

SKRIPSI

PENGEMBANGAN APLIKASI PEMANTAUAN KUALITAS TANAH SAWAH BERBASIS WSN



Reynaldi Irfan Anwar

NPM: 2016730045

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2021

UNDERGRADUATE THESIS

**DEVELOPMENT OF PADDY LAND QUALITY
APPLICATION BASED ON WIRELESS SENSOR NETWORK**



Reynaldi Irfan Anwar

NPM: 2016730045

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2021**

LEMBAR PENGESAHAN

PENGEMBANGAN APLIKASI PEMANTAUAN KUALITAS TANAH SAWAH BERBASIS WSN

Reynaldi Irfan Anwar

NPM: 2016730045

Bandung, 2 Februari 2021

**Menyetujui,
Pembimbing Utama**

Elisatih Hulu, M.T.

Ketua Tim Penguji

Anggota Tim Penguji

**Pascal Alfadian Nugroho,
S.Kom,M.Comp**

**Dr.rer.nat. Cecilia Esti Nugraheni,
S.T., M.T.**

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, SSi, MSc, PDEng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

PENGEMBANGAN APLIKASI PEMANTAUAN KUALITAS TANAH SAWAH BERBASIS WSN

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 2 Februari 2021



Reynaldi Irfan Anwar
NPM: 2016730045

ABSTRAK

Pertumbuhan tanaman padi sangat dipengaruhi oleh kondisi tanah sawah yang dikelola. Jenis pengelelolaan tanah sawah yang tepat dapat meningkatkan kualitas tanaman padi. Oleh karena itu, perlu diketahuinya variabel-variabel yang mempengaruhi kualitas tanah sawah yang dikelola oleh para petani. Pengembangan aplikasi pemantauan kualitas tanah sawah berbasis WSN dibangun untuk membantu para petani untuk mendapatkan data kondisi tanah sawahnya. Data diambil menggunakan sensor yang dapat mengambil variabel tanah yang diantaranya kadar keasaman tanah, kelembaban dan suhu tanah, dan kelembaban dan suhu area persawahan.

Terdapat beragam jenis perangkat keras yang dapat dilakukan untuk melakukan pemantauan terhadap suatu lingkungan. Pada skripsi ini, perangkat yang digunakan untuk melakukan pemantauan adalah Arduino, sensor *sensing*, dan Raspberry Pi. Arduino akan berperan sebagai node sensor, sedangkan Raspberry Pi akan berperan sebagai *base station*. Arduino yang terhubung dengan sensor *sensing* akan disebar di lingkungan persawahan untuk mengambil variabel tanah, yang lalu akan dikirimkannya ke Raspberry Pi untuk ditampilkan ke pengguna.

Pada Skripsi ini juga dibangun aplikasi untuk melakukan pemantauan kualitas tanah sawah. Aplikasi yang dibangun terdiri dari aplikasi untuk admin dan aplikasi untuk pengguna. Aplikasi admin memiliki fitur-fitur yang mengontrol aktifitas node sensor maupun *base station*, sedangkan aplikasi pengguna berfokuskan untuk menampilkan hasil pemantauan yang dilakukan ke pengguna.

Hasil dari pengujian yang dilakukan menunjukan bahwa aplikasi pemantauan kualitas tanah sawah yang dibangun mampu memprediksi kualitas tanah dari tanah persawahan yang diuji.

Kata-kata kunci: padi, tanah, sawah, WSN, jaringan, sensor padi, tanah, sawah, WSN, jaringan, sensor

ABSTRACT

The growth of rice plants is very much influenced by the condition of the paddy fields being managed. The right type of rice field management can improve the quality of rice plants. Therefore, it is necessary to know the variables that affect the quality of paddy soils managed by farmers. Development of WSN-based rice quality monitoring application is built to help farmers to get data on the condition of their paddy soil. Data is taken using sensors that can take soil variables including soil acidity, humidity and soil temperature, and humidity and temperature of rice fields.

There are various types of hardware that can be done to monitor an environment. In this thesis, the devices used for monitoring are Arduino, sensing sensor, and Raspberry Pi. The Arduino will act as a sensor node, while the Raspberry Pi will act as a base station. Arduino which is connected to the sensing sensor will be deployed in the rice fields to take soil variables, which will then be sent to the Raspberry Pi to be displayed to the user.

In this thesis an application is also built to monitor the quality of the paddy soil. Applications built consist of applications for admin and applications for users. The admin application has features that control the activity of sensor nodes and base stations, while the user application focuses on displaying the results of monitoring carried out to the user.

The results of the tests carried out show that the application of monitoring the quality of the paddy soil that was built is able to estimate the quality of the soil from the rice fields being tested.

Keywords: rice, land, rice fields, WSN, networks, sensors

Dipersembahkan kepada Allah SWT, diri sendiri, keluarga, dan kerabat dekat yang telah mendukung

KATA PENGANTAR

Puji dan syukur penulis ucapkan pada kehadiran Allah SWT, karena dengan rahmat dan izin dari-Nya penulis dapat menyelesaikan penyusunan skripsi ini, sebagai syarat untuk memperoleh gelar Sarjana Komputer, pada Program Studi S1 Teknik Informatika Fakultas Teknologi Informasi dan Sains Universitas Katolik Parahyangan. Penulis menyadari bahwa penyusunan skripsi ini juga tidak terlepas dari bantuan berbagai pihak, baik langsung maupun tidak langsung. Secara khusus, penulis ingin berterima kasih kepada:

1. Allah SWT atas segala Rahmat-Nya.
2. Keluarga yang selalu memberikan dukungan baik secara mental, doa, dan finansial.
3. Bapak Elisati Hulu, M.T. selaku dosen pembimbing yang telah membimbing dan membantu penulis dalam proses penulisan skripsi maupun kendala pemrograman yang penulis hadapi.
4. Bapak Pascal Alfadian Nugroho, S.Kom, M.Comp dan Dr.rer.nat. Cecilia Esti Nugraheni, ST, M.T. selaku dosen penguji yang telah memberikan masukan kritik serta saran terhadap buku skripsi ini, sehingga pembangunan aplikasi serta penulisan buku skripsi ini menjadi lebih baik lagi.
5. Ivan Kristanto, S.Kom yang telah memberikan banyak referensi terkait teori jaringan dan implementasi perangkat yang digunakan dalam pengujian.
6. Intan Crystina Zainuddin, S.Kom dan Anugrah Jaya Sakti, S.Kom yang banyak memberikan support moral, juga membantu dalam memperbaiki tata cara penulisan skripsi ini agar menjadi lebih baik lagi.

Semoga Allah SWT membalas kebaikan kepada seluruh pihak yang telah memberikan bantuan serta dukungan kepada penulis, dalam menyelesaikan skripsi ini. Penulis menyadari bahwa penelitian ini masih jauh dari kata sempurna. Oleh karena itu, penulis memohon maaf jika terdapat kesalahan. Semoga buku skripsi ini dapat memberi informasi yang bermanfaat dan menjadi inspirasi untuk pengembangan aplikasi pemantauan berikutnya.

Bandung, Februari 2021

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
DAFTAR KODE PROGRAM	xxiii
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	3
1.6 Sistematika Pembahasan	4
DAFTAR NOTASI	1
2 LANDASAN TEORI	5
2.1 Tanah sawah	5
2.1.1 Klasifikasi tanah sawah	5
2.1.2 Variabel yang mempengaruhi kondisi tanah sawah	6
2.1.3 Klasifikasi kondisi tanah sawah	7
2.1.4 Metode pemantauan kualitas tanah sawah	8
2.2 WSN (<i>Wireless Sensor Network</i>)	9
2.2.1 Pemanfaatan <i>Wireless Sensor Network</i>	9
2.2.2 Klasifikasi Node	10
2.2.3 Komponen Node Sensor	10
2.2.4 Arsitektur <i>Wireless Sensor Network</i>	11
2.2.5 Topologi <i>Wireless Sensor Network</i>	13
2.2.6 Protokol <i>Wireless Sensor Network</i>	17
2.3 Zigbee	19
2.3.1 Jenis-jenis perangkat zigbee	19
2.4 Arduino	20
2.4.1 Jenis-jenis arduino	20
2.4.2 Jenis-jenis sensor <i>sensing</i> arduino	23
2.4.3 Pemrograman Arduino	26
2.5 Raspberry	29
2.5.1 Jenis-jenis Raspberry	29
2.5.2 Pemrograman Raspberry	32
2.5.3 Pemrograman Python untuk Raspberry	32

2.6	Basis Data	32
3	ANALISIS	33
3.1	Deskripsi Sistem	33
3.2	Analisis Arsitektur dan Topologi WSN	33
3.3	Analisis Perangkat lunak	35
3.3.1	Analisis Fungsi Aplikasi	36
3.3.2	Analisis Kelas	42
3.3.3	Analisis Basis Data	44
3.3.4	Analisis Paket/Pesan	45
4	PERANCANGAN	47
4.1	Perancangan Interaksi Antar Node	47
4.1.1	Diagram Sequence "Check Status Node" Aplikasi Pengguna	47
4.1.2	Diagram Sequence "Check Status Node" Aplikasi Admin	48
4.1.3	Diagram Sequence "Sensing" Aplikasi Pengguna	49
4.1.4	Diagram Sequence "Mulai Sensing" Aplikasi Admin	50
4.1.5	Diagram Sequence "Stop Sensing" Aplikasi Admin	51
4.2	Perancangan Antarmuka Aplikasi	51
4.2.1	Mockup Halaman Utama	51
4.2.2	Mockup Halaman "Sensing"	52
4.2.3	Mockup Halaman "Check Status"	52
4.2.4	Mockup Halaman "Cara Penggunaan"	53
4.2.5	Mockup Halaman "Print Sensing"	54
4.3	Perancangan Kelas Aplikasi	54
4.3.1	Kelas Node Sensor	54
4.3.2	Kelas Base Station	57
4.4	Perancangan <i>Input</i> dan <i>Output</i>	59
5	IMPLEMENTASI DAN PENGUJIAN	61
5.1	Implementasi	61
5.1.1	Lingkungan Implementasi	61
5.1.2	Hasil Implementasi	62
5.2	Pengujian	70
5.2.1	Pengujian Fungsional	70
5.2.2	Pengujian Eksperimental	79
5.2.3	Pengujian Lapangan	79
5.2.4	Skala Penilaian Kualitas Tanah Hasil Pengujian	81
5.3	Kendala	86
5.3.1	Kendala Pemrograman	87
5.3.2	Kendala Perangkat Keras	87
6	KESIMPULAN DAN SARAN	89
6.1	Kesimpulan	89
6.2	Saran	89
DAFTAR REFERENSI		91
A	KODE PROGRAM NODE SENSOR	93
B	KODE PROGRAM BASE STATION	97
C	KODE PROGRAM APLIKASI PENGGUNA (WEBSITE)	101

DAFTAR GAMBAR

2.1	Node sensor	11
2.2	Arsitektur <i>Wireless Sensor Network</i>	12
2.3	<i>Arsitektur Flat</i>	12
2.4	<i>Arsitektur Single Hop</i>	13
2.5	<i>Arsitektur Multi Hop</i>	13
2.6	Topologi <i>Bus</i>	14
2.7	Topologi <i>Star</i>	14
2.8	<i>Topologi Tree</i>	15
2.9	Jaringan <i>Star</i> pada Topologi <i>Tree</i>	15
2.10	Topologi <i>Ring</i>	16
2.11	Topologi <i>Linear</i>	16
2.12	<i>Partially connected mesh</i>	17
2.13	<i>Fully connected mesh</i>	17
2.14	Protokol Jaringan	19
2.15	Arduino Uno	20
2.16	Arduino Due	20
2.17	Arduino Mega	21
2.18	Arduino Leonardo	21
2.19	Arduino Fio	21
2.20	Arduino Nano	22
2.21	Arduino Mini	22
2.22	Arduino Micro	22
2.23	Arduino Ethernet	23
2.24	Arduino Esplora	23
2.25	Sensor pengukur kadar keasaman tanah (pH)	24
2.26	Sensor pengukur tingkat kelembaban tanah	24
2.27	Sensor pengukur suhu temperatur tanah	25
2.28	Sensor pengukur suhu kelembaban udara	25
2.29	<i>Wireless Data Transceiver</i>	25
2.30	Fungsi setup dan loop pada Arduino	26
2.31	<i>Blink Test On-Board</i>	28
2.32	<i>Blink Test Pin</i>	29
2.33	Raspberry pi A	30
2.34	Raspberry pi A+	30
2.35	Raspberry pi B	30
2.36	Raspberry pi B+	31
2.37	Raspberry pi 2 B+	31
2.38	Raspberry pi 3 B+	31
2.39	Shell python setelah eksekusi <i>module</i>	32
3.1	Node Sensor	34
3.2	Arsitektur dan Topologi WSN	35

