e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

# DEVELOPMENT OF WATER LEVEL MONITORING APPLICATIONS IN SMART HOME SYSTEMS USING FLUTTER

*Pengembangan Aplikasi Monitoring Ketinggian Air Di Sistem Smart Home Menggunakan Flutter*

**Muhammad Hendriawan[1], Haryono[2], Thomas Budiman[3]**

Program Studi Teknologi Informasi[1],
Program Studi Teknologi Informasi[2],
Program Studi Teknik Informatika[3]
Universtias Pradita[1], Universitas Pradita[2], STMIK Jayakarta[3]

Muhammad.hendriawan@student.pradita.ac.id[1],
haryono@pradita.ac.id[2], thomas@stmik.jayakarta.ac.id[3]

**Received:** August 8, 2023**. Revised:** August 22, 2023**. Accepted:** August 23, 2023
**Issue Period:** Vol.7 No.2 (2023), Pages 213-240

**Abstrak:** Monitoring ketinggian air di sistem smart home merupakan salah satu upaya untuk meningkatkan kenyamanan dan keamanan penghuninya. Oleh karena itu, penelitian ini bertujuan untuk merancang dan mengembangkan sebuah aplikasi monitoring ketinggian air yang dapat diimplementasikan di sistem smart home dengan menggunakan ESP8266 NodeMCU, sensor ultrasonik HC-SR04, protokol MQTT, dan framework Flutter. Dengan mengimplementasikan Flutter sebagai opensource framework untuk membuat aplikasi diharapkan bisa memonitoring ketinggian air secara real time dan dapat diakses dari berbagai platform: seperti Android, iOS, Web, Windows, macOS, dan Linux. Hasil pengujian diharapkan dapat berfungsi dengan baik dan memberikan informasi yang akurat dan real-time terkait ketinggian air pada area sekitar dan bisa diakses menggunakan berbagai platform. Penelitian ini sangat mungkin untuk berhasil dilakukan karena sudah banyak penelitian terdahulu yang melakukan penelitian terhadap komponen yang sama dalam penelitian ini.

**Kata kunci:** IoT, Flutter, smart home system

***Abstract:*** *Monitoring the water level at smart home system is an effort to increase the comfort and safety of its residents. Therefore, this study aims to design and develop a water level monitoring application that can be implemented to smart home system using the ESP8266 NodeMCU, HC-SR04 ultrasonic sensor, MQTT protocol, and the Flutter framework. By implementing Flutter as an open-source framework for creating applications, it can be monitor water levels in real-time accessed from various platforms: such as Android, iOS, Web, Windows, macOS, and Linux. The test*

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

*expected to function correctly and provide accurate and real-time information regarding water levels in the surrounding area and can be accessed using various platforms. This research likely to be successful because previous studies conducted the same components in this study.*

***Keywords:*** *IoT, Flutter, smart home system; formatting; style; styling; insert*

## I. PENDAHULUAN

1.1 Latar Belakamg Masalah

Pengembangan yang masif di bidang teknologi menciptakan beberapa teknologi yang baru. Salah satunya adalah Internet of Thing atau yang sering kita sebut dengan IoT. Konsep Internet of Thing pertama kali muncul dalam pidato Peter T.Lewis, kepada Congressional Black Caucus Foundation 15th Annual Legislative Weekend di Washington, D.C, diterbitkan pada September 1986. Menurut Lewis, "The Internet of Things, or IoT, is the integration of people, processes and technology with connectable devices and sensors to enable remote monitoring, status, manipulation and evaluation of trends of such devices."

Tujuan utama IoT adalah menghadirkan kenyamanan dan memudahkan dalam kehidupan manusia. IoT juga dapat diimplementasikan di berbagai segment. The extensive set of applications for IoT devices (Vongsingthong & Smanchat, 2014) is often divided into consumer, commercial, industrial, and infrastructure spaces (Perera, Liu, & Jayawardena, 2015)

Monitoring ketinggian air dalam sebuah rumah atau bangunan sangat penting untuk mencegah terjadinya banjir, kebocoran atau kerusakan pada sistem pipa air, serta membantu pemilik rumah untuk mengelola penggunaan air dengan lebih efektif dan efisien.

Dengan adanya teknologi smart home, monitoring ketinggian air dapat dilakukan secara otomatis dan real-time dengan menggunakan perangkat seperti ESP8266, MQTT, dan Flutter. Dengan memanfaatkan teknologi ini, pemilik rumah dapat memantau ketinggian air di dalam tangki penyimpanan air atau dalam saluran pembuangan air secara real-time, sehingga mereka dapat dengan cepat mengambil tindakan jika terjadi perubahan atau keadaan darurat. Data ketinggian air dapat ditampilkan dalam bentuk grafik atau angka di berbagai platform menggunakan framework Flutter. Sehingga pemilik rumah dapat dengan mudah memantau dan mengelola penggunaan air di rumah mereka

1.2 Identifikasi Masalah

Masyarakat umum masih banyak yang belum memanfaatkan teknologi untuk mengetahui ketinggian air secara real time. Hal ini mempengaruhi ketidaktahuan diketahuinya kapan secara pasti air sudah mencapai batas ketinggian tertentu. Oleh karena itu penelitian ini merancang sebuah perangkat IoT dengan menggunakan jaringan nirkabel berbasis sensor pendeteksi ketinggian air yang bekerja secara otomatis dengan membaca ketinggian air dengan menggunakan sensor ultrasonik yang diterapkan untuk smart home. Data yang didapatkan kemudian dikomunikasikan melalui MQTT dan akan diterima oleh aplikasi yang dibangun di atas framework Flutter. Sehingga, pengguna bisa mengakses data ketinggian air melalui 6 platform (Android, iOS, Web, Windows, MacOS, Linux).

Dengan melakukan penelitian pada sistem ini, akan memungkinkan untuk mengembangkan solusi yang lebih baik untuk masalah nyata yang dihadapi oleh masyarakat dan dapat membantu mengembangkan teknologi terbaru yang dapat diterapkan pada sistem lainnya. Hal ini dapat membantu meningkatkan efisiensi dan efektivitas pada sistem yang lebih besar.

1.3 Rumusan Masalah

Berikut adalah rumusan masalah dari pengembangan aplikasi monitoring ketinggian air menggunakan Arduino Uno, sensor ultrasonik HC-SR04, MQTT, dan Flutter:

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

1. Bagaimana cara mengimplementasikan sensor ultrasonik HC-SR04 pada Arduino Uno untuk mengukur ketinggian air dengan akurasi yang tinggi?
2. Bagaimana cara mengirim data hasil pengukuran ketinggian air dari Arduino Uno ke Flutter menggunakan protokol MQTT?
3. Bagaimana cara membuat aplikasi mobile yang dapat menampilkan hasil pengukuran ketinggian air secara real-time dan dapat diakses dari jarak jauh?.

1.4 Tujuan dan Manfaat Penelitian

Tujuan dari penelitian ini adalah untuk merancang dan mengembangkan sebuah sistem monitoring ketinggian air yang dapat diimplementasikan pada smart home, sehingga pengguna dapat memantau ketinggian air pada area tertentu secara real-time dan akurat melalui berbagai platform yang terhubung dengan sistem tersebut. Adapun manfaat dari penelitian ini antara lain:

1. Memberikan kemudahan bagi pengguna untuk memantau ketinggian air pada area tertentu di dalam smart home.
2. Meningkatkan keamanan dan kenyamanan bagi pengguna smart home dengan memberikan informasi yang akurat dan real-time terkait ketinggian air pada area sekitar.
3. Memberikan solusi bagi pengguna smart home yang membutuhkan sistem monitoring ketinggian air dengan biaya yang terjangkau dan mudah dalam pengoperasiannya.
4. Mengoptimalkan penggunaan teknologi IoT pada smart home, sehingga dapat memberikan informasi yang lebih lengkap dan akurat bagi pengguna.
5. Memberikan pengalaman dalam mengembangkan aplikasi mobile dengan menggunakan teknologi Flutter dan protokol MQTT.

## II. METODE DAN MATERI

2.2 Software Pendukung

Software adalah program komputer dan dokumentasi terkait. Produk perangkat lunak mungkin dikembangkan untuk customer tertentu atau mungkin untuk pasar umum (Sommerville, 2011). Software pendukung adalah program atau aplikasi yang digunakan sebagai alat bantu dalam melaksanakan tugas tertentu pada suatu proyek atau pekerjaan Software pendukung dapat digunakan untuk membantu dalam berbagai aktivitas seperti analisis, pengolahan data, perancangan, pengembangan, dan pengujian aplikasi.

2.2.1 Arduino IDE

Arduino IDE adalah Integrated Development Environment (IDE) yang digunakan untuk mengembangkan program pada mikrokontroler Arduino. Arduino IDE menyediakan berbagai fitur seperti editor kode, kompiler, dan perangkat lunak pengunggah (uploader) program ke mikrokontroler Arduino. Dalam Arduino IDE, kita dapat menulis, mengedit, dan menguji kode pada mikrokontroler Arduino dengan mudah.

Kelebihan dari Arduino IDE adalah sederhana, mudah digunakan, dan gratis. Arduino IDE sangat cocok bagi pemula yang ingin mempelajari program mikrokontroler dan mengembangkan proyek-proyek elektronik. Selain itu, Arduino IDE memiliki komunitas yang besar dan dukungan yang baik, sehingga memudahkan para pengembang untuk mencari bantuan dan memecahkan masalah yang muncul selama pengembangan proyek.

Kita perlu memakai Arduino IDE karena IDE ini menyederhanakan proses pengembangan program pada mikrokontroler Arduino. Selain itu, dengan menggunakan Arduino IDE, kita dapat memanfaatkan banyak library yang tersedia untuk mempercepat pengembangan proyek. Selain itu, Arduino IDE juga memungkinkan kita untuk menggunakan bahasa pemrograman yang mudah dipahami, sehingga dapat diakses oleh banyak orang tanpa memerlukan latar belakang teknis yang mendalam.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

2.2.2 Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu (Integrated Development Environment/IDE) resmi untuk pengembangan aplikasi Android, yang didasarkan pada IntelliJ IDEA. Kelebihan dari Android Studio adalah mudah digunakan, dukungan yang baik dari Google, serta menyediakan banyak fitur yang membantu pengembangan aplikasi Android seperti debugging, pengujian, dan integrasi dengan layanan Google.

2.2.3 Visual Studio Code

Visual Studio Code adalah sebuah text editor yang sangat populer yang dikembangkan oleh Microsoft dan didukung oleh komunitas pengembang yang besar. Visual Studio Code memiliki fitur lengkap seperti syntax highlighting, auto completion, debugging, dan banyak lagi yang dapat membantu dalam pengembangan aplikasi.

