

IMPLEMENTASI SISTEM *GATEWAY DISCOVERY* PADA *WIRELESS SENSOR NETWORK (WSN)* BERBASIS MODUL KOMUNIKASI LORA

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Alfian Rizky Susanto
NIM: 145150201111024



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI SISTEM GATEWAY DISCOVERY PADA WIRELESS SENSOR NETWORK (WSN) BERBASIS MODUL KOMUNIKASI LORA

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh :
Alfian Rizky Susanto
NIM: 145150201111024

Skripsi ini telah diuji dan dinyatakan lulus pada
17 Desember 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I



Adhitya Bhawiyuga, S.Kom., M.Sc.
NIP: 19890720 201803 1 002

Dosen Pembimbing II



Kasyful Amron, S.T., M.Sc.
NIP: 19750803 200312 1 003

Mengetahui

Ketua Jurusan Teknik Informatika



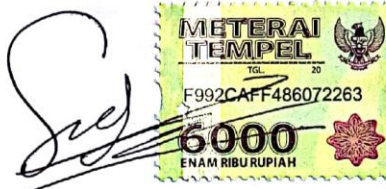
Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 17 Desember 2018



Alfian Rizky Susanto

NIM: 145150201111024

KATA PENGANTAR

Puji syukur penulis panjatkan kehadiran Tuhan Yang Maha Esa, karena atas berkat, rahmat dan hidayah-Nya penulis dapat menyelesaikan penulisan skripsi berjudul “Implementasi Sistem Gateway Discovery pada Wireless Sensor Network (WSN) Berbasis Modul Komunikasi Lora” dengan lancar. Penulisan skripsi ini ditujukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer pada Proram Studi Teknik Informatika Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.

Dalam penyusunan dan penulisan skripsi ini, penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari seluruh pihak yang telah memberikan bantuan, dukungan dan bimbingan selama proses pembuatan skripsi ini. Oleh karena itu, dalam kesempatan ini penulis ingin menyampaikan rasa hormat dan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Kedua Orang tua penulis, Bapak Suroso dan Ibu Nuryati yang selalu mendukung, mengerti, berdoa untuk penulis agar dapat menyelesaikan penelitian dan penulisan skripsi ini.
2. Bapak Adhitya Bhawiyuga, S.Kom., M.S dan Bapak Kasyful Amron, S.T., M.Sc selaku dosen pembimbing penulis yang telah dengan sabar membimbing, mengarahkan, serta meluangkan waktu dan pikiran untuk penulis sehingga dapat menyelesaikan skripsi ini,
3. Seluruh dosen dan civitas Program Studi Informatika Universitas Brawijaya ataskesediaan membagi ilmunya kepada penulis,
4. Teman-teman Sellow yang selalu bersedia membantu dan mendukung penulis.
5. Teman-teman angkatan 2014 serta para senior dan junior atas segala bantuan dan kenangan selama menempuh studi di Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya,
6. Perempuan tangguh yang selalu mendukung dan memberi semangat lebih untuk saya,
7. Dan seluruh pihak yang telah membantu kelancaran penulisan skripsi yang tidak dapat penulis sebutkan satu persatu.

Semoga Allah SWT selalu melimpahkan rahmat dan karunia-Nya kepada semua pihak yang telah memberikan bantuan kepada penulis dalam menyelesaikan penulisan dan penyusunan skripsi ini.

Penulis juga menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun senantiasa penulis harapkan demi perbaikan skripsi in. Semoga skripsi ini dapat memberikan manfaat bagi penulis, pihak yang terlibat dan semua pihak membacanya.

Malang, 17 Desember 2018

Penulis

alfianrizky14@gmail.com

ABSTRAK

Wireless Sensor Network (WSN) merupakan suatu jaringan nirkabel dari beberapa node yang melakukan *sensing* dan dapat mengontrol lingkungan, dapat berisi interaksi antara manusia dengan komputer dan keadaan lingkungan sekitar. WSN memiliki beberapa komponen yang digunakan untuk pemantauan lingkungan seperti *sensor node* dan *gateway* untuk mengumpulkan data sensor. Kedua komponen tersebut berkomunikasi dengan perantara teknologi *wireless*. Salah satu teknologi *wireless* adalah *LoRa*. *LoRa* adalah teknologi nirkabel berdaya rendah yang menggunakan spektrum radio. Karakteristik *LoRa* yang dapat mengakomodasi jarak yang jauh dan konsumsi daya yang rendah, menjadi keuntungan untuk diterapkan pada WSN. Tetapi terdapat kekurangan dalam implementasi *LoRa*, yaitu belum adanya sistem *gateway discovery* pada *LoRa*. Dari penjelasan tersebut, maka penelitian ini berfokus untuk membuat sistem *gateway discovery* untuk mengatasi kekurangan tersebut. Hasil pengujian performa yang menunjukkan bahwa sistem yang dibangun masih dapat berfungsi hingga jarak 1000 meter. Pada jarak maksimal pengujian performa yaitu 1000 meter nilai *association time* sebesar 354.4 milidetik, *unicast loss* sebesar 6% dan *broadcast loss* sebesar 20%. Hal ini berdampak pada kinerja sistem *gateway discovery* karena dengan besarnya nilai dari parameter tersebut, maka kinerja sistem akan semakin buruk.

Kata kunci: *Wireless Sensor Network, sensor node, gateway, LoRa, gateway discovery*

ABSTRACT

Wireless Sensor Network (WSN) is a network of several nodes that do sensing and can control the environment, can contain interactions between humans and computers and the state of the environment. WSN has several components that are used for environmental monitoring such as sensor nodes and gateways to collect sensor data. Both components communicate with wireless technology intermediaries. One of the wireless technologies is LoRa. LoRa is a low-power wireless technology that uses the radio spectrum. LoRa characteristics that can accommodate long distances and low power consumption, are suitable to be applied to the WSN. But there are shortcomings in the implementation of LoRa, namely the absence of a gateway discovery system in LoRa. From this explanation, this research focuses on creating gateway discovery system to overcome these shortcomings. The performance test results show that the system built can function up to a distance of 1000 meters. At the maximum distance of performance testing, namely 1000 meters, the association time value is 354.4 milliseconds, unicast loss is 6% and broadcast loss is 20%. This has an impact on the performance of gateway discovery system because increase value of these parameters, the system performance will get worse.

Keywords: *Wireless Sensor Network, sensor node, gateway, LoRa, gateway discovery*

DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR.....	iv
ABSTRAK.....	vi
ABSTRACT	vii
DAFTAR ISI	viii
DAFTAR TABEL.....	xi
DAFTAR GAMBAR.....	xii
BAB 1 PENDAHULUAN.....	1
1.1 Latar belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	3
1.4 Manfaat.....	3
1.5 Batasan masalah	3
1.6 Sistematika pembahasan	4
BAB 2 LANDASAN KEPUSTAKAAN	6
2.1 Kajian Pustaka	6
2.2 Dasar teori.....	7
2.2.1 <i>Wireless Sensor Network (WSN)</i>	7
2.2.2 <i>LoRa (Long Range)</i>	9
2.2.3 <i>Modul LoRa HopeRF-RFM9x</i>	11
2.2.4 <i>Arduino Nano</i>	13
2.2.5 <i>Arduino IDE</i>	14
2.2.6 <i>Parameter Pengujian</i>	14
BAB 3 METODOLOGI PENELITIAN	16
3.1 Studi literatur	18
3.2 Rekayasa Kebutuhan.....	19
3.2.1 <i>Analisis Kebutuhan Pengguna</i>	19

3.2.2 Analisis Kebutuhan Sistem	19
3.2.3 Analisis Kebutuhan Perangkat Keras.....	20
3.2.4 Analisis Kebutuhan Perangkat Lunak	21
3.3 Perancangan	22
3.4 Implementasi	25
3.5 Pengujian	27
3.6 Pengambilan kesimpulan dan saran	27
BAB 4 REKAYASA KEBUTUHAN.....	28
4.1 Pendahuluan	28
4.2 Tujuan	28
4.2.1 Ruang Lingkup	28
4.2.2 Istilah	28
4.2.3 Sistematika	30
4.3 Deskripsi Umum.....	30
4.3.1 Perspektif Produk / Sistem.....	30
4.3.2 Kegunaan.....	30
4.3.3 Karakteristik Pengguna	31
4.3.4 Lingkungan Operasi.....	31
4.3.5 Batasan Perancangan dan Implementasi.....	31
4.3.6 Asumsi dan Ketergantungan	31
4.4 Kebutuhan Antarmuka Eksternal.....	32
4.4.1 Antarmuka Pengguna.....	32
4.4.2 Antarmuka Perangkat Keras	32
4.4.3 Antarmuka Perangkat Lunak.....	32
4.5 Kebutuhan Fungsional	33
BAB 5 PERANCANGAN DAN IMPLEMENTASI	34
5.1 Perancangan	34
5.1.1 Gambaran umum sistem.....	34
5.1.2 Perancangan Alur Kerja Sistem	35
5.1.3 Perancangan <i>Sensor Node</i>	38

5.1.4 Perancangan <i>Gateway</i>	41
5.1.5 Perancangan Pengujian	43
5.2 Implementasi Sistem	47
5.2.1 Implementasi <i>Sensor Node</i>	47
5.2.2 Implementasi <i>Gateway</i>	49
BAB 6 HASIL DAN PEMBAHASAN	53
6.1 Hasil Implementasi	53
6.2 Pengujian Fungsional	53
6.2.1 Pengujian Kode PF-01	54
6.2.2 Pengujian Kode PF-02	55
6.2.3 Pengujian Kode PF-03	56
6.2.4 Pengujian Kode PF-04	57
6.2.5 Pengujian Kode PF-05	58
6.2.6 Pengujian Kode PF-06	60
6.2.7 Pengujian Kode PF-07	61
6.2.8 Pengujian Kode PF-08	62
6.2.9 Pengujian Kode PF-09	63
6.2.10 Pengujian Kode PF-10	64
6.3 Pengujian Performa	65
6.4 Pembahasan Pengujian.....	70
BAB 7 PENUTUP	72
7.1 Kesimpulan.....	72
7.2 Saran	73
DAFTAR PUSTAKA.....	74
LAMPIRAN A	76
LAMPIRAN B	78

DAFTAR TABEL

Tabel 2.1 Kajian Pustaka	6
Tabel 2.2 Pin pada Modul LoRa HopeRF-RFM9x.....	12
Tabel 3.1 Kebutuhan Perangkat Keras	20
Tabel 3.2 Kebutuhan Perangkat Lunak	21
Tabel 4.1 Tabel Istilah	28
Tabel 4.2 Kebutuhan Fungsional	33
Tabel 5.1 <i>Pinout</i> kabel <i>female-female</i> Arduino Nano dan Modul Lora	39
Tabel 5.2 <i>Pinout</i> dengan kabel <i>female-female</i> Arduino Nano dan sensor DHT11	39
Tabel 5.3 Perancangan skenario pengujian fungsional	44
Tabel 5.4 Tabel Kode program <i>include</i> file dari RadioHead dan DHT sensor	48
Tabel 5.5 Pseudocode untuk kode program <i>sensor node</i>	48
Tabel 5.6 Tabel Kode program <i>include</i> file dari RadioHead.....	50
Tabel 5.7 Pseudocode untuk kode program <i>gateway</i>	51
Tabel 6.1 Tabel pengujian kode PF-01	54
Tabel 6.2 Tabel pengujian kode PF-02	55
Tabel 6.3 Tabel pengujian kode PF-03	56
Tabel 6.4 Tabel pengujian kode PF-04	57
Tabel 6.5 Tabel pengujian kode PF-05	59
Tabel 6.6 Tabel pengujian kode PF-06	60
Tabel 6.7 Tabel pengujian kode PF-07	61
Tabel 6.8 Tabel pengujian kode PF-08	62
Tabel 6.9 Tabel pengujian kode PF-09	63
Tabel 6.10 Tabel pengujian kode PF-10	64
Tabel 6.11 Skenario Pengujian Performa.....	65
Tabel 6.12 Hasil pengujian performa	70

DAFTAR GAMBAR

Gambar 2.1 Arsitektur WSN.....	8
Gambar 2.2 <i>LoRa</i> , <i>Wifi/BLE</i> dan <i>Cellular</i>	9
Gambar 2.3 Struktur <i>physical frame</i> <i>LoRa</i>	11
Gambar 2.4 Modul <i>LoRa</i> HopeRF-RFM9x	12
Gambar 2.5 Spesifikasi Arduino Nano	13
Gambar 3.1 Metodologi Penelitian.....	16
Gambar 3.2 Rancangan Umum Sistem	22
Gambar 3.3 Alur Kerja Sistem Secara Umum	23
Gambar 5.1 Gambaran umum sistem.....	34
Gambar 5.2 Perancangan Alur Kerja Sistem	36
Gambar 5.3 Alur komunikasi sistem	37
Gambar 5.4 Alur kerja <i>sensor node</i>	40
Gambar 5.5 Alur kerja <i>gateway</i>	42
Gambar 5.6 Implementasi perangkat keras pada <i>sensor node</i>	47
Gambar 5.7 Implementasi perangkat keras pada <i>gateway</i>	50
Gambar 6.1 Tampilan hasil pengujian PF-01 pada Arduino IDE	55
Gambar 6.2 Tampilan hasil pengujian PF-02 pada Arduino IDE	56
Gambar 6.3 Tampilan hasil pengujian PF-03 pada Arduino IDE	57
Gambar 6.4 Tampilan hasil pengujian PF-04 pada Arduino IDE	58
Gambar 6.5 Tampilan hasil pengujian PF-05 pada Arduino IDE	59
Gambar 6.6 Tampilan hasil pengujian PF-06 pada Arduino IDE	60
Gambar 6.7 Tampilan hasil pengujian PF-07 pada Arduino IDE	61
Gambar 6.8 Tampilan hasil pengujian PF-08 pada Arduino IDE	62
Gambar 6.9 Tampilan hasil pengujian PF-09 pada Arduino IDE	64
Gambar 6.10 Tampilan hasil pengujian PF-10 pada Arduino IDE	65
Gambar 6.11 Konfigurasi <i>output file</i> pada <i>Putty</i>	66
Gambar 6.12 Diagram <i>association time</i> terhadap jarak	67
Gambar 6.13 Diagram <i>broadcast loss</i> terhadap jarak	68

Gambar 6.14 Diagram <i>unicast loss</i> terhadap jarak.....	69
---	----

BAB 1 PENDAHULUAN

1.1 Latar belakang

Teknologi informasi telah berkembang sedemikian pesat dan terus mendorong manusia untuk menciptakan teknologi dan konsep-konsep baru yang dapat mempermudah pekerjaan manusia. Dunia teknologi informasi baru-baru ini telah diramaikan dengan munculnya konsep *Wireless Sensor Network* atau biasa disingkat WSN. Menurut Yinbiao S. (2014), WSN secara umum dapat diartikan sebagai jaringan dari beberapa node yang melakukan *sensing* dan dapat mengontrol lingkungan, dapat berisi interaksi antara manusia dengan komputer dan keadaan lingkungan sekitar. Konsep dan teknologi WSN memudahkan manusia untuk melakukan proses *monitoring* terhadap lingkungan dan berbagai hal lainnya. Sebagai contohnya dengan adanya teknologi WSN kita dapat mengontrol temperatur, kelembapan, kondisi cahaya dan bahkan pergerakan suatu objek.

WSN ini terdiri dari minimal dua node yang saling terhubung dan bertukar data melalui jaringan nirkabel. Node tersebut dapat terdiri dari perangkat untuk melakukan *sensing* maupun untuk meneruskan data yang diterima bergantung pada arsitektur WSN yang digunakan dan tujuan pembentukan jaringan WSN. Setiap node dalam WSN biasanya dilengkapi dengan sensor, mikrokontroler kecil (menyediakan konversi analog ke digital dan kemampuan komputasi dan penyimpanan), perangkat radio transceiver (menyediakan kemampuan komunikasi nirkabel) dan sumber energi atau penyimpanan energi (biasanya dalam bentuk baterai elektrokimia) (Flammini A. dan Sisinni E., 2014). Contohnya pada jaringan WSN untuk memantau suhu dan kelembapan pada suatu wilayah. Menurut Yinbiao S. (2014), umumnya WSN terdiri dari beberapa jaringan *sensor node* dan sebuah *gateway*. Komunikasi antara *sensor node* dan *gateway* tersebut dijumpai oleh perangkat atau teknologi yang mendukung jaringan nirkabel sehingga disebut sebagai WSN. Perangkat atau teknologi yang mendukung jaringan nirkabel pada WSN ada berbagai macam, salah satunya adalah *LoRa*.

LoRa adalah teknologi nirkabel berdaya rendah yang menggunakan spektrum radio (Wixted, A. J. et al., 2016). *LoRa* mempunyai beberapa kelebihan salah satunya bisa mengakomodasi jarak mencapai 1 km lebih dengan pengaturan yang tepat dan juga lingkungan yang mendukung seperti di pedesaan yang tidak padat penduduk dan sedikitnya gangguan serta jenis perangkat yang mendukung implementasi perangkat *LoRa*. Dengan kelebihan ini, *LoRa* sangat cocok untuk monitoring daerah pertanian di Indonesia yang dikenal sebagai negara agraris karena luasnya lahan pertanian yang dimiliki. Selain hal itu, sinyal yang dihasilkan *LoRa* tahan terhadap interferensi dan sulit disadap. Hal ini memberi keuntungan pada pemrosesan yang memungkinkan ketahanan terhadap gangguan dan *noise* (Reynders, B., Meert, W. and Pollin, S., 2016). Teknologi *LoRa* ini menjadi keuntungan tersendiri pada perkembangan

teknologi seperti *WSN* yang membutuhkan pengiriman data yang tahan terhadap *noise*, konsumsi daya kecil dan dapat mengakomodasi jarak *sensor node* dan *gateway* yang jauh.

Meskipun *LoRa* mempunyai banyak kelebihan, terdapat beberapa potensi kelemahan yang perlu untuk diselesaikan. Salah satu kelemahannya adalah belum adanya sistem *gateway discovery* secara otomatis pada teknologi *LoRa* untuk pengiriman data dari *sensor node*. Pengiriman data dari *sensor node* dilakukan kepada semua *gateway LoRa* yang berada pada area jangkauan. Hal ini dapat menjadi penyebab munculnya masalah jika terdapat lebih dari satu *gateway* untuk menerima datanya dimana terdapat *gateway* yang seharusnya tidak menerima data dari *sensor node* atau disebut sebagai *gateway* yang salah dan belum adanya sistem *gateway discovery* untuk menemukan *gateway* yang benar. Maka dari itu untuk mempermudah pemakaian perangkat *LoRa* dibutuhkan pengaturan secara otomatis untuk *gateway discovery* ketika perangkat *LoRa* digunakan terutama oleh masyarakat awam yang kurang mengerti teknologi maupun yang malas untuk melakukan banyak pengaturan. Hal itulah yang menjadi latar belakang dilakukannya penelitian ini agar masalah tersebut dapat teratasi. Pada penelitian ini, perangkat *LoRa* akan terhubung dengan mikrokontroler yaitu Arduino Nano pada sisi *sensor node* maupun pada sisi *gateway* karena harganya yang relatif terjangkau dan kemudahan untuk memperolehnya. Karena itu dengan penggunaan Arduino Nano ini, pengaturan untuk sistem *gateway discovery* pada perangkat *LoRa* tanpa melakukan konfigurasi teknis kembali ditanamkan pada mikrokontroler yang sudah terhubung dengan modul *LoRa* menggunakan Arduino IDE pada *sensor node* dan pada sisi *gateway*. Sistem *gateway discovery* sebagian besar ditanamkan pada sisi *sensor node*, sedangkan pada sisi *gateway* hanya mengambil sedikit peran untuk menerapkan sistem ini.

1.2 Rumusan Masalah

Berdasarkan dari uraian latar belakang di atas, maka rumusan masalah yang akan dibahas dalam penelitian ini yaitu :

1. Bagaimana cara untuk menerapkan sistem *gateway discovery* pada *wireless sensor network* berbasis pada perangkat *LoRa* ?
2. Bagaimana cara *sensor node* membedakan *gateway* miliknya dan *gateway* yang lain?
3. Bagaimana cara pengolahan data yang telah dipertukarkan antara *sensor node* dan *gateway* dalam sistem *gateway discovery* ?
4. Bagaimana performa dari sistem untuk sistem *gateway discovery* yang telah diuji terhadap jarak antara *sensor node* dan *gateway*?

1.3 Tujuan

Dari rumusan masalah di atas, maka dapat ditentukan tujuan dari penelitian pada skripsi ini yaitu :

1. Agar sistem *gateway discovery* dapat diterapkan pada WSN yang berbasis perangkat *LoRa* agar tidak mengirim ke *gateway* yang salah, sehingga pengguna cukup menjalankan modul *LoRa* dengan pengaturan seminimal mungkin atau tanpa pengaturan dalam proses penentuan *gateway*.
2. Agar perangkat *LoRa* tetap dapat mengirim data ke *gateway* yang benar meskipun terdapat lebih dari satu *gateway* yang tersedia serta membuat *gateway* hanya dapat menerima data dari *sensor node* yang benar.
3. Mengetahui cara pengolahan data yang dipertukarkan antara *sensor node* dan *gateway* dalam sistem *gateway discovery* yang dibuat.
4. Mengetahui performa dari penerapan sistem *gateway discovery* yang telah dibuat

1.4 Manfaat

Manfaat yang diharapkan dari skripsi ini adalah sebagai berikut:

1. Penelitian ini diharapkan agar menjadi solusi untuk pengguna *LoRa* agar dapat menggunakan secara langsung perangkat *LoRa* tanpa melakukan pengaturan apapun atau dengan pengaturan seminimal mungkin untuk transmisi data karena sudah ada fitur untuk *gateway discovery* secara otomatis.
2. Selain itu skripsi ini juga dapat bermanfaat untuk pengembangan *LoRa* di masa depan maupun segala penelitian yang memiliki keterkaitan dengan penelitian ini.
3. Sedangkan bagi penulis, skripsi ini dapat menjadi wadah untuk menerapkan ilmu yang didapatkan dan menambah wawasan tentang berbagai hal yang digunakan dalam skripsi ini maupun kondisi penggunaan perangkat di dunia nyata.

1.5 Batasan masalah

Dalam penyusunan skripsi yang baik diberikan beberapa batasan masalah yang akan menjadi batasan dalam melakukan penelitian dan juga dapat memudahkan penyusunan laporan yang sistematis sehingga mudah dipahami. Batasan-batasan yang digunakan dalam skripsi ini antara lain :

1. Perangkat modul *LoRa* yang digunakan bertipe HopeRF-RFM9x.
2. Frekuensi yang digunakan oleh modul *LoRa* adalah 433 MHz.

3. Mikrokontroller yang digunakan pada sisi *sensor node* adalah Arduino Nano.
4. Mikrokontroller yang digunakan pada sisi *gateway* adalah Arduino Nano.
5. Faktor keamanan pada implementasi sistem, tidak menjadi fokus skripsi ini.
6. Data dari *sensor node* tidak akan disimpan ke dalam database, namun hanya ditampilkan.
7. Data yang dikirim dari *sensor node* merupakan data yang diperoleh dari sensor DHT 11.
8. Pengujian performa dilakukan di desa Kidal, kecamatan Tumpang, kabupaten Malang dengan kondisi terdapat lahan hijau persawahan yang luas dengan minimnya bangunan dan bukan pada wilayah padat penduduk.
9. Pengujian menggunakan waktu relatif dari perangkat Arduino yang berjalan ketika arduino dijalankan.

1.6 Sistematika pembahasan

Agar skripsi ini lebih mudah untuk dipahami oleh pembaca, maka penulis memberikan penjelasan singkat dari sistematika pembuatan skripsi dengan urutan sebagai berikut:

BAB 1 : PENDAHULUAN

Pendahuluan terdiri dari latar belakang, rumusan masalah, tujuan, manfaat dan batasan masalah serta sistematika pembahasan dari penelitian ini. Selain itu, sebuah sistematika pembahasan digunakan sebagai acuan untuk melakukan pembahasan terhadap penelitian yang dilakukan secara sistematis.

BAB 2 : LANDASAN KEPUSTAKAAN

Bab ini berisi kajian pustaka berdasarkan beberapa referensi dan teori-teori yang terkait dengan penelitian ini. Selain itu juga dapat berisi penelitian-penelitian sebelumnya yang menunjang penelitian ini. Hal ini diperlukan guna mempermudah melakukan penelitian dan juga sebagai acuan atau pedoman penelitian yang resmi karena telah diterbitkan.

BAB 3 : METODOLOGI PENELITIAN

Bab ini akan membahas tentang sistematika dalam melakukan penelitian. Langkah-langkah seperti perancangan, implementasi, pengujian dibahas secara umum.

BAB 4 : REKAYASA KEBUTUHAN

Bab ini membahas tentang kebutuhan apa saja yang dibutuhkan oleh sistem. Kebutuhan fungsional dari sistem akan diberikan pada bab ini.

BAB 5 : PERANCANGAN DAN IMPLEMENTASI

Bab ini membahas tentang perancangan dan implementasi dari sistem. Pada bab ini dijabarkan perancangan umum maupun uraian lebih lanjut mengenai perancangan sistem secara mendetail dari sistem yang dibuat. Perancangan sistem akan menjabarkan komponen dan cara kerja secara umum sistem yang akan dibuat. Selain itu juga akan dijabarkan tentang perancangan pengujian dari sistem yang telah dibuat. Sedangkan implementasi akan sesuai dengan perancangan yang telah dibuat sebelumnya secara rinci beserta langkah-langkah pengerjaan penelitian. Selain itu juga memberikan *pseudocode* utama yang digunakan dalam penelitian ini.

BAB 6 : HASIL DAN PEMBAHASAN

Bab ini menjelaskan tentang bagaimana saat sistem telah diimplementasikan dan dijalankan pada lingkungan uji yang telah ditentukan terlebih dahulu. Hasil-hasil yang diperoleh akan dijabarkan pada bab ini. Selanjutnya akan dilakukan pengujian dan hasil pengujian didapatkan dari pengujian yang telah dilakukan sesuai dengan skenario pengujian yang telah ditentukan. Hasil dari pengujian akan dianalisis, dibahas dan dijelaskan pada bagian ini.

BAB 7 : PENUTUP

Bab terakhir ini dipaparkan hasil kesimpulan dari pelaksanaan penelitian. Kesimpulan ini dibuat dengan hasil pengujian dan analisis terhadap perancangan dan implementasi arsitektur sistem sebagai dasarnya. Saran-saran untuk penelitian juga diberikan agar hasil dari penelitian ini dapat diperbaiki dan disempurnakan kemudian apabila akan dikembangkan diwaktu yang akan datang.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisi tinjauan pustaka yang meliputi kajian pustaka dan dasar teori yang diperlukan untuk mendukung penelitian ini serta menjadi dasar penelitian. Kajian Pustaka membahas penelitian sebelumnya yang telah dilakukan dan berhubungan dengan penelitian ini. Dasar Teori membahas tentang teori yang berhubungan dan diperlukan untuk menyusun penelitian.