3.3	Flowchart keseluruhan sistem	36
3.4	Use Case Diagram Aplikasi	37
3.5	Kelas Diagram Node Sensor	42
3.6	Kelas Diagram <i>Base station</i>	43
3.7	ERD Perangkat lunak	44
4.1	Sequence Diagram Check Status Aplikasi Pengguna	47
4.2	Sequence Diagram Check Status Aplikasi Admin	48
4.3	Sequence Diagram Sensing Aplikasi Pengguna	49
4.4	Sequence Diagram Sensing Aplikasi Admin	50
4.5	Sequence Diagram Stop Sensing Aplikasi Admin	51
4.6	Mockup Halaman Utama	52
4.7	Mockup Halaman Sensing Online	52
4.8	Mockup Halaman Check Status	53
4.9	Mockup Halaman Cara Penggunaan	53
4.10	Mockup Halaman Print Sensing	54
4.11	Kelas Diagram Node Sensor	54
4.12	Kelas Diagram <i>Base station</i>	57
5.1	Arduino menampilkan hasil sensing	71
5.2	Raspberry Pi menampilkan opsi fitur	71
5.3	Raspberry Pi menerima respon status node	72
5.4	Raspberry Pi tidak menerima respon status node	72
5.5	Raspberry Pi menerima hasil <i>sensing</i>	73
5.6	Raspberry Pi menyimpan data sensing	73
5.7	Raspberry Pi mengirim perintah stop <i>sensing</i>	74
5.8	Raspberry Pi mengirim perintah keluar dari aplikasi	74
5.9	Halaman Utama	75
5.10	Halaman Check Status sebelum mendapat pengiriman paket	75
5.11	Halaman Check Status setelah mendapat pengiriman paket	76
5.12	Halaman Check Status setelah opsi perintah stop <i>sensing</i> dieksekusi	76
5.13	Halaman Sensing setelah mendapat pengiriman paket	77
5.14	Halaman Cara Pakai	77
5.15	Halaman Cara Pakai 2	78
5.16	Halaman Print Sensing	78
5.17	Pengujian Eksperimental	79
5.18	Peta Sebaran Node	79
5.19	Pengujian Lapangan - Node Disebar	80
5.20	Pengujian Lapangan - Base Station	81
5.21	Kadar Ph Tanah Siang Hari	85
5.22	Kadar Ph Tanah Pagi Hari	85
5.23	Suhu Tanah di Siang Hari	85
5.24	Suhu Tanah di Pagi Hari	85
5.25	Kelembaban Tanah di Siang Hari	86
5.26	Kelembaban Tanah di Pagi Hari	86
5.27	Suhu Udara di Siang Hari	86
5.28	Suhu Udara di Pagi Hari	86

DAFTAR TABEL

2.1	Tabel kelembaban ideal tanaman	7
2.2	Parameter dan Metode Analisis Kualitas Tanah (Lal,1994)	8
3.1	Tabel Skenario Memeriksa Status Node Sensor Aplikasi Admin	38
3.2	Tabel Skenario Perintah <i>Sensing</i> Aplikasi Admin	39
3.3	Tabel Skenario Memeriksa Status Node Sensor Aplikasi Pengguna	40
3.4	Tabel Skenario Perintah <i>Sensing</i> Aplikasi Pengguna	40
3.5	Tabel Skenario Menghentikan <i>Sensing</i> Aplikasi Admin	41
3.6	Tabel Skenario Keluar Aplikasi Aplikasi Admin	41
3.7	Tabel Skenario Keluar Aplikasi Aplikasi Pengguna	42
5.1	Tabel Pemantauan di Siang Hari	82
5.2	Tabel Pemantauan di Pagi Hari	84

DAFTAR KODE PROGRAM

2.1 rasp_test.py	32
5.1 Metode setup()	62
5.2 Metode loop()	63
5.3 Metode bacaSensorKelembabanTanah()	63
5.4 Metode bacaSensorKelembabanUdara()	63
5.5 Metode bacaSensorSuhuTanah()	64
5.6 Metode bacaSensorSuhuUdara()	64
5.7 Metode bacaSensorPhTanah()	64
5.8 Metode conversiDataFloatToString()	64
5.9 Respon Perintah Base Station	65
5.10 Metode bacaSensorPhTanah()	65
5.11 Metode bacaSensorKelembabanTanah()	65
5.12 Metode checkStatusNode()	65
5.13 Metode checkStatusSensing()	66
5.14 Metode transmisiPengiriman()	66
5.15 Metode getDataSensing()	67
5.16 Metode getDataSensing()	67
5.17 Metode sendDataSensing()	67
5.18 Metode mainMenu()	68
5.19 Metode getPingArduino()	68
5.20 Metode counterAdd()	68
5.21 Metode counterSet()	69
5.22 Metode terimaRespon()	69
5.23 Metode tester()	69
A.1 Node.ino	93
B.1 BaseStation.py	97
C.1 skripsi_header.blade.php	101
C.2 skripsi_home.blade.php	101
C.3 skripsi_checkstatus.blade.php	102
C.4 skripsi_sensing.blade.php	104
C.5 skripsi_perangkat.blade.php	108
C.6 skripsi_carapakai.blade.php	109
C.7 web.php	110
C.8 controller.php	111
C.9 StatusViewController.php	111
C.10 PrintViewController.php	111
C.11 SensingViewController.php	112

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan negara agraris terbesar di dunia, sehingga sebagian besar profesi utama masyarakat Indonesia adalah petani. Sebagai negara agraris, Indonesia dianugerahi kekayaan alam yang melimpah dengan letak geografis yang dinilai sangat strategis untuk bercocok tanam. Dipandang dari segi geografis, Indonesia terletak pada daerah tropis yang memiliki curah hujan yang tinggi. Iklim tropis ini memungkinkan banyak jenis tumbuh-tumbuhan berkembang dengan cepat, terutama tumbuhan tanaman padi¹.

Tanaman padi merupakan bahan pangan utama masyarakat Indonesia. Masyarakat Indonesia terbiasa untuk mengonsumsi bahan olahan padi menjadi beras, sehingga penting untuk mendapatkan tanaman padi dengan kualitas yang baik untuk dikonsumsi sehari-hari. Kualitas pertumbuhan dan perkembangan tanaman padi sangat dipengaruhi oleh kualitas tanah sawah. Banyaknya variasi jenis tanah juga mempengaruhi jenis pengelolaan tanaman padi agar dapat tumbuh secara optimal. Tidak hanya jenis tanah yang beragam, waktu dan jumlah pupuk yang diberikan juga berpengaruh pada kualitas tanah sawah tanaman padi.

Tingkat produktivitas tanaman padi di Indonesia bervariasi bergantung pada jenis tanah dan cara pengelolaannya. Produktivitas lahan sawah terendah terletak di Bangka Belitung dengan hasil 27,81 ku/ha, sedangkan untuk produktivitas tertinggi terletak di Jawa Barat dengan hasil 59,53 ku/ha². Dari data yang didapatkan oleh BPS 2014 pada tahun 2013, didapatkan rata-rata produktivitas padi nasional sekitar 51,52 ku/ha. Berdasarkan data yang dihimpun oleh Senior Expatriate Technological Cooperation Asia Pasific Food Agriculture Organization (FAO) Ratno Soetjiptadie, 69% dari tanah Indonesia berada dalam kategori buruk. Hal ini menyebabkan kualitas tanaman padi yang dihasilkan tidak berbanding lurus dengan usaha yang dilakukan oleh petani.

Faktor lain yang menyebabkan kurang baiknya kualitas padi di Indonesia adalah masih banyaknya petani yang awam mengenai variabel-variabel penting pada pengelolaan tanah sawah. Kondisi tanah sawah sangat mempengaruhi jenis pengelolaan dan kualitas panen tanaman padi. Variabel-variabel yang mempengaruhi kualitas tanah tanaman padi antara lain tingkat keasaman(pH), kelembaban, dan suhu dari kondisi tanah sawah padi. Variabel-variabel tersebut menginformasikan kondisi tanah sawah padi.

Kurangnya pengetahuan mengenai variabel-variabel tersebut, berpotensi menyebabkan salahnya pemilihan cara pengelolaan tanah pada tanaman padi. Salah satu contoh salahnya pemilihan pengelolaan tanah tanaman padi adalah pemberian pupuk pada waktu yang tidak tepat atau pemberian kuantitas pupuk yang tidak sesuai (terlalu banyak ataupun terlalu sedikit pada tanah sawah) dengan kondisi tanah sawah. Hal-hal tersebut mempengaruhi kualitas hasil panen, zat kimia yang berasal dari pupuk tersebut tidak memberikan perkembangan terhadap tanah sawah padi, dikarenakan kondisi tanah yang berbeda-beda pada titik-titik tertentu.

Selain itu, sampai saat ini pengelolaan tanah sawah oleh para petani masih menggunakan

¹<https://money.kompas.com/read/2017/02/19/163912926/negara.agraris.mengapa.harga.pangan.di.indonesia.rawan.bergejolak.?page=all>

²<http://balittanah.litbang.pertanian.go.id/ind/dokumentasi/juknis/pemulihan%20lahan.pdf>

perkiraan, dan pengalaman. Kurangnya informasi yang akurat dan cara yang masih tradisional untuk mengetahui kondisi tanah sawah yang dikelola, menyebabkan salahnya pemilihan pengelolaan, yang berdampak pada kurangnya produktivitas tanaman padi. Salahnya pemilihan cara pengelolaan juga menimbulkan tanah sawah menjadi tidak subur, dan berpengaruh terhadap kualitas hasil panen³.

Untuk mendapatkan informasi kondisi tanah sawah padi yang akurat, metode pemantauan berbasis penelitian di laboratorium masih cukup sering dilakukan. Namun, metode ini dianggap kurang praktis baik dari sisi waktu dan biaya. Luasnya lahan tanah sawah dan variasi kondisi tanah sawah yang berbeda-beda, menyebabkan sulitnya mendapatkan hasil dengan tingkat keakuratan yang tinggi. Sehingga diperlukan alat untuk mengetahui kondisi tanah sawah padi yang dapat disebar di area lahan tanah sawah dan menampilkan data yang berisikan informasi kondisi tanah sawah secara keseluruhan⁴.

Pemanfaatan WSN (*Wireless Sensor Network*) dapat menjadi alternatif untuk melakukan pemantauan kondisi tanah sawah padi. Dengan WSN, proses pemantauan kondisi tanah dapat dilakukan dengan lebih mudah dan praktis. WSN memungkinkan pengguna untuk mendapatkan informasi seperti tingkat keasaman(pH), kelembapan, dan suhu tanah sawah secara realtime, dengan tingkat akurasi yang tinggi. Penggunaan WSN juga lebih menghemat biaya dibandingkan metode pemantauan berbasis penelitian laboratorium.

Oleh karena itu dibutuhkan pengembangan aplikasi pemantauan kualitas tanah sawah menggunakan sensor berbasis WSN (*Wireless Sensor Network*), yang dapat membantu petani mendapatkan informasi kondisi tanah sawah berdasarkan hasil pengolahan sensor-sensor (sensor keasaman(pH), sensor kelembapan, dan sensor suhu) terhadap tanah sawah padi. Dari informasi tersebut diharapkan petani dapat menentukan jenis pengelolaan yang tepat untuk tanah sawah padi dan mendapatkan hasil panen yang optimal, serta meningkatkan produktivitas tanaman padi dengan kualitas yang baik.

1.2 Rumusan Masalah

Berikut adalah masalah dari pengembangan aplikasi ini:

1. Bagaimana memantau kualitas tanah sawah padi ?
2. Bagaimana membangun aplikasi pemantau kualitas tanah sawah padi menggunakan WSN ?

1.3 Tujuan

Berikut adalah tujuan dari pengembangan aplikasi ini:

1. Mempelajari kualitas tanah sawah berdasarkan pengukuran keasaman(pH), kelembaban, dan suhu dari kondisi tanah sawah padi dengan menggunakan sensor dan mengirimkan data pengukuran tersebut secara *wireless* ke komputer.
2. Membangun aplikasi pemantau kualitas tanah sawah berbasis WSN (*Wireless Sensor Network*) dengan menggunakan perangkat keras Arduino dan Raspberry.

1.4 Batasan Masalah

Batasan masalah yang digunakan pada Pengembangan Aplikasi Pemantauan Kualitas Tanah Sawah Berbasis WSN adalah:

³<https://media.neliti.com/media/publications/170054-ID-none.pdf>

⁴https://www.researchgate.net/publication/329387376_Pemantauan_Kualitas_Air_dan_Tanah_Pertanian_Secara_Daring_dan_Waktu_Nyata_untuk_Mewujudkan_Ketahanan_Pangan

1. Jenis tanah yang digunakan untuk penelitian adalah tanah sawah irigasi
2. Pembangunan jaringan tidak lebih dari satu petak sawah
3. Titik penyebaran sensor dilakukan di pinggir sawah
4. Proses sensing dilakukan saat tanah sawah padi panen

1.5 Metodologi

Berikut adalah tahapan-tahapan yang digunakan untuk mengerjakan pengembangan aplikasi ini, antara lain :

1. Melakukan studi literatur
 - Melakukan studi literatur mengenai kualitas kondisi tanah sawah tanaman padi
 - Melakukan studi literatur mengenai WSN (*Wireless Sensor Network*)
 - Melakukan studi literatur mengenai komunikasi *wireless*
 - Melakukan studi literatur mengenai node sensor berbasis Arduino dan pemrograman sensor
 - Melakukan studi literatur mengenai sensor *sensing* berbasis Arduino
 - Melakukan studi literatur mengenai Raspberry dan pemrograman Raspberry
2. Analisis kebutuhan perangkat lunak
 - Mempelajari bahasa pemrograman C dan Python di Arduino dan Raspberry
 - Mempelajari pemrograman node Arduino dan sensor
 - Mempelajari pemrograman Raspberry sebagai *base station*
 - Mempelajari penggunaan GIS (*Geographic Information System*) sebagai sistem penyimpanan data
 - Melakukan survei penelitian tanah sawah padi
3. Perancangan perangkat lunak
 - Melakukan perancangan perangkat lunak
 - Membangun perangkat lunak
4. Implementasi perangkat lunak
 - Melakukan studi lapangan
 - Melakukan pengukuran kelembaban tanah sawah
 - Melakukan pengukuran tingkat keasaman(pH) tanah sawah
 - Melakukan pengukuran suhu temperatur *area* persawahan
5. Pengujian perangkat lunak
6. Menulis dokumen skripsi

1.6 Sistematika Pembahasan

Sistematika pembahasan pada Pengembangan Aplikasi Pemantauan Kualitas Tanah Sawah Berbasis WSN adalah sebagai berikut:

Bab 1 memuat latar belakang masalah, rumusan masalah, tujuan, batasan masalah, dan metodologi penelitian yang menjadi rujukan pengembangan aplikasi, dan sistematika pembahasan.