Untuk pengembangan aplikasi MQTT dengan menggunakan library MQTTnet, Visual Studio Code memiliki beberapa kelebihan seperti:

a. Dukungan yang kuat untuk bahasa pemrograman C# dan .NET, sehingga dapat memudahkan dalam pengembangan aplikasi MQTT dengan menggunakan library MQTTnet yang ditulis dengan bahasa C#.
b. Kemampuan untuk memudahkan pengembangan dengan dukungan untuk debugging, source control, dan pengembangan kolaboratif dengan fitur seperti Live Share.
c. Plugin yang banyak tersedia untuk memperluas fitur dan fungsionalitas Visual Studio Code, sehingga memungkinkan untuk menyesuaikan pengalaman pengembangan sesuai dengan kebutuhan. Visual Studio Code dapat membantu dalam pengembangan aplikasi MQTT dengan menggunakan library MQTTnet dengan lebih mudah dan efisien.

2.2.4 Flutter

Flutter merupakan sebuah SDK untuk pengembangan aplikasi mobile yang dikembangkan oleh Google untuk membangun a (Raharjo, 2019) aplikasi yang memiliki kinerja tinggi serta dapat dipublikasikan ke platform Android dan iOS dari codebase tunggal (Raharjo, 2019). Flutter memiliki kelebihan sebagai berikut:

a. Cross-platform: Flutter memungkinkan pengembangan aplikasi yang dapat berjalan di berbagai platform, termasuk Android, iOS, web, dan desktop, dengan menggunakan kode yang sama.
b. Fast development: Flutter memungkinkan pengembang untuk mengembangkan aplikasi dengan lebih cepat karena memiliki fitur hot reload yang memungkinkan pengembang untuk melihat perubahan dalam waktu nyata tanpa perlu melakukan kompilasi ulang.
c. Widget-based: Flutter menggunakan widget sebagai elemen dasar untuk membangun UI pada aplikasi, sehingga memungkinkan pengembang untuk membangun UI dengan lebih mudah dan fleksibel.
d. Native performance: Flutter menggunakan teknologi rendering sendiri yang disebut dengan Skia untuk menghasilkan tampilan yang halus dan responsif pada berbagai platform, sehingga aplikasi yang dibangun dengan Flutter memiliki performa yang mirip dengan aplikasi native.

Flutter menjadi salah satu framework populer yang banyak digunakan oleh pengembang untuk membangun aplikasi mobile, web, dan desktop. Flutter juga memiliki komunitas pengembang yang besar dan dokumentasi yang lengkap, sehingga memudahkan pengembang untuk belajar dan mengembangkan aplikasi dengan Flutter.

2.2.5 MQTTnet

MQTTnet adalah salah satu implementasi MQTT (Message Queuing Telemetry Transport) pada platform .NET. MQTT adalah sebuah protokol komunikasi yang ringan dan efisien untuk pengiriman pesan antar perangkat yang terhubung ke Internet of Things (IoT). MQTTnet merupakan library yang memungkinkan pengembang untuk menggunakan protokol MQTT di aplikasi .NET.

Beberapa kelebihan dari MQTTnet antara lain:

a. Ringan dan efisien: Protokol MQTT dirancang untuk pengiriman pesan

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

yang ringan dan efisien sehingga cocok digunakan pada perangkat IoT
dengan sumber daya terbatas.
b. Mendukung koneksi dengan banyak broker: MQTTnet mendukung
koneksi dengan banyak broker MQTT, sehingga memungkinkan pengembang untuk mengintegrasikan aplikasi dengan berbagai jenis perangkat IoT yang terhubung dengan broker MQTT.
c. Mendukung banyak platform: MQTTnet dapat digunakan pada berbagai platform .NET, termasuk Windows, Linux, dan macOS.
d. Terdapat banyak fitur: MQTTnet menyediakan banyak fitur, seperti publish/subscribe, retained messages, Quality of Service (QoS), dan masih banyak lagi. Hal ini memudahkan pengembang untuk membangun aplikasi IoT yang kompleks.

Karena kelebihan-kelebihan tersebut, MQTTnet merupakan pilihan yang baik untuk digunakan pada aplikasi IoT yang dibangun dengan platform .NET

2.3 Hardware Pendukung

Hardware pendukung adalah perangkat keras atau komponen fisik yang digunakan dalam penelitian atau pengembangan aplikasi sebagai bagian dari implementasi sistem. Hardware pendukung ini memiliki peran penting dalam menentukan keberhasilan dari sistem yang sedang dikembangkan, sehingga pemilihan dan penggunaannya perlu dilakukan dengan cermat dan tepat agar dapat mencapai tujuan penelitian yang diinginkan. Selain itu, hardware pendukung juga menjadi salah satu aspek yang perlu dipertimbangkan dalam perencanaan anggaran dan waktu dalam penelitian atau pengembangan aplikasi.

2.3.1 NodeMCU

NodeMCU ESP866 adalah modul WiFi yang dapat digunakan untuk mengakses jaringan WiFi, mengirim dan menerima data melalui jaringan, dan mengontrol perangkat melalui koneksi internet. Kelebihan dari modul WiFi ESP8266 adalah ukurannya yang kecil, konsumsi daya yang rendah, kemampuan terhubung ke jaringan WiFi, dan kemampuan untuk memproses data dan menjalankan program di dalam modul itu sendiri.

Kita perlu memakai modul WiFi ESP8266 karena banyak aplikasi IoT yang membutuhkan koneksi internet dan pengiriman data secara nirkabel. Dengan menggunakan modul WiFi ESP8266, kita dapat mengakses jaringan WiFi dan mengirimkan data melalui internet dengan mudah. Modul ini juga banyak digunakan dalam proyek-proyek DIY atau proyek-proyek kecil yang memerlukan koneksi internet, namun tidak memerlukan daya komputasi yang tinggi.

2.3.2 Sensor Ultrasonik HC-SR04

Sensor Ultrasonik HC-SR04 adalah sensor jarak yang menggunakan gelombang ultrasonik untuk mengukur jarak suatu benda dari sensor tersebut. Sensor ini terdiri dari dua komponen utama, yaitu pemancar (transmitter) dan penerima (receiver), yang bekerja secara bersamaan untuk mengukur jarak dengan memantulkan gelombang suara dari objek yang diukur.

Kelebihan dari Sensor Ultrasonik HC-SR04 antara lain adalah:
a. Akurasi pengukuran: Sensor Ultrasonik HC-SR04 memiliki akurasi pengukuran yang baik dan mampu mengukur jarak dari beberapa
centimeter hingga beberapa meter dengan akurasi yang tinggi.
b. Harga terjangkau: Sensor Ultrasonik HC-SR04 relatif murah dibandingkan
dengan sensor jarak lainnya.
c. Mudah digunakan: Sensor Ultrasonik HC-SR04 mudah digunakan dan
dihubungkan dengan mikrokontroler karena hanya memerlukan sedikit
komponen tambahan.
d. Tidak terpengaruh oleh cahaya: Sensor Ultrasonik HC-SR04 dapat
digunakan di lingkungan dengan cahaya yang rendah atau bahkan gelap,
karena sensor ini tidak terpengaruh oleh cahaya.

Kita perlu memakai Sensor Ultrasonik HC-SR04 karena sensor ini dapat membantu kita mengukur jarak dengan akurasi yang tinggi, sehingga cocok digunakan dalam berbagai proyek seperti pengukuran jarak

**DOI:** 10.52362/jisicom.v7i2.1197

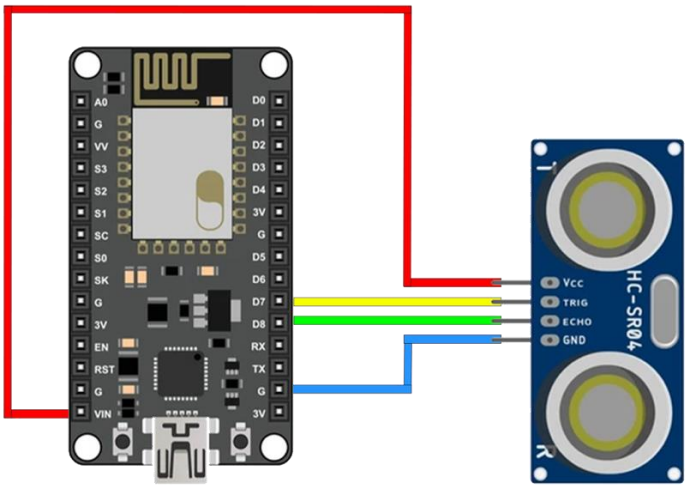
e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

kendaraan, pengukuran tinggi air, dan lain sebagainya. Selain itu, Sensor Ultrasonik HC-SR04 juga mudah digunakan dan dihubungkan dengan berbagai mikrokontroler.

2.4 Cara Kerja Sistem

Sensor NodeMCU ESP8266 dan HC-SR04 dapat digabungkan untuk membuat sistem pemantauan jarak yang dapat dihubungkan ke server MQTT. NodeMCU ESP8266 terhubung ke sensor HC-SR04 menggunakan kabel jumper, dan mengirimkan pengukuran jarak ke server MQTT menggunakan pustaka PubSubClient. Server MQTT menerima pengukuran jarak dan menerbitkannya ke klien yang berlangganan.

Untuk merancang sistem seperti itu, kita perlu menghubungkan sensor HC-SR04 ke NodeMCU ESP8266 menggunakan kabel jumper. Pin VCC dari HC-SR04 dihubungkan ke pin VIN dari NodeMCU ESP8266, dan pin GND harus dihubungkan ke pin GND. Pin TRIG dan ECHO harus dihubungkan ke pin GPIO dari NodeMCU ESP8266.




Gambar 2.1 Rancangan NodeMCU ESP8266 dan HC-SR04

Setelah koneksi perangkat keras dibuat, kita dapat merancang metode pengambilan data. Metode pengambilan data dengan NodeMCU ESP8266, MQTTnet dan Flutter melibatkan beberapa komponen dan langkah-langkah berikut:

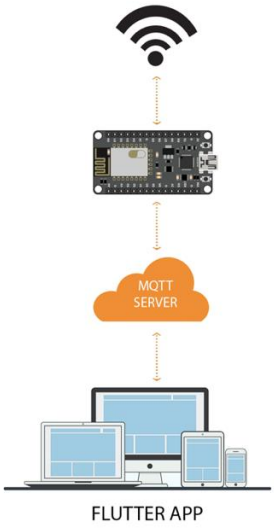
1. Pertama, dibutuhkan sebuah mikrokontroller ESP8266 NodeMCU
2. Selanjutnya, diperlukan sebuah server MQTT. Server ini berfungsi sebagai pusat komunikasi antara ESP8266 dan Flutter, sehingga data yang dikirim dari ESP8266 dapat diterima oleh aplikasi Flutter.
3. ESP8266 perlu terhubung ke jaringan WiFi agar dapat mengirim data ke server MQTT. Koneksi ini diatur melalui kode program ESP8266.
4. PubSubClient. PubSubClient adalah library yang digunakan pada ESP8266 untuk mengirim dan menerima pesan MQTT. Library ini menyederhanakan proses komunikasi dengan server MQTT.
5. MQTT Package pada Flutter. Flutter menyediakan package untuk menghubungkan aplikasi dengan server MQTT, seperti mqtt_client dan mqtt5_client. Package ini memungkinkan aplikasi Flutter untuk menerima data yang dikirimkan oleh ESP8266.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

6. UI Flutter. Terakhir, data yang diterima dari server MQTT dapat ditampilkan pada UI Flutter, seperti menggunakan widget Text atau Chart.