2.1 Kajian Pustaka

Terdapat beberapa penelitian yang telah dilakukan sebelumnya yang mempunyai kemiripan maupun membahas tentang hal yang berhubungan dengan penelitian sekarang ini. Tabel 2.1 di bawah ini menunjukkan kajian pustaka yang dilakukan.

Tabel 2.1 Kajian Pustaka

No	Judul	Tahun	Penulis	Penelitian	Penelitian Penulis
1	LoRaWAN - A Low Power WAN Protocol for Internet of Things: a Review and Opportunities	2017	Jonathan de Carvalho Silva, Joel J. P. C. Rodrigues, Antonio M. Alberti, Petar Solic, Andre L. L. Aquino	Menganalisis protokol LoRa dalam penggunaannya untuk <i>IoT</i> dan membandingkan nya dengan teknologi LPWAN lain	Mengimplementasi protokol <i>LoRa</i> untuk digunakan pada <i>WSN</i> dan menerapkan sistem <i>gateway discovery</i> untuk menemukan <i>gateway</i> yang benar secara otomatis
2	Evaluation of LoRa and LoRaWAN for Wireless Sensor Networks	2016	Andrew J Wixted; Peter Kinnaid; Hadi Larijani; Alan Tait; Ali Ahmadinia; Niall Strachan	Mengevaluasi kerja dan performansi dari penggunaan <i>LoRa</i> pada <i>Wireless Sensor Network</i>	Menggunakan sistem <i>gateway discovery</i> pada <i>LoRa</i> untuk pengiriman data dari <i>sensor node</i> dan menguji performa dari <i>LoRa</i> menggunakan modul <i>LoRa</i> yang telah ditentukan sebelumnya

3	Internet of Things and LoRa™ Low-Power Wide-Area Networks Challenges	2017	Alexandru Lavric, Valentin Popa	Mengevaluasi <i>LoRa</i> untuk bekerja pada bidang <i>IoT</i> dan mengatasi beberapa tantangan pada <i>IoT</i> dan menyarankan untuk menggunakan banyak <i>gateway</i>	Membuat perangkat <i>LoRa</i> menemukan <i>gateway</i> nya secara otomatis jika terdapat lebih dari satu <i>gateway</i>
4	Design and Implementati on of Smart Irrigation System Based on LoRa	2017	Wenju Zhao; Shengwei Lin; Jiwen Han; Rongtao Xu; Lu Hou	Menggunakan <i>LoRa</i> untuk <i>IoT</i> dibidang irigasi dan memanfaatkan keunggulan <i>LoRa</i> dalam hal jarak	Menerapkan penggunaan <i>LoRa</i> pada pengiriman data dari <i>sensor node</i> menggunakan mikrokontroller yang terhubung dengan modul <i>LoRa</i> dengan data yang diambil dari sensor suhu dan kelembapan

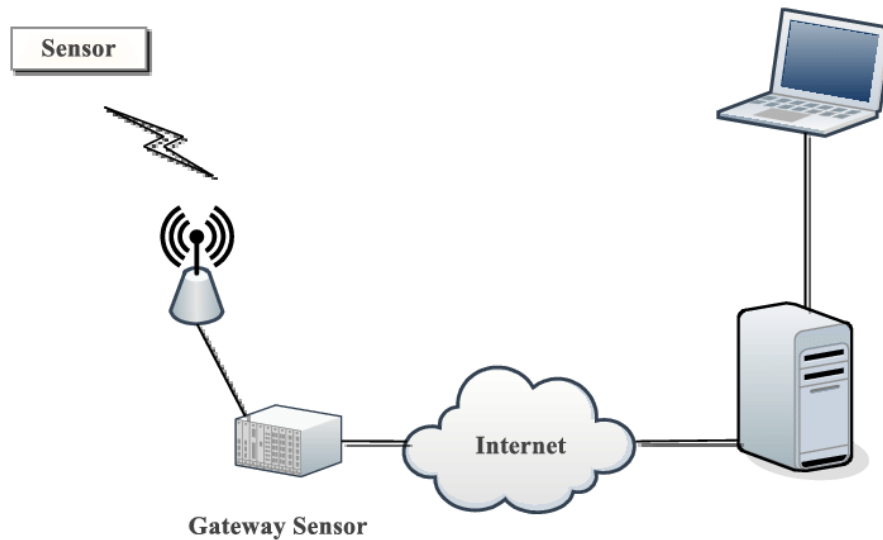
2.2 Dasar teori

Untuk melakukan penelitian ini, diperlukan beberapa referensi dan dasar teori yang mendukung penelitian. Hal ini guna memperlancar pada proses selanjutnya seperti perancangan dan implementasi. Berikut adalah beberapa teori yang menjadi dasar penelitian:

2.2.1 *Wireless Sensor Network (WSN)*

Wireless Sensor Network (WSN) merupakan jaringan nirkabel yang terdiri dari perangkat otonom yang didistribusikan secara spasial menggunakan sensor untuk memantau lingkungan sekitar (National Instruments, 2016). Data yang dibutuhkan untuk pemantauan lingkungan sekitar pada WSN contohnya adalah data suhu dan kelembapan. Pada implementasi WSN bisa terdapat beberapa *sensor node* yang telah dibekali dengan sensor yang diperlukan sistem dan akan mengirimkan data kepada suatu node yang dianggap sebagai *gateway*. Sesuai dengan namanya, pengiriman

data pada WSN bergantung pada komunikasi nirkabel yang disematkan pada perangkat yang akan berkomunikasi. Konsep sederhana yang menjelaskan tentang arsitektur WSN dapat dilihat pada gambar 2.1.



Gambar 2.1 Arsitektur WSN

Sumber: wazir et. al., 2016

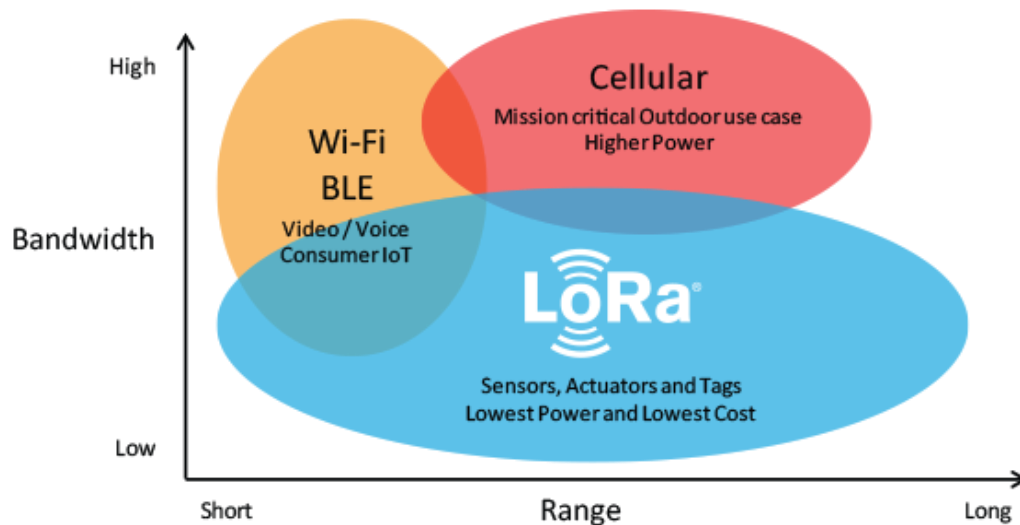
Gambar 2.1 menunjukkan arsitektur WSN secara sederhana dengan beberapa komponen yang terdapat di dalamnya. Dalam penelitian ini, cakupan penelitian hanya berfokus pada bagian komunikasi antara sensor dan *gateway* sensor yang komunikasinya akan dijumpai oleh modul *LoRa*. WSN memiliki beberapa topologi yang umumnya digunakan saat implementasi dilapangan, seperti model flat yang terdiri dari dua bagian utama yaitu *sensor node* dan *gateway*. WSN memiliki berbagai macam arsitektur tergantung keperluan dan tujuan digunakannya, seperti penjelasan sebelumnya penelitian ini hanya berfokus pada komunikasi *sensor node* dan *gateway* yang bertugas mengumpulkan data dari *sensor node*.

Pada dasarnya sebuah *sensor node* terdiri dari komponen pengendali (*controller*), sensor, perangkat komunikasi dan catu daya (*power supply*) (Sugiarto, B., 2010). *Sensor node* ini nantinya akan mengirimkan data yang diperoleh dari proses *sensing* kepada *gateway* yang benar. Bagian *gateway discovery* atau menemukan *gateway* secara otomatis pada jaringan WSN ini akan menjadi fokus penelitian ini. Implementasi pada jaringan WSN menggunakan teknologi *LoRa* akan menjadi fokus penelitian ini karena kekurangan yang dimiliki oleh teknologi ini.

2.2.2 LoRa (Long Range)

LoRa adalah teknologi nirkabel yang dikembangkan oleh *Semtech* untuk memungkinkan komunikasi dengan *data rate* rendah yang dibuat untuk komunikasi dengan jarak yang jauh (Poole, 2017). Sedangkan menurut Lavric dan Popa (2017), LoRa (Long Range) adalah teknik modulasi yang memungkinkan pengiriman informasi jarak jauh dengan *transfer rate* yang rendah. Nilai *transfer rate* pada *LoRa* berada pada kisaran 0.3-50 kbps. Meskipun dengan keterbatasan nilai *transfer rate* yang rendah, *LoRa* masih bisa digunakan untuk pengiriman data dari *sensor node* pada WSN yang berisi data sensor seperti data sensor suhu dan kelembapan.

LoRa menggunakan modulasi *Chirp-Spread-Spectrum (CSS)* dengan beberapa pilihan untuk *Spreading Factor* yang berbeda untuk mengoptimalkan modulasi demi keperluan jarak dan data (Wixted, A. J. et al., 2016). Teknik modulasi CSS memungkinkan panjang jangkauan komunikasi dengan data kecil, ketebalan tinggi terhadap interferensi serta konsumsi daya yang rendah, sehingga memiliki keunggulan untuk komunikasi jarak jauh dengan konsumsi daya yang rendah. Teknik modulasi *LoRa* yang bekerja pada *physical layer* ini menggunakan frekuensi ISM 433 MHz, 868 MHz atau 915 MHz pada *LoRa* bergantung pada letak negara dan kebijakan suatu wilayah atau negara. Frekuensi dari *LoRa* yang digunakan dalam penelitian ini adalah 433 MHz. Selain itu, *LoRa* juga dapat mengatasi keseimbangan kelemahan yang ada pada *cellular* dan *Wi-fi/BLE* yang memerlukan bandwidth tinggi/daya tinggi atau memiliki jangkauan terbatas atau ketidakmampuan untuk menembus ruangan. Skema tentang hal tersebut dapat dilihat pada gambar 2.2.



Gambar 2.2 LoRa, Wifi/BLE dan Cellular

Sumber: www.semtech.com, diakses 27-11-2018

Gambar 2.2 menunjukkan skema perbandingan *LoRa*, *Wifi/BLE* dan *Cellular* dengan ukuran menurut jarak dan *bandwidth*. Dalam gambar 2.2 terlihat bahwa *LoRa* dapat menjangkau jarak yang jauh seperti *cellular* namun dapat mengirim data dengan *bandwidth* yang rendah. Berikut adalah beberapa kelebihan yang disediakan oleh *LoRa*:

Fitur:

- Jarak yang jauh bisa mencapai 2 km atau lebih antar node tergantung jenis perangkat dan tambahan seperti antena untuk penguat sinyal, lingkungan perangkat saat dijalankan seperti daerah pedesaan yang tidak padat penduduk dan gangguan yang sedikit maupun dengan pengaturan perangkat yang digunakan. Selain itu jarak jangkauan komunikasi nirkabel ini juga dipengaruhi oleh jenis modul *LoRa* yang digunakan.
- Jutaan Node yang dapat terhubung seperti pada arsitektur *LoRaWAN*
- Masa pakai baterai yang lama, dapat mencapai lebih dari 10 tahun karena konsumsi daya yang kecil sekitar 13 Ma sampai 15 Ma.

LoRa bekerja pada *physical layer*, dimana *LoRa* menggunakan teknik modulasi yang merupakan teknologi hak milik Semtech yang tidak sepenuhnya terbuka. *LoRa* menggunakan modulasi radio *Chirp Spread Spectrum* (CSS) yang menggunakan *chirp* dengan variasi frekuensi linear dari waktu ke waktu untuk menyandikan informasi (Augustin, A. et al., 2016). Hal ini membuat modulasi ini kebal terhadap efek Doppler. Teknik modulasi yang digunakan *LoRa* ini dikembangkan oleh Semtech dan tidak sepenuhnya terbuka untuk dipublikasikan. Terdapat beberapa parameter dalam modulasi *LoRa*, yaitu:

1. *Bandwidth* (BW)

Bandwidth merupakan parameter yang sangat penting pada penentuan *chip rate*. *Chip* sendiri merupakan satuan elemen dalam konteks *chirp spread spectrum* yang berbentuk pulsa (*pulse*) sehingga *chip rate* adalah laju perubahan *chip* persatuan waktu. Pada modulasi *LoRa* besarnya *bandwidth* diatur sebesar 125kHz, 250 kHz, dan 500 kHz.

2. *Spreading Factor* (SF)

Spreading Factor merupakan durasi dari *chirp*. *Chirp* adalah “*Compressed High Intensity Radar Pulse*” merupakan meningkatnya sinyal frekuensi (up-chirp) atau turun (down-chirp) terhadap waktu. *LoRa* beroperasi dengan *spreading factor* dari 7 hingga 12. *Spreading factor* 7 adalah waktu tersingkat di udara, sedangkan *spreading factor* 12 adalah waktu yang terpanjang.

3. Coding Rate (CR)

Coding rate diformulasikan sebagai $CR = 4/(4+n)$ untuk menangani *Packet Error Rate* (PER) akibat adanya interferensi dimana n merupakan $\{1,2,3,4\}$. Menurunkan *code rate* akan membantu menurunkan *packet error rate* dari interferensi. Misalnya paket yang dikirimkan dengan *code rate* 4/8 akan lebih toleran terhadap interferensi daripada *code rate* 4/5.

Semtech juga telah menentukan format *physical frame* pada LoRa. Struktur format *physical frame* pada LoRa ditunjukkan pada Gambar 2.3.



Gambar 2.3 Struktur *physical frame* LoRa

Sumber: (Augustin, A. et al., 2016)

Pada Gambar 2.3, *physical frame* LoRa dimulai dengan *preamble* dimana *preamble* digunakan untuk membedakan jaringan LoRa yang menggunakan frekuensi yang sama. Setelah *preamble*, terdapat *header* dimana *header* bersifat opsional. *Header* digunakan untuk menunjukkan ukuran *payload* (dalam byte), *code rate* untuk akhir transmisi dan apakah CRC 16-bit untuk *payload* ada di bagian akhir *frame*. *Header* ditransmisikan dengan *code rate* 4/8. Kemudian *payload* dikirimkan setelah *header* dan pada akhir terdapat *frame CRC* yang bersifat opsional. Ukuran *payload* disimpan dengan ukuran 1 byte dan dibatasi hingga 255 byte.

2.2.3 Modul LoRa HopeRF-RFM9x

Implementasi teknologi *LoRa* dapat diaplikasikan melalui modul *LoRa* yang banyak dijual dipasaran. Modul *LoRa* ini mempunyai berbagai tipe yang dapat dibeli tergantung kebutuhan dan dana yang dimiliki. Beberapa contohnya dapat dilihat pada situs resmi Semtech seperti SX1272, SX1273 dan lainnya (Semtech, 2018). Selain itu terdapat beberapa produk yang beredar di pasar Indonesia seperti modul *LoRa* tipe HopeRF-RFM9x dan HopeRF-RFM69.

Skripsi ini menggunakan perangkat modul *LoRa* bertipe HopeRF-RFM9x untuk mengimplementasikan teknologi *LoRa*. Perangkat ini merupakan *transceiver* yang dapat dihubungkan dengan mikrokontroller ataupun mikrokomputer untuk komunikasi nirkabel. Pada modul *LoRa* bertipe HopeRF-RFM9x, terdapat beberapa frekuensi yang dapat digunakan seperti frekuensi 915 MHz, 868 MHz dan 433 MHz. Frekuensi yang dipakai oleh modul *LoRa* pada sistem ini adalah 433 MHz. Berikut adalah perangkat modul *LoRa* bertipe HopeRF-RFM9x yang ditunjukkan pada gambar 2.4.



Gambar 2.4 Modul *LoRa* HopeRF-RFM9x

Sumber: www.tindie.com, diakses 24-05-2018

Modul LoRa tersebut mempunyai jarak jangkauan yang bervariasi tergantung dari banyak faktor seperti yang telah disebutkan pada subbab sebelumnya. Untuk menggunakan modul LoRa dengan tipe seperti gambar 2.3 harus menghubungkannya dengan mikrokontroler maupun mikrokomputer melalui pin-pin yang tersedia pada papan modul LoRa tersebut. Modul LoRa HopeRF-RFM9x ini dibekali dengan antena *helical* pada saat pertama dibeli. Modul LoRa HopeRF-RFM9x merupakan *breakout board* dari HopeRF RFM95/96/97/98W yang merupakan *transceiver* pada seri 9x. Modul ini dapat digunakan dalam berbagai hal seperti pada *remote* drone dan implementasi *Internet of Things*, maupun pada *Wireless Sensor Network*.

Pada modul ini terdapat beberapa pin yang mempunyai fungsi yang berbeda satu sama lain. Pin-pin ini nantinya akan digunakan untuk menghubungkan modul *LoRa* ini dengan perangkat yang lain yang dijematani kabel *jumper*. Untuk penjelasan mengenai pin yang digunakan dalam penelitian ini dapat dilihat pada tabel 2.2.

Tabel 2.2 Pin pada Modul *LoRa* HopeRF-RFM9x

Nama Komponen	Fungsi
Power	Pin power ini berguna untuk menerima aliran listrik sebagai daya utama untuk perangkat ini. Pada modul <i>LoRa</i> ini terdapat pin dengan tanda 3.3V
Ground (GND)	Pin Ground (GND) ini digunakan untuk menerima <i>ground</i> dari perangkat yang tersambung. Ground digunakan untuk melindungi perangkat dari tegangan listrik berlebihan dan untuk logika dalam perangkat.
SCK	Pin ini adalah pin <i>SPI clock</i> yang mengeluarkan <i>clock</i> sinkronisasi untuk menghindari kesalahan komunikasi
MISO	Pin ini adalah pin Master In Slave Out, untuk data yang dikirim dari radio ke prosesor
MOSI	Pin ini adalah pin Master Out Slave In, untuk data yang dikirim dari prosesor ke radio

NSS	Pin ini akan diatur ke posisi <i>low</i> untuk memulai kerja SPI
RST	ini adalah pin Reset untuk radio. Pin ini diatur pada mode high secara <i>default</i> .
DIO0	Pin ini digunakan untuk IRQ yang memberikan pemberitahuan permintaan interupsi dari radio ke mikrokontroller

2.2.4 Arduino Nano

Arduino Nano adalah salah satu mikrokontroller yang berukuran kecil, lengkap dan mendukung penggunaan *breadboard*. Arduino Nano diciptakan dengan basis mikrokontroller ATmega328 (untuk Arduino Nano versi 3.x) atau ATmega 168 (untuk Arduino versi 2.x). Arduino Nano kurang memiliki fungsi yang hampir sama dengan Arduino lainnya, tetapi dalam paket yang berbeda. Arduino Nano tidak menyertakan colokan DC berjenis *Barrel Jack*, dan dihubungkan ke komputer menggunakan port USB Mini-B. Arduino Nano dirancang dan diproduksi oleh perusahaan Gravitech. Arduino Nano dalam pemrogramannya menggunakan Arduino IDE (Arduino, 2018).

Penelitian ini dilaksanakan menggunakan papan Arduino Nano versi 3.x. Berikut adalah spesifikasi dari Arduino Nano yang ditunjukkan pada gambar 2.5:

Name	Processor	Operating/Input Voltage	CPU Speed	Analog In/Out	Digital IO/PWM	EEPROM [kB]	SRAM [kB]	Flash [kB]	USB
Nano	ATmega168	5 V / 7-9 V	16 MHz	8/0	14/6	0.512	1	16	Mini
	ATmega328P					1	2	32	

Gambar 2.5 Spesifikasi Arduino Nano

Sumber: www.arduino.cc, diakses 24-05-2018

Pada mikrokontroller ini akan diterapkan *gateway discovery* pada sisi *sensor node* agar *sensor node* dapat menemukan *gateway* secara otomatis para perangkat *LoRa*. Pengguna hanya dengan melakukan *upload* kode program pada mikrokontroller dengan pengaturan seminim mungkin, dapat mengirimkan data dari sensor ke *gateway* yang benar. Pengaturan seminim mungkin atau bahkan tidak adanya pengaturan tambahan yang dilakukan bisa disebut *zero configuration*. *Zero Configuration* sendiri juga dapat diartikan sebagai cara untuk memasang perangkat komputer atau menghubungkan komputer atau perangkat ke jaringan tanpa memerlukan konfigurasi teknis lagi oleh pengguna (LovetoKnow, 2017). Tetapi untuk membuat sistem *gateway discovery* ini dapat berjalan dengan baik, diperlukan pengaturan tambahan di sisi *gateway*. Mikrokontroller Arduino Nano ini nantinya akan digunakan pada sisi *sensor node* dan juga sisi *gateway* serta dikaitkan dengan

modul LoRa. Kombinasi urutan pin yang digunakan untuk menghubungkan kedua perangkat akan dijabarkan pada bab perancangan dan implementasi.

2.2.5 Arduino IDE

IDE merupakan singkatan dari *Integrated Development Environment*, atau bisa dikatakan sebagai lingkungan terintegrasi yang digunakan untuk melakukan pengembangan. Disebut sebagai lingkungan karena melalui *software* ini dapat dilakukan pemrograman untuk melakukan fungsi-fungsi yang ditanamkan melalui sintaks pemrograman pada suatu perangkat. Arduino menggunakan bahasa pemrograman sendiri yang menyerupai bahasa C. Bahasa pemrograman Arduino sudah dilakukan perubahan untuk memudahkan pemula dalam melakukan pemrograman dari bahasa aslinya. Sebelum dijual ke pasaran, IC mikrokontroler Arduino telah ditanamkan suatu program bernama *bootlader* yang berfungsi sebagai perantara antara *compiler* Arduino dengan mikrokontroler (sinuarduino,2016).

Arduino IDE adalah *software open-source* yang dibuat dari bahasa pemrograman JAVA. Arduino IDE dibuat untuk memudahkan penulisan kode dan mengunggah perintah ke board Arduino. Arduino IDE dapat berjalan pada Windows, Mac OS X, dan Linux (Arduino, 2017). Arduino IDE juga dilengkapi dengan library C/C++ yang biasa disebut *wiring* yang membuat operasi *input* dan *output* menjadi lebih mudah. Arduino IDE ini dikembangkan dari *software processing* yang dirombak menjadi Arduino IDE khusus untuk pemrograman dengan Arduino. Pemrograman menggunakan Arduino IDE pada penelitian ini menggunakan *library* RadioHead untuk mempermudah pengaturan untuk perangkat *LoRa*. *Library* ini dapat ditambahkan seperti layaknya *library* lainnya atau bisa diunduh pada github RadioHead. Selain itu juga digunakan sensor DHT *library* untuk mempermudah pengaturan sensor dan pengambilan datanya yang disini digunakan sensor DHT11.

2.2.6 Parameter Pengujian

Parameter yang diuji pada penerapan sistem *gateway discovery* menggunakan perangkat modul *LoRa* adalah *association time*, *broadcast loss* dan *unicast loss*.

- *Association time* merupakan jumlah waktu yang dibutuhkan sistem untuk melakukan sistem *gateway discovery*. *Association time* akan dihitung mulai dari saat *gateway* melakukan *broadcast* dan berakhir ketika data dari *sensor node* diterima oleh *gateway*. Berikut merupakan persamaan dari *association time*:

$$\text{Association time} = u - b$$

Keterangan:

u = waktu *unicast* diterima oleh *gateway*

b = waktu *broadcast* dilakukan oleh *gateway*

- *Broadcast loss* merupakan perhitungan dalam bentuk persen (%) yang dihitung berdasarkan proses *broadcast* yang dilakukan oleh *gateway*. Ketika *gateway* melakukan *broadcast* untuk menandakan *gateway* yang benar kepada setiap *sensor node* yang berada pada area jangkauannya, pastinya terdapat hambatan seperti gangguan sinyal ataupun jauhnya jarak antara *gateway* dan *sensor node* yang dapat menyebabkan *broadcast* tidak dapat diterima oleh *sensor*. Nantinya angka 100% akan dikurangi *broadcast* yang diterima oleh *sensor node* akan dibagi *broadcast* yang dilakukan dan dikalikan 100%. Hal inilah yang disebut sebagai *broadcast loss*. Berikut persamaan dari *broadcast loss*:

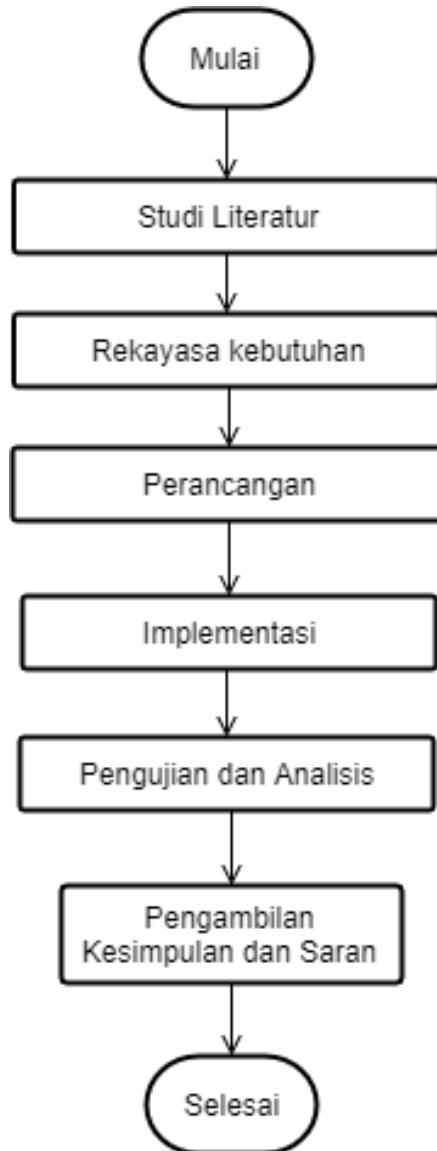
$$\text{broadcast loss} = 100\% - \left(\frac{\text{broadcast yang diterima sensor node}}{\text{broadcast yang dilakukan gateway}} \times 100\% \right)$$

- *Unicast loss* merupakan perhitungan dengan hasil dalam bentuk persen (%) yang dihitung berdasarkan proses *unicast* yang dilakukan oleh *sensor node*. Setelah *broadcast* diterima oleh *sensor node*, *sensor node* melakukan *unicast* kepada alamat *gateway* yang benar berdasarkan hasil *broadcast* tadi. Ketika melakukan proses *unicast* tersebut, paket yang dikirimkan tidak dijamin dengan pasti untuk diterima oleh tujuannya. Hal ini dapat terjadi karena beberapa gangguan yang kemungkinan terdapat di tempat pengujian. Berikut adalah persamaan dari *unicast loss*:

$$\text{unicast loss} = 100\% - \left(\frac{\text{unicast yang diterima gateway}}{\text{unicast yang dikirim sensor node}} \times 100\% \right)$$

BAB 3 METODOLOGI PENELITIAN

Bagian ini membahas tentang metode yang digunakan untuk implementasi sistem *gateway discovery* pada WSN berbasis modul komunikasi Lora. Membahas juga langkah-langkah yang dilalui untuk melakukan penelitian. Untuk mempermudah dalam pemahaman pengerjaan penelitian, berikut disajikan diagram alir penelitian yang dapat dilihat pada gambar 3.1. Berikut adalah diagram alirnya:



Gambar 3.1 Metodologi Penelitian

Gambar 3.1 menunjukkan alur penelitian yang akan dilakukan peneliti dimulai dari studi literatur untuk memperdalam pemahaman tentang penelitian terdahulu yang telah dilakukan. Studi literatur merupakan bagian dimana dilakukan pencarian referensi tentang penelitian terdahulu dan dasar teori yang berhubungan dengan penelitian yang akan dilakukan. Studi literatur dapat diambil dari buku, jurnal ilmiah atau referensi di internet yang mempunyai keterkaitan dengan penelitian. Hal ini diperlukan untuk nantinya sebagai bahan referensi utama untuk membantu penelitian. Selanjutnya adalah membuat rekayasa kebutuhan untuk penelitian yang diawali dengan menganalisis kebutuhan perangkat keras dan perangkat lunak yang dilakukan pada bab tiga. Perangkat keras yang dimaksud disini adalah perangkat keras yang dipakai untuk menjalankan sistem. Sedangkan perangkat lunak disini dapat berupa sistem operasi maupun aplikasi yang dapat menunjang keberhasilan penelitian.