Bab 2 membahas teori-teori dasar yang berkaitan dengan perancangan aplikasi, yang digunakan untuk mendukung aplikasi yang dibangun. Pada bab ini akan dibahas pengertian tanah sawah, klasifikasi jenis tanah sawah, variabel yang mempengaruhi kondisi tanah sawah, dan klasifikasi kondisi tanah sawah. Akan dibahas juga deskripsi singkat sensor dan pengertian dari WSN (*Wireless Sensor Network*) beserta arsitektur dan topologinya. Deskripsi perangkat keras yang digunakan juga akan dibahas pada bab ini, antara lain deskripsi singkat sensor arduino dan *base station*, perbandingan jenis-jenis sensor arduino, dan jenis-jenis sensor *sensing* yang digunakan untuk penelitian tanah sawah berbasis arduino. Perangkat keras lain yang akan dibahas pada bab ini adalah Raspberry yang berperan sebagai titik penghubung data *sensing* ke *server* atau komputer. Pemrograman yang dilakukan pada kedua perangkat keras diatas juga akan dibahas secara sederhana di bab ini.

Bab 3 berisikan deskripsi singkat perangkat lunak yang dibangun, analisis kebutuhan perangkat lunak, dan analisis cara kerja sistem.

Bab 4 memuat perancangan komunikasi antar node sensor, dari pengiriman data ke SINK (*base station*) sampai diterima oleh gateway atau notebook.

Bab 5 memuat implementasi perangkat lunak yang dibangun sesuai dengan hasil analisis dan perancangan yang telah dibuat, hasil pengujian Fungsional dan Eksperimental, dan masalah yang dihadapi saat implementasi.

Bab 6 memuat pengujian dan kesimpulan dari perangkat lunak yang telah dibangun, beserta saran dari penulis untuk pengembangan perangkat lunak yang lebih baik.

BAB 2

LANDASAN TEORI

2.1 Tanah sawah

Tanah sawah merupakan kata majemuk yang dihasilkan dari penggabungan dua suku kata, tanah dan sawah. Berdasarkan KBBI (Kamus Besar Bahasa Indonesia), tanah dapat diartikan sebagai bahan-bahan dari bumi; bumi sebagai bahan sesuatu, sedangkan sawah dapat diartikan sebagai tanah yang digarap dan diairi untuk tempat menanam padi. Secara umum, tanah sawah dapat didefinisikan sebagai lahan tanah yang dikelola untuk bercocok tanam tanaman padi. Berbagai jenis tanah dapat disawahkan, namun kebutuhan tanaman untuk hidup tetap harus terpenuhi, seperti kadar air yang memadai [1].

2.1.1 Klasifikasi tanah sawah

Tanah kering yang diberi air atau tanah rawa yang "dikeringkan" dengan membuat saluran drainase dapat menjadi tanah sawah. Sawah dapat diklasifikasikan berdasarkan asal air yang diterimanya. Sawah yang menerima air dari hasil irigasi disebut dengan sawah irigasi, sedangkan yang menerima dari air hujan disebut dengan sawah tada hujan. Sawah pasang surut dapat ditemukan di daerah pasang surut, begitupula sawah lebak yang dikembangkan di daerah rawa-rawa lebak¹[2].

1. Sawah irigasi

Sawah irigasi adalah sawah yang menggunakan sistem pengairan (pengelolaan air) yang berasal dari bendungan atau waduk secara teratur. Dengan sistem pengairan yang diatur secara teknis, maka air yang dibutuhkan untuk menjaga kondisi tanah sawah tidak bergantung pada curah hujan. Hal ini disebabkan pengairan tersebut dapat diperoleh dari sungai, waduk atau bendungan. Sawah irigasi menggunakan air terbesar dibandingkan jenis sawah lainnya, hal ini menyebabkan sebanyak 85% sawah beririgasi masih dihadapkan kepada masalah efisiensi. Masalah ini disebabkan oleh hilangnya air selama proses irigasi (*distribution losses*) maupun proses pemakaian (*field application losses*)².

2. Sawah tada hujan

Sawah tada hujan adalah sawah yang memanfaatkan air hujan sebagai sumber utama sistem pengolahan air untuk lahan pertanian. Sistem pengairan sawah tanah hujan bekerja dengan cara menampung air hujan. Hal ini menyebabkan lahan sawah tada hujan sangat berisiko terkena kekeringan saat musim kemarau tiba. Sawah tada hujan juga memiliki tingkat kesuburan yang rendah. Tingkat kesuburan sawah tada yang rendah disebabkan oleh bergantungnya pada curah hujan dan rentannya lahan sawah terhadap serangan hama penyakit.

3. Sawah bencah atau sawah pasang surut

Sawah pasang surut adalah sawah yang umumnya dapat ditemukan disekitar muara sungai dan rawa dekat pantai. Jenis sawah ini hanya dapat diolah satu kali dalam setahun, dan

¹<http://balittanah.litbang.pertanian.go.id/ind/dokumentasi/buku/tanahsawah/tanahsawah1.pdf>

²<https://dosenpertanian.com/sawah-irigasi/>

keperluan air untuk tanaman padi dipengaruhi oleh pasang surut air laut. Tanah sawah pasang surut merupakan sebagian dari hasil sedimentasi lumpur, yang dibentuk oleh luapan air sungai saat air laut pasang.

4. Sawah lebak

Sawah lebak memiliki kemiripan dengan sawah pasang surut. Sawah lebak umumnya terletak didataran rendah sekitar sungai, untuk ditanamani berbagai macam tanaman padi. Letaknya lahan sawah yang berdekatan dengan sungai, mengakibatkan rawannya tanaman padi terhadap banjir pada musim hujan. Hal ini yang menyebabkannya cukup seringnya lahan sawah lebak dialihfungsikan menjadi lahan perkebunan.

Perbedaan yang signifikan antara sawah lebak dengan sawah pasang surut adalah sumber utama sistem pengairan yang didapatkan. Sawah pasang surut mengandalkan pasang surut air laut sebagai pemasokan kebutuhan air lahan sawah, sedangkan sawah lebak mengandalkan air hujan sebagai sumber utama sistem pengelolaan air lahan pertanian. Sehingga resiko kekeringan pada sawah tadah hujan juga dapat terjadi pada lahan sawah lebak.

2.1.2 Variabel yang mempengaruhi kondisi tanah sawah

Tanah yang subur dan memiliki kandungan organik yang cukup, sangat mempengaruhi pertumbuhan dan perkembangan tanaman padi. Untuk mencapai kondisi tanah sawah yang baik maka perlu diketahui variabel yang mempengaruhi kondisi tanah. Variabel yang mempengaruhi kondisi tanah sawah antara lain tingkat keasaman tanah(pH), tingkat kelembaban tanah, dan suhu tanah serta persawaan.

1. Kadar keasaman tanah (pH)

pH tanah adalah kadar tingkat keasaman atau kebasaan suatu tanah yang dapat diukur dengan skala pH antara 0 sampai 14. Jika angka skala pH dari tanah yang diuji menunjukkan nilai kurang dari 7, maka tanah yang diuji bersifat asam. Sedangkan tanah yang memiliki nilai pH lebih dari 7 merupakan tanah bersifat basa.

Tanah dengan tingkat keasaman yang tinggi, unsur kalsium, magnesium, dan fosfor akan terikat secara kimiawi. Hal ini menyebabkan tanaman tidak dapat menyerap unsur-unsur tersebut. Kondisi tersebut dapat mengakibatkan tanaman tidak tumbuh secara optimal dan menghasilkan produktifitas yang rendah. Untuk menyesuaikan nilai pH tanah, penggunaan dolomit dengan dosis tertentu dapat diberikan dengan dosis yang sesuai kebutuhan³[3].

2. Tingkat kelembaban tanah

Komponen mikro yang mempengaruhi kondisi tanah sawah, dan pertumbuhan tanaman padi sangat dipengaruhi oleh kelembaban tanah. Kelembaban yang berlebihan dapat menimbulkan penyerapan air yang berlebihan oleh akar tanaman yang berakibat pada busuknya tanaman, sedangkan kekurangan kelembaban dapat menimbulkan pertumbuhan tanaman yang tidak optimal karena kurangnya air untuk kebutuhan tanaman. Setiap jenis tanaman memiliki tingkat kelembaban tanah ideal yang berbeda-beda. Kelembaban tanah juga mempengaruhi aktivitas mikroorganisme dalam tanah dan ketersediaan unsur hara[4].

3. Suhu tanah persawahan

Aktivitas biologi tanah seperti organisme tanah, sangat dipengaruhi oleh temperatur tanah. Proses penguraian bahan organik, reaksi kimia, dan bahan induk tanah juga sangat dipengaruhi oleh suhu tanah. Proses pertumbuhan tanaman padi juga dipengaruhi oleh suhu tanah melalui tingkat kelembaban tanah. Suhu dalam tanah dapat berubah-ubah (fluktuasi), bergantung pada perubahan sumber energi, waktu dan tingkat kedalaman tanah. Temperatur tanah

³<https://mitalom.com/pengaruh-derajat-keasaman-tanah-ph-terhadap-tanaman/>

dipagi hari relatif lebih kecil dibandingkan disiang hari. Hal ini disebabkan oleh pancaran radiasi matahari yang diterima oleh permukaan tanah lebih besar pada siang hari.

Suhu tanah juga berbeda-beda bergantung pada tingkat kedalaman tanah (10cm, 20cm, dan 30cm). Semakin dalam tanah, maka semakin rendah suhunya. Perbedaan suhu ini dipengaruhi oleh transfer panas dari pancaran radiasi matahari, yang belum mencapai kedalaman tertentu. Transfer panas yang dipancarkan radiasi matahari dirambatkan ke lapisan tanah yang lebih dalam secara konduksi. Faktor lain yang mempengaruhi perubahan suhu tanah adalah sumber energi panas yang diterima dari matahari. Perubahan suhu ini dipengaruhi oleh iklim cuaca, keadaan tanah dan bentuk topografi (bentuk permukaan tanah)⁴.

2.1.3 Klasifikasi kondisi tanah sawah

Untuk mengetahui kondisi tanah sawah maka diperlukan pengukuran variable yang mempengaruhi kondisi tanah sawah seperti kadar keasaman tanah sawah(pH), tingkat kelembaban tanah, dan suhu tanah. Tanaman padi memiliki nilai ideal tertentu untuk masing-masing variabel, bergantung dari jenis tanah sawahnya.

Nilai keasaman tanah sawah(pH) yang ideal untuk tanaman padi adalah 7, yang mengindikasikan bahwa tanah sawah bersifat netral, tidak asam maupun basa. Namun, beberapa jenis tanaman padi masih dapat tumbuh dan berkembang pada tanah dengan pH yang asam, dengan tingkat toleransi nilai pH maksimal 5.

Selain kadar keasaman, variabel lain yang mempengaruhi pertumbuhan tanaman padi adalah kelembaban. Berdasarkan pengukuran yang dilakukan menggunakan metode SRI (*System of Rice Intensification*) dan model Algoritma Genetika, didapatkan hasil kelembaban tanah optimum untuk tanaman padi adalah 0.622 (basah) atau setara dengan 62%⁵. Namun, hal ini tidak berlaku untuk semua jenis tanaman. Tingkat kelembaban yang dibutuhkan untuk setiap tanaman berbeda-beda, sebagaimana yang diperlihatkan pada Tabel 2.1 berikut.

Tabel 2.1: Tabel kelembaban ideal tanaman

Jenis tanaman	Kelembaban ideal	Jenis tanaman	Kelembaban ideal
Tomat	40%-60%	Kubis	40%-60%
Terong	40%-60%	Bawang	30%-50%
Kacang tanah	30%-50%	Kentang	30%-50 %
Mentimun	50%-70%	Bunga kol	40%-60 %
Wortel	30%-50%	Buah-buahan	30%-50 %
Seledri	40%-60%	Cabai	40%-60 %

Suhu merupakan salah satu faktor lain yang mempengaruhi pertumbuhan tanaman padi. Pada dataran rendah suhu udara yang ideal untuk tanaman padi adalah sekitar 22°C sampai 26°C, sedangkan untuk dataran tinggi suhu udara yang ideal adalah sekitar 18°C sampai 22,5°C. Suhu tanah yang kurang dari 10°C akan menghambat perkembangan mikroba tanah dan menghambat penyerapan hara oleh tanah. Sedangkan suhu yang melebihi 40°C akan menyebabkan mikroba dalam tanah menjadi tidak aktif, kecuali untuk mikroorganisme tertentu. Pada umumnya, tingkat aktivitas optimum dari organisme tanah memerlukan suhu tanah sekitar 18°C sampai 30°C⁶.

⁴<https://www.slideshare.net/iqrinhayamada/suhu-tanah>

⁵<https://repository.ipb.ac.id/bitstream/handle/123456789/69941/JI2014-Vol9%2cNo1.pdf?sequence=1&isAllowed=y>

⁶<https://www.slideshare.net/iqrinhayamada/suhu-tanah>

2.1.4 Metode pemantauan kualitas tanah sawah

Terdapat beberapa cara untuk melakukan pemantauan kualitas tanah sawah. Metode yang masih sering dilakukan adalah *sampling* atau pemanfaatan IoT (Internet of Things). Perbedaan kedua metode ini terletak pada biaya, waktu, dan tempat. Pemantauan yang dilakukan dengan melakukan pengujian kondisi tanah sawah menggunakan metode *sampling* memiliki ketepatan yang lebih akurat, namun membutuhkan waktu yang lebih lama dan biaya yang lebih mahal.

- *Sampling*

Sampling adalah proses pengambilan atau memilih n buah elemen dari populasi yang berukuran N (Lohr, 1999). *Sampling* dalam pemantauan kualitas tanah dilakukan dengan cara mengambil tanah sawah yang diteliti menggunakan ring sampel, bor, dan pisau lapang. Tanah sawah yang dijadikan *sampling* akan dibawa ke Labotarium Tanah dan Lingkungan untuk diuji kualitasnya. Uji labotarium dilakukan menggunakan metode tertentu kualitas tanah ditetapkan dengan metode yang disajikan seperti Tabel 2.2 ⁷.

Tabel 2.2: Parameter dan Metode Analisis Kualitas Tanah (Lal,1994)

Parameter	Satuan	Metode
Sifat Fisik		
Kadar Air Tanah	%	Gravimetri
Tekstur Tanah	%	Metode Pipet
Berat Volume (bulk density)	g/cm ³	Ring sampler
Porositas	%	Ring sampler
Sifat Kimia		
pH		Potensiometri H ₂ O 1:2,5
N-Total	%	Destilasi Makro Kjeldahl
P - Tersedia	mg/kg	Bray-1
K - Tersedia	mg/kg	Bray-1
KTK	me/100g	Ekstraksi NH ₄ OAc 1 N pH 7
KB - Tersedia	%	Ekstraksi NH ₄ OAc 1 N pH 7
Sifat Biologi		
C-organik	%	Walkley & Black
Total Mikroba	cfu/g tanah mg C-C _o 2	Plate Count
Respirasi	mg C-CO ₂ /kg	Evolusi CO ₂
C-biomassa	mg CO ₂ /kg	Simulasi Respirasi

- Pemanfaatan IoT (*Internet of Things*)

Perkembangan teknologi yang begitu pesat membantu berbagai jenis penelitian, termasuk pemantauan kondisi tanah sawah. Pemanfaatan IoT untuk pemantauan kualitas tanah sawah sudah cukup sering dilakukan terutama di Indonesia. Aplikasi berbasis android maupun desktop yang terintegrasi dengan sensor yang disebar di area persawahan, membantu para petani dalam mendapatkan informasi tanah sawah yang mereka kelola. Pemanfaatan IoT memungkinkan pengguna untuk mendapatkan data secara *real-time* dan tanpa perlu mengeluarkan biaya yang mahal. Namun data yang didapatkan tidak akan seakurat metode *sampling* yang diuji di labotarium.

⁷https://simdos.unud.ac.id/uploads/file_penelitian_1_dir/bbc895c97c05ccd2a106b91f9843705b.pdf

2.2 WSN (*Wireless Sensor Network*)

WSN (*Wireless Sensor Network*) merupakan jaringan sensor nirkabel yang mengacu pada sekumpulan node sensor yang saling terhubung. Node sensor akan disebar dalam bentuk spasial pada area yang ingin dilakukan pengukuran atau *sensing*. WSN juga berguna untuk memantau hasil sensing kondisi suatu lingkungan yang dilakukan oleh sensor. Selain itu WSN juga dapat mengatur data yang akan dikirimkan dan dikumpulkan di *base station*. Setiap node sensor memiliki radio *transceiver* yang digunakan untuk berkomunikasi dan mengirimkan data ke lokasi pusat. Bentuk radio *transceiver* untuk setiap node menggunakan antena internal ataupun antena eksternal.

2.2.1 Pemanfaatan *Wireless Sensor Network*

Saat ini, teknologi WSN tidak hanya dimanfaatkan untuk keperluan ilmu komputasi saja. Penggunaan teknologi WSN telah digunakan untuk melakukan pemantauan ataupun pengukuran, diberbagai bidang ilmu pengetahuan. Selain digunakan untuk membantu penelitian di bidang ilmu pengetahuan, teknologi WSN juga dimanfaatkan oleh pihak militer dan industri.

1. Bidang Kesehatan

Pemanfaatan WSN di bidang kesehatan telah banyak diterapkan. Beberapa penerapan WSN di bidang kesehatan adalah pemanfaatan IoT (*Internet of Things*) untuk perangkat medis untuk mendeteksi kondisi tubuh manusia. Selain itu, WSN juga dimanfaatkan untuk melakukan pemantauan dan diagnosa kesehatan pasien secara *real-time*.

2. Pemantauan Lingkungan

Untuk pemantauan lingkungan, WSN memiliki banyak pengaplikasian yang berbeda di berbagai jenis lingkungan. Beberapa contoh pemanfaatan WSN untuk pemantauan lingkungan antara lain, pemantauan polusi udara, mendeteksi kebakaran hutan, mendeteksi longsor, mendeteksi ancaman bencana alam, dan lain-lain.

3. Bidang Agrikultur

Di bidang agrikultur WSN digunakan untuk meningkatkan produktivitas hasil panen. Penerapan WSN dapat dilakukan untuk melakukan pengamatan, pengukuran, dan respon terhadap variable tanah. Pemantauan tersebut bertujuan untuk meningkatkan ataupun menjaga kualitas hasil panen, dari pemanfaatan data lingkungan pertanian.

4. Bidang Infrastruktur

Wireless sensor network juga digunakan dalam pembangunan infrastruktur. Pemanfaatan WSN biasa digunakan untuk melakukan pengamatan kondisi pembangunan infrastruktur, baik dari segi bangunan maupun geografis. Penerapan WSN untuk membantu pembangunan infrastruktur dilakukan secara berkala dan *real-time* untuk menjaga kondisi lingkungan selama pemabangunan.

5. Bidang Militer

Salah satu contoh pemanfaatan WSN di bidang militer adalah Wide Area Tracking System (WATS). WATS merupakan prototipe jaringan yang menggunakan teknologi WSN untuk mendeteksi ancaman nuklir. Pemanfaatan WSN lainnya di bidang militer adalah peluru kendali, yang memiliki sistem pengendali otomatis untuk mencari sasaran berdasarkan suhu temperatur.

6. Bidang Planologi

Dibidang planologi, salah satu pemanfaatan WSN digunakan untuk memantau jumlah kendaraan pada suatu area untuk mengatasi masalah kemacetan, ataupun membantu proyek

rekayasa arus lalu-lintas. Pemanfaatan WSN memungkinkan pengiriman informasi jumlah kendaraan secara *real-time* ke pengguna.

7. Bidang Industri

Pemanfaatan WSN di bidang industri cukup beragam, bergantung jenis barang yang diproduksinya. Untuk industri otomotif, penggunaan WSN digunakan sebagai fitur keamanan ataupun kenyamanan. Salah satu pemanfaatannya adalah untuk mendeteksi lingkungan sekitar kendaraan, untuk mencegah terjadinya benturan yang disebabkan oleh *blind-spot*.

2.2.2 Klasifikasi Node

Titik komunikasi atau titik distribusi komunikasi yang saling terhubung dapat didefinisikan sebagai node. Dari sudut pandang jaringan, node dapat didefinisikan sebagai anggota jaringan yang dapat saling berkomunikasi untuk menerima atau menghasilkan data, ataupun keduanya⁸. Pada *Wireless Sensor Network* terdapat 3 jenis node yang memiliki karakteristik dan fungsional yang berbeda. Variasi node yang terdapat pada WSN antara lain node sensor, router, dan sink node (*base station*).

1. Node Sensor

Node sensor adalah node dalam jaringan yang digunakan untuk melakukan pemrosesan, pengumpulan informasi yang didapatkan dari hasil *sensing*, dan berkomunikasi dengan node lain yang terhubung dalam jaringan. Node sensor juga berfungsi untuk melakukan pengukuran kondisi suatu lingkungan berdasarkan hasil *sensing* yang diterimanya untuk dikirimkan ke *gateway*.

2. Router

Router adalah node yang menjadi perantara antara node sensor dengan node sensor lainnya, agar node dapat saling berkomunikasi. Router dapat menerima paket yang didapatkan dari suatu node untuk diteruskan ke node lain.

3. Sink Node

Sink node atau *Base station* adalah komponen WSN yang bertindak (*attach*) sebagai *gateway* antara node sensor dengan *notebook*. *Base station* juga merupakan bagian terakhir dari rangkaian WSN yang berfungsi untuk meneruskan data hasil sensing node sensor ke *notebook* atau server. Untuk perangkat keras arduino, komponen yang digunakan sebagai *base station* adalah Raspberry.

2.2.3 Komponen Node Sensor

Node sensor terdiri dari komponen-komponen yang saling terhubung dengan fungsionalnya masing-masing. Komponen utama node sensor antara lain *controller*, *transceiver*, *memory*, *power source*, dan sensor.

- *Controller*

Controller bertindak sebagai pengontrol fungsionalitas komponen-komponen lain pada node sensor. Selain itu, *controller* juga memiliki tugas untuk memproses data. Mikrokontroler menjadi alternatif yang paling umum digunakan untuk melakukan pemrosesan pada node sensor. Hal ini dikarenakan mikrokontroler lebih mudah untuk dihubungkan dengan perangkat lain. Mikrokontroler juga menggunakan konsumsi daya yang lebih rendah dan biaya yang lebih murah.

⁸<https://www.nesabamedia.com/pengertian-node/>

- *Transceiver*

Transceiver adalah perangkat yang digunakan untuk menerima dan memancarkan sinyal dalam jaringan. Dengan *transceiver*, node sensor dapat terhubung dengan jaringan dan berkomunikasi dengan node sensor lainnya. Teknologi yang umum digunakan untuk terhubung dalam jaringan adalah *baseband*.

- *Memory*

RAM (Random Access Memory) merupakan perangkat *volatile* yang digunakan sebagai tempat penyimpanan data sementara. Data yang diterima dari hasil *sensing* oleh sensor, akan disimpan pada RAM. Namun, penyimpanan ini hanya bersifat sementara, sebelum data tersebut dikirimkan ke *base station*. *Memory* yang terdapat pada node sensor bersifat *volatile*, hal ini dapat mengakibatkan hilangnya data jika node sensor mati.

- *Power source*

Energi yang dibutuhkan untuk mengoperasikan fungsionalitas seluruh komponen berasal dari *power source* atau *power supply*. Penyediaan energi ini dapat memiliki dua jenis metode, yaitu *storing energy* atau *energy scavenging*. Pada metode *storing energy*, node sensor menggunakan baterai sebagai sumber energi utamanya. Sedangkan pada pendekatan *energy scavenging*, node sensor menggunakan perubahan energi sebagai sumber energi utamanya. Energi pada metode *energy scavenging* didapatkan setelah mengonversi pemanfaatan energi cahaya matahari, angin, ataupun air menjadi listrik untuk mengoperasionalkan node sensor.

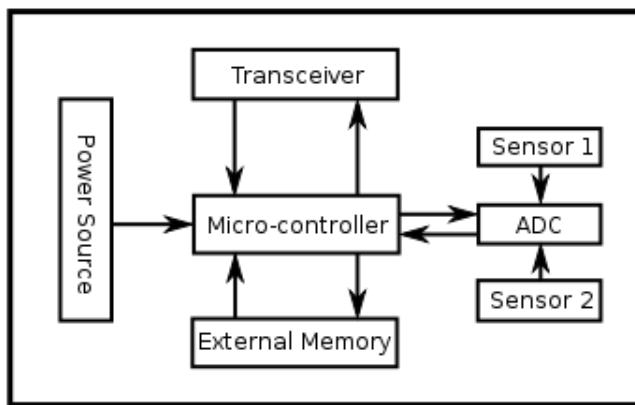
- *Analog to Digital Converter (ADC)*

ADC adalah perangkat kecil berbentuk chip yang berfungsi untuk mengubah sinyal yang dikirimkan oleh sensor dari analog menjadi digital. *Analog to Digital Converter* juga dapat berbentuk suatu modul yang terdiri dari barisan kode yang di *upload* ke node sensor.

- Sensor

Sensor merupakan perangkat yang digunakan untuk melakukan *sensing* dan pengukuran, terhadap suatu kondisi pada lingkungan yang diamati. Sensor juga dapat merespon terhadap perubahan suatu kondisi lingkungan seperti suhu ataupun kelembaban.

Secara umum sebuah node sensor memiliki struktur seperti yang ditunjukkan pada Gambar 2.1.

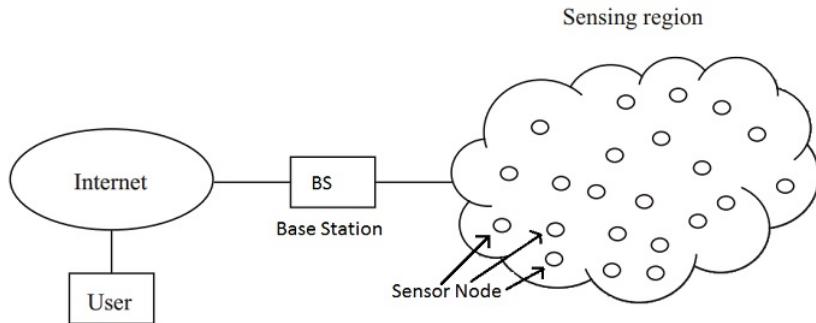


Gambar 2.1: Node sensor

2.2.4 Arsitektur Wireless Sensor Network

Arsitektur pada WSN terbagi menjadi dua jenis, yaitu arsitektur *flat* dan arsitektur *hirarkikal*. Perbedaan dua jenis arsitektur ini terdapat pada cara node sensor dalam berkomunikasi. Secara

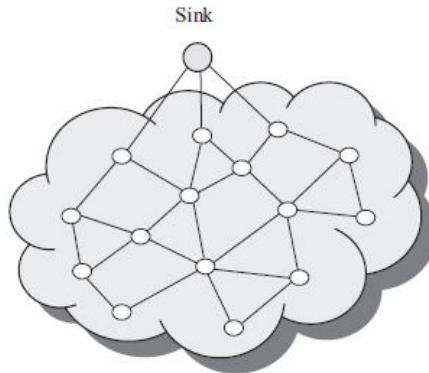
sederhana, pada arsitektur *flat*, node yang disebar dapat langsung mengirimkan data hasil *sensing* ke *base station*. Sedangkan pada arsitektur *hirarkikal*, node yang disebar harus mengirimkan data ke *cluster head* terlebih dahulu, sebelum diteruskan ke *base station* [5].



Gambar 2.2: Arsitektur *Wireless Sensor Network*

1. *Flat*

Pada arsitektur *flat*, seluruh node sensor yang disebar memiliki tugas yang sama. Node-node tersebut melakukan *sensing*, dan mengirimkan data ke *base station*. Data hasil *sensing* yang didapatkan oleh node sensor dapat langsung diteruskan ke *base station* tanpa memerlukan perantara.



Gambar 2.3: Arsitektur *Flat*

2. *Hirarkikal*

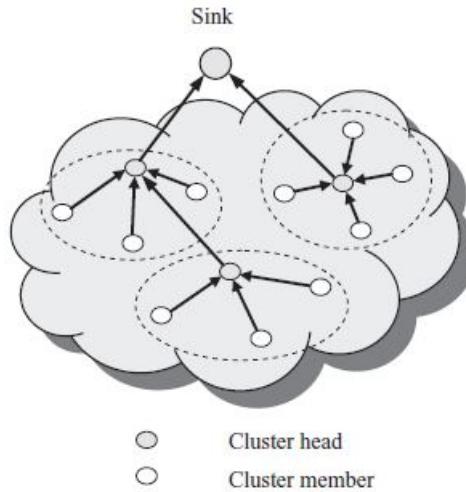
Berbeda dengan arsitektur *flat* pada arsitektur *hirarkikal* pengiriman hasil *sensing* oleh node sensor tidak dapat secara langsung dikirimkan ke *base station*. Arsitektur *Hirarkikal* mengelompokan node-node yang disebar menjadi beberapa kelompok yang disebut dengan *cluster*. Tiap *cluster* terdiri dari sebuah *cluster head* dan *cluster member*. *Cluster head* bertindak sebagai penerima data hasil sensing dari *cluster member*, untuk diteruskan ke *base station*.

Pada arsitektur *hirarkikal*, jenis komunikasi dapat dipilih berdasarkan jarak antara *cluster head* ke *cluster member*. Jenis-jenis komunikasi yang terdapat pada arsitektur hirarkikal adalah *single hop* dan *multi hop*. Perbedaan jenis komunikasi ini terletak pada jumlah *hop* atau 'lompatan' yang dibutuhkan paket untuk mencapai tujuan.

- *Single hop*

Pada jaringan *single hop*, data yang dikirimkan oleh *cluster member* hanya membutuhkan satu kali lompatan (*hop* tunggal) untuk mencapai *cluster head*. Dengan kata lain, data

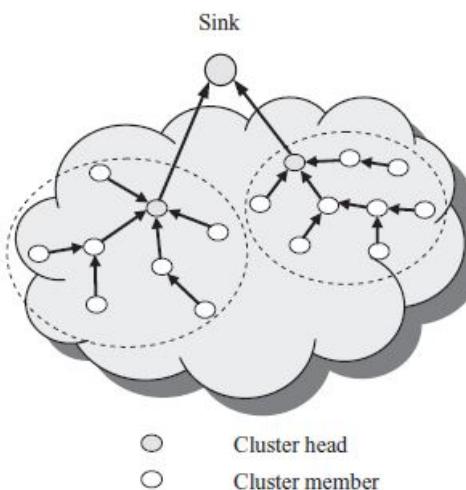
yang diterima oleh *cluster member* dapat langsung diterima oleh *cluster head*, untuk diteruskan ke *base station*.



Gambar 2.4: Arsitektur *Single Hop*

- *Multi hop*

Pada jaringan *multi hop*, data yang dikirimkan oleh *cluster member* membutuhkan lebih dari satu kali lompatan untuk mencapai *cluster head*. Data yang diterima oleh *cluster head* dari *cluster member*, akan diteruskan ke *cluster member* lain yang memiliki jarak lebih dekat dengan *cluster head*. Hal ini akan dilakukan berulang-ulang sampai paket sampai di *cluster head* untuk diteruskan ke *base station*.



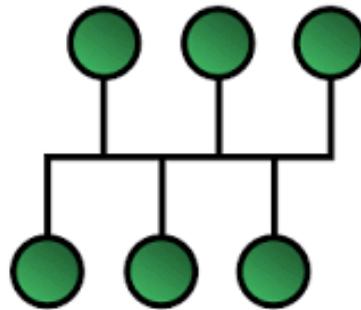
Gambar 2.5: Arsitektur *Multi Hop*

2.2.5 Topologi *Wireless Sensor Network*

Topologi digunakan untuk menjelaskan hubungan antar node, dan dasar penyusunan jaringan secara geometris. Terdapat beberapa jenis topologi dalam WSN, yang tiap jenisnya memiliki fungsionalitasnya masing-masing. Pemilihan jenis topologi dipengaruhi oleh tujuan, skala jaringan, kondisi lingkungan, dan hal lainnya. Beberapa topologi jaringan pada WSN antara lain topologi *bus*, topologi *tree*, topologi *star*, topologi *ring*, topologi *point-to-point*, dan topologi *mesh*.

1. Topologi *Bus*

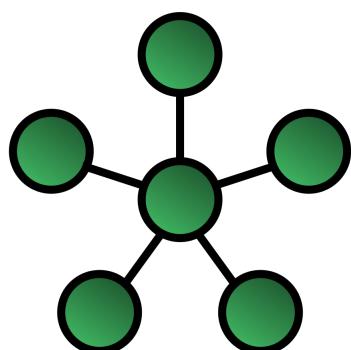
Pada topologi bus, node-node sensor akan terhubung pada sebuah jalur. Jalur ini digunakan node-node tersebut untuk saling berkomunikasi. Komunikasi node pada topologi bus bersifat satu arah, yang mengartikan bahwa komunikasi node dilakukan secara bergantian. Komunikasi dilakukan dengan cara paket yang ingin dikirimkan oleh node pengirim akan di-*broadcast* ke node sensor lain yang berada dalam jaringan. Node-node lain akan menerima paket yang dikirimkan, namun hanya sensor yang dituju yang dapat memproses hasil paket yang diterima. Topologi ini mudah udah diimplementasikan, namun untuk jaringan dengan skala yang besar, topologi ini dapat mengakibatkan kerugian dari sisi keamanan maupun *bandwidth*.



Gambar 2.6: Topologi *Bus*

2. Topologi *Star*

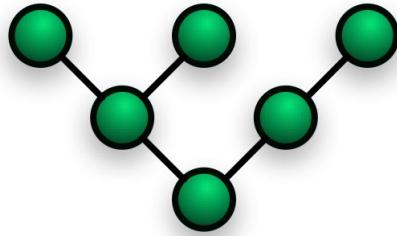
Pada topologi *star* node-node yang terhubung tidak dapat secara langsung berkomunikasi dengan node lainnya. Node-node dalam jaringan terhubung dengan *main device* (*central node*). Sifat *main device* terbagi menjadi dua, pasif dan aktif. *Main device* yang pasif hanya bertugas sebagai perantara komunikasi saja, sehingga pesan yang dikirimkan oleh node yang dikirimkan ke *central node* akan diterima oleh seluruh node lain dalam jaringan. Pada *main device* yang bersifat aktif bertindak sebagai pengontrol komunikasi antar node, sehingga pesan yang dikirimkan oleh node pengirim akan diteruskan oleh *central node* ke node tujuan saja. Bentuk jaringan topologi *star* meminimalisir adanya gangguan jika ada salah satu node yang mati. Namun, jika *central node* pada jaringan ini mati atau mengalami gangguan, maka seluruh komunikasi node juga akan terhenti.



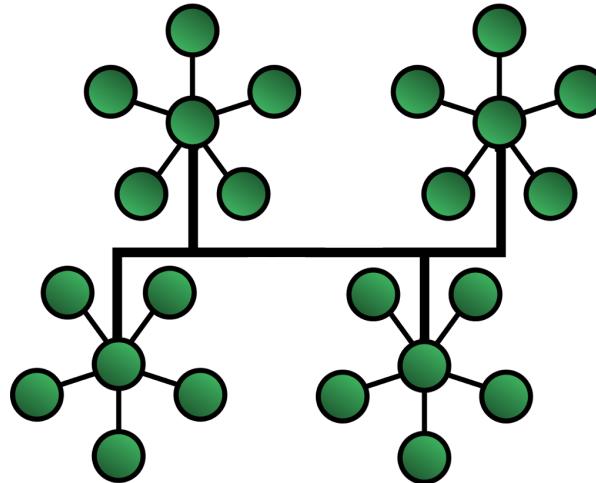
Gambar 2.7: Topologi *Star*

3. Topologi *Tree*

Pada topologi *tree*, rangkaian jaringan disusun secara hirarki dengan sebuah node yang berada pada level teratas yang disebut dengan *root node*. Root note juga bertindak sebagai komunikasi router utama. Node-node yang berada di bawah root node disebut dengan *children* atau *parent*, bergantung dari kondisi node lain yang terhubung dengan node tersebut. Node yang terhubung dengan node lain yang memiliki *level* lebih rendah disebut dengan *parent*, sedangkan node yang tidak memiliki hubungan dengan node lain dengan *level* lebih rendah disebut dengan *children*. Penggunaan topologi lebih mudah untuk melakukan identifikasi dan meminimalisir kesalahan, namun hirarki pada topologi ini dapat mengakibatkan putusnya komunikasi pada *children* jika salah satu *parent* putus atau mati. Kelemahan lain dari topologi ini adalah sulit untuk dikonfigurasi apabila *level* pada rangkaian jaringan sudah cukup besar seperti pada Gambar 2.9.



Gambar 2.8: Topologi *Tree*

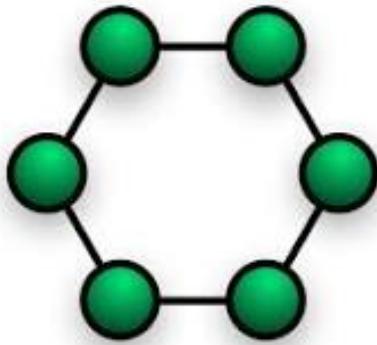


Gambar 2.9: Jaringan *Star* pada Topologi *Tree*

4. Topologi *Ring*

Berbeda dengan topologi *star*, jaringan topologi ring tidak memiliki *central node* di dalamnya. Jaringan topologi *ring* berbentuk rangkaian node yang saling terhubung ke dua node lainnya (node yang paling dekat) dan membentuk lingkaran seperti cincin. Pada topologi ini, pesan node sensor pengirim akan diteruskan oleh node yang tetangganya. Proses komunikasi pengiriman pesan ini akan dilakukan terus menerus, sampai pesan yang dikirimkan oleh node pengirim diterima oleh node yang dituju. Proses komunikasi yang dilakukan pada jaringan ini mengikuti jalur melingkar (searah jarum jam ataupun sebaliknya) yang terbentuk seperti

pada Gambar 2.10. Bentuk jaringan yang saling bekergantungan antara node dengan node tetangganya, dapat mengakibatkan putusnya seluruh komunikasi apabila salah satu node dalam jaringan mati.



Gambar 2.10: Topologi *Ring*

5. Topologi *Linear*

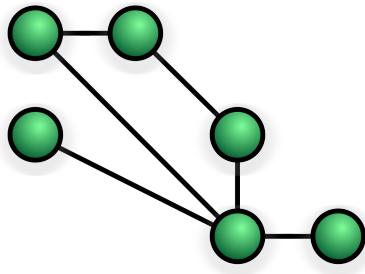
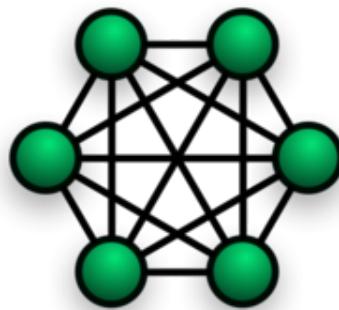
Topologi *linear* memiliki bentuk jaringan yang paling sederhana dibandingkan dengan topologi lainnya. Topologi ini menghubungkan node dengan node lainnya secara linear. Topologi *linear* juga merupakan pengembangan dari topologi *point-to-point* yang hanya membutuhkan dua buah node. Bentuk jaringan topologi linear yang sederhana membuatnya lebih mudah dikembangkan dan murah. Namun, topologi ini memiliki kekurangan yang sama dengan topologi *ring*. Matinya salah satu node dalam jaringan, dapat berakibat putusnya seluruh komunikasi jaringan.



Gambar 2.11: Topologi *Linear*

6. Topologi *Mesh*

Terdapat dua jenis bentuk jaringan pada topologi *mesh*, yaitu *partially connected mesh* dan *fully connected mesh*. Perbedaan kedua jaringan ini terletak pada hubungan antar node. Pada *partially connected mesh*, suatu node dapat terhubung dengan node lainnya secara bebas. Pada *fully connected mesh*, setiap node harus terhubung dengan seluruh node lain yang berada di dalam jaringan. Bentuk jaringan pada topologi ini memungkinkan minimalnya gangguan yang terjadi apabila terdapat salah satu node mati di dalam jaringan. Namun, besarnya bentuk jaringan ini membuat sulitnya melakukan perawatan atau *maintenance*.

Gambar 2.12: *Partially connected mesh*Gambar 2.13: *Fully connected mesh*

2.2.6 Protokol *Wireless Sensor Network*

Protokol jaringan pada *Wireless Sensor Network* biasa disebut dengan WSN *protocol stack*. Protokol jaringan ini terdiri dari lima *layer* atau lapisan. *Layer-layer* yang menyusun protokol jaringan WSN, terdiri dari *application layer*, *transport layer*, *network layer*, *data link layer* dan *physical layer*.

- *Application Layer*

Application Layer merupakan lapisan yang bertindak sebagai penyedia antar muka aplikasi. Aplikasi yang dimaksud adalah aplikasi yang digunakan untuk melakukan komunikasi di dalam jaringan. *Application Layer* terdiri dari sejumlah protokol yang disebut dengan *application layer protocol*. Protokol ini digunakan untuk aplikasi sensor *network*.

- *Transport Layer*

Transport layer memiliki tugas untuk mengirimkan data yang bersifat *reliable* antar node sensor ataupun node sensor ke *base station*. Cara pengiriman yang dilakukan oleh *transport layer* terbagi menjadi dua jenis, yaitu *upstream* dan *downstream*. Perbedaan cara pengiriman ini dapat dilihat berdasarkan sumber pengirim dan penerima data. Pada *upstream*, node sensor bertindak sebagai pengirim data, dan *base station* bertindak sebagai penerima data. *Downstream* merupakan kebalikan dari *upstream*. Pada *downstream*, yang bertindak sebagai pengirim data adalah *base station*, sedangkan yang bertindak sebagai penerima data adalah node sensor.

Masing-masing cara pengiriman, memiliki kebutuhan *reliability* yang berbeda-beda. Pada jenis pengiriman *upstream*, kebutuhan *reliability* tidak terlalu dibutuhkan. Hal ini disebabkan data yang dikirimkan oleh node sensor ke *base station* dilakukan secara berulang-ulang, sehingga data yang tidak *reliable* dapat ditoleransi. Berbeda dengan jenis pengiriman *downstream*,

kebutuhan *reliability* sangat dibutuhkan. Pada pengiriman *downstream*, jika data yang dikirimkan tidak *reliable* maka aplikasi tidak dapat dijalankan.

- *Network Layer*

Network Layer merupakan lapisan ketiga dari protokol WSN. Network layer bertugas untuk menentukan jenis komunikasi antar node, *single hop* atau *multi hop*. *Network layer* juga merupakan lapisan yang menyediakan jalur komunikasi jaringan. Data yang dikirimkan melalui lapisan ini disebut dengan paket. Paket akan dikirimkan melalui jalur yang telah disediakan dan dikendalikan oleh *network layer*.

- *Data Link Layer*

Data Link layer adalah lapisan yang bertindak untuk melakukan kontrol terhadap akses media yang dilakukan oleh node sensor, dan mencegah adanya paket yang bertabrakan. Proses kontrol ini juga biasa disebut dengan *Medium Access Control* (MAC), sedangkan jaringan yang digunakan mencegah tabrakan antar paket disebut dengan *Carrier Sense Multiple Access with Collision Avoidance*(CSMA-CA). Data link layer juga memiliki tugas untuk melakukan *multiplexing* pada aliran data, membentuk *data frame*, mendekripsi *data frame*, *medium access*, dan mendekripsi kesalahan yang mungkin terjadi saat melakukan pengiriman data.

- *Physical Layer*

Physical layer merupakan lapisan yang bertindak sebagai *converter* yang mengubah bit stream dari data link layer menjadi sinyal yang cocok untuk media komunikasi yang tersedia. Konversi ini dilakukan agar transmisi dapat dilakukan melalui *transceiver*. *Radio Frequency* (RF) digunakan untuk melakukan konferensi tersebut. RF cukup sering digunakan pada node sensor, dikarenakan biaya yang murah, dan ukuran perangkat yang kecil.

Selain lima *layer* yang disebutkan, terdapat tiga *layer* lain yang terletak di bidang lintas lapisan yang terdiri dari *task management plane*, *mobility management plane*, dan *power management plane*.

- *Power management plane*

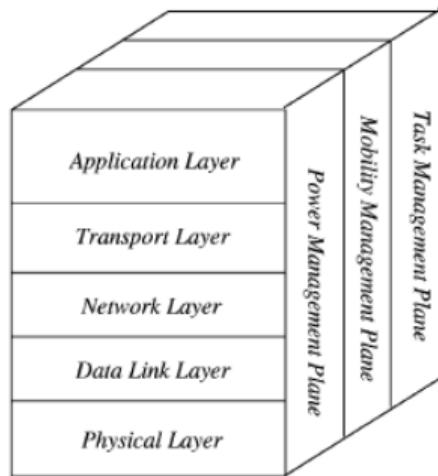
Power management plane bertindak sebagai pengontrol penggunaan sumber daya atau energi untuk setiap node sensor. Pengaturan sumber daya oleh power management plane dilakukan pada saat node sensor melakukan *sensing*, *processing*, ataupun saat *transmission* dan *reception* (mengirim dan menerima data).

- *Connection management plane*

Connection management plane berfungsi untuk melakukan konfigurasi terhadap segala sesuatu yang berkaitan dengan koneksi antar node sensor. Konfigurasi atau rekonfigurasi biasa dilakukan saat terjadi perubahan topologi pada wireless sensor network.

- *Task management plane*

Task management plane memiliki tugas untuk melakukan pembagian atau distribusi tugas (task) untuk setiap node sensor. Pembagian tugas ini dilakukan agar penggunaan energi dapat dioptimalkan dan lebih efisien.



Gambar 2.14: Protokol Jaringan

2.3 Zigbee

Zigbee merupakan standar spesifikasi untuk jaringan protokol yang menggunakan radio digital. Protokol zigbee berbasiskan pada standar IEEE 802.15.4-2003 untuk jaringan nirkabel tingkat rendah. Kebutuhan konsumsi daya yang rendah untuk zigbee beroperasi, menjadi solusi untuk perangkat yang memiliki persediaan daya terbatas. Oleh karena itu, zigbee biasa digunakan dalam perangkat yang menggunakan baterai. Zigbee juga biasa digunakan untuk aplikasi pengontrolan atau pemantauan yang bersifat nirkabel, seperti alarm peringatan kebakaran (asap) ataupun diteksi penyusup. Zigbee juga merupakan spesifikasi standar yang digunakan dalam jaringan nirkabel *mesh*.

Pada umumnya, chip yang terpasang pada zigbee telah terintegrasi dengan mikrokontroler dan tidak membutuhkan *memory* yang besar. Namun, zigbee hanya dapat mengirimkan komunikasi dengan latensi yang rendah (*low-latency communication*). Rendahnya latensi komunikasi yang dimiliki oleh zigbee menyebabkan kurang tepatnya penggunaan zigbee untuk perangkat yang memerlukan kecepatan transfer data yang tinggi.

2.3.1 Jenis-jenis perangkat zigbee

Terdapat tiga jenis perangkat zigbee antara lain:

- ZigBee Coordinator (ZC)

Zigbee Coordinator merupakan perangkat yang digunakan sebagai repositori informasi jaringan maupun keamanan. Zigbee Coordinator juga merupakan perangkat pertama yang beroperasi ketika jaringan diaktifkan. Hal ini dikarenakan Zigbee Coordinator membentuk dasar dari jaringan dan menghubungkan jaringan-jaringan lainnya.

- ZigBee Router (ZR)

Zigbee Router merupakan perangkat yang bertindak sebagai perantara dalam pengiriman atau penyampaian data. Selain berfungsi untuk meneruskan data antar perangkat, Zigbee router juga bertindak dalam menjalankan fungsi aplikasi.

- ZigBee End Device (ZED)

Zigbee End Device adalah perangkat yang berperan sebagai parent node. Hal tersebut menyebabkan Zigbee End Device tidak dapat mengirim ulang ataupun meneruskan data dari perangkat lain. Peran zigbee sebagai parent node terjadi pada koordinator maupun router.

2.4 Arduino

Arduino adalah perangkat keras pengendali mikrokontroler yang bersifat *open-source* untuk membangun perangkat digital. Arduino dirancang untuk menjadi alternatif bagi pelajar dalam membangun perangkat digital yang dapat berkomunikasi dengan lingkungan, dengan harga yang lebih murah. Mikrokontroler arduino dapat diprogram menggunakan bahasa pemrograman C. Kompiler dan Integrated Development Environment (IDE) untuk proyek yang menggunakan arduino-pun juga sudah banyak tersedia.

2.4.1 Jenis-jenis arduino

Arduino memiliki banyak jenis mikrokontroler dengan spesifikasi yang berbeda-beda. Jumlah pin *input/output*, jenis chip, dan modul yang tersedia bergantung pada tipe arduino yang digunakan.

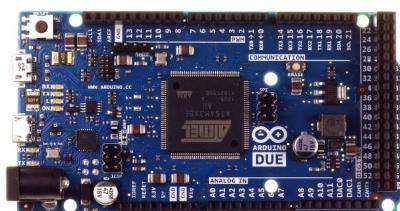
1. Arduino Uno



Gambar 2.15: Arduino Uno

Arduino uno merupakan mikrokontroler yang paling populer dan paling sering digunakan. Referensi yang membahas mengenai arduino uno juga sudah cukup banyak, sehingga mudah untuk dipelajari untuk para pemula. Arduino uno menggunakan ATMEGA328P sebagai mikrokontrolernya, dan dilengkapi dengan 14 pin *input/output* digital dan 6 pin *input analog*. Untuk pemrograman, arduino uno menggunakan koneksi USB type B. Arduino Uno R3 merupakan versi terakhir dari arduino uno sampai saat ini.

2. Arduino Due



Gambar 2.16: Arduino Due

Arduino Due menggunakan mikrokontroler yang lebih tinggi dibandingkan arduino uno, yaitu 32bit ARM Cortex CPU. Mikrokontroler yang digunakan pada arduino due memungkinkan pengguna untuk tetap menggunakan bahasa pemrograman yang *compatible*. Node ini memiliki 54 input/output pin digital, 12 pin input analog, dan 84MHz clock. Arduino due menggunakan micro USB untuk pemrograman.

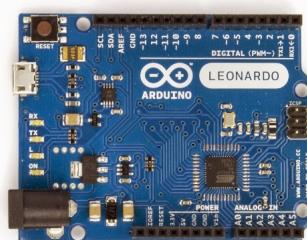
3. Arduino Mega



Gambar 2.17: Arduino Mega

Arduino Mega merupakan pengembangan dari arduino uno. Keduanya menggunakan USB type B untuk pemrogramannya. Namun, arduino mega menggunakan chip yang lebih tinggi yaitu ATMEGA2560. Penggunaan chip ATMEGA2560, memungkinkan penyimpanan ukuran program yang lebih besar. Arduino mega juga memiliki ukuran papan yang lebih besar dibandingkan papan lainnya. Hal ini dikarenakan jumlah pin input/output dan pin input analognya yang lebih banyak dibandingkan arduino uno.

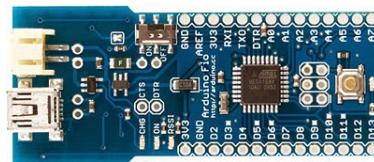
4. Arduino Leonardo



Gambar 2.18: Arduino Leonardo

Arduino Leonardo memiliki kemiripan yang hampir identik dengan arduino uno, jumlah pin input/output dan pin input analognya-pun sama dengan arduino uno. Hanya saja, Micro USB digunakan untuk pemrograman pada arduino leonardo, berbeda dengan arduino uno yang menggunakan USB type B untuk basis pemrogramannya.

5. Arduino Fio



Gambar 2.19: Arduino Fio

Arduinio Fio menggunakan socket berbeda dengan jenis arduino lainnya. Socket yang digunakan pada arduino fio adalah XBee. Socket XBee memungkinkan arduino fio untuk digunakan

dalam proyek yang mengharuskan penggunaan nirkabel. Dengan menggunakan arduino fio, pengguna dapat mengirimkan data secara *wireless*.

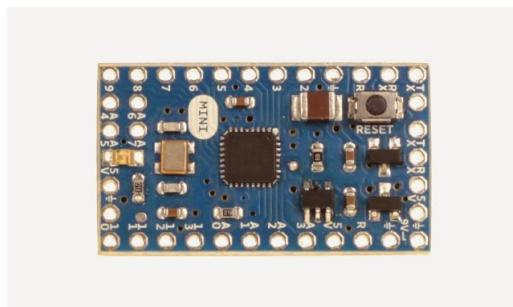
6. Arduino Nano



Gambar 2.20: Arduino Nano

Arduino nano adalah mikrokontroler dengan papan berukuran kecil. Chip yang digunakan pada arduino nano adalah ATmega328P. Untuk pemrograman, arduino nano dilengkapi dengan FTDI dengan Micro USB. Jumlah pin input analog yang dimiliki android nano lebih banyak dibandingkan dengan android uno (8 Pin). Android nano memiliki dua versi yang berbeda pada chip yang digunakannya, ATMEGA168 atau ATMEGA328.

7. Arduino Mini



Gambar 2.21: Arduino Mini

Ukuran papan arduino mini yang kecil membuatnya sedikit lebih rumit untuk dihubungkan, dibandingkan dengan papan arduino lainnya. Namun, ukurannya yang kecil memungkinkan arduino mini untuk digunakan pada tempat sempit yang memerlukan node sensor berukuran kecil.

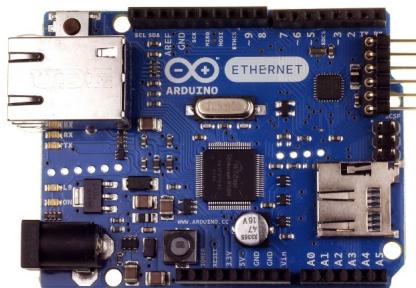
8. Arduino Micro



Gambar 2.22: Arduino Micro

Arduino micro merupakan perangkat keras kecil seperti arduino nano dan arduino mini. Namun arduino micro memiliki kelebihan dibanding perangkat arduino berukuran kecil lainnya. Mudahnya dalam mengintegrasikan arduino micro dengan benda sehari-hari menjadikannya lebih interaktif dibandingkan jenis arduino lainnya.

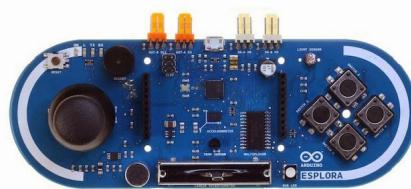
9. Arduino Ethernet



Gambar 2.23: Arduino Ethernet

Seperti namanya arduino ethernet telah dilengkapi dengan fasilitas ethernet. Dengan fitur tersebut, arduino ethernet dapat melakukan komunikasi melalui jaringan LAN pada komputer. Arduino ethernet juga dilengkapi dengan fitur *reset* otomatis, yang memungkinkan pengguna untuk mengunggah data tanpa harus menekan tombol reset.

10. Arduino Esplora



Gambar 2.24: Arduino Esplora

Seperti arduino leonardo, arduino esplora menggunakan chip ATMEGA32U4. Namun, arduino esplora menyediakan sejumlah sensor yang telah terintegrasi dan siap untuk digunakan. Arduino esplora juga telah dilengkapi dengan *joystick*, dan tombol interaksi yang terhubung dengan papan.

2.4.2 Jenis-jenis sensor *sensing* arduino

Pada setiap node yang disebar di area persawahan yang akan dilakukan pengukuran, terpasang sensor yang dapat melakukan *sensing* terhadap lingkungan. Untuk pengukuran yang berkaitan dengan kondisi tanah sawah, sensor *sensing* yang digunakan harus dapat mengukur variabel yang mempengaruhi kondisi tanah sawah. Sensor yang dapat mengukur faktor-faktor yang mempengaruhi kondisi tanah sawah antara lain, sensor pengukur kadar keasaman tanah (pH), sensor pengukur kelembaban tanah, sensor pengukur suhu temperatur tanah, sensor pengukur suhu dan kelembaban udara, dan *Wireless Data Transceiver*.

1. Sensor pengukur kadar keasaman tanah (pH)

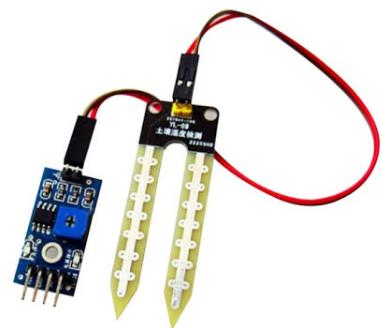
Rangkaian ini terdiri dari komponen sensor keasaman yang digunakan untuk melakukan pengukuran pH yang berada di tanah. Pengukuran dilakukan untuk mengetahui kandungan yang berada dalam tanah seperti Nitrogen (N), Potassium/kalium (K), dan Pospor (P) yang sangat berpengaruh terhadap tanaman padi untuk tumbuh dan menghasilkan kualitas yang baik. Contoh sensor pengukur kadar keasaman tanah ini dapat dilihat pada Gambar 2.25



Gambar 2.25: Sensor pengukur kadar keasaman tanah (pH)

2. Sensor pengukur tingkat kelembaban tanah

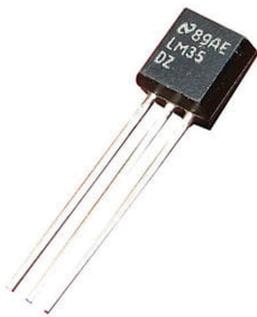
Untuk mengetahui kadar air yang berada di dalam tanah maka diperlukan sensor kelembapan yang berfungsi untuk mendapatkan tingkat kelembapan tanah berdasarkan arus listrik yang dihantarkan. Semakin banyak kadar air dalam tanah mengindikasikan semakin lembab tanah tersebut, dan semakin tinggi tingkat kelembaban tanah maka arus listrik yang dihantarkan akan semakin tinggi (resistansi tinggi). Jika tanah memiliki kadar air yang rendah (tanah kering) maka arus listrik yang dihantarkan akan rendah (lebih banyak perlawanan). Contoh sensor pengukur tingkat kelembaban ini dapat dilihat pada Gambar 2.26



Gambar 2.26: Sensor pengukur tingkat kelembaban tanah

3. Sensor pengukur suhu temperatur tanah

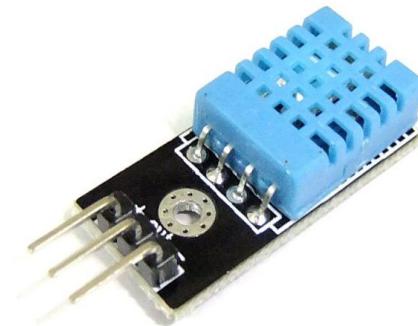
Sensor suhu diperlukan untuk memantau temperatur kondisi tanah sawah yang diteliti untuk mengetahui suhu telah mencapai temperatur yang tepat. Contoh sensor pengukur suhu temperatur tanah dapat dilihat pada Gambar 2.27



Gambar 2.27: Sensor pengukur suhu temperatur tanah

4. Sensor pengukur suhu dan kelembaban udara

Sensor suhu dan kelembaban udara diperlukan untuk memantau suhu dan kelembaban suatu daerah, dan mengetahui suhu udara dan tingkat kelembaban udara daerah yang diteliti telah mencapai temperature yang tepat. Contoh sensor pengukur suhu dan kelembaban udara dapat dilihat pada Gambar 2.28



Gambar 2.28: Sensor pengukur suhu kelembaban udara

5. WDT (*Wireless Data Transceiver*)

WDT atau *Wireless Data Transceiver* digunakan untuk mengirimkan dan menerima data, tanpa memerlukan kabel sebagai perantara pengiriman. WDT juga berfungsi sebagai alat komunikasi antar node sensor. Contoh Wireless Data Transceiver dapat dilihat pada Gambar 2.29



Gambar 2.29: *Wireless Data Transceiver*

2.4.3 Pemrograman Arduino

Arduino menggunakan bahasa pemrograman mandiri (bahasa pemrograman arduino) yang digunakan pada AVR. Bahasa pemrograman arduino, memiliki banyak kemiripan dengan bahasa pemrograman C. Kemiripan kedua bahasa ini dapat terlihat dari fungsi, syntax, variabel, operator matematika, operator pembanding, dan struktur pengaturan yang tersedia. Bahasa pemrograman Arduino juga sering dianggap sebagai bahasa *processing*⁹.

- Fungsi

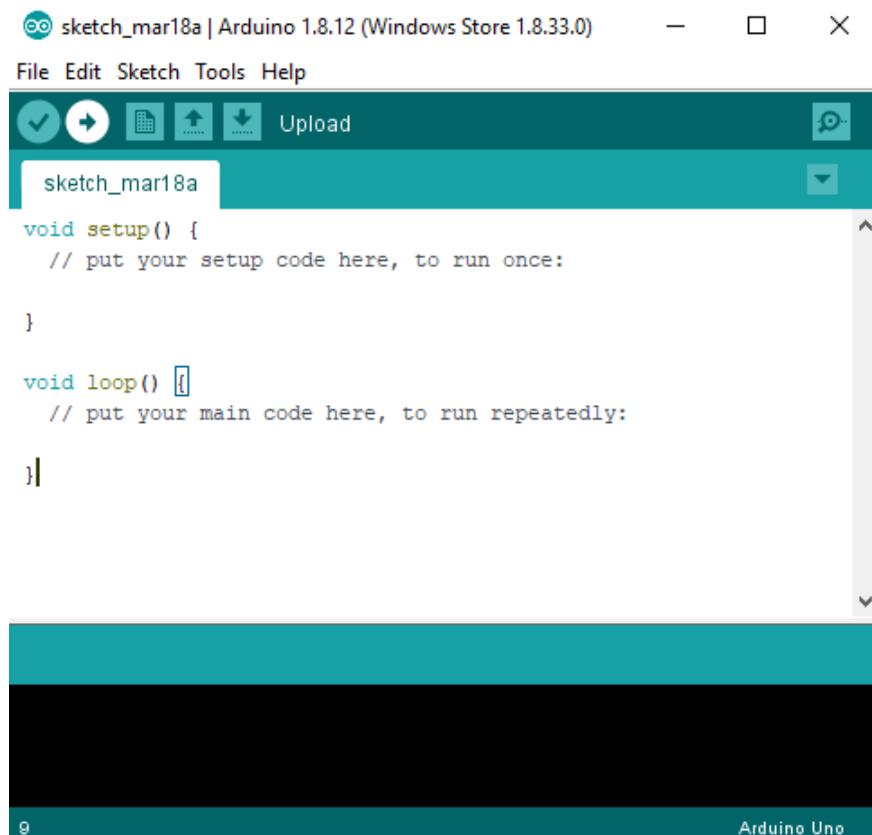
Program yang terdapat di arduino juga biasa disebut dengan *sketch*. Terdapat dua buah fungsi yang harus ada pada setiap *sketch*, yaitu fungsi setup dan fungsi loop.

- Void setup

Void setup merupakan fungsi pertama yang akan dieksekusi dalam pemrograman arduino sebelum fungsi atau kode lainnya. Fungsi void setup memiliki *task* untuk menentukan kegunaan dari sebuah pin. Selain itu fungsi ini juga dapat digunakan untuk memulai komunikasi antara arduino dengan komputer.

- Void loop

Semua kode yang berada di dalam fungsi ini akan dibaca setelah fungsi void setup dieksekusi, dan akan terus menerus dijalankan sampai daya yang terhubung pada perangkat dilepaskan. Fungsi ini berisikan kode-kode perintah untuk pin *input* dan *output* pada arduino.



```
sketch_mar18a | Arduino 1.8.12 (Windows Store 1.8.33.0)
File Edit Sketch Tools Help
Upload
sketch_mar18a
void setup() {
  // put your setup code here, to run once:
}

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

Gambar 2.30: Fungsi setup dan loop pada Arduino

⁹<https://kelasrobot.com/belajar-pemrograman-dasar-arduino/>

- Struktur pengaturan

Didalam struktur pengaturan bahasa pemrograman arduino, terdapat struktur-struktur yang umum terdapat pada bahasa pemrograman lainnya. Beberapa contoh struktur pengaturan di dalam bahasa pemrograman arduino adalah *loop* menggunakan *for* untuk melakukan pengulangan dan penggunaan *if-else* untuk pengolahan kode program dengan kondisi bercabang (*branching*).

- Digital

Kode digital digunakan untuk pemrograman yang menggunakan pin digital pada arduino. Kode program yang berkaitan dengan nilai digital adalah pinMode yang digunakan untuk melakukan pengaturan input-output mode pin, digitalRead yang digunakan untuk membaca nilai sensor yang terdapat pada pin, dan digitalWrite untuk melakukan pengaturan pada pin output.

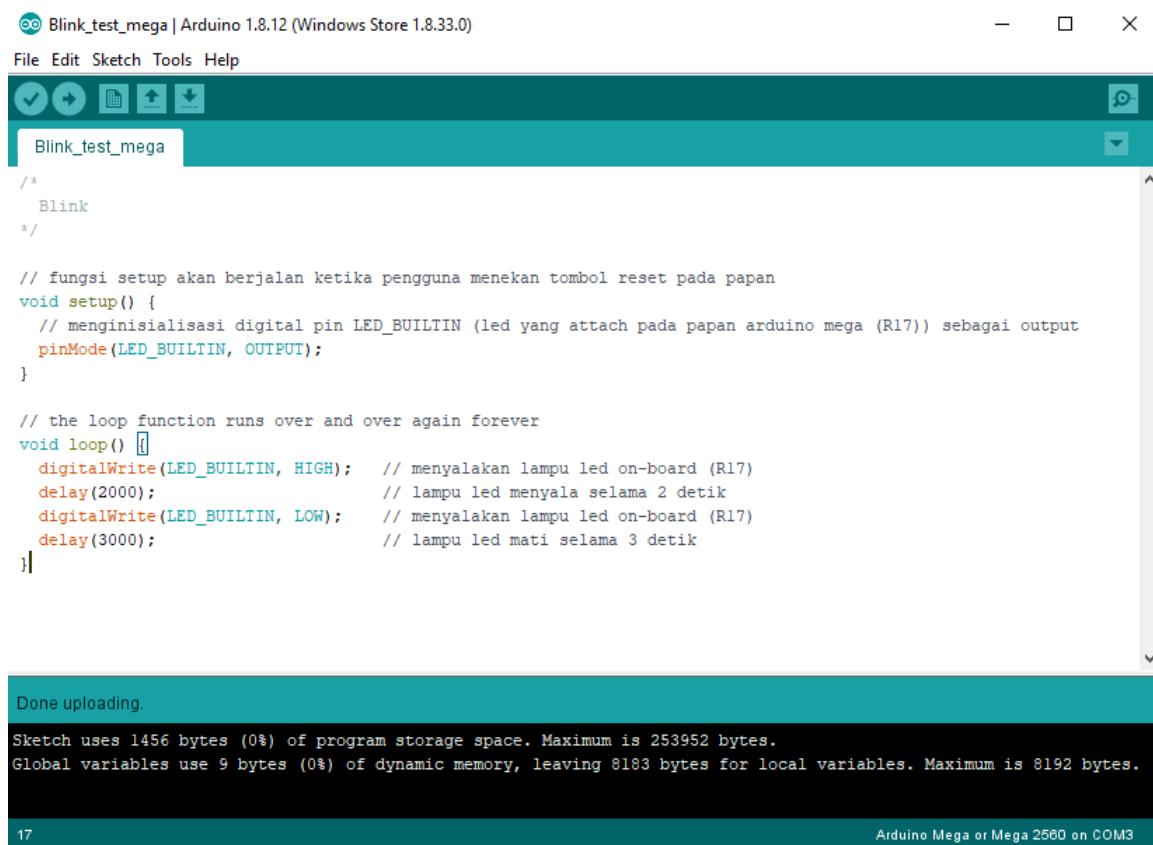
- Analog

Tidak hanya mengelola hasil dalam format digital, arduino juga dapat menggunakan fungsi analognya pada pin digital arduino. Sama seperti digital fungsi *read* dan *write* juga dapat diolah dalam format analog.

- *Blink Test*

Salah satu contoh pemrograman pada Arduino yang cukup sederhana adalah mengedipkan LED *on-board* pada papan Arduino, atau biasa disebut dengan *blink test*. Untuk melakukan *blink test*, pengguna harus memastikan jenis board yang dipilih pada IDE Arduino telah sesuai dengan perangkat keras yang digunakan (yang terhubung dengan komputer). Setelah IDE Arduino dan perangkat keras telah sesuai dan terhubung, maka pemrograman dapat dilakukan seperti Gambar 2.31. Setelah kode program selesai ditulis, pengguna dapat menverifikasi program dengan cara menekan tombol dengan ikon check diatas kiri IDE Arduino. Setelah kode program terverifikasi, pengguna dapat mengupload kode program tersebut ke papan arduino, dengan cara menekan tombol upload yang berada disamping tombol verifikasi. Jika upload sukses dilakukan maka IDE akan memberikan informasi "Done uploading", dan papan arduino yang terhubung akan melakukan kode perintah yang diupload. Untuk kode program pada Gambar 2.31, papan akan mengedipkan led on-board setiap 3 detik dengan lama lampu menyala selama 2 detik.

Blink test juga dapat digunakan untuk menguji pin yang terdapat pada papan, berfungsi dengan baik atau tidak. Blink test untuk menguji fungsional pin dapat dilakukan dengan cara memasang led eksternal pada pin yang tersedia pada papan arduino, dan menjalankan program seperti pada Gambar 2.32.



The screenshot shows the Arduino IDE interface. The title bar reads "Blink_test_mega | Arduino 1.8.12 (Windows Store 1.8.33.0)". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for file operations like Open, Save, and Print. The sketch window contains the following code:

```
/*
Blink

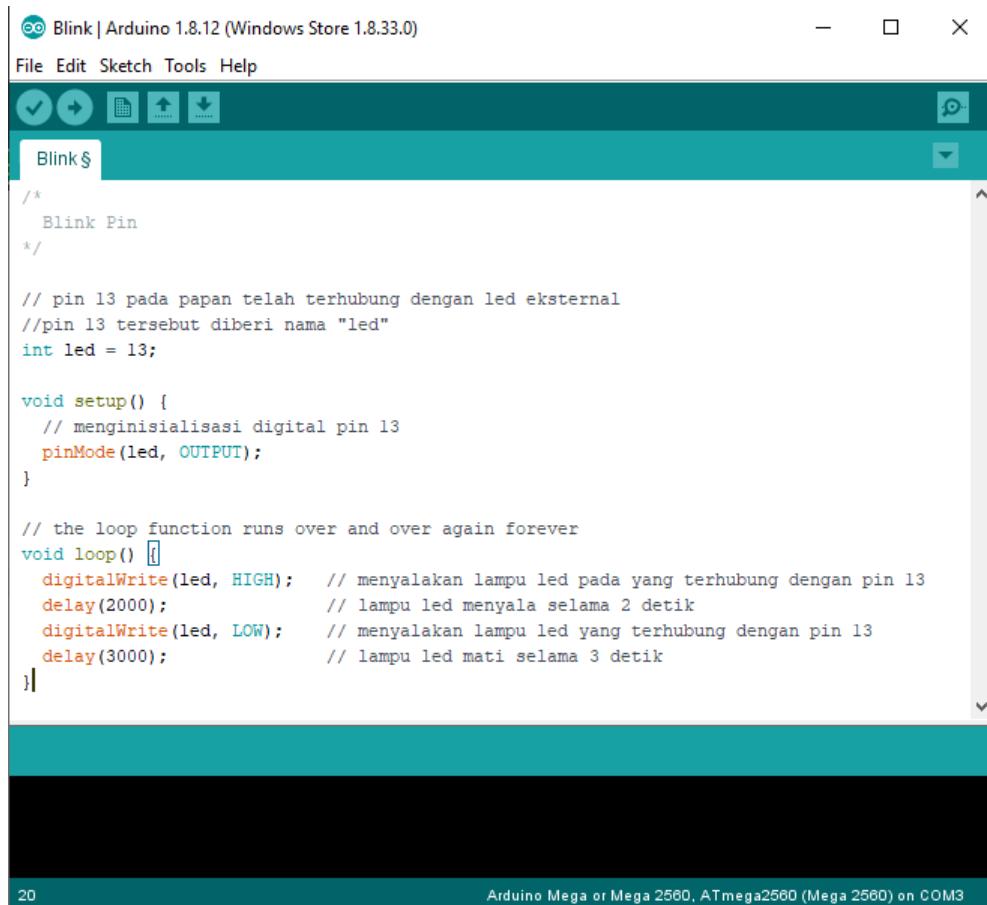
*/
// fungsi setup akan berjalan ketika pengguna menekan tombol reset pada papan
void setup() {
    // menginisialisasi digital pin LED_BUILTIN (led yang attach pada papan arduino mega (R17)) sebagai output
    pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);      // menyalaikan lampu led on-board (R17)
    delay(2000);                      // lampu led menyala selama 2 detik
    digitalWrite(LED_BUILTIN, LOW);     // menyalaikan lampu led on-board (R17)
    delay(3000);                      // lampu led mati selama 3 detik
}


```

The status bar at the bottom shows "Done uploading." and "Sketch uses 1456 bytes (0%) of program storage space. Maximum is 253952 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 8183 bytes for local variables. Maximum is 8192 bytes." The bottom right corner indicates "Arduino Mega or Mega 2560 on COM3".

Gambar 2.31: *Blink Test On-Board*



Gambar 2.32: *Blink Test Pin*

2.5 Raspberry

Raspberry merupakan komputer berukuran kecil yang dapat terhubung dengan desktop ataupun *notebook*. Penggunaan sistem operasi linux oleh Raspberry, memungkinkan pengguna untuk meng-
erjakan proyek dan bereksperimen dengan sistem operasi yang bersifat *open-source*. Raspberry
juga dapat bertindak sebagai SINK (*base station*), yang bertujuan untuk meneruskan data yang
dikirimkan oleh node sensor ke server[6].

2.5.1 Jenis-jenis Raspberry

Sama dengan perangkat keras arduino, Raspberry memiliki beberapa model dengan spesifikasi yang berbeda-beda. Pada umumnya perangkat keras ini menggunakan SoC (System on Chip) ARM 64-bit dan dilengkapi dengan RAM sebesar 512MB. Beberapa model Raspberry antara lain:

- Raspberry pi A

Raspberry pi A, merupakan perangkat keras generasi pertama yang hanya memiliki sebuah USB port dan SDRAM sebesar 256MB. Perangkat keras ini juga dilengkapi dengan processor ARM 11. Jumlah USB Port yang sedikit memungkinkan penggunaan konsumsi daya yang lebih kecil dibanding model Raspberry lainnya.



Gambar 2.33: Raspberry pi A

- Raspberry pi A+

Pengembangan yang dilakukan pada Raspberry pi A, memunculkan perangkat keras model baru dengan nama Raspberry pi A+. Raspberry pi A+ memiliki spesifikasi yang mirip dengan Raspberry pi A. Namun, Raspberry A+ memiliki ukuran yang lebih kecil dibandingkan model lainnya, yaitu hanya 65mm saja.



Gambar 2.34: Raspberry pi A+

- Raspberry pi B

Pada Raspberry pi B, terjadi penambahan di berbagai spesifikasi. Kapasitas RAM pada perangkat keras ini lebih besar dibanding generasi sebelumnya yang hanya 256MB, begitupula dengan jumlah port yang dimilikinya. Kapasitas RAM yang dimiliki Raspberry pi B adalah sebesar 512MB dan memiliki dua buah port USB. Raspberry pi B juga telah dilengkapi dengan sebuah ethernet, yang merupakan salah satu dari dua port USB yang disebutkan.



Gambar 2.35: Raspberry pi B

- Raspberry pi B+

Raspberry pi B+ merupakan pengembangan dari model sebelumnya, Raspberry pi B. Namun, konsumsi daya yang digunakan lebih kecil dibanding model sebelumnya.



Gambar 2.36: Raspberry pi B+

- Raspberry pi 2 B+

Model Raspberry pi 2 B+ memungkinkan pengguna untuk menjalankan banyak distribusi sistem operasi. Hal ini dikarenakan model ini telah menggunakan processor dengan jenis ARMv7.



Gambar 2.37: Raspberry pi 2 B+

- Raspberry pi 3 B+

Model Raspberry pi 3 B+, telah terintegrasi dengan Bluetooth 4.1 dan Bluetooth Low Energy(BLE). Dengan memanfaatkan teknologi bluetooth, pengguna tidak lagi memerlukan modul USB wireless LAN tambahan, dalam perangkat keras Raspberry pi 3 B+.



Gambar 2.38: Raspberry pi 3 B+

2.5.2 Pemrograman Raspberry

Saat ini sudah banyak bahasa pemrograman yang telah disesuaikan untuk Raspberry pi, seperti bahasa pemrograman Python, C, C ++, Java, Scratch, dan Ruby. Semua bahasa pemrograman tersebut telah terinstall secara default pada Raspberry pi, dan siap digunakan oleh pengguna sesuai dengan bahasa pilihannya masing-masing.

2.5.3 Pemrograman Python untuk Raspberry

Salah satu contoh pemrograman di Raspberry menggunakan bahasa pemrograman python adalah mengubah nama *host* menjadi *ip address*. Pada IDE / Python Shell masukan kode seperti yang terdapat pada kode program 2.1. Program akan mencari alamat ip dari nama website yang terdapat dalam variabel website. Program dapat dijalankan dengan menekan tombol 'Run Module' pada menu 'Run' atau menekan key 'F5' pada keyboard. Ketika module dijalankan maka akan tampil Python Shell yang merupakan hasil pengolahan kode program yang dieksekusi, seperti yang ditunjukkan pada Gambar 2.39.

Kode 2.1: rasp_test.py

```

1 #Mengubah nama Host menjadi alamat IP
2
3 import socket
4
5 website = 'www.google.com'
6 ip = socket.gethostbyname (website)
7 print('*****')
8 print('nama_website:', website)
9 print('alamat_ip:', ip)
10 print('*****')

```

```

Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/USER/AppData/Local/Programs/Python/Python38-32/rasp_test.py
*****
nama website : www.google.com
alamat ip : 216.239.38.120
*****
>>> |

```

Gambar 2.39: Shell Python setelah eksekusi *module*

2.6 Basis Data

Basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematis sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Pada tahun 1960 Charles Bachman merancang *Database Management System* (DBMS) yang digunakan untuk menyimpan data. Kemudian pada tahun yang sama, *International Business Machines Corporation* (IBM) mengembangkan sistem manajemen informasi DBMS. Terdapat beberapa jenis bahasa komputer yang digunakan saat membangun dan memanipulasi sebuah basis data, yaitu *Data Definition Language* (DDL), *Data Manipulation Language* (DML), dan *Data Control Language* (DCL). *Entity Relation Diagram* atau ERD adalah suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. ERD juga digunakan untuk memodelkan struktur data dan hubungan antardata, dan menggambarkannya digunakan beberapa notasi dan simbol.

BAB 3

ANALISIS

Bab ini akan menjelaskan mengenai analisis sistem merujuk pada hasil studi yang telah dibahas pada bab sebelumnya. Pada bab ini juga akan dibahas perancangan sistem secara umum yang ditunjukkan dalam bentuk *flowchart* dan *usecase diagram*

3.1 Deskripsi Sistem

Perangkat lunak yang dibangun merupakan aplikasi yang berfungsi untuk memantau kondisi tanah sawah. Pemantauan dilakukan dengan menggunakan sensor-sensor yang terhubung dalam jaringan yang berfungsi untuk melakukan pengambilan variabel-variabel yang mempengaruhi kualitas tanah sