SISTEM WIRELESS SENSOR NETWORK MENGGUNAKAN ARDUINO NANO DAN LABVIEW NI UNTUK APLIKASI MONITORING SUHU DAN KELEMBABAN

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rizka Suhana
NIM: 125150300111013

PROGRAM STUDITEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

Sistem *Wireless Sensor Network* Menggunakan Arduino Nano Dan Labview Ni Untuk Aplikasi Monitoring Suhu Dan Kelembaban

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Rizka Suhana
NIM: 125150300111013

Skripsi ini telah diuji dan dinyatakan lulus pada
1 Juli 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Mochammad Hannats Hanafi I., S.ST, M.T.	Sabriansyah Rizqika Akbar, S.T, M.Eng.
NIK: 201405 881229 1 001	NIP: 19820809 201212 1 004

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 didalam 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, 1 Juli 2016

Rizka Suhana
NIM: 125150300111013

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan penyusunan usulan penelitian ini. Shalawat serta salam semoga selalu terscurahkan kepada Baginda Besar Nabi Muhammad SAW.

Dalam penyusunan usulan penelitian ini, penulis berupaya semaksimal mungkin agar dapat memenuhi harapan semua pihak, namun penulis menyadari tentunya masih banyak kekurangan yang terdapat dalam usulan penelitian ini.

Dalam kesempatan ini pula, penulis menyampaikan ucapan terima kasih dan penghargaan yang sebesar-besarnya atas bantuan, motivasi, didikan dan bimbingan yang diberikan kepada penulis selama ini, antara lain kepada yang terhormat:

1. Ayahanda dan Ibunda dan seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya memberikan semangat kepada penulis, serta senantiasa tiada hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
2. Bapak Mochammad Hannats Hanafi I., S.ST, M.T. dan Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. selaku dosen pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
3. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Informatika.
4. Seluruh civitas akademika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih terdapat banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 1 Juli 2015

Penulis
16hand.sgod91@gmail.com

ABSTRAK

Telemetri adalah proses pengukuran parameter suatu obyek (benda, ruang, kondisi alam) yang hasil pengukurannya dikirimkan ke tempat lain melalui proses pengiriman data baik dengan menggunakan kabel maupun tanpa menggunakan kabel (*wireless*). Demikian dibutuhkan sebuah sistem yang dapat melakukan pengukuran dan pemantauan suhu dan kelembaban dari lokasi yang berjauhan dengan cara *wireless*. Sehingga sistem ini dapat mengurangi hambatan untuk mendapatkan informasi.

Pada penelitian ini, penulis melakukan penelitian yang berhubungan dengan *wireless sensor network* dengan menggunakan Arduino nano sebagai mikrokontroler dan Labview NI sebagai pengolah data suhu dan kelembaban untuk ditampilkan sebagai aplikasi monitoring.

Hasil analisis dari pengujian *delay* yaitu, node client disimpulkan : 1. *Delay* ke 1 dan 2 akan memiliki *delay* tetap/stabil pada delay 1000 ms keatas; 2. *Delay* ke 2 dan 3 akan memiliki nilai tetap/stabil pada semua delay; 3. *Delay* ke 3 dan 1 (eksekusi berikutnya) akan bernilai kurang lebih sama dengan *delay* yang diberikan. Hasil pengujian *delay* pada node server disimpulkan : 1. *Delay* ke 1 dan 2 akan memiliki *delay* tetap/stabil pada delay 1000 ms keatas; 2. *Delay* ke 2 1 (eksekusi berikutnya) akan bernilai kurang lebih sama dengan *delay* yang diberikan.

Dari ketiga pengujian diambil beberapa kesimpulan , alat penelitian (node client dan server) bisa digunakan dengan baik, pengujian delay memberikan hasil bahwa pemberian *delay* pada node server berbanding lurus dengan *delay* yang diamati pada pengujian dan pengujian terakhir adalah pengujian fitur aplikasi monitoring. Pengujian fitur sudah baik dalam penggunaannya, seperti tombol-tombol yang digunakan, kotak pengaturan dan grafik-grafik yang dipakai sudah sesuai dengan sistem yang dirancang.

Kata kunci: Telemetri; *Wireless Sensor Network;* LabView NI; Arduino Nano

ABSTRACT

Telemetry is the process parameter measurements of an object (object, space, natural conditions) that the measurement results are sent to other places through the process of sending data using either wired or wirelessly (wireless). Thus we need a system that can perform the measurement and monitoring of temperature and humidity from remote locations by means of wireless. So that this system can reduce hindrances to access to information.

In this study, the authors conducted research related to wireless sensor network using the Arduino nano as microcontroller and NI Labview as a data processor temperature and humidity to be displayed as a monitoring application.

The results of the analysis of the test delay, namely, client node concluded: 1. Delay to 1 and 2 will have a fixed delay / delay stable at 1000 ms and above; 2. Delay to 2 and 3 will have the value of fixed / stable on all delay; 3. Delay to 3 and 1 (next execution) would be worth more or less equal to the delay given. The test results concluded delay at node server: 1. Delay to 1 and 2 will have a fixed delay / delay stable at 1000 ms and above; 2. Delay to 2 1 (next execution) would be worth more or less equal to the delay given.

From the three tests taken some conclusion, research tools (client and server nodes) can be used well, delay testing provides results that the administration delay on the server node is proportional to the delay observed in the testing and final testing is testing the application monitoring feature. Tests have good features in use, such as the buttons are used, box settings and graphics used are in accordance with the system designed.

Keywords: *Telemetry*; *Wireless Sensor Network*; NI LabView; Arduino Nano;

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Teknologi yang ada membuat segala sesuatu yang kita lakukan menjadi lebih mudah. Manusia selalu berusaha untuk menciptakan sesuatu yang dapat mempermudah aktifitasnya, hal ini yang mendorong perkembangan teknologi yang telah menghasilkan alat sebagai piranti untuk mempermudah kegiatan manusia bahkan menggantikan peranan manusia dalam suatu fungsi tertentu. Teknologi memegang peranan penting di era moderenisasi seperti saat ini, dimana teknologi menjadi bagian yang tidak dapat dipisahkan dalam kehidupan sehari-hari.

Zaman dan teknologi yang telah berkembang, kebutuhan atas informasi sangat dibutuhkan dalam berbagai bidang, baik pertanian, perindustrian, maupun bidang yang lainnya sehingga bisa menunjang kinerja bidang tersebut. Salah satunya informasi suhu dan kelembaban adalah informasi yang sangat penting bagi petani atau orang yang bekerja di sawah maupun di perkebunan. Namun untuk sistem pemantauan dan pengukurannya masih secara manual dan tidak bisa memungkinkan untuk pemantauan secara terus menerus karena faktor geografis dan jarak, sehingga hal ini bisa menghambat untuk memperoleh informasi tersebut.

Kendala pengukuran pada lokasi yang sulit terjangkau dapat diatasi dengan menggunakan metode pengukuran jarak jauh (telemetri). Telemetri dan telekontrol merupakan pengukuran dan pengendalian jarak jauh, yang dapat menggunakan kabel atau tanpa kabel (*wireless*). (Azam Muzakhim, 2011).

Telemetri adalah proses pengukuran parameter suatu obyek (benda, ruang, kondisi alam) yang hasil pengukurannya dikirimkan ke tempat lain melalui proses pengiriman data baik dengan menggunakan kabel maupun tanpa menggunakan kabel (*wireless*). Dengan demikian dibutuhkan sebuah sistem yang dapat melakukan pengukuran dan pemantauan suhu dan kelembaban dari lokasi yang berjauhan dengan cara *wireless*. Sehingga diharapkan dapat mengurangi hambatan untuk mendapatkan informasi.

Penelitian sebelumnya berhubungan dengan pengembangan sistem telemetri *wireless* diantaranya Azam Muzakhim (2011) yaitu Telemetri dan Telekontrol Antara Mikrokontroller Menggunakan *Xbee Pro Wireless.* Pada Penelitian ini sistem Telemetri suhu dan kelembaban menggunakan *Xbee Pro* Mencapai jarak 110 meter dan belum mempunyai data *logger.* Penelitian Hendrit (2011) menggunakan modul *Xbee Pro* untuk komunikasi data antara mikrokontroller dengan personal komputer untuk monitoring suhu dan kelembaban. (Heri Susanto, et al, 2013)

Teknologi ini menggunakan *Wireless sensor network* karena kelebihannya yang menunjang kinerja Sistem Monitoring Suhu dan Kelembaban. *Wireless sensor network* merupakan sebuah perangkat komputasi *embeded* yang memiliki *interface* dengan sensor yang saling berkomunikasi menggunakan pemancar berupa nirkabel yang nantinya akan membentuk sebuah jaringan yang tersusun.

Wireless Sensor Network terdiri atas satu atau lebih node sensor yang digunakan untuk menangkap informasi sesuai dengan karakteristiknya.

Program LabView dapat disebut sebagai virtual instrument atau Vis karena operasi dan penampilannya meniru secara fisik seperti multimeter dan osiloskop. labView ini berisi macam-macam peralatan untuk menghasilkan ketelitian, tampilan, penyimpanan data, seperti halnya perlengkapan untuk membantu anda melakukan pemecahan masalah pengkodean. Setiap VI (Virtual Istrumen) menggunakan fungsi-fungsi yang menggerakkan masukan dari pemakai di panel antar muka atau sumber lain dan menampilkan informasi tersebut atau disimpan pada komputer tersebut atau komputer lain.

Dari penelitian dan peninjauan di atas, penulis tertarik untuk melanjutkan dan mengembangkan penelitian tersebut dengan kondisi metode dan cara yang berbeda.

1.2 Rumusan masalah

Berdasarkan dari latar belakang tersebut maka didapat beberapa rumusan masalah dalam mengerjakan tugas akhir ini antara lain:

1. Bagaimana merancang sebuah sistem monitoring antara Arduino Nano sebagai pemroses data dan DHT11 (Sensor Suhu dan Kelembaban) sebagai *input* dengan LABView NI sebagai *output* ?
2. Bagaimana menerapkan komunikasi data nirkabel dengan menggunakan nRF24L01?
3. Bagaimana menerapkan akuisisi waktu didalam jalannya program pada bagian fungsi tertentu?

1.3 Tujuan

Dalam penelitian ini memiliki beberapa tujuan yang ingin dicapai oleh penulis, yaitu:

1. Mengimplementasikan sistem yang dapat menampilkan dan menyimpan data suhu dan kelembaban pada LabView NI.
2. Mengimplementasikan sistem yang dapat mengukur suhu dan kelembaban sercara nirkabel menggunakan arduino nano dan nRF24L01.
3. Mengimplementasikan akuisisi (pengambilan) waktu untuk mengetahui delay dari suatu fungsi ke fungsi yang lain dalam satu program.

1.4 Manfaat

Penelitian ini diharapkan dapat bermanfaat baik bagi penulis maupun bagi pengguna antara lain:

a. Bagi Penulis
    1. Mengaplikasikan ilmu yang diperoleh selama mengikuti perkuliahan di Sistem Komputer Universitas Brawijaya
    2. Mendapatkan pengetahuan bagaimana mengimplementasikan komunikasi secara nirkabel dengan Arduino Nano dan LABView NI sebagai aplikasi monitoring.

b. Bagi Pengguna
   1. Bisa memonitoring sewaktu-waktu suhu dan kelembaban tanah perkebunan pada database (Excel).
   2. Terciptanya teknologi yang tidak menggunakan kabel, mengurangi pemborosan listrik dan mencegah terjadinya *global warning*.

1.5 Batasan masalah

Dikarenakan luasnya permasalahan di dalam pembahasan dan agar tidak terjadi kesalahpahaman maksud dari apa yang ada di dalam penulisan tugas akhir ini maka dibutuhkan pembatasan masalah tersebut antara lain:

1. Sensor Suhu yang digunakan adalah DHT11.
2. Modul pengiriman data secara nirkabel menggunakan NRF24L01. Dimana memiliki keterbatasan akan jangkauan pengiriman data yaitu 100m. akan terdapat banyak paket loss apabila peletakan node client yang jauh.
3. Pada pengujian ini menggunakan jarak sejauh 1 – 2 meter.
4. Kisaran suhu yang digunakan dari 0^{o} C sampai 50^{o} C.
5. Kisaran kelembaban yang digunakan dari 20 sampai 90% RH.
6. Tidak melakukan penelitian lebih lanjut tentang memori dan buffer pada data.

1.6 Sistematika pembahasan

Untuk mencapai tujuan yang diharapkan, maka sistematika penulisan yang disusun dalam laporan tugas akhir ini sebagai berikut:

BAB I Pendahuluan

Pada pendahuluan dijelaskan mengenai latar belakang, rumusan masalah, tujuan, manfaat penelitian, batasan penelitian dan sistematika pembahasan dari Perancangan Sistem *Wireless Sensor Network* menggunakan Arduino Nano dan LABView NI untuk Aplikasi Monitoring Suhu dan Kelembaban .

BAB II Landasan Kepustakaan

Membahas mengenai teori-teori yang berkaitan dan menunjang dalam penyelesaian tugas akhir ini. Dasar teori yang diambil berasal dari jurnal, buku, dan sumber referensi lainnya yang berhubungan dengan topik yang akan diteliti.

BAB III Metodologi

Membahas tentang langkah kerja yang dilakukan dalam penulisan diantaranya studi literatur, analisis kebutuhan sistem, impementasi dan pengujian dari perancangan Sistem *Wireless Sensor Network* menggunakan Arduino Nano dan LABView NI untuk Aplikasi Monitoring Suhu dan Kelembaban.

BAB IV Persyaratan

Membahas analisis kebutuhan dan perancangan sesuai dengan teori yang ada.

BAB V Perancangan dan Implementasi

Membahas tentang implementasi berdasarkan metodologi yang akan dibuat.

BAB VI Pengujian dan Analisis

Memuat tentang hasil pengujian dan analisis pada sistem yang telah direalisasikan.

BAB VII Penutup

Memuat kesimpulan yang diperoleh dari pembuatan dan pengujian perangkat lunak yang dikembangkan serta saran-saran untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Tinjauan Pustaka

Penelitian tentang telemetri dan telekontrol telah dilakukan sebelumnya menggunakan modul XBEE PRO Wireless. Penelitian pertama yang dilakukan oleh Azam Muzakhim pada tahun 2011 dalam jurnal yang berjudul "Telemetri dan Telekontrol Antar Mikrokontroler Menggunakan *XBEE PRO WIRELESS*". Penelitian ini membahas tentang pengukuran dan pengendalian jarak jauh yang tanpa menggunakan kabel untuk menghubungkannya. *XBEE PRO WIRELESS* sebagai modul pengiriman data dan ATMega8535 sebagai mikrokontroler, sistem ini terdapat Mikrokontroler (MCU) *Master* dan *Slave.* Pada MCU *Master* dihubungkan dengan modul *switch button*, modul tampilan LCD dan modul *Xbee Pro.* Pada MCU Slave terdapat modul sensor temperature dan kelembaban SHT11, motor servo DC dan modul Xbee Pro.

Penelitian kedua tentang telemetri dan telekontrol yang telah dilakukan sebelumnya oleh Heri Susanto, et al pada tahun 2013 dengan jurnal yang berjudul "Perancangan Sistem Telemetri *Wireless* Untuk Mengukur Suhu Dan Kelembaban Berbasis Arduino Uno R3 Atmega328p Dan *Xbee Pro*". Penelitian ini membahas tentang pengukuran jarak jauh. Penelitian ini merancang sistem telemetri *wireless* yang dapat mengukur suhu dan kelembaban dengan desain *portable* yang dilengkapi perekam data, hasil pengukuran tersebut bisa ditampilkan melalui LCD. Sistem telemetri *wireless*. Sistem terbagi dua bagian yaitu Unit pengirim terdiri dari sensor DHT11, I/O *expansion,* Arduino Uno R3, mikrokontroler ATmega328P, modul *Xbee Pro* dan baterai. Unit penerima terdiri dari Unit penerima terdiri dari Modul *Xbee Pro*, I/O *expansion*, Arduino Uno R3, *mikrokontroller* ATmega328P, LCD, Modul *SD Card* dan baterai.

Penelitian ketiga tentang telemetri dan telekontrol yang telah dilakukan sebelumnya oleh Eko Kristianto pada tahun 2013 dengan jurnal yang berjudul "Monitoring Suhu Jarak Jauh Generator AC Berbasis Mikrokontroler". Penelitian ini berfungsi untuk memonitor suhu generator dari jarak jauh tanpa kabel pada saat generator bekerja dengan tujuan generator dapat dipantau dari jarak jauh sehingga memudahkan dalam memantau suhu generator.

2.2 Dasar Teori

Dasar teori yang digunakan sebagai penunjang untuk menyelesaikan system ini yaitu penjelasan Arduino, *wireless sensor network,* DTH11, LabView NI, nRF24L01.

2.2.1 Arduino

Papan Arduino Nano merupakan papan mikrokontroler yang berukuran kecil atau dapat diartikan juga dengan suatu rangkaian berukuran kecil yang didalamnya terdapat komputer berbentuk suatu chip yang kecil.

Pada Gambar 2.1. dapat dilihat sebuah papan Arduino dengan beberapa bagian komponen didalamnya.

Gambar 2.1 *Hardware* Arduino Nano tampak depan (kiri) dan tampak belakang (kanan)

Sumber : https://www.arduino.cc/en/Main/ArduinoBoardNano

Arduino Nano adalah papan mikrokontroler berdasarkan Atmel ATmega168 or ATmega328 . Ini memiliki 14 digital pin input / output (dimana 6 dapat digunakan sebagai output PWM), 8 input *analog*, *resonator on-board*, tombol reset, dan lubang untuk pemasangan *pin header*.

2.2.1.1 Ringkasan

Berikut adalah ringkasan dari Arduino Nano:

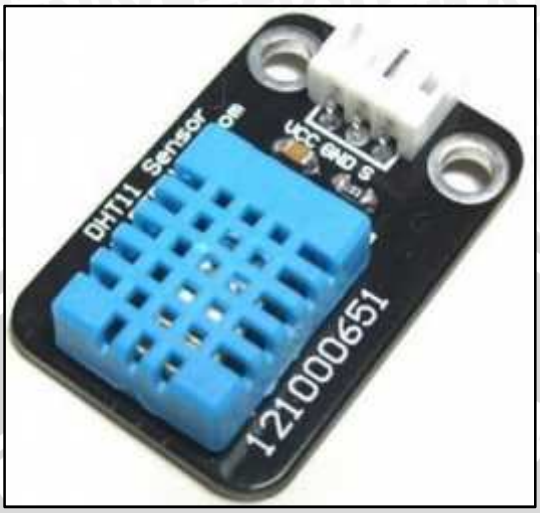
Tabel 2.1 Ringkasan Arduino Nano

Mikrokontroler	Atmel ATmega168 or ATmega328
Tegangan Operasi	5 V
Tegangan input (Rekomendasi)	7 – 12 V
Tegangan Input (Batas)	6 – 20 V
Pin Digital I/O	14 (6 diantaranya menyediakan *output* PWM)
Pin Analog input	8
Arus DC per pin I/O	40 mA
Flash Memory	16 KB (ATmega168) atau 32 KB (ATmega328) yang mana 2 KB digunakan *bootloader*
SRAM	1 KB (ATmega168) atau 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) atau 1 KB (ATmega328)
Clock Speed	16 MHz
Dimensi	0,73 x 1,70 inchi
Panjang	45 mm
Lebar	18 mm
Berat	5 gram

2.2.2 Sensor Temperature dan Kelembaban DHT11

DHT11 Suhu & Kelembaban Sensor memiliki suhu & kelembaban sensor kompleks dengan output sinyal digital dikalibrasi. Menggunakan *exclusive digital-signal-acquisition technique* dan *temperature & humidity sensing technology,* memastikan keandalan yang tinggi dan stabilitas jangka panjang yang sangat baik. Sensor ini mencakup pengukuran kelembaban resistif-jenis komponen dan

pengukuran komponen suhu NTC, dan menghubungkan ke berkinerja tinggi 8-bit mikrokontroler, menawarkan kualitas yang sangat baik, respon cepat, anti-gangguan kemampuan dan efektivitas biaya.