Dengan metode ini, data dapat dikirimkan dari ESP8266 ke server MQTT dan diterima oleh aplikasi Flutter, sehingga pengguna dapat memantau data secara real-time.




Gambar 2.2 Rancangan NodeMCU ESP8266 dan HC-SR04

Setelah merancang metode pengambilan data, selanjutnya menulis kode untuk membaca pengukuran jarak dari sensor HC-SR04 dan mengirimkannya ke server MQTT menggunakan pustaka PubSubClient. Define dan include adalah dua kata kunci dalam bahasa pemrograman C++ yang digunakan di lingkungan Arduino IDE. Define digunakan untuk membuat konstanta atau alias dalam kode program. Saat program dijalankan, nilai konstanta akan tetap sama dan tidak dapat diubah selama program berjalan. Include digunakan untuk memasukkan kode program dari file header atau library ke dalam kode program utama. Contoh penggunaan define pada gambar 3.2 adalah konstanta tigPin diberi nilai D7 dan konstanta echoPin diberi nilai D8. Penggunaan include pada gambar 3.2 menyertakan ESP8266Wifi.h dan PubSubClient.h yang berarti file tersebut dimasukkan ke dalam program. File header ini berisi definisi dari semua fungsi dan variabel yang digunakan di lingkungan Arduino IDE.

WiFiClient dan PubSubClient adalah library atau pustaka yang tersedia di Arduino IDE untuk membantu mengakses dan mengontrol protokol jaringan WiFi dan protokol publish-subscribe (pub-sub) dalam sebuah sistem IoT.

WiFiClient adalah library yang digunakan untuk membantu koneksi WiFi pada board Arduino, seperti NodeMCU ESP8266. Library ini memberikan fungsi-fungsi yang memudahkan penggunaan koneksi WiFi seperti melakukan koneksi ke sebuah SSID, mengambil alamat IP, melakukan koneksi ke server, dan sebagainya.

PubSubClient adalah library yang digunakan untuk memudahkan implementasi protokol publish-subscribe (pub-sub) di dalam program yang dijalankan pada board Arduino. Protokol pub-sub merupakan cara untuk melakukan komunikasi antara beberapa perangkat di dalam jaringan, dimana perangkat dapat berlangganan (subscribe) ke topik tertentu dan menerima pesan yang dikirim ke topik tersebut, atau mengirim pesan (publish) ke topik yang telah ditentukan.

Dalam penggunaan PubSubClient, library ini memberikan fungsi-fungsi yang memudahkan dalam melakukan koneksi ke broker MQTT dan mengirim atau menerima pesan dari topik-topik yang telah ditentukan.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

Library ini telah terintegrasi dengan WiFiClient sehingga dapat digunakan bersama-sama dengan library tersebut untuk membuat sebuah aplikasi IoT yang dapat terhubung ke jaringan WiFi dan broker MQTT.

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

#define trigPin D7
#define echoPin D8

const char* ssid = "Hidden Network 1";
const char* password = "Hidden0!";
const char* mqtt_server = "192.168.16.31";
const char* mqtt_topic = "distance";

WiFiClient espClient;
PubSubClient client(espClient);
```

Gambar 3.3 Initial Value NodeMCU ESP8266

Pada Arduino IDE, fungsi setup() digunakan untuk menyalakan atau menginisialisasi variabel dan perangkat keras yang akan digunakan dalam program.

Fungsi Serial.begin() digunakan untuk memulai koneksi serial pada baud rate tertentu. Baud rate yang harus digunakan sama dengan baud rate yang digunakan pada perangkat yang terhubung ke Arduino.

Fungsi pinMode() digunakan untuk menentukan apakah sebuah pin pada board Arduino digunakan sebagai input atau output. Pada umumnya, parameter pertama adalah nomor pin, dan parameter kedua adalah tipe mode (INPUT atau OUTPUT).

Pada Gambar 3.3 dijelaskan bahwa di dalam function setup terdapat function Serial.begin(9600) yang mengartikan baud rate 9600 pada perangkat yang teehubung ke Arduino, function pinMode trigPin sebagai output dan echoPin sebagai input. Function setup_wifi() adalah fungsi yang digunakan untuk menghubungkan ESP8266 ke jaringan WiFi. Fungsi ini akan memulai koneksi WiFi dengan menggunakan nama jaringan (SSID) dan kata sandi (password) yang telah ditentukan, dan menunggu hingga koneksi terhubung. Function client.setServer() adalah fungsi yang digunakan untuk mengatur alamat server MQTT yang akan digunakan oleh ESP8266 untuk mengirim data. Fungsi ini membutuhkan parameter berupa alamat server MQTT dalam bentuk string. Setelah diset, ESP8266 dapat terhubung ke server tersebut dan mengirimkan pesan MQTT ke topik yang telah ditentukan.

```
void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  setup_wifi();
  client.setServer(mqtt_server, 1883); // Set the MQTT broker address and port
}
```

Gambar 3.4 Function setup NodeMCU ESP8266

Fungsi loop() pada Arduino IDE adalah bagian dari program yang dijalankan secara terus-menerus setelah fungsi setup() selesai dieksekusi. Fungsi ini berisi kode program yang akan diulang secara terus-menerus selama board Arduino berjalan, kecuali jika board tersebut dimatikan atau program dihentikan.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

Pada umumnya, fungsi loop() digunakan untuk mengontrol perangkat elektronik atau sensor, membaca atau mengirim data, atau melakukan operasi berulang. Kode program yang terdapat pada fungsi loop() biasanya berisi looping, yaitu perulangan perintah-perintah tertentu dengan menggunakan pernyataan seperti for, while, atau do-while.

Dalam aplikasi IoT yang menggunakan protokol MQTT, fungsi loop() juga digunakan untuk mengecek koneksi dengan broker MQTT, memperbarui status perangkat, membaca data sensor, dan mengirimkan data ke broker.

Beberapa fungsi yang terdapat di dalam fungsi loop() pada gambar 3.4 adalah

1. digitalWrite() digunakan untuk mengirimkan sinyal HIGH atau LOW ke suatu pin pada mikrokontroler, dalam hal ini digunakan untuk mengirimkan sinyal ke pin trigPin.
2. delayMicroseconds() digunakan untuk memberikan jeda (delay) dalam mikrodetik (µs), dalam hal ini digunakan untuk memberikan delay selama 2 µs setelah mengirimkan sinyal HIGH ke pin trigPin.
3. pulseIn() digunakan untuk mengukur durasi pulsa pada suatu pin pada mikrokontroler, dalam hal ini digunakan untuk mengukur durasi pulsa yang diterima pada pin echoPin.
4. Serial.print() digunakan untuk mencetak (print) teks atau nilai pada monitor serial, dalam hal ini digunakan untuk mencetak nilai jarak yang diukur menggunakan sensor ultrasonik.
5. client.publish() digunakan untuk mempublish pesan ke broker MQTT, dalam hal ini digunakan untuk mempublish nilai jarak ke topik "distance".
6. delay() digunakan untuk memberikan jeda (delay) dalam milidetik (ms), dalam hal ini digunakan untuk memberikan delay selama 500 ms sebelum membaca kembali jarak

```
void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration / 2) / 29.1; // Divide by 29.1 or multiply by 0.0343 for distance in cm
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");

  // Send distance to MQTT broker
  if (!client.connected()) {
    reconnect();
  }
  client.publish(mqtt_topic, String(distance).c_str());

  delay(500); // Wait for 500 milliseconds before taking the next measurement
}
```

Gambar 3.5 Function loop NodeMCU ESP8266

Function setup_wifi() adalah fungsi yang digunakan untuk menghubungkan ESP8266 ke jaringan WiFi. Fungsi ini akan memulai koneksi WiFi dengan menggunakan nama jaringan (SSID) dan kata sandi (password) yang telah ditentukan, dan menunggu hingga koneksi terhubung dengan status WL_CONNECTED. Setelah terhubung, fungsi ini akan menampilkan alamat IP yang digunakan.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

```
void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

Gambar 3.6 Function Setup Wifi NodeMCU ESP8266

Fungsi reconnect() digunakan untuk melakukan koneksi kembali (reconnect) pada MQTT server apabila koneksi terputus atau gagal terhubung. Fungsi ini akan dijalankan di dalam loop utama untuk memastikan koneksi selalu terjaga. Pada fungsi reconnect(), pertama-tama dicek terlebih dahulu apakah klien (client) sudah terhubung dengan server atau belum menggunakan fungsi client.connected(). Jika belum terhubung, maka klien akan mencoba terhubung kembali ke server menggunakan fungsi client.connect().

