasan selanjutnya di Bab IV mengenai hasil pengujian.

### **2.20.2 Keunggulan dan Kelemahan *Prototyping***

Beberapa keunggulan *prototyping* diantaranya :

- a) Komunikasi antara pengembang dan pengguna sangat baik.
- b) Kebutuhan pengguna akan lebih terpenuhi oleh pengembang.
- c) Kedua belah pihak berperan aktif dalam pengembangan sistem.
- d) Lebih menghemat waktu dan tenaga.
- e) Sistem lebih mudah diterapkan

Sedangkan kelemahan dari *prototyping* adalah sebagai berikut:

- a) Kualitas perangkat lunak secara keseluruhan dan kemampuan pemeliharaan untuk jangka waktu lama sulit diketahui.
- b) Perangkat lunak yang dihasilkan biasanya untuk jangka pendek atau sederhana dan sulit untuk dikembangkan dikemudian hari.

## **2.1. Penelitian Terkait**

Dari beberapa telaah terhadap penelitian , ada beberapa yang terkait dengan penelitian yang penulis lakukan.

Penelitian pertama yang berhasil peneliti temukan adalah penelitian yang dilakukan oleh Ahmad Maqhribi Daulay, Andik Bintoro, dan Muchlis Abdul Muthalib (2022) yang berjudul “*Sistem Monitoring Air Pada Tangki Berbasis Internet of Things (Iot) Blynk App*” Tujuan dari penelitian ini adalah untuk untuk memantau ketinggian air pada tangka dan penggunaan air dari tangka penampungan tersebut dan mengontrol kinerja pompa air. Penelitian ini telah berhasil mensimulasikan pemantauan dan pengontrolan penggunaan air bersih menggunakan *smartphone* yang terkoneksi *wireless* (wifi) dengan antarmuka pengguna menggunakan aplikasi Blynk.

Penelitian kedua adalah penelitian yang dilakukan oleh Putra Rifqi Mahardika, Fuji April Lani, dan Rini Suwartika (2022) tentang “*Perancangan Sistem Control Tandon Air Menggunakan Sensor Hc-Sr04 Berbasis Internet Of Things*” penelitian ini juga telah berhasil memantau ketersediaan air dalam tangka penampungan dan menampilkannya melalui aplikasi Blynk dan layer LCD secara real-time juga dapat mengontrol pompa air apakah dalam kondisi On atau Off.

Penelitian ketiga adalah penelitian yang dilakukan oleh Arief Widodo, Farid Baskoro, dan Nur Kholis (2021) yaitu tentang “*Sistem Monitoring Level Ketinggian Air Pada Tandon Rumah Tangga Berbasis IoT (Internet of Things)*” pada penelitian ini fokus perhatian adalah pada sisi notifikasi Ketika ketinggian air dalam tangka penampungan mencapai titik tertentu dalam penelitian ini tampilan antarmuka yang digunakan untuk memantau ketinggian air dan mengontrol pompa adalah aplikasi Blynk.

Penelitian keempat adalah penelitian yang dilakukan oleh Yudi Herdiana dan Angga Triatna (2020) dengan judul “*Prototype Monitoring Ketinggian Air Berbasis Internet Of Things Menggunakan Blynk Dan Nodemcu Esp8266 Pada Tangki*” penelitian telah berhasil merancang sebuah sistem pemantauan

ketersediaan air dalam tangki dan keseluruhan alat monitoring ketinggian air menampilkan volume air pada aplikasi Blynk sehingga pengguna mengetahui berapa kapasitas air yang ada didalam tangki dan berapa ketinggian air secara *real-time*.

Penelitian kelima adalah penelitian yang dilakukan oleh Mochamad Susantok dan Tama Ramadhan (2021) tentang “*Manajemen Ketersediaan dan Penggunaan air pada Rumah Tangga Berbasis IoT*” penelitian ini telah berhasil membuat fungsi manajemen ketersediaan air dalam tandon berjalan dengan sangat baik melalui algoritma percabangan bersyarat yang menghasilkan sistem otomasi. Sistem mampu mengontrol ketersediaan air dalam tandon antara 15% - 85% dari kapasitas dengan prosentase kesalahan 1.96% - 3.84%. Selain itu fungsi manajemen penggunaan air juga berhasil mengukur air yang telah digunakan dengan prosentase kesalahan 1.72%. Sistem ini mencegah air meluber saat pengisian tandon yang tidak terkontrol akibat pompa air masih hidup disaat air sudah penuh..