Gambar 2.2 DHT11

Sumber : https://www.mysensors.org/build/humidity

Setiap sensor DHT11 memiliki fitur kalibrasi sangat akurat dari kelembaban ruang kalibrasi. Koefisien kalibrasi yang disimpan dalam memori program OTP, sensor internal mendeteksi sinyal dalam proses, kita harus menyebutnya koefisien kalibrasi. *Singgle wire serial interface* terintegrasi untuk menjadi cepat dan mudah. Ukurannya kecil, daya rendah, sinyal transmisi jarak hingga 20 meter.

2.2.2.1 Spesifikasi Teknik

a. Overview

Berikut adalah penjelasan mengenai overview pada DHT11:

Tabel 2.2 *Overview*

Item	Jangkauan Pengukuran	Akurasi Kelembaban	Akurasi Suhu	Resolusi	Paket
DHT11	20 – 90 % RH 0 – 50 °C	± 5 % RH	± 2°C	1	4 pin baris tunggal

b. Detail Spesifikasi

Berikut adalah detai spesifikasi dari DHT11:

Tabel 2.3 Detail Spesifikasi

Parameter	Kondisi	Minimum	Tipikal	Maksimum
Kelembaban				
Resolusi		1%RH	1%RH	1%RH
			8bit	
Pengulangan			± 1%RH	
Akurasi	25%		± 4%RH	
	0-50°C			± 5%RH
interchangeability	Perubahan penuh			
Jangkauan Pengukuran	0°C	30%RH		90%RH
	25°C	20%RH		90%RH
	50°C	20%RH		80%RH
Waktu Respon (Detik)	1/e(63%)25°C, 1m/s Air	6 detik	10 detik	15 detik
Jangka Panjang Stabilitas	Tipikal		± 1%RH/tahun	
Temperatur				
Resolusi		1°C	1°C	1°C
		8 Bit	8 Bit	8 Bit
Akurasi		± 1°C		± 2°C

2.2.3 LabView NI

LabVIEW adalah pemrograman grafis yang dapat digunakan untuk dengan cepat dan efisien membuat aplikasi dengan *user interface* profesional. Jutaan insinyur dan ilmuwan menggunakan LabVIEW untuk mengembangkan pengukuran canggih, tes, dan aplikasi sistem kontrol menggunakan ikon intuitif dan kabel. Selain itu, platform LabVIEW *scalable* di target yang berbeda dan OS. Faktanya, LabVIEW menawarkan integrasi yang tak tertandingi dengan ribuan perangkat keras dan menyediakan ratusan *built-in* perpustakaan untuk analisis canggih dan visualisasi data bagi Anda untuk membuat maya instrumen Anda dapat menyesuaikan dengan kebutuhan Anda.

Program LabVIEW dapat memberikan tampilan dan pengoperasian instrumen fisik, seperti sebagai osiloskop dan multimeter, program LabVIEW disebut instrumen virtual atau, lebih umum, Vis. Vis memiliki *front panel* dan diagram blok. *Front panel* adalah *user interface*. *Blockdiagram* adalah pemrograman di belakang user *interface*. Setelah kita membangun *front panel*, kita bisa menambahkan kode menggunakan representasi grafis dari fungsi untuk mengontrol objek *front panel*. Kode pada diagram blok kode grafis, juga dikenal sebagai kode G atau kode *blockdiagram* Berbeda dengan teks berbasis bahasa pemrograman, seperti C ++ dan Visual Basic, LabVIEW menggunakan ikon

bukannya baris teks untuk membuat aplikasi. Dalam pemrograman berbasis teks, instruksi menentukan urutan eksekusi program. LabVIEW menggunakan pemrograman dataflow grafis. Dalam grafis pemrograman dataflow, aliran data melalui node pada diagram blok menentukan perintah eksekusi. Pemrograman grafis dan eksekusi dataflow adalah dua cara utama LabVIEW berbeda dari kebanyakan bahasa pemrograman tujuan umum lainnya.

2.2.3.1 Karakteristik LabVIEW

Program LabView memiliki beberapa karakteristik:

1. ***A graphical and compiled nature***

   Pada pemrograman LabVIEW diwakili oleh grafis, dengan ikon dan kabel bukan dengan teks, kode pada *blokdiagram* berisi konsep pemrograman yang sama ditemukan di sebagian besar bahasa tradisional. Untuk misalnya, kode G meliputi jenis data, loop, penanganan event, variabel, rekursi, dan pemrograman berorientasi objek. LabVIEW mengkompilasi kode G langsung ke kode mesin sehingga prosesor komputer dapat melaksanakannya. Anda tidak perlu mengkompilasi kode G dalam langkah terpisah.

2. ***Dataflow and Event-Driven Programming***

   Program LabVIEW mengeksekusi sesuai dengan aturan pemrograman dataflow bukan prosedural pendekatan ditemukan di sebagian besar bahasa pemrograman berbasis teks seperti C dan C ++. Aliran data eksekusi adalah *data-driven*, atau *data-dependent*. Aliran data antara node dalam kode G menentukan urutan eksekusi

   Fitur pemrograman-*event* memperpanjang lingkungan LabVIEW dataflow untuk memungkinkan pengguna interaksi langsung dengan program tanpa perlu *polling*. Pemrograman berbasis acara juga memungkinkan kegiatan *asynchronous* lainnya untuk mempengaruhi eksekusi kode G pada diagram blok.

3. ***Multi-Target and Multi-Platform***

   Dengan aplikasi LabVIEW, Anda dapat menargetkan *prosesor multicore* dan perangkat keras paralel lainnya seperti sebagai bidang *programmable gate array* (FPGA). Anda secara otomatis dapat skala aplikasi LabVIEW untuk CPU dengan dua, empat, atau lebih core, sering tanpa usaha pemrograman tambahan.

   Kode G, dengan pengecualian beberapa fungsi spesifik *platform*, adalah portabel antara berbeda Sistem LabVIEW untuk sistem operasi yang berbeda. Oleh karena itu, Anda dapat sering menggunakan kode yang sama apakah berjalan LabVIEW pada Windows, Mac OS X atau sistem Linux.

4. ***Object-Oriented***

   Pemrograman berorientasi objek adalah pendekatan pemrograman populer di berbagai macam bahasa pemrograman. Hal ini memungkinkan berbagai serupa, namun item yang berbeda, untuk diwakili sebagai kelas objek dalam perangkat lunak. LabVIEW menyediakan alat dan fungsi

sehingga Anda dapat menggunakan *object-oriented* teknik pemrograman dalam kode G.

5. Multithreading and Memory Managemen

LabVIEW memungkinkan kode Anda untuk memiliki paralelisme otomatis. Dalam bahasa lain jika Anda ingin menjalankan kode secara paralel, Anda harus mengelola beberapa *thread* secara manual. Lingkungan LabVIEW, dengan compiler dan eksekusi sistem bekerja sama, secara otomatis menjalankan kode secara paralel setiap kali mungkin. Sebagian besar waktu rincian sistem eksekusi tidak penting untuk Anda karena sistem melakukan hal yang benar tanpa intervensi. Namun, LabVIEW juga menyediakan Anda dengan pilihan untuk meningkatkan kinerja.

2.2.4 Wireless Sensor Network

Wireless Sensor Network (WSN) adalah suatu kesatuan dari proses pengukuran, komputasi , dan komunikasi yang memberikan kemampuan administartif kepada sebuah perangkat, observasi dan melakukan tindakan ketika terjadi sesuatu dalam lingkungan yang mengimplementasikan wireless. Sistem WSN ini lebih jauh efisien dibandingkan dengan penggunaan kabel. Sistem ini memiliki fungsi unuk berbagai jenis aplikasi yang dimana WSN mampu memenuhi kebutuhan teknologi dalam berbagai bidang ilmu, seperti halnya pada bidang *biology*, pertanian, perikanan dan lain sebagainya. Sebagai contoh penelitian bidang pertanian menginginkan monitoring akan kelembapan tanah pada tanaman tertentu (Kazem S,2007).

Komponen pada WSN ini meliputi sensor, modul *wireless*, serta komputer. Seluruh komponen yang dibutuhkan akan membentuk suatu jaringan yang dimana membentuk suatu fungsi monitoring yang nantinya akan mampu menampilkan informasi atau data yang di dapat pada lapangan berupa karakteristik dari sensor yang digunakan dengan menggunakan media wireless. Dalam hal ini karena WSN dapat digunakan untuk berbagai aplikasi maka penggunaan sensor dapat dipilih sesuai kebutuhan sistem.

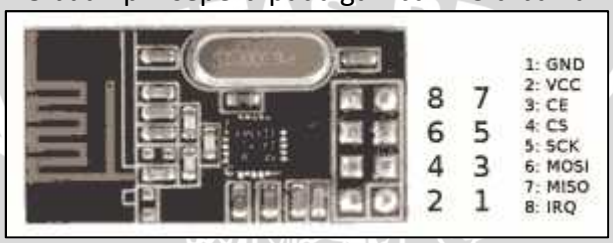
Arsiteksur pada WSN ini umumnya terbagi menjadi 4 bagian yaitu sistem *sensing, processing, communication* dan *power*. Penyatuan sistem ini haruslah diperhatikan saat awal melakukan perancangan sistem, terutama pada sistem *processing*. *System processing* dianggap sangat penting dikarenakan sistem tersebut dapat berpengaruh terhadap performa serta konsusmsi energi karena pada sistem proses merupakan pusat pengontrolan program yang akan dieksekusi maupun belum dieksekusi. Dalam WSN sendiri dapat menggunakan beberapa sensor node. WSN dapat mengimplementasikan sensor node cukup banyak, namun banyak atau sedikitnya node yang digunakan dikembalikan lagi kepada sistem itu sendiri. Sensor node ini dapat merasakan, mengukur dan mengumpulkan informasi dan data dari lingkungan.

2.2.5 Modul *Wireless* nRF24L01

Modul Wireless nRF24L01 adalah sebuah modul komunikasi jarak jauh yang memanfaatkan pita gelombang RF 2.4GHz ISM *(Industrial, Scientific and Medical*). Modul ini menggunakan antarmuka SPI untuk berkomunikasi. Tegangan kerja dari modul ini adalah 5V DC.

NRF24L01 memiliki *baseband logic Enhfanced ShockBurst™ hardware protocol accelerator* yang *support "high-speed SPI interface for the application controller*". nRF24L01 memiliki *true ULP solution*, yang memungkinkan daya tahan baterai berbulan-bulan hingga bertahun-tahun. Modul ini dapat digunakan untuk pembuatan pheripheral PC, piranti permainan, piranti fitnes dan olahraga, mainan anak-anak dan alat lainnya.

Modul ini memiliki 8 buah pin seperti pada gambar 2.3 dibawah ini.



Gambar 2.3 Modul *wireless* nRF24L01

Sumber: http://www.vcc2gnd.com/pic/NRF20L01_SI24RI_PinOuts.png

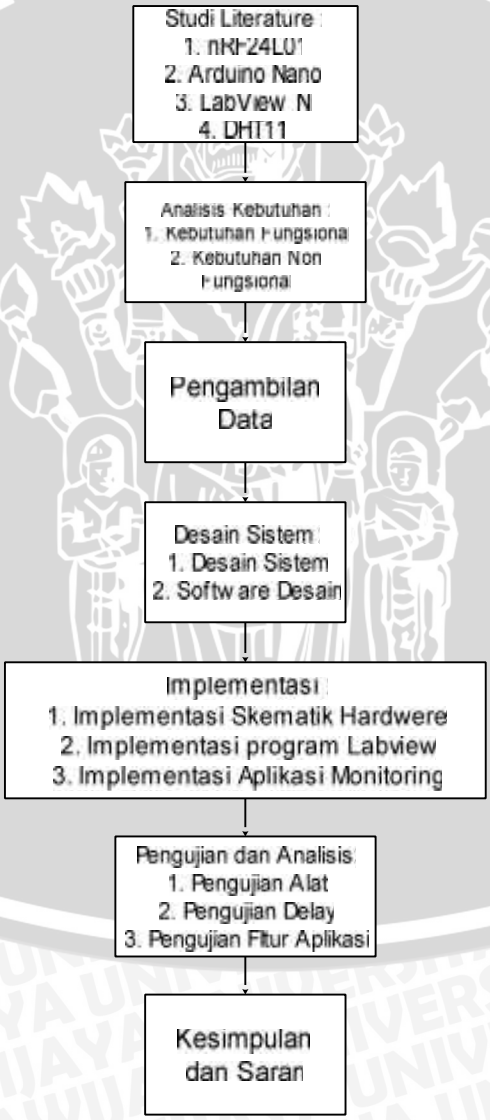
Fitur modul wireless nRF24L01:

a. Kecepatan operasi maksimum hingga 2Mbps, modulasi GFSK yang efisiensi, kemampuan Anti-gangguan, Sanizgat cocok untuk aplikasi kontrol industri.
b. *Built-in hardware* dan *link layer*
c. Respon otomatis dan fungsi transmisi otomatis
d. Wireless rate: 1 atau 2 Mbps
e. Antarmuka SPI rate: 0 ~ 8Mbps
f. 125 opsional saluran kerja
g. Saluran waktu *switching* yang sangat singkat, dapat digunakan untuk frekuensi hopping
h. Kompatibel dengan seri nRF24XX
i. I / O dapat menerima tingkat 5v input
j. 60ppm 16MHz kristal
k. Tegangan kerja: 1.9 ~ 3.6 V
l. Pita frekuensi ISM terbuka global, 0dBm ditransmisikan tegangan, bebas lisensi maksimum untuk menggunakan
m. Transmisi jarak hingga 100 meter di luar ruangan terbuka.

BAB 3 METODOLOGI

Pada bab ini menjelaskan metode yang digunakan dalam melakukan penelitian tentang sistem monitoring suhu dan kelembaban udara pada perkebunan menggunakan Arduino Nano dan LabView NI. Tipe penelitian ini adalah implementatif yaitu bersifat praktek langsung. Langkah pertama yang dilakukan adalah mengumpulkan teori-teori pendukung dan menggabungkannya ke dalam studi literatur. Kemudian dilanjutkan dengan proses analisis kebutuhan. Setelah analisis kebutuhan, proses selanjutnya adalah melakukan perancangan yang dilanjutkan dengan implementasi praktek sesuai dengan perancangan. Selanjutnya dilakukan pengujian dan analisis paa simulasi dan kemungkinan arah pengembangan selanjutnya.

Langkah-langkah yang dilakukan dalam penelitian ini ditunjukkan pada gambar 3.1.



Gambar 3.1 Diagram Alir Metode Penelitian

3.1 Studi Literatur

Studi literatur digunakan untuk menambah studi pustaka dan pengetahuan yang dilakukan untuk mengerjakan penulisan laporan dan penelitian. Studi literatur dilakasanakan dengan cara mengumpulkan teori dan pustaka yangberkaitan dengan penelitian ini meliputi:

1. Komunikasi radio frekuensi nRF24L01
2. Arduino Nano
3. Pemrograman LabView NI
4. DHT11

3.2 Analisis Kebutuhan

Analisis kebutuhan sistem akan mengidentifikasi kebutuhan yang akan digunakan sistem dalam membangun dan mengimplementasikan sistem tersebut. Kebutuhan yang digunakan harus sesuai dengan peran yang akan dijalankan dalam kerja sistem sehingga dapat mempermudah dalam perancangan. Berikut kebutuhan user dapat didefinisikan sebagai berikut:

1. User dapat memilih port yang berfungsi sebagai *input* dari Labview.
2. User dapat memasukkan *delay* untuk mengatur penampilan suhu dan kelembaban.
3. User dapat memasukkan maksimum dan minimum suhu.
4. User dapat memasukkan maksimum dan minimum kelembaban udara.

Pada analisis kebutuhan sistem, kebutuhan fungsional dan non fungsional harus dijelaskan. Kebutuhan fungsional yaitu menjelaskan apa saja yang dibutuhkan oleh sistem agar dapat berjalan dengan baik dan normal. Kemudian untuk kebutuhan non-fungsional menjelaskan apa saja yang menjadi batasan terhadap kebutuhan perancangan. Berikut kebutuhan fungsional dan non-fungsional :

Kebutuhan Fungsional:

Hardware:

1. 2 buah Arduino Nano
2. Sensor Suhu dan Kelembaban udara (DHT11).
3. 2 buah nRF24L01.
4. PC atau Laptop (untuk aplikasi pemrograman LabView NI).
5. Kabel penghubung antara sensor dengan Arduino dan nRF24L01
6. Kabel USB

Software:

1. LabView NI 2013.
2. NI Visa (Serial Komunikasi).
3. Arduino (untuk memrogram node server dan client).

Kebutuhan Non-Fungsional:

Umum:

1. Sistem sekurang-kurangnya dalam pemrosesan data (dari mendapatkan data suhu dan kelembaban sampai di tampilkan ke labview) hanya membutuhkan 2 detik (tergantung dari setiap delay yang diatur).
2. Sistem akan mulai tersedia saat user sudah menghidupkan alatnya.
3. Perangkat/alat yang di lapangan/ sebagai sensornya harus tertutup, tidak terlalu lembab, dan terdapat alat penyangga. (kondisional)

Khusus:

1. Harus dinyatakan dalam kondisi terbaik dan tersedia apabila system akan diingikan.
2. Tidak ada gangguan pada sensor.
3. Node client tidak boleh putus daya listrik apabila pada saat system berjalan.
4. PC atau Laptop harus menyala sebagai output tampilan pada Labview NI.

3.3 Pengambilan Data

Pengambilan data dilakukan dengan cara mengambil data *output* dari sensor suhu dan kelembaban udara (DHT11) dimana data pada sensor tersebut sudah dalam bentuk digital. Range sensor yaitu pada suhu dari 0 ^{o} C sampai 50^{o} C dan kelembaban udara dari 20% sampai 90%.