Jika koneksi berhasil terhubung, maka fungsi akan mengeluarkan pesan "connected". Namun, jika koneksi gagal, maka akan ditampilkan pesan "failed" beserta kode status kesalahan (RC) dan akan diulang kembali dalam jangka waktu 5 detik menggunakan fungsi delay().

```
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

Gambar 3.7 Function reconnect wifi NodeMCU ESP8266

Di sisi perangkat lunak, kita perlu menyiapkan server MQTT dan membuat topik tempat pengukuran jarak akan dipublikasikan. Anda dapat menggunakan server MQTT berbasis cloud atau menyiapkan broker MQTT lokal Anda sendiri. Kemudian, Anda perlu menulis kode untuk menghubungkan NodeMCU ESP8266 ke server MQTT dan mempublikasikan pengukuran jarak ke topik.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

```
using MQTTnet;
using MQTTnet.Server;
using System.Text;

0 references
class Program
{
    0 references
    static async Task Main(string[] args)
    {
        var options = new MqttServerOptionsBuilder()
            .WithConnectionBacklog(100)
            .WithDefaultEndpointPort(1883)
            .WithDefaultEndpoint()
            .Build();

        var mqttServer = new MqttFactory().CreateMqttServer();
        mqttServer.UseApplicationMessageReceivedHandler(e =>
        {
            Console.WriteLine($"Received message on topic '{e.ApplicationMessage.Topic}': {Encoding.UTF8.GetString(e.ApplicationMessage.Payload)}");
        });

        await mqttServer.StartAsync(options);

        Console.WriteLine("MQTT server started. Press any key to exit.");
        Console.ReadLine();

        await mqttServer.StopAsync();
    }
}
```

Gambar 3.8 MQTT Server Code

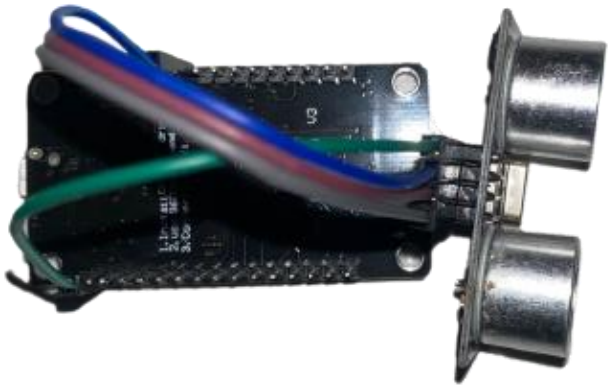
Untuk memvisualisasikan pengukuran jarak, kita dapat membuat aplikasi Flutter yang subscribe topik MQTT dan menampilkan pengukuran jarak secara real-time. Untuk menghubungkan aplikasi Flutter ke server MQTT, kita dapat menggunakan paket mqtt_client. Paket ini menyediakan klien MQTT yang dapat digunakan untuk subscribe topik MQTT dan menerima pesan. Setelah aplikasi terhubung ke server MQTT, kita dapat menggunakan widget Flutter untuk menampilkan pengukuran jarak.

Dengan mengikuti langkah-langkah tersebut dapat membuat sebuah perangkat monitoring ketinggian air yang terhubung dengan jaringan MQTT dan dapat diakses melalui aplikasi Flutter.

## III. PEMBAHASA DAN HASIL

### 3.1 Hasil Penelitian

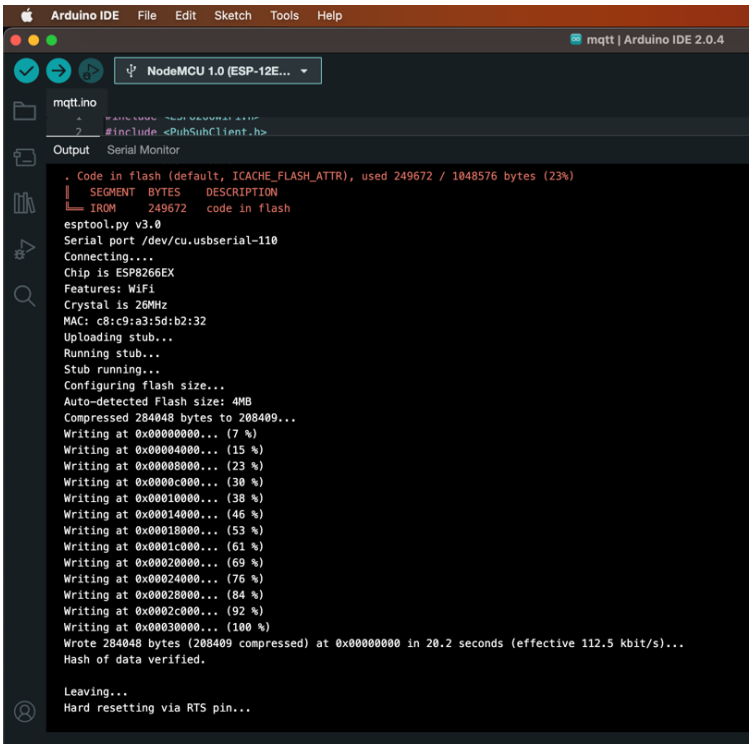
Hasil tampilan fisik dari merangkai rancangan pada gambar 3.1 dapat dilihat pada gambar 4.1. Dari gambar tersebut dapat dilihat bahwa NodeMCU ESP8266 dapat terhubung langsung dengan Sensor ultrasonic HC-SR04 menggunakan kabel female to female. Sebelum menggunakan perangkat NodeMCU ESP8266, kita harus mengupload konfigurasi yang sudah dibuat di Arduino Uno. Bisa lihat gambar 4.2.



Gambar 4.1 NodeMCU ESP8266 + HC-SR04

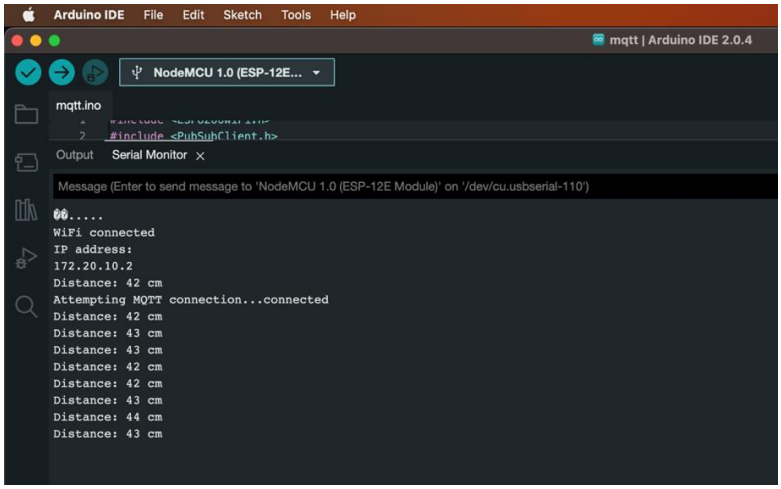
**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com



Gambar 4.2 Upload Config Arduino IDE

Setelah berhasil mengunggah konfigurasi dari Arduino IDE ke NodeMCU ESP8266, kita bisa melihat hasil running dari konfigurasi di Serial Monitor yang terdapat di Arduino IDE. Di Gambar 4.3 terlihat bahwa NodeMCU ESP8266 berhasil menghubungkan ke perangkat wifi. Selanjutnya berhasil menghubungkan ke MQTT Server lalu menampilkan hasil dari sensor ultrasonik HC-SR04 secara berkala.



Gambar 4.3 Result Serial Monitor Arduino IDE

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

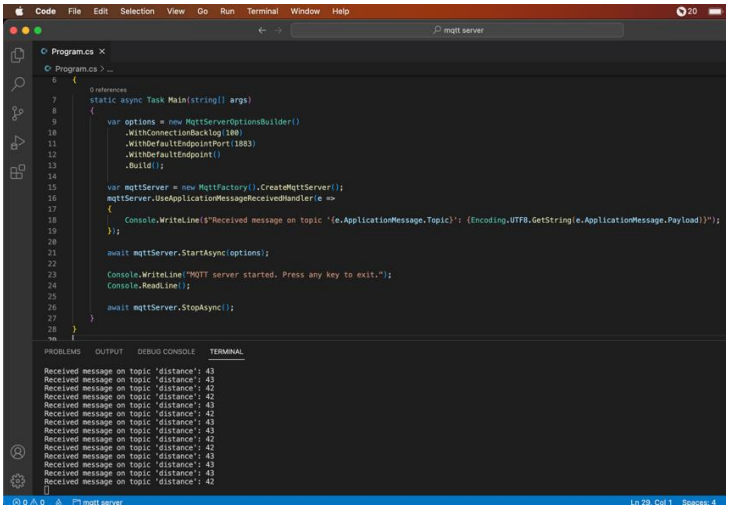
Perangkat NodeMCU ESP8266 membutuhkan daya minimal 3.3V. oleh karena itu kita bisa menggunakan batterai ataupun sumber listrik lainnya yang memiliki sumber daya lebih dari 3.3V. Disini saya memprototype perangkat NodeMCU ESP8266 menggunakan Powerbank berkapasitas 10000mAH 3.7V. Gambar 4.4 merupakan hasil rangkaian NodeMCU ESP8266 dan HC-SR04 yang terhubung ke Powerbank 3.7V

Gambar 4.4 NodeMCU ESP8266 + HC-SR04 + Powerbank

Setelah perangkat tersambung daya, maka NodeMCU ESP8266 akan mencoba untuk terhubung ke internet wifi yang sudah dikonfigurasi sebelumnya. Selanjutnya NodeMCU ESP8266 akan mencoba untuk terhubung ke MQTT Server yang sudah dikonfigurasi sebelumnya. Hasilnya perangkat NodeMCU ESP8266 berhasil terhubung ke internet wifi lalu terhubung ke MQTT Server. Hasil pada gambar 4.5 menunjukkan bahwa perangkat tersebut berhasil mengirimkan data ke MQTT Server.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com



Gambar 4.5 Hasil MQTT

Untuk menampilkan hasil data yang sudah dikirim ke MQTT Server, kita memerlukan MQTT Client dan beberapa paket lainnya seperti rxdart dan get_it. MQTT Client, RxDart, dan Get It merupakan beberapa package atau library yang sering digunakan dalam pengembangan aplikasi berbasis Flutter yang terhubung dengan MQTT broker.

MQTT Client adalah sebuah package yang digunakan untuk menghubungkan aplikasi Flutter dengan MQTT broker. Package ini menyediakan fitur publish dan subscribe yang memungkinkan pengiriman dan penerimaan pesan dari MQTT broker.

RxDart adalah package yang mengimplementasikan reactive programming pada aplikasi Flutter. Dengan menggunakan RxDart, pengembang dapat memanfaatkan Observable, Stream, dan banyak operator lainnya untuk memanage state aplikasi secara efektif.

Get It adalah package yang menyediakan service locator untuk mengatur dependensi dalam aplikasi Flutter. Dengan Get It, pengembang dapat menginisialisasi dan mengakses instance dari sebuah class secara global tanpa perlu menggunakan konstruktor.

Dalam pengembangan aplikasi yang terhubung dengan MQTT broker, ketiga package ini dapat digunakan untuk mempermudah implementasi fitur-fitur yang dibutuhkan, seperti koneksi ke broker, handling message, serta manajemen state aplikasi. Di gambar 4.6 dijelaskan version paket dan framework yang digunakan di dalam penelitian

```
environment:
  sdk: '>=2.18.6 <3.0.0'

dependencies:
  flutter:
    sdk: flutter

  cupertino_icons: ^1.0.2
  mqtt_client: 9.7.4
  rxdart: 0.27.7
  get_it: 7.2.0
```
Gambar 4.6 Version Paket

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

Sebelum menghubungkan Flutter dengan MQTT Server, kita perlu menyiapkan tampilan agar user dapat dengan mudah melihat dan membacanya. Untuk membuat tampilan di Flutter saya mengimplementasi widget AnimatedBuilder, ClipPath, Painter dan CustomPaint supaya lebih menarik.

AnimatedBuilder adalah sebuah widget di Flutter yang digunakan untuk membuat animasi dengan menggunakan controller. Controller tersebut akan menentukan bagaimana animasi berjalan dan berakhir. AnimatedBuilder akan membangun widget tree sesuai dengan perubahan dari controller.

ClipPath adalah widget di Flutter yang digunakan untuk memotong atau meng-clip sebuah widget dengan bentuk tertentu. ClipPath biasanya digunakan untuk membuat shape yang tidak biasa seperti lingkaran, segitiga, atau bentuk-bentuk lainnya.

CustomPaint adalah widget di Flutter yang digunakan untuk membuat custom painting. CustomPaint memberikan kontrol yang lebih besar pada proses rendering dengan memungkinkan kita untuk menggambar sesuatu secara langsung pada canvas.

```
return AnimatedBuilder(
  animation: CurvedAnimation(
    parent: _animationController,
    curve: Curves.easeInOut,
  ),  // CurvedAnimation
  builder: (context, child) => ClipPath(
    clipper: _WaveClipper(
      animationValue: _animationController.value,
      value: widget.value,
      direction: widget.direction,
    ),  // _WaveClipper
    child: Container(
      color: widget.color,
    ),  // Container
  ),  // ClipPath
);  // AnimatedBuilder
```

Gambar 4.6 Animated Builder

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

```
return ClipPath(
  clipper: _LinearClipper(
    radius: widget.borderRadius,
  ),  // _LinearClipper
  child: CustomPaint(
    painter: _LinearPainter(
      color: widget._getBackgroundColor(context),
      radius: widget.borderRadius ?? 0,
    ),  // _LinearPainter
    foregroundPainter: _LinearBorderPainter(
      color: widget.borderColor ?? Colors.black,
      width: widget.borderWidth ?? 0,
      radius: widget.borderRadius ?? 0,
    ),  // _LinearBorderPainter
    child: Stack(
      children: <Widget>[
        if (widget.center != null) Center(child: widget.center),
        Wave(
          value: widget.value,
          color: widget._getValueColor(context),
          direction: widget.direction,
        ),  // Wave
      ],  // <Widget>[]
    ),  // Stack
  ),  // CustomPaint
);  // ClipPath
```

Gambar 4.7 ClipPath dan CustomPaint

Setelah selesai menyiapkan tampilan, kita harus mengkonfigurasi MQTT Client untuk menerima data dari MQTT Server. Di penelitian ini menggunakan package mqtt_client.

Pada Gambar 4.8 terdapat beberapa function yang digunakan dalam konfigurasi mqtt client. Berikut diantaranya:

1. MQTTServerClient: Merupakan kelas pada package mqtt_client yang digunakan untuk melakukan koneksi ke server MQTT dan melakukan operasi seperti publish dan subscribe.
2. Logging: Logging digunakan untuk memudahkan proses debugging pada aplikasi. Dengan menggunakan logging, kita bisa melihat pesan-pesan yang di-generate oleh aplikasi pada konsol.
3. Connect: Method connect pada MQTTServerClient digunakan untuk melakukan koneksi ke server MQTT dengan parameter berupa alamat server, port, dan informasi kredensial (username dan password).
4. Disconnect: Method disconnect pada MQTTServerClient digunakan untuk memutus koneksi dengan server MQTT.
5. Subscribe: Method subscribe pada MQTTServerClient digunakan untuk melakukan subscribe ke topik tertentu pada server MQTT.
6. MQTTPublishMessage: Kelas ini pada package mqtt_client digunakan untuk mengirim pesan ke server MQTT. Pesan yang dikirim bisa berupa teks atau biner.
7. Async Sleep: Fungsi sleep pada Dart digunakan untuk memberikan jeda pada eksekusi kode. Fungsi async sleep pada MQTTClient digunakan untuk memberikan jeda pada proses async (seperti koneksi ke server MQTT).

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

```
final client = MqttServerClient('172.20.10.3', '');
var dataMqtt = BehaviorSubject<String>();

setup() async {
  client.logging(on: true);
  client.setProtocolV311();
  client.keepAlivePeriod = 20;
  client.connectTimeoutPeriod = 2000;

  try {
    await client.connect();
  } on NoConnectionException catch (e) {
    client.disconnect();
  } on SocketException catch (e) {
    client.disconnect();
  }

  const topic = 'distance';
  client.subscribe(topic, MqttQos.atMostOnce);
  client.updates!.listen((List<MqttReceivedMessage<MqttMessage?>>? c) {
    final recMess = c![0].payload as MqttPublishMessage;
    MqttPublishPayload.bytesToStringAsString(recMess.payload.message);
  });

  await MqttUtilities.asyncSleep(60);
  client.unsubscribe(topic);
  await MqttUtilities.asyncSleep(2);
  client.disconnect();
  return 0;
}
```

Gambar 4.8 MQTT Flutter Konfigurasi

Untuk menyimpan hasil data dari MQTT Server kita perlu membuat satu function sebagai function untuk mengambil data kemudian menyimpannya ke tipe data stream. Di penelitian ini penulis menggunakan BehaviourSubject dari rxdart sebagai paket untuk mengatasi tipe data stream.

BehaviorSubject adalah salah satu tipe dari Rx Subject yang didefinisikan dalam paket Rx Dart. Subject adalah sumber data yang dapat dipublikasikan (observable) dan sekaligus juga dapat menerima (observer) data. BehaviorSubject dapat menyimpan satu nilai terbaru dan mengembalikan nilai tersebut kepada subscriber yang baru bergabung. Artinya, saat sebuah subscriber baru bergabung, ia akan langsung menerima nilai terbaru yang disimpan dalam BehaviorSubject.

Perbedaan antara BehaviorSubject dengan tipe Subject yang lain seperti PublishSubject dan ReplaySubject terletak pada bagaimana data dipublikasikan. Pada PublishSubject, data yang dihasilkan hanya diteruskan ke subscriber yang bergabung setelah data tersebut dibuat (sebagai contoh, hanya data yang dihasilkan setelah subscriber bergabung yang akan dikirimkan). Pada ReplaySubject, seluruh data yang dihasilkan akan disimpan dan diteruskan ke setiap subscriber yang bergabung. Sedangkan pada BehaviorSubject, nilai terbaru akan disimpan dan diteruskan hanya kepada subscriber baru yang bergabung.

Dalam penggunaannya, BehaviorSubject cocok digunakan ketika kita membutuhkan sumber data yang dapat mempertahankan satu nilai terbaru dan mengirimkannya kepada subscriber yang baru bergabung. Contoh penggunaannya adalah ketika kita ingin membuat suatu tombol yang statusnya dapat berubah-ubah, maka BehaviorSubject dapat digunakan untuk mengirimkan status terakhir dari tombol tersebut kepada setiap

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

subscriber yang baru bergabung.Di gambar 4.9 dijelaskan di dalam function tersebut hal yang pertama dilakukan adalah mengambil payload dari Mqtt kemudian menambahkannya ke dalam dataMqtt sebagai tipe data stream.

```
getMqttData() {
  client.updates!.listen((List<MqttReceivedMessage<MqttMessage?>>? data) {
    final recMess = data![0].payload as MqttPublishMessage;
    final recMessMqtt = MqttPublishPayload.bytesToStringAsString(
      recMess.payload.message,
    );

    dataMqtt.sink.add(recMessMqtt);
  });
}
```

Gambar 4.9 Function simpan data MQTT

Setelah data yang didapatkan dari MQTT Server disimpan, kemudian perlu untuk ditampilkan. Gambar 4.10 menjelaskan beberapa widget yang digunakan, antara lain Scaffold, StreamBuilder, Custom Widget, dll.

Scaffold adalah salah satu widget di Flutter yang digunakan sebagai kerangka (template) dari sebuah tampilan aplikasi yang akan dibuat. Widget ini menyediakan beberapa properti dan method yang membantu dalam membangun tampilan aplikasi yang responsif dan mudah diatur.

Scaffold memiliki properti seperti appBar, body, floatingActionButton, dan lain-lain. Properti appBar digunakan untuk menampilkan sebuah bar yang biasanya berisi judul atau logo aplikasi, serta tombol-tombol aksi. Properti body digunakan untuk menampilkan konten utama aplikasi, seperti daftar atau form. Properti floatingActionButton digunakan untuk menampilkan sebuah tombol aksi yang melayang di atas tampilan aplikasi.

Selain properti, Scaffold juga memiliki method seperti showSnackBar, yang digunakan untuk menampilkan pesan singkat di bagian bawah layar, dan openDrawer, yang digunakan untuk membuka sebuah panel samping (drawer). Dengan menggunakan Scaffold, membangun tampilan aplikasi di Flutter menjadi lebih mudah dan efektif.

StreamBuilder adalah widget Flutter yang memungkinkan pengguna untuk membangun UI berdasarkan data yang berasal dari stream. Stream adalah objek yang menghasilkan serangkaian nilai, dan StreamBuilder memungkinkan kita untuk membangun UI yang akan diperbarui setiap kali nilai baru diterima dari stream. Dengan menggunakan StreamBuilder, kita dapat membuat aplikasi yang responsif dan dinamis.

Pada dasarnya, StreamBuilder memiliki dua argumen utama, yaitu stream dan builder. Argumen stream adalah objek stream yang akan digunakan sebagai sumber data, sedangkan argumen builder adalah fungsi yang akan dijalankan setiap kali ada data baru yang diterima dari stream. Fungsi builder harus mengembalikan widget yang akan digunakan untuk membangun UI.

Contoh penggunaan StreamBuilder adalah ketika kita ingin menampilkan daftar item yang diperbarui secara dinamis dari data yang diterima dari server. Dalam hal ini, kita dapat menggunakan stream untuk menerima data dari server dan menggunakan StreamBuilder untuk membangun daftar item secara dinamis setiap kali ada data baru yang diterima dari server.

StreamBuilder digunakan untuk melisten data stream yang didapatkan dari MQTT Server. Kemudian data tersebut ditampilkan dalam bentuk Animated yang sudah dibuat sebelumnya.

LiquidLinearProgressIndicator adalah sebuah widget di Flutter yang menampilkan indikator progres linier dalam bentuk cairan. Widget ini umumnya digunakan untuk menampilkan progres download, upload, atau proses lainnya yang membutuhkan tampilan indikator linier.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

Cara kerja LiquidLinearProgressIndicator adalah dengan menampilkan animasi berupa gelembung yang bergerak sepanjang garis progres. Gelembung ini memiliki kecepatan dan interval yang dapat dikustomisasi, sehingga dapat menampilkan tampilan progres yang sesuai dengan kebutuhan.

LiquidLinearProgressIndicator memiliki beberapa properti seperti backgroundColor, valueColor, value, borderColor, borderWidth, borderRadius, dan lain-lain yang dapat disesuaikan untuk memperoleh tampilan indikator progres yang sesuai dengan kebutuhan aplikasi yang sedang dibangun.

Widget Text pada Flutter digunakan untuk menampilkan teks atau tulisan pada antarmuka pengguna aplikasi. Text memiliki beberapa properti seperti data, style, textAlign, overflow, dan sebagainya untuk memformat tampilan teks

