SOCIAL DISTANCING DETECTOR MENGGUNAKAN BLUETOOTH BERBASIS *WEBSITE*

TUGAS AKHIR

Program Studi
S1 TEKNIK KOMPUTER

Oleh:
Hendra Daniswara
19410200019

FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2023

SOCIAL DISTANCING DETECTOR MENGGUNAKAN BLUETOOTH BERBASIS *WEBSITE*

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan

Program Sarjana Teknik

Disusun Oleh :

Nama	**: Hendra Daniswara**
NIM	**: 19410200019**
Program Studi	**: S1 Teknik Komputer**

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2023

TUGAS AKHIR

SOCIAL DISTANCING DETECTOR MENGGUNAKAN BLUETOOTH BERBASIS *WEBSITE*

Dipersiapkan dan disusun oleh

Hendra Daniswara

NIM: 19410200019

Telah diperiksa, diuji dan disetujui oleh Dewan Penguji

Pada: Desember 2022

Susunan Dewan Pembahas

Pembimbing:

I. Musayyanah, S.ST., M.T.

NIDN: 0730069102

II. Pauladie Susanto, S.Kom., M.T.

NIDN: 0729047501

Digitally signed by Musayyanah
DN: cn=Musayyanah, o=Universitas Dinamika, ou=S1 Teknik Komputer, email=musayyanah@dinamika.ac.id, c=ID
Date: 2023.01.11 07:41:42 +07'00'
Adobe Acrobat Reader version: 2022.003.20282

Universitas Dinamika
2023.01.11 16:20:14 +07'00'

Pembahas:

Harianto, S.Kom., M.Eng.

NIDN: 0722087701

cn=Harianto Harianto, o=Universitas Dinamika, ou=Prodi S1 Teknik Komputer, email=hari@dinamika.ac.id, c=ID
2023.01.11 17:16:05 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan

untuk memperoleh gelar sarjana

Digitally signed by Universitas Dinamika
Date: 2023.01.18 07:19:29 +07'00'

Tri Sagirani, S.Kom., M.MT.

NIDN: 0731017601

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA

“From Zero to Hero”

~ Hendra D. ~

Ku persembahkan karya Tugas Akhir ini untuk kedua orang tua saya, seluruh keluarga yang saya cintai dan semua orang yang selalu memberi semangat serta motivasi sehingga saya bisa menyelesaikan karya ini.

PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa **Universitas Dinamika**, Saya :

Nama	: **Hendra Daniswara**
NIM	: **19410200019**
Program Studi	: **S1 Teknik Komputer**
Fakultas	: **Fakultas Teknologi dan Informatika**
Jenis Karya	: **Laporan Tugas Akhir**
Judul Karya	: **SOCIAL DISTANCING DETECTOR MENGGUNAKAN BLUETOOTH BERBASIS WEBSITE**

Menyatakan dengan sesungguhnya bahwa :

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, Saya menyetujui memberikan kepada **Universitas Dinamika** Hak Bebas Royalti Non-Eksklusif (***Non-Exclusive Royalty Free Right***) atas seluruh isi/sebagian karya ilmiah Saya tersebut diatas untuk disimpan, dialihmediakan, dan dikelola dalam bentuk pangkalan data (***database***) untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama Saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.
2. Karya tersebut diatas adalah hasil karya asli Saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya, atau pendapat orang lain yang ada dalam karya ilmiah ini semata-mata hanya sebagai rujukan yang dicantumkan dalam Daftar Pustaka Saya.
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiasi pada karya ilmiah ini, maka Saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada Saya.

Surabaya, 9 Desember 2022

Hendra Daniswara
NIM : 19410200019

ABSTRAK

Seluruh dunia dikejutkan dengan wabah virus COVID-19, korban dari virus ini tidak sedikit yang ada diseluruh dunia. Salah satu cara untuk mencegah tertularnya virus COVID-19 adalah dengan menjaga jarak minimal dua meter dengan orang lain. Namun banyak orang yang bisa mengetahui jarak antar orang lain dengan perkiraan. Tujuan dari penelitian ini adalah menentukan jarak antar orang dengan menerapkan Kalman *filter* dan di*monitoring* dari *website*. Dengan memanfaatkan teknologi Bluetooth yang merupakan salah satu teknologi yang dapat memperkirakan jarak dengan *Received Signal Strength Indicator* (RSSI). RSSI dikonversi menjadi jarak dengan menggunakan model *log distance path loss.* RSSI akan melewati Kalman *filter* untuk memberikan hasil yang stabil. Bluetooth sebagai *End node* akan mengirimkan data ke *webserver* menggunakan komunikasi LoRa. Kemudian *website* akan mengambil data dari d*atabase* untuk monitoring. Berdasarkan hasil pengujian menunjukkan bahwa semakin jauh jarak, nilai RSSI akan semakin kecil. Penerapan Kalman *filter* dapat memperkecil kesalahan konversi jarak sebesar 42,649%. Penerapan kalman *filter* pada komunikasi 3 *end node* memiliki *error* terkecil 2,75%. Pengujian transmisi data dilakukan selama 19 menit dengan tingkat keberhasilan 100%.

Kata Kunci: *Social Distancing, Bluetooth Low Energy, RSSI, LoRa, Kalman Filter, log distance path loss model.*

KATA PENGANTAR

Puja dan puji syukur kehadirat Tuhan Yang Maha Esa atas segala karunia dan hidayah-Nya yang telah diberikan sehingga penulis dapat menyelesaikan Laporan Tugas Akhir yang berjudul "Social Distancing Detector Menggunakan Bluetooth Berbasis Website".

Dengan menyelesaikan laporan ini, penulis mengucapkan banyak terima kasih kepada semua pihak yang telah membantu dan mendukung dalam penyusunan laporan Tugas Akhir ini, diantaranya:

1. Allah SWT, karena dengan rahmatnya dan hidayahnya penulis dapat menyelesaikan Laporan Tugas Akhir ini.
2. Kedua Orang Tua dan keluarga yang selalu memberikan doa serta dukungan.
3. Ibu Tri Sagirani, S.Kom., M.MT. selaku Dekan Fakultas Teknologi dan Informatika (FTI) Universitas Dinamika.
4. Bapak Pauladie Susanto, S.Kom., M.T. selaku Ketua Program Studi S1 Teknik Komputer dan dosen pembimbing, terima kasih atas bimbingan yang diberikan baik secara tertulis maupun lisan sehingga penulis dapat menyelesaikan Tugas Akhir ini.
5. Ibu Musayyanah, S.ST., M.T. selaku dosen pembimbing yang telah memberikan masukan dan solusi agar Tugas Akhir ini dapat selesai dan menjadi lebih baik lagi.
6. Bapak Harianto, S.Kom., M.Eng. selaku Dosen Pembahas atas saran dan masukannya sehingga penulis dapat menyelesaikan laporan Tugas Akhir ini.
7. Teman – teman S1 Teknik Komputer viiingkatan 2019 Universitas Dinamika, yang telah membantu penulis selama proses pengerjaan.
8. Hera Puspita Riantiningtyas, yang selalu memberikan semangat dan dukungan kepada penulis untuk dapat segera menyelesaikan laporan Tugas Akhir ini.

Penulis menyadari bahwa karya yang telah disusun ini jauh dari kata sempurna. Tetapi Penulis berharap semoga laporan ini dapat berguna dan bermanfaat untuk menambah wawasan dan pengetahuan bagi pembacanya.

Surabaya, 9 Desember 2022

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pandemi Coronavirus *disease* (COVID-19) menurut WHO (*World Health Organization*) adalah virus yang dapat menyebabkan penyakit pada hewan dan manusia. Virus Corona menyebar melalui tetesan kecil dari hidung atau mulut yang menyebar ketika orang bersin atau batuk (*droplets*). Penularan dapat terjadi melalui *droplet* dari saluran nafas. Penularan juga bisa terjadi akibat kontak erat dengan penderita (Nany & Husnun, 2020). Senantiasa memperhatikan jarak setidaknya 1-2 meter saat berinteraksi dengan orang lain berguna untuk menghentikan atau memperlambat penyebaran penyakit menular (Arief & Juni, 2020). Namun banyak orang yang tidak mengetahui berapa jarak antara satu sama lain, dan tidak mengetahui dengan siapa saja melakukan kontak fisik kurang dari 2 meter dengan orang yang terjangkit virus Corona.

Bluetooth *Low Energy* (BLE) sebagai media untuk perhitungan jarak *physical distancing* menggunakan RSSI sebagai parameternya. BLE sendiri adalah salah satu perangkat tambahan terbaru untuk teknologi Bluetooth yang ditambahkan sebagai bagian dari spesifikasi Bluetooth 4.0 (Robi & Rahmi, 2021). BLE sendiri diharapkan bisa beroperasi dengan waktu yang cukup lama karena konsumsi daya yang sangat rendah. RSSI dari Bluetooth dapat dirubah menjadi jarak menggunakan model *log distance path loss*.

Sebelum melakukan perhitungan jarak, RSSI akan di*filter* terlebih dahulu dikarenakan hasil pengukuran *raw* dari RSSI tidak stabil. *Filter* yang digunakan adalah Kalman *filter*. *Filter ini* merupakan sebuah proses berulang yang menggunakan persamaan matematis dan input data yang berurutan (Willy, Andi & Abhitama, 2016). Hasil dari Kalman *filter* yang akan menjadi *variable* untuk perhitungan jarak RSSI.

Berdasarkan permasalahan diatas, maka Tugas Akhir ini bertujuan menerapkan Bluetooth untuk deteksi jarak berbasis model *log distance path loss*. Serta Kalman Filter untuk memperkecil kesalahan konversi jarak. Selain itu, Tugas

Akhir ini membuat algoritma komunikasi antar *End node* dengan *gateway* kemudian datanya ditampilkan di *website.*

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dirumuskan masalah pada Tugas Akhir ini sebagai berikut:

1. Bagaimana menentukan jarak antar Bluetooth dengan *log distance path loss* model?
2. Bagaimana menerapkan Kalman *Filter* untuk mendapatkan RSSI yang stabil?
3. Bagaimana membuat website yang terintegrasi dengan *Database*?

1.3 Batasan Masalah

Dalam pembuatan Tugas Akhir ini, pembahasan masalah dibatasi pada beberapa hal berikut:

1. Mengukur jarak *typical* bukan *actual.*
2. Komunikasi Bluetooth hanya bisa dengan tipe Bluetooth yang sama.
3. Tidak memperhatikan konsumsi energi tiap *end node.*
4. LoRa hanya berperan untuk komunikasi *end node* ke *gateway* dan melanjutkan data ke server.

1.4 Tujuan

Berdasarkan latar belakang dan rumusan masalah diatas, mendapatkan tujuan pada tugas akhir ini sebagai berikut:

1. Menentukan jarak antar Bluetooth menggunakan *log distance path loss model.*
2. Menerapkan Kalman *filter* untuk mendapatkan RSSI yang stabil.
3. Membuat *website* yang terintegrasi dengan *database*.

1.5 Manfaat

Adapun dari Tugas Akhir ini dapat diperoleh manfaat sebagai berikut:

1. Menegtahui penyebaran virus Corona.
2. Mengetahui jarak yang ideal untuk *physical distancing.*

3. Membuat alat untuk *physical distancing* dan bisa dikontrol untuk *tracing* di dalam *website.*

BAB II
TINJAUAN PUSTAKA

2.1 Bluetooth *Low Energy* (BLE)

Bluetooth *Low Energy* (BLE) adalah teknologi terbaru untuk jarak pendek yang beroperasi di frekuensi 2.4 GHz. BLE berbeda dengan Bluetooth konvensional, BLE mempunyai protokol terbaru yang mengoptimalkan kinerjanya dalam aspek efektivitas penggunaan daya.

Gambar 2.1 Bluetooth Low Energy
(Sumber: global-tag.com)

Jika membicarakan Bluetooth, maka tidak lepas dengan *Received Signal Strength Indicator* (RSSI), dimana RSSI adalah ukuran seberapa baik perangkat dapat mendapatkan sinyal di titik tertentu. Satuan dari RSSI adalah dBm (*decibel milliwatt*). RSSI dan dBm adalah unit pengukuran yang berbeda tetapi mewakili hal yang sama yaitu kekuatan daya sebar sinyal. Namun perbedaannya adalah bahwa RSSI adalah indeks relatif, sedangkan dBm adalah angka yang mewakili tingkat daya dalam mW (milliwatt).

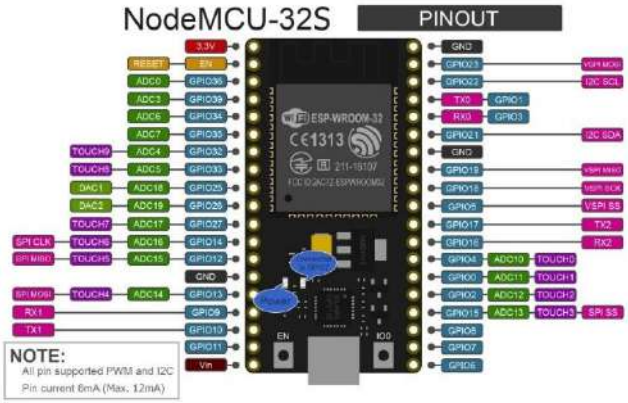
2.2 ESP32

Gambar 2.2 ESP32
(Sumber: leantec.es)

Espressif *System* mempunyai banyak mikrokontroler salah satunya adalah ESP32. Mikrokontroler ini kompatibel dengan Arduino IDE. ESP32 juga sudah tersedia modul WiFi dan ditambah dengan BLE (Bluetooth *Low Energy*) di dalamnya, sehingga sangat mendukung dan dapat menjadi pilihan bagus untuk membuat sistem berbasis *Internet of Things*.

Tabel 2.1 Datasheet ESP32

Categories	Items	Specifications
	Audio	CVSD and SBC
Hardware	Module interfaces	SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC, Two-Wire Automotive Interface (TWAI®), compatible with ISO11898-1 (CAN Specification 2.0)
	On-chip sensor	Hall sensor
	Integrated crystal	40 Mhz crystal
	Integrated SPI flash	4 MB
	Operating voltage/Power supply	3.0 V ~ 3.6 V
	Operating current	Average: 80 mA
	Minimum current delivered by power supply	500 mA
	Recommended operating ambient temperature range	–40 °C ~ +85 °C
	Pakcage size	18 mm × 25.5 mm × 3.10 mm
	Moisture sensitivity level (MSL)	Level 3



Gambar 2.3 *Pinout* ESP32
(Sumber: 99tech.com)

2.3 *Log distance path loss model*

Pada perhitungan RSSI, kuat sinyal merupakan hal yang sangat penting dalam menentukan estimasi jarak antar *end node.* Banyak faktor yang mempengaruhi kuat sinyal yang diterima, *multipath fading,* efek antena, dan efek peralatan transmisi itu sendiri. Karena itu dibutuhkan model kanal untuk mengurangi kerugian propagasi.

Log distance path loss model dalah model *generic* dan pengembangan dari *Friis Free Space Model.* Hal ini digunakan untuk memprediksi kerugian propagasi untuk berbagai lingkungan. Sedangkan *Friis Free space model* dibatasi untuk area yang jelas terdapat penghalang antara pemancar dan penerima.

$$-RSSI = 10nLog_{10}(d) - RSS_{d0} \quad \text{..... (1)}$$

Dari persamaan (1) dapat dicari jarak dengan substitusi :

$$distance = 10^{(\frac{RSS_{d0}-RSSI}{10n})} \quad \text{..... (2)}$$

Penjelasan:

n: *Path loss exponent*

RSS_{d0}: Referensi RSSI pada jarak 1 meter

2.4 Kalman *Filter*

Kalman *filter* adalah algoritma dimana digunakan untuk memprediksi atau mengestimasi data selanjutnya berdasarkan data sebelumnya. Kalman *filter* tidak di kategorikan seperti *filter* pada umumnya seperti, LPF, HPF, dan BPF (Alfian, Iswanto, Aninditya dan Rio, 2019)

Kalman *filter* diasumsikan sebagai linear sistem. Kalman *filter* meminimalisasi rata rata *estimation error square*. Dengan proses ini akan mengestimasi data sebenarnya secara cepat, nilai yang diukur tersebut umumnya terdapat error acak atau variasi. Sebagai contoh nilai RSSI dari sebuah sumber yang memiliki data raw yang sangat acak. Maka dari itu, menggunakan Kalman *filter* akan membuat estimasi nilai dengan secara cepat mendekati nilai pengukuran yang sesungguhnya.

***Predict*:**

$$X_{t|t-1} = F_t X_{t-1|t-1} + B_t U_t \quad \text{.....(3)}$$

$$P_{t|t-1} = F_t P_{t-1|t-1} F_t^T + Q_t \quad \text{.....(4)}$$

***Update*:**

$$X_{t|t} = X_{t|t-1} + K_t(y_t - H_t X_{t|t-1}) \quad \text{.....(5)}$$

$$K_t = P_{t|t-1} H_t^T (H_t P_{t|t-1} H_t^T + R_t)^{-1} \quad \text{.....(6)}$$

$$P_{t|t} = (1 - K_t H_t) P_{t|t-1} \quad \text{.....(7)}$$

Dimana X adalah estimated *state*, F adalah *state* transition matrix, u adalah kontrol variabel, P adalah *state* variance matrix, Q adalah proses variance matrix, y adalah measurement variables, H adalah measurement matrix, K adalah Kalman gain, R adalah measurement matrix, $t|t$ adalah current time period, $t-1|t-1$ is previous time period, dan $t|t-1$ adalah *intermediate steps*.

Kalman *filter* bisa di modifikasi sesuai dengan kebutuhan yang akan digunakan (Alfian, Iswanto, Aninditya dan Rio, 2019). Untuk kebutuhan pengurangan *noise* pada sensor, memerlukan modifikasi persamaan (3) sampai (7) sebagai berikut.

1. *Predicting the state*

Untuk Persamaan (3) Memberikan $F_t = 1$ karena tidak ada transisi *state*. Menghilangkan B_t dan U_t karena sistem yang digunakan tidak ada input kontrol variabel. Hasil persamaan yang telah di sesuaikan ditunjukkan pada (8).

$$X_{t|t-1} = X_{t-1|t-1} \quad \text{.....(8)}$$

2. *Predicting the error*

Ada perubahan $F_t = 1$, persamaan (4) berubah menjadi (9).

$$P_{t|t-1} = P_{t-1|t-1} + Q_t \quad \text{.....(9)}$$

3. *Updating the state value*

Modifikasi persamaan (5), $H_t = 1$ karena hanya ada 1 data sensor. Maka perubahannya menjadi (10).

$$X_{t|t} = X_{t|t-1} + K_t(y_t - X_{t|t-1}) \quad \text{.....(10)}$$

4. *Calculating the gain of Kalman*

Ada perubahan $H_t = 1$, maka persamaan (6) berubah menjadi (11).

$$K_t = P_{t|t-1}(P_{t|t-1} + R)^{-1} \quad \text{.....(11)}$$

5. *Updating the error value*

Perubahan $H_t = 1$, maka persamaan (7) berubah menjadi (12).

$$P_{t|t} = (1 - K_t)P_{t|t-1} \quad \text{.....(12)}$$

Setelah semua penyesuaian dilakukan, kalman *filter* untuk penguranan *noise* pada sensor dapat ditulis ulang (13) sampai (17).

Predict :

$$X_{t|t-1} = X_{t-1|t-1} \quad \text{.....(13)}$$

$$P_{t|t-1} = P_{t-1|t-1} + Q_t \quad \text{.....(14)}$$

Update :

$$X_{t|t} = X_{t|t-1} + K_t(y_t - X_{t|t-1}) \quad \text{.....(15)}$$

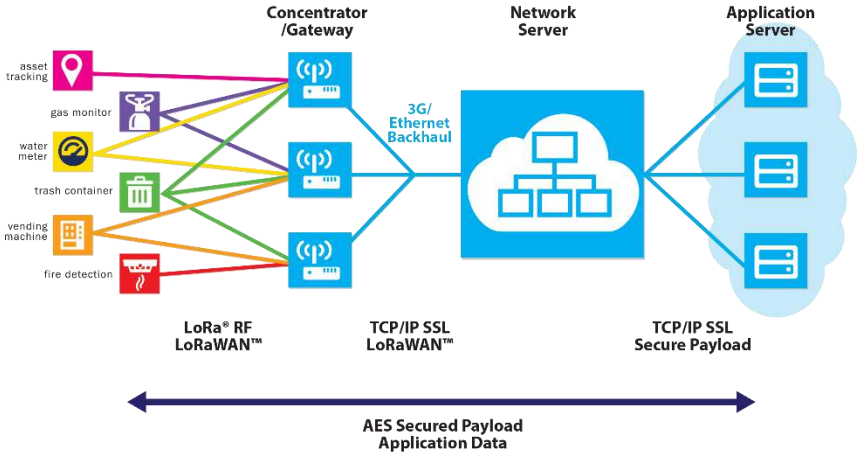
$$K_t = P_{t|t-1}(P_{t|t-1} + R)^{-1} \quad \text{.....(16)}$$

$$P_{t|t} = (1 - K_t)P_{t|t-1} \quad \text{.....(17)}$$

2.5 LoRa

Teknologi LoRa (*Long Range*) menggunakan teknologi modulasi CSS (*Chirp Spread Spectrum*) yang memungkinkan untuk mengirim data jarak jauh dengan berdaya rendah melalui pita ISM (*Instrumentation Science and Medical*) yang tidak berlisensi. Modulasi yang dibuat oleh Semtech ini mempunyai bermacam macam nilai frekuensi sesuai daerahnya, jika di Asia frekuensi yang digunakan yaitu 433 Mhz, di Eropa nilai frekuensi yang digunakan yaitu 868 Mhz, sedangkan di Amerika Utara frekuensi yang digunakan yaitu 915 Mhz.

Kelebihan pada LoRa berdaya rendah dan jarak jangkaunya jauh. Dengan kelebihannya, LoRa banyak digunakan di project IoT (*Internet of Things*) untuk mengantarkan *end node* ke *network* server melewati LoRa *gateway*.

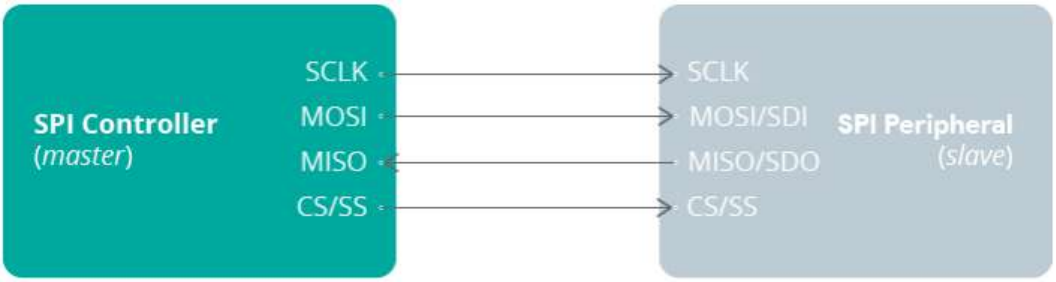


Gambar 2.4 Sistem komunikasi LoRaWAN
(Sumber: kmtech.id)

Dari Gambar 2.4 dapat diketahui bahwa *end node* seperti sensor-sensor akan dilanjutkan ke *network* server melewati LoRa *gateway*. Setelah itu data akan diteruskan ke *Application* server untuk pengolahan data.

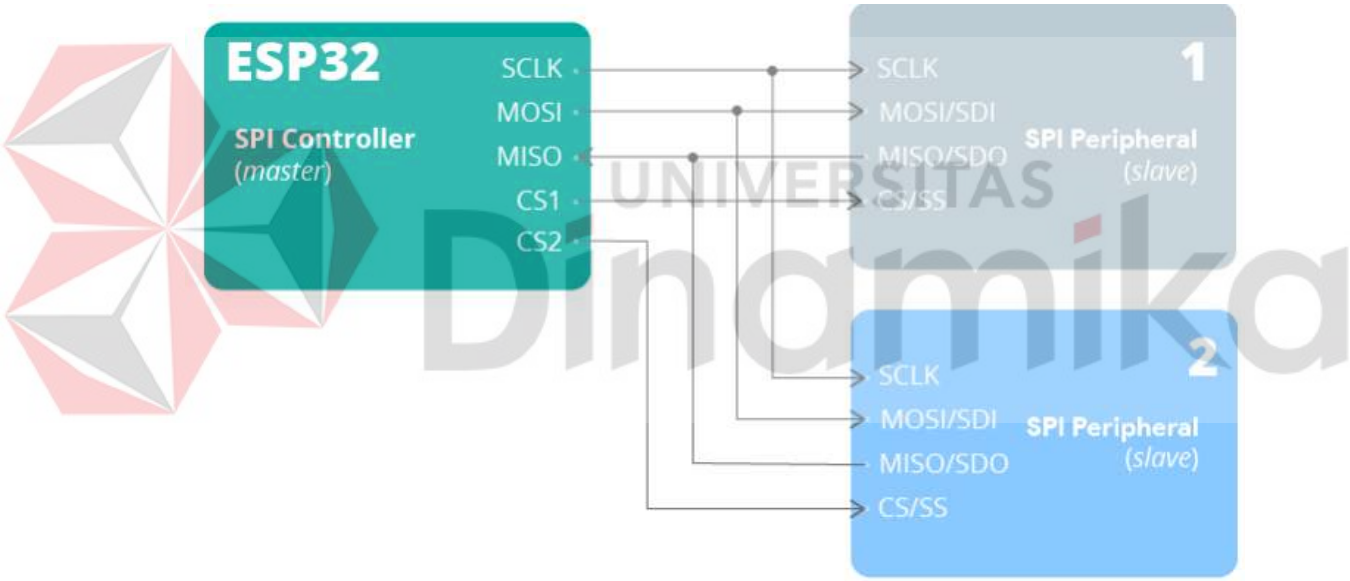
2.6 *Serial Peripheral Interface* (SPI)

Serial Peripheral Interface (SPI) merupakan protokol serial data yang digunakan mikrokontroler untuk komunikasi dengan 1 perangkat atau lebih dan tentunya perangkat tersebut juga mendukung komunikasi menggunakan protokol SPI. Komunikasi SPI terdapat kontroler (*master*) dan perangkat (*slaves). Master* akan kirim data ke *slaves* dan begitupun juga *slaves* juga kirim data ke *master*. Itu berarti *master* dan *slaves* bisa kirim data secara bersamaan. Didalam komunikasi SPI hanya bisa mempunyai 1 *master* itu berarti adalah mikrokontroler. Tetapi bisa mempunyai lebih dari 1 *slaves. Slaves* bisa berupa sensor, *microSD card, displa screen*, dan lain lain.



Gambar 2.5 konfigurasi pin SPI
(Sumber: randomnerdtutorials.com)

Untuk menerapkan komunikasi SPI, dibutuhkan beberapa *interface* diantaranya yaitu *Master In Slave Out* (MISO), *Master Out Slave In* (MOSI), *Serial Clock* (SCK), *Chip Set* (CS/SS).



Gambar 2.6 SPI konfigurasi pin SPI *multiSlaves*
(Sumber: randomnerdtutorials.com)

Jika ingin menggunakan *multislaves* caranya adalah dengan mengatur CS/SS nya. Pada umumnya untuk pilih perangkat adalah dengan memberi sinyal *LOW* pada CS/SS. Pada Gambar 2.6 jika ingin pilih perangkat 1 yang aktif, maka CS1 harus *LOW* dan CS2 harus *HIGH.* Begitupun juga sebaliknya.

2.7 Flask

Flask adalah satu "*micro-framework*" atau yang disebut *framework* kecil dengan menggunakan bahasa phyton sebagai bahasa pemrogramannya. Menjadi kecil bukan berarti Flask kurang dari *framework* yang lain. Flask dirancang sebagai *framework* yang dapat diperluas dengan memberikan layanan dasar, sementara ekstensi menyediakan sisanya.

Gambar 2.7 Logo Flask
(Sumber: kindpng.com)

Flask mempunyai dependensi utama yaitu, *The routing*, *Debugging*, dan *Web Server Gateway Interface* (WSGI). Dukungan *template* disediakan oleh Jinja 2, dan integrasi *command line* datang dari *click*.

2.8 MongoDB

MongoDB adalah sistem manajemen *database open source* menggunakan model basis data berorientasi dokumen yang mendukung berbagai bentuk data.

Gambar 2.8 Logo MongoDB
(Sumber: thehotskills.com)

MongoDB diciptakan oleh Dwight Merriman dan Eliot Borouwitz, dua orang yang sangat berpengalaman dalam dunia basis data. Nama MongoDB diambil dari kata *humongous* memilik arti gagasan yang mendukung sejumlah besar data. Sejak MongoDB pertama kali didirikan, MongoDB telah digunakan oleh banyak badan dan perusahaan kelas dunia seperti, Craiglist, CERN, hingga New York Times.

MongoDB memberikan fleksibilitas kepada pengguna untuk membuat sejumlah dokumen dan membuatnya lebih mudah. Dan salah satu keuntungan menggunakan MongoDB adalah objek dapat ditetapkan ke tipe data asli dalam sejumlah bahasa pemrograman.

2.9 Visual Studio Code

Visual Studio Code merupakan *software code* editor yang bisa digunakan pada perangkat windows, MacOS, maupun Linux. Pengembang atau developer dari Visual Code adalah perusahaan teknologi terkemuka di dunia, yaitu Microsoft. Visual Code sangatlah ringan saat digunakan. *Software* ini bisa digunakan untuk edit kode dari banyak macam bahasa pemrograman, mulai dari HTML, NodeJS, C++, PHP, JAVA, dan masih banyak lagi.

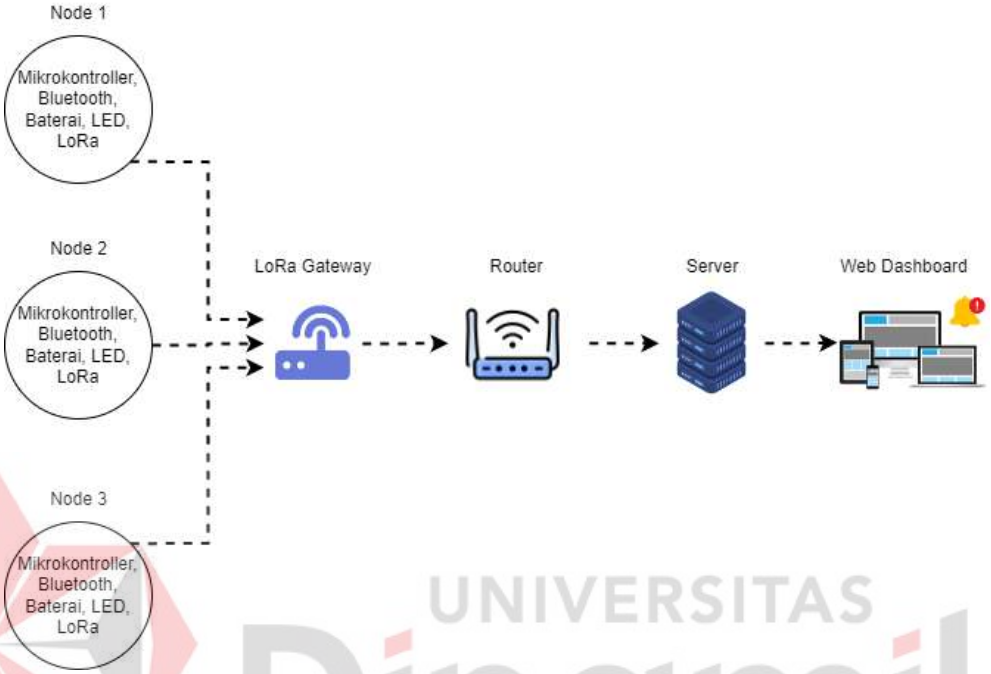
Gambar 2.9 Logo Visual Studio Code
(Sumber: medium.com)

Selain itu, Visual Code memiliki ekosistem yang luas dan banyak ekstensi. Fitur dari Visual Code juga ada diantaranya Basic Editing yaitu untuk menulis kode bahasa pemrograman. *Debugging* berfungsi untuk membantu dalam eksekusi, *Extension marketplace* adalah fitur unggulan dari Visual Code yang tidak di dapat di *software* code yang lain. Fitur ini memungkinkan untuk menginstal berbagai tools pendukung dari macam macam bahasa pemrograman yang digunakan.

Selain fitur, Visual Code juga mempunyai beberapa kelebihan diantaranya adalah soal performa yang cepat dan ringan, *Open Source*, Multi-platform, mendukung banyak bahasa pemrograman dan fitur lengkap seperti ekstensi yang disediakan.

BAB III
METODOLOGI PENELITIAN

3.1 *Blok Diagram*



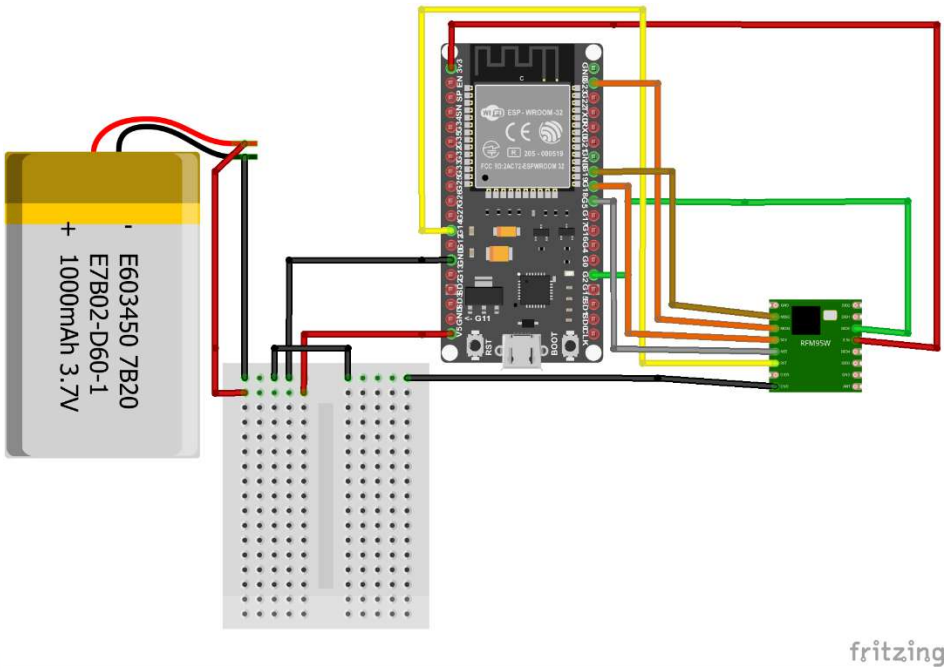
Gambar 3.1 Komunikasi antar *end node* dan *gateway*

Penjelasan dari blok diagram antar komponan pada Gambar 3.1 adalah sebagai berikut:

1. Mikrokontroler menggunakan ESP32 untuk mengatur keluar dan masuknya data.
2. BLE berada didalam mikrokontroler karena *built in* dengan ESP32.
3. LED untuk indikator ketika *end node* berdekatan.
4. Baterai digunakan untuk *supply* daya ke mikrokontroler.
5. LoRa digunakan untuk *transmit* data yang diterima mikrokontroler ke LoRa *Gateway*.
6. LoRa *Gateway* ini berfungsi untuk meneruskan data melewati router menuju server.
7. Server akan menampilkan data ke *dashboard* berbasis website.

3.2 Rangkaian Elektronika

3.2.1 Rangkaian skematik *end node* dan *gateway* LoRa



Gambar 3.2 Rangkaian *gateway* LoRa

Gambar 3.2 merupakan rangkaian *end node* sekaligus LoRa *gateway*. rangkaian *end node* terdapat ESP32 yang sudah *include* dengan Bluetooth *Low Energy* yang dimana nantinya akan menjadi data sensor untuk estimasi jarak antar *end node.* Selain itu, ada juga baterai untuk supply ESP32, serta ada LoRa untuk *transmit* data hasil bluetooth ESP32.

Pada rangkaian *End node* terdapat sebuah baterai *rechargeable* untuk *supply* ESP32. Sebelum masuk tegangan *supply* dari baterai, tegangan akan diregulasi menjadi 3.3V terlebih dahulu. Setelah itu baru bisa aman untuk supply ESP32. Karena pada *board* ESP32 untuk *input* 3.3V tidak ada regutor bawaan jadi tegangannya harus pas 3.3V tidak bisa lebih dari itu.

Selain rangkaian *End node,* rangkaian pada Gambar 3.2 juga berlaku untuk LoRa *gateway.* LoRa *gateway* mempunyai peran untuk meneruskan data dari *end node* menuju ke server. Dengan memanfaatkan *build in* WiFi pada ESP32, *gateway* bisa dengan mudah kirim data ke server dengan menggunakan webserver ESP32.

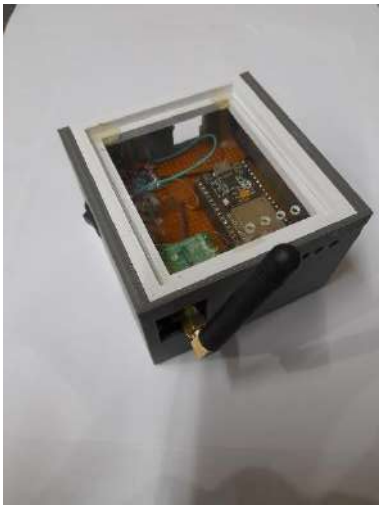
Tabel 3.1 Konfigurasi pin LoRa ke rangkaian ESP32

LoRa pin	ESP32 pin
ANA	-
DIO3	-
DIO4	-
3.3V	3.3V
DIO0	GPIO 2
DIO1	-
DIO2	-
GND	-
GND	GND
DIO5	-
RESET	GPIO 14
NSS	GPIO 5
SCK	GPIO 18 / VSPI CLK
MOSI	GPIO 23 / VSPI MOSI
MISO	GPIO 19 / VSPI MISO
GND	-

Tabel 3.2 Pin Baterai ke ESP32

Baterai	ESP32
VCC (+)	3.3V
GND (-)	GND

3.3 Rancangan mekanik *end node*



Gambar 3.3 Tampak depan *end node*

Rancangan mekanik *end node* terdiri dari ESP32, LoRa, *charging module*, regulator baterai, dan baterai 18650. Dipacking didalam 3d *printing* yang sudah didesain sedemikian rupa.

Gambar 3.4 Tampak dalam *End node*

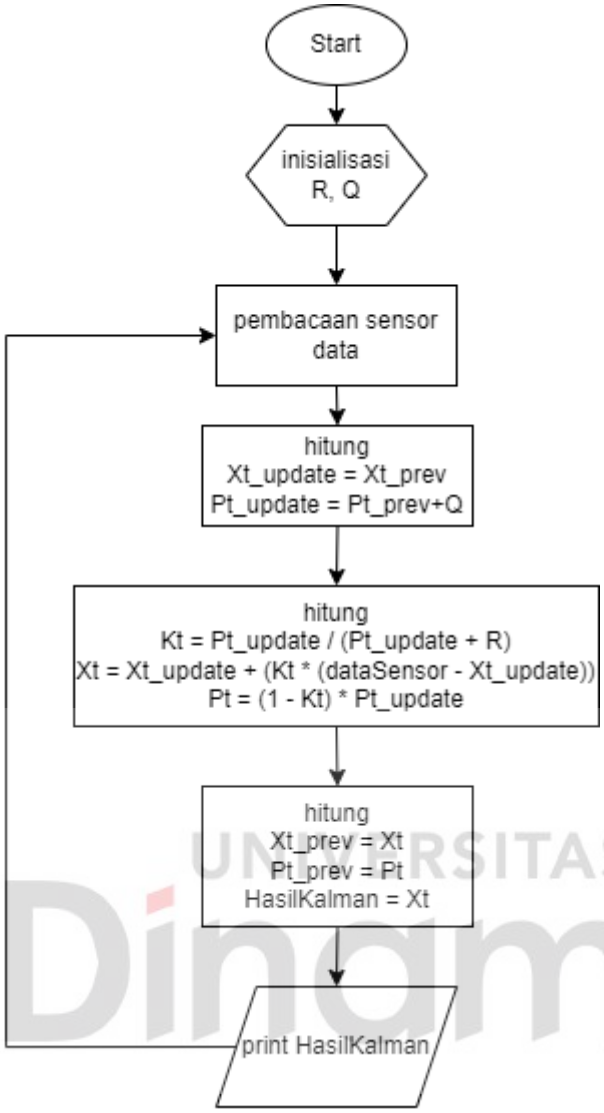
Gambar 3.5 Tampak belakang *end node*

3.4 Rancangan mekanik *gateway*

Rancangan mekanik *gateway* terdiri dari ESP32 dan LoRa.

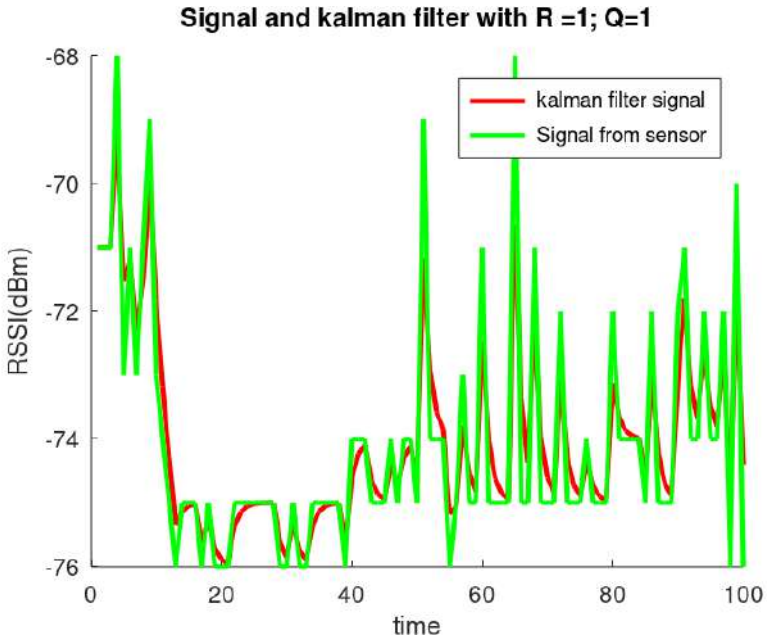
Gambar 3. 6 Rancangan mekanik gateway

3.5 Implementasi Kalman Filter

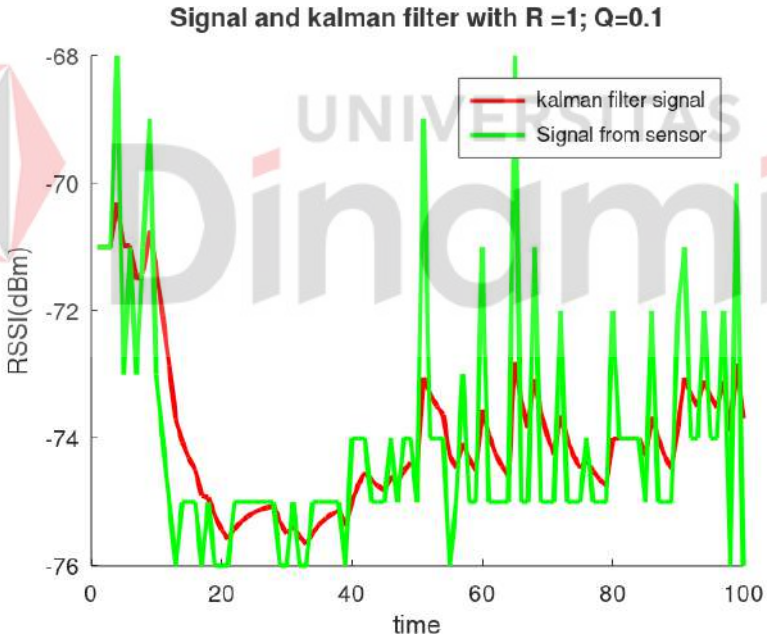


Gambar 3.7 *Flowchart* Kalman *filter*

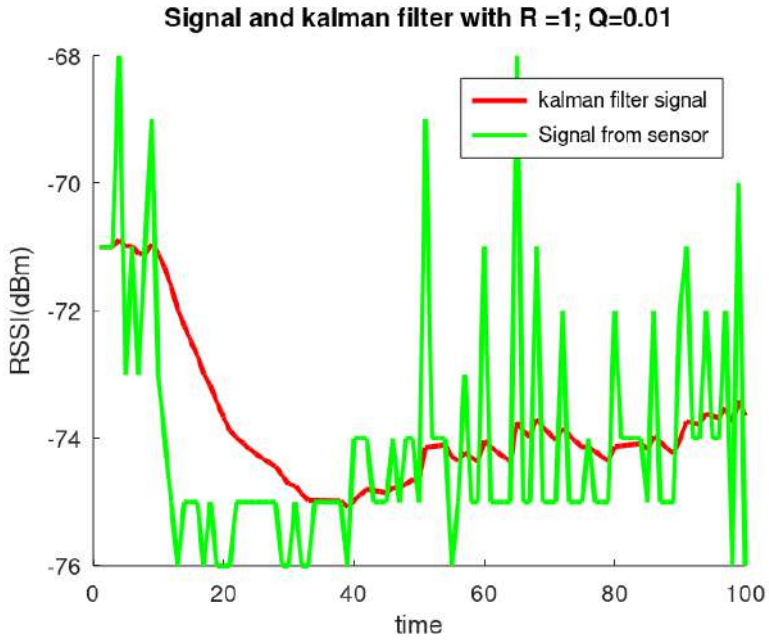
Pada Gambar 3.7 dijelaskan alur dari Kalman *filter*. Dimana adanya inisialisasi R dan Q sebagai tingkat kehalusan filter. Data dari sensor akan dihitung untuk mencari Kt, Xt, dan Pt. yang nantinya Xt akan menjadi keluaran atau hasil dari data yang sudah di*filter*. Pada penelitian ini digunakan nilai R sebesar 1 dan nilai Q sebesar 0,01. Untuk mendapatkan nilai tersebut adalah dengan percobaan dan mau seberapa halus keluaran dari *filter*nya. Nilai R dan Q bisa berbeda beda tiap orang, dikarenakan tingkat kehalusan filter adalah tergantung persepsi masing-masing. Berikut adalah beberapa contoh menggunakan nilai R dan Q yang berbeda yang pernah dicoba.



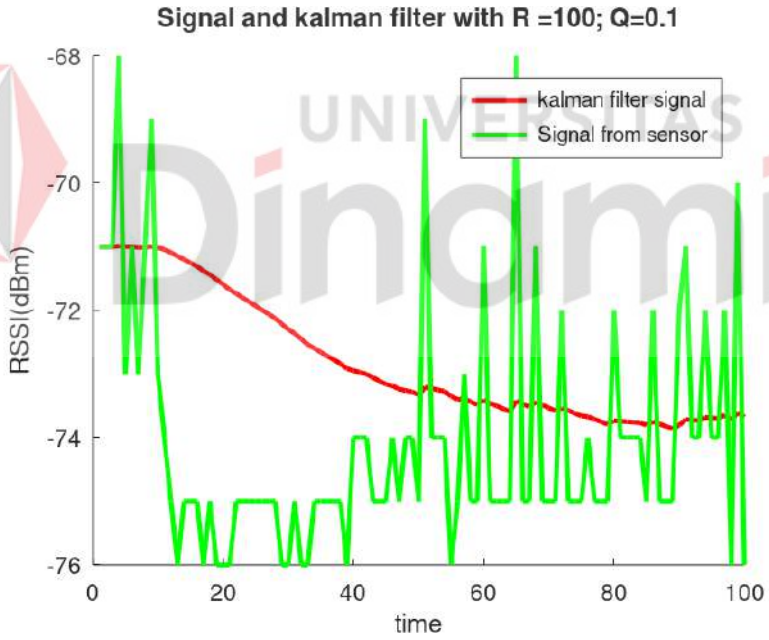
Gambar 3.8 Percobaan kalman *filter* pertama



Gambar 3.9 Percobaan kalman *filter* kedua



Gambar 3.10 Percobaan kalman *filter* ketiga



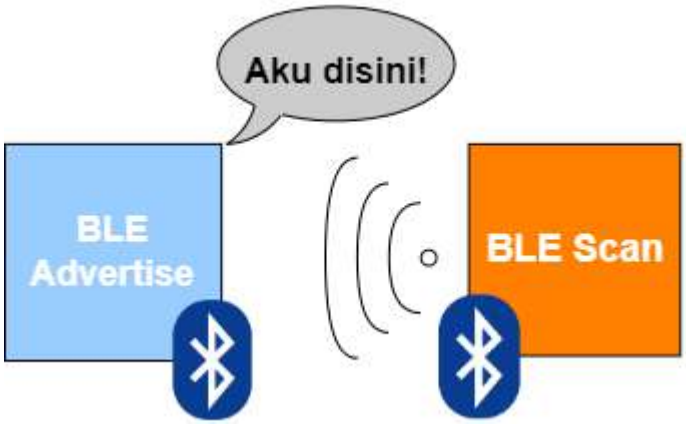
Gambar 3.11 Percobaan kalman *filter* keempat

```
float hasilKalman, Kt, R, Q, Xt_prev, Pt_prev, Xt_update, Xt, Pt_update, Pt;

void setup(){
    Serial.begin(115200);
    R = 1;
    Q = 0.01;
}

void loop(){
    sensorData = analogRead(A0);
    Xt_update = Xt_prev;
    Pt_update = Pt_prev + Q;

    Kt = Pt_update / (Pt_update + R);
    Xt = Xt_update + (Kt * (sensorData - Xt_update));
    Pt = (1 - Kt) * Pt_update;

    Xt_prev = Xt;
    Pt_prev = Pt;

    KalmanFilterData = Xt;
    Serial.print("data asli = ");
    Serial.println(sensorData);
    Serial.print("data kalman = ");
    Serial.println(KalmanFilterData);
    Serial.println()
    delay(200);
}
```

Gambar 3.12 Implementasi Kalman *filter* di arduino IDE

3.6 Protokol komunikasi antar *device*

3.6.1 Komunikasi antar *End node*

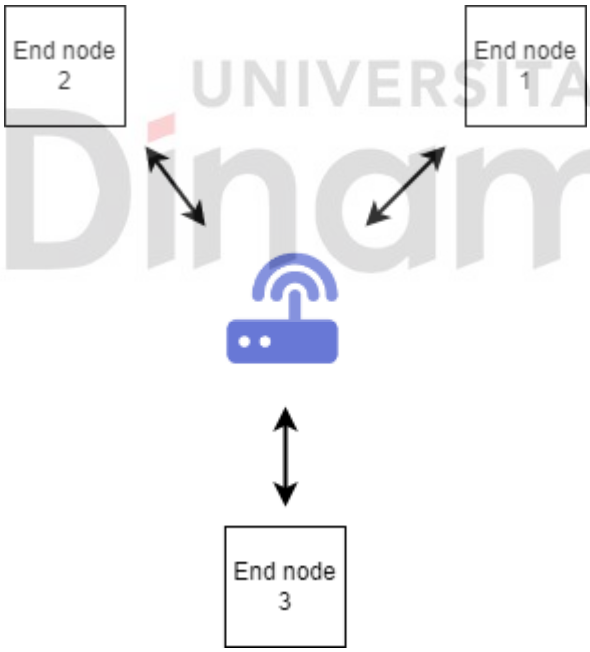


Gambar 3.13 Ilustrasi BLE *Advertised*

BLE *Beacon* digunakan untuk komunikasi antar *End node*. Dimana didalam *Beacon* ada fitur yang dinamakan *advertising*. Maksud dari *advertising* adalah membagikan informasi dirinya ke perangkat lain. Jadi dengan *advertising*, perangkat BLE bisa dikenali atau diketahui keberadaannya oleh BLE lain yang ada disekitarnya, dengan cara *scan* atau memindai *Beacon*. Jika keberadaanya sudah diketahui, maka dengan mudah bisa mengetahui informasi *Beacon* tersebut dari MAC *Address* dan RSSI.

Setiap *device* secara bergantian akan melakukan *advertising* dan *scanning* secara berkala. Dengan cara tersebut bisa menjadikan tiap *End node* dikenali dan mengenali *End node* lain. Program *basic* untuk *advertise* dalam *beacon* ada pada Lampiran 8. Namun untuk implementasi didalam *end node* dapat dilihat pada Lampiran 10 sampai 12

3.6.2 Komunikasi *end node* dengan *gateway*



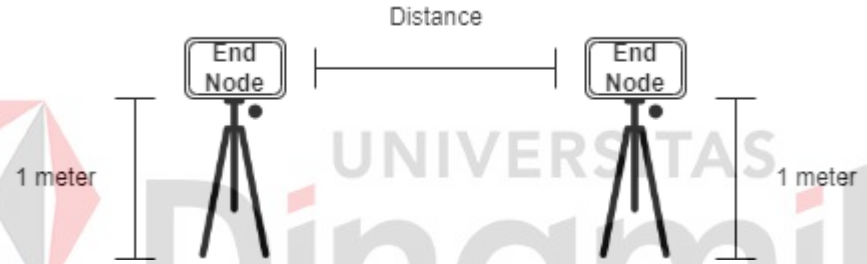
Gambar 3.14 Komunikasi *gateway* dengan *end node*

Gateway mempunyai tugas memanggil setiap *end node* selama 1 detik untuk mengambil data yang dihasilkan oleh *end node*. Ketika data *end node* belum siap, maka *gateway* akan melanjutkan ke *end node* lain. Indikator *end node* belum siap adalah ketika *end node* masih belum mendengarkan panggilan dari *gateway* berarti

end node masih memindai sinyal BLE sekitar. Selain itu, *gateway* juga bertugas untuk membagi data ketika data yang diterima lebih dari satu dengan cara mengidentifikasi simbol awalan dan akhiran yang sudah dijelaskan di struktur pengiriman data. Setelah data diterima, Langkah selanjutnya *gateway* akan kirim data ke server dan disimpan didalam *database*.

Kekurangan dari algoritma alur kerja *gayeway* ini adalah *end node* harus menunggu dipanggil *gateway* untuk mengirim data. Meskipun ketika kondisi semua *end node* siap untuk kirim data, *end node* masih harus menunggu giliran selama kurang lebih 3 detik untuk dipanggil *gateway.*

3.7 Skenario pengambilan data



Gambar 3.15 Posisi antar *end node*

Pengambilan sampel RSSI dengan jarak 1 meter dan 2 meter sebagai referensi untuk mendapatkan koefisien *path loss* (*n*). Persamaan koefisien *path loss* ditunjukkan pada (18).

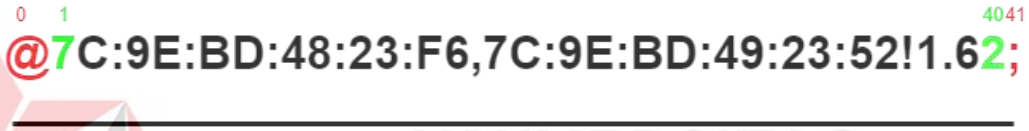
$$n = \frac{RSSI_{d0} - RSSI}{10Log_{10}(d)} \quad \text{..... (18)}$$

Berdasarkan pengambilan data di lantai 12 kampus Universitas Dinamika, didapatkan median dari RSSI yang sudah terfilter dengan jarak 1 meter adalah sebesar $-65{,}79$ dBm. Sedangkan untuk RSSI jarak 2 meter sebesar $-74{,}74$ dBm. Selanjutnya dengan nilai RSSI tersebut nilai n didapatkan sebesar $2{,}97$.

End node akan bekerja selama 10 detik untuk memindai Bluetooth yang ada disekitarnya. Setelah 10 detik, selanjutnya akan menunggu panggilan dari *gateway* untuk kirim data ke *gateway*. Pengiriman format data ditunjukkan pada Gambar 3.16.

Gambar 3.16 Struktur pengiriman paket data

Gambar 3.16 menjelaskan tentang struktur paket data yang dikirimkan dari *end node* ke *gateway*. Dimana didalamnya mengandung 'simbol' yang diharapkan agar *gateway* mengerti awalan dan akhiran dari data yang dikirimkan. ID Bluetooth Rx yaitu MAC dari penerima, sedangkan ID Bluetooth Tx yaitu MAC dari pengirim. Keduanya digunakan untuk identitas sebagai penerima dan pengirim yang nantinya akan digunakan untuk sistem *history* didalam *database*, yaitu *history* dengan siapa *end node* melanggar ketentuan *physical distancing*. Jarak mengindikasi berapa jarak antara ID Bluetooth Rx dan ID Bluetooth Tx.



Gambar 3.17 Struktur *Index* paket data

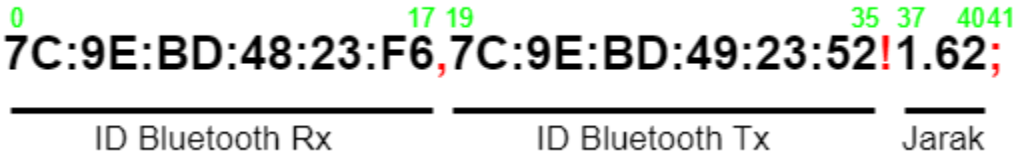
Jumlah data yang dikirim mempunyai total panjang 42 *byte*. Dimana simbol untuk awalan berada pada *index* 0 dan simbol akhiran berada pada *index* 41. *Gateway* akan menggunakan fungsi *substring(1, 41)* untuk mengambil data dari *index* 1 hingga 40.

Jika jumlah data yang dikirim adalah 2, maka total panjang data yang dikirim sebanyak 84 *byte*, untuk bisa memisah data menjadi 2 diperlukan fungsi *substring* sebanyak 2x juga. Yang pertama *parsing* data pertama menggunakan fungsi *substring(1, 41)* untuk mengambil data yang mempunyai *index* 1 hingga 40. Selanjutnya dipastikan apakah setelah index 41 sudah tidak ada data lagi. Jika ada data maka akan dilakukan *parsing* data kedua menggunakan fungsi *substring(43, 83)* untuk mengambil data yang mempunyai *index* 43 hingga 82. Untuk pemaparan *index* dapat dilihat di Lampiran 1.

Pada Lampiran 2 dipaparkan penerimaan data dari *gateway* dan *parsing* data menggunakan *substring* yang sudah dijelaskan sebelumnya. Untuk garis merah

adalah hasil dari *parsing* data. sedangkan untuk garis hijau adalah kumpulin data final yang sudah diterima. Setelah itu data akan dilanjutkan ke server untuk *parsing* data ID, dan jarak.

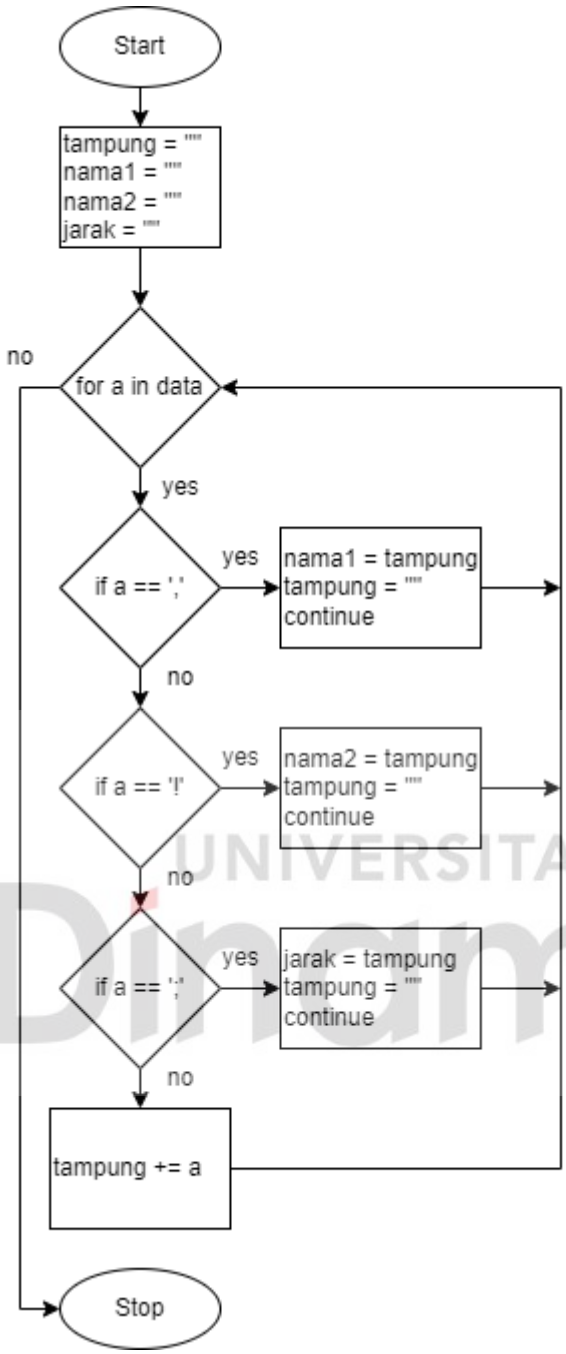
3.8 *Parsing* data server



Gambar 3.18 *Parsing index* data server

Parsing data dilakukan berdasarkan simbol yang ada *marker* merah seperti pada Gambar 3.13 Dimana tanda koma (,) yang pertama untuk ID Bluetooth Rx. Untuk tanda seru (!) untuk ID Bluetooth Tx. Tanda titik koma (;) untuk jarak.

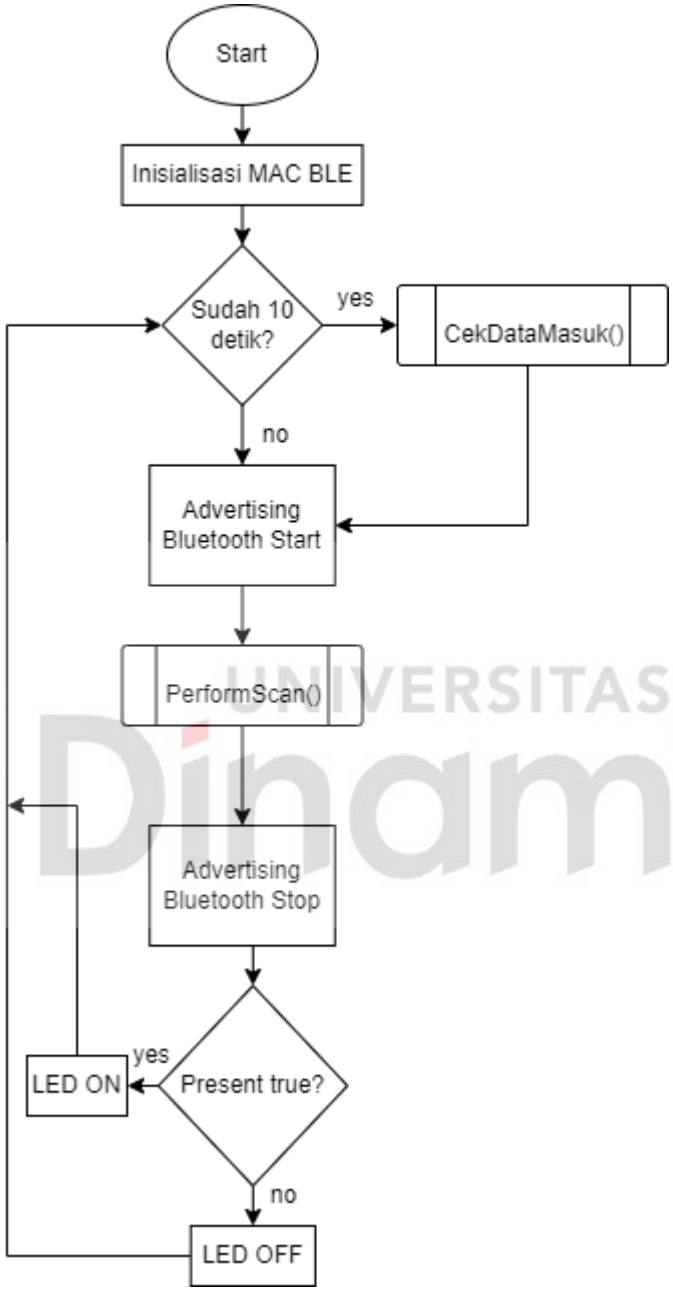
Index 0 sampai 17 akan disimpan didalam variabel ‘nama1’. *Index* 19 sampai 35 akan disimpan didalam variabel ‘nama2’. *Index* 37 sampai 40 akan disimpan didalam vairabel ‘jarak’.



Gambar 3.19 *Flowchart parsing* data server

Data akan dilakukan perulangan setiap *index*nya. Jika *value index* tidak sama dengan simbol seperti yang dijelaskan di Gambar 3.14, maka *value* dari *index* tersebut akan dimasukkan kedalam variabel ‘tampung’. Namun jika *value index* sama dengan simbol, maka *value* dari variabel ‘tampung’ akan dimasukkan kedalam variabel yang sesuai dengan kondisi percabangan. Selanjutnya variabel tersebut akan dikirim ke *website.*

3.9 Algoritma *end node*

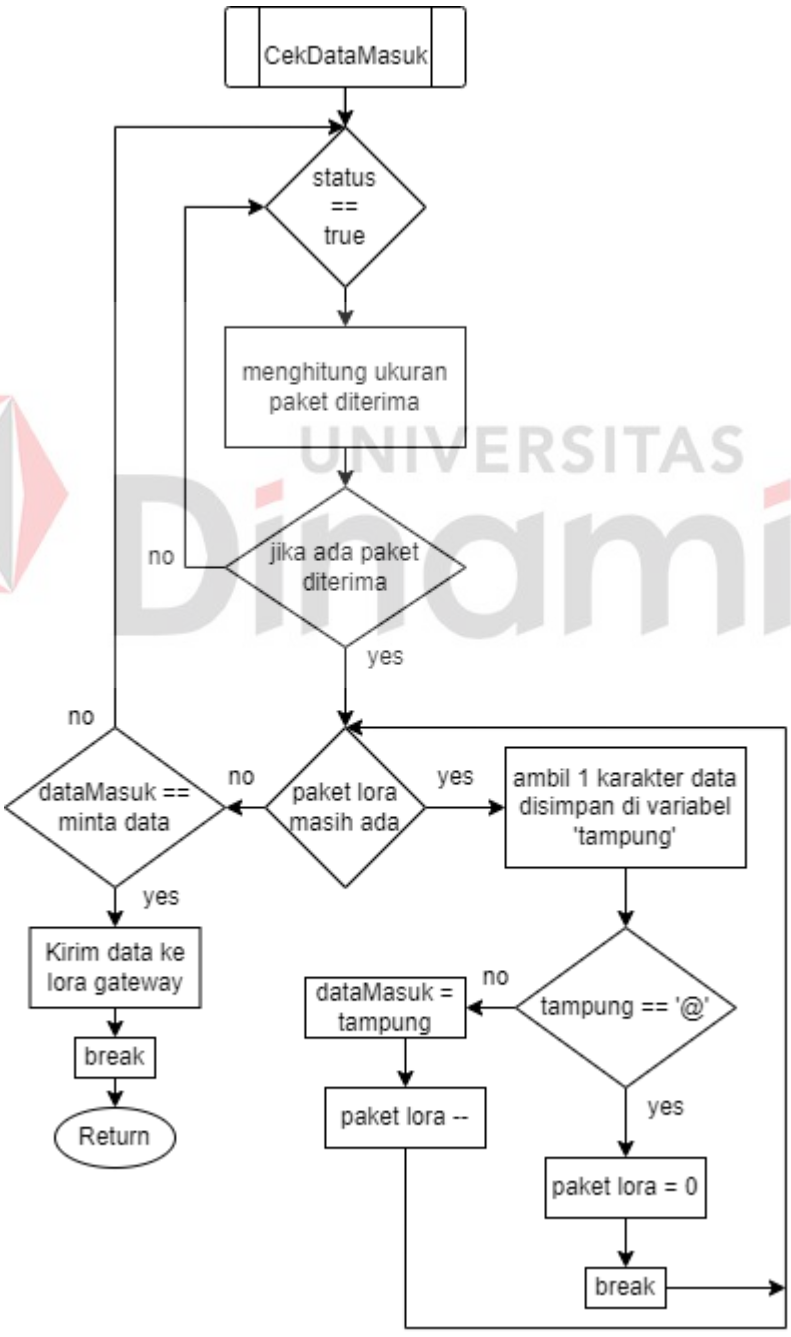


Gambar 3.20 *Flowchart end node*

Pada Gambar 3.16 menjelaskan *flowchart* dari *end node.* Mikro akan memastikan apakah waktu sudah berjalan selama 10 detik. Jika belum maka *adverstising* Bluetooth akan dimulai untuk mencari *device* BLE disekitar. Untuk kegiatan *scan* ada didalam fungsi *PerformScan().* Jika sudah menjalankan

PeformScan() maka *advertise* Bluetooth akan dihentikan lalu akan dicek apakah ada data dengan jarak dibawah 2 meter. Jika ada maka LED akan *ON*. Sebaliknya jika tidak ada, maka LED akan *OFF*. Jika Mikro sudah berjalan selama 10 detik, maka Mikro akan menunggu dipanggil oleh *gateway* LoRa untuk pengiriman data.

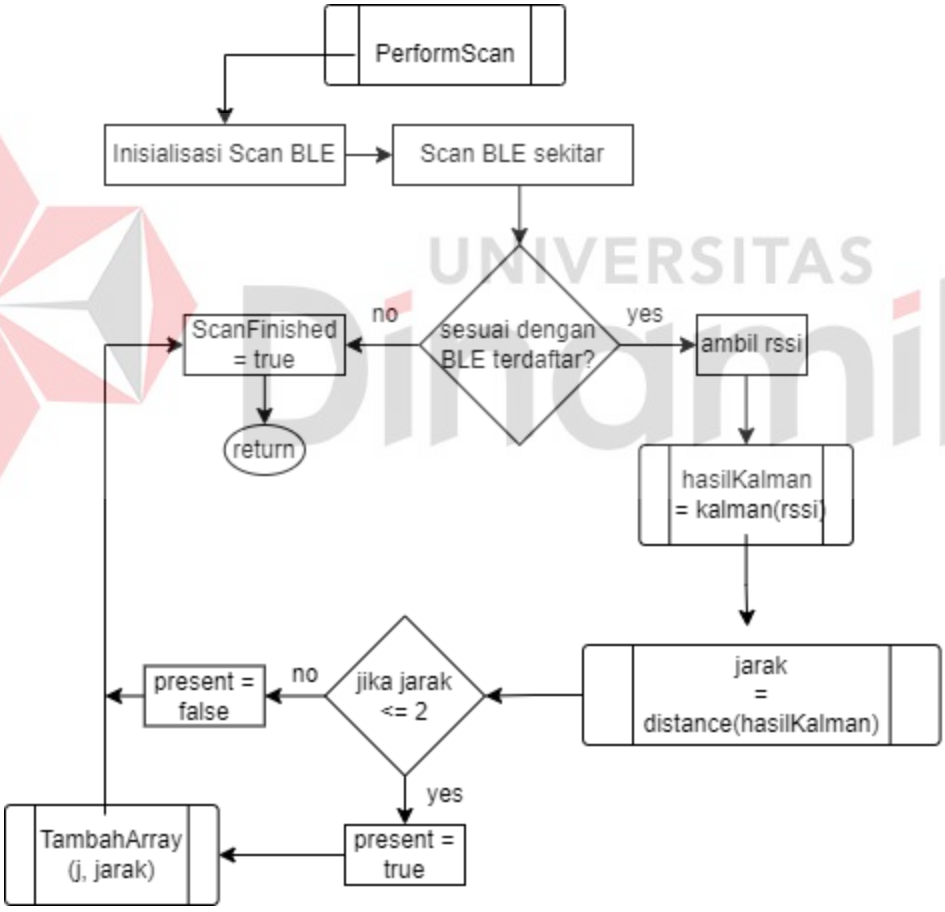
3.9.1 Fungsi CekDataMasuk



Gambar 3.21 *Flowchart* fungsi *cekDataMasuk()*

Didalam fungsi *CekDataMasuk()* ada pengulangan dimana untuk menunggu *gateway* memanggil *End node* untuk mengirim data. karena jika tidak menggunakan cara perulangan, terkadang ketika *gateway* sudah memanggil *End node* untuk meminta data namun *End node* tidak bisa menangkap pesan tersebut dikarenakan alur program sedang ada proses lain. Proses akan keluar dari perulangan jika *End node* sudah kirim data ke *gateway.*

3.9.2 Fungsi *PerformScan*

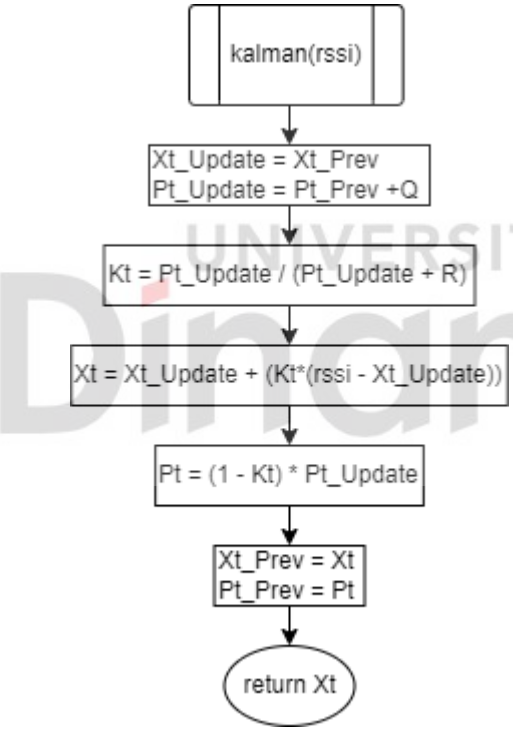


Gambar 3.22 *Flowchart* fungsi *PerformScan()*

Didalam fungsi *PerformScan* terdapat proses pengambilan RSSI, Kalman *Filter*, dan konversi RSSI ke jarak menggunakan *path loss model.* Proses pertama dalam fungsi ini adalah dengan inisialisasi *scan* BLE. Setelah inisialisasi akan

dilakukan *scanning* terhadap BLE sekitar. Lalu akan dilakukan pencocokan MAC BLE *new* dengan MAC BLE yang sebelumnya sudah didaftarkan sebagai *End node.* Jika MACnya sama, maka akan diambil RSSInya lalu akan dilakukan *filter* Kalman untuk mendapatkan RSSI yang stabil. Setelah mendapatkan RSSI Kalman, selanjutnya mengkonversi RSSI menjadi jarak dengan memanggil fungsi *distance()* untuk menerapkan *path loss model.* Lalu akan dibandingkan hasilnya, apakah jarak kurang dari 2 meter? Jika iya *present* sama dengan *true.* Sebaliknya maka *present* sama dengan *false.*

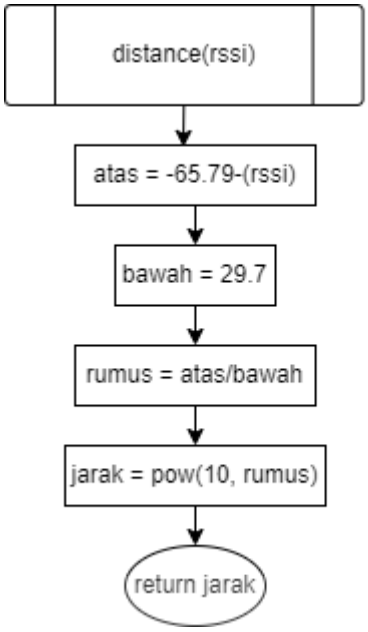
3.9.3 Fungsi Kalman



Gambar 3.23 *Flowchart* fungsi *Kalman()*

Didalam fungsi Kalman terdapat beberapa rumus yang sudah dijelaskan di BAB II tinjauan pustaka. Variabel R dan Q adalah variabel yang telah ditentukan untuk tingkat kehalusan *filter*. Dimana nantinya akan dibutuhkan variabel Xt untuk mengetahui hasil dari RSSI Kalman.

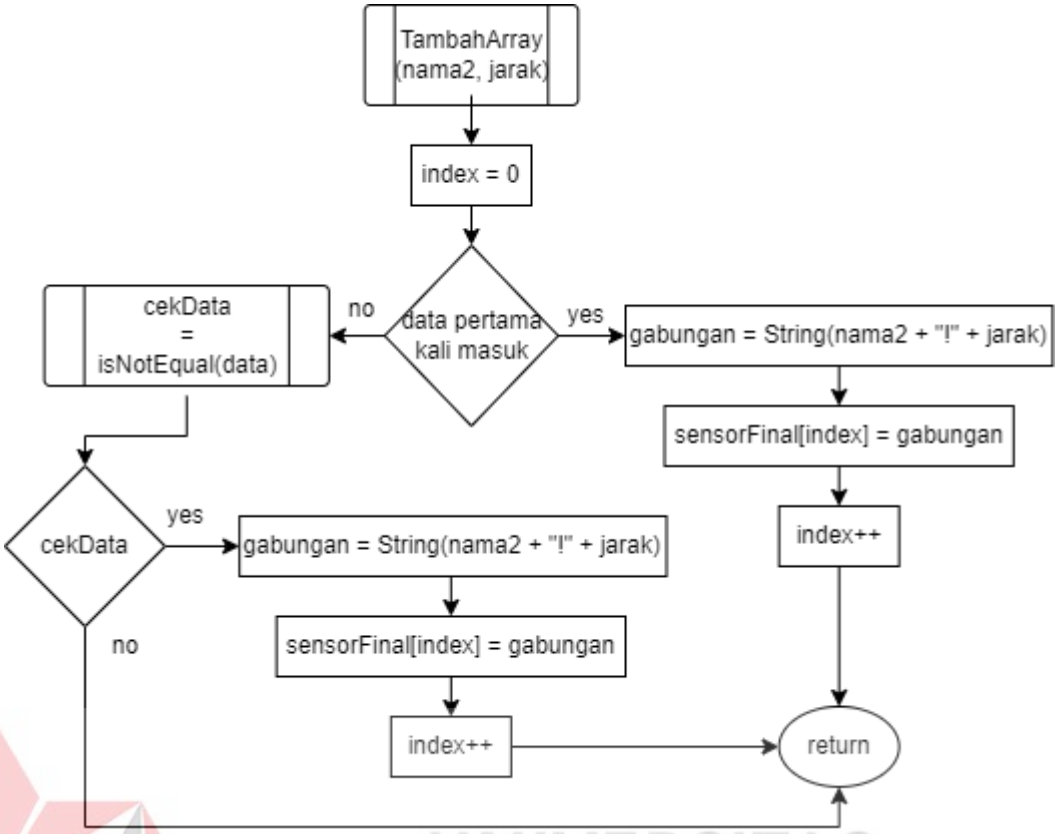
3.9.4 Fungsi *distance*



Gambar 3.24 *Flowchart* fungsi *distance()*

Didalam fungsi *distance* ada sebuah perhitungan rumus *log distance path loss model* seperti yang dijelaskan di persamaan (2). -65.79 sebagai RSSI referensi pada jarak 1 meter dan 29.7 adalah nilai 10n yang didapat dari persamaan (18).

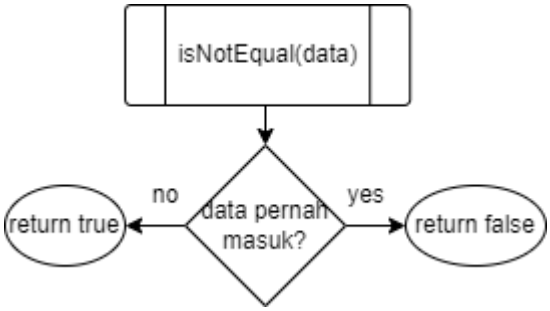
3.9.5 Fungsi Tambah*Array*



Gambar 3.25 *Flowchart* fungsi *TambahArray()*

Didalam fungsi Tambah*Array* terdapat penyimpanan sebuah data yang telah diterima oleh *End node*. Jika data pertama kali masuk, maka data langsung disimpan dalam sebuah *array*. Namun jika data tidak pertama kali, maka data akan dibandingkan terlebih dahulu dengan data yang pernah masuk sebelumnya. Apakah data pernah masuk atau belum. Jika data pernah masuk, maka data akan diabaikan. Namun jika data belum pernah masuk, maka data akan ditambahkan didalam sebuah *array*.

3.9.6 Fungsi *isNotEqual*



Gambar 3.26 *Flowchart* fungsi *isNotEqual()*

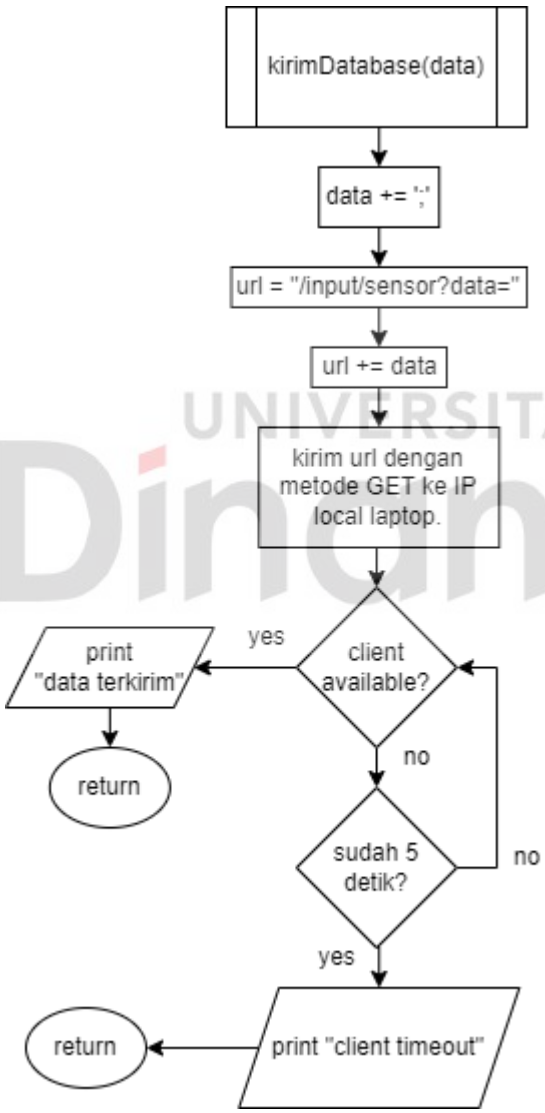
Didalam fungsi *isNotEqual* adalah perbandingan antara data yang baru dengan data yang pernah masuk sebelumnya. Jika data pernah masuk sebelumnya maka akan fungsi akan *return false.* Jika sebaliknya maka fungsi akan *return true.*

3.10 Algoritma gateway

Gambar 3.27 *Flowchart gateway*

Gambar 3.23 menjelaskan *flowchart gateway*. Dimana *gateway* setiap 1 detik akan meminta data kepada setiap *end node.* Jika data belum siap, *gateway* tetap melanjutkan memanggil *end node* lain. Namun jika *gateway* menerima data, maka *gateway* akan *parsing* data tersebut sesuai dengan simbol yang ada dan segera mengirimkan ke server untuk disimpan didalam *database*.

3.10.1 Fungsi kirim*Database*

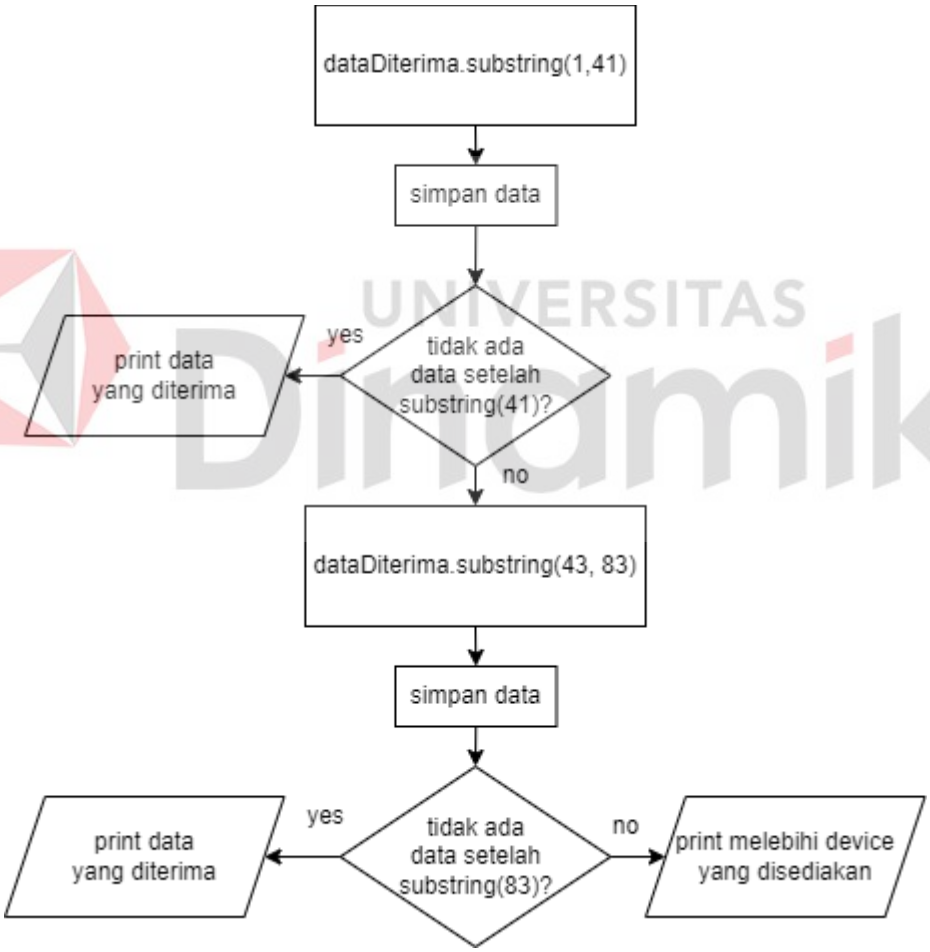


Gambar 3.28 *Flowhcart* fungsi *kirimDatabase()*

Didalam fungsi *database* terdapat pengiriman data yang diterima oleh *gateway* ke *database.* Untuk kirim ke *database* terdapat beberapa variabel yang

dibutuhkan, seperti alamat *web* yang dituju yaitu "/input/sensor?data=". lalu pada alamat tersebut akan ditambahkan data yang akan dikirim. Selanjutnya akan dipastikan apakah *client available.* Jika tidak ada, akan ditunggu selama 5 detik. Jika sudah 5 detik tetapi masih belum ada, maka sudah dipastikan bahwa *client timeout.* Namun jika *client available,* maka data sudah dipastikan dikirim. Untuk *web server* pengiriman adalah IP local komputer. Jadi *gateway* harus berada pada 1 jaringan yang sama dengan komputer untuk melakukan pegiriman data.

3.10.2 Proses *parsing* data



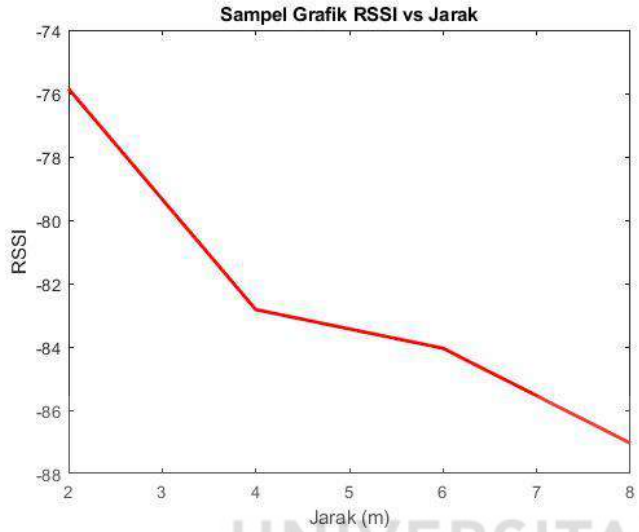
Gambar 3.29 *Flowchart* proses *parsing* data

Flowchart proses *parsing* data menjabarkan alur yang sudah dijelaskan pada sub bab struktur pengiriman data. Dimana data yang diterima akan di *parsing* menggunakan fungsi *substring index* ke 1 hingga 41. Jika tidak ada data setelah

index 41, maka data akan melakukan output berupa serial *print* untuk melihat data yang berhasil di*parsing.* Namun jika setelah *index* 41 masih ada data, maka akan dilakukan lagi proses *substring index* ke 43 hingga 83. Jika tidak ada data setelah *index* 83, maka data akan melakukan output berupa serial *print* untuk melihat data yang berhasil di*parsing.* Namun jika masih ada data setelah *index* 83, akan melakukan serial *print* yang menandakan bahwa melebihi *device* yang disediakan.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Perbandingan RSSI terhadap jarak



Gambar 4.1 Perbandingan RSSI terhadap jarak

Gambar 4.1 diambil dengan cara *point to point* dan dilakukan terhadap jarak 2,4,6, dan 8 meter. Pengambilan *point to point* antara *hastag* H2 dan H3 dilakukan dilantai 12 kampus Universitas Dinamika. Data *raw* dari pengambilan tersebut bisa dilihat pada Lampiran 13. Untuk hasil pengolahan data dapat dilihat di Tabel 4.1.

Tabel 4.1 Tabel pengujian RSSI terhadap jarak

Jarak	**Rata rata RSSI (dBm)**	
	H2-H3	**H3-H2**
2 Meter	-75,98	-75,85
4 Meter	-84,10	-82,83
6 Meter	-83,60	-84,05
8 Meter	-90,93	-87,04

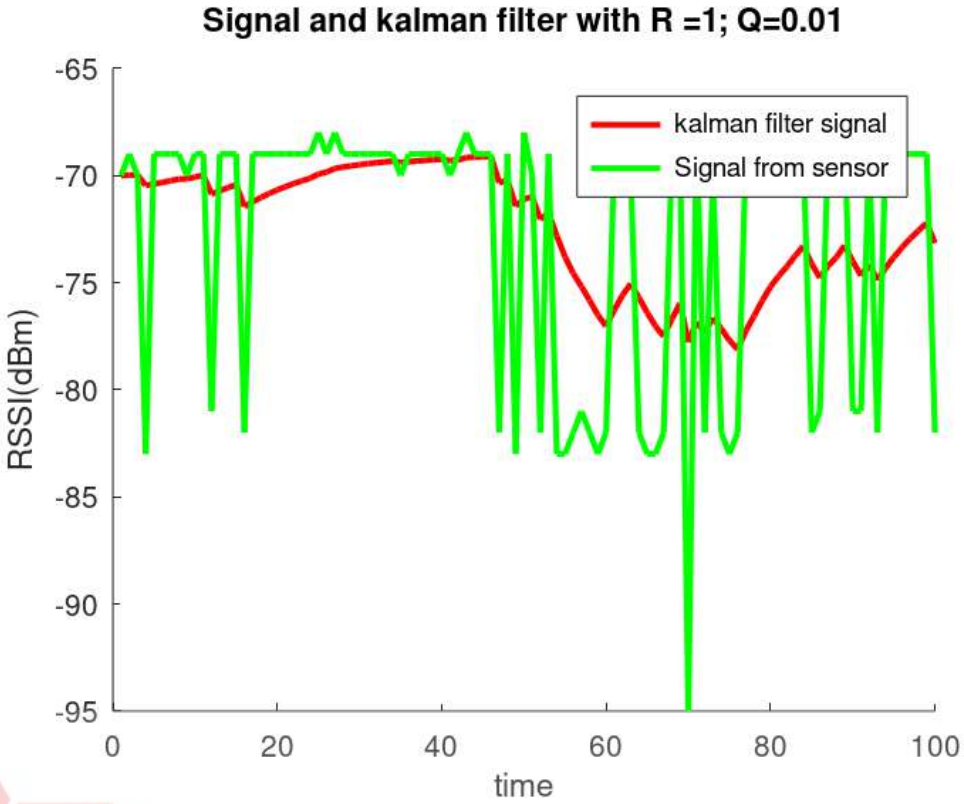
Masing masing pengambilan data dilakukan selama 10 menit dan didapatkan sekitar 445 data yang diolah didalam *excel.* Hasil dari tabel 4.1 adalah rata rata dari

445 data menggunakan rumus *average()* di excel. Dari tabel dan grafik tersebut dapat membuktikan bahwa jarak sangat berpengaruh terhadap sinyal RSSI. Jika jarak semakin jauh, maka RSSI akan semakin kecil nilainya.

Gambar 4.2 Pengujian jarak 8 meter

4.2 Hasil Pengambilan RSSI Kalman *Filter*

Pengambilan RSSI akan dibedakan menjadi 2 jenis. RSSI tanpa *filter* dan dengan *filter*.



Gambar 4.3 RSSI dan RSSI Kalman

Gambar 4.3 diperoleh dari pengambilan data sejauh 2 meter. Terlihat grafik warna hijau dengan nilai asli RSSI tidaklah stabil. Seperti yang sudah disinggung didalam karya ilmiah (Shue & Conrad, 2016) dijelaskan bahwa tantangan terbesar dari estimasi jarak di lingkungan *indoor* menggunakan RSSI adalah faktor *multipath fading. Multipath fading* adalah sinyal yang ter-refleksi oleh lantai, dinding dan lingkungan sekitar. Oleh karena itu, Kalman filter berperan penting untuk meredam sinyal yang tidak stabil tersebut.

Perocbaan dilakukan dengan nilai $R = 1$ dan $Q = 0{,}01$ untuk variabel kehalusan keluaran dari Kalman *Filter*. Selain grafik, hasil olah data Kalman *filter* bisa dilihat pada Tabel 4.2. dengan data *raw* terdapat pada Lampiran 14. Dijelaskan pada Tabel 4.2 untuk mencari efisiensi Kalman *filter* dengan membandingkan selisih dari hasil konversi jarak *filter* maupun tidak *filter* dengan 2 meter sebagai jarak *set point.* Lalu dicari *error* (%) dengan persamaan (19).

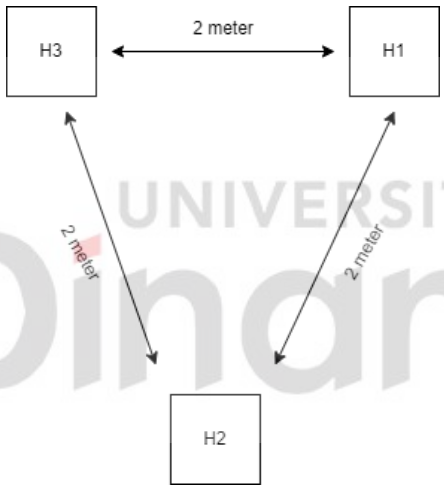
$$\frac{Rata{-}rata\ selisih\ terhadap\ 2\ meter}{2} x100 \quad \text{..... (19)}$$

Tabel 4.2 Hasil olah data RSSI tanpa *filter* & RSSI *filter*

Rata-rata selisih terhadap 2 meter	**1,132**
Error %	**56,579**
Rata-rata selisih terhadap 2 meter *filter*	**0,279**
Error %	**13,930**

Dari grafik pada Gambar 4.3 dan Tabel 4.2 membuktikan bahwa konversi jarak dengan menggunakan Kalman *Filter* lebih baik 42,649% daripada konversi jarak dengan tidak menggunakan Kalman *filter*.

4.3 Pengujian jarak *End node*



Gambar 4.4 Skenario pengujian jarak *End node*

Pengujian dilakukan dengan 3 *end node* yang sudah diukur jaraknya masing masing sejauh 2 meter. Untuk variabel yang dibutuhkan dari hasil perhitungan jarak referensi adalah RSS_{d0} sebesar $-65{,}79$ dan n sebesar 2,97. Variabel referensi tersebut didapat dari pengambilan referensi pada lantai 12 kampus Universitas Dinamika.

Gambar 4.5 Pengambilan data jarak Bluetooth

Pengujian dilakukan di kampus lantai 12 yang dimana lantai tersebut tidak ada sinyal WiFi sama sekali dan dilakukan pengujian tanpa penghalang antara *end node*. Pengambilan data diambil selama 10 menit.

Data *raw* yang dihasilkan dari lantai 12 ada pada Lampiran 3 dan 4. Pada kumpulan data tersebut terdapat RSSI sebelum *filter*, RSSI sudah ter-*filter*, dan konversi jarak. Jarak yang dihasilkan oleh mikrokontroler adalah berdasarkan RSSI yang sudah ter*filter* oleh Kalman. Dimana data tersebut akan diolah seperti Tabel 4.3.

Tabel 4.3 Pengolahan data *multi nodes*

END NODE	**RATA RSSI**	**MEDIAN RSSI**	**RATA RSSI *FILTER***	**MEDIAN RSSI *FILTER***	**RATA JARAK**
H1-H2	-74,34	-76,00	-74,33	-74,51	1,94
H2-H1	-74,37	-75,00	-74,37	-74,35	1,95
H2-H3	-77,26	-78,00	-77,25	-77,28	2,43
H3-H2	-77,59	-78,00	-77,59	-77,63	2,50
H3-H1	-87,28	-92,00	-87,20	-87,19	5,33
H1-H3	-74,97	-70,00	-74,92	-75,12	2,06

Rata RSSI adalah rata-rata dari RSSI belum ter*filter* menggunakan rumus *excel average().* Median RSSI adalah median dari RSSI yang belum ter*filter* menggunakan rumus *excel median().* Rata RSSI *Filter* adalah rata-rata dari RSSI yang sudah ter*filter* Kalman menggunakan rumus *excel average().* Median RSSI

Filter adalah median dari RSSI yang sudah ter*filter* Kalman menggunakan rumus *excel median().* Rata Jarak adalah rata-rata dari jarak yang diketahui dari keluaran mikrokontroler menggunakan rumus *excel average().*

Dari data olahan pada Tabel 4.3 dapat dicari *Error* dari jarak sebenarnya dengan jarak yang dikonversi oleh mikrokontroler.

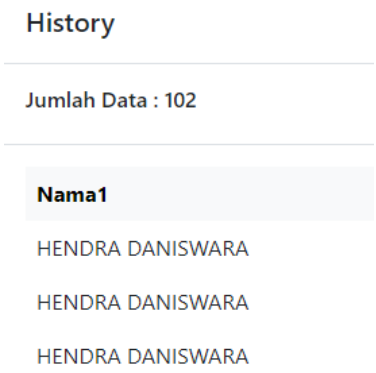
Tabel 4.4 Tabel pengujian jarak bluetooth

End node	**Jarak *real* (m)**	**Rata Rata RSSI Kalman**	**Jarak Rata Rata RSSI Kalman (m)**	***Error* (%)**
H1-H2	2	-74,33	1,94	2,83
H2-H1	2	-74,37	1,95	2,75
H2-H3	2	-77,25	2,43	21,67
H3-H2	2	-77,59	2,5	24,99
H3-H1	2	-87,2	5,33	166,30
H1-H3	2	-74,92	2,06	3,06

Hasil dari pengujian pertama untuk estimasi jarak antar 3 *end node* pada lantai 12 kampus Universitas Dinamika mempunyai *error* terkecil sebesar 2,75% dan *error* terbesar 166,3%. Namun *error* 166,3% diduga adalah data pencilan. Pencilan adalah data yang nilainya jauh dari pola data lainnya (Indra, et al., 2013). Bisa dikarenakan *start* dan *stop* dalam pengambilan data tidak sama, atau bisa disebut karena *human error* dalam pengambilan data.

4.4 Pengujian transmisi data ke *web*

Pengujian dilakukan selama kurang lebih 19 menit dengan jarak kurang lebih 1 meter. Didapat sebanyak 308 data yang sudah diterima dan dikirim ke server. Dan yang diterima MongoDB juga dengan jumlah yang sama yaitu 308 data yang disimpan. Gambar bisa dilihat pada Lampiran 5 dan 6.



Gambar 4.6 Jumlah data spesifik di *website*

Pengujian selanjutnya yaitu testing apakah data dengan nama ‘HENDRA DANISWARA’ jumlahnya sesuai dengan yang disimpan didalam *database*. Jumlah dari data di *website* terhitung 102 data yang Bernama ‘HENDRA DANISWARA’. dan di*database* pada Lampiran 7 juga sama yaitu berjumlah 102 data. Dengan pengujian selama 19 menit, bisa disimpulkan pada jarak kurang lebih 1 meter, data LoRa berhasil dikirim dan diterima dengan keberhasilan 100%. Gambar penerimaan data pada *database* dapat dilihat di Lampiran 7.

BAB V
PENUTUP

5.1 Kesimpulan

Dari hasil beberapa pengambilan data, didapatkan beberapa kesimpulan, diantara lain:

1. Konversi jarak menggunakan model *distance path loss* yang memanfaatkan RSSI sebagai parameter pengukuran mempunyai *error* terkecil pada penerapan model tersebut sekitar 2,75%. Hal ini disebabkan oleh kondisi lingkungan yang menyebabkan *noise*, pantulan, atau peredaman sinyal.
2. Kalman *filter* dapat meminimalisir tingkat kesalahan konversi jarak dengan RSSI. Lonjakan RSSI diredam sebesar 42,649%. Dengan itu, Kalman *filter* dapat membuat jarak lebih stabil dan kesalahan konversi jarak dapat diperkecil.
3. Penelitian ini berhasil mengirimkan data dari *end node* ke server *database* menggunakan komunikasi LoRa, pada jarak pengiriman data kurang lebih 1 meter dengan tingkat keberhasilan 100% tanpa ada data yang loss.

5.2 Saran

Adapun saran untuk mengembangan penelitian ini agar lebih baik, terdapat beberapa saran yaitu:

1. Mencoba menggunakan Bluetooth external. Mungkin saja bisa mendapatkan akurasi yang lebih baik daripada Bluetooth *builtin* ESP32.
2. Mencoba pendekatan lain untuk konversi RSSI ke jarak. Seperti pendekatan *triangular*.
3. Ambil data di tempat yang minim *noise* dan loss. Jauh dari dinding, pilar, dan halangan lain.

DAFTAR PUSTAKA

Anon., 2022. *ESP32-WROOM-32 Datasheet.* [Online] Available at: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

Dharmawan, W., Kurnianto, A. & Ar-Rasyiid, A., 2016. Peningkatan Akurasi Estimasi Jarak RSSI Dengan Model Log Normal Menggunakan Metode Kalman Filter Pada Bluetooth Low Energy. *Seminar Nasional Sains dan Teknologi,* 8 November.pp. 1-5.

Hadi, S., Widayaka, P. D., M.D.L, R. P. & Diharja, R., 2020. Pengukuran Jarak Pada Mobile Robot Menggunakan Xbee Berdasarkan Nilai Receive Signal Strength Indicator (RSSI). *Jurnal Bumigora Information Technology,* pp. 66-70.

Indra, S., Vionanda, D. & Sriningsih, R., 2013. Pendeteksian Data Pencilan dan Pengamatan Berpengaruh pada Beberapa Kasus Data Menggunakan Metode Diagnostik. *Journal Of Mathematics UNP,* 1(2), pp. 67-74.

Ma'arif, A., Iswanto, Nuyono, A. A. & Alfian, R. I., 2019. Kalman Filter for Noise Reducer on Sensor Readings. *Signal and Image Processing Letters,* pp. 50-61.

Mohammed, S. L., 2016. Distance Estimation Based on RSSI and Log-Normal Shadowing Models for ZigBee Wireless Sensor Network. *Eng. &Tech.Journal,* pp. 2950-2959.

nesr, 2020. *Apa itu LoRa?.* [Online] Available at: https://nesr.labs.telkomuniversity.ac.id/apa-itu-lora/#:~:text=LoRa%20%28Long%20Range%29%20adalah%20teknik%20modulasi%20radio%20yang,berlisensi.%20Wahh%2C%20pasti%20kalian%20bingung%20kan%20dengan%20penjelasannya.
[Diakses 3 september 2022].

Santos, R., 2019. *Getting Started with ESP32 Bluetooth Low Energy (BLE) on Arduino IDE.* [Online]

Available at: https://randomnerdtutorials.com/esp32-bluetooth-low-energy-ble-arduino-ide/

Shue, S. & Conrad, j. M., 2016. *Reducing the Effect of Signal Multipath Fading in RSSI-Distance Estimation using Kalman Filters.* San Diego, Society for Computer Simulation International.

Suroso, D. J., Arifin, M. & Cherntanomwong, P., 2020. Distance-based Indoor Localization System Utilizing General Path Loss Model and RSSI. *Journal of Robotics and Control,* pp. 199-207.

Telkom, 2019. *Pengukuran Jarak Antar Node Menggunakan X-Bee.* Surabaya: Politeknik Elektronika Negeri Surabaya.