Tahap selanjutnya dalam penelitian ini adalah melakukan perancangan untuk membentuk sistem yang menerapkan sistem *gateway discovery* pada *Wireless Sensor Network (WSN)* untuk menemukan *gateway* secara otomatis yang diterapkan pada modul *LoRa*. Perancangan disini menjabarkan tentang gambaran umum dari sistem yang akan dibuat, alur kerja dari sistem, arsitektur utama dari sistem, perancangan pengujian yang akan dilakukan dan juga tentunya komponen apa saja yang ada dalam sistem. Penelitian ini memiliki dua komponen utama yaitu *sensor node* dan *gateway* yang akan dijelaskan perancangannya pada bab perancangan dan implementasi secara lebih detail. Tahap perancangan ini merupakan tahap penting yang akan menjadi acuan untuk tahap implementasi. Pada tahap implementasi, menggunakan rancangan dari tahap selanjutnya. Implementasi ini bertujuan untuk mengimplementasikan rancangan komponen sistem yang telah dibuat pada tahap sebelumnya. Komponen yang harus diimplementasikan ada dua yaitu *sensor node* dan *gateway*. implementasi ini berupa implementasi dari perangkat keras yang telah dirancang juga berisi kode sumber untuk membentuk sistem secara utuh.

Selanjutnya terdapat tahap pengujian dari sistem yang telah diimplementasikan. Tahap ini baru bisa dilakukan jika implementasi selesai dilakukan dan dapat berjalan dengan baik. Tahap pengujian ini dilakukan menurut skenario yang telah disiapkan terlebih dahulu. Skenario tersebut dibuat untuk memenuhi kebutuhan fungsional dari sistem yang telah dibuat. Selain itu juga dilakukan pengujian performa dari protokol *LoRa* yang diimplementasikan melalui modul *LoRa* bertipe *HopeRF-RFM9x*. Performa yang akan diuji meliputi performa jarak efektif dimana kedua komponen dari sistem dapat bertukar data. Hasil dari pengujian ini akan dianalisis guna untuk memenuhi tahap selanjutnya yaitu penarikan kesimpulan dan saran. Setelah hasil dari pengujian dianalisis, maka selanjutnya akan dilakukan penarikan kesimpulan berdasarkan penelitian yang telah dilakukan. Selanjutnya akan diberikan saran yang dapat digunakan untuk referensi penelitian di masa depan.

3.1 Studi literatur

Studi literatur digunakan sebagai acuan untuk melakukan penelitian yang bisa berasal dari sumber manapun yang berkaitan dengan penelitian. Studi literature digunakan juga untuk memberikan pengetahuan yang cukup guna menunjang penelitian untuk implementasi *gateway discovery* pada LoRa. Berikut teori atau literatur yang digunakan untuk melengkapi dan menunjang penelitian:

1. *Wireless Sensor Network (WSN)*

Studi literatur tentang WSN membahas tentang pengertian secara umum dan juga komponen penyusun WSN yang berhubungan dengan penelitian. Bagian ini juga menjelaskan bagian mana yang akan dijabarkan dan menjadi fokus penelitian ini pada WSN.

2. *LoRa*

Bagian ini menunjukkan definisi tentang *LoRa* dan kelebihan dari teknologi *LoRa*. Selain itu juga ditunjukkan elemen penyusun dari teknologi ini dan kelebihan yang dimiliki oleh teknologi *LoRa*. Kelebihan yang dijelaskan juga diiringi oleh beberapa batasan yang harus dipenuhi untuk dapat menjalankan kelebihannya secara maksimal.

3. Modul *LoRa HopeRF-RFM9x*

Pada bagian ini dilakukan studi tentang apa itu modul *LoRa HopeRF-RFM9x* dan juga dijabarkan alasan menggunakan modul komunikasi dengan tipe ini pada penelitian. Salah satu alasannya adalah kemudahan untuk mendapatkannya di wilayah negara penulis. juga diberikan gambaran visual tentang bentuk dari modul ini.

4. Arduino Nano

Bagian ini menunjukkan studi tentang mikrontroller Arduino Nano dan tipe mana yang dipakai untuk penelitian ini. Dijelaskan juga mengenai spesifikasi dari tipe Arduino Nano yang dipilih. Pemilihan perangkat ini untuk penelitian karena kemudahan mendapatkannya di internet maupun di toko elektronik selain dari harganya yang relatif terjangkau yaitu dibawah 100000 rupiah. Pada bagian ini juga dijelaskan mengenai sistem *gateway discovery* yang nantinya akan diterapkan pada penelitian ini.

5. Arduino IDE

Pada bagian ini dilakukan studi tentang pengertian dari Arduino IDE sebagai perangkat lunak utama untuk pemrograman pada Arduino Nano. Selain itu juga dijabarkan tentang penambahan library yang dibutuhkan untuk menjalankan sistem. Pada sistem yang akan dibuat perlu ditambahkan *library*

yang akan membantu penelitian agar dapat mempermudah penggunaan suatu perangkat maupun membantu suatu fungsi tertentu.

6. Parameter Pengujian

Pada bagian ini dijabarkan parameter pengujian yang digunakan untuk menguji performa dari sistem yang telah dibuat. Parameter tersebut dijelaskan menurut pandangan penulis dan diberikan rumus perhitungan untuk setiap parameter.

Literatur ini diperoleh dari bermacam-macam sumber yang tersedia seperti ebook, jurnal, website, buku dan lain-lain.

3.2 Rekayasa Kebutuhan

Rekayasa kebutuhan diperlukan untuk membantu memperlancar penelitian yang akan dilakukan. Hal ini bisa menjadi tahap yang menentukan hasil kesuksesan penelitian ke depannya. Analisis kebutuhan sistem juga berfungsi untuk mengetahui hal apa saja yang dibutuhkan untuk membangun sistem yang menerapkan sistem *gateway discovery* untuk menemukan *gateway* secara otomatis pada perangkat *Wireless Sensor Network (WSN)* berbasis modul komunikasi LoRa. Mendefinisikan kebutuhan sistem terlebih dahulu dapat memberi kemudahan untuk melakukan tahap perancangan dan implementasi sistem. Analisis kebutuhan pada penelitian ini dapat dijabarkan sebagai berikut:

3.2.1 Analisis Kebutuhan Pengguna

Kebutuhan pengguna dalam penelitian ini adalah *sensor node* yang dimiliki oleh pengguna dapat menemukan *gateway* secara otomatis dengan menerapkan sistem *gateway discovery*. Selanjutnya *sensor node* juga harus dapat mengirimkan data yang diperoleh dari sensor menuju ke *gateway* yang benar dan *gateway* akan menampilkan data dari *sensor node* tadi.

3.2.2 Analisis Kebutuhan Sistem

Kebutuhan sistem tentang hal-hal yang terdapat pada penelitian ini dijelaskan sebagai berikut:

1. *Sensor node* dapat menemukan *gateway* secara otomatis dengan pengaturan yang telah dilakukan.
2. *Sensor node* dapat mengambil data sensor suhu dan kelembaban.
3. *Sensor node* dapat mengirim data yang diperoleh dari sensor ke *gateway* yang tepat.

4. *Gateway* dapat mengirimkan *broadcast* untuk mendeklarasikan bahwa *gateway* ini adalah *gateway* yang tepat. Selanjutnya *gateway* akan menunggu data yang akan dikirim oleh *sensor node*.
5. *Gateway* hanya akan mengolah data yang diterima jika memenuhi syarat yang ada.

3.2.3 Analisis Kebutuhan Perangkat Keras

Pada penelitian ini dibutuhkan perangkat keras untuk melakukan implementasi dan pengujian. Kebutuhan perangkat keras ini harus dipenuhi agar sistem dapat diketahui hasil kerjanya dalam dunia nyata. Kebutuhan perangkat keras dapat dilihat pada tabel 3.1.

Tabel 3.1 Kebutuhan Perangkat Keras

Nama Perangkat	Keterangan
Modul LoRa HopeRF-RFM9x	Perangkat ini berfungsi sebagai perantara untuk mengirim data pada <i>sensor node</i> dan menerima data sensor pada <i>gateway</i> (dibutuhkan <i>sensor node</i> dan <i>gateway</i>)
Sensor DHT 11	Sensor yang berfungsi untuk mengambil data kelembapan dan suhu yang ditanamkan pada sisi <i>sensor node</i> (dibutuhkan <i>sensor node</i>)
Arduino Nano	Mikrokontroler yang berisi pengaturan dan algoritma sistem untuk mengatur alur kerja <i>sensor node</i> (dibutuhkan <i>sensor node</i> dan <i>gateway</i>)
Laptop	Berfungsi untuk membuat kode yang akan ditanamkan pada Arduino dan media untuk <i>upload</i> kode program menggunakan aplikasi Arduino IDE (dibutuhkan <i>sensor node</i> dan <i>gateway</i>)

Tabel kebutuhan perangkat keras di atas sudah termasuk kebutuhan perangkat keras pada kedua sisi komponen yang ada pada sistem yaitu pada sisi *sensor node* dan *gateway*. Perangkat keras di atas akan dihubungkan untuk membentuk komponen sistem sesuai dengan perancangan yang akan dilakukan nantinya untuk menghasilkan sistem sesuai tujuan penelitian. Selain dari perangkat keras yang menyusun dua komponen utama, juga menggunakan laptop yang bisa digunakan untuk membuat kode program dan juga untuk memberi daya pada kedua komponen tersebut. Laptop ini juga dapat berperan sebagai *interface* dari sistem ketika dijalankan karena dapat tersambung dengan komponen utama.

3.2.4 Analisis Kebutuhan Perangkat Lunak

Pada bagian ini akan berisi tentang perangkat lunak apa saja yang digunakan untuk melakukan penelitian. Kebutuhan perangkat lunak ini dapat berupa *library* yang dipakai, sistem operasi yang dipakai serta aplikasi. Perangkat lunak yang digunakan akan dijabarkan berdasarkan kebutuhan perangkat lunak pada kedua komponen utama sistem yaitu pada sisi *sensor node* dan *gateway*. Kedua komponen tersebut terdiri dari berbagai perangkat keras harus dibekali dan dibantu oleh perangkat lunak yang tertanam pada mikrokontroler maupun tertanam pada laptop. Dan untuk pembuatan kode program juga dibutuhkan perangkat lunak pada laptop yaitu Arduino IDE. Untuk lebih jelasnya, berikut adalah perangkat lunak yang dibutuhkan dalam penelitian dan dapat dilihat pada tabel 3.2.

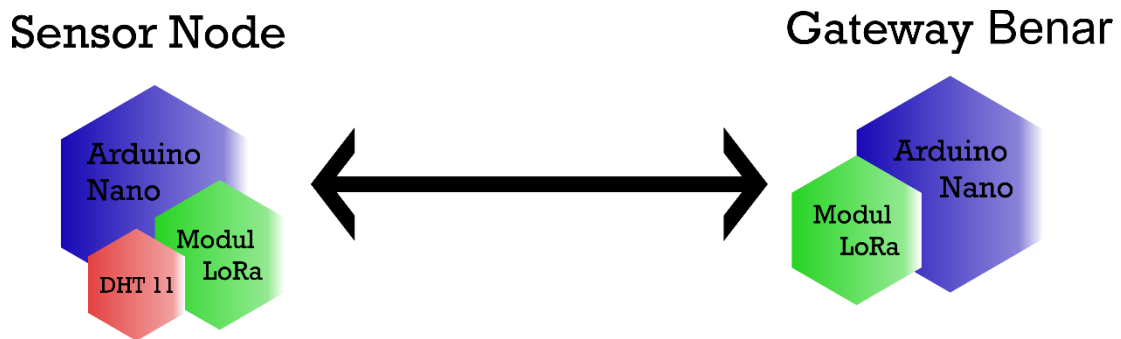
Tabel 3.2 Kebutuhan Perangkat Lunak

Nama Perangkat	Keterangan
Sistem Operasi Windows	Sistem operasi untuk laptop yang digunakan untuk membuat kode
Arduino IDE	IDE untuk melakukan pemrograman pada Arduino Nano dan untuk <i>uploading</i> kode ke Arduino Nano
Library Radiohead pada Arduino IDE	Library yang ditambahkan ke Arduino IDE untuk dapat mempermudah pemrograman yang berhubungan dengan LoRa
Library DHT sensor pada Arduino IDE	<i>Library</i> yang ditambahkan ke Arduino IDE untuk dapat memudahkan pemrograman saat menggunakan sensor DHT11

Perangkat lunak yang dibutuhkan pada penelitian ini bermacam jenisnya seperti yang dapat dilihat pada tabel 3.2 di atas, dimana terdapat sistem operasi yang dibutuhkan untuk penelitian yaitu sistem operasi yang digunakan pada laptop. Selain itu terdapat Arduino IDE untuk melakukan pembuatan kode program ataupun editor dan *uploading* kode program ke mikrokontroler Arduino Nano. *Library* juga termasuk dalam perangkat lunak yang digunakan karena tanpa adanya *library* tersebut maka sistem tidak akan dapat berjalan sebagaimana mestinya. *Library* yang dibutuhkan terdiri dari dua *library* dan harus ditambahkan pada Arduino IDE terlebih dahulu. *Library* tersebut meliputi *library* RadioHead dan *library* DHT sensor. *Library* RadioHead merupakan *library* yang mendukung penggunaan perangkat modul LoRa dan dapat membantu menjalankan beberapa fungsi yang diperlukan dalam penelitian ini. Sedangkan *library* DHT sensor membantu penggunaan sensor DHT11 untuk mengambil data suhu dan kelembapan yang akan diletakkan pada sisi *sensor node*.

3.3 Perancangan

Dalam tahap ini dilakukan perancangan untuk sistem yang akan dibuat. Rancangan ini akan dibuat berdasarkan analisis kebutuhan yang telah dibuat sebelumnya. Perancangan disini meliputi bagaimana sistem akan dibuat dan dioperasikan untuk melancarkan tahap selanjutnya yaitu implementasi. Pada tahap ini terdapat 2 jenis perancangan komponen utama sistem yaitu *gateway* dan *sensor node*. Untuk membantu pemahaman dari komponen sistem yang ada dan akan digunakan dapat dilihat pada gambar berikut:

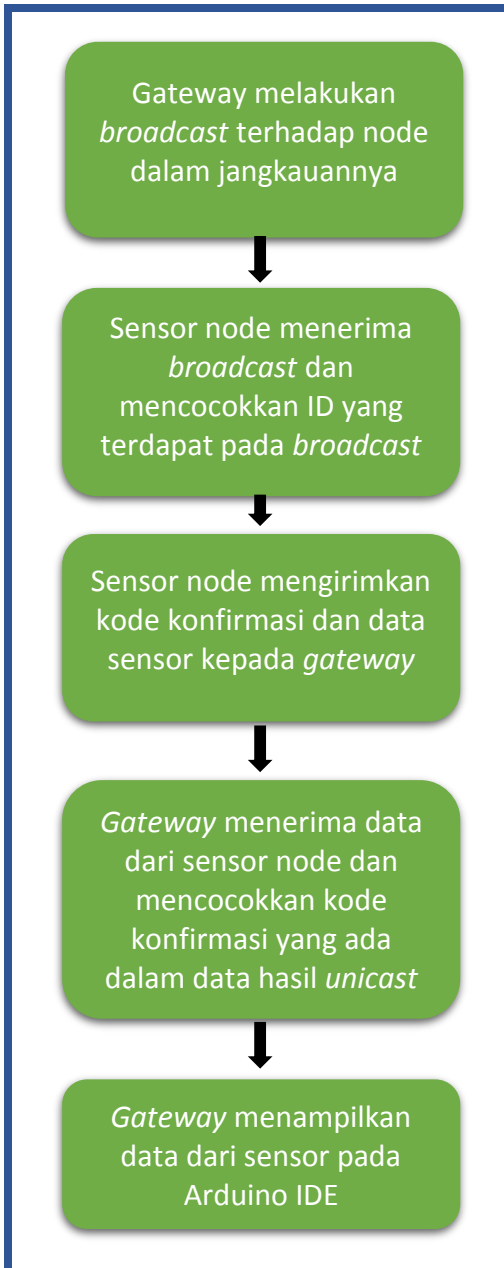


Gambar 3.2 Rancangan Umum Sistem

Gambar 3.2 menjelaskan rancangan sistem secara umum yang akan dibangun pada penelitian ini. Seperti yang dijelaskan sebelumnya, sistem terdiri dari dua komponen yang terdiri dari *gateway* dan *sensor node*. *Gateway* akan bertindak sebagai penerima data dari *sensor node* dan akan dijabarkan gambaran tentang cara kerjanya. Selain itu juga akan dijabarkan perangkat apa saja yang digunakan untuk membentuk *gateway*. Komponen kedua yaitu *sensor node*. Pada *sensor node* inilah sistem *gateway discovery* sebagian besar diletakkan untuk menemukan *gateway* secara otomatis. *Sensor node* akan bisa menerima data dari *gateway* yang salah tetapi data akan dibuang jika tidak memberikan ID yang tepat. Oleh karena itu *sensor node* tidak akan dapat mengirimkan data ke *gateway* yang salah. Berdasarkan uraian tersebut, perancangan sistem pada penelitian ini direncanakan terdiri dari perancangan alur kerja sistem, perancangan *node sensor*, perancangan *gateway* dan perancangan pengujian. Perancangan akan mempermudah untuk tahap implementasi dan tahap setelahnya karena sudah diberikan kerangka untuk mengerjakan sistem yang direncanakan pada penelitian ini. Perancangan sistem juga dilakukan dengan mempertimbangkan teori dan studi literatur tentang beberapa bagian yang menjadi pendukung penelitian.

Bagian pertama perancangan adalah perancangan alur kerja sistem yang berisi gambaran tentang perancangan cara kerja *sensor node* dan perangkat penyusunnya.

Kedua komponen tersebut dijelaskan cara kerjanya satu persatu. Sebelum itu juga diberikan alur kerja secara umum dari sistem yang dibuat dalam penelitian ini. Alur kerja ini mencakup cara kerja dari sistem secara umum dan tidak menggunakan cara pandang dari satu sisi komponen saja melainkan dari cara pandang sistem secara utuh. Untuk mempermudah penjelasan mengenai alur kerja sistem secara umum dapat dilihat pada gambar 3.3.



Gambar 3.3 Alur Kerja Sistem Secara Umum

Gambar 3.3 di atas menunjukkan alur kerja sistem secara umum. Hal yang perlu dilakukan sistem pertama kali adalah melakukan *broadcast* oleh *gateway* dengan tujuan awalnya adalah mengumumkan bahwa *gateway* ini adalah *gateway* yang benar. *Broadcast* tadi ditujukan kepada *sensor node* dan *sensor node* harus mencocokkan ID yang diterima waktu *broadcast*. Jika ID cocok maka data dari sensor akan dikirimkan bersama dengan kode konfirmasi ke *gateway* yang tepat. Kode konfirmasi dan data dari *sensor node* akan dicocokkan dan ditampilkan hasilnya setelah kode konfirmasi dinyatakan benar dan sesuai ketentuan. Setelah itu juga akan dirancang bagaimana proses pengujian akan dilakukan dan parameter apa saja yang akan diuji nanti. Untuk penjelasan lebih lanjut dan detail mengenai perancangan yang akan dilakukan, dapat dilihat pada bab perancangan. Setelah penjelasan alur kerja sistem perancangan akan dibagi menjadi 3 bagian yang akan dirancang meliputi perancangan *sensor node*, perancangan *gateway* dan perancangan pengujian.

Perancangan pertama yaitu perancangan *sensor node* yang akan melakukan sistem *gateway discovery*. *Sensor node* yang dikembangkan dalam penelitian ini akan dikembangkan menggunakan Arduino IDE untuk membuat kode program. Kode program ini nantinya akan ditanamkan pada mikrokontroler Arduino Nano. *Sensor node* pada penelitian ini harus mampu menemukan *gateway* secara otomatis dengan beberapa pengaturan yang telah dilakukan pada kode program yang ditanamkan dan dengan sedikit bantuan dari sisi *gateway*. Kemampuan *sensor node* untuk menemukan *gateway* secara otomatis inilah yang menjadi sasaran penerapan sistem *gateway discovery*. Selain penerapan sistem *gateway discovery*, *sensor node* juga harus bisa untuk mengambil data dari sensor yang terhubung dengan mikrokontroler dan mengolah datanya sehingga dapat dikirim ke *gateway* sesuai format yang telah disepakati sebelumnya.

Tahap perancangan selanjutnya adalah perancangan *gateway* yang akan menjadi awal proses dari sistem *gateway discovery*. *Gateway* akan dibekali dengan kode program yang terlebih dahulu dibuat menggunakan Arduino IDE dan selanjutnya ditanamkan pada mikrokontroler Arduino Nano sama seperti pada *sensor node*. Pada kode program di sisi *gateway* diperlukan *library* untuk menangani beberapa fungsi modul *LoRa* yang nantinya akan menjadi bagian dari sistem *gateway discovery*. Perancangan *gateway* ini bertujuan untuk mempersiapkan implementasi *gateway*, agar nantinya *gateway* dapat bekerja sesuai dengan keinginan. *Gateway* pada awal berjalannya harus mampu melakukan *broadcast* yang berisi informasi tertentu yang sudah ditentukan dan harus dapat diuraikan dengan benar pada sisi *sensor node*, karena format dari data sudah disepakati oleh kedua sisi sebelumnya. *Gateway* akan memasuki mode *listen* yang berarti berada pada keadaan siap untuk menerima data dari *sensor node* yang menerima *broadcast* tadi. Tugas dari *gateway* akan selesai ketika data dari *sensor node* diterima dan ditampilkan hasilnya serta dapat menyimpan ID *sensor node* yang telah mengirimkan data sebelumnya.

Pada akhir bagian perancangan ini membahas tentang perancangan pengujian yang berguna untuk mempermudah memahami apa yang akan diujikan nantinya. Perancangan pengujian yang akan dilakukan meliputi perancangan pengujian fungsionalitas dan perancangan pengujian performa. Bagian pertama perancangan pengujian yaitu pengujian fungsionalitas yang akan membahas pengertian pengujian fungsionalitas dan menunjukkan kriteria tempat untuk pengujian. Selanjutnya bagian perancangan untuk pengujian performa yang akan menjelaskan tentang performa apa saja yang akan diuji dari penerapan *LoRa*.

Pada pengujian fungsionalitas sistem akan digunakan beberapa skenario untuk menguji sistem yang telah dibuat dan tentunya kebutuhan fungsional sistem telah diberikan sebelumnya. Pengujian fungsionalitas juga dapat diartikan sebagai pengujian keberhasilan sistem karena dengan terpenuhinya kebutuhan fungsional sistem, penelitian ini juga bisa dikatakan berhasil. Selanjutnya adalah pengujian performa sistem yang dibuat ketika dijalankan. Pengujian performa yang direncanakan terdiri dari tiga parameter utama yang akan diuji terhadap jarak. Parameter pertama adalah pengujian waktu yang diperlukan sistem untuk berkomunikasi dari waktu *broadcast* sampai diterima oleh *sensor node* dan berakhir waktu *unicast* yang berisi data diterima *gateway*. Waktu ini nantinya akan disebut sebagai *association time* yang berarti waktu total untuk berkomunikasi dari waktu *broadcast* sampai waktu data sensor diterima *gateway*. Untuk lebih detailnya akan dijelaskan pada subbab perancangan pengujian. Waktu yang akan digunakan untuk menguji merupakan waktu relatif yang diambil dari waktu dari awal Arduino Nano dihidupkan dengan memanfaatkan fungsi *millis* pada Arduino IDE. Parameter selanjutnya adalah *broadcast loss* yang akan diukur berdasarkan 100% dikurangi dengan berapa *broadcast* yang diterima oleh *sensor node* dibagi dengan jumlah *broadcast* yang dikirimkan oleh *gateway* dan dikali 100%. Parameter terakhir adalah *unicast loss* yang akan dihitung berdasarkan 100% dikurangi jumlah *unicast* yang diterima oleh *gateway* yang akan dibagi dengan jumlah *unicast* yang dikirim oleh *sensor node* dan dikali 100%. Persamaan ketiga parameter tersebut sudah dijelaskan pada bab sebelumnya. Ketiga parameter di atas akan diuji berdasarkan jarak antara *gateway* dan *sensor node* yang akan divariasikan.

3.4 Implementasi

Implementasi akan dilakukan berdasarkan perancangan sistem yang telah dibuat pada bab perancangan. Selain itu, implementasi juga dilakukan jika semua tahap dalam perancangan telah selesai dilakukan. Dalam penelitian ini implementasinya juga termasuk membuat *sensor node* dan *gateway* yang telah dilengkapi perangkat *LoRa* saling berkomunikasi. Komunikasi antar dua komponen ini dijemput dengan adanya perangkat teknologi *LoRa* yaitu modul *LoRa* di kedua sisi komponen. Dengan begitu *sensor node* dapat menentukan *gateway* secara otomatis untuk mengirim data.

Implementasi juga tidak terbatas pada implementasi kode sumber (*source code*) saja, namun juga implementasi perangkat yang dibutuhkan untuk membentuk masing-masing komponen penyusun sistem. Implementasi ini menjadi bagian penting dalam penelitian karena pengujian hanya bisa dilakukan ketika implementasi selesai dan berhasil dilakukan dengan baik. Implementasi secara garis besar dibagi menjadi dua komponen utama yang harus diimplementasikan yaitu *sensor node* dan *gateway*.

1. Implementasi *Sensor Node*

Tahap implementasi *sensor node* ini dimulai dengan implementasi perangkat keras yang menyusun komponen pertama ini. Komponen perangkat keras tersebut terdiri dari tiga komponen utama yaitu Arduino Nano sebagai mikrokontroler, sensor DHT 11 sebagai sensor untuk mengambil data suhu dan kelembapan dan yang terakhir modul *LoRa* HopeRF-RFM9x. ketiga perangkat keras ini dihubungkan melalui pin-pin yang ada pada ketiga perangkat tersebut menggunakan kabel *jumper female-female*. Untuk kombinasi pin-pin pada ketiga perangkat akan dibahas lebih detail pada bab perancangan.

Tahap selanjutnya setelah implementasi perangkat keras adalah implementasi perangkat lunak yang dibutuhkan berupa kode program untuk *node sensor*. Instalasi untuk aplikasi editor kode program tidak akan ditampilkan karena bukan merupakan inti dari penelitian ini. Hal yang diimplementasikan disini adalah implementasi dari kode program dan penjelasan tentang fungsi-fungsi yang digunakan.

2. Implementasi *Gateway*

Implementasi *gateway* merupakan tahap kedua dalam bab implementasi. Pada bagian ini dijelaskan implementasi untuk pembuatan *gateway*. Seperti halnya implementasi *sensor node*, implementasi *gateway* dimulai dengan implementasi perangkat keras yang menjadi penyusun komponen *gateway*. *Gateway* terdiri dari dua perangkat keras utama yang harus dihubungkan yang hampir sama dengan *sensor node* tetapi dengan menghilangkan sensor DHT11 yang berarti terdiri dari mikrokontroler Arduino Nano dan modul *LoRa* HopeRF-RFM9x. Kedua perangkat keras ini dihubungkan dengan kabel *jumper female-female* melalui pin-pin yang tersedia pada kedua perangkat tersebut sama seperti halnya pada komponen *sensor node*.

Setelah implementasi perangkat keras selesai dilakukan, selanjutnya adalah implementasi dari perangkat lunak. Sama halnya seperti komponen sebelumnya yaitu *sensor node*, instalasi aplikasi juga tidak akan menjadi fokus disini. Fokus utama pada implementasi perangkat lunaknya pastinya kode program penyusun dari *gateway*. Kode program ini akan diberi penjelasan detail mengenai fungsi-fungsi yang digunakan pada komponen ini.

3.5 Pengujian

Pengujian sistem dilakukan untuk mengetahui apakah kode sumber dan juga implementasi yang sudah kita lakukan berhasil memenuhi tujuan penelitian ini atau tidak. Pengujian akan dilakukan dengan beberapa skenario yang telah disiapkan. Skenario tersebut dibuat berdasarkan kebutuhan fungsional yang telah dijabarkan pada subbab analisis kebutuhan dan diperjelas pada bab perancangan. Skenario ini akan menjadi bahan analisis hasil penelitian dan juga tolak ukur keberhasilan penelitian dilihat dari terpenuhinya kebutuhan fungsional atau tidak. Selain itu tidak menutup kemungkinan untuk terciptanya skenario baru diluar kebutuhan fungsional dikarenakan beberapa hal seperti bertambahnya parameter pengujian.

Dalam penelitian ini, tahap pengujian akan menguji komunikasi antar perangkat *LoRa* yang sudah tersambung dengan mikrokontroler Arduino Nano pada kedua komponen utama sistem dalam menentukan *gateway* untuk pengiriman data dari *sensor node*. *Gateway* harus ditemukan secara otomatis oleh perangkat tanpa melakukan pengaturan atau dengan pengaturan seminim mungkin yang disebut *zero configuration*. Mekanisme *zero configuration* ini akan difungsikan setelah implementasi sistem berhasil, yang artinya setelah implementasi dilakukan tidak boleh melakukan pengaturan yang terlalu banyak agar tidak keluar dari konsep *zero configuration*. Pengujian ini dilakukan untuk mencapai tujuan penelitian dalam *gateway discovery* dan juga untuk memenuhi kebutuhan fungsional.

Selain itu juga akan dilakukan pengujian terhadap performa dari sistem untuk pengiriman data dari kedua sisi. Performa yang akan diuji adalah *association time*, *broadcast loss* dan *unicast loss* yang akan diuji terhadap variasi jarak antara *sensor node* dan *gateway*. Penjelasan tentang pengujian secara rinci akan dijabarkan pada perancangan pengujian dan istilah yang menyangkut parameter pengujian telah dijelaskan sebelumnya dalam studi literatur. Setelah pengujian ini selesai dilakukan maka dapat ditarik kesimpulan berdasarkan hasil pengujian dan analisis hasilnya.

3.6 Pengambilan kesimpulan dan saran

Pengambilan kesimpulan dilakukan setelah melakukan pengujian dan melakukan analisis pada hasil pengujian yang telah dilakukan. Kesimpulan harus sesuai dengan fakta dan hasil yang diperoleh pada bab sebelumnya. Selain itu akan diberikan saran untuk pengembangan sistem berikutnya maupun pengembangan penelitian di masa depan. Saran ini juga dapat diteruskan dan dijadikan bahan penelitian di masa depan untuk kemajuan ilmu pengetahuan.

Bab ini adalah bab terakhir dalam rangkaian penelitian yang sudah dilakukan. Bab ini berfungsi sebagai bab penutup dan juga sebagai bab dimana hasil penelitian yang sudah dilakukan disimpulkan dan diringkas agar mudah dipahami oleh pembaca. Bab kesimpulan ini sendiri harus dibuat setelah semua tahap penelitian sudah dilakukan.

BAB 4 REKAYASA KEBUTUHAN

Pada bab ini akan dijabarkan kebutuhan untuk sistem gateway discovery yang dibuat pada *Wireless Sensor Network* berbasis modul komunikasi *LoRa*. Bab ini terdiri beberapa bagian, yaitu pendahuluan, tujuan, deskripsi umum, kebutuhan antarmuka eksternal dan kebutuhan fungsional.

4.1 Pendahuluan

4.2 Tujuan

Bab rekayasa kebutuhan ini menjelaskan tujuan dan fitur dari sistem, antarmuka sistem, apa yang akan dilakukan sistem, ruang lingkup dari sistem dan batasan tentang bagaimana sistem akan beroperasi.

4.2.1 Ruang Lingkup

Ruang lingkup dari sistem ini yaitu menerapkan sistem *gateway discovery* untuk menemukan *gateway* pada perangkat *WSN* yang berbasis modul komunikasi *LoRa*. *Sensor node* akan menemukan *gateway* secara otomatis dengan bantuan dari *broadcast* dari *gateway* pada awalnya. *Broadcast* mengandung informasi yang akan digunakan untuk autentikasi dan berisi informasi alamat *gateway*. Jika proses autentikasi berhasil, maka *sensor node* akan mengirimkan data beserta kode konfirmasi ke *gateway*. *Gateway* hanya akan mengolah data jika kode konfirmasinya cocok dengan yang dimiliki. Data tersebut akan ditampilkan pada terminal pada sisi *gateway*. Pembuatan sistem ini ditujukan kepada penguji, developer, user sehingga kedepannya sistem ini dapat dikembangkan lebih lanjut dan dapat digunakan untuk keperluan pembelajaran, industri atau penelitian.

4.2.2 Istilah

Tabel 4.1 Tabel Istilah

Term	Definition
<i>Wireless Sensor Network (WSN)</i>	<i>Wireless Sensor Network (WSN)</i> merupakan jaringan nirkabel yang terdiri dari perangkat otonom yang didistribusikan secara spasial dan menggunakan sensor untuk memantau lingkungan sekitar
<i>Gateway Discovery</i>	<i>Gateway discovery</i> dalam penelitian ini dapat diartikan sebagai proses untuk menemukan <i>gateway</i> pada <i>WSN</i> dalam pengiriman data dari <i>sensor node</i>

<i>LoRa</i>	LoRa atau <i>Long-Range</i> adalah teknologi nirkabel yang dikembangkan oleh <i>Semtech</i> untuk memungkinkan komunikasi dengan <i>data rate</i> rendah yang dibuat untuk komunikasi dengan jarak yang jauh
<i>Sensor Node</i>	Merupakan sebuah node dalam jaringan sensor yang mampu melakukan beberapa pemrosesan, mengumpulkan informasi sensorik dan berkomunikasi dengan node lain yang terhubung dalam jaringan
<i>Gateway</i>	Merupakan komponen yang akan menerima data dari <i>sensor node</i> dan mengawali komunikasi dalam sistem dengan melakukan <i>broadcast</i> .
<i>Stakeholder</i>	Setiap orang yang berkepentingan dengan sistem ini atau proyek yang bersangkutan, selain <i>developer</i>
<i>Developer</i>	Seseorang yang mempunyai kemampuan meneliti artikel atau penelitian dan juga memiliki kemampuan untuk merekomendasikan persetujuan dari artikel untuk publikasi atau meminta perubahan harus dibuat dalam artikel
<i>Software Requirements Specification (SRS)</i>	Sebuah dokumen yang benar-benar menggambarkan sebuah fungsi dari sistem yang diusulkan dan bagaimana kebutuhan agar sistem dapat beroperasi
<i>Broadcast</i>	Merupakan suatu metode pengiriman data dimana data tersebut dikirim ke banyak titik sekaligus tanpa melakukan pemeriksaan atau pengecekan apakah titik tersebut siap untuk menerima atau tidak dan tanpa memperhatikan tersampainya data.
<i>User</i> atau pengguna	<i>Reviewer</i> atau penulis.
<i>Unicast</i>	Merupakan metode pengiriman data yang berasal dari satu titik dan mempunyai tujuan satu titik pula. <i>Unicast</i> ini akan digunakan untuk pengiriman data dari <i>sensor node</i> ke <i>gateway</i> yang tepat.

4.2.3 Sistematika

Untuk mempermudah dalam meninjau pembahasan yang ada, maka diperlukan sistematika yang digunakan sebagai pedoman penulisan *Software Requirements Specification* (SRS) ini. Bagian ini dibagi menjadi 3 bagian yang diuraikan sebagai berikut:

1. Pendahuluan

Bagian ini terdiri dari tujuan, ruang lingkup, istilah dan sistematika penulisan bagian.

2. Deskripsi Umum

Bagian ini terdiri dari perspektif produk, fungsi produk, karakteristik pengguna, batasan-batasan sistem, asumsi dan ketergantungan.

3. Spesifikasi Kebutuhan

Bagian ini terdiri dari kebutuhan fungsional dan kebutuhan antarmuka eksternal.

4.3 Deskripsi Umum

4.3.1 Perspektif Produk / Sistem

Sistem ini dibagi menjadi dua komponen utama yaitu *sensor node* dan *gateway*. *Gateway* akan mengirimkan *broadcast* terlebih dahulu dan selanjutnya akan diterima dan diseleksi oleh *sensor node*. Setelah itu *sensor node* akan mengambil data dari sensor dan mengirimkannya kepada *gateway*. *Gateway* akan menerima data dari *sensor node* dan menampilkannya pada terminal pada *gateway*. Komunikasi antara *sensor node* dan *gateway* dijembatani oleh teknologi *LoRa* yang diimplementasikan melalui modul *LoRa* yang terpasang di kedua komponen tersebut. Sistem ini berfokus kepada kemampuan dari *sensor node* untuk menemukan *gateway* yang benar secara otomatis berbasis pada teknologi *LoRa*. Sistem dalam menangani masalah untuk menemukan *gateway* secara otomatis dengan menerapkan sistem *gateway discovery* berakhir dengan menampilkan data suhu dan kelembapan yang dikirimkan oleh *sensor node*.

4.3.2 Kegunaan

Dengan adanya sistem ini diharapkan pemakaian teknologi *LoRa* maupun perangkat yang menerapkan teknologi ini seperti modul *LoRa HopeRF-RFM9x* dapat lebih efektif. Efektivitas ini dapat bertambah dengan fitur untuk menemukan *gateway* secara otomatis pada *sensor node* khususnya pada *Wireless Sensor Network (WSN)*. Para pengguna dengan pengetahuan yang minim diharapkan dapat menggunakan modul *LoRa* dengan tipe ini dengan lebih mudah. Selain itu karena menggunakan teknologi *LoRa* maka jarak yang bisa ditempuh oleh *sensor node* untuk mengirimkan

data bisa mengakomodasi jarak yang jauh pada lingkungan yang sesuai dan perangkat yang sesuai pula.

4.3.3 Karakteristik Pengguna

Pengguna disini bertindak untuk mengaktifkan perangkat *gateway* dan *sensor node* dengan menyambungkan dengan aliran daya untuk mengaktifkan mikrokontroller pada sisi *sensor node* dan *gateway*. Pada sisi *sensor node*, kode program akan diunggah pada papan mikrokontroller. Sedangkan pada sisi *gateway*, pengguna diharuskan menjalankan kode program yang sudah ditentukan. Sistem ini menerapkan sistem *gateway discovery* yang didukung oleh sedikit mekanisme *zero configuration*, oleh karena itu pengguna hanya bertindak menjalankan program dan menunggu tampilan data suhu dan kelembapan pada sisi *gateway* melalui terminal.

4.3.4 Lingkungan Operasi

Persyaratan kebutuhan lingkungan yang mendukung kebutuhan sistem dapat dijabarkan sebagai berikut:

1. *Gateway* harus dapat berkomunikasi dengan *sensor node* melalui *broadcast*,
2. *Sensor node* harus dapat mengirim data ke *gateway* yang benar melalui *unicast*.

4.3.5 Batasan Perancangan dan Implementasi

Beberapa batasan dalam perancangan dan implementasi akan dijelaskan sebagai berikut:

1. Implementasi akan dijalankan dengan menghubungkan *sensor node* dengan laptop dimana telah terpasang Arduino IDE untuk kepentingan pengujian,
2. Pengujian akan dijalankan saat kedua komponen utama, yaitu *sensor node* dan *gateway* telah dijalankan,
3. Data suhu dan kelembapan dari *sensor node* hanya akan ditampilkan pada terminal tanpa disimpan pada sisi *gateway*,
4. Data suhu dan kelembapan yang dikirimkan ke *gateway* berasal dari sensor DHT 11,
5. *Output* dari pengujian didapatkan pada Arduino IDE dengan menjalankan antarmuka dari port yang digunakan untuk memberi daya pada Arduino Nano.
6. Data yang dipertukarkan berbentuk *struct*.

4.3.6 Asumsi dan Ketergantungan

Beberapa asumsi dan ketergantungan sebagai persyaratan sistem dapat dijabarkan sebagai berikut:

1. *Gateway* dan *sensor node* harus sepakat tentang jenis data yang dipertukarkan,
2. *Gateway* dan *sensor node* harus saling mengetahui format data yang akan diterima maupun dikirim,
3. Format data telah ditentukan terlebih dahulu, untuk menerapkan sistem *gateway discovery* ,
4. Komunikasi antara kedua komponen yang dijumpai modul *LoRa* dipengaruhi oleh beberapa faktor seperti kondisi geografis lingkungan pengujian, kepadatan bangunan, antena yang digunakan, tipe modul *LoRa* yang digunakan hingga gangguan *noise* disekitar modul *LoRa* digunakan.

4.4 Kebutuhan Antarmuka Eksternal

4.4.1 Antarmuka Pengguna

Pengguna dapat melihat data suhu dan kelembapan serta pernyataan apakah ada kontak dari *gateway* yang benar dari sisi *sensor node*. Sebelum itu diperlukan proses *broadcast* dari *gateway* yang akan diterima *sensor node*. Data lengkap yang dapat dilihat pengguna pada sisi *sensor node* hanya ditampilkan jika *sensor node* menerima data. Pengguna juga dapat melihat informasi tentang data suhu dan kelembapan pada *gateway*. Selain itu pengguna juga dapat melihat ID *node sensor* yang telah mengirimkan data pada *gateway*. Data ini bisa dilihat pengguna pada sisi *gateway* jika sebelumnya pencocokan kode konfirmasi berhasil pada sisi *gateway*.

4.4.2 Antarmuka Perangkat Keras

1. *Sensor Node*
 - a. Mikrokontroler/Laptop
2. *Gateway*
 - a. Mikrokontroler/Laptop

4.4.3 Antarmuka Perangkat Lunak

1. *Sensor Node*
 - a. Library RadioHead
 - b. Library DHT sensor
2. *Gateway*
 - a. Library RadioHead

4.5 Kebutuhan Fungsional

Kebutuhan fungsional berisi tentang kebutuhan yang harus dipenuhi sistem dan fungsi yang harus dapat dilakukan oleh sistem sehingga sistem dapat bekerja sesuai dengan tujuan penelitian. Kebutuhan fungsional dapat dilihat pada tabel 4.2.

Tabel 4.2 Kebutuhan Fungsional

No	Kebutuhan Fungsional
1.	<i>Sensor node</i> dapat mengambil data dari sensor suhu dan kelembapan
2.	<i>Sensor node</i> dapat menerima <i>broadcast</i> dari <i>gateway</i> dan mencocokkan id dari <i>gateway</i> dengan id yang dimiliki <i>sensor node</i>
3.	<i>Sensor node</i> dapat memilah data hasil <i>broadcast</i> sesuai format yang disepakati <i>sensor node</i> dan <i>gateway</i>
4.	<i>Sensor node</i> dapat mengetahui alamat <i>gateway</i> yang tepat berdasarkan pemilihan data yang diterima dari <i>broadcast</i>
5.	<i>Sensor node</i> dapat mengirim data melalui modul LoRa ke <i>gateway</i> yang tepat berdasarkan informasi yang diperoleh dari hasil <i>broadcast</i>
6.	<i>Gateway</i> dapat mengirimkan <i>broadcast</i> berisi informasi yang menandakan bahwa <i>gateway</i> tersebut adalah <i>gateway</i> yang tepat
7.	<i>Gateway</i> dapat menerima data suhu dan kelembapan dari <i>sensor node</i>
8.	<i>Gateway</i> dapat memilah data yang diterima dari <i>sensor node</i> berdasarkan format yang telah ditentukan
9.	<i>Gateway</i> menampilkan data suhu dan kelembapan yang diterima sesuai data dari <i>sensor node</i>
10.	<i>Gateway</i> dapat menyimpan ID <i>sensor node</i> yang pernah mengirimkan data

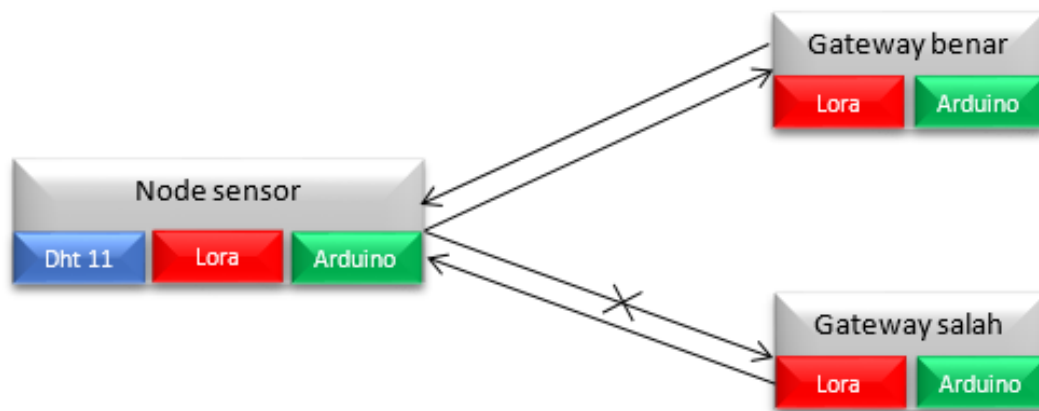
BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada bab ini akan dijelaskan perancangan dan implementasi sistem untuk membentuk sistem perbagian secara lengkap. Proses implementasi pada bab ini dilakukan berdasarkan perancangan sistem. Implementasi yang dilakukan mencakup implementasi pada sisi *sensor node* dan *gateway*.

5.1 Perancangan

5.1.1 Gambaran umum sistem

Sistem yang akan dibuat harus memenuhi tujuan penelitian yaitu *sensor node* yang telah terpasang perangkat LoRa harus bisa mengirim data ke *gateway*. *Sensor node* yang telah terpasang perangkat LoRa juga harus dapat menemukan *gateway* secara otomatis dengan sistem yang telah diatur sebelumnya dalam penelitian ini menggunakan sistem *gateway discovery* yang akan dirancang. Sistem *gateway discovery* diterapkan pada kode program yang terdapat pada kedua komponen yang terdapat pada penelitian yaitu node sensor dan *gateway*. Gambaran sistem secara umum yang menunjukkan perangkat apa saja yang terdapat pada sistem dan bagaimana sistem berkomunikasi dapat dilihat pada Gambar 5.1.



Gambar 5.1 Gambaran umum sistem

Berdasarkan Gambar 5.1 diatas dapat dilihat bahwa Sistem yang akan dibuat terdapat dua macam komponen utama yaitu *sensor node* dan *gateway*. Komponen pertama yaitu *sensor node* berfungsi untuk mengambil data dari sensor dan mengirimkannya ke *gateway* yang tepat dengan perangkat LoRa. Sebagian besar pengaturan penerapan sistem *gateway discovery* diletakkan pada komponen ini. Komponen ini terdiri dari tiga perangkat keras yaitu Arduino nano sebagai mikrokontroller, sensor DHT11 sebagai sensor untuk mengambil data suhu dan

kelembapan serta modul *LoRa HopeRF-RFM9x* sebagai modul untuk berkomunikasi secara *wireless* antar node. Ketiga perangkat ini dihubungkan melalui pin-pin yang ada pada perangkat dengan kabel *jumper*.

Komponen kedua pada sistem ini yaitu *gateway* yang terdiri dari dua perangkat keras saja. Perangkat tersebut adalah Arduino Nano dan modul *LoRa HopeRF-RFM9x* yang keduanya menjadi penyusun utama *gateway* dan berfungsi untuk menerima data dari *sensor node* serta melakukan *broadcast* untuk menunjang sistem *gateway discovery* pada awal dijalankan. Komponen ini sama halnya seperti komponen pertama, menggunakan kode program yang dibuat pada Arduino IDE dan selanjutnya telah dilakukan proses *uploading* pada mikrokontroler Arduino Nano. Perangkat keras pada komponen *gateway* yang hampir mirip komposisinya dengan komponen *sensor* juga dihubungkan melalui pin-pin yang tersedia pada kedua perangkat keras tersebut.

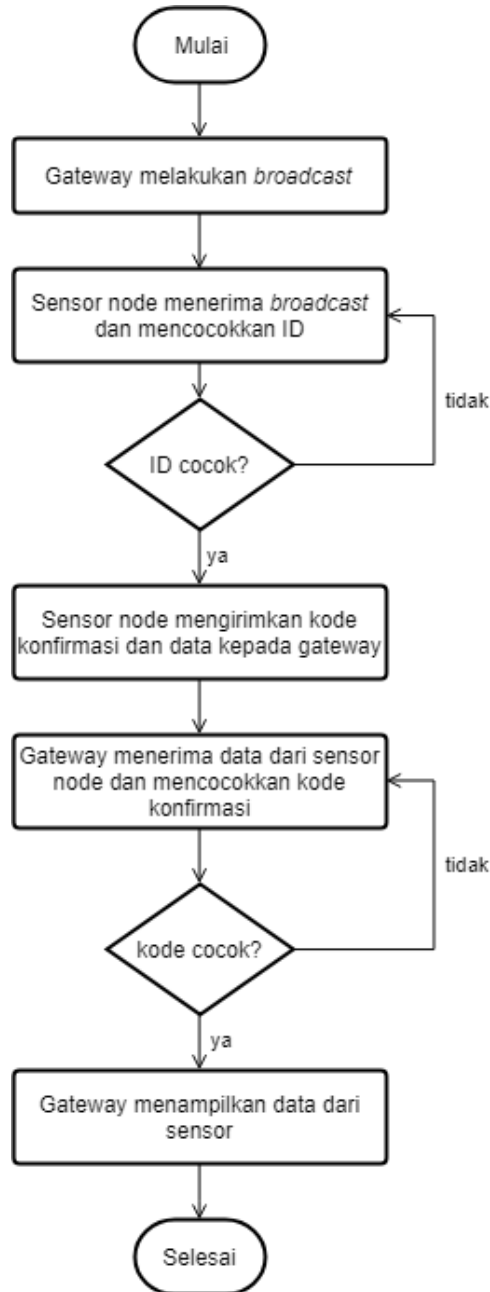
Gambar 5.1 juga menggambarkan sistem yang menerapkan *gateway discovery* dalam aspek dimana *sensor node* harus dapat menemukan *gateway* yang benar untuk pengiriman data secara otomatis dengan beberapa komponen yang terlibat yaitu perangkat *sensor node* dan perangkat *gateway*. Pada gambar diatas terdapat tanda silang yang berarti *sensor node* hanya dapat mengirim data sensor ke *gateway* yang benar. *Sensor node* memang dapat menerima data dari *gateway* lain selain *gateway* yang benar karena berada pada mode *listen* pada awalnya, tetapi data yang diterima jika tidak memiliki ID yang sama dengan ID yang dimiliki *sensor node* akan diabaikan. *Sensor node* hanya akan bereaksi jika data yang diterima berisi ID yang sama dengan yang dimiliki. Dalam hal ini bereaksi berarti *sensor node* akan menyimpan alamat *gateway* dan mengirimkan data sensor beserta kode konfirmasi kepada *gateway* yang benar. Jika *gateway* yang benar mengalami *down* maka *sensor node* akan menunggu *gateway* yang baru untuk melakukan *broadcast*.

Pertukaran data antara kedua komponen ini sudah ditentukan tipe datanya terlebih dahulu dan format datanya sudah disepakati juga oleh keduanya. Proses selanjutnya yang dilakukan *sensor node* adalah mengirimkan data suhu dan kelembapan yang telah diambil dari sensor. Data yang dikirimkan *sensor node* melalui modul *LoRa*, nantinya akan diterima oleh modul *LoRa* juga pada sisi *gateway*. Data yang diterima tadi akan ditampilkan pada sisi *gateway* yang dapat dilihat pada keluaran di Arduino IDE sesuai dengan *port* dimana mikrokontroler itu terhubung. Untuk memperjelas tentang alur kerja sistem secara detail serta bagaimana data diterima oleh masing-masing komponen, dapat dilihat pada subbab berikutnya.

5.1.2 Perancangan Alur Kerja Sistem

Pada bagian ini membahas tentang alur kerja sistem secara keseluruhan dari awal sampai akhir bukan dilihat dari salah satu komponen sistem saja. Cara pandang alur sistem ini mencakup alur data yang dikirim dan data apa saja yang akan dikirimkan baik dari sisi *sensor node* maupun dari sisi *gateway*. Perancangan alur kerja sistem ini

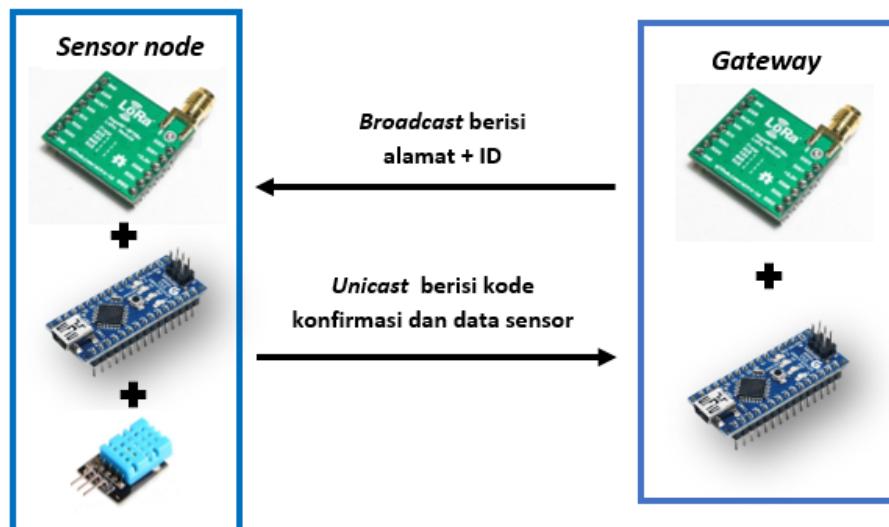
dimaksudkan untuk mempermudah pengerjaan implementasi sistem nantinya dan juga pada saat pengujian. Selain itu juga akan diberikan alur komunikasi dari sistem yang tentu saja melibatkan dua komponen utama sistem. Bentuk penggambaran alur kerja sistem dan alur komunikasi sistem dibuat terpisah untuk lebih memperjelas penjelasan tentang sistem yang akan dibuat. Untuk mempermudah pemahaman diberikan diagram alir yang dapat dilihat pada gambar 5.2.



Gambar 5.2 Perancangan Alur Kerja Sistem

Gambar 5.2 menunjukkan perancangan alur kerja sistem yang dilihat dari cara kerja sistem keseluruhan. Terlihat pada diagram alir di atas bahwa hal yang perlu dilakukan sistem pertama kali adalah *broadcast* yang dilakukan oleh *gateway* dengan tujuan awal berupa pengumuman kepada *sensor node* dalam jangkauannya bahwa *gateway* ini adalah *gateway* yang benar. *Broadcast* tadi juga bertujuan untuk memberikan informasi penting yang nantinya akan membantu terciptanya sistem *gateway discovery* pada sistem ini. Informasi tersebut berisi ID dan alamat dari *gateway* yang benar. Selanjutnya *sensor node* yang menerima *broadcast* tadi akan mencocokkan ID dari *gateway* dengan ID yang dipunyai, jika cocok maka data dari sensor suhu dan kelembapan akan diambil dari sensor DHT 11. Data yang telah didapatkan tersebut akan dikirim kepada *gateway* yang tepat berdasarkan alamat yang tertera pada hasil *broadcast*. Data sensor ini akan dikirimkan bersamaan dengan konfirmasi berupa kode ataupun ID kepada *gateway*. Setelah data diterima oleh *gateway* yang tepat, data tersebut terlebih dahulu dicek kode konfirmasinya. Jika kode konfirmasi cocok, maka data sensor dapat diterima dan disimpan alamat yang berupa ID sensornya. Alamat atau ID sensor ini akan ditampilkan pada sisi *gateway* beriringan dengan data suhu dan kelembapan yang didapat.

Uraian di atas akan diperjelas oleh perancangan alur komunikasi dari sistem secara utuh. Jadi dari uraian di atas, secara garis besar terdapat dua macam komunikasi yang dilakukan antara *sensor node* dan *gateway*. Komunikasi tersebut berupa *broadcast* dari *gateway* dan *unicast* dari *sensor node*. *Broadcast* dari *gateway* berisi alamat *gateway* dan ID yang telah ditentukan sebelumnya sebagai penanda ini adalah *gateway* yang benar, sedangkan *unicast* berisi kode konfirmasi berupa angka yang akan dikirimkan dari *sensor node* dan pastinya data sensor dari DHT11. Untuk lebih jelasnya alur komunikasi sistem dapat dilihat pada gambar 5.3.



Gambar 5.3 Alur komunikasi sistem

Gambar 5.3 menunjukkan komunikasi sistem antara dua komponen yaitu *gateway* dan *sensor node* yang telah dijelaskan sebelumnya. Komunikasi dua komponen tersebut yang dijembatani oleh modul *LoRa* dikirim menggunakan teknik modulasi yang ada pada teknologi *LoRa* yaitu *Chirp Spread Spectrum (CSS)*. Sedangkan alur kerja pada masing-masing komponen dapat dilihat pada subbab alur kerja masing-masing komponen. Data yang dipertukarkan oleh kedua komponen pada gambar 5.3 berbentuk *struct* pada saat *broadcast* maupun *unicast*. Hal ini dimaksudkan agar data yang dikirim tidak terlalu besar jika dibandingkan dengan data berformat JSON yang mengirim data dalam bentuk *string*.

5.1.3 Perancangan *Sensor Node*

Sensor node akan ditanamkan kode program yang sudah dibuat menggunakan Arduino IDE dan dilakukan proses *uploading* kode program pada mikrokontroler Arduino Nano. Pemrograman pada Arduino IDE membutuhkan beberapa *library* yang harus ditambahkan pada Arduino IDE untuk mempermudah menjalankan fungsi tertentu yang berhubungan dengan penggunaan perangkat keras. *Library* pertama yang harus ditambahkan secara manual pada *sensor node* yaitu *RadioHead*. *Library* ini merupakan *library* yang diperlukan untuk mempermudah penggunaan beberapa fungsi yang dibutuhkan pada modul *LoRa*. *Library* kedua yang diperlukan adalah *library* DHT sensor. *Library* DHT sensor ini ditambahkan guna mempermudah penggunaan sensor suhu dan kelembapan yaitu sensor DHT11. Setelah kedua *library* itu ditambahkan maka persiapan untuk membuat kode program pada sisi *sensor node* dapat dilakukan dengan baik dan dapat berjalan tanpa terjadi *error* untuk menjalankan fungsi yang dibutuhkan oleh perangkat keras. Selanjutnya adalah pendefinisian variabel yang diperlukan untuk pertukaran data yang telah disepakati. Data yang dipertukarkan dikirim dengan tipe data *struct*, oleh sebab itu diperlukan pendefinisian variabel bertipe *struct* untuk membuat pertukaran data dapat berlangsung dengan baik.

Selain hal tersebut ada beberapa hal yang harus disiapkan untuk pembuatan *sensor node* ini secara utuh terutama dari sektor perangkat keras. Perangkat keras juga menjadi bagian terpenting dari komponen *sensor node* karena tanpa perangkat keras sistem tidak akan berjalan karena sistem berbentuk nyata bukan *virtual*. Perangkat keras yang digunakan pada sisi *sensor node* salah satunya adalah Arduino nano dan modul *LoRa* yang harus terkoneksi dengan benar melalui perantara kabel *female-female*. Kabel ini biasa disebut dengan *jumper* yang penggunaannya dapat hanya dengan menancapkannya pada pin-pin yang tersedia pada kedua perangkat yang akan dihubungkan, dalam hal ini Arduino Nano dan modul *LoRa*. Komposisi pin yang dihubungkan dari kedua perangkat keras tersebut disebut dengan *pinout*. *Pinout* untuk Arduino Nano dan modul *LoRa* bisa dilihat pada tabel 5.1 untuk lebih jelasnya.

Tabel 5.1 *Pinout* kabel *female-female* Arduino Nano dan Modul LoRa

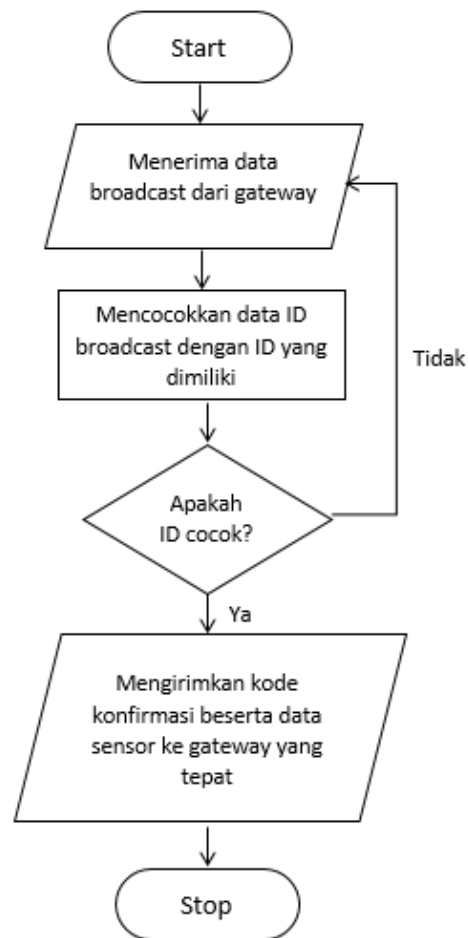
Arduino Nano	Modul LoRa
GND	GND
3V3	+3.3V
D2	DIO0
D10	NSS
D11	MOSI
D12	MISO
D13	SCK

Pada tabel 5.1 terdapat 7 pin dari masing-masing perangkat yang wajib terkoneksi dengan benar dan jika terjadi kesalahan pada koneksi antar pin pada kedua perangkat tersebut, maka kemungkinan terburuk yang terjadi adalah kegagalan sistem. Setelah Arduino Nano dan modul LoRa terkoneksi dengan benar, selanjutnya adalah menghubungkan Arduino Nano dan sensor DHT11. Sensor ini dibutuhkan di sisi *sensor node* untuk mengambil data suhu dan kelembapan yang nantinya akan dikirimkan ke sisi *gateway*. Sama halnya seperti untuk menghubungkan kedua perangkat sebelumnya, kedua perangkat ini dihubungkan dengan kabel *jumper female-female*. Berikut adalah *pinout* untuk menyambungkan Arduino Nano dan sensor DHT11 versi 3 pin yang digunakan dalam penelitian ini dan dapat dilihat pada tabel 5.2.

Tabel 5.2 *Pinout* dengan kabel *female-female* Arduino Nano dan sensor DHT11

Arduino Nano	DHT11
GND	GND
+5V	VCC (+)
D3	Serial Data

Tabel 5.2 yang menunjukkan pin yang harus dihubungkan antara kedua perangkat pada *sensor node* juga harus diperhatikan dengan teliti untuk membuat sistem bekerja dengan baik. Setelah perancangan perangkat keras pada sisi *sensor node* yang sudah sampai pada tahap perakitan bagian-bagian dari komponen ini selesai, akan dilanjutkan dengan perancangan alur kerja dari komponen *sensor node*. Perancangan alur kerja dari *sensor node* juga merupakan hal penting yang harus dilakukan. Hal ini dimaksudkan agar dapat diketahui alur kerja dari sistem dari sudut pandang *sensor node*. Untuk lebih jelasnya perancangan alur kerja dari *sensor node* dapat dilihat pada diagram alir yang ditunjukkan oleh gambar 5.4.



Gambar 5.4 Alur kerja *sensor node*

Gambar 5.4 menunjukkan bahwa *sensor node* menerima *broadcast* dari *gateway*, hal ini sebelumnya disebabkan karena pada awal dijalankan *sensor node* berada pada mode *listen*, dimana *sensor node* akan menunggu *broadcast* dari *gateway*. Pada waktu tersebut, data suhu dan kelembapan yang diperoleh dari sensor belum diambil. Data tersebut akan diambil jika format *broadcast* yang diterima sesuai kesepakatan. Ketika *broadcast* sudah diterima, maka isinya akan dicek. Pengecekan dilakukan dengan mencocokkan ID yang diterima oleh *sensor node* dengan ID yang dimiliki. Hal inilah yang menjadi *trigger* untuk mengambil data dari sensor. Jadi ketika hasil pencocokkan sesuai, maka selanjutnya terjadi proses pengambilan data suhu dan kelembapan. Data yang dibaca dari sensor DHT11 tadi akan dikirimkan ke satu *gateway* yang tepat dengan tambahan kode konfirmasi angka yang telah disepakati sebelumnya. Alamat *gateway* yang tepat didapatkan dari isi *broadcast* yang diterima pada awal berjalannya proses *listen*. Jika pencocokan ini tidak sesuai, maka sistem dari *sensor node* akan kembali kepada mode *listen* untuk menunggu data yang benar.

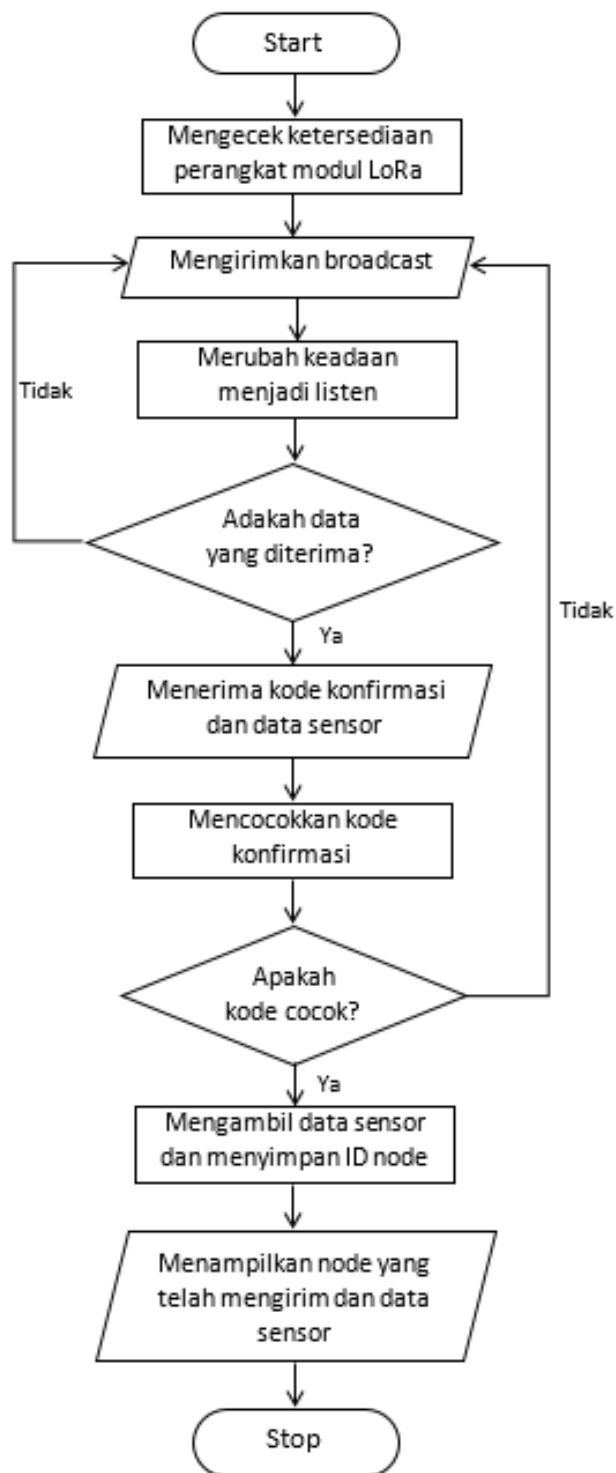
Alur kerja dari *sensor node* berakhir dengan dikirimkannya kode konfirmasi dan data sensor kepada *gateway*.

5.1.4 Perancangan *Gateway*

Gateway merupakan bagian penting dalam penelitian ini karena *gateway* bertugas untuk mengirimkan *broadcast* kepada *sensor node* untuk menandakan bahwa *gateway* ini adalah *gateway* yang benar dan merupakan salah satu pengaturan untuk menerapkan sistem *gateway discovery* nantinya untuk menemukan *gateway*. Untuk itu *gateway* perlu dirancang dengan baik agar sistem dapat berjalan sesuai dengan hasil yang diinginkan. Beberapa hal yang harus disiapkan untuk membuat *gateway* ini sama seperti pada sisi *sensor node* yaitu pertama mempersiapkan perangkat keras yang dibutuhkan terlebih dahulu. *Gateway* terdiri dari dua perangkat keras utama yang saling terhubung yaitu mikrokontroler Arduino Nano dan modul *LoRa* yang sudah terkoneksi dengan benar dengan perantara kabel *female-female*. *Pinout* untuk modul *LoRa* dan mikrokontroler Arduino Nano telah diberikan sebelumnya pada perancangan *sensor node*, bisa dilihat pada tabel 5.1.

Tahap kedua adalah mempersiapkan perangkat lunak yang akan dipakai dan perancangan kode program *gateway*. Kode program pada sisi *gateway* akan dibuat dalam bahasa c atau bisa disebut dengan bahasa Arduino. Kode program ini dibuat dengan editor Arduino IDE dan dilakukan proses *upload* juga dengan perangkat lunak ini. Persamaan antara kedua komponen yaitu komponen *sensor node* dan *gateway* juga merambah pada sisi penambahan *library* yang digunakan. Komponen *gateway* dalam pembuatan kode programnya juga perlu menambahkan *library* yang hampir sama dengan komponen *sensor node*. *Library* tersebut adalah *library RadioHead* untuk mendukung perangkat *LoRa*. Selain itu juga perlu pendefinisian variabel bertipe *struct* untuk mendukung format pengiriman dan penerimaan data yang sudah disepakati terlebih dahulu.

Arduino Nano dipilih sebagai perangkat yang berperan sebagai *gateway* dikarenakan kemudahan untuk memperoleh perangkat ini di Indonesia serta banyaknya referensi dalam bahasa pemrograman Arduino yang dapat membantu kelancaran penelitian ini. Perancangan cara kerja *gateway* telah disesuaikan dengan cara kerja *sensor node* untuk mewujudkan penerapan *gateway discovery* yang didukung oleh kedua komponen sistem tersebut. Untuk memperjelas alur kerja komponen *gateway* pada perangkat Arduino Nano dapat dilihat pada gambar 5.5 dalam bentuk diagram alir.



Gambar 5.5 Alur kerja *gateway*

Program pada sisi *gateway* yang ditunjukkan pada gambar 5.5 berjalan diawali dengan pengecekan ketersediaan perangkat dan pengaturan modul *Lora*, lalu selanjutnya akan dilakukan *broadcast*. Dalam *broadcast* tersebut berisi alamat dari *gateway* yang benar dan ID yang menandakan bahwa *gateway* ini adalah *gateway* yang tepat. Data *broadcast* tersebut akan dikirim dalam format *struct* yang telah disepakati antara kedua komponen. Setelah itu, *gateway* akan berada dalam keadaan *listen* untuk menunggu kode konfirmasi dan data dari *sensor node* yang juga dikirim dalam bentuk *struct*. Jika belum terdapat data yang diterima oleh *gateway*, maka akan dilakukan *broadcast* lagi sampai data yang diterima tersedia. Ketika kode konfirmasi beserta data dari *sensor node* sudah tersedia dan diterima, kode konfirmasi akan dicocokkan dengan kode yang dimiliki oleh *gateway*. Jika kode konfirmasi cocok, maka data suhu dan kelembapan akan diambil dan ditampilkan serta alamat node yang sudah mengirimkan kode konfirmasi dan data sensor akan disimpan. Sebaliknya, jika kode konfirmasi salah atau tidak cocok, maka data akan diabaikan dan akan mengirimkan *broadcast* kembali.

5.1.5 Perancangan Pengujian

Pengujian perlu dirancang untuk menentukan parameter keberhasilan penelitian sesuai dengan skenario yang terlebih dahulu ditentukan. Skenario tersebut harus dilakukan untuk mengetahui hasil dari penelitian ini.

5.1.5.1 Pengujian Fungsional

Pengujian fungsional adalah pengujian yang berfokus pada pengujian fungsional sistem sudah terpenuhi dan dapat berjalan dengan baik. Pengujian ini dilakukan berdasarkan hal-hal yang telah didefinisikan dalam subbab analisis kebutuhan. Sesuai dengan subbab tersebut, pengujian dilakukan dengan menjalankan interaksi antara *gateway* dan *sensor node*. Jika interaksi yang terjadi sesuai dengan yang telah ditentukan, maka pengujian fungsional dianggap berhasil. Selain itu pengujian ini merupakan fokus utama dalam penelitian ini.

Pengujian ini akan dilakukan di tempat yang mendukung modul komunikasi LoRa untuk berjalan dengan lebih baik tanpa banyak gangguan pada perangkat. Tempat yang dimaksud adalah tempat dimana tidak pada lingkungan padat penduduk dan tidak banyak alat yang dapat mengganggu pertukaran data antara perangkat LoRa seperti di lingkungan pedesaan atau persawahan yang luas. Oleh sebab itu, lingkungan pengujian akan dilakukan di desa Kidal, kecamatan Tumpang, kabupaten Malang yang masih mempunyai lahan kosong ataupun area persawahan yang luas dan tidak padat bangunan. Untuk pengujian fungsional sebenarnya dapat dilakukan selain pada kriteria tempat yang dijelaskan sebelumnya, namun hal tersebut ada kemungkinan gangguan yang tidak diinginkan sewaktu pengujian berlangsung. Selain itu juga harus disiapkan skenario pengujian fungsional yang akan dilakukan agar nantinya

mempermudah proses pengujian. Berikut adalah tabel kebutuhan fungsional beserta skenario yang telah disiapkan pada tabel 5.3:

Tabel 5.3 Perancangan skenario pengujian fungsional

No	Kode	Fungsi	Skenario
1.	PF-01	<i>Sensor node</i> dapat mengambil data dari sensor suhu dan kelembapan	<ol style="list-style-type: none"> 1. <i>Sensor node</i> dalam keadaan diaktifkan 2. Terdapat sensor suhu dan kelembapan pada <i>sensor node</i> 3. Mengambil data dari sensor DHT 11 menggunakan fungsi dari <i>library</i> DHT sensor 4. Menampilkan data suhu dan kelembapan
2.	PF-02	<i>Sensor node</i> dapat menerima <i>broadcast</i> dari <i>gateway</i> dan mencocokkan id dari <i>gateway</i> dengan id yang dimiliki <i>sensor node</i>	<ol style="list-style-type: none"> 1. <i>Sensor node</i> sudah diaktifkan. 2. <i>Sensor node</i> dalam mode siap menerima paket (<i>listen</i>). 3. <i>Broadcast</i> diterima oleh <i>sensor node</i>. 4. Data <i>broadcast</i> dimasukkan ke dalam <i>struct</i>. 5. Mengambil data id dari <i>struct</i> broad. 6. Mencocokkan id dari <i>struct</i> dengan id yang dimiliki.
3.	PF-03	<i>Sensor node</i> dapat memilah data hasil <i>broadcast</i> sesuai format yang disepakati <i>sensor node</i> dan <i>gateway</i>	<ol style="list-style-type: none"> 1. <i>Sensor node</i> dalam keadaan aktif dan siap menerima <i>broadcast</i>. 2. <i>Broadcast</i> diterima oleh <i>sensor node</i>. 3. Data <i>broadcast</i> dimasukkan ke dalam <i>struct</i>. 4. Data hasil <i>broadcast</i> dibedakan menjadi variabel yang telah disepakati
4.	PF-04	<i>Sensor node</i> dapat mengetahui alamat <i>gateway</i> yang tepat berdasarkan pemilihan data yang diterima dari <i>broadcast</i>	<ol style="list-style-type: none"> 1. Terdapat dua <i>gateway</i> yang diaktifkan dan salah satunya adalah <i>gateway</i> yang salah serta berisi informasi <i>broadcast</i> yang berbeda. 2. <i>Broadcast</i> diterima oleh <i>sensor node</i>. 3. Data <i>broadcast</i> dimasukkan ke dalam <i>struct</i>.

			4. Alamat gateway yang tepat dapat diakses melalui variabel <i>gtwAddr</i> pada <i>struct</i>
5.	PF-05	<i>Sensor node</i> dapat mengirim data melalui modul LoRa ke <i>gateway</i> yang tepat berdasarkan informasi yang diperoleh dari hasil <i>broadcast</i>	<ol style="list-style-type: none"> 1. Terdapat dua <i>gateway</i> dengan alamat yang berbeda. 2. <i>Sensor node</i> dalam keadaan aktif dan sudah menerima dan mengolah <i>broadcast</i>. 3. <i>Sensor node</i> mengakses variabel yang berisi alamat <i>gateway</i> yang benar dari <i>struct</i> hasil <i>broadcast</i> 4. Data yang akan dikirim dimasukkan ke dalam <i>struct</i> baru 5. <i>Sensor node</i> melakukan <i>unicast</i> ke alamat <i>gateway</i> yang didapat dari hasil <i>broadcast</i>
6.	PF-06	<i>Gateway</i> dapat mengirimkan <i>broadcast</i> berisi informasi yang menandakan bahwa <i>gateway</i> tersebut adalah <i>gateway</i> yang tepat	<ol style="list-style-type: none"> 1. Perangkat <i>gateway</i> dihidupkan dan menjalankan kode program 2. Isi dari data <i>broadcast</i> dimasukkan ke dalam <i>struct</i> 3. <i>Gateway</i> melakukan <i>broadcast</i> kepada node dalam jangkauannya
7.	PF-07	<i>Gateway</i> dapat menerima data suhu dan kelembapan dari <i>sensor node</i>	<ol style="list-style-type: none"> 1. <i>Gateway</i> berada pada mode <i>listen</i> setelah melakukan <i>broadcast</i> 2. <i>Gateway</i> menerima <i>unicast</i> dari sensor node 3. <i>Unicast</i> yang berisi data suhu dan kelembapan disimpan ke dalam <i>struct</i>
8.	PF-08	<i>Gateway</i> dapat memilah data yang diterima dari <i>sensor node</i> berdasarkan format yang telah ditentukan	<ol style="list-style-type: none"> 1. Data dari <i>sensor node</i> disimpan ke dalam <i>struct</i> 2. Data dimasukkan ke dalam variabel dalam <i>struct</i> sesuai dengan format
9.	PF-09	<i>Gateway</i> menampilkan data	<ol style="list-style-type: none"> 1. <i>Gateway</i> sudah menerima <i>unicast</i> dari <i>sensor node</i>

		suhu dan kelembapan yang diterima sesuai data dari <i>sensor node</i>	2. Perangkat <i>gateway</i> memasukkan data suhu dan kelembapan ke dalam <i>struct</i> 3. Data suhu dan kelembapan ditampilkan pada sisi <i>gateway</i>
10.	PF-10	<i>Gateway</i> dapat menyimpan ID <i>sensor node</i> yang pernah mengirimkan data	1. Data dari <i>sensor node</i> disimpan ke dalam <i>struct</i> 2. Data dimasukkan ke dalam variabel dalam <i>struct</i> sesuai dengan format 3. ID dari <i>sensor node</i> yang diambil dari <i>struct</i> dimasukkan ke dalam <i>array</i>

5.1.5.2 Pengujian Performa

Pengujian performa disini akan berfokus pada tiga parameter yaitu *association time*, *broadcast loss* dan *unicast loss*. Ketiga parameter ini akan diuji terhadap jarak yang akan divariasikan menjadi jarak 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter. Pengujian ini akan diukur dengan 50 kali percobaan sistem *gateway discovery* pada masing-masing variasi jarak dan akan dirata-rata hasilnya. Berdasarkan penjelasan di atas maka pengujian performa akan dibagi sebagai berikut:

1. Pengujian *association time*

Pengujian ini adalah pengujian yang akan menguji waktu asosiasi sistem *gateway discovery* yang dihitung dari awal proses pengiriman *broadcast* sampai pengiriman data sensor pada waktu *unicast* dan berakhir sewaktu data sensor diterima oleh *gateway*. Waktu asosiasi ini bersifat relatif karena perhitungannya melalui waktu *broadcast* dilakukan, dikurangi waktu *unicast* diterima oleh *gateway* yang diperoleh dari fungsi *millis* yang terdapat pada Arduino. Fungsi *millis* sendiri adalah fungsi pada arduino yang akan mengembalikan nilai waktu dalam milidetik sejak papan arduino mulai menjalankan program. Rata-rata *association time* yang dihitung adalah *association time* yang diketahui dari *broadcast* yang berhasil mendapatkan jawaban (*unicast*) sebanyak 50 kali percobaan *gateway discovery*.