```
return Scaffold(
  body: StreamBuilder<String>(
    stream: getIt<MqttViewModel>().dataMqtt,
    initialData: "0",
    builder: (context, snapshot) {
      var dataMqtt = snapshot.data;
      return LiquidLinearProgressIndicator(
        value: getIt<MqttViewModel>().setHeightWater(int.parse(dataMqtt!)),
        valueColor: AlwaysStoppedAnimation(
          const Color(0xff3B6ABA).withOpacity(.8),
        ),  // AlwaysStoppedAnimation
        backgroundColor: const Color(0xff2B2C56),
        borderColor: Colors.white,
        borderWidth: 0,
        direction: Axis.vertical,
        center: Text(
          '< $dataMqtt cm',
          style: TextStyle(
            fontWeight: FontWeight.w600,
            color: Colors.white.withOpacity(.7),
          ),  // TextStyle
          textScaleFactor: 5,
        ),  // Text
      );  // LiquidLinearProgressIndicator
    }
  ),  // StreamBuilder
);  // Scaffold
```

Gambar 4.10 Tampilan Utama Aplikasi

3.2 Uji Coba

Uji terhadap sensor ultrasonik dalam hal mengukur ketinggan jarak air yaitu dengan membandingkan pengukuran jarak sebenarnya tanpa sensor secara manual menggunakan alat ukur dan menggunakan sensor ultrasonik HC-SR04. Hasil pengukuran jarak sebenarnya secara manual dan menggunakan sensor ultrasonik adalah sama.

Uji coba selanjutnya yaitu menjalankan aplikasi monitoring ketinggian air di Android, iOS, Web, MacOS, Windows, dan Linux. Flutter adalah sebuah framework untuk membuat aplikasi mobile yang dapat

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

digunakan untuk membuat aplikasi untuk platform Android, iOS, Web, MacOS, Windows, dan Linux. Dalam mengembangkan aplikasi Flutter, salah satu tahap penting adalah melakukan testing pada aplikasi.

Ada beberapa cara untuk melakukan testing aplikasi Flutter, salah satunya adalah dengan menjalankan aplikasi di emulator atau perangkat fisik. Berikut adalah cara menjalankan aplikasi flutter di Android:

1. Persiapkan perangkat Android yang akan digunakan untuk testing, pastikan USB debugging sudah diaktifkan di perangkat tersebut.
2. Hubungkan perangkat Android ke komputer dengan kabel USB.
3. Buka terminal atau command prompt, dan masuk ke direktori project Flutter yang ingin dijalankan di perangkat Android.
4. Ketikkan perintah flutter run pada terminal atau command prompt.
5. Tunggu beberapa saat hingga proses build selesai, dan aplikasi akan secara otomatis terinstal di perangkat Android yang terhubung ke komputer.
6. Setelah aplikasi terinstal, Anda dapat menjalankannya dengan membuka aplikasi di perangkat Android tersebut. Perubahan yang dilakukan pada kode sumber aplikasi dapat langsung terlihat pada aplikasi yang dijalankan di perangkat Android, tanpa perlu melakukan instalasi ulang.

Untuk membangun aplikasi Flutter pada perangkat Android, ada beberapa langkah yang perlu dilakukan. Berikut adalah langkah-langkah untuk membangun aplikasi Flutter menjadi file APK:

1. Buka terminal atau command prompt. Pindah ke direktori proyek Flutter Anda dengan menggunakan perintah "cd".
2. Jalankan perintah "flutter build apk". Proses ini akan membangun file APK untuk aplikasi Anda.
3. Setelah proses selesai, file APK dapat ditemukan di direktori "build/app/outputs/flutter-apk/". File ini akan memiliki nama yang sama dengan nama proyek Anda.

Gambar 4.12 App run on Android

**DOI:** 10.52362/jisicom.v7i2.1197

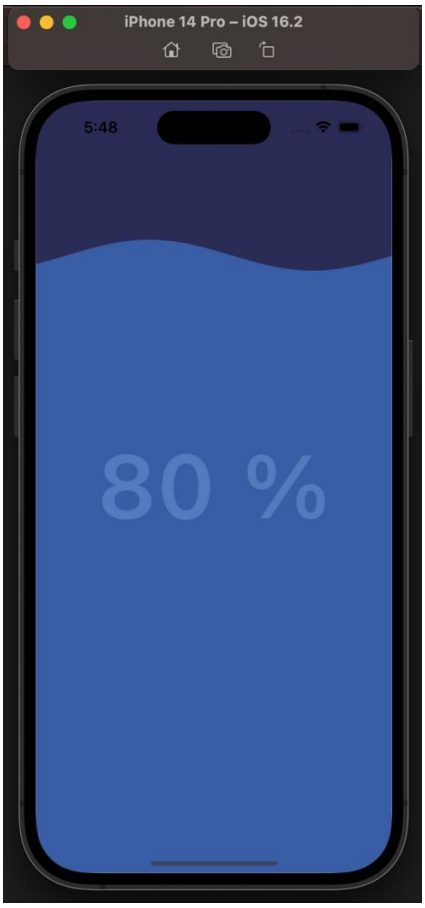
e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

Untuk menjalankan aplikasi Flutter pada iOS, kita perlu menggunakan Xcode yang hanya tersedia pada sistem operasi macOS. Berikut adalah cara untuk menjalankan aplikasi Flutter pada iOS melalui Xcode:

1. Pastikan bahwa perangkat iOS atau simulator telah terhubung dengan komputer Anda.
2. Buka terminal dan navigasi ke direktori aplikasi Flutter Anda.
3. Jalankan perintah flutter build ios untuk membangun aplikasi untuk iOS.
4. Setelah berhasil dibangun, buka file .xcworkspace yang berada di dalam direktori ios aplikasi Anda dengan menggunakan Xcode.
5. Pilih perangkat atau simulator yang ingin Anda gunakan untuk menjalankan aplikasi di Xcode.
6. Tekan tombol "Run" untuk menjalankan aplikasi pada perangkat iOS atau simulator.

Untuk melakukan build aplikasi Flutter pada iOS, Anda perlu menggunakan perangkat MacOS dan Xcode. Berikut adalah langkah-langkah yang dapat Anda ikuti:

1. Pastikan Anda telah menginstal Xcode pada perangkat MacOS Anda.
2. Buka terminal dan arahkan ke direktori proyek Flutter Anda.
3. Jalankan perintah flutter build ios pada terminal.
4. Tunggu hingga proses build selesai dan aplikasi berhasil dikompilasi.
5. Buka proyek di Xcode dengan membuka file Runner.xcworkspace yang berada di dalam direktori /ios.
6. Pastikan telah mengatur signing certificate untuk aplikasi.
7. Klik tombol Build pada Xcode untuk membangun aplikasi.
8. Setelah proses build selesai, unggah aplikasi ke App Store Connect.

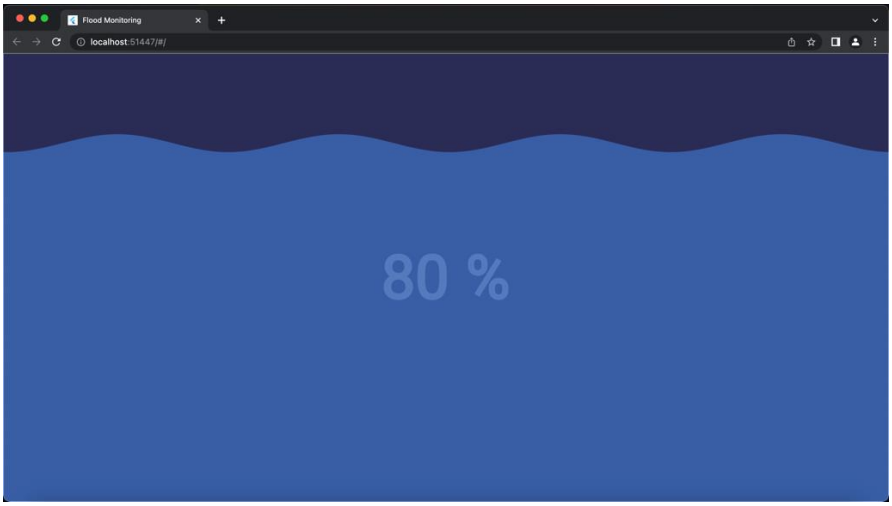



Gambar 4.13 App run on iOS

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

Flutter juga mendukung pengembangan aplikasi web. Untuk menjalankan aplikasi Flutter di web, Anda perlu melakukan beberapa konfigurasi pada proyek Flutter Anda dan menginstal paket yang diperlukan. Setelah itu, Anda dapat menjalankan aplikasi di browser menggunakan perintah flutter run -d chrome. Untuk membuild flutter web bisa menggunakan perintah flutter build web. Flutter Build Web adalah perintah yang digunakan untuk menghasilkan aplikasi Flutter dalam bentuk website atau web app. Proses build akan menghasilkan file HTML, CSS, dan Javascript yang dapat dijalankan di browser. Setelah build selesai, Anda dapat menemukan file HTML, CSS, dan Javascript yang dihasilkan dalam direktori "build/web". Kemudian, Anda dapat memuat file tersebut pada server web yang telah dipilih.



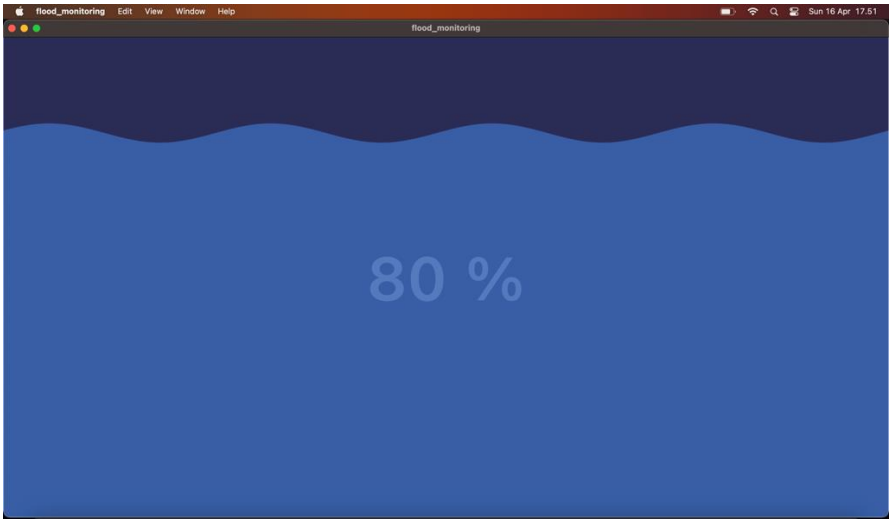

Gambar 4.14 App run on Web

Untuk menjalankan dan mem-build aplikasi desktop macOS menggunakan Flutter, ikuti langkah-langkah berikut:

1. Buka terminal dan arahkan ke direktori proyek.
2. Jalankan flutter create . untuk menghasilkan file yang diperlukan untuk platform macOS.
3. Jalankan flutter config --enable-macos-desktop untuk mengaktifkan platform macOS untuk proyek Flutter.
4. Jalankan flutter run -d macOS untuk membuat dan menjalankan aplikasi di perangkat macOS.
5. Jalankan flutter build macos untuk menghasilkan bundel aplikasi desktop macOS.
6. Setelah menjalankan flutter build macos, bundel aplikasi macOS akan dibuat di direktori build/macos/Build/Products/Release/. Bundel yang dihasilkan dapat diinstal pada perangkat macOS atau diunggah ke App Store untuk didistribusikan.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

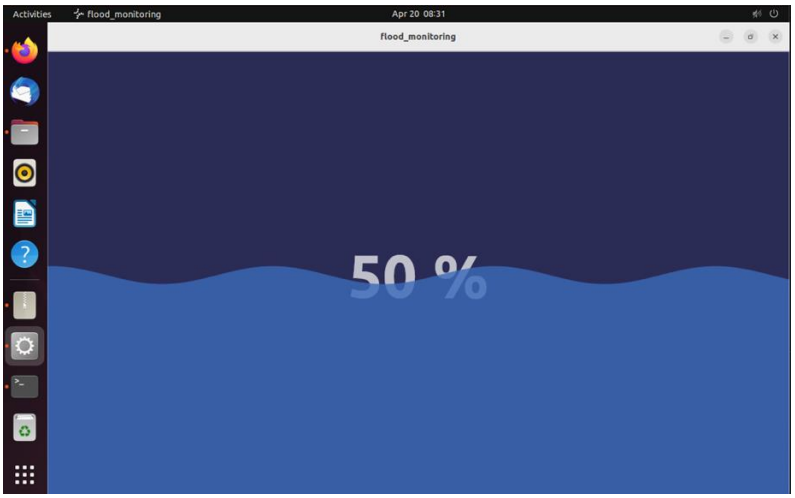



Gambar 4.15 App run on macOS

Untuk membuild aplikasi linux menggunakan flutter bisa menjalankan perintah flutter build linux. File executable dapat ditemukan di proyek di bawah build/linux/<build mode>/bundle/. Di dalam file biner yang dapat dieksekusi di direktori bundel, ada dua direktori:

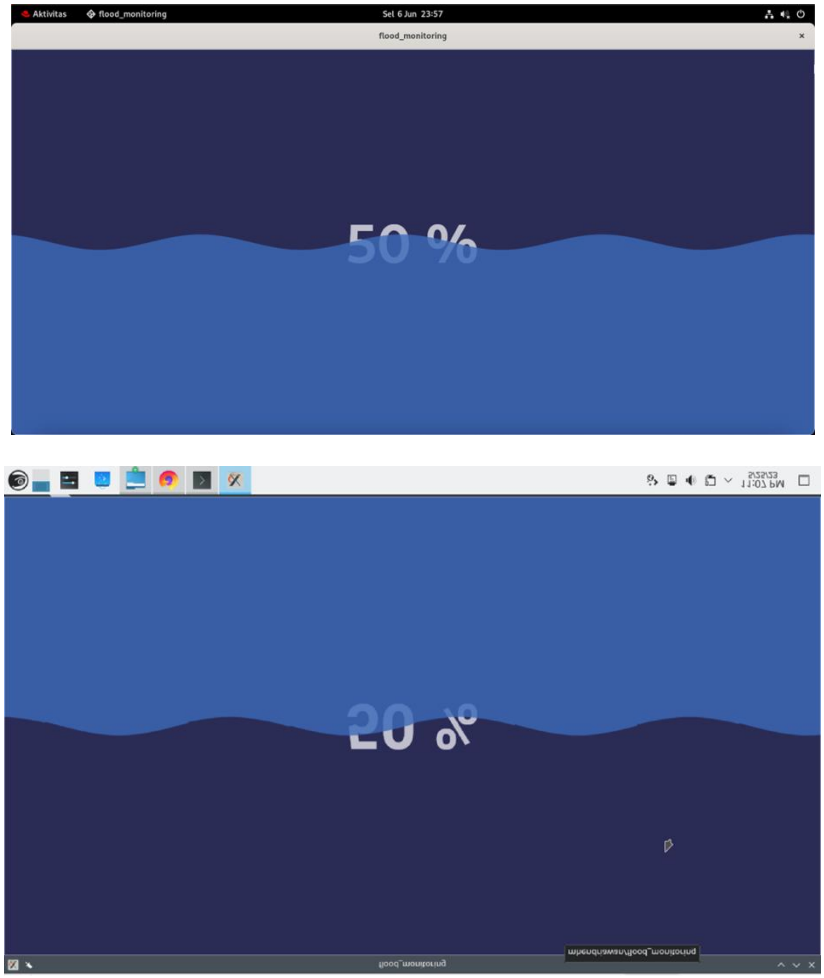
1. lib berisi file perpustakaan .so yang diperlukan
2. data berisi aset data aplikasi, seperti font atau gambar

Berikut hasil aplikasi flutter yang dapat berjalan di berbagai distro linux (Ubuntu,Redhat, OpenSUSE).




**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com



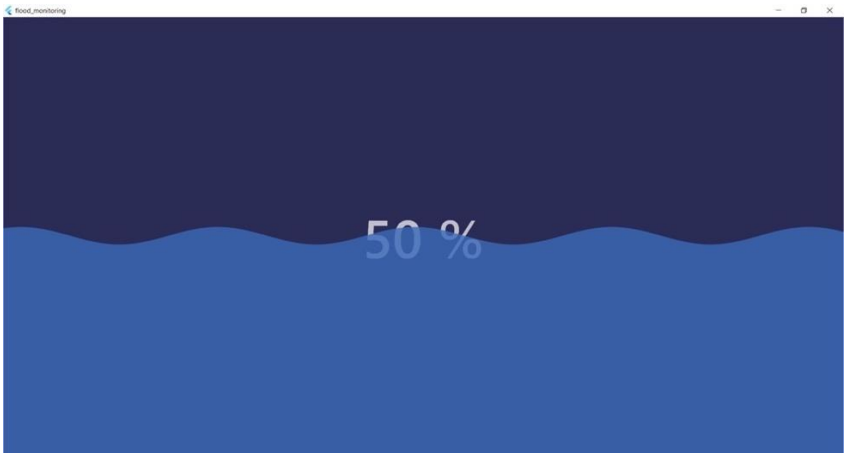

Gambar 4.16 App run on Linux

Untuk sebagian besar aplikasi, Flutter sudah cukup untuk menangani proses kompilasi menggunakan perintah flutter run dan flutter build. Untuk membuild aplikasi windows dengan flutter bisa mengikuti langkah ini:

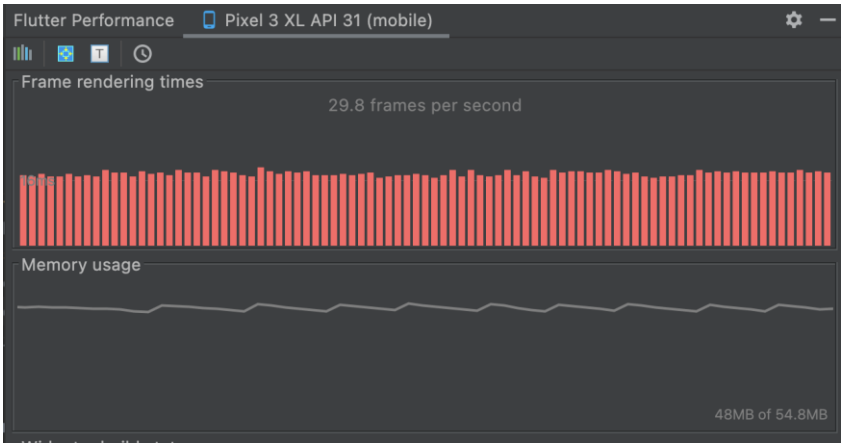
1. Jalankan flutter build windows untuk membuat direktori build\.
2. Buka file solusi Visual Studio untuk Windows runner, yang sekarang dapat ditemukan di direktori build\windows, dinamai menurut aplikasi induk Flutter.
3. Di Solution Explorer, Anda akan melihat sejumlah proyek. Klik kanan yang memiliki nama yang sama dengan aplikasi Flutter, dan pilih Set as Startup Project.
4. Jalankan Build / Build Solution (atau tekan Ctrl+Shift+B) untuk menghasilkan dependensi yang diperlukan. Anda sekarang seharusnya dapat menjalankan Debug / Mulai Debugging (atau tekan F5) untuk menjalankan aplikasi Windows dari Visual Studio.
5. Gunakan bilah alat untuk beralih antara konfigurasi Debug dan Rilis yang sesuai.

**DOI:** 10.52362/jisicom.v7i2.1197

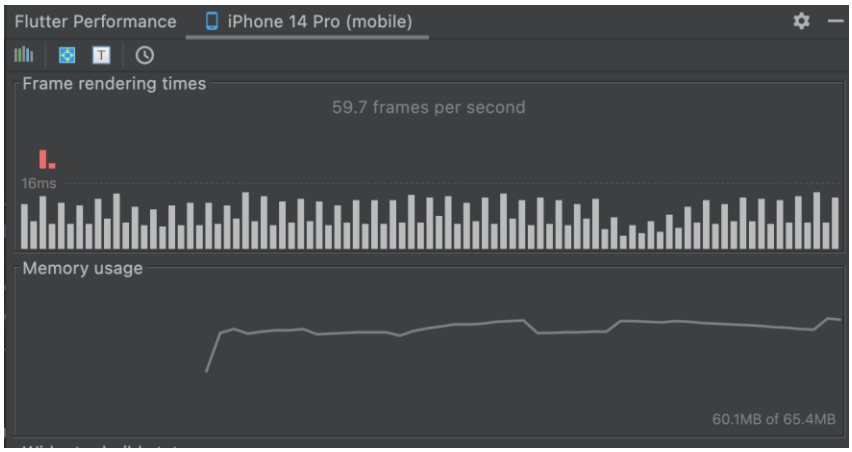
e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com




Gambar 4.17 App run on Windows



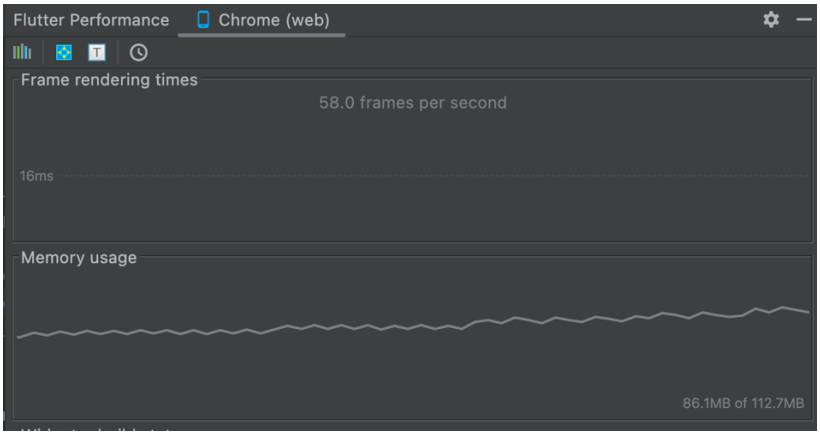

Gambar 4.18 App run on Android Performance



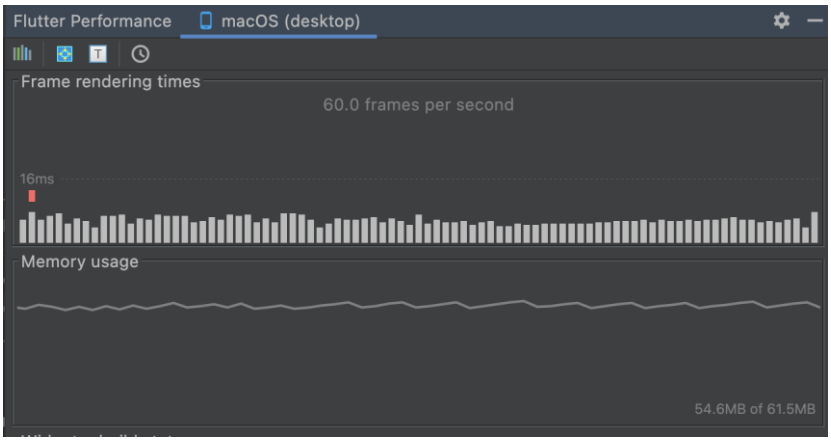

Gambar 4.19 App run on iPhone Performance

**DOI:** 10.52362/jisicom.v7i2.1197

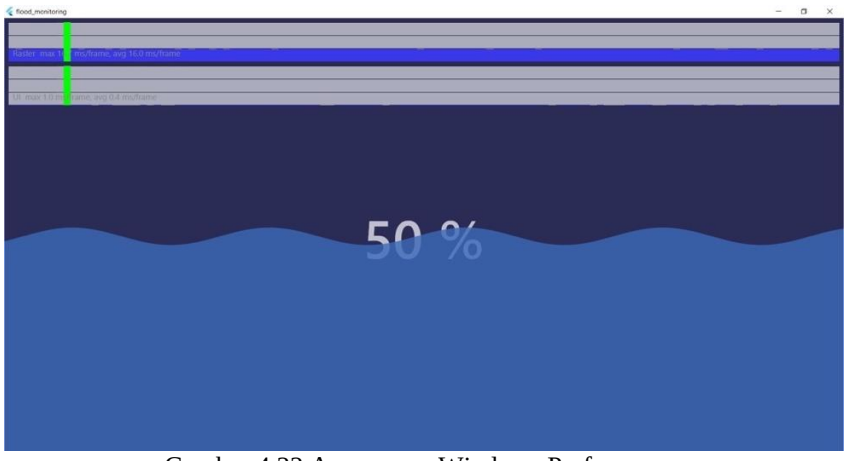
e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com




Gambar 4.20 App run on Web Performance



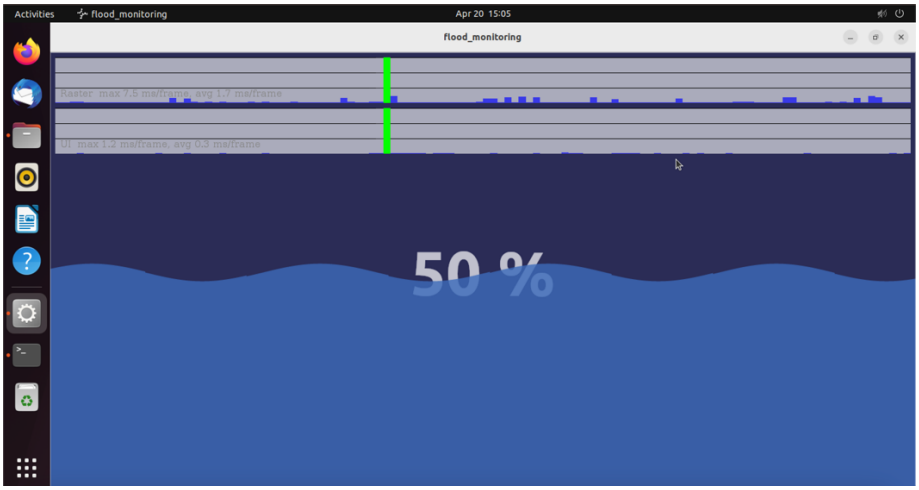

Gambar 4.21 App run on macOS Performance




Gambar 4.22 App run on Windows Performance

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com



Gambar 4.23 App run on Linux Performance

.

## IV. KESIMPULAN

4.1 Kesimpulan

Berdasarkan hasil penelitian yang dilakukan, dapat disimpulkan bahwa menggunakan NodeMCU ESP8266 dan HC-SR04 untuk memonitoring ketinggian air menggunakan MQTTnet dan Flutter adalah solusi yang efektif dan efisien. Hasil fleksibilitas aplikasi dapat berjalan di berbagai platform hampir 100%, karena dapat berjalan di OS aplikasi mobile (Android dan iOS), Web, Windows, MacOS, dan Linux dengan system deb (ubuntu) dan rpm (Redhat, OpenSUSE).

Dalam penggunaan teknologi ini, terdapat beberapa hal yang perlu diperhatikan. Pertama, perlu dilakukan pengaturan pada jaringan WiFi dan konfigurasi server MQTT agar dapat berkomunikasi dengan baik. Kedua, belum terintegrasi langsung dengan sumber listrik. Ketiga, belum dilengkapi dengan casing/cover agar mudah dan aman untuk digunakan dimanapun. Terakhir, fleksibilitas aplikasi dapat berjalan di berbagai platfrom terbatas mengikuti kompabilitas framework flutter dan paket library yang digunakan dalam pengembangan aplikasi.

4.2 Saran

Dalam pengembangan selanjutnya, dapat dilakukan integrasi dengan seluruh system smart home yang tersedia dan dapat dikembangkan fitur-fitur tambahan, seperti pengaturan konfigurasi wifi dan sensor ultrasonik melalui aplikasi. Melengkapi perangkat dengan casing/cover agar aman, mengintegrasikan power charge batterai agar memiliki opsi tambahan sebagai sumber daya perangkat.

## REFERENASI

Disusun dan diberi nomor urut berdasarkan urutan kutipan. Penulisan pustaka: nama penulis (tanpa gelar), tahun, judul, penerbit, dan kota penerbit. Berikut adalah contoh penulisan daftar pustak/referensi:

[1] Sommerville.Ian (2011) “Software Engineering” 9th Edition, Published by Addison-WesleyVongsingthong, S., & Smanchat, S. (2014). INTERNET OF THINGS: A REVIEW OF APPLICATIONS & TECHNOLOGIES. Suranaree Journal of Science and Technology.

**DOI:** 10.52362/jisicom.v7i2.1197

e-ISSN : 2597-3673 (Online) , p-ISSN : 2579-5201 (Printed) Vol.7 No.2 (December 2023)
**JISICOM (Journal of Information System, Informatics and Computing)**
Website/URL: http://journal.stmikjayakarta.ac.id/index.php/jisicom
Email: jisicom@stmikjayakarta.ac.id , jisicom2017@gmail.com

[2] Perera, C., Liu, C. H., & Jayawardena, S. (2015). The Emerging Internet of Things Marketplace From an Industrial Perspective: A Survey. IEEE Transactions on Emerging Topics in Computing, 4, 585-598.
[3] Putri, A. S. (2020). Retrieved from kompas.com: https://www.kompas.com/skola/read/2020/01/27/100000369/wawancara--pengertian-dan-tahapan?page=all
[4] Mustari, M., & Rahman, M. T. (2012). PENGANTAR METODE PENELITIAN. Laksbang Pressindo Yogyakarta.
[5] Arifin, N. Y., Borman, R. I., Ahmad, I., Tyas, S. S., Sulistiani, H., Hardiansyah, A., & Suri, G. P. (2022). Analisa Perancangan Sistem Informasi. Yayasan Cendikia Mulia Mandiri.
[6] Moreno, C., Aquino, R., Ibarreche J, Pérez I, Castellanos E, Alvarez, e., . . . Clark, B. (2019). RiverCore: IoT Device for River Water Level Monitoring over Cellular Communications. Sensors.
[7] Rani, S. S., Balakrishnan, S., Sundari, V. K., & Ramya, K. C. (2019). IoT Based Water Level Monitoring System for Lake in a Cloud Environment. International Journal of Lakes and Rivers, 12, 21-25.
[8] Perumal, T., Sulaiman, M. N., & Leong, C. Y. (2015). Internet of Things (IoT) enabled water monitoring system. Global Conference on Consumer Electronics, 86-87.
[9] Sommerville. (2011). Software Engineering (9th Edition). Pearson Education.
[10] Raharjo, B. (2019). Pemrograman Android dengan Flutter. Informatika.
[11] Khedo, K. K. (2013). REAL-TIME FLOOD MONITORING USING WIRELESS SENSOR NETWORKS. 59-69.
[12] Saputra, J. (2020). Automatic water level monitoring integrated with IoT through smartphone. Scientific Publication.
[13] Edifianto, G. H., Damarjati, C., & Asroni, A. (2021). Water Level Monitoring System Simulation Using Flutter Framework in Pekalongan City. Emerging Information Science and Technology, 2, 46-56.
[14] Kamaruidzaman, N. S., & Rahmat, S. N. (2020). Water Monitoring System Embedded with Internet of Things (IoT) Device: A Review. IOP Conference Series: Earth and Environmental Science, 498.
[15] Pasika, S., & Gandla, S. T. (2020). Smart water quality monitoring system with cost-effective using IoT. Science Direct, 6(7).
[16] Jisha, R. C., Vignesh, G., & Deekshit, D. (2019). IOT based Water Level Monitoring and Implementation on both Agriculture and Domestic Areas. 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), 1119-1123.
[17] Malche, T., & Maheshwary, P. (2017). Internet of Things (IoT) Based Water Level Monitoring System for Smart Village. Proceedings of International Conference on Communication and Networks, 305-312.
[18] Siddula, S. S., Babu, P., & Jain, P. C. (2018). Water Level Monitoring and Management of Dams using IoT. 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU).
[19] Sachio, S., Noertjahyana, A., & Lim, R. (2018). IoT Based Water Level Control System. The 2018 Technology Innovation Management and Engineering Science International Conference (TIMES-iCON2018).

**DOI:** 10.52362/jisicom.v7i2.1197