Berdasarkan beberapa penelitian diatas penulis ingin mengkombinasi beberapa kelebihan dari satu penelitian untuk menutupi kekurangan dari penelitian yang lain yaitu dengan mengkombinasikan sensor ultrasonic dengan sensor kelembaban tanah serta menggunakan solenoid valve untuk mengatur ketersediaan air dalam tangki penampungan dan pompa air mini untuk menyirami tanaman, selain itu penulis juga ingin menggunakan Google Firebase dan MIT App Inventor 2 sebagai basis data dan antarmuka pengguna.

Adapun alasan penulis menggunakan kombinasi Google Firebase dan MIT App Inventor 2 untuk menggantikan aplikasi Blynk adalah sebagai berikut :

1. Aplikasi Blynk IoT (v2) mengalami banyak perubahan dan fitur untuk menampilkan data ketinggian air dalam bentuk grafik adalah fitur berbayar



2. MIT App Inventor 2 lebih mudah dirancang sesuai kebutuhan
3. Google Firebase menyediakan layanan *real-time database* secara gratis
4. Aplikasi (.apk) yang dihasilkan dari MIT App Inventor 2 bisa digunakan pada beberapa smartphone tidak seperti Aplikasi Blynk IoT (v2) dalam versi gratis.

## **BAB III**

### **METODOLOGI PENELITIAN**

#### **3.1. Teknik Pengumpulan Data**

Pengumpulan data dapat dilakukan dalam berbagai *setting* dan berbagai sumber dan berbagai cara. Bila dilihat dari *setting*nya data dapat dikumpulkan pada *setting* alamiah (*natural seting*), pada laboratorium dengan metode eksperimen, di rumah dengan berbagai responden, dan lain-lain. Bila dilihat dari sumber datanya, maka pengumpulan data dapat menggunakan sumber *primer* dan *sekunder*. Sumber *primer* adalah sumber data yang langsung memberikan data kepada pengumpul data, dan sumber sekunder merupakan sumber yang tidak langsung memberikan data pada pengumpul data, misalnya lewat orang lain atau lewat dokumen. Selanjutnya kalau dilihat dari segi cara atau teknik pengumpulan data, maka teknik pengumpulan data dapat dilakukan dengan interview, kuesioner (angket), observasi (Sugiyono, 2012: 193-194).

##### **1. Observasi**

Observasi merupakan salah satu teknik pengumpulan data yang sangat lazim dalam metode penelitian kualitatif. Observasi hakikatnya merupakan kegiatan dengan menggunakan pancaindera, bisa penglihatan, penciuman, pendengaran, untuk memperoleh informasi yang diperlukan untuk menjawab masalah penelitian. Hasil observasi berupa aktivitas, kejadian, peristiwa, objek, kondisi atau suasana tertentu, dan perasaan emosi seseorang. Observasi dilakukan untuk memperoleh gambaran riil suatu peristiwa atau kejadian untuk menjawab pertanyaan penelitian (Guba dan Lincoln, 1981: 191-193).

Bungin (2007: 115-117) mengemukakan beberapa bentuk observasi, yaitu: 1). Observasi partisipasi, 2). observasi tidak terstruktur, dan 3). observasi kelompok. Berikut penjelasannya:

- Observasi partisipasi adalah (*participant observation*) adalah metode pengumpulan data yang digunakan untuk menghimpun data penelitian melalui pengamatan dan penginderaan di mana peneliti terlibat dalam keseharian informan.
- Observasi tidak terstruktur ialah pengamatan yang dilakukan tanpa menggunakan pedoman observasi, sehingga peneliti mengembangkan pengamatannya berdasarkan perkembangan yang terjadi di lapangan.
  - Observasi kelompok ialah pengamatan yang dilakukan oleh sekelompok tim peneliti terhadap sebuah isu yang diangkat menjadi objek penelitian.

Observasi yang penulis lakukan adalah kombinasi antara observasi partisipasi dan observasi tidak terstruktur karena ide dan gagasan tentang “Rancang Bangun Sistem Pemantauan Tangki Air Dan Kelembaban Tanah Berbasis *Internet Of Thing (IoT)* dengan Google Firebase dan Mit App Inventor 2” datang dari pengalaman dan pengamatan masalah yang terjadi dalam kehidupan sehari-hari yang penulis alami.

Sejauh ini ada beberapa masalah yang sering penulis hadapi berkaitan dengan sering terjadi kekosongan tangki saat air akan digunakan apalagi ketika pasokan air dari PDAM sedang mengalami gangguan. Adapula kelalaian pengisian air tangki yang tidak terkontrol hingga membuat air meluber dan mengakibatkan pemborosan air yang akan berdampak pada pembengkakan biaya tagihan PDAM tiap bulannya.

Untuk mengatasi masalah ini, sistem pemantauan dan pengontrolan air melalui *smartphone* tentu sangat diperlukan. Dengan adanya perkembangan teknologi Informasi yang semakin pesat tentu dapat membantu dalam mengatasi permasalahan tersebut..

## **2. Wawancara**

Wawancara ialah proses komunikasi atau interaksi untuk mengumpulkan informasi dengan cara tanya jawab antara peneliti dengan informan atau subjek penelitian (Emzir, 2010: 50). Dengan kemajuan teknologi informasi seperti saat ini, wawancara bisa saja dilakukan tanpa tatap muka, yakni melalui media telekomunikasi. Pada hakikatnya wawancara merupakan kegiatan untuk memperoleh informasi secara mendalam tentang sebuah isu atau tema yang diangkat dalam penelitian. Atau, merupakan proses pembuktian terhadap informasi atau keterangan yang telah diperoleh lewat teknik yang lain sebelumnya.

Byrne (2001) menyarankan agar sebelum memilih wawancara sebagai metoda pengumpulan data, peneliti harus menentukan apakah pertanyaan penelitian dapat dijawab dengan tepat oleh orang yang dipilih sebagai partisipan. Studi hipotesis perlu digunakan untuk menggambarkan satu proses yang digunakan peneliti untuk memfasilitasi wawancara.

Dalam melakukan wawancara penulis memilih rekan-rekan penulis yaitu para staff programmer dan network engineer untuk menggali informasi tentang teknologi sistem monitoring dan *IOT*.

### 3. Studi Pustaka

Selain melalui wawancara dan observasi, informasi juga bisa diperoleh lewat fakta yang tersimpan dalam bentuk surat, catatan harian, arsip foto, hasil rapat, cenderamata, jurnal kegiatan dan sebagainya. Data berupa dokumen seperti ini bisa dipakai untuk menggali informasi yang terjadi di masa silam. Peneliti perlu memiliki kepekaan teoretik untuk memaknai semua dokumen tersebut sehingga tidak sekadar barang yang tidak bermakna (Faisal, 1990: 77).

Studi pustaka tentang teknologi sistem monitoring *dan IoT* penulis banyak mendapatkan informasi dari *ebook* dan buku-buku tentang penerapan teknologi sistem monitoring *dan IoT* yang banyak tersedia di internet dan toko buku, selain itu penulis juga sering berdiskusi dan membaca kembali tulisan-tulisan di *group* facebook yang berkaitan dengan teknologi sistem monitoring *dan IoT*.

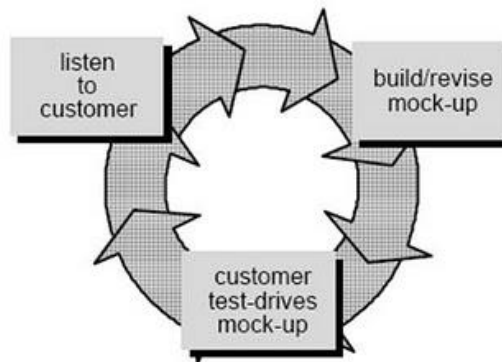
#### 3.2. Model Pengembangan Sistem

*Prototyping* merupakan salah satu metode pengembangan perangkat lunak yang menggunakan pendekatan untuk membuat rancangan dengan cepat dan bertahap sehingga dapat segera dievaluasi oleh calon pengguna/klien.

Dengan metode *prototyping* ini pengembang dan klien dapat saling berinteraksi selama proses pembuatan *prototype* sistem. Terkadang sering terjadi, klien hanya mendefinisikan secara umum apa yang dikehendaki tanpa menyebutkan proses masukan (*input*) dan keluaran (*output*) dari sistem yang akan dibuat. Untuk mengatasi ketidakselarasan tersebut maka harus dibutuhkan kerjasama yang baik di antara keduanya, sehingga pengembang akan mengetahui dengan benar apa yang dibutuhkan klien. Dengan demikian

nantinya akan menghasilkan sebuah rancangan sistem yang interaktif sesuai dengan kebutuhan

### 1. Langkah-langkah *prototyping*



Gambar III.1

Ilustrasi metode *prototype* (Pressman,2002)

Penjelasan untuk Gambar III.1 diatas adalah sebagai berikut:

- a. *Listen to Customer*, pada tahap ini merupakan identifikasi kebutuhan pengguna, permasalahan yang dialami oleh pengguna. Data yang diperoleh dari permasalahan dijadikan sebagai acuan pencarian solusi dan pengembangan pada tahap selanjutnya.
- b. *Build and Revise Mock-up* , yaitu proses perancangan *prototype* pada sistem yang diusulkan oleh pengguna, dengan tahapan sebagai berikut:
  - Perancangan proses masukan dan keluaran

- Perancangan *Use-Case Diagram* dan *Activity Diagram*
- Perancangan antarmuka pengguna

c. *Customer Test Drives Mock-up*, Pada tahap ini akan dilakukan pengujian terhadap *prototype* pertama dari sistem yang telah dibuat. Jika *prototype* pertama belum sesuai dengan kebutuhan pengguna maka akan dilakukan proses perbaikan terus menerus hingga sistem yang dihasilkan benar-benar sesuai dengan kebutuhan pengguna. Proses pengujian *prototype* sistem nantinya menggunakan teknik pengujian *black box*. Hasil dari pengujian *black box* tersebut nantinya akan dibahas pada pembahasan selanjutnya di Bab IV mengenai hasil pengujian.

## 2. Keunggulan dan kelemahan *prototyping*

Beberapa keunggulan *prototyping* diantaranya :

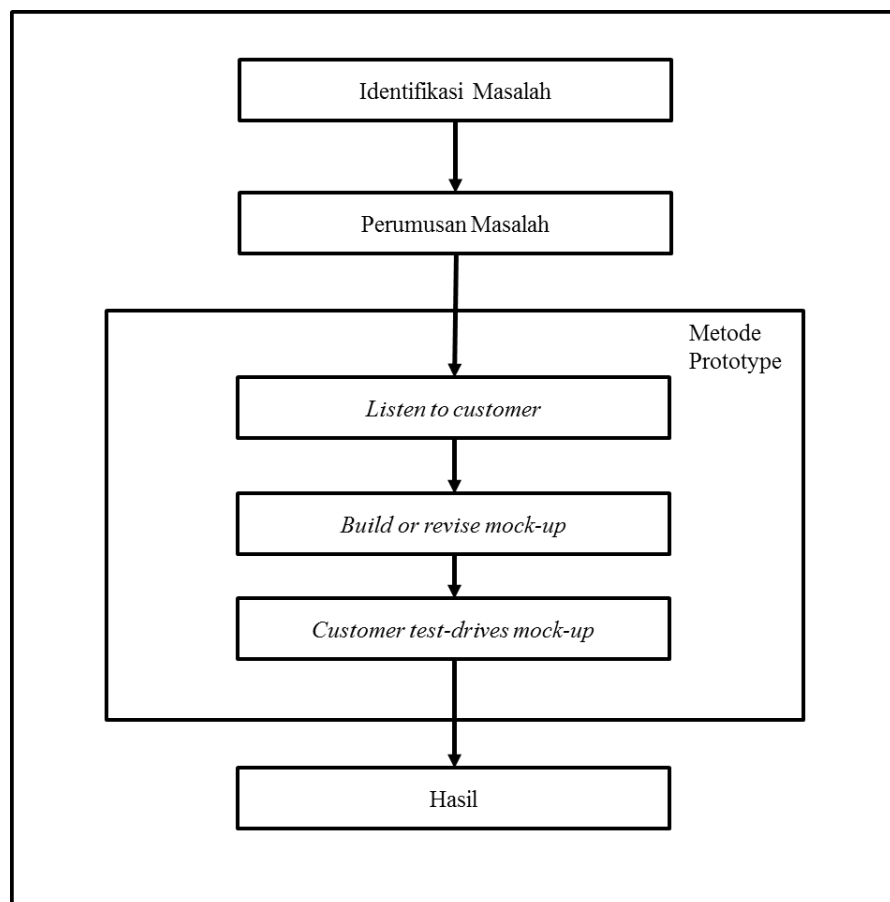
- a. Komunikasi antara pengembang dan pengguna sangat baik.
- b. Kebutuhan pengguna akan lebih terpenuhi oleh pengembang.
- c. Kedua belah pihak berperan aktif dalam pengembangan sistem.
- d. Lebih menghemat waktu dan tenaga.
- e. Sistem lebih mudah diterapkan

Sedangkan kelemahan dari *prototyping* adalah sebagai berikut:

- a. Kualitas perangkat lunak secara keseluruhan dan kemampuan pemeliharaan untuk jangka waktu lama sulit diketahui.
- b. Perangkat lunak yang dihasilkan biasanya untuk jangka pendek atau sederhana dan sulit untuk dikembangkan dikemudian hari.

### 3.3.Kerangka Pemikiran

Dalam penelitian ini penulis ingin mencoba menggabungkan teknologi *Internet of Thing* dengan peralatan-peralatan rumah tangga untuk meningkatkan keamanan dan kenyamanan rumah tinggal dan agar penulis menjadi lebih mudah dalam melakukan kontrol terhadap peralatan tersebut baik secara manual maupun otomatis. Berikut penulis sajikan kerangka pemikiran yang mendasari dilakukannya penelitian ini





.Sumber : Hasil Penelitian (2022)

### Gambar III.2

#### Ilustrasi kerangka pemikiran

Berikut adalah uraian pembahasan masing masing tahap dalam penelitian seperti yang digambarkan oleh gambar III.2 diatas :

1. Identifikasi masalah, dalam tahapan ini menjelaskan tentang masalah umum yang dihadapi yaitu kesulitan untuk memantau ketersediaan air bersih pada tangk penampungan dan kelembaban tanah.
2. Perumusan masalah, dalam tahapan ini menjelaskan lebih rinci dan terarah masalah apa saja yang harus diselesaikan dan didapatkan solusinya dari masalah ketersediaan air bersih , pengendalian katup pengisian air dari PDAM dan penyiraman tanaman yang terkendali.
3. *Listen to customer*, pada tahap ini proses menganalisa kasus-kasus permasalahan yang sering terjadi sebagai bahan untuk proses perancangan perangkat lunak dan perangkat keras.
4. *Build* atau *revise mock-up*, adalah tahapan membuat dan memperbaiki kembali *prototype* yang telah dibuat secara cepat sesuai permasalahan yang ditemui saat proses pengembangan.
5. *Customer test-drives mock-up*, tahap mencoba aplikasi dan evaluasi *prototype* dengan metode black box testing untuk diketahui apakah aplikasi yang dibuat telah sesuai dengan harapan dan mampu menyelesaikan masalah yang dihadapi.
6. Hasil, Dengan adanya rancang bangun ini memantau ketersediaan air dan kelembaban tanah jadi lebih mudah karena semuanya dapat dilakukan melalui *smartphone*.