2. Pengujian *broadcast loss*

Pengujian ini akan menguji berapa persen *broadcast* yang gagal diterima oleh *sensor node*. Pada proses *broadcast* yang dilakukan *gateway* dapat terjadi gangguan dari beberapa hal seperti jarak antar node maupun gangguan lainnya yang dapat menyebabkan *broadcast* tidak diterima oleh *sensor node*.

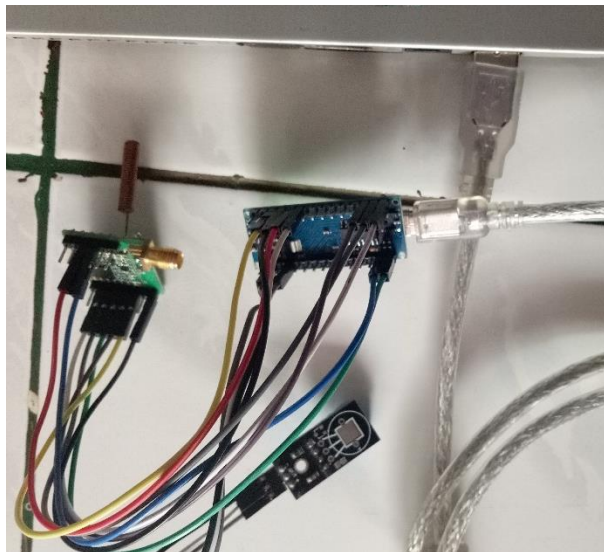
3. Pengujian *unicast loss*

Pengujian ini akan menguji berapa persen *unicast* yang gagal diterima oleh *gateway*. Pada proses *unicast* yang dilakukan oleh *sensor node* untuk mengirimkan data sensor suhu dan kelembapan dapat terjadi gangguan dari beberapa hal seperti jarak antar node maupun gangguan lainnya yang dapat menyebabkan *unicast* tidak diterima oleh *gateway*.

5.2 Implementasi Sistem

5.2.1 Implementasi *Sensor Node*

Perangkat *sensor node* terdiri dari tiga buah perangkat keras yang yang harus dihubungkan melalui pin-pin yang terdapat pada ketiga perangkat tersebut dengan kabel *jumper*. Perangkat keras tersebut adalah mikrokontroller Arduino Nano, modul LoRa HopeRF-RFM9x dan sensor DHT11 untuk mengambil data suhu dan kelembapan. *Pinout* pada ketiga perangkat keras tersebut telah dijabarkan pada bab perancangan dan diimplementasikan pada sisi *sensor node*. Setelah ketiga perangkat tersebut terhubung dengan benar, selanjutnya adalah memberikan daya pada *sensor node* melalui mikrokontroller Arduino Nano. Arduino Nano dapat disambungkan dengan adaptor ataupun laptop untuk mendapatkan sumber daya listrik seperti pada gambar 5.6. Jika terhubung dengan laptop, hal ini bertujuan agar *output* dari *sensor node* dapat dilihat melalui bantuan *software* Arduino IDE.



Gambar 5.6 Implementasi perangkat keras pada *sensor node*

Selanjutnya setelah semua perangkat penyusun *sensor node* siap adalah menambahkan *library* yang dibutuhkan untuk menjalankan kode program nantinya. Seperti yang disebutkan pada bab perancangan, terdapat dua *library* yang dibutuhkan pada *sensor node* yaitu RadioHead dan DHT11. Setelah menginstal dua *library* ini,

tahap selanjutnya yaitu pembuatan kode program dilakukan dan dapat dilakukan proses *uploading* kode program ke mikrokontroler Arduino Nano. Pembuatan kode program diawali dengan melakukan *include* beberapa file yang berisi fungsi yang akan digunakan pada kode program. *Include* file yang digunakan dari *library* RadioHead ada dua dan dari *library* DHT sensor ada satu yang dapat dilihat pada tabel 5.4.

Tabel 5.4 Tabel Kode program *include* file dari RadioHead dan DHT sensor

1	#include <RH_RF95.h>
2	#include <RHDatagram.h>
3	#include <DHT.h>

RH_RF95.h akan digunakan untuk mendukung beberapa fungsi perangkat modul *LoRa* yang berhubungan dengan komunikasi seperti penentuan frekuensi perangkat *LoRa* dan pengecekan pengaktifan modul *LoRa*. Sedangkan RHDatagram.h akan digunakan untuk mendukung fungsi *unicast* pada modul *LoRa* guna mendukung sistem *gateway discovery*. Selanjutnya DHT.h akan mengakomodasi fitur yang berhubungan dengan sensor DHT11. Setelah persiapan selesai, pembuatan kode utama program dapat dilakukan. Pseudocode program implementasi *sensor node* dapat dilihat pada tabel 5.5 di bawah ini.

Tabel 5.5 Pseudocode untuk kode program *sensor node*

1	Menambahkan file tambahan dari beberapa <i>library</i> yang digunakan
2	Mendefinisikan variabel
3	Membuat pengaturan untuk perangkat yang digunakan
4	Perulangan utama
5	Jika ada paket yang diterima maka,
6	Masukkan isi paket ke dalam <i>struct</i>
7	Jika format paket sesuai dengan ketentuan maka,
8	Jika nilai variabel <i>cons</i> = 13 maka,
9	Isi <i>struct</i> <i>dataSet</i> dengan data yang akan dikirim
10	Kirim data ke alamat <i>gateway</i>
11	Tampilkan nilai suhu dan kelembapan
12	Selain itu,
13	Cetak pemberitahuan bahwa <i>gateway</i> belum diketahui
14	Selain itu,
15	Cetak bahwa penerimaan data gagal

Tabel 5.5 berisi *pseudocode* dari kode program *sensor node* agar mudah untuk dipahami. Maksud baris pertama tabel 5.5 adalah melakukan penambahan berkas yang ada pada *library* yang nantinya akan digunakan untuk mempermudah melakukan pembuatan kode program. Pada baris kedua, variabel yang didefinisikan termasuk variabel yang digunakan untuk pengaturan perangkat seperti nilai alamat

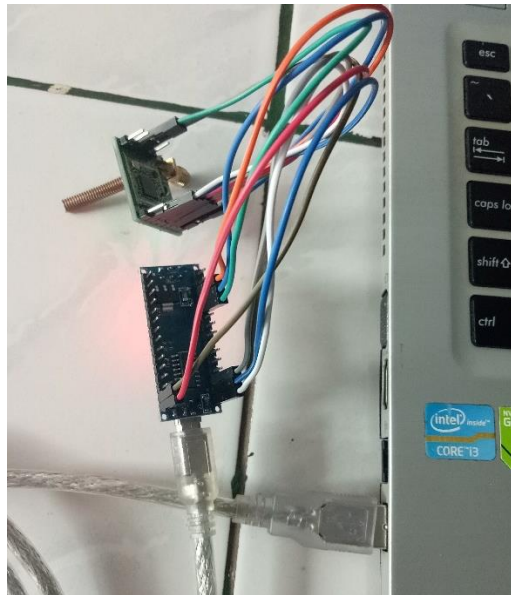
dan frekuensi dari modul *LoRa* yang digunakan. Selain itu juga terdapat definisi dari *struct* yang digunakan untuk menampung data yang diterima maupun untuk menampung data yang akan dikirim nantinya. Terdapat dua *struct* yang didefinisikan yaitu *struct* *broad* yang akan digunakan untuk menampung data dari hasil *broadcast* dan *struct* *dataSet* untuk menampung data yang akan dikirimkan ke *gateway*. Selanjutnya pada baris ketiga mempunyai maksud untuk melakukan pengaturan terlebih dahulu yang menyangkut persiapan perangkat-perangkat yang akan digunakan. Pada kode program *sensor node* terdapat pengaturan tiga perangkat yang terhubung yaitu Arduino Nano, sensor DHT 11 dan modul *LoRa HopeRF-rfm9x*. Pengaturan pada Arduino Nano meliputi pengaturan untuk mengaktifkan fungsi Serial dengan *baud rate* 9600. Sedangkan pada pengaturan pada sensor DHT 11 meliputi pemanggilan fungsi *begin()* untuk mengaktifkan sensor DHT 11. Pengaturan terakhir yaitu pada modul *LoRa* yang meliputi pengecekan untuk aktivasi modul *LoRa*, pengaturan frekuensi yang digunakan modul *LoRa* untuk berkomunikasi dan pengaturan *transmit power* pada modul *LoRa*.

Baris ke empat pada *pseudocode* tabel 5.5 mengindikasikan sudah memasuki perulangan utama program *sensor node*. Baris 5 akan ada kondisi dimana terjadi pengecekan ketersediaan paket, jika paket tersedia maka isi paket akan dimasukkan ke dalam *struct* *broad* yang sudah didefinisikan terlebih dahulu dan hal ini dapat dilihat pada baris ke 6. Selanjutnya masih dalam kondisi dimana data tersedia, terdapat pengecekan format paket dan pengecekan nilai dari variabel *cons*. Jika syarat yang diberikan sesuai, maka seperti yang dapat dilihat pada baris 9, akan dilakukan pengisian *struct* berisi data yang akan dikirimkan kepada *gateway*. Selanjutnya pada baris 10 *sensor node* akan mengirimkan data yang sudah dimasukkan ke dalam *struct* sebelumnya ke alamat *gateway* yang diketahui melalui data hasil *broadcast*. Setelah data dikirim selanjutnya nilai suhu dan kelembapan yang didapat melalui sensor DHT 11 akan ditampilkan pada keluaran serial yang dapat dilihat pada Arduino IDE. Baris 12-13 menandakan bahwa jika syarat yang diberikan tidak sesuai (dalam hal ini nilai *cons* tidak sesuai), maka hal ini berarti *gateway* belum diketahui. Baris 14-15 berfungsi sebagai penanda jika format data yang diterima tidak sesuai, maka akan ada pemberitahuan bahwa penerimaan data gagal. Kode program secara utuh pada sisi *sensor node* dapat dilihat pada lampiran A.

5.2.2 Implementasi Gateway

Implementasi pada sisi *gateway* dimulai dengan merakit perangkat keras yang menjadi penyusun *gateway*. Perangkat keras penyusun *gateway* terdiri dari dua perangkat keras yaitu mikrokontroler Arduino Nano dan modul *LoRa HopeRF-RFM9x*. mikrokontroler Arduino Nano akan digunakan sebagai otak dari *gateway* dimana kode program utama dari *gateway* akan disematkan. Sedangkan modul *LoRa* dengan seri dan tipe yang sama dengan yang ada pada sisi *sensor node* akan digunakan sebagai penghubung komunikasi antara *sensor node* dan *gateway*. Untuk

mendapatkan daya, kedua perangkat tersebut membutuhkan sumber energi listrik yang bisa didapatkan dari laptop dan dapat dilihat pada gambar 5.7.



Gambar 5.7 Implementasi perangkat keras pada *gateway*

Pada sisi *gateway*, penggunaan laptop ataupun personal komputer selain sebagai sumber daya utama perangkat *gateway* juga berfungsi untuk melihat data suhu dan kelembapan yang dikirimkan oleh *sensor node*. Data ini akan ditampilkan melalui *output* dari *port* dimana Arduino Nano terhubung melalui kabel USB. Tampilan ini didapatkan dengan bantuan perangkat lunak Arduino IDE yang sudah terdapat pada laptop.

Tahap selanjutnya adalah penambahan *library* RadioHead yang akan digunakan untuk menunjang fungsi utama pada modul *LoRa* pada perangkat lunak Arduino IDE. Setelah *library* siap, dapat dilanjutkan dengan implementasi kode program utama *gateway* yang nantinya akan dilakukan proses *uploading* pada mikrokontroler. Kode program pada sisi *gateway* diawali dengan dilakukan *include* berkas dari *library* RadioHead yang dapat dilihat pada tabel 5.6.

Tabel 5.6 Tabel Kode program *include* file dari RadioHead

1	#include <RH_RF95.h>
2	#include <RHDatagram.h>

Tabel di atas dengan kode program yang berfungsi untuk dapat mengambil fungsi yang dibutuhkan kode program utama untuk menjalankan beberapa fungsi modul *LoRa* untuk melengkapi sistem *gateway discovery*. Selanjutnya *pseudocode* untuk kode program utama yang mendukung sistem *gateway discovery* pada sisi *gateway* diimplementasikan dan dapat dilihat pada tabel 5.7.

Tabel 5.7 Pseudocode untuk kode program *gateway*

1	Menambahkan file tambahan dari beberapa <i>library</i> yang digunakan
2	Mendefinisikan variabel
3	Membuat pengaturan untuk perangkat yang digunakan
4	Perulangan utama
5	Jika waktu yang diberikan sesuai ketentuan maka,
6	Masukkan data ke dalam struct untuk dilakukan <i>broadcast</i>
7	Kirim <i>broadcast</i>
8	Tunggu sampai ada paket yang diterima
9	Inisialisasi variabel untuk menampung isi paket
10	Jika ada paket yang diterima sebelum waktu tunggu habis maka,
11	Inisialisasi struct yang akan digunakan untuk menampung data
12	Jika format paket sesuai ketentuan maka,
13	Jika nilai ack sesuai maka,
14	Lakukan pengecekan terhadap pengirim,
15	Jika belum tersimpan alamatnya maka simpan ke dalam array
16	Tampilkan alamat node yang tersedia
17	Lakukan penghapusan node yang tidak mengirim dalam-
18	jangka waktu tertentu
19	Tampilkan isi dari paket yang diterima dari node
20	Selain itu
21	Tampilkan bahwa penerimaan gagal
22	Selain itu
23	Tampilkan bahwa tidak ada jawaban <i>broadcast</i>

Tabel 5.7 berisi *pseudocode* dari kode program *gateway* yang bertujuan untuk memberi kemudahan pemahaman dari jalannya kode program. Baris pertama mempunyai arti bahwa pada awalnya program akan melakukan *include* atau penambahan berkas dari *library* yang digunakan seperti *library* RadioHead. Baris kedua setelah penambahan berkas yang dibutuhkan adalah mendefinisikan variabel yang digunakan dan juga tentunya *struct* yang digunakan untuk menampung berbagai data yang akan dipertukarkan. Selanjutnya pada baris ketiga dilanjutkan dengan melakukan pengaturan perangkat yang digunakan pada sisi *gateway* yaitu Arduino Nano dan modul *LoRa*. Pengaturan pada kedua perangkat ini disamakan dengan pengaturan pada sisi *sensor node*, hal ini bertujuan agar komunikasi antar kedua komponen sistem ini tidak terjadi kesalahan dan dapat berjalan lancar. Pengaturan tersebut khususnya berada pada perangkat modul *LoRa* yang harus disamakan pengaturan frekuensi dan juga *transmit power*. Jika frekuensi tidak sama antar kedua sisi komponen sistem, maka data antar kedua sisi komponen sistem tidak akan dapat tersampaikan karena modul *LoRa* tidak dapat berkomunikasi satu sama lain.

Baris ke 4 menandakan dimulainya tahap pada perulangan utama pada sisi *gateway*. Baris ke 5 terdapat kondisi yang mengecek waktu untuk dapat mengirimkan *broadcast*. Waktu pengiriman *broadcast* dapat dibatasi dengan menggunakan kondisi ini dengan diberikan batas sesuai keinginan pengguna. Jika sudah memenuhi waktu untuk melaksanakan *broadcast*, maka data yang akan dikirim melalui *broadcast* akan dimasukkan ke dalam *struct* seperti terlihat pada baris 6. Setelah itu selesai, maka tahap selanjutnya yang akan dilakukan adalah mengirimkan *broadcast* yang berisi data dari *struct* yang telah dipersiapkan.

Baris 8 menandakan kondisi dimana *gateway* sedang menunggu balasan dari *broadcast* yang telah dikirimkan. Selanjutnya ada inisialisasi variabel yang akan digunakan untuk menampung isi paket secara keseluruhan dari *buffer*. Baris 10 masuk ke dalam kondisi pengecekan ada tidaknya paket yang diterima selama waktu tunggu yang diberikan. Jika ada data yang diterima, maka akan dilanjutkan dengan tahap dari baris 11-21 dan jika tidak memenuhi kondisi tersebut, maka akan ada pemberitahuan pada keluaran serial bahwa tidak ada jawaban dari *broadcast* yang sudah dilakukan. Keluaran ini dapat dilihat dengan bantuan terminal yang ada pada Arduino IDE. Baris 11 bermaksud untuk melakukan inisialisasi *struct* yang digunakan untuk menampung data yang sudah dipersiapkan formatnya dalam bentuk *struct* dan merupakan data yang dikirimkan dari *sensor node*. Selanjutnya yaitu baris 12 terdapat kondisi yang jika terpenuhi yaitu format data yang dikirim sesuai, maka akan dilakukan tahap dari baris 13-19 dan jika tidak sesuai maka akan ditampilkan bahwa penerimaan gagal yang dapat dilihat pada baris 20-21.

Baris 13 *pseudocode* menunjukkan kondisi lain yaitu pengecekan nilai dari variabel *ack*. Jika nilai variabel tersebut sesuai dengan yang telah ditentukan, maka akan dijalankan tahap selanjutnya. Tahap selanjutnya ketika syarat tersebut terpenuhi, diawali dengan melakukan pengecekan terhadap pengirim dari data yang diterima. Jika pengirim data tersebut belum tersimpan atau belum dikenali, maka alamat pengirim data akan disimpan ke dalam *array* yang telah dibuat. *Array* ini memuat alamat dari perangkat yang pernah mengirim data dan masih mengirim data sampai jarak waktu tertentu. Selanjutnya adalah menampilkan alamat *node* yang tersedia. Yang dimaksud alamat *node* yang tersedia ini adalah *node* yang sudah pernah mengirimkan datanya ke *gateway* dan datanya berhasil diterima oleh *gateway* serta masih mengirim data dalam jarak waktu tertentu. Jika *node* yang alamatnya tersimpan dalam *array* tadi tidak lagi mengirimkan data pada jarak waktu yang telah ditentukan, maka alamatnya akan dihapus dari *array*. Dan langkah terakhir yang dilakukan adalah menampilkan isi dari paket yang telah diterima dari *sensor node*. Tentunya isi dari paket tersebut telah dipilah dan dimasukkan ke dalam variabel yang ditentukan. Tampilan dari isi paket tersebut dapat dilihat pada keluaran melalui terminal pada Arduino IDE. Kode program secara utuh pada sisi *gateway* dapat dilihat pada lampiran B.

BAB 6 HASIL DAN PEMBAHASAN

Bab hasil dan pembahasan ini mempunyai tujuan untuk mengetahui hasil dari implementasi yang telah dilakukan, cara untuk menentukan keberhasilan penelitian dan pembahasan tentang pengujian terhadap implementasi yang telah dilakukan. Bab ini dijabarkan menjadi empat bagian yaitu hasil implementasi, pengujian fungsional, pengujian performa dan pembahasan pengujian.

6.1 Hasil Implementasi

Implementasi yang telah dilakukan yaitu membentuk sistem *gateway discovery* pada *Wireless Sensor Network (WSN)* yang berbasis pada modul komunikasi *LoRa* berhasil diterapkan sesuai yang diharapkan pada tujuan penelitian. Hal ini dilatarbelakangi oleh tidak adanya error pada saat implementasi perangkat keras dan implementasi perangkat lunak sistem dan juga karena sistem dapat berjalan dengan baik. Sesuai dengan bab sebelumnya yaitu bab implementasi, implementasi sistem diterapkan sesuai dengan perancangan yang telah dibuat. Dalam implementasi tersebut juga mencakup implementasi perangkat keras dan implementasi perangkat lunak. Perangkat keras yang diimplementasikan terdiri dari Arduino Nano ATmega 328, modul *LoRa* HopeRF-RFM9x dengan antena helical bawaan dari modul ini dan sensor DHT 11 yang merupakan sensor suhu dan kelembapan pada sisi *sensor node*, sedangkan pada *gateway* hanya Arduino Nano ATmega 328 dan modul *LoRa* HopeRF-RFM9x.

Implementasi perangkat lunak yang terdapat pada sistem dilakukan dengan cara *uploading* kode program ke papan Arduino untuk *gateway discovery* pada kedua sisi sistem yaitu *sensor node* dan *gateway*. Proses *uploading* ini diperantarai oleh Arduino IDE sebagai aplikasi standar untuk melakukan proses *uploading* pada papan Arduino. Kode program yang telah diimplementasikan pada kedua sisi sistem dapat dilihat pada lampiran A dan B. Kedua kode program ini berhasil diimplementasikan pada sistem sesuai yang diharapkan sehingga bisa membentuk sistem *gateway discovery*. Untuk membuktikan keberhasilan dari implementasi ini maka dilakukanlah pengujian sistem. Untuk mengetahui keberhasilan memenuhi kriteria dasar dan tujuan sistem, maka dilakukan pengujian fungsional sistem seperti yang telah dijabarkan pada perancangan. Sedangkan untuk menguji performa sistem, dilakukan pengujian performa dengan tiga parameter yang telah disebutkan sebelumnya pada bab perancangan yang akan diuji terhadap jarak antar perangkat *sensor node* dan *gateway*.

6.2 Pengujian Fungsional

Pengujian fungsional adalah pengujian yang dilakukan secara *black-box* yang berarti berpatokan pada hasil akhir pengujian. Pada pengujian ini, hasil akhir

pengujian yang didapatkan harus memenuhi fungsional sistem yang telah dijabarkan sebelumnya untuk dapat menentukan keberhasilannya. Pengujian fungsional dilakukan dengan melakukan interaksi langsung di lapangan antara *sensor node* dan *gateway*.

Interaksi yang dilakukan antara *sensor node* dan *gateway* pada pengujian ini akan disesuaikan dengan interaksi sistem yang ada di lapangan sesuai dengan perancangan yang sudah dilakukan. Untuk memenuhi hal itu, pengujian fungsional ini telah diberikan beberapa skenario seperti dijabarkan pada bab sebelumnya. Pengujian fungsional ini terdiri dari 10 kebutuhan fungsional yang harus terpenuhi beserta skenario yang telah ditentukan.

6.2.1 Pengujian Kode PF-01

Pengujian dengan kode PF-01 dan kasus uji dimana *sensor node* dapat mengambil data dari sensor suhu dan kelembapan dapat dilihat pada tabel 6.1 berikut ini.

Tabel 6.1 Tabel pengujian kode PF-01

Kode	PF-01
Fungsi	<i>Sensor node</i> dapat mengambil data dari sensor suhu dan kelembapan
Tujuan pengujian	Menguji proses pengambilan data suhu dan kelembapan pada <i>sensor node</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. Sensor node dalam keadaan diaktifkan 2. Terdapat sensor suhu dan kelembapan pada <i>sensor node</i> 3. Mengambil data dari sensor DHT 11 menggunakan fungsi dari <i>library</i> DHT sensor 4. Menampilkan data suhu dan kelembapan
Hasil yang diharapkan	<i>Sensor node</i> berhasil mengambil data suhu dan kelembapan dari sensor DHT 11
Hasil pengujian	Berhasil mengambil data suhu dan kelembapan dari sensor DHT 11.

```

COM7
node address|5
available message
kurir masuk
Got: true gateway
1
Sent a reply
Current humidity = 78% temperature = 25 C

```

Gambar 6.1 Tampilan hasil pengujian PF-01 pada Arduino IDE

Gambar 6.1 menunjukkan tampilan pada Arduino IDE yang diambil dari sisi *sensor node*. Kotak merah pada gambar 6.1 menunjukkan keberhasilan pengujian PF-01 yang berarti *sensor node* sudah berhasil mengambil data dari sensor DHT 11 yang terpasang pada sisi *sensor node*.

6.2.2 Pengujian Kode PF-02

Pengujian dengan kode PF-02 dan kasus uji dimana *sensor node* dapat menerima *broadcast* dari *gateway* dan mencocokkan id dari *gateway* dengan id yang dimiliki *sensor node* dapat dilihat pada tabel 6.2 berikut ini.

Tabel 6.2 Tabel pengujian kode PF-02

Kode	PF-02
Fungsi	<i>Sensor node</i> dapat menerima <i>broadcast</i> dari <i>gateway</i> dan mencocokkan id dari <i>gateway</i> dengan id yang dimiliki <i>sensor node</i>
Tujuan pengujian	Menguji proses penerimaan <i>broadcast</i> dari <i>gateway</i> dan pencocokkan id dari <i>gateway</i> dengan id yang dimiliki <i>sensor node</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. <i>Sensor node</i> sudah diaktifkan. 2. <i>Sensor node</i> dalam mode siap menerima paket (<i>listen</i>). 3. <i>Broadcast</i> diterima oleh <i>sensor node</i>. 4. Data <i>broadcast</i> dimasukkan ke dalam <i>struct</i>. 5. Mengambil data id dari <i>struct</i> broad. 6. Mencocokkan id dari <i>struct</i> dengan id yang dimiliki.

Hasil yang diharapkan	<i>Sensor node</i> berhasil menerima <i>broadcast</i> dari <i>gateway</i> dan mencocokkan id dari <i>gateway</i> dengan id yang dimiliki <i>sensor node</i>
Hasil pengujian	Berhasil menerima <i>broadcast</i> dari <i>gateway</i> dan mencocokkan id dari <i>gateway</i> dengan id yang dimiliki <i>sensor node</i>

```

node address|5
available message
kurir masuk
Got: true gateway
1
Sent a reply
Current humidity = 78% temperature = 25 C

```

Gambar 6.2 Tampilan hasil pengujian PF-02 pada Arduino IDE

Keadaan awal *sensor node* adalah berada pada mode *listen* untuk bersiap menerima *broadcast* dari *gateway*. Jika ada *broadcast* masuk maka akan menampilkan bahwa pesan telah tersedia dan pada kotak merah gambar 6.2 adalah tanda bahwa id yang berada dalam data berisi id *gateway* yang benar.

6.2.3 Pengujian Kode PF-03

Pengujian dengan kode PF-03 dan kasus uji dimana *sensor node* dapat memilah data hasil *broadcast* sesuai format yang disepakati *sensor node* dan *gateway* dapat dilihat pada tabel 6.3 berikut ini.

Tabel 6.3 Tabel pengujian kode PF-03

Kode	PF-03
Fungsi	<i>Sensor node</i> dapat memilah data hasil <i>broadcast</i> sesuai format yang disepakati <i>sensor node</i> dan <i>gateway</i>
Tujuan pengujian	Menguji proses pengelolaan data hasil <i>broadcast</i> yang diterima <i>sensor node</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. <i>Sensor node</i> dalam keadaan aktif dan siap menerima <i>broadcast</i>. 2. <i>Broadcast</i> diterima oleh <i>sensor node</i>. 3. Data <i>broadcast</i> dimasukkan ke dalam <i>struct</i>.