3.4 Desain Sistem

Pada desain sistem ini menjelaskan seperti apa sistem ini akan dibangun. Berikut merupakan blok sistem yang akan dibangun:

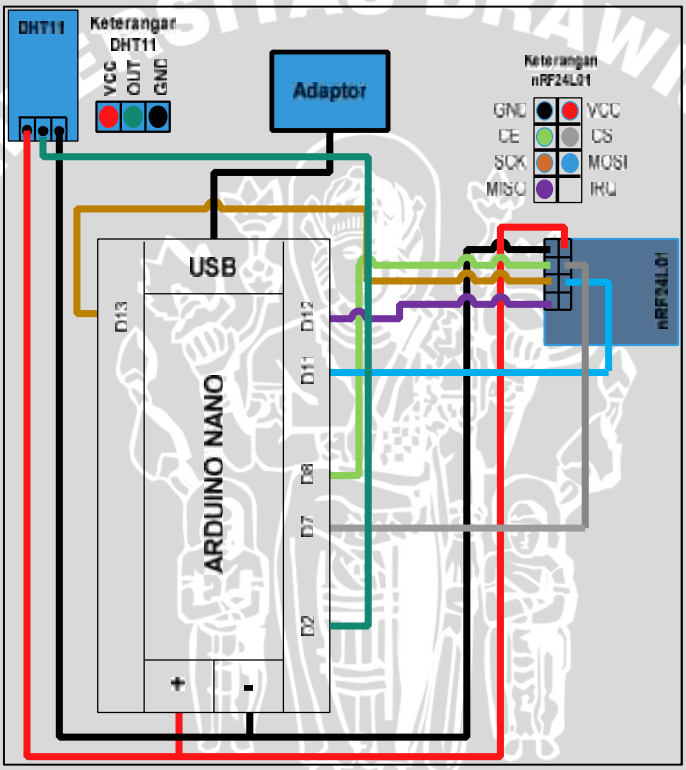
Gambar 3.2 Diagram Blok Perancangan Sistem

Penjelasan untuk diagram blok rancang sistem, terdiri dari node client, node server, komputer server dan database. Sistem yang akan dibangun berupa node client atau bisa disebut node sensor karena berfungsi sebagai sensor suhu dan kelembaban udara. Terdiri dari DHT11, Arduino nano, dan nRF24L01. Untuk prosesnya DHT11 mengindra suhu dan kelembaban udara sekitar kemudian diproses oleh Arduino nano sebagai mikrokontroler kemudian dikirim via *wireless* oleh nRF24L01 menuju ke node server. Diagram blok selanjutnya yaitu node server yang berfungsi menerima data dari node client yang terdiri dari Arduino nano sebagai mikrokontroler dan nRF24L01. Komputer server peneliti menggunakan laptop sebagai pemroses data melalui pemrograman LabView NI. Untuk memonitoring data suhu dan kelembaban yang berasal dari node client. Untuk database sendiri ini berfungsi sebagai media penyimpanan data suhu dan kelembaban.

Agar terciptanya sistem yang bagus dan bisa berjalan dengan lancar, diperlukan desain sistem yang lebih detail, terdapapat pembagian 2 sistem desain, yaitu sistem desain dan *software* desain. Pada sistem desain akan menjelaskan dan menggambarkan desain sistem secara *hardware*. Dan sedangkan sistem *software* akan menjelaskan dan menggambarkan desain sistem secara algoritma program pada sistem.

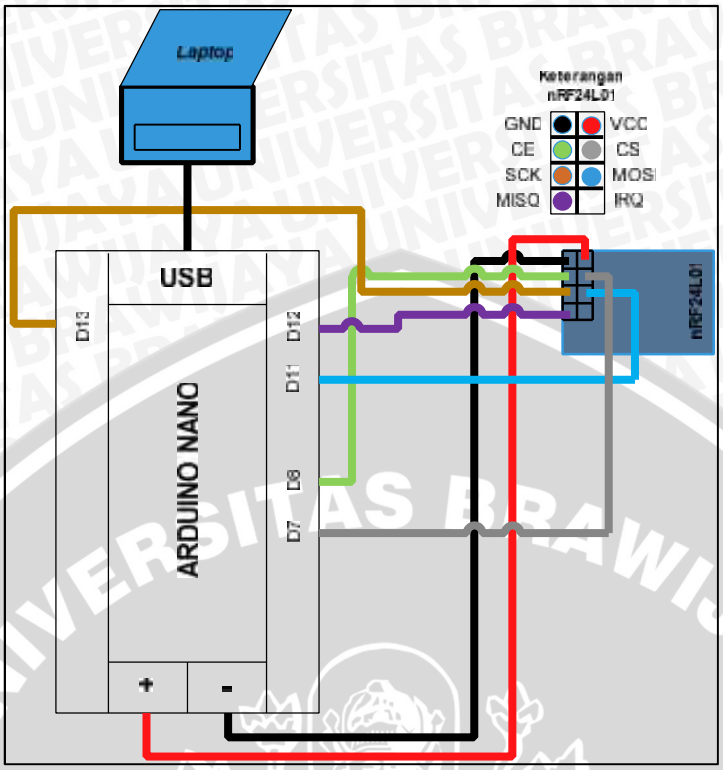
3.4.1 Desain Sistem

Sistem desain berkaitan dengan desain perangkat keras (*Hardware*) atau *prototype* dari sistem yang akan dibuat. Berikut terdapat 2 skematik dari sistem yaitu skematik pada node client dan skematik pada node server yang terhubung dengan laptop:



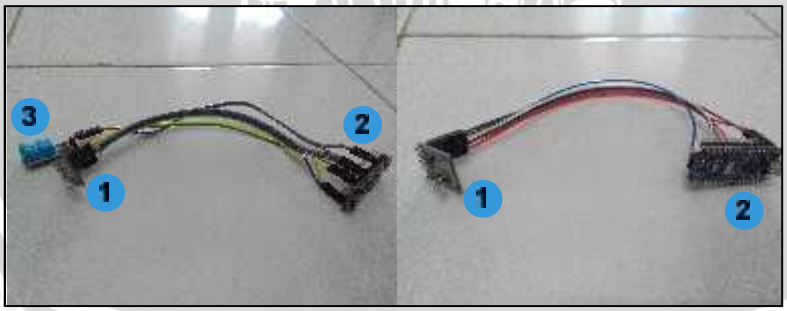
Gambar 3.3 Skematik Sistem pada Node Client

Skematik node client gambar 3.3, sensor DHT11 (suhu dan kelemababan udara) dihubungkan dengan pin digital pada mikrokontroler Arduino nano. Dan begitu juga dengan nRF24L01 dihubungkan pada pin digital mikrokontroler Arduino nano. Untuk daya listrik pada node client ini menggunakan adaptor atau *powerbank*.



Gambar 3.4 Skematik Sistem pada Node Server

Skematik node server gambar 3.4, sama halnya dengan node client hanya saja tanpa tersambung dengan adaptor dan DHT11. Karena pada node server ini hanya menerima data suhu dan kelembaban udara dari node client. Node server ini laptop berfungsi untuk mengolah data yang diterima dari node client. Data tersebut langsung diolah oleh pemrograman LabView NI. Sehingga bisa langsung tampil pada monitor laptop tersebut.



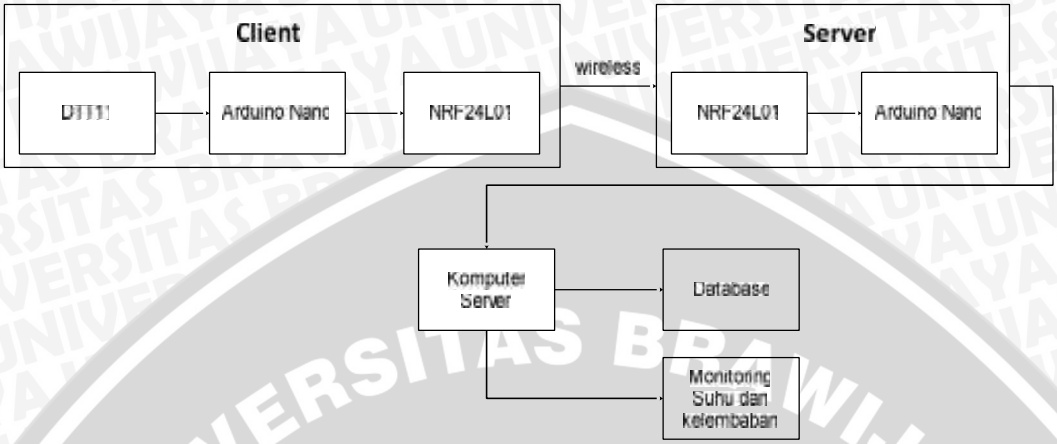
Gambar 3.5 Desain Sistem node server (kiri) dan node client (kanan)

Keterangan:

1. nRF24L01
2. Arduino Nano
3. DHT11

3.4.2 *Software* Desain

Software desain berkaitan dengan algoritma program yang ada pada sistem. Pada *software* desain pada sistem ini akan di buat sesuai kebutuhan yang telah di tentukan. Berikut merupakan diagram blok dari *software* desain:



Gambar 3.6 Diagram Blok *Software* Desain

Diagram blok pada gambar 3.6 *software* desain menjelaskan jalan alur dari program mulai dari node client dan server yang menggunakan pemrograman Arduino sampai menuju ke komputer server yang memakai pemrograman LabView NI. DHT11 akan mengindera suhu dan kelembaban sekitar untuk menghasilkan *output* dan akan diproses oleh arduuino nano pada node client. Kemudian akan dikirimkan via *wireless* melalui nRF24L01 menuju node server yang sudah di *setting* otomatis.

Pada node server akan menerima data dari node client. Kemudian pada Arduino akan menampilkan data tersebut melalu *Serial print* yang nantinya akan diproses oleh komputer *server* yang menggunakan pemrograman LabView NI. Data akan diproses dan ditampilkan melalui monitor dan apabila telah selesai sistem dapat menyimpan data suhu dan kelembaban udara.

3.5 Implementasi

Pada implemantasi sistem akan dilaksanakan sesuai dengan perancangan yang telah ditetapkan, mulai dari analisis kebutuhan sampai dengan kebutuhan desain. pada implementasi sistem ini memiliki beberapa tahapan yaitu:

1. Implementasi Skematik *Hardware* Sistem.
   Implementasi ini peneliti akan merancang system secara hardware dan penempatan kabel yang akan digunanakan.
2. Implementasi pemrograman pada Labview.
   Dalam implementasi ini peneliti akan membuat program menggunakan Labview NI sebagai Bahasa pemrograman yang berbasis *graphical programming.* Dimana program yang dibangun akan membacara *serial print* dari node server, sebagai aplikasi monitoring.
3. Implementasi pembuatan aplikasi monitoring suhu dan kelembaban udara.

Implementasi pada sistem ini yang sebagai *user interface* atau bisa disebut sebagai antarmuka pada system ini yang akan menampilkan monitoring suhu dan kelembaban udara

3.6 Pengujian dan Analisis Hasil

Pada pengujian dan analisi hasil ini akan dilakukan untuk mengetahui seberapa besar kinerja dan hasil dari sistem yang di buat ini. Berikut adalah beberapa pengujian yang akan dilaksanakan:

1. Pada sensor DHT11 akan diberi gangguan menggunakan korek api untuk merubah suhu dan kelembaban udara sekitar.
2. Pengujian *delay* pada node server untuk mengetahui akuisisi waktu tiap-tiap langkah-langkah program yang berjalan pada node client dan server.
3. Pengujian fitur-fitur pada aplikasi monitoring.

3.7 Penarikan Kesimpulan dan Saran

Kesimpulan didapatkan setelah melakukan perancangan, implementasi, pengujian dan analisis terhadap sistem. Isi dari kesimpulan dapat menjadi acuan pada penelitian lain yang akan mengembangkan sistem ini. Didalam penulisan akhir terdapat saran yang bertujuan untuk memberikan kemudahan kepada peneneliti sebelumnya, apabila akan meneruskan penelitian ini.

BAB 4 PERSYARATAN

Pada bab ini menjelaskan persyaratan minimal yang harus terpenuhi supaya tahap perancangan dan implementasi dapat berjalan dengan baik.

4.1 Analisa Permasalahan

Pada analisa permasalahan penguji akan menjelaskan kebutuhan fungsional secara *hardware* meliputi 2 buah Arduino Nano, Sensor Suhu dan Kelembaban udara (DHT11), 2 buah nRF24L01, PC atau Laptop (untuk aplikasi pemrograman LabView NI), Kabel jumper , Kabel USB pin 5, Korek api (untuk pengujian suhu dan kelembaban). Sensor suhu dan kelembaban udara (DHT11) sudah termasuk sensor digital sehingga dalam pemrograman di arduino hanya diberikan *library* dari DHT11.

Kemudian pada sub-bab tiga Analisi Kebutuhan, juga disebutkan kebutuhan secara software yang merupakan tahap pertama peneliti untuk memulai pemrograman pada LabView yang berbasis *graphical programing* dan pada Arduino sendiri. Pada bagian sub-bab tiga yaitu kebutuhan *software* harus di kerjakan secara urut agar peneliti bisa menyelesaikan pemrograman dalam pembuatan system ini.

4.2 Batasan Sistem

Sistem memiliki batasan dalam penelitian ini. Batasan ini merupakan hasil uji atau *testing* peneliti sebagai acuan untuk mengolah data agar bisa menghasilkan data yang maksimal. Berikut batasan sistem :

Khusus:

1. Dalam pengujian telemetri pada system masih terdapat RTO (*Request time out*) dimana banyak terdapat data *lost* dalam pengiriman data. Sehingga tidak berfokus pada pengujian telemetri karena terkendala oleh alat.
2. Untuk pengujiannya hanya dilakukan dalam ruangan.

Sensor DHT11:

1. Range pada sensor DHT11 pada kelembaban udara yaitu 20 -90 % RH dan pada suhu yaitu 0 – 50 ^{o}C.

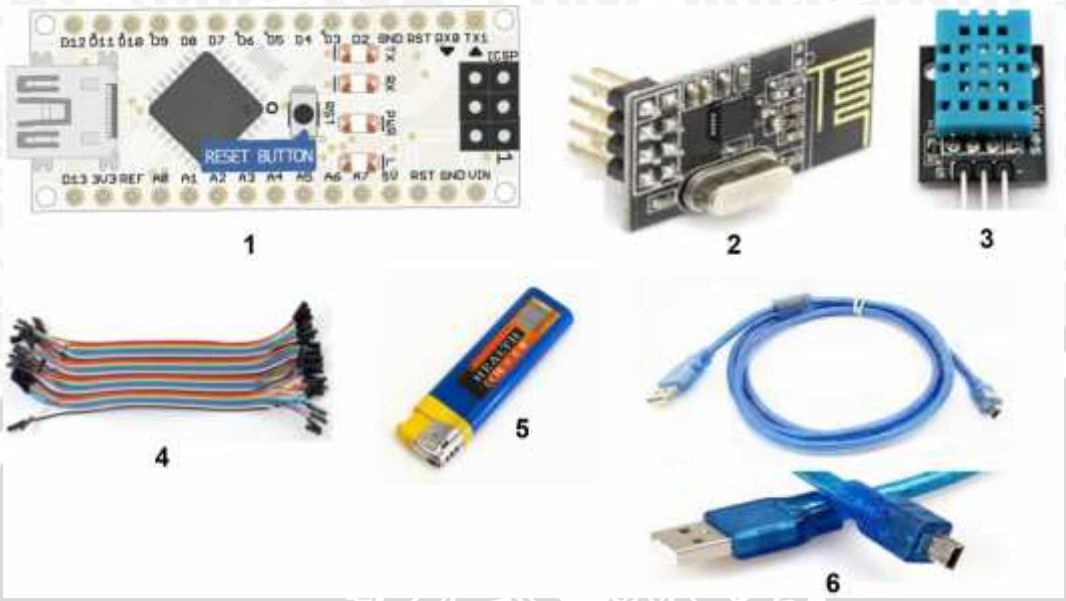
Output:

1. *Output* pada sistem yaitu tampilan suhu dan kelembaban udara, grafik, dan batasan maksimum dan minimum suhu dan kelembaban.

4.3 Persyaratan *Hardware*

Pada sub-bab empat analisa permasalahan telah dijelaskan bahwa permasalahan *hardware* meliputi poin kebutuhan terhadap *hardware* yang telah dirangkum pada sub-bab tiga analisis kebutuhan *hardware.* Berikut kebutuhan *hardware* yang harus terpenuhi sebelum tahap perancangan dan implementasi dilaksanankan:

1. Arduino nano
2. nRF24L01
3. Sensor suhu dan kelembaban (DHT11)
4. Kabel jumper
5. Kabel usb pin 5



Gambar 4.1 Kebutuhan Instalasi *Hardware*

Penjelasan gambar 4.1 yaitu 1. Arduino nano, 2. nRF24L01, 3. DHT11, 4. Kabel Jumper, 5. Korek api, 6. Kabel usb pin 5. Pada tahap pertama dalam instalasi kebutuhan hardware ialah mengtahui jenis Arduino nano. Penelitian ini peneliti menggunakan Arduino Nano yang papan mikrokontrolernya Atmel ATmega328 . Ini memiliki 14 digital pin *input* / *output* (dimana 6 dapat digunakan sebagai *output* PWM), 8 *input analog*, *resonator on-board*, tombol reset, dan lubang untuk pemasangan *pin header*.

Tahap kedua yaitu modul *wireless network*. Peneliti menggunakan nRF24L01. nRF24L01 memiliki keterbatasan masalah jarak pengiriman maupun penerimaan data, yaitu hanya memiliki jangkauan 100 m tanpa halangan.

Kemudian untuk tahap selanjutnya sensor suhu dan kelembaban yaitu DHT11. Sensor sudah bekerja secara digital yang dapat memberikan 2 *input* sekaligus yaitu suhu dan kelembaban udara. Peneliti akan memberikan ketelitian data suhu dan kelembaban udara sampai sebesar 2 angka dielakang koma (contoh : suhu = 25,00 dan kelembaban udara = 66,00). Menggunakan korek api sebagai alat penguji untuk mempengaruhi perubahan suhu dan kelembaban.

Alat berikutnya adalah kabel jumper dan kabel usb pin5. Dimana kabel jumper sebagai penghubung nRF24L01 dan DHT11 ke Arduino nano, sedangkan untuk kabel usb pin 5 sebagai peghubung Arduino ke laptop.

4.4 Persyaratan *Software*

Pada sub-bab empat analisa permasalahan telah dijelaskan bahwa permasalahan software meliputi poin kebutuhan sistem terhadap *software* yang telah di rangkum pada sub-bab tiga Analisis kebutuhan *software.* Berikut kebutuhan *software* yang harus terpenuhi sebelum tahap perancangan dan implementasi dilaksanakan:

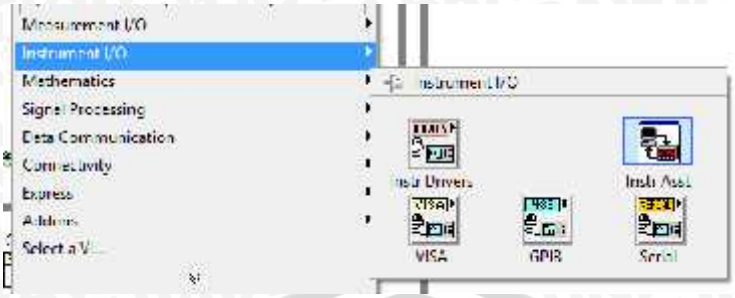
1. LabView NI 2013
2. NI VISA (Serial komunikasi)
3. IDE Arduino

Gambar 4.2 Gambar Kebutuhan *Software*

Tahap pertama dalam instalasi kebutuhan *software* ialah mengistal LabView NI 2013. Tujuan dalam instalasi LabView NI 2013 adalah sebagai memprogram aplikasi monitoring pada sistem.

Gambar 4.3 LabView NI 2013

Tahab kedua yaitu menginstal NI VISA sebagai aplikasi yang berperan sebagai serial komunikasi LabView NI dengan Arduino nano. Karena LabView yang digunakan peneliti adalah LabView NI 2013 maka peneliti bisa menggunakan versi NI VISAnya 503. Jika sudah terinstal NI VISA maka akan terapat tampilan seperti ini :



Gambar 4.4 Fitur dari NI VISA

Tahap yang terakhir peneliti harus menyiapkan *software* Arduino. Disini peneliti menggunakan versi 1.0.5-r2. Peneliti menggunakan IDE Arduino untuk membuat sketch node client dan node server. Dimana sketch ini sangat berpengaruh dengan sistem ini. IDE yang digunakan peneliti hanya langsung pakai saja tidak perlu mengistal, karena pada web Arduino serndri menyediakan IDE Arduino yang portable. Berikut adalah gambaran dari IDE Arduino itu sendiri :

Gambar 4.5 IDE Arduino

Setelah seluruh tahap persyaratan diatas terpenuhi maka sistem siap dirancang dan diimplementasikan sesuai dengan kebutuhan Sistem yang telah didefinisikan pada bab sebelum-sebelumnya.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada bab ini akan dijelaskan metode perancangan berdasarkan *hardware* dan *software* serta proses pengimplementasiannya dari seluruh perancangan yang telah dibuat oleh peneliti.

5.1 Perancangan dan Implementasi *Hardware*

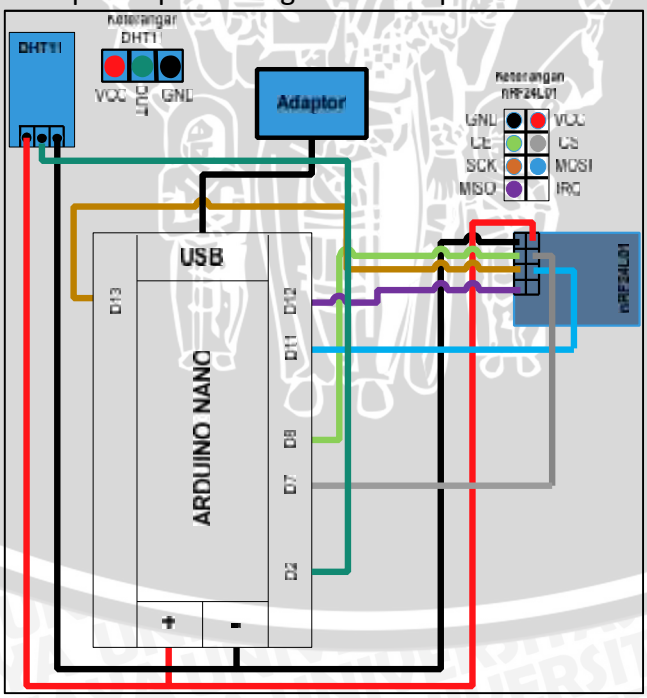
Dalam perancangan *hardware* akan dijelaskan proses perancangan yang terkait dengan sistem *hardware* atau perangkat keras, baik itu sistem penempatan sensor, rangkaian elektrik sistem agar dalam pengimplementasiannya dapat berjalan sesuai dengan harapan peneliti. Pengimplementasian *hardware* akan dijelaskan hasil implementasi yang terkait dengan sistem *hardware* atau perangkat keras, baik itu sistem penempatan sensor, rangkaian elektrik sistem, maupun *prototype* sistem berdasarkan hasil perancangan yang telah peneliti buat sebelumnya.

5.1.1 Perancangan Rangkaian Elektrik

Perancangan rangkaian elektrik sistem monitoring komponen utama dari system ini adalah:

1. Rangkaian elektrik sensor suhu dan kelembaban udara (DHT11)
2. nRF24L01
3. Arduino Nano

Dibawah ini merupakan perancangan elektrik pada node client.



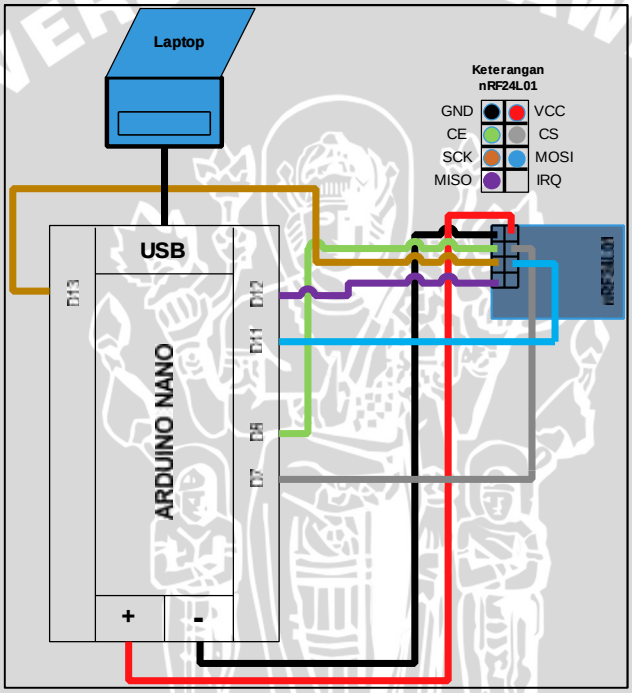
Gambar 5.1 Perancangan Rangkaian Elektrik pada Node Client

Pada gambar 5.1, rangkaian elektrik utama berupa nRF24L01 dan DHT11. Pada rangkaian nRF24L01 terdapat 8 pin, tetapi yang dipakai hanya 7 pin. Diantaranya pin VCC, GND, CE, CS, SCK, MOSI, dan MISO. Pin VCC akan

dihubungkan ke pin 3,3v, pin GND akan dihubungkan ke pin GND, pin CE akan dihubungkan dengan pin D6, pin CS akan dihubungkan dengan pin D7, pin SCK akan dihubungkan dengan pin D13, pin MOSI akan dihubungkan dengan pin D11 dan pin MISO akan dibuhungkan dengan pin D12.

Kemudian pada rangkaian sensor suhu dan kelembaban udara (DHT11) terdapat 3 pin yang akan digunakan, yaitu pin VCC, OUT, GND. Diantaranya pin VCC akan dihubungkan dengan pin 5v pada Arduino, pin OUT akan dihubungkan dengan pin D2 arduino, pin GND akan dihubungkan dengan pin GND pada Arduino. Pada pin USB bisa dihubungkan ke usb laptop/komputer atau bisa di hubungkan ke adaptor sebagai daya listriknya.

Selanjutnya pada gambar di bawah ini akan menjelaskan perancangan elektrik pada node server.



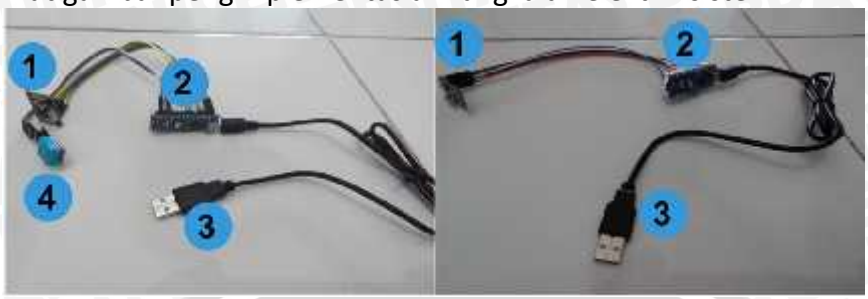
Gambar 5.2 Perancangan Rangkaian Elektrik pada Node Server

Pada gambar 5.1, rangkaian elektrik utama berupa nRF24L01. Sama seperti node client pada rangkaian nRF24L01 terdapat 8 pin, tetapi yang dipakai hanya 7 pin. Diantaranya pin VCC, GND, CE, CS, SCK, MOSI, dan MISO. Pin VCC akan dihubungkan ke pin 3,3v, pin GND akan dihubungkan ke pin GND, pin CE akan dihubungkan dengan pin D6, pin CS akan dihubungkan dengan pin D7, pin SCK akan dihubungkan dengan pin D13, pin MOSI akan dihubungkan dengan pin D11 dan pin MISO akan dibuhungkan dengan pin D12.

Kemudian pada pin USB harus terhubung dengan port USB pada laptop/komputer karena sebagai penghubung antara Arduino nano dengan laptop/komputer.

5.1.2 Implementasi Rangkaian Elektrik

Berikut gambar pengimplementasian rangkaian elektrik sistem:



Gambar 5.3 implementasi Rangkaian Elektrik Sistem pada Node Client (Kiri) dan Node Server (Kanan)

Pada gambar 5.8, implementasi rangkaian elektrik menggunakan Arduino nano, nRF24L01, DHT11, dan kabel USB pin 5. Pada node client menggunakan Arduino nano, nRF24L01, DHT11, dan kabel USB pin 5. Sedangkan pada node server menggunakan Arduino nano, nRF24L01, dan kabel USB pin 5.

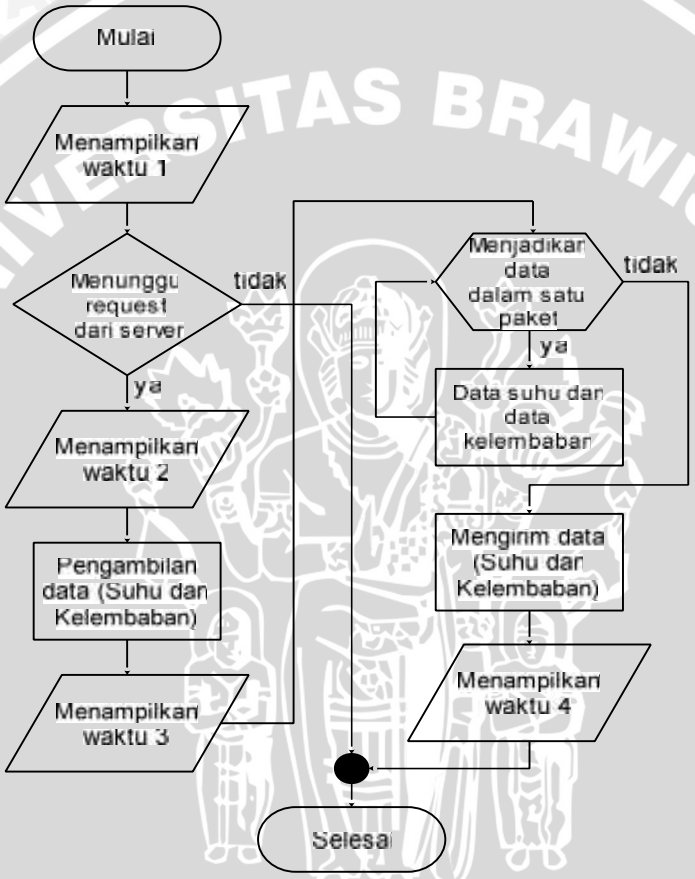
5.2 Perancangan dan Implementasi *Software*

Pada sub-bab ini akan dijelaskan beberapa perancangan serta implementasinya mengenai cara kerja sistem yang akan dibuat, baik itu berupa algoritma program sampai tahap akhir pembuatan sistem tersebut.

5.2.1 Perancangan Sketch untuk Node Client dan Node Server

5.2.1.1 Perancangan pada Node Client

Pada sub-bab ini merupakan tahap pembuatan algoritma untuk node client dan node server. Adapun kebutuhan yang akan diimplementasikan adalah sebagai berikut:



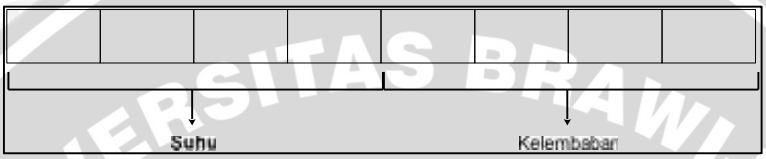
Gambar 5.4 Flowchart Perancangan pada Node *Client*

Pada gambar 5.4, merupakan flowchart perancangan untuk node client berawal dari menampilkan waktu 1 sebagai waktu acuan mulainya program. Selanjutnya node client menunggu request/permintaan dari node server. Apabila tidak terdapat request/permintaan dari server maka program akan berhenti dan mengulang kembali ke awal. Apabila terdapat request maka akan melanjutkan ke step berikutnya yaitu menampilkan waktu 2. Selanjutnya akan terjadi pengambilan data suhu dan kelembaban. Kemudian menampilkan waktu 3. Karena terdapat dua data yaitu suhu dan kelembaban udara, maka peneliti menjadikan data tersebut menjadi satu menggunakan metode struct, dimana struct tersebut adalah kumpulan data yang memiliki tipe data yang berbeda.

Setelah kemudian data dikirim. Setelah mengirim akan menampilkan waktu 4. Fungsi dari pemwaktuan adalah akan digunakan pada bab pengujian dan analisis.

Gambar 5.5 Data Receive pada node client

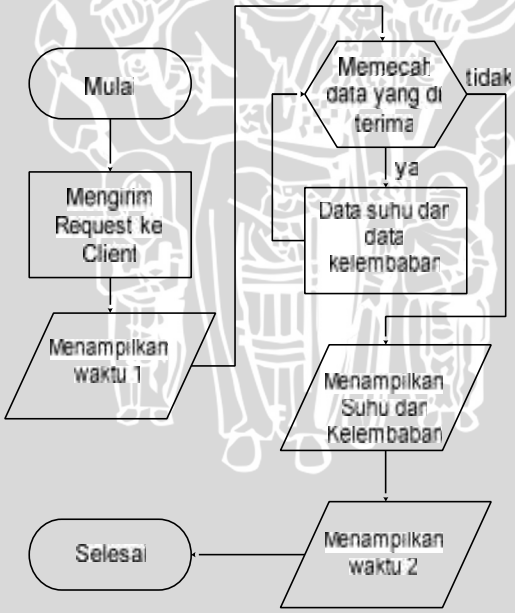
Pada gambar 5.5 menjelaskan paket data dalam bentuk array yang diterima oleh node client dari node server apabila pada array 0 nilainya 0 dan array 1 nilainya 1 maka akan masuk pada step berikutnya. Apabila nilainya tidak sama dengan yang dijelaskan diatas maka akan berhenti dan akan kembali ke awal.



Gambar 5.6 Data Send pada node client

Pada gambar 5.6 mejelaskan paket data dalam bentuk array yang dikirim oleh node client menuju ke node server. Pada 4 nilai array pertama adalah data suhu dan 4 nilai array berikutnya adalah kelembaban.

5.2.1.2 Perancangan pada Node Server untuk Pengujian Delay

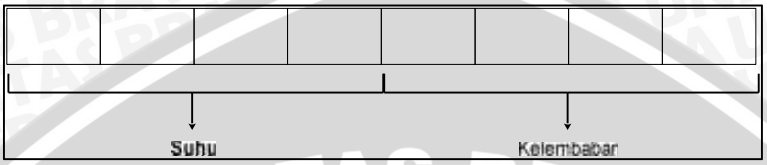


Gambar 5.7 Flowchart Perancangan pada Node Server untuk Pengujian *Delay*

Pada gambar 5.7, merupakan flowchart perancangan untuk node server dimana berawal dari mengirim request/permintaan ke client, kemudian menampikan waktu 1, apabila data yang diminta sudah tersedia maka pada step berikutnya data akan di pecah menjadi 2. Yaitu data suhu dan kelembaban. Setelah data di pecah maka akan di tampilkan sebagai suhu dan kelembaban. Dan yang terakhir menampilkan waktu 2. Sama halnya seperti node client pemwaktuan akan digunakan pada bab pengujian dan analisis.

Gambar 5.8 Data Send pada node server

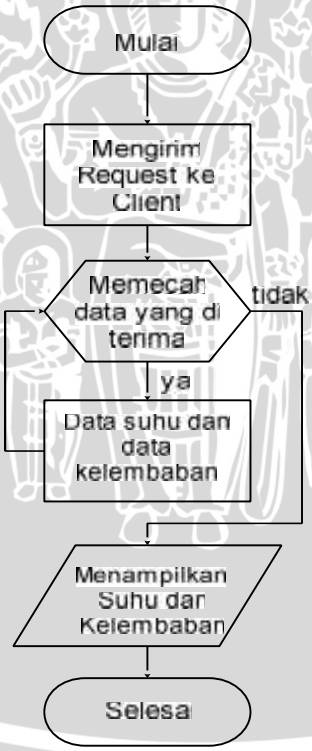
Pada gambar 5.8 menjelaskan paket data dalam bentuk array yang dikirim oleh node server menuju node client apabila pada array 0 nilainya 0 dan array 1 nilainya 1 maka akan masuk pada step berikutnya. Apabila nilainya tidak sama dengan yang dijelaskan diatas maka akan berhenti dan akan kembali ke awal.



Gambar 5.9 Data Receive pada node server

Pada gambar 5.9 mejelaskan paket data dalam bentuk array yang diterima oleh node server dari node client. Pada 4 nilai array pertama adalah data suhu dan 4 nilai array berikutnya adalah kelembaban.

5.2.1.3 Perancangan pada Node Server untuk Pengujian Aplikasi Monitoring



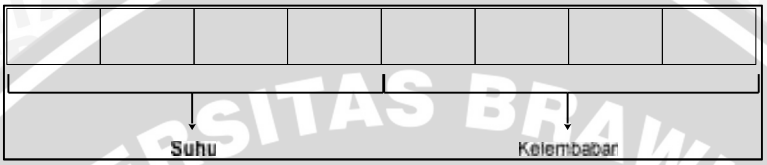
Gambar 5.10 Flowchart Perancangan pada Node *Server* untuk Pengujian Aplikasi Monitoring

Pada gambar 5.10, merupakan flowchart perancangan untuk node server dimana berawal dari mengirim request/permintaan ke client, kemudian data

dipecah menjadi dua data yaitu data suhu dan kelembaban. Setelah itu akan ditampilkan.

Gambar 5.11 Data Send pada node server

Pada gambar 5.11 menjelaskan paket data dalam bentuk array yang dikirim oleh node server menuju node client apabila pada array 0 nilainya 0 dan array 1 nilainya 1 maka akan masuk pada step berikutnya. Apabila nilainya tidak sama dengan yang dijelaskan diatas maka akan berhenti dan akan kembali ke awal.



Gambar 5.12 Data Receive pada node server

Pada gambar 5.12 mejelaskan paket data dalam bentuk array yang diterima oleh node server dari node client. Pada 4 nilai array pertama adalah data suhu dan 4 nilai array berikutnya adalah kelembaban.

5.2.2 Implementasi Sketch untuk Node Client dan Node Server

5.2.2.1 Implementasi pada Node Client

Berikut merupakan hasil perancangan dari flowchart yang menghasilkan sketch untuk node Client.

```
#include "DHT.h"
#define dhtpin 2
#define dhttype DHT11
#include <SPI.h> //library spi
#include <Mirf.h> //library komunikasi untuk nrf24l01
#include <nRF24L01.h> //library komunikasi untuk nrf24l01
#include <MirfHardwareSpiDriver.h> //library komunikasi spi
DHT dht (dhtpin, dhttype, 6);

byte DataReceive[2];
byte DataSend[8];

union{ // memakai struck untuk suhu
  float suhu;
  byte b [4];
}data1;

union{ // memakai struck untuk Kelembaban
  float kelembaban;
  byte b [4];
}data2;

void setup(){
  Serial.begin(9600);
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  dht.begin();
  Mirf.setRADDR((byte *)"sens1");
  Mirf.payload = 8;
  Mirf.channel = 105;
```

```
31    Mirf.config();
32
33    millis();
34    Serial.print("Time 1:");
35    Serial.println(millis());
36  }
37
38  void loop(){
39    if(!Mirf.isSending() && Mirf.dataReady()){
40      Mirf.getData((byte *)&DataReceive);
41      if(DataReceive[0]==0&&DataReceive[1]==1){
42
43    Serial.print("Time 2:");
44    Serial.println(millis());
45
46    data1.suhu = dht.readTemperature();
47    data2.kelembaban = dht.readHumidity();
48    Serial.print("T=");
49    Serial.print(data1.suhu);
50    Serial.print(", ");
51    Serial.print("H=");
52    Serial.println(data2.kelembaban);
53
54    Serial.print("Time 3:");
55    Serial.println(millis());
56
57    for (int i = 0; i < 4 ; i++){ // menjadikan dalam satu paket
58      DataSend[i] = data1.b[i];
59      DataSend[i+4] = data2.b[i];
60    }
61    Mirf.setTADDR((byte *)"serve");
62    Mirf.send((byte *)&DataSend);
63
64    Serial.print("Time 4:");
65    Serial.println(millis());
66    Serial.println("");
67    }
68    }
69  }
```

Gambar 5.13 Implementasi dari Node *Client*

Penjelasan tiap-tiap jalannya program akan dijelaskan pada gambar berikut :

```
1  #include "DHT.h"
2  #define dhtpin 2
3  #define dhttype DHT11
4  #include <SPI.h> //library spi
5  #include <Mirf.h> //library komunikasi untuk nrf24l01
6  #include <nRF24L01.h> //library komunikasi untuk nrf24l01
7  #include <MirfHardwareSpiDriver.h> //library konunikasi spi
8  DHT dht (dhtpin, dhttype, 6);
```

Gambar 5.14 Library pada Node *Client*

Pada gambar 5.14 merupakan library yang dipakai dalam Node Client. Pada node client memakai library DHT11 dan nRF24L01. Dimana fungsi library itu sendiri sebagai unutk mengawali sebuah sketch pada IDE Arduino.

```
byte DataReceive[2];
byte DataSend[3] ;

union{ // memakai struck untuk suhu
  float suhu;
  byte b [4];
}data1;

union{ // memakai struck untuk Kelembaban
  float kelembaban;
  byte b [4];
}data2;
```

Gambar 5.15 Pendeklarasian *Type data* dan *Struck*

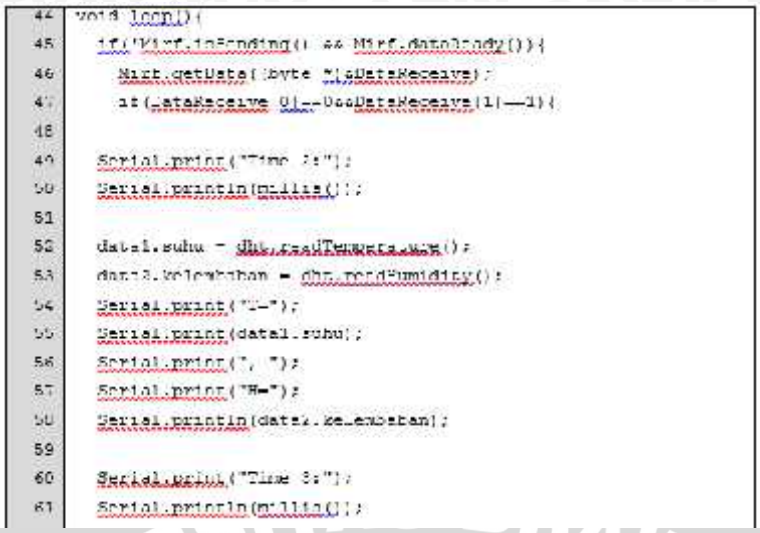
Pada gambar 5.15, merupakan pendeklarasian type data dan pemakaian struck pada Arduino. Yang di deklarasikan adalah untuk DataReceive dan DataSend. Dideklarasikan dalam bentuk aray. Kemudian untuk struck,nya menggunakan 2 struck, struck untuk suhu dan kelembaban. Mengapa menggunakan struck pada sketch ini, karena data yang diambil dalam bentuk fload yang memiliki dua angka dibelakang koma. Kemudian dirubah kedalam bentuk byte data.

```
void setup(){
  Serial.begin(9600);
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  dht.begin();
  Mirf.setRADDR((byte *)"sens1");
  Mirf.payload = 8;
  Mirf.channel = 105;
  Mirf.config();

  millis();
  Serial.print("Time 1:");
  Serial.println(millis());
}
```

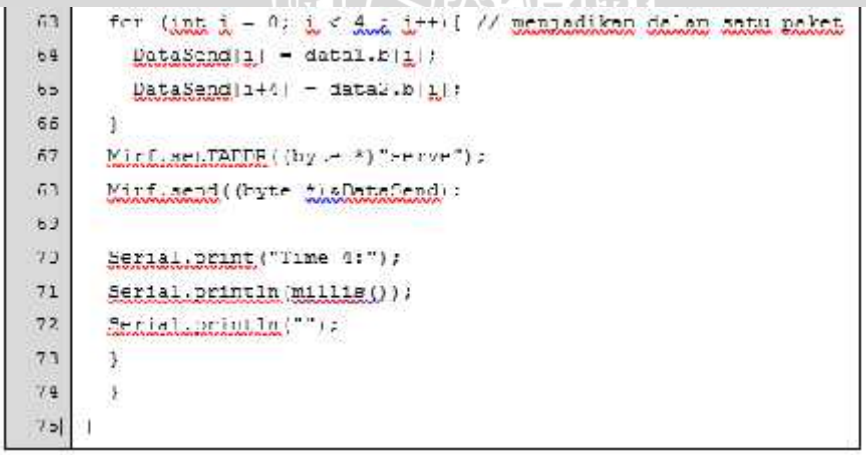
Gambar 5.16 *Void Setup*

Pada gambar 5.16, merupakan bagian program void setup dari node client. Berfungsi sebagai salah satu fungsi yang hanya satu kali eksekusi ketika awal program berjalan. Pada bagian void setup node client ini berisi baudrate dari Arduino nano yaitu 9600, jenis komunikasi nRF24L01, mulai penggunaan nRF24L01 dan DHT, mengatur nama node client sebagai sens1, mengatur panjang data maksimal yang di terima maupun yang dikirim, mengatur chanel yang dipakai adalah 105, mengkonfigurasi nRF24L01, yang terakhir adalah menampilkan millis yang pertama (waktu dari system Arduino itu sendiri).



Gambar 5.17 *Void Loop*

Pada gambar 5.17, merupakan bagian program node client yaitu void loop. Dimana berfungsi yang akan di eksekusi terus menerus saat program ini berjalan. Pada bagian awal void loop menjelaskan apabila Mirf tidak sedang mengirim data dan data sudah siap maka akan mengeksekusi mengambil data yang diterima. Dan data tersebut akan difilter lagi, apabila DataReceive[0]==0&&DataReceive[1]==1 maka akan mengeksekusi program dibawahnya. Selanjutnya akan menampilkan millis yang ke dua. Pada subprogram berikutnya data sensor temperature akan dinamakan dan disimpan sementara pada struck data1. Dan untuk data kelembaban akan dinamakan dan disimpan pada struck data2. Kemudian akan ditampilkan data suhu dan kelembaban. Milis ke 3 akan ditampilkan lagi.



Gambar 5.18 Lanjutan dari Subprogram *Void Loop*

Pada gambar 5.18, merupakan lanjutan sub program void loop dari node client. Perulangan pada sub program diatas adalah untuk memasukkan nilai suhu dan kelembaban yang belum bersatu sebelumnya. Data data1.b adalah variable struck dari suhu dan data2.b dari kelembaban. Setelah disatukan kedalam variable DataSend yang memiliki nilai 8byte kemudian mengatur alamat pengiriman meuju serve. Setelah itu akan menampilkan millis ke empat. Milis

pada program ini berfungsi untuk mengetahui delay dari sub program yang berjalan.

5.2.2.2 Implementasi pada Node Server untuk Pengujian Delay

Berikut merupakan hasil perancangan dari flowchart yang menghasilkan sketch node Server untuk pengujian delay.

```
#include <SPI.h> //library spi
#include <Mirf.h> //library komunikasi untuk nrf24l01
#include <nRF24L01.h> //library komunikasi untuk nrf24l01
#include <MirfHardwareSpiDriver.h> //library komunikasi spi

union{ // memakai struck untuk suhu
  float suhu;
  byte b [4];
}data1;

union{ // memakai struck untuk Kelembaban
  float kelembaban;
  byte b [4];
}data2;

byte DataReceive[8];
byte DataSend[2];

void setup(){
  Serial.begin(9600);
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  Mirf.setRADDR((byte *)"serve");
  Mirf.payload = 8;
  Mirf.channel = 105;
  Mirf.config();
}

void loop(){
  DataSend[0]=0;
  DataSend[1]=1;
  unsigned long time = millis();
  Mirf.setTADDR((byte *)"sens1");
  Mirf.send((byte *)&DataSend);

  Serial.print("Time 1:");
  Serial.println(millis());

  while(Mirf.isSending()){
  }

  delay(10);
  while(!Mirf.dataReady()){
    if ( ( millis() - time ) > 1000 ) {
      Serial.println("RTO!");
      Serial.println(" ");
      return;
    }
  }

  Mirf.getData((byte   *)&DataReceive);  //  memecah  data
menjadi
   for (int i = 0; i < 4 ; i++){
    data1.b[i] = DataReceive[i] ;
    data2.b[i] = DataReceive[i+4] ;
  }
  Serial.print("T=");
  Serial.print(data1.suhu);
```

```
59    Serial.print(", ");
60    Serial.print("H=");
61    Serial.println(data2.kelembaban);
62    Serial.print("Time 2:");
63    Serial.println(millis());
64    Serial.println(" ");
65    delay (1000);
    }
```

Gambar 5.19 Implementasi dari Node *Server* untuk Pengujian *Delay*

Penjelasan tiap-tiap jalannya program akan dijelaskan berikut ini :

```
1  #include <SPI.h> //library spi
2  #include <Mirf.h> //library komunikasi untuk nrf24l01
3  #include <nRF24L01.h> //library komunikasi untuk nrf24l01
4  #include <MirfHardwareSpiDriver.h> //library komunikasi spi
5
6  union{ // memakai struck untuk suhu
7    float suhu;
8    byte b [4];
9  }data1;
10
11 union{ // memakai struck untuk Kelembaban
12   float kelembaban;
13   byte b [4];
14 }data2;
15
16 byte DataReceive[8];
17 byte DataSend[2];
```

Gambar 5.20 Gambar *library* pada Node *Server* untuk Pengujian *Delay*

Pada gambar 5.20, merupakan library dari node server untuk pengujian delay. Yang membedakan dari node client pada librarynya adalah library DHT11nya. Dimana pada node Client ini tidak menggunakan library DHT11 hanya menggunakan library nRF24L01 saja. Pada deklarasi strucknya juga sama, karena pada node client juga menggunakan variable Struck juga. Kemudian untuk deklarasi variable DataReceive dan DataSend juga berbeda panjang datanya. Untuk DataReceive memiliki panjang datanya 8byte. Dan DataSendnya memiliki 2byte. Karena pada node server menerima data yang panjangnya 8byte dari node client. Dan hanya mengirim 2byte menuji node client.

```
19 void setup(){
20   Serial.begin(9600);
21   Mirf.spi = &MirfHardwareSpi;
22   Mirf.init();
23   Mirf.setRADDR((byte *)"serve");
24   Mirf.payload = 8;
25   Mirf.channel = 105;
26   Mirf.config();
27 }
```

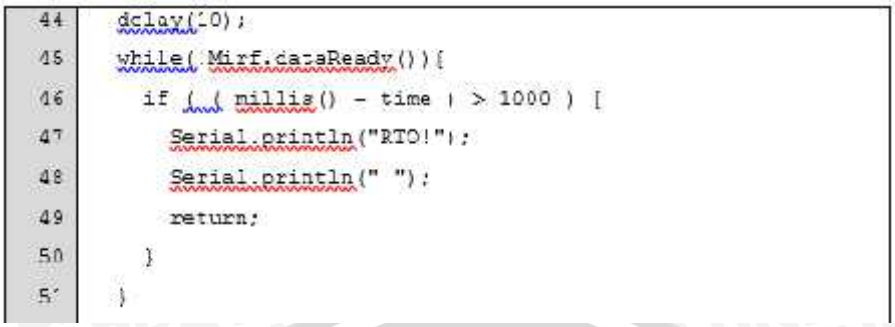
Gambar 5.21 *Void Setup* pada Node *Server* untuk Pengujian *Delay*

Pada gambar 5.21, merupakan sub program dari node server untuk pengujian, yaitu void setup. Sekilas serti apa yang kita lihat hampir sama dengan node client, tetapi yang membedakan haya tidak ad configurasi DHT11nya. Pada void setup ini hanya terdapat configurasi baudrate, SPI (komunikasi arduinonya), inisialisasi node yaitu serve, panjang data yang digunakan 8byte, menggunakan *channel* 105 dan configurasi nRF24L01.

```
29 void loop(){
30   DataSend[0]=0;
31   DataSend[1]=1;
32   unsigned long time = millis();
33   Mirf.setTADDR((byte *)"sens1");
34   Mirf.send((byte *)&DataSend);
35
36   Serial.print("Time 1:");
37   Serial.println(millis());
38
39   while(Mirf.isSending()){
40   }
```

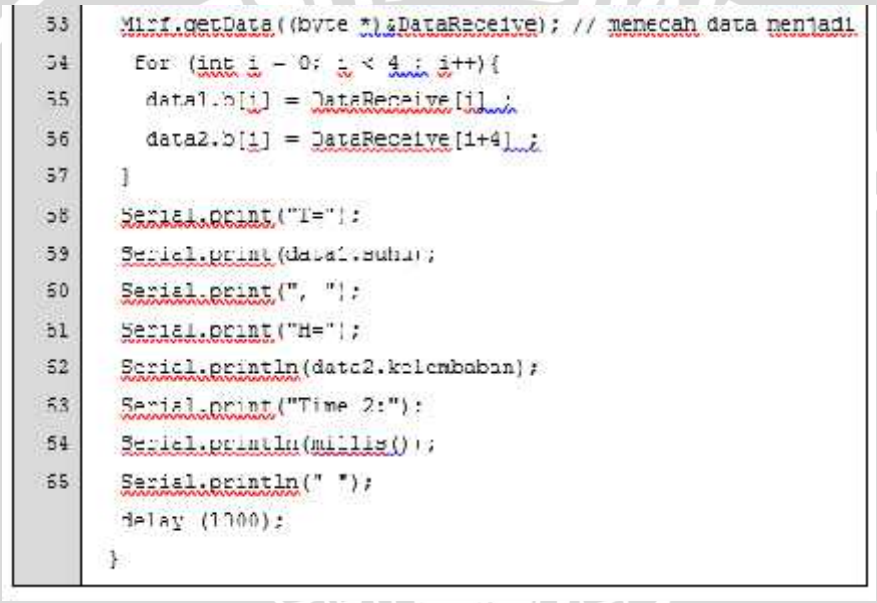
Gambar 5.22 *Void Loop* pada Node *Server* untuk Pengujian *Delay*

Pada gambar 5.22, merupakan void loop dari node server untuk pengujian delay. Pada void loop akan dieksekusi terus menerus sesuai pada program yang telah diatur pada node server. Awal void loop diatas menjelaskan untuk pengalamatan pada node client dengan mengatur DataSend[0]=0 dan DataSend[1]=1. Kemudian deklarasi variable time menjadi unsigned long dimana sama dengan millis pada node server itu sendiri. Mengatur pengalamatan DataSend menuju ke sens1. Dan setelah itu menampilkan millis pertama.



Gambar 5.23 Lanjutan dari *Void Loop* pada Node *Client* untuk Pengujian *Delay*

Pada gambar 5.23, merupakan lanjutan dari sub program void loop, dimana berfungsi sebagai mengunggu data kiriman dari node client agar tidak terjadi paket loss.



Gambar 5.24 Lanjutan dari *Void Loop* pada Node *Client* untuk Pengujian *Delay*

Pada gambar 5.24, merupakan sub program lanjutan pada node client untuk pengujian delay. Pada program diatas menjelaskan bahwa DataReceive di proses untuk mendapatkan data suhu dan kelembaban. Dengan cara menggunakan perulangan. Karena data suhu berada di 4byte pertama dan data kelembaban di 4byte terakhir. Maka 4byte pertama di masukkan ke variable data1.b sebagai data suhu dan 4byte terakhir dimasukkan ke variable data2.b sebagai kelembaban. Kemudian akan ditampilkan melalui serial monitor. Dan setelah itu akan menampilkan millis ke dua.

5.2.2.3 Implementasi pada Node Server untuk Pengujian Aplikasi Monitoring

Berikut merupakan hasil implementasi flowchart yang menghasilkan sketch node Server untuk pengujian aplikasi monitoring.

```
1  #include <SPI.h> //library spi
2  #include <Mirf.h> //library komunikasi untuk nrf24l01
3  #include <nRF24L01.h> //library komunikasi untuk nrf24l01
4  #include <MirfHardwareSpiDriver.h> //library komunikasi spi
5
6  union{ // memakai struck untuk suhu
```

```
  float suhu;
  byte b [4];
}data1;

union{ // memakai struck untuk Kelembaban
  float kelembaban;
  byte b [4];
}data2;

byte DataReceive[8];
byte DataSend[2];

void setup(){
  Serial.begin(9600);
  Mirf.spi = &MirfHardwareSpi;
  Mirf.init();
  Mirf.setRADDR((byte *)"serve");
  Mirf.payload = 8;
  Mirf.channel = 105;
  Mirf.config();
}

void loop(){
  DataSend[0]=0;
  DataSend[1]=1;
  unsigned long time = millis();
  Mirf.setTADDR((byte *)"sens1");
  Mirf.send((byte *)&DataSend);

  while(Mirf.isSending()){
  }

  delay(10);
  while(!Mirf.dataReady()){
    if ( ( millis() - time ) > 1000 ) {
      Serial.println("RTO!");
      Serial.println(" ");
      return;
    }
  }

  Mirf.getData((byte  *)&DataReceive);  //  memecah  data
menjadi
   for (int i = 0; i < 4 ; i++){
    data1.b[i] = DataReceive[i] ;
    data2.b[i] = DataReceive[i+4] ;
  }
  Serial.print("T=");
  Serial.print(data1.suhu);
  Serial.print(", ");
  Serial.print("H=");
  Serial.println(data2.kelembaban);
  Serial.println(" ");
  delay (1000);
 }
```

Gambar 5.25 Implementasi dari Node *Server* untuk pengujian Aplikasi Monitoring

Pada gambar 5.19, menjelaskan selebihnya sama persis dengan sketch pada node server untuk Pengujian Delay, hanya saja penampilan millis pertama dan kedua tidak ditampilkan, karena pada pengujian aplikasi monitoring hanya membaca serial monitor pada Arduino.

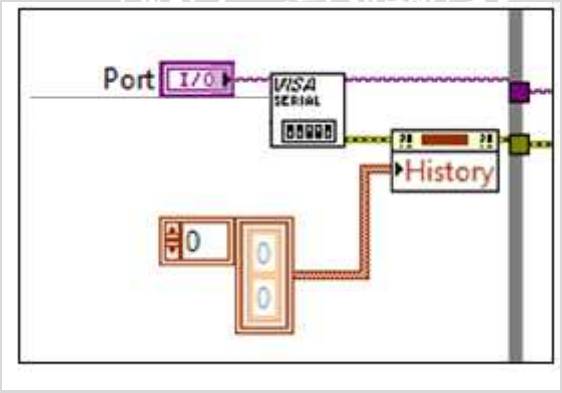
5.3 Implementasi Fitur Program

Pada bab ini akan dijelaskan implementasi dari fitur program yang secara garis besar memiliki fungsi unutk menunjang program agar berjalan dengan baik sesuai kebutuhan peneliti.

5.3.1 Implementasi Tampilan Grafik Suhu dan Kelembaban

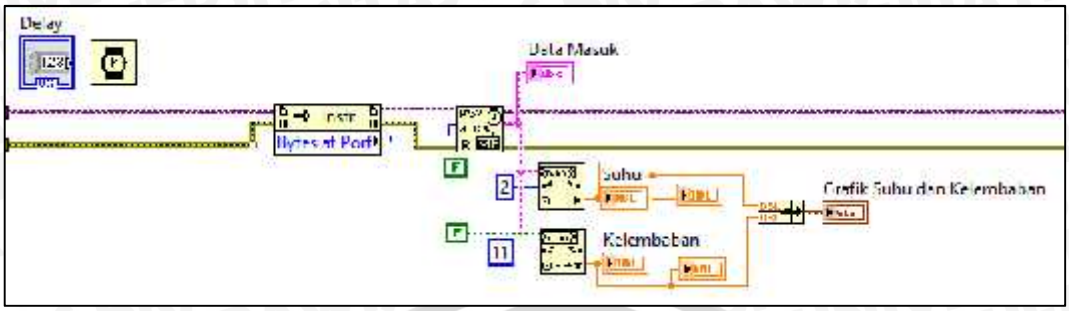
Gambar 5.26 Implementasi Tampilan Grafik Suhu dan Kelembaban

Pada gambar 5.26, adalah tampilan utama untuk menampilkan suhu dan kelembaban udara dalam bentuk grafik. Selain menampilkan suhu dan kelembaban udara, pada gambar diatas juga terdapat tombol-tombol untuk mengatur aplikasi monitoring, yaitu penentuan Port USB, tampilan Data Masuk, pengaturan Delay dan tombol stop.



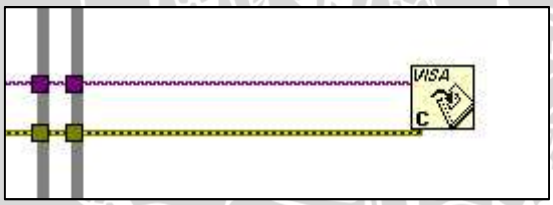
Gambar 5.27 Program *Clear History*

Pada gambar 5.27 adalah potongan program yang digunakan untuk mengatur Port dan *clear history*. Fungsi dari kotak *History* diatas adalah mereset ulang semua data pada tampilan monitoring, yang awalnya terdapat nilainya maka akan kosong kembali setelah ditekan tombol start pada aplikasi monitoring.



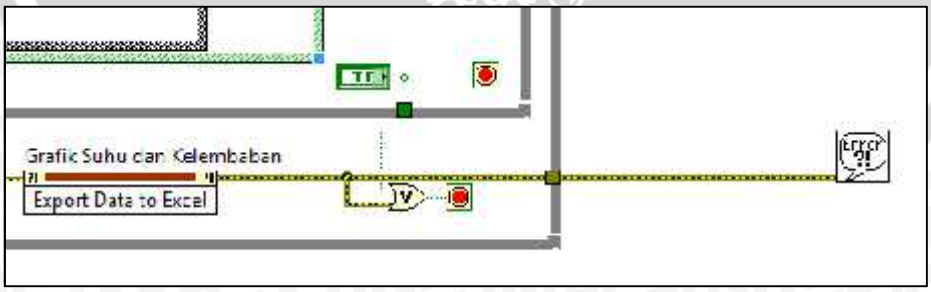
Gambar 5.28 Program Pembacaan dan Menampilkan Grafik Suhu dan Kelembaban

Pada gambar 5.28 adalah potongan program dari aplikasi monitoring yang berfungsi untuk mengolah data dari port USB menjadi data yang bisa ditampilkan ke monitor. Data dari node server dibaca melalui port USB menggunakan VISA *Read* kemudian akan ditampilkan data dari port USB tersebut ke Kolom Data Masuk. Untuk memisah data yang masi menjadi satu, menggunakan Fract/Exp String To Number Function. Pada fungsi program tersebut untuk menampilkan data suhu harus memberikan batasan bahwa setelah huruf ke dua dari Data Masuk, kemudian untuk data kelembaban setelah huruf ke 11 dari Data Masuk. Kemudian akan ditampilkan menjadi grafik yang sebelumnya termasuk data string dirubah menjadi data double.



Gambar 5.29 Program Penutupan VISA

Pada gambar 5.29 adalah penutup dari fungsi VISA. Pada kabel/*wire* yang berwarna ungu adalah VISA resource name yang berfungsi sebagai data masuk dari VISA *configuration* sebelumnya. Dan yang berwarna kuning adalah error in yang berfungsi untuk pengecekan error pada system ini yang telah menjadi satu paket pada VISA *configuration.*



Gambar 5.30 Program untuk Menyimpan Data Grafik ke Excel

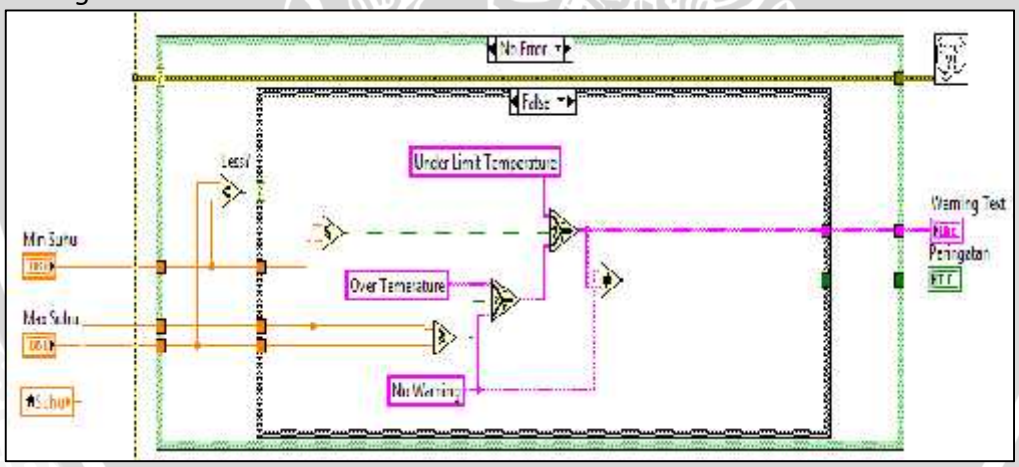
Pada gambar 5.30 adalah program untuk menyimpan plot data grafik dalam bentuk nilai angka yaitu pada *invoke node* yang bernama “Export Data to Excel”. *Invoke node* akan dieksekusi apabila *while loop* yang paling dalam telah selesai. Degan cara menekan tombol stop (Kotak Boolean yang berwarna hijau). Dan

seperti biasa memberikan *Simple Error Handler.vi* untuk menindikasikan terdapatnya *error* atau tidak.

5.3.2 Implementasi Setting Maksimum dan Minimum Suhu

Gambar 5.31 Implementasi *Setting* maksimum dan minimum suhu

Pada gambar 5.31 merupakan tampilan untuk mengatur maksimum dan minimum suhu. Apabila suhu terlalu tinggi dari suhu max maka pada kotak Warning Text akan muncul "*Over Temperature*" dan apabila suhu terlalu rendah dari min suhu maka akan muncul "*Under Limit Temperature*". Dan apabila suhu masi dalam range antara min suhu dengan max suhu maka akan muncul "*No Warning*".



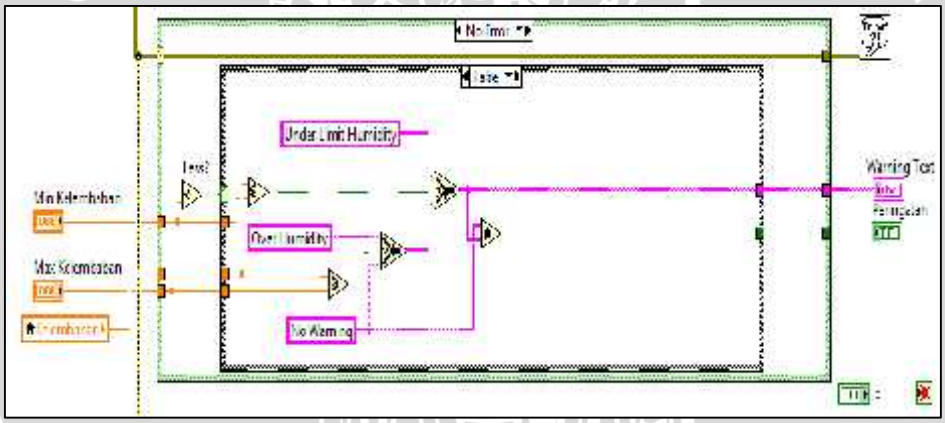
Gambar 5.32 Program *Setting* Maksimum dan Minimum Suhu

Pada gambar 5.32 adalah program untuk pengaturan maksimum dan minimum suhu. Pada potongan program diatas nilai suhu diatas adalah Local Variabel. "*Under Limit Temperature*" akan muncul pada kolom *Warning Text* apabila Min Suhu lebih rendah sama dengan Suhu. Dan "*Over Temperature*" akan muncul pada kolom *Warning Text* apabila Max Suhu Lebih besar sama dengan Suhu. Untuk "*No Warning*" apabila nilai dari suhu berada diantara Min Suhu dan Max Suhu. Untuk tanda peringatan apabila terjadi "*Under Limit Temperature*" dan "*Over Temperature*" maka akan menyala. Dan apabila tidak terdapat kedua keadaan tersebut maka tidak akan menyala.

5.3.3 Implementasi Setting Maksimum dan Minimum Kelembaban

Gambar 5.33 Implementasi *Setting* Maksimum dan Minimum Kelembaban

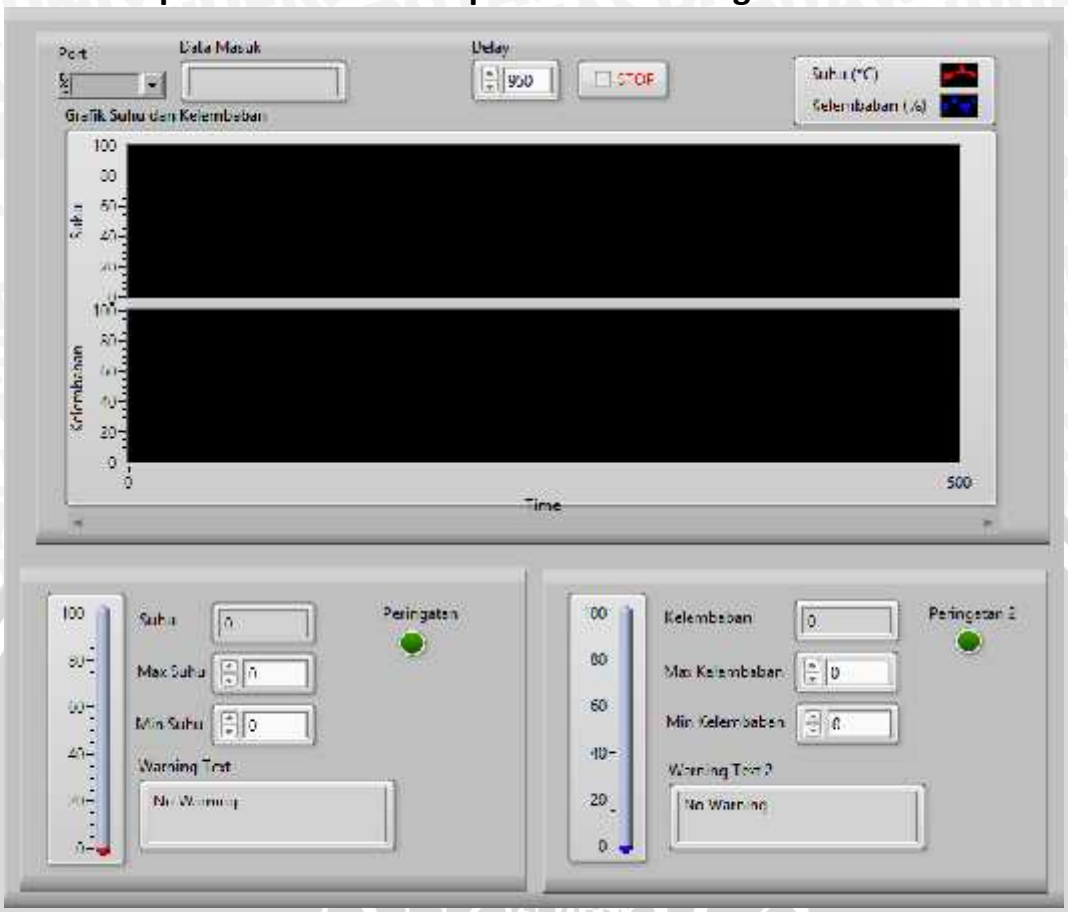
Pada gambar 5.33 merupakan tampilan untuk mengatur maksimum dan minimum kelembaban. Apabila kelembaban terlalu tinggi dari kelembaban max maka pada kotak Warning Text akan muncul "*Over Humidity*" dan apabila kelembaban terlalu rendah dari min kelembaban maka akan muncul "*Under Limit Humidity*". Dan apabila kelembaban masih dalam range antara min kelembaban dengan max kelembaban maka akan muncul "*No Warning*".



Gambar 5.34 Program *Setting* Maksimum dan Minimum Kelembaban

Pada gambar 5.34 adalah program untuk pengaturan maksimum dan minimum kelembaban. Pada potongan program diatas nilai kelembaban diatas adalah Local Variabel. "*Under Limit Humidity*" akan muncul pada kolom *Warning Text* apabila Min kelembaban lebih rendah sama dengan kelembaban. Dan "*Over Humidity*" akan muncul pada kolom *Warning Text* apabila Max kelembaban Lebih besar sama dengan Suhu. Untuk "*No Warning*" apabila nilai dari kelembaban berada diantara Min kelembaban dan Max kelembaban. Untuk tanda peringatan apabila terjadi "*Under Limit Humidity*" dan "*Over Humidity*" maka akan menyala. Dan apabila tidak terdapat kedua keadaan tersebut maka tidak akan menyala.

5.3.4 Tampilan Keseluruhan Aplikasi Monitoring



Gambar 5.35 Tampilan Keseluruhan Aplikasi Monitoring

Pada gambar 5.35 adalah tampilan keseluruhan dari aplikasi monitoring suhu dan kelembaban. Data suhu dan kelembaban akan muncul dalam bentuk angka dan grafik. Dan untuk suhu dan kelembabannya juga terdapat *Warning Box* yang berfungsi untuk memantau nilai data dari suhu dan kelembaban. Dengan memberikan batasan maksimum dan minimum. Dan apabila ditekan tombol stop maka data plot grafik akan disimpan dalam excel.

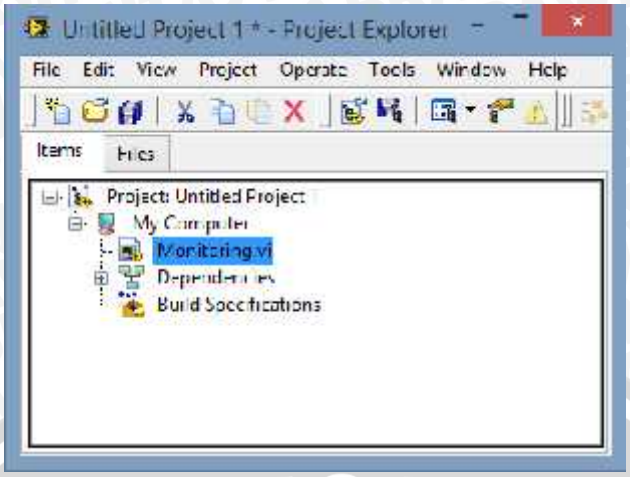
5.4 Implementasi Pembuatan Installer Aplikasi Monitoring

Dalam implementasi ini, peneliti membuat hasil akhir system berupa installer aplikasi yang diharpkan akan membantu user lain unutk mengaplikasikan system ini. Dalam proses pembuatan installer aplikasi monitoring ini memiliki beberapa tahap yaitu:

1. Pembuatan aplikasi monitoring *executable*
2. Build *installer*
3. Cara *install* dan *Uninstall* program

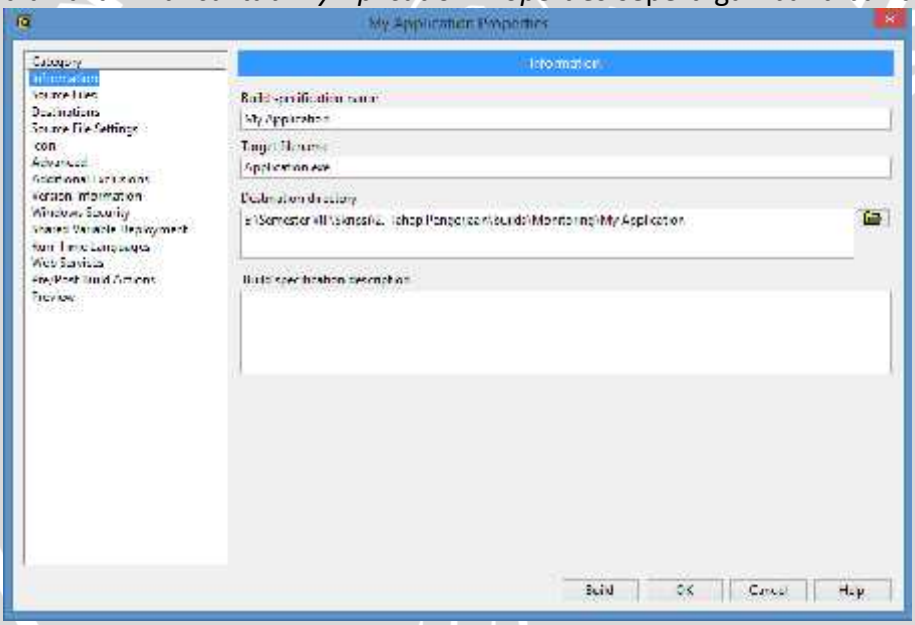
5.4.1 Pembuatan Aplikasi *Executable*

Pada tahap ini akan membuat file aplikasi executable yang sebelumnya harus membuat file project baru dari LabView NI.



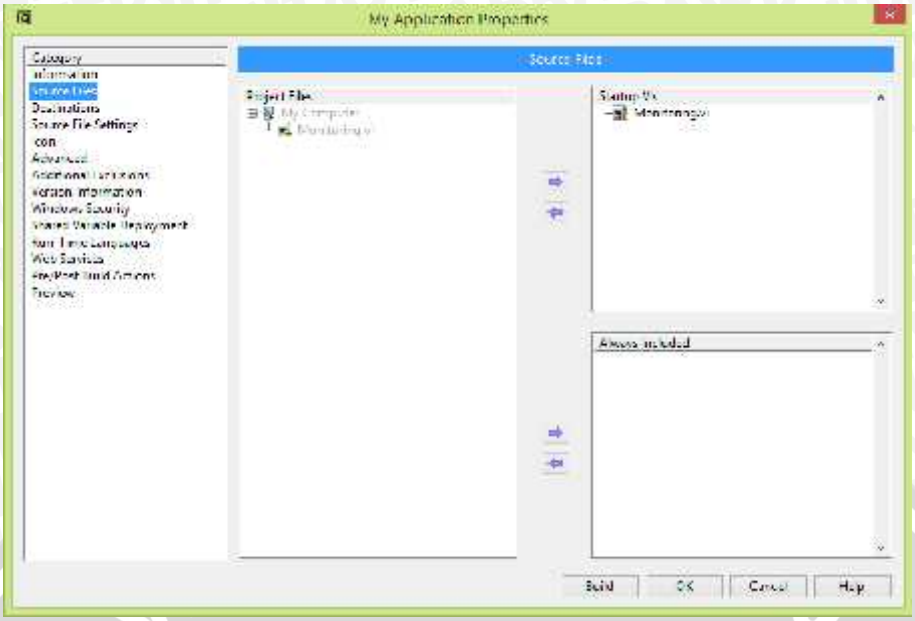
Gambar 5.36 Pembuatan *Project* Baru

Pada gambar 5.36 adalah tampilan pembuatan project baru sebelum pembuatan file *executable.* Untuk pembuatan monitoring.exe dapat dilakukan dengan klik kanam pada *Build Specifications* dan pilih *new* -> *Aplication*(EXE). Kemudian akan muncul tab *My Aplication Properties* seperti gambar dibawah ini.



Gambar 5.37 *My Tab Application Properties*

Pada gambar 5.37 adalah tab yang digunakan peneliti untuk mengatur *build executable* sesuai dengan keinginan perneliti, baik itu nama aplikasi, tempat penyimpanan hasil *build* dan lainnya.



Gambar 5.38 Pembuatan file EXE

Pada gambar 5.38 merupakan tahap tahap lanjutan, untuk menjadikan file.exe, pilihlah *Source File* kemudian pilih Monitoring.vi masukkan kedalam *Startup Vis*. Setelah itu pilihlah *Build* setelah itu Monitoring.exe sudah jadi.

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini akan dilakukan proses pengujian dan analisis dari system yang telah dibuat. Pengujian ini dilakukan untuk memastikan system secara *functional* dapat berjalan dengan baik. Selain itu pengujian pada system ini adalah menguji *delay* setiap proses program pada sketch Arduino dari node client sampai ke node server.

6.1 Pengujian Alat

Pengujian alat pada sub bab ini adalah pengujian pada Arduino nano, DHT11 dan nRF24L01 yang sudah dirangkai menjadi satu. Dan pengujian ini dimaksudkan untuk mengetahui apakah alat ini bisa berjalan lancar atau tidak.

6.1.1 Skenaro Pengujian

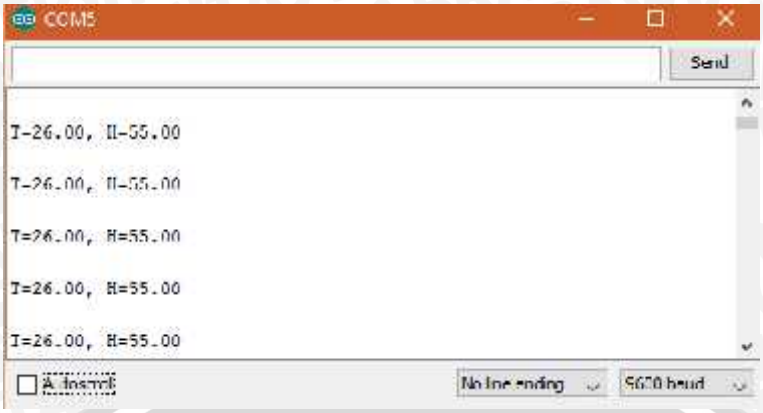
Skenaro pengujian ini akan dilakukan pengujian dasar pada alat yang dimaksudkan untuk mengecek apakah alat sudah bisa digunakan atau tidak. Pengujian dilakukan dengan cara melakukan *request* dan *receive* data pada node client dan node server. Berikut dibawah ini adalah ping_client yang *compile* pada node client yang merupakan contoh dalam *request* dan *receive* data. Pengujian ini menggunakan scenario jarak kurang lebih 1,5 meter.

6.1.2 Hasil Pengujian

Dalam pengujian pada alat (Adruino, DHT11, nRF24L01) pada skenario pengujian 6.1.1, didapatkan hasil sebagai berikut.

Gambar 6.1 Hasil Pengujian Ping_Client

Pada gambar 6.1 diatas merupakan hasil pengujian ping_client untuk memastikan bahwa node client bisa digunakan dan menampilkan data dari sensor.



Gambar 6.2 Hasil Pengujian Ping_Server

Pada gambar 6.2 diatas merupakan hasil pengujian *ping_server* untuk memastika bahwa *node server* bisa digunakan dan dapat menampilkan data sensor dari node client.

6.1.3 Analisis Hasil

Hasil pengujian 6.1.2 yaitu pengujian apakah alat bisa digunakan dengan lancar atau tidak. Disimpulkan bahwa alat (Arduino nano, nRF24L01) dapat digunakan dengan baik, karena pada pengiriman datanya lancar dan datanya sudah benar dengan apa yang diterima dan di kirim.

6.2 Pengujian *Delay* pada Node Server dan Node Client

Pengujian ini dilakukan untuk mengetahui *delay* tiap – tiap sub program dari tiap – tiap node. Karena untuk mengetahui keefektifan waktu pengambilan data sampai ditampilkan ke monitor.

6.2.1 Skenario Pengujian

Skenario pengujian ini, peneliti akan memasukkan fungsi penampilan waktu kedalam node client dan node server, seperti yang sudah dijelaskan dalam perancangan dan implementasi software. Pada pengujian ini menggunakan *delay* untuk menguji, yaitu menggunakan *delay* 0,5 detik; 1 detik; 1,5 detik yang diletakkan pada node server di akhir program yang bertujuan untu mengetahui selisih waktu / delay pada sub-sub program.

Berikut merupakan penempatan *delay* pada program node client dan node server :

a. Node Client

```
23  void setup(){
24    Serial.begin(9600);
25    Mirf.spi = &MirfHardwareSpi;
26    Mirf.init();
27    dht.begin();
28    Mirf.setRADDR((byte *)"sens1");
29    Mirf.payload = 8;
30    Mirf.channel = 105;
31    Mirf.config();
32
33    millis();
34    Serial.print("Time 1:");
35    Serial.println(millis());
36  }
```

Gambar 6.3 Menampilkan Time 1

Pada gambar 6.3 merupakan penempatan *delay* untuk menampilkan Time 1/ *delay* antara program akhir void setup dengan awal program void loop.

```
44  void loop(){
45    if(!Mirf.isSending() && Mirf.dataReady()){
46      Mirf.getData((byte *)&DataReceive);
47      if(DataReceive[0]==0&&DataReceive[1]==1){
48
49    Serial.print("Time 2:");
50    Serial.println(millis());
51
52    data1.suhu = dht.readTemperature();
53    data2.kelembaban = dht.readHumidity();
54    Serial.print("T=");
55    Serial.print(data1.suhu);
56    Serial.print(", ");
57    Serial.print("H=");
58    Serial.println(data2.kelembaban);
```

Gambar 6.4 Menampilkan Time 2

Pada gambar 6.4 merupakan penempatan *delay* untuk menampilkan Time 2/delay antara proses penerimaan data dari node server dengan penampilan data suhu dan kelembaban.

```
54    Serial.print("T=");
55    Serial.print(data1.suhu);
56    Serial.print(", ");
57    Serial.print("H=");
58    Serial.println(data2.kelembaban);
59
60    Serial.print("Time 3:");
61    Serial.println(millis());
62
63    for (int i = 0; i < 4 ; i++){ // menjad
64      DataSend[i] = data1.b[i];
65      DataSend[i+4] = data2.b[i];
66    }
```

Gambar 6.5 Menampilkan Time 3

Pada gambar 6.5 merupakan penempatan *delay* untuk menampilkan Time 3/*delay* antara proses penampilan data suhu dan kelembaban dengan proses pengiriman data suhu dan kelembaban menuju ke node server.

```
67    Mirf.setTADDR((byte *)"serve");
68    Mirf.send((byte *)&DataSend);
69
      Serial.print("Time 4:");
      Serial.println(millis());

      Serial.println("");
      }
    }
  }
```

Gambar 6.6 Menampilkan Time 4

Pada gambar 6.6 merupakan penempatan *delay* untuk menampilkan Time 4/*delay* antara proses pengiriman data menuju node server dengan akhir program node client.

b. Node Server

```
44  void loop(){
45    DataSend[0]=0;
46    DataSend[1]=1;
47    unsigned long time = millis();
48    Mirf.setTADDR((byte *)"sens1");
49    Mirf.send((byte *)&DataSend);
50
51    Serial.print("Time 1:");
52    Serial.println(millis());
53
54    while(Mirf.isSending()){
55    }
```

Gambar 6.7 Menampilkan Time 1

Pada gambar 6.7 merupakan penempatan *delay* untuk menampilkan Time 1/*delay* antara proses pengiriman data menuju node client dengan proses menunggu untuk menerima data dari node client.

```
      Serial.print("T=");
      Serial.print(data1.suhu);
      Serial.print(", ");
      Serial.print("H=");
      Serial.println(data2.kelembaban);
      Serial.print("Time 2:");
      Serial.println(millis());
      Serial.println(" ");
```

Gambar 6.8 Menampilkan Time 2

Pada gambar 6.8 merupakan penempatan *delay* untuk menampilkan Time 2/*delay* antara proses penampilan data suhu dan kelembaban dengan akhir program.

6.2.2 Hasil Pengujian

Dengan pengujian *delay* pada node client dan node server didapatkan hasil sebagai berikut:

6.2.2.1 Dengan *Delay* 0,5 detik atau 500 ms

a. Node Client

Tabel 6.1 Node Client

Eksekusi Program ke-	Waktu ke – 2 (ms)	Waktu ke – 3 (ms)	Waktu ke – 4 (ms)
1	1114	1386	1387
2	1890	1891	1891
3	2402	2403	2404
4	2915	2916	2917
5	3428	3701	3701
6	4204	4206	4206
7	4717	4718	4719
8	5230	5231	5232
9	5742	6014	6016
10	6518	6519	6520

Pada table 6.1 menjelaskan tentang *tracking time* (pencatatan waktu) pada node client dengan menggambil hanya sampai ke sepuluh kali eksekusi program. Hasil dari pengujian diatas semakin naik menyesuaikan dengan *internal timing* di Arduino nano pada node client.

Delay tiap 1 kali eksekusi program pada node client :

Tabel 6.2 Selisih waktu pada Node Client

Eksekusi Program ke-	Selisih waktu ke 2 dan 3 (ms)	Selisih waktu ke 3 dan 4 (ms)	Selisih waktu ke 4 dan 2 pada eksekusi program berikutnya (ms)
1	272	1	503
2	1	0	511
3	1	1	511
4	1	1	511
5	273	0	503
6	1	0	511
7	1	1	511
8	1	1	510
9	272	2	502
10	1	1	-
Rata - rata	82,4	0,8	457,4

Pada table 6.2 menjelaskan tentang *delay* (selisih waktu) di node client dari *tracking time* pada table 6.1. Dari penghitungan rata-rata didapatkan, yaitu : 1. *Delay* ke 1 dan 2 = 82,4 ms; 2. *Delay* ke 2 dan 3 = 0,8 ms; 3. *Delay* ke 3 dan 1 (eksekusi progam berikutnya) = 457,4 ms.

b. Node Server

Tabel 6.3 N**ode Server**

Eksekusi program ke -	Waktu ke – 1 (ms)	Waktu ke – 2 (ms)
1	0	277
2	778	790
3	1291	1302
4	1803	1814
5	2315	2592
6	3093	3105
7	3606	3617
8	4118	4129
9	4630	4908
10	5408	5421

Pada table 6.3 menjelaskan tentang *tracking time* (pencatatan waktu) pada node server dengan menggambil hanya sampai ke sepuluh pengeksekusian program. Hasil dari pengujian diatas semakin naik menyesuaikan dengan *internal timing* di Arduino nano pada node server.

Delay tiap 1 kali eksekusi program pada node server:

Tabel 6.4 Selisih Waktu pada Node Server

Eksekusi program ke -	Selisih waktu ke 1 dan 2 (ms)	Selisih waktu ke 2 dan 1 pada eksekusi program berikutnya (ms)
1	277	501
2	12	501
3	11	501
4	11	501
5	277	501
6	12	501
7	11	501
8	11	501
9	278	500
10	13	-
Rata - rata	91,3	450,8

Pada table 6.4 menjelaskan tentang *delay* (selisih waktu) di node server dari *tracking time* pada table 6.3. Dari penghitungan rata-rata didapatkan, yaitu : 1. *Delay* ke 1 dan 2 = 91,3 ms; 2. *Delay* ke 2 dan1 (eksekusi progam berikutnya) = 450,8 ms.

6.2.2.2 Dengan *Delay* 1 detik atau 1000 ms

a. Node Client

Tabel 6.5 Node Client

Eksekusi Program ke-	Waktu ke – 2 (ms)	Waktu ke – 3 (ms)	Waktu ke – 4 (ms)
1	963	1235	1236
2	2239	2241	2241
3	3253	3525	3525
4	4529	4530	4531
5	5542	5814	5815
6	6818	6819	6820
7	7831	8103	8104
8	9108	9109	9110
9	10121	10393	10394
10	11398	11399	11400

Pada table 6.5 menjelaskan tentang *tracking time* (pencatatan waktu) pada *node client* dengan menggambil hanya sampai ke sepuluh pengeksekusian program. Hasil dari pengujian diatas semakin naik menyesuaikan dengan *internal timing* di Arduino nano pada node client.

Delay tiap 1 kali eksekusi program pada node client:

Tabel 6.6 Selisih Waktu pada Node Client

Eksekusi Program ke-	Selisih waktu ke 2 dan 3 (ms)	Selisih waktu ke 3 dan 4 (ms)	Selisih waktu ke 4 dan 2 pada eksekusi program berikutnya (ms)
1	272	1	1003
2	2	0	1012
3	272	0	1004
4	1	1	1011
5	272	1	1003
6	1	1	1011
7	272	1	1004
8	1	1	1011
9	272	1	1004
10	1	1	-
Rata - rata	136,6	0,8	906,3

Pada table 6.6 menjelaskan tentang *delay* (selisih waktu) di node client dari *tracking time* pada table 6.5. Dari penghitungan rata-rata didapatkan, yaitu : 1. *Delay* ke 1 dan 2 = 136,6 ms; 2. *Delay* ke 2 dan 3 = 0,8 ms; 3. *Delay* ke 3 dan 1 (eksekusi progam berikutnya) = 906,3 ms.

b. Node Server

Tabel 6.7 Delay 1 detik pada Node Server

Eksekusi program ke -	Waktu ke – 1 (ms)	Waktu ke – 2 (ms)
1	0	277
2	1278	1290
3	2290	2568
4	3568	3580
5	4581	4858
6	5859	5871
7	6872	7149
8	8150	8162
9	9162	9440
10	10440	10452

Pada table 6.7 menjelaskan tentang *tracking time* (pencatatan waktu) pada node server dengan menggambil hanya sampai ke sepuluh pengeksekusian program. Hasil dari pengujian diatas semakin naik menyesuaikan dengan *internal timing* di Arduino nano pada node server..

Delay tiap 1 kali eksekusi program pada node server:

Tabel 6.8 Selisih Waktu pada Node Server

Eksekusi program ke -	Selisih waktu ke 1 dan 2 (ms)	Selisih waktu ke 2 dan 1 pada eksekusi program berikutnya (ms)
1	277	1001
2	12	1000
3	278	1000
4	12	1001
5	277	1001
6	12	1001
7	277	1001
8	12	1000
9	278	1000
10	12	-
Rata - rata	144,7	900,5

Pada table 6.8 menjelaskan tentang *delay* (selisih waktu) di node server dari *tracking time* pada table 6.7. Dari penghitungan rata-rata didapatkan, yaitu : 1. *Delay* ke 1 dan 2 = 144,7 ms; 2. *Delay* ke 2 dan1 (eksekusi progam berikutnya) = 900,5 ms.

6.2.2.3 Dengan Delay 1,5 detik atau 1500 ms

a. Node Client

Tabel 6.9 Node Client

Eksekusi Program ke-	Waktu ke – 2 (ms)	Waktu ke – 3 (ms)	Waktu ke – 4 (ms)
1	968	1241	1241
2	2745	2746	2747
3	4258	4531	4531
4	6035	6036	6037
5	7548	7821	7822
6	9325	9326	9327
7	10839	11111	11112
8	12615	12616	12617
9	14129	14401	14402
10	15905	15907	15907

Pada table 6.9 menjelaskan tentang *tracking time* (pencatatan waktu) pada *node client* dengan menggambil hanya sampai ke sepuluh pengeksekusian program. Hasil dari pengujian diatas semakin naik menyesuaikan dengan *internal timing* di Arduino nano pada node client.

Delay tiap 1 kali eksekusi program pada node client:

Tabel 6.10 Selisih Waktu pada Node Client

Eksekusi Program ke-	Selisih waktu ke 2 dan 3 (ms)	Selisih waktu ke 3 dan 4 (ms)	Selisih waktu ke 4 dan 2 pada eksekusi program berikutnya (ms)
1	273	0	1504
2	1	1	1511
3	273	0	1504
4	1	1	1511
5	273	1	1503
6	1	1	1512
7	272	1	1503
8	1	1	1512
9	272	1	1503
10	2	0	-
Rata – rata	136,9	0,7	1356,3

Pada table 6.10 menjelaskan tentang *delay* (selisih waktu) di node client dari *tracking time* pada table 6.9. Dari penghitungan rata-rata didapatkan, yaitu : 1. *Delay* ke 1 dan 2 = 136,9 ms; 2. *Delay* ke 2 dan 3 = 0,7 ms; 3. *Delay* ke 3 dan 1 (eksekusi progam berikutnya) = 1356,3 ms.

b. Node Server

Tabel 6.11 Node Server

Eksekusi program ke -	Waktu ke – 1 (ms)	Waktu ke – 2 (ms)
1	0	277
2	1778	1789
3	3291	3568
4	5068	5081
5	6581	6858
6	8359	8371
7	9872	10149
8	11650	11662
9	13162	13441
10	14941	14953

Pada table 6.11 menjelaskan tentang *tracking time* (pencatatan waktu) pada *node server* dengan menggambil hanya sampai ke sepuluh pengeksekusian program. Hasil dari pengujian diatas semakin naik menyesuaikan dengan *internal timing* di Arduino nano pada node server.

Delay tiap 1 kali eksekusi program pada node Server:

Tabel 6.12 Selisih Waktu pada Node Server

Eksekusi program ke -	Selisih waktu ke 1 dan 2 (ms)	Selisih waktu ke 2 dan 1 pada eksekusi program berikutnya (ms)
1	277	1501
2	11	1502
3	277	1500
4	13	1500
5	277	1501
6	12	1501
7	277	1501
8	12	1500
9	279	1500
10	12	-
Rata – rata	144,7	1350,6

Pada table 6.12 menjelaskan tentang *delay* (selisih waktu) di *node server* dari *tracking time* pada table 6.11. Dari penghitungan rata-rata didapatkan, yaitu : 1. *Delay* ke 1 dan 2 = 144,7 ms; 2. *Delay* ke 2 dan1 (eksekusi progam berikutnya) = 1350,6 ms.

6.2.3 Analisi Pengujian

Dari hasil pengujian *delay* pada *node client* dan node server didapatkan rata – rata sebagai berikut:

1. Node Client

Tabel 6.13 Node Client

Keterangan	Node Client		
Waktu	Selisih waktu ke 1 dan 2 (ms)	Selisih waktu ke 2 dan 3 (ms)	Selisih waktu ke 3 dan 1 pada eksekusi program berikutnya (ms)
Delay 500 ms	82,4	0,8	457,4
Delay 1000 ms	136,6	0,8	906,3
Delay 1500 ms	136,9	0,7	1356,3

Pada table 6.13 disimpulkan, yaitu : 1. *Delay* ke 1 dan 2 akan memiliki *delay* tetap/stabil pada delay 1000 ms keatas; 2. *Delay* ke 2 dan 3 akan memiliki nilai tetap/stabil pada semua delay; 3. *Delay* ke 3 dan 1 (eksekusi berikutnya) akan bernilai kurang lebih sama dengan *delay* yang diberikan.

2. Node Server

Tabel 6.14 Node Server

Keterangan	Node Server	
Waktu	Selisih waktu ke 1 dan 2 (ms)	Selisih waktu ke 2 dan 1 pada eksekusi program berikutnya (ms)
Delay 500 ms	91,3	450,8
Delay 1000 ms	144,7	900,5
Delay 1500 ms	144,7	1350,6

Pada table 6.14 disimpulkan, yaitu : 1. *Delay* ke 1 dan 2 akan memiliki *delay* tetap/stabil pada delay 1000 ms keatas; 2. *Delay* ke 2 1 (eksekusi berikutnya) akan bernilai kurang lebih sama dengan *delay* yang diberikan.

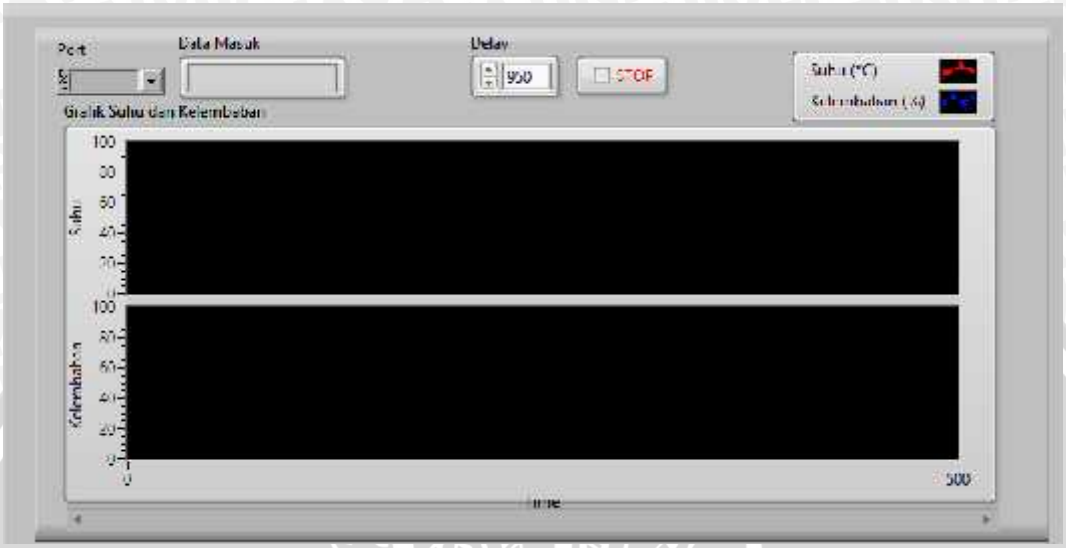
6.3 Pengujian Fitur Aplikasi Monitoring

Pengujian fitur aplikasi monitoring bertujuan untuk mengetahui apakah fitur-fitur yang digunakan dalam aplikasi ini bisa berjalan dengan baik dan lancar atau tidak.

6.3.1 Skenario Pengujian

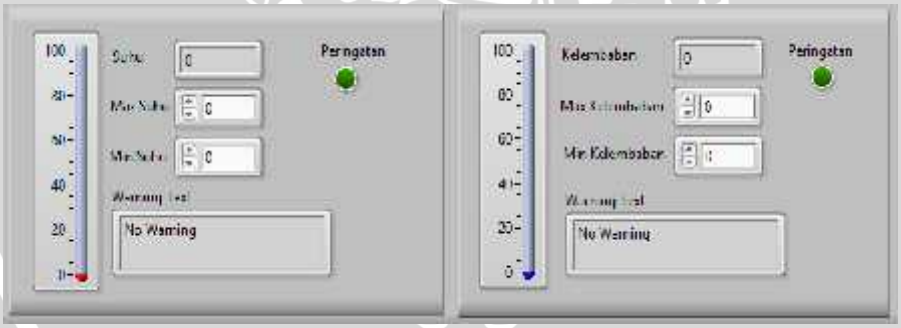
Skenario ini akan diuji semua fitur yang ada, seperti apakah kolak pilihan *delay* sudah bisa digunakan atau tidak, dan lain-lain.

Pengujian pertama yaitu menguji fitur pada grafik suhu dan kelembaban. Pada tampilan awal akan seperti gambar dibawah ini, dan akan mengatur *port USB* mana yang akan digunakan, mengatur fungsi *delay*, terdapat tombol *stop* untuk mengakhiri berjalannya program. Berikut gambar permulaan dari pengujian pertama.



Gambar 6.9 Tampilan Pengujian pada Grafik dan fitur pendukung lainnya

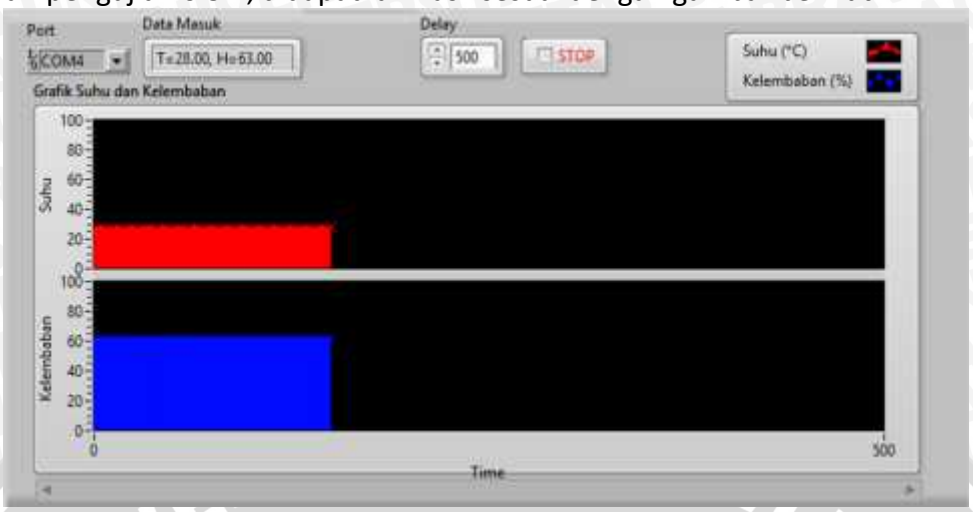
Pengujian kedua adalah fitur batas maksimal dan minimal suhu dan kelembaban, dalam hal ini peneliti melakukan pengujian untuk mengetahui apakah fitur ini bisa berjalan dengan baik dan lancar atau tidak.



Gambar 6.10 Tampilan Batas Maksimal dan Minimal suhu dan kelemababan

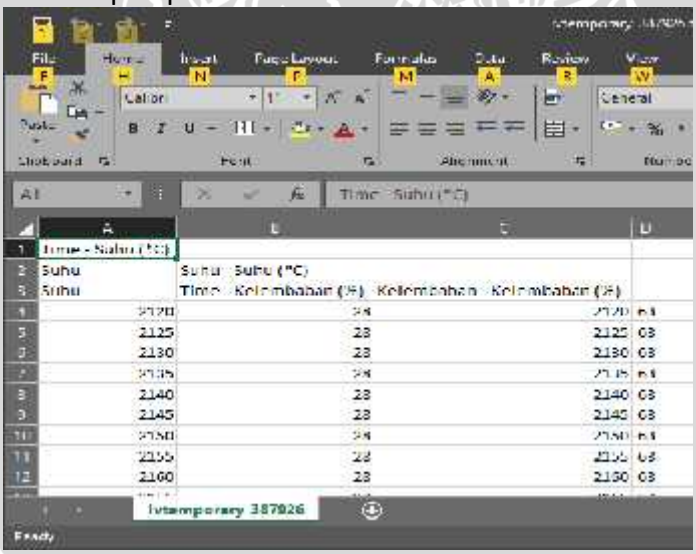
6.3.2 Hasil Pengujian

Dari pengujian 6.3.1, didapatkan hasil sesuai dengan gambar berikut ini :



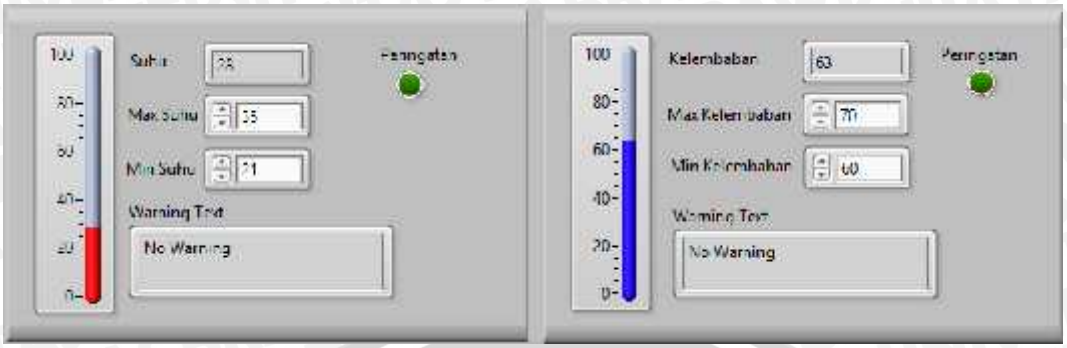
Gambar 6.11 Hasil dari Tampilan Pengujian pada Grafik dan fitur pendukung lainnya

Pada gambar 6.13 akan muncul seperti diatas apabila *port USB* telah dipilih. Pemilihan *port USB* adalah *port USB* yang digunakan pada *node server*. Untuk *delay* secara otomatis adalah 950 ms. Apabila tombol stop ditekan maka data plot grafik akan tersimpan pada excel.



Gambar 6.12 Tampilan Setelah tombol

Pada gambar 6.12 merupakan hasil *output* dari data plot grafik yang tersimpan di excel.



Gambar 6.13 Tampilan Pengujian Fitur Maksimal dan Minimal Suhu dan Kelembaban

Pada gambar 6.13 merupan tampilan fitur maksimal dan minimal suhu dan kelembaban. Pada batas maksimum dan minimum bisa diganti sesuai keinginan user. Dan nantinya akan dibandingkan dengan data suhu maupun kelembaban. Apabila masih di dalam batas anatara maksimum dan minimum makan pada *warning text* akan menampilkan *No Warning*. Apabila nilai suhu maupun kelembaban sama dengan maksimum maka akan menampilkan *Over Temperature* maupun *Over Humidity.* Apabila nilai suhu maupun kelembaban dibawah samadengan nilai minimum makan *warning text* akan menampilkan *Under Limit Temperature* maupun *Under Limit Humidity.*

6.3.3 Analisis Pengujian

Pengujian 6.3.1 peneliti menarik kesimpulan bahwan fitur-fitur aplikasi monitoring suhu dan kelembaban sudah bisa digunakan dengan baik dan sesuai dengan kebutuhan sistem. Data suhu dan kelembaban yang didapatkan bisa ditampilkan dalam bentuk grafik dan dapat dibandingkan dengan maksimum dan minimum suhu dan kelembaban. Led pada tampilan maksimum dan minimum suhu dan kelembaban akan menyala apabila tedapat *text warning* yang muncul.

BAB 7 PENUTUP

Bab ini berisikan kesimpulan yang diperoleh dari hasil pengujian dan analisa serta saran yang perlu ditambahkan uantuk penelitian selanjutnya.

7.1 Kesimpulan

Berdasarkan hasil pengujian, analisa, dan seluruh proses pengerjaan skripsi yang telah peneliti laukan dapat disimpulkan bahwa:

1. Sistem monitoring suhu dan kelembaban udara yang menggunakan Arduino nano sebagai pemroses data dan DHT11 (sensor suhu dan kelembaban udara) sebagai *input* dengan LabView NI sebagai *output* telah berjalan dengan baik. Secara teknis dengan menyalakan *node client* dan menghubungkan *node server* pada laptop/komputer sehingga *user* bisa mengakses data suhu dan kelembaban udara melalui LabView NI.
2. Sistem monitoring suhu dan kelembaban udara menggunakan komunikasi data nirkabel (nRF24L01) dengan cara *send and receive request* yang diimplementasikan pada *node client* dan *node server*.
3. Aplikasi monitoring suhu dan kelembabann dirancang dengan memiliki beberapa fitur yaitu pemilihan port usb yang akan digunakan, tombol stop yang berfungsi sebagai menghentikan berjalannya program dan fungsi maksimum dan minimum pada suhu dan kelembaban yang bertujuan untuk mengontrol suhu dan kelembaban sesuia dengan kebutuhan pengguna.
4. Hasil pengujian *delay* yaitu, node client disimpulkan : 1. *Delay* ke 1 dan 2 akan memiliki *delay* tetap/stabil pada delay 1000 ms keatas; 2. *Delay* ke 2 dan 3 akan memiliki nilai tetap/stabil pada semua delay; 3. *Delay* ke 3 dan 1 (eksekusi berikutnya) akan bernilai kurang lebih sama dengan *delay* yang diberikan. Hasil pengujian *delay* pada node server disimpulkan : 1. *Delay* ke 1 dan 2 akan memiliki *delay* tetap/stabil pada delay 1000 ms keatas; 2. *Delay* ke 2 1 (eksekusi berikutnya) akan bernilai kurang lebih sama dengan *delay* yang diberikan.

7.2 Saran

Setelah menganalisa proses pengerjaan skripsi ini dari awal hingga akhir, diharapkan untuk kedepannya sistem dapat dikembangkan dengan:

1. Memperbarui sensor suhu dan kelembaban dengan kualitas yang lebih akurat dan cepat dalam pembacaan perubahaan suhu dan kelembaban.
2. Memperbarui modul *tranciever* dan *receiver* dengan yang lebih kuat dalam jangkauan penerimaan dan pengiriman data.
3. Mengembangkan sistem kontrol yang berbasis web dan terhubung dengan Internet.
4. Mengembangkan sistem penyimpanan data suhu dan kelembaban dengan menggunakan database.
5. Menambahkan alarm atau speaker untuk deteksi suhu dan kelembaban yang *Over* atau *underlimit.*

DAFTAR PUSTAKA

Arduino, 2015. Product Arduino. [online] Tersedia di: < http://www.arduino.cc/en/Main/Products>[Diakses 1 Desember 2015].

DhT11 , 2010. *DHT11 Humidity & Temperature Sensor,* 9(3), p.5-6.

Kazem S., Daniel M., dan Taieb Z., 2007. *WIRELESS SENSOR NETWORK Technology, Protocols, and Applications.*[E-book].Wiley. [Diakses 19 September 2015].

Kristianto Eko., 2013. Monitoring Suhu Jarak Jauh Generator Ac Berbasis Mikrokontroler, 17(3), p.5.

LabVIEW, 2012. *LabVIEW™ Core 1 Course Manual,* 230(12), p.6-8.

nRF24L01 , 2008. *nRF24L01 pluss Preliminary Product Specification v1.0,* 71(7), p.8-9.

Muzakhim, Azam., 2011. *Telemetri dan Telekontrol Antral Mikrokontroler Menggunakan Xbee-Pro Wireless*, 13(2), p.1.

Susanto, Heri., Rozeff, Pramana, S.T, M.T., & Muhammad, Mujahidin, S.T, M.T., 2013. *Perancangan Sistem Telemetri Wireless untuk Mengukur Suhu dan Kelembaban Berbasis Arduino Uno R3 Atmega328P dan Xbee Pro,* 12(2), p.1.