	4. Data hasil broadcast dibedakan menjadi variabel yang telah disepakati
Hasil yang diharapkan	<i>Sensor node</i> berhasil memilah data hasil <i>broadcast</i> sesuai format yang disepakati <i>sensor node</i> dan <i>gateway</i>
Hasil pengujian	Berhasil memilah data hasil <i>broadcast</i> sesuai format yang disepakati <i>sensor node</i> dan <i>gateway</i>

```

COM7
node address|5
available message
kurir masuk
Got: true gateway
1
Sent a reply
Current humidity = 78% temperature = 25 C

```

Gambar 6.3 Tampilan hasil pengujian PF-03 pada Arduino IDE

Hasil pengujian PF-03 dapat dilihat secara tersirat pada kotak merah gambar 6.3 yang menunjukkan bahwa id yang diterima dari hasil *broadcast* adalah id yang sesuai dengan kesepakatan antara *gateway* dan *sensor node*. Keluaran yang ada dalam kotak merah ditampilkan jika id dari hasil *broadcast* sesuai dengan id yang dimiliki oleh *sensor node*. Id tersebut dapat diperoleh hanya dengan cara memilah data *broadcast* yang diterima dan formatnya sesuai kesepakatan. Hal tersebut menjadi tolak ukur keberhasilan pengujian PF-03.

6.2.4 Pengujian Kode PF-04

Pengujian dengan kode PF-04 dan kasus uji dimana *sensor node* dapat mengetahui alamat *gateway* yang tepat berdasarkan pemilihan data yang diterima dari *broadcast* dapat dilihat pada tabel 6.4 berikut ini.

Tabel 6.4 Tabel pengujian kode PF-04

Kode	PF-04
Fungsi	<i>Sensor node</i> dapat mengetahui alamat <i>gateway</i> yang tepat berdasarkan pemilihan data yang diterima dari <i>broadcast</i>
Tujuan pengujian	Menguji proses untuk mendapatkan alamat <i>gateway</i> yang tepat dari <i>broadcast</i>

Skenario pengujian	<ol style="list-style-type: none"> 1. Terdapat dua <i>gateway</i> yang diaktifkan dan salah satunya adalah <i>gateway</i> yang salah serta berisi informasi <i>broadcast</i> yang berbeda. 2. <i>Broadcast</i> diterima oleh <i>sensor node</i>. 3. Data <i>broadcast</i> dimasukkan ke dalam <i>struct</i>. 4. Alamat <i>gateway</i> yang tepat dapat diakses melalui variabel <i>gtwAddr</i> pada <i>struct</i>
Hasil yang diharapkan	<i>Sensor node</i> berhasil mengetahui alamat <i>gateway</i> yang tepat berdasarkan pemilihan data yang diterima dari <i>broadcast</i>
Hasil pengujian	Berhasil mengetahui alamat <i>gateway</i> yang tepat berdasarkan pemilihan data yang diterima dari <i>broadcast</i>

```

COM7
node address|5
available message
kurir masuk
Got: true gateway
1
Sent a reply
Current humidity = 78% temperature = 25 C

```

Gambar 6.4 Tampilan hasil pengujian PF-04 pada Arduino IDE

Hasil pengujian PF-04 dapat dilihat berdasarkan gambar 6.4 dimana keluaran pada kotak merah ditampilkan jika *gateway* tersebut adalah *gateway* yang benar. Angka “1” pada kotak merah adalah alamat *gateway* yang melakukan *broadcast* sebelumnya. Data alamat tersebut dapat diperoleh pada *struct* yang berisi beberapa variabel yang telah didapatkan melalui hasil *broadcast*. Hal tersebut menandakan keberhasilan pengujian ini karena alamat *gateway* berhasil diketahui.

6.2.5 Pengujian Kode PF-05

Pengujian dengan kode PF-05 dan kasus uji dimana *sensor node* dapat mengirim data melalui modul LoRa ke *gateway* yang tepat berdasarkan informasi yang diperoleh dari hasil *broadcast* dapat dilihat pada tabel 6.5 berikut ini.

Tabel 6.5 Tabel pengujian kode PF-05

Kode	PF-05
Fungsi	<i>Sensor node</i> dapat mengirim data melalui modul LoRa ke <i>gateway</i> yang tepat berdasarkan informasi yang diperoleh dari hasil <i>broadcast</i>
Tujuan pengujian	Menguji proses pengiriman data ke <i>gateway</i> yang tepat
Skenario pengujian	<ol style="list-style-type: none"> 1. Terdapat dua <i>gateway</i> dengan alamat yang berbeda. 2. <i>Sensor node</i> dalam keadaan aktif dan sudah menerima dan mengolah <i>broadcast</i>. 3. <i>Sensor node</i> mengakses variabel yang berisi alamat <i>gateway</i> yang benar dari <i>struct</i> hasil <i>broadcast</i> 4. Data yang akan dikirim dimasukkan ke dalam <i>struct</i> baru 5. <i>Sensor node</i> melakukan <i>unicast</i> ke alamat <i>gateway</i> yang didapat dari hasil <i>broadcast</i>
Hasil yang diharapkan	<i>Sensor node</i> berhasil mengirim data melalui modul LoRa ke <i>gateway</i> yang tepat berdasarkan informasi yang diperoleh dari hasil <i>broadcast</i>
Hasil pengujian	Berhasil mengirim data melalui modul LoRa ke <i>gateway</i> yang tepat berdasarkan informasi yang diperoleh dari hasil <i>broadcast</i>

```
gateway 1]
Sending broadcast to Node
Waiting for reply...
Ack node OK
Available node : 5
Mendapat data dari node : 5
ID = 5 : Current humidity = 78% temperature = 25 C
```

Gambar 6.5 Tampilan hasil pengujian PF-05 pada Arduino IDE

Hasil pengujian PF-04 dapat dilihat pada gambar 6.5 yang menunjukkan keluaran pada sisi *gateway* yang benar. Kalimat yang ada dalam kotak merah menandakan bahwa data dari *sensor node* sudah berhasil diterima oleh *gateway*. Pengiriman data dari *sensor node* menuju *gateway* yang tepat dengan perantara modul LoRa dapat dilakukan dengan melakukan *unicast* yang berarti hanya mengirimkan ke satu alamat.

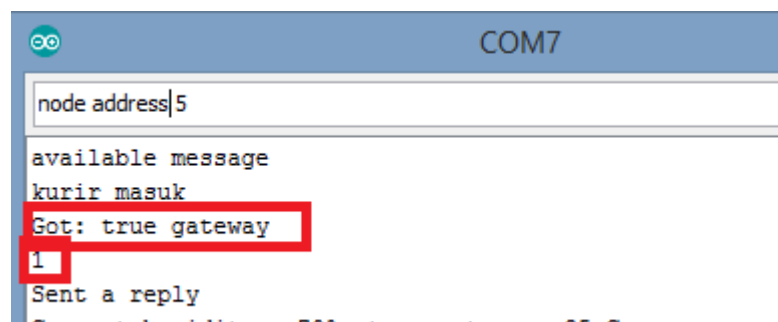
Alamat ini diperoleh melalui *broadcast* sebelumnya yang dilakukan dan dimasukkan ke dalam variabel.

6.2.6 Pengujian Kode PF-06

Pengujian dengan kode PF-06 dan kasus uji dimana *gateway* dapat mengirimkan *broadcast* berisi informasi yang menandakan bahwa *gateway* tersebut adalah *gateway* yang tepat dapat dilihat pada tabel 6.6 berikut ini.

Tabel 6.6 Tabel pengujian kode PF-06

Kode	PF-06
Fungsi	<i>Gateway</i> dapat mengirimkan <i>broadcast</i> berisi informasi yang menandakan bahwa <i>gateway</i> tersebut adalah <i>gateway</i> yang tepat
Tujuan pengujian	Menguji proses pengiriman <i>broadcast</i> yang membawa informasi yang diperlukan untuk <i>gateway discovery</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. Perangkat <i>gateway</i> dihidupkan dan menjalankan kode program 2. Isi dari data <i>broadcast</i> dimasukkan ke dalam <i>struct</i> 3. <i>Gateway</i> melakukan <i>broadcast</i> kepada node dalam jangkauannya
Hasil yang diharapkan	<i>Gateway</i> berhasil mengirimkan <i>broadcast</i> berisi informasi yang menandakan bahwa <i>gateway</i> tersebut adalah <i>gateway</i> yang tepat
Hasil pengujian	Berhasil mengirimkan <i>broadcast</i> berisi informasi yang menandakan bahwa <i>gateway</i> tersebut adalah <i>gateway</i> yang tepat



Gambar 6.6 Tampilan hasil pengujian PF-06 pada Arduino IDE

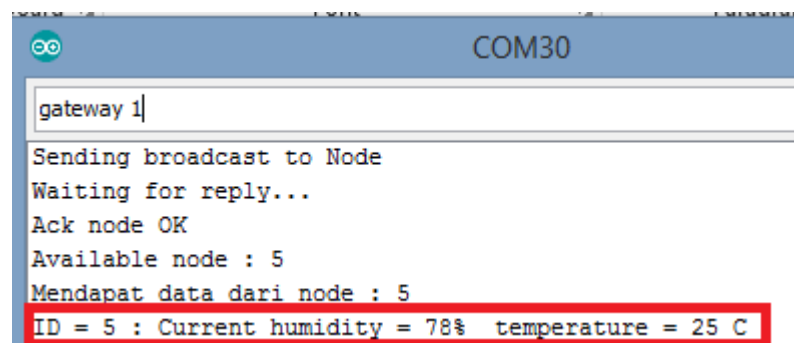
Gambar 6.6 menunjukkan hasil keberhasilan dalam memenuhi pengujian PF-06. Keberhasilan ini ditandai dengan kotak merah yang ada pada gambar 6.6 yang didapatkan dari keluaran pada sisi *sensor node*. Kotak merah tersebut menandakan bahwa *broadcast* dari *gateway* berhasil dilakukan dan sesuai dengan format yang telah disepakati. Keluaran pada kotak merah yang pertama ditampilkan jika informasi id yang dikirimkan *gateway* benar. Hal inilah alasan yang menandakan keberhasilan pengujian PF-06.

6.2.7 Pengujian Kode PF-07

Pengujian dengan kode PF-07 dan kasus uji dimana *gateway* dapat menerima data suhu dan kelembapan dari *sensor node* dapat dilihat pada tabel 6.7 berikut ini.

Tabel 6.7 Tabel pengujian kode PF-07

Kode	PF-07
Fungsi	<i>Gateway</i> dapat menerima data suhu dan kelembapan dari <i>sensor node</i>
Tujuan pengujian	Menguji proses penerimaan data suhu dan kelembapan dari <i>sensor node</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. <i>Gateway</i> berada pada mode <i>listen</i> setelah melakukan <i>broadcast</i> 2. <i>Gateway</i> menerima <i>unicast</i> dari <i>sensor node</i> 3. <i>Unicast</i> yang berisi data suhu dan kelembapan disimpan ke dalam <i>struct</i>
Hasil yang diharapkan	<i>Gateway</i> berhasil menerima data suhu dan kelembapan dari <i>sensor node</i>
Hasil pengujian	Berhasil menerima data suhu dan kelembapan dari <i>sensor node</i>



```

COM30
gateway 1
Sending broadcast to Node
Waiting for reply...
Ack node OK
Available node : 5
Mendapat data dari node : 5
ID = 5 : Current humidity = 78% temperature = 25 C

```

Gambar 6.7 Tampilan hasil pengujian PF-07 pada Arduino IDE

Kotak merah pada gambar 6.7 menandakan keberhasilan pengujian PF-07 dengan kasus uji *gateway* dapat menerima data suhu dan kelembapan dari *sensor node*. Gambar 6.7 diperoleh dari keluaran yang dilihat dari keluaran pada sisi *gateway*. Keluaran yang berada dalam kotak merah yang menampilkan data suhu dan kelembapan didapatkan melalui data yang diterima dari *sensor node*. Hal inilah yang menjadi indikator keberhasilan pengujian PF-07.

6.2.8 Pengujian Kode PF-08

Pengujian dengan kode PF-08 dan kasus uji dimana *gateway* dapat memilah data yang diterima dari *sensor node* berdasarkan format yang telah ditentukan dapat dilihat pada tabel 6.8 berikut ini.

Tabel 6.8 Tabel pengujian kode PF-08

Kode	PF-08
Fungsi	<i>Gateway</i> dapat memilah data yang diterima dari <i>sensor node</i> berdasarkan format yang telah ditentukan
Tujuan pengujian	Menguji proses pengelolaan data yang diterima dari <i>sensor node</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. Data dari <i>sensor node</i> disimpan ke dalam <i>struct</i> 2. Data dimasukkan ke dalam variabel dalam <i>struct</i> sesuai dengan format
Hasil yang diharapkan	<i>Gateway</i> berhasil memilah data yang diterima dari <i>sensor node</i> berdasarkan format yang telah ditentukan
Hasil pengujian	Berhasil memilah data yang diterima dari <i>sensor node</i> berdasarkan format yang telah ditentukan

```

COM30
gateway 1|
Sending broadcast to Node
Waiting for reply...
Ack node OK
Available node : 5
Mendapat data dari node : 5
ID = 5 : Current humidity = 78% temperature = 25 C

```

Gambar 6.8 Tampilan hasil pengujian PF-08 pada Arduino IDE

Gambar 6.8 menunjukkan keluaran yang didapat pada Arduino IDE di sisi *gateway*. Keluaran yang berada dalam kotak merah menandakan bahwa *gateway* sudah dapat memilah data yang diterima dari *sensor node* sesuai format yang ditentukan dan menjadi tolak ukur keberhasilan pengujian PF-08. Hal ini dapat disimpulkan sedemikian rupa karena data yang ditampilkan sesuai dan hanya dapat diperoleh setelah data yang dikirim oleh *sensor node* berhasil dipilah dan dimasukkan ke dalam variabel yang sudah disediakan.

6.2.9 Pengujian Kode PF-09

Pengujian dengan kode PF-09 dan kasus uji dimana *gateway* menampilkan data suhu dan kelembapan yang diterima sesuai data dari *sensor node* dapat dilihat pada tabel 6.9 berikut ini.

Tabel 6.9 Tabel pengujian kode PF-09

Kode	PF-09
Fungsi	<i>Gateway</i> menampilkan data suhu dan kelembapan yang diterima dari <i>sensor node</i>
Tujuan pengujian	Menguji proses untuk menampilkan data suhu dan kelembapan pada <i>gateway</i> sesuai data dari <i>sensor node</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. <i>Gateway</i> sudah menerima <i>unicast</i> dari <i>sensor node</i> 2. Perangkat <i>gateway</i> memasukkan data suhu dan kelembapan ke dalam <i>struct</i> 3. Data suhu dan kelembapan ditampilkan pada sisi <i>gateway</i>
Hasil yang diharapkan	<i>Gateway</i> berhasil menampilkan data suhu dan kelembapan yang diterima sesuai data dari <i>sensor node</i>
Hasil pengujian	Berhasil menampilkan data suhu dan kelembapan yang diterima sesuai data dari <i>sensor node</i>

```

COM30
gateway 1
Sending broadcast to Node
Waiting for reply...
Ack node OK
Available node : 5
Mendapat data dari node : 5
ID = 5 : Current humidity = 78% temperature = 25 C

```

Gambar 6.9 Tampilan hasil pengujian PF-09 pada Arduino IDE

Data suhu dan kelembapan yang dikirimkan oleh *sensor node* dapat ditampilkan pada sisi *gateway* dengan melihat pada keluaran Arduino IDE. Data yang ditampilkan oleh *gateway* telah sesuai dengan data yang dikirimkan *sensor node*. Dengan adanya hal tersebut, maka pengujian PF-09 dapat dikatakan berhasil.

6.2.10 Pengujian Kode PF-10

Pengujian dengan kode PF-10 dan kasus uji dimana *gateway* dapat menyimpan ID *sensor node* yang pernah mengirimkan data dapat dilihat pada tabel 6.10 berikut ini.

Tabel 6.10 Tabel pengujian kode PF-10

Kode	PF-10
Fungsi	<i>Gateway</i> dapat menyimpan ID <i>sensor node</i> yang pernah mengirimkan data
Tujuan pengujian	Menguji proses untuk menampilkan data suhu dan kelembapan pada <i>gateway</i>
Skenario pengujian	<ol style="list-style-type: none"> 1. Data dari <i>sensor node</i> disimpan ke dalam <i>struct</i> 2. Data dimasukkan ke dalam variabel dalam <i>struct</i> sesuai dengan format 3. ID dari <i>sensor node</i> yang diambil dari <i>struct</i> dimasukkan ke dalam <i>array</i>
Hasil yang diharapkan	<i>Gateway</i> berhasil menyimpan ID <i>sensor node</i> yang pernah mengirimkan data
Hasil pengujian	Berhasil menyimpan ID <i>sensor node</i> yang pernah mengirimkan data

```
COM30
gateway 1]
Sending broadcast to Node
Waiting for reply...
Ack node OK
Available node : 5
Mendapat data dari node : 5
ID = 5 : Current humidity = 78% temperature = 25 C
```

Gambar 6.10 Tampilan hasil pengujian PF-10 pada Arduino IDE

Hasil pengujian PF-10 dapat dilihat pada gambar 6.10 yang menunjukkan alamat node dengan keadaan pernah mengirimkan data suhu dan kelembapan kepada *gateway*. Alamat *sensor node* ini akan dihapus jika dalam waktu interval tertentu tidak mengirimkan data lagi kepada *gateway*.

6.3 Pengujian Performa

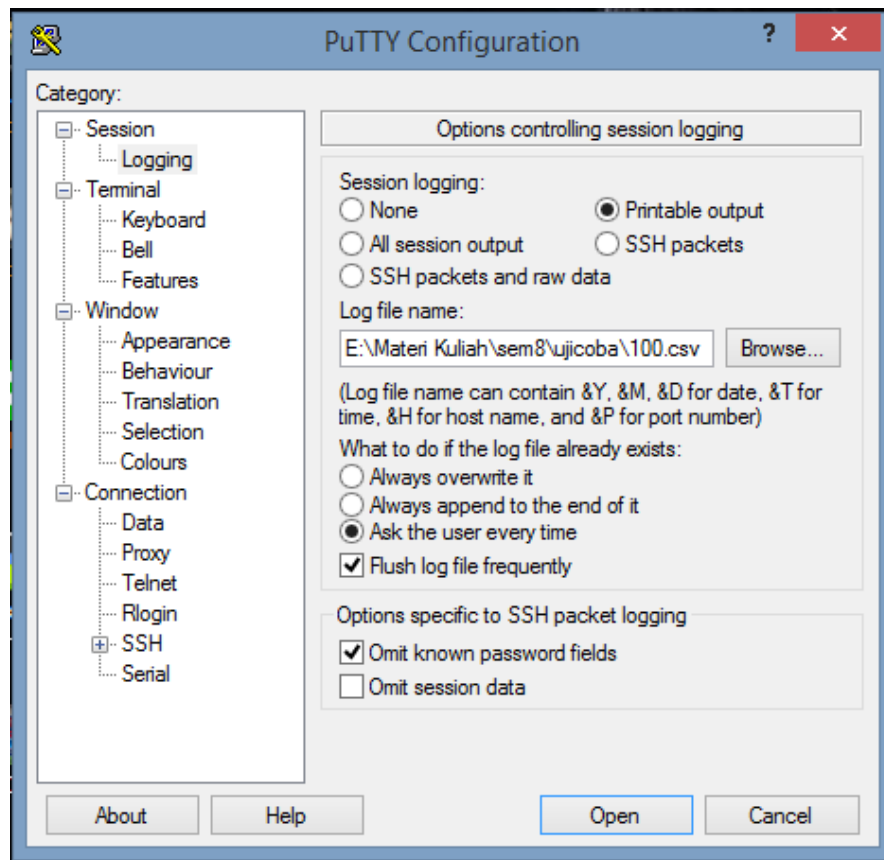
Pengujian performa ini dilakukan untuk menguji keandalan sistem yang telah dibuat. Pengujian performa ini dilakukan terhadap jarak variasi antara *sensor node* dan *gateway*. Parameter pengujian yang akan diuji terhadap jarak ada tiga parameter. Parameter tersebut adalah *association time*, *broadcast loss* dan *unicast loss*. Ketiga parameter tersebut telah dijelaskan pada kajian pustaka dan perancangan pengujian performa. Pengujian ini dilakukan di desa Kidal dengan pertimbangan lokasi yang masih terdapat lahan kosong yang luas, sehingga gangguan untuk melakukan sistem *gateway discovery* berbasis perangkat *LoRa* dapat diminimalisir. Pengujian ini dilakukan dengan menggunakan *tool Putty* guna menunjang proses pengujian performa. *Putty* akan dimanfaatkan fitur untuk menangkap keluaran pada suatu port yang telah terhubung dengan papan Arduino yang akan dimasukkan ke dalam *file* bertipe *csv*. *File* dalam bentuk inilah sumber dari hasil pengujian performa didapatkan dan selanjutnya dilakukan analisis. Skenario pengujian performa telah dijelaskan pada bab perancangan, sehingga disini akan langsung ditunjukkan hasil yang diperoleh pengujian performa yang sudah dilakukan. Kebutuhan, batasan serta hasil yang diharapkan dalam pengujian performa dapat dilihat pada tabel 6.11.

Tabel 6.11 Skenario Pengujian Performa

Kode	Parameter	Keterangan
PP_001	<i>Association time</i>	Menggunakan <i>tool Putty</i> untuk melakukan pengumpulan data dan dengan variasi jarak sebesar 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter

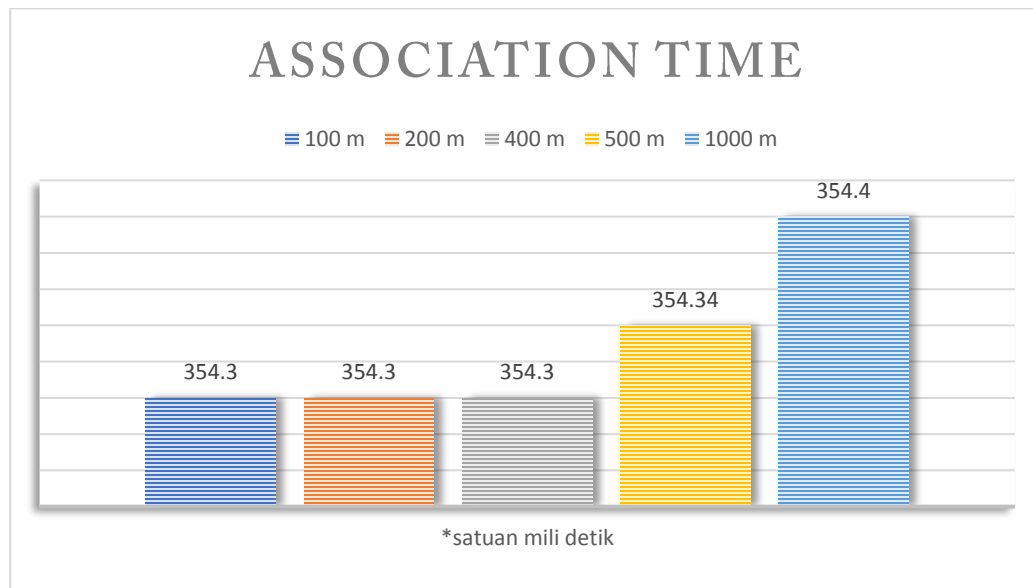
PP_002	<i>Broadcast loss</i>	Menggunakan <i>tool Putty</i> untuk melakukan pengumpulan data dan dengan variasi jarak sebesar 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter
PP_003	<i>Unicast loss</i>	Menggunakan <i>tool Putty</i> untuk melakukan pengumpulan data dan dengan variasi jarak sebesar 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter

Tahap pengujian performa diawali dengan mempersiapkan perangkat yang digunakan yaitu perangkat *sensor node* dan *gateway*. Sumber daya untuk *gateway* didapatkan dari laptop, laptop dibutuhkan pada *gateway* guna mengumpulkan data parameter yang akan diuji terhadap jarak. Seperti dapat dilihat pada tabel 6.11, jarak akan divariasikan sebesar 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter. Pada setiap jarak akan dilakukan proses *gateway discovery* sebanyak 50 kali untuk mendapatkan hasil yang akurat. Selanjutnya, konfigurasi untuk menghasilkan data yang dibutuhkan dari pengujian performa melalui *Putty* tertera pada gambar 6.11.



Gambar 6.11 Konfigurasi output file pada Putty

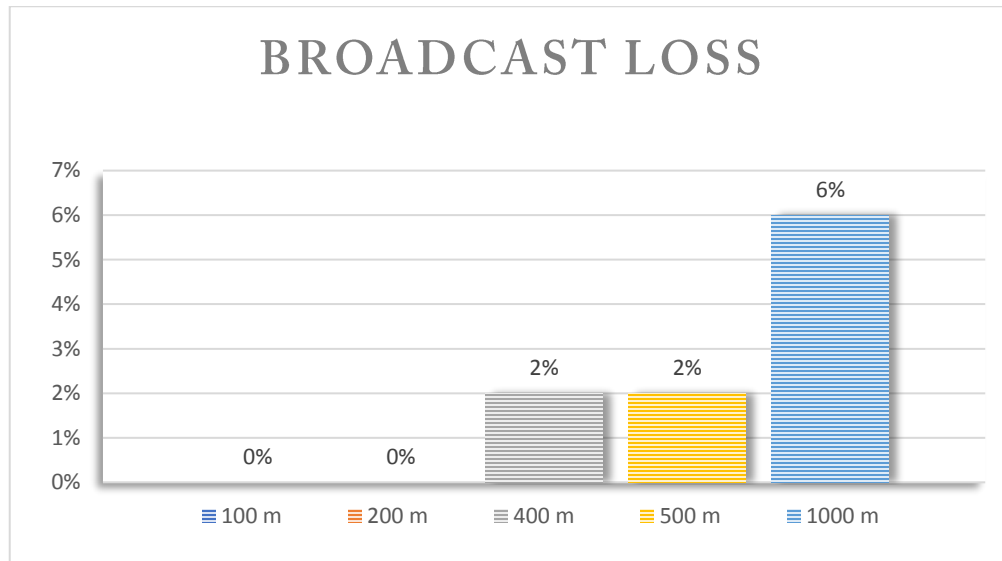
Pengujian pertama yang dilakukan pertama kali adalah *association time* (PP_001). Pengujian ini dilakukan dengan penerapan sistem *gateway discovery* yang diawali dengan pengiriman *broadcast*. Hasil *association time* yang akan dihitung rata-ratanya adalah hasil *association time* yang berhasil membentuk sistem *gateway discovery* atau yang berarti saat *broadcast* berhasil diterima *sensor node* dan *unicast* juga berhasil diterima oleh *gateway*. Pengujian ini dilakukan dalam variasi jarak yang telah ditentukan, yaitu 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter. Pada 50 kali percobaan tersebut didapatkan data hasil pengujian performa dengan rata-rata hasil *association time* pada setiap jarak dapat dilihat pada gambar 6.12.



Gambar 6.12 Diagram *association time* terhadap jarak

Gambar 6.12 merupakan diagram yang menunjukkan hasil pengujian performa terhadap jarak dengan parameter *association time*. Diagram tersebut menunjukkan perbedaan variasi waktu untuk penerapan sistem *gateway discovery* dalam satuan mili detik. Urutan nilai jarak pada gambar 6.12 dari kiri ke kanan yaitu jarak 100 meter, 200 meter, 400 meter, 500 meter dan 1000 meter. Menurut diagram pada gambar, nilai rata-rata *association time* bernilai konsisten pada jarak 100 meter, 200 meter dan 400 meter yaitu dengan waktu 354.3 mili detik. Selanjutnya pada jarak 500 meter, nilai rata-rata *association time* mengalami peningkatan sebesar 0.04 mili detik sehingga nilainya pada diagram menjadi 354.34 mili detik. Pada pengujian jarak yang terakhir yaitu jarak 1000 meter, nilai rata-rata *association time* mengalami kenaikan yang signifikan dari nilai rata-rata *association time* pada jarak sebelumnya. Pada jarak 1000 meter ini, nilai rata-rata *association time* naik 0.06 mili detik sehingga menghasilkan nilai rata-rata *association time* sebesar 354.4 mili detik. Hal ini tentunya dapat menghasilkan kesimpulan yang nantinya akan dibahas pada subbab pembahasan di bab ini.

Pengujian performa berikutnya adalah pengujian *broadcast loss* (PP_002) yang diuji terhadap variasi jarak yang telah disebutkan sebelumnya. *Broadcast* akan dikirimkan sebanyak 50 kali pada sisi *gateway* dan nantinya *broadcast* itu nantinya akan dihitung persentase *broadcast* yang hilang atau *loss*. Data ini diperoleh dengan menggunakan bantuan *tools Putty* untuk mengambil data dari keluaran *port* yang terhubung dengan papan Arduino Nano. Hasil pengujian performa PP_002 dapat dilihat pada diagram berikut:

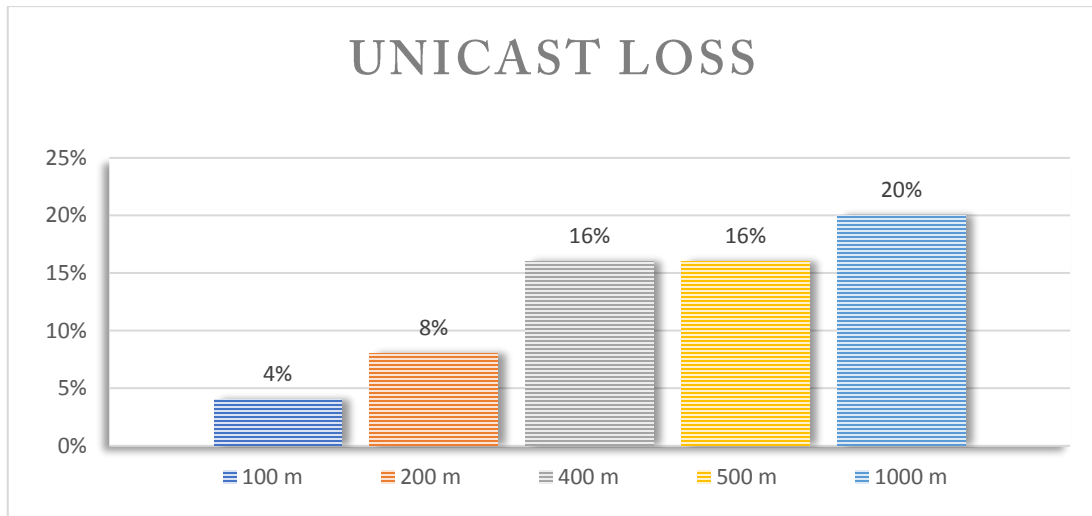


Gambar 6.13 Diagram *broadcast loss* terhadap jarak

Berdasarkan hasil pengujian performa pada gambar 6.13, dapat dilihat bahwa *broadcast loss* paling besar terjadi pada jarak 1000 meter yang merupakan jarak maksimum pada pengujian performa ini, sedangkan *broadcast loss* pada jarak yang lain hampir berimbang dengan selisih hanya sekitar 2%. Dengan pengujian performa untuk *broadcast loss* yang dilakukan dengan mengirimkan *broadcast* sebanyak 50, nilai 2% dapat berarti bahwa *broadcast* yang hilang atau *loss* hanya 1 *broadcast*. Hal ini dapat disimpulkan bahwa pada jarak 100 meter hingga 500 meter *broadcast* masih mempunyai persentase kesempatan untuk diterima oleh *sensor node* mendekati 100%. Sedangkan pada jarak 1000 meter, persentase *broadcast loss* mengalami kenaikan yang signifikan yaitu menyentuh angka 6% untuk persentasenya.

Seperti dapat dilihat pada diagram pada gambar 6.13 jarak antara *sensor node* dan *gateway* mempunyai dampak signifikan terhadap perhitungan *broadcast loss* meskipun diagramnya turun naik. Pada jarak 100-500 meter, jumlah *broadcast loss* masih minim dengan *range* 0-2% dan mulai menanjak naik secara signifikan pada jarak 1000 meter antar komponen. Dari pengujian performa yang dilakukan, didapatkan hasil bahwa jarak antara *sensor node* dan *gateway* berpengaruh pada sistem yang telah dibuat. Untuk penjelasan menyeluruh dapat dilihat pada subbab pembahasan.

Pengujian performa yang terakhir adalah pengujian *unicast loss* (PP_003) yang akan diuji terhadap jarak antara *sensor node* dan *gateway*. *Unicast* akan dilakukan oleh *sensor node* yang menanggapi *broadcast* yang dikirimkan oleh *gateway*. *Unicast* yang tidak diterima oleh *gateway* dapat dikategorikan sebagai *unicast loss*. Data hasil pengujian PP_003 dapat dilihat pada gambar 6.14 berikut.



Gambar 6.14 Diagram *unicast loss* terhadap jarak

Melalui diagram pada gambar 6.14 dapat dilihat bahwa *unicast loss* terjadi paling sedikit pada jarak yang paling dekat yaitu 100 meter dan menjadi jarak paling optimal dalam pengujian performa, tetapi dapat dilihat juga pada jarak selanjutnya yaitu pada jarak 200 meter, 400 meter, 500 meter dan 1000 meter perlahan jumlah *unicast loss* meningkat. Pada jarak 200 meter terjadi peningkatan sebesar 4% dari jarak sebelumnya dan naik kembali sebesar 8% pada jarak 400 meter sehingga nilai persentase *unicast loss* menjadi 16%. Selanjutnya jumlah *unicast loss* tidak berkurang maupun bertambah dari jumlah sebelumnya pada jarak 500 meter. *Unicast loss* pada jarak 1000 meter mengalami kenaikan hingga menyentuh angka 20%. Hal ini dapat berarti pada jarak 1000 meter sistem *gateway discovery* pada waktu *unicast* masih dapat bekerja tetapi dengan jumlah persentase *unicast loss* yang cukup buruk. Dari gambar 6.14 dapat disimpulkan bahwa semakin jauh jarak *sensor node* dan *gateway*, maka jumlah *unicast loss* akan cenderung naik dan berakibat semakin buruk pada sistem. Untuk analisa lebih lanjut tentang hasil dari pengujian performa akan dijabarkan pada pembahasan pengujian.

6.4 Pembahasan Pengujian

Pengujian fungsional yang telah dilakukan sebelumnya untuk mengetahui keberhasilan sistem dalam hal fungsionalitasnya menghasilkan kesimpulan bahwa sistem sudah bekerja dengan baik sesuai dengan tujuan penelitian. Dari sepuluh kebutuhan fungsional yang diberikan, kesepuluh kebutuhan fungsional tersebut menunjukkan tanda keberhasilan yang berarti terpenuhinya tujuan dibuatnya sistem *gateway discovery* yang berbasis pada modul komunikasi *LoRa*. Sesuai yang dapat dilihat pada subbab pengujian fungsional, bukti untuk keberhasilan fungsionalitas sistem diberikan dalam bentuk gambar yang diperoleh melalui tampilan keluaran sistem dengan bantuan Arduino IDE. Penjelasan singkat mengenai bukti tersebut telah disebutkan pada masing-masing kode pengujian fungsional. Pengujian fungsional yang dilakukan dengan lebih dari satu node menghasilkan pengiriman data sensor ke *gateway* secara bergantian. Jadi data sensor yang diterima oleh *gateway* bergantian tergantung dari data dari sensor mana yang diterima terlebih dahulu.

Selanjutnya adalah pengujian performa dari sistem yang telah dibuat. Pengujian performa sistem diperlukan guna menguji performa sistem setelah berhasil diimplementasikan dan memenuhi kebutuhan fungsional sistem. Pengujian performa yang telah dilakukan sebelumnya dilakukan pengujian terhadap jarak yang menjadi sorotan penulis. Hal ini dikarenakan komunikasi perangkat komunikasi *wireless* bergantung terhadap jarak. Rangkuman hasil pengujian performa dapat dilihat pada tabel 6.12.

Tabel 6.12 Hasil pengujian performa

Parameter	Jarak				
	100 meter	200 meter	400 meter	500 meter	1000 meter
Association time	354.3 ms	354.3 ms	354.3 ms	354.34 ms	354.4 ms
Broadcast loss	0%	0%	2%	2%	6%
Unicast loss	4%	8%	16%	16%	20%

Hasil dari pengujian performa yang telah dilakukan sebelumnya dengan parameter *association time*, *broadcast loss* dan *unicast loss* yang diuji terhadap jarak mendapatkan hasil bahwa jarak berpengaruh terhadap performa sistem. Hal terlihat sekali pada parameter *association time* dan *unicast loss* dimana nilai uji keduanya mengalami kenaikan seiring dengan bertambahnya jarak 100 meter hingga 1000 meter meskipun ada beberapa bagian yang tetap seperti nilai *association time* pada jarak 100 meter hingga 400 meter yang konsisten sebesar 354.3 mili detik dan *unicast loss* pada jarak 400 meter dan 500 meter yang sama besar persentasenya sebesar 16%. Tetapi pada akhirnya pada jarak maksimal yaitu 1000 meter, ketiga parameter

mengalami kenaikan yang signifikan daripada jarak uji sebelumnya. Dengan diperolehnya hasil pengujian yang telah dilakukan sebelumnya, dapat disimpulkan bahwa sistem *gateway discovery* telah bekerja dengan baik dan performanya juga dipengaruhi oleh jarak antara *sensor node* dan *gateway*. Semakin dekat jarak antara *sensor node* dan *gateway*, maka akan semakin baik performa dari sistem *gateway discovery* yang dibuat dan berlaku juga sebaliknya, jika jarak semakin jauh maka performa sistem *gateway discovery* yang dibuat akan semakin buruk.

BAB 7 PENUTUP

Bab terakhir ini yaitu bab penutup berisi tentang kesimpulan dari hasil penelitian yang telah dilakukan dan saran terhadap penelitian di masa depan. Kesimpulan dan saran pada bab ini diperoleh dari hasil proses perancangan, implementasi dan pengujian.

7.1 Kesimpulan

Berdasarkan hasil dari proses perancangan, implementasi, dan pengujian yang telah dilakukan sebelumnya, penulis mendapatkan beberapa kesimpulan sebagai berikut:

1. Sistem *gateway discovery* pada *Wireless Sensor Network (WSN)* menggunakan perangkat mikrokontroller Arduino Nano, sensor DHT 11 dan modul komunikasi *LoRa* HopeRF-RFM9x pada sisi *sensor node* dan mikrokontroller Arduino Nano dan modul komunikasi *LoRa* HopeRF-RFM9x pada sisi *gateway*. Perangkat modul komunikasi *LoRa* HopeRF-RFM9x menjembatani komunikasi nirkabel pada sistem *WSN* yang telah dibuat dengan dibekali antena helical yang merupakan kelengkapan bawaan dari modul komunikasi *LoRa* HopeRF-RFM9x. Sistem *gateway discovery* pada *WSN* yang berbasis modul komunikasi *LoRa* diimplementasikan dengan terlebih dahulu melakukan *broadcast* dari *gateway* yang berisi informasi alamat *gateway* dan ID yang menjadi ID konfirmasi untuk *gateway* yang benar. *Broadcast* yang diterima *sensor node* diolah dan selanjutnya data suhu dan kelembapan akan dikirimkan secara *unicast* kepada alamat *gateway* yang benar.
2. *Sensor node* dapat membedakan *gateway* miliknya atau bukan dengan bantuan ID yang disematkan pada data yang dikirimkan oleh *gateway*. Hal yang sama juga berlaku untuk *gateway*, *gateway* dapat mengenali *sensor node* miliknya dengan bantuan kode konfirmasi yang disematkan saat pengiriman data ke *gateway*.
3. Data yang diterima pada saat *unicast* maupun *broadcast* diolah dengan menggunakan *struct*. Data yang diterima akan dibuatkan format *struct* yang sesuai dengan format data yang dikirimkan pada sisi penerima. Dalam *struct* yang telah dibuat pada sisi penerima telah berisi tipe data sesuai dengan data yang dikirimkan.
4. Performa dari sistem *gateway discovery* yang telah dibuat dan disematkan pada *WSN* yang berbasis modul komunikasi *LoRa* dipengaruhi oleh jarak setelah dilakukan pengujian terhadap jarak antara *sensor node* dan *gateway*. Menurut hasil pengujian performa yang telah dilakukan sebelumnya, dari tiga parameter yang diberikan menghasilkan data diagram menanjak di jarak maksimal pengujian pada parameter *association time*, *unicast loss* dan

broadcast loss. Pada jarak maksimal pengujian performa yaitu jarak 1000 meter nilai masing-masing parameter adalah *association time* sebesar 354.4 mili detik, *unicast loss* sebesar 6% dan *broadcast loss* sebesar 20%. Hal ini berdampak pada kinerja sistem *gateway discovery* karena dengan besarnya nilai dari parameter tersebut, maka kinerja sistem akan semakin buruk. Oleh karena itu, dapat disimpulkan bahwa jarak berpengaruh sekali terhadap performa dari sistem *gateway discovery* yang telah dibuat. Semakin jauh jarak antara *sensor node* dan *gateway* maka performa dari sistem akan semakin buruk.

7.2 Saran

Berdasarkan kesimpulan penelitian yang telah penulis uraikan sebelumnya, penulis merekomendasikan beberapa saran untuk penelitian selanjutnya. Saran dapat berguna untuk penelitian berikutnya yang terkait dengan topik penelitian ini. Penulis memberikan beberapa saran sebagai berikut:

1. Penelitian berikutnya dapat dilakukan dengan mengimplementasikan algoritma keamanan pada perangkat *LoRa* seperti *AES128* untuk menjaga kerahasiaan data.
2. Pada penelitian ini hanya menggunakan mikrokontroler *Arduino nano*, diharapkan pada penelitian selanjutnya dapat menggunakan mikrokontroler lain seperti *Raspberry pi*.
3. Jumlah sensor untuk penelitian selanjutnya dapat ditambah sehingga sistem dapat mengirim beragam jenis data sensor dan dapat mendeteksi sensor apa yang digunakan.

DAFTAR PUSTAKA

- Arduino, 2017. *Software: Download the Arduino IDE*. [Online]
Available at: <https://www.arduino.cc/en/Main/Software>
[Diakses 10 Februari 2018]
- Augustin, A. et al., 2016. A Study of LoRa: Long Range & Low Power Networks for the Internet of Things. *Sensors* 2016, 16(9), p.1466.
- Flammini A. dan Sisinni E., 2014). " *Wireless Sensor Networking in the Internet of Things and Cloud Computing Era*", Science Direct : Procedia Engineering 62-679
- Lavric, A. dan Popa, V., 2017. " *Internet of Things and LoRa™ Low-Power Wide-Area Networks Challenges*". Romania: Stefan cel Mare University of Suceava
- Lopez Research. (2013, 2013). An Introduction to the Internet of Things (IoT). *Part 1. of The IoT Series*, pp. 1-6.
- LoveToKnow, 2017. *Zero configuration - Computer Definition*. [Online]
Available at: <http://www.yourdictionary.com/zero-configuration#computer>
[Diakses 10 Februari 2018]
- Poole, I., 2017. *LoRa Wireless for M2M & IoT*. [Online]
Available at: <http://www.radio-electronics.com/info/wireless/lora/basics-tutorial.php>
[Diakses 10 Februari 2018]
- Pundir, Y., 2015. *Internet of Things(IoT): Challenges and Future Direction*, s.l.: s.n.
- Reynders, B., Meert, W. and Pollin, S., 2016. " *Range and coexistence analysis of long range unlicensed communication*", Proc. 23rd Int. Conf. Telecommun. (ICT), Thessaloniki, pp. 1–6.
- Silva, J. C., Rodrigues J. J. P. C., Alberti A. M., Solic P. and Aquino A. L. L., 2017. " *LoRaWAN - A Low Power WAN Protocol for Internet of Things: a Review and Opportunities*". Split: Computer and Energy Science (SpliTech)
- Sinauarduino, 2016. *Mengenal Arduino Software (IDE)*. [Online]
Available at: <http://www.sinauarduino.com/artikel/mengenal-arduino-software-ide/>
[Diakses 10 Februari 2018].
- Sugiarto, B., 2016. " *Perancangan Sistem Pengendalian Suhu pada Gedung Bertingkat dengan Teknologi Wireless Sensor Network*". Jakarta: LIPI vol. 4 no. 1
- Wazir, Jaffer & Barukab, Omar & Almagrabi, Alaa & Khan, Sher afzal. (2016). Minimizing Denial of Service Attack for Multiple Base Stations in Wireless Sensor Network. *VFAST Transactions on Software Engineering*. 11. 10.21015/vtse.v11i2.429.

- Wixted, A. J., Kinnaird P., Larijani H., Tait A., Ahmadinia A. and Strachan N., 2016. "Evaluation of LoRa and LoRaWAN for Wireless Sensor Networks". Orlando: IEEE SENSORS, pp. 1–3.
- Yinbiao S., 2014. *Internet of Things: Wireless Sensor Networks*. Geneva: International Electrotechnical Commission.
- Zhao, W. et al., 2017. *"Design and Implementation of Smart Irrigation System Based on LoRa"*. Beijing: Beijing University of Posts and Telecommunications

LAMPIRAN A

```
1 //NODE
2 #include <SPI.h>
3 #include <RH_RF95.h>
4 #include <RHDatagram.h>
5 #include <DHT.h>
6 #define dhtType DHT11
7
8 #define RFM95_CS 10
9 #define RFM95_RST 9
10 #define RFM95_INT 2
11
12 // Change to 434.0 or other frequency, must match RX's freq!
13 #define RF95_FREQ 433.0
14 #define node_addr 5
15 #define dhtPin 3 // Use d3 pin to connect the data line of DHT11, it
16 is up to you
17
18 DHT dht11 (dhtPin,dhtType);
19 // Singleton instance of the radio driver
20 RH_RF95 rf95(RFM95_CS, RFM95_INT);
21
22 RHDatagram kurir(rf95, node_addr);
23 // Blinky on receipt
24 #define LED 13
25 struct broad {
26     uint8_t gtwAddr,cons;
27 };
28
29 struct dataSet
30 {
31     uint8_t ack, id, temp, hum;
32 };
33
34 void setup()
35 {
36     dht11.begin();
37     pinMode(LED, OUTPUT);
38     pinMode(RFM95_RST, OUTPUT);
39     digitalWrite(RFM95_RST, HIGH);
40
41     while (!Serial);
42     Serial.begin(9600);
43     delay(100);
44
45     Serial.println("Lora Node Sensor Humidity and Temperature\n\n");
46
47     // manual reset
48     digitalWrite(RFM95_RST, LOW);
49     delay(10);
50     digitalWrite(RFM95_RST, HIGH);
51     delay(10);
52
53     while (!rf95.init()) {
54         Serial.println("LoRa radio init failed");
55         while (1);
56     }
57     Serial.println("LoRa radio init OK!");
```

```

58
59 // Defaults after init are 434.0MHz, modulation GFSK_Rb250Fd250,
60 +13dbM
61 if (!rf95.setFrequency(RF95_FREQ)) {
62     Serial.println("setFrequency failed");
63     while (1);
64 }
65 Serial.print("Set Freq to: "); Serial.println(RF95_FREQ);
66
67 rf95.setTxPower(23, false);
68 }
69
70 void loop()
71 {
72     if (kurir.available())
73     {
74         Serial.println("available message");
75         // Should be a message for us now
76         uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
77         uint8_t len = sizeof(buf);
78         uint8_t from;
79         Serial.println("kurir masuk");
80         struct broad auth;
81         if (kurir.recvfrom((uint8_t*) &auth, &len, &from))
82         {
83             // RH_RF95::printBuffer("Received: ", buf, len);
84             if (auth.cons == 13) // auth from gateway
85             {
86                 digitalWrite(LED, HIGH);
87                 // RH_RF95::printBuffer("Received: ", buf, len);
88                 Serial.println("Got: true gateway");
89
90                 struct dataSet data;
91                 data.ack = 201 ;
92                 data.id = node_addr ; //Device ID / sensor node
93                 data.hum = dht11.readHumidity(); // store humidity data
94                 data.temp = dht11.readTemperature(); // store temperature data
95                 kurir.setHeaderFrom(node_addr);
96                 kurir.sendto((uint8_t *) &data, sizeof(struct dataSet),
97 auth.gtwAddr); // Send out Ack + ID + Sensor data to LoRa gateway
98                 kurir.waitPacketSent();
99                 Serial.println(from, DEC);
100                 Serial.println("Sent a reply");
101                 digitalWrite(LED, LOW);
102                 Serial.print("Current humidity = ");
103                 Serial.print((uint8_t)dht11.readHumidity(), DEC);
104                 Serial.print("% ");
105                 Serial.print("temperature = ");
106                 Serial.print((uint8_t) dht11.readTemperature(), DEC);
107                 Serial.println(" C ");
108             } else {
109                 Serial.println("gateway belum diketahui");
110             }
111         }
112         else
113         {
114             Serial.println("Receive failed");
115         }
116     }
117 }

```

LAMPIRAN B

```

1 //GATEWAY
2 #include <SPI.h>
3 #include <RH_RF95.h>
4 #include <RHDatagram.h>
5 #include <stdio.h>
6
7 #define RFM95_CS 10
8 #define RFM95_RST 9
9 #define RFM95_INT 2
10
11 // Change to 434.0 or other frequency, must match RX's freq!
12 #define RF95_FREQ 433.0
13 #define gtw_addr 9
14 // Singleton instance of the radio driver
15 RH_RF95 rf95(RFM95_CS, RFM95_INT);
16
17 RHDatagram kurir(rf95, gtw_addr);
18 uint32_t displayTimer = 0;
19 struct broad {
20     uint8_t gtwAddr, cons;
21 };
22 struct dataSet
23 {
24     uint8_t ack;
25     uint8_t id;
26     uint8_t temp;
27     uint8_t hum;
28 };
29 struct nodes {
30     uint8_t addr;
31 };
32 struct waktu {
33     unsigned long t;
34 };
35 struct nodes avail[10] = {};
36 struct waktu w[10] = {};
37
38 void setup()
39 {
40     pinMode(RFM95_RST, OUTPUT);
41     digitalWrite(RFM95_RST, HIGH);
42
43     //while (!Serial);
44     Serial.begin(9600);
45     delay(100);
46
47     Serial.println("Arduino LoRa TX Test!");
48
49     // manual reset
50     digitalWrite(RFM95_RST, LOW);
51     delay(10);
52     digitalWrite(RFM95_RST, HIGH);
53     delay(10);
54
55     while (!kurir.init()) {
56         Serial.println("kurir init failed");
57         while (1);

```

```

58     }
59     Serial.println("LoRa radio init OK!");
60     Serial.println(kurir.thisAddress());
61     // Defaults after init are 434.0MHz, modulation GFSK_Rb250Fd250,
62 +13dbM
63     if (!rf95.setFrequency(RF95_FREQ)) {
64         Serial.println("setFrequency failed");
65         while (1);
66     }
67     Serial.print("Set Freq to: "); Serial.println(RF95_FREQ);
68
69     // Defaults after init are 434.0MHz, 13dBm, Bw = 125 kHz, Cr = 4/5,
70 Sf = 128chips/symbol, CRC on
71
72     // The default transmitter power is 13dBm, using PA_BOOST.
73     // If you are using RFM95/96/97/98 modules which uses the PA_BOOST
74 transmitter pin, then
75     // you can set transmitter powers from 5 to 23 dBm:
76     rf95.setTxPower(23, false);
77 }
78
79 void loop()
80 {
81     if (millis() - displayTimer >= 5000) {
82         displayTimer = millis();
83         Serial.println("Sending broadcast to Node");
84         // Send a message to node
85         struct broad auth;
86         auth.gtwAddr = gtw_addr;
87         auth.cons = 14;
88         rf95.send((uint8_t *) &auth, sizeof(struct broad));
89     }
90     // Now wait for a reply
91     uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
92     uint8_t len = sizeof(buf);
93     Serial.println("Waiting for reply...");
94     if (kurir.waitAvailableTimeout(1000))
95     {
96         // Should be a reply message for us now
97         // Serial.println(buf[0]);
98         uint8_t from;
99         struct dataSet data;
100         if (kurir.recvfrom((uint8_t *)&data, &len, &from))
101         {
102             if (data.ack == 201)
103             {
104                 Serial.println("Ack node OK");
105                 Serial.print("Available node : ");
106                 for (int j = 0; j < (sizeof(avail) / sizeof(avail[0])); ++j)
107                 {
108                     if (j == (sizeof(avail) / sizeof(avail[0]) - 1)) {
109                         for (int a = 0; a < (sizeof(avail) / sizeof(avail[0]));
110 ++a)
111                         {
112                             if (avail[a].addr == 0) {
113                                 avail[a].addr = from;
114                                 w[a].t = millis();
115                                 break;
116                             }
117                         }
118                     }
119                 }
120             }
121         }
122     }
123 }

```

```

118         } else if (avail[j].addr == from ) {
119             w[j].t = millis();
120             break;
121         } else if (avail[j].addr != from ) {
122             continue;
123         }
124     }
125     unsigned long param = millis();
126     for (int c = 0; c < (sizeof(w)/sizeof(w[0])); ++c){
127         if ((param-w[c].t)>=50000){
128             avail[c].addr = 0;
129             w[c].t = 0;
130         }
131     }
132     for (int i = 0; i < (sizeof(avail) / sizeof(avail[0])); ++i)
133     {
134         if (avail[i].addr != 0) {
135             Serial.print(avail[i].addr);
136             Serial.print(" ");
137         }
138     }
139     //      Serial.print(from);
140     Serial.print("\nMendapat data dari node : ");
141     Serial.println(from, DEC);
142     Serial.print("ID = ");
143     Serial.print(data.id);
144     Serial.print(" : Current humidity = ");
145     Serial.print(data.hum);
146     Serial.print("% ");
147     Serial.print("temperature = ");
148     Serial.print(data.temp);
149     Serial.println(" C ");
150 }
151 }
152 else
153 {
154     Serial.println("Receive failed");
155 }
156 }
157 else
158 {
159     Serial.println("No reply, is there a listener around?");
160 }
161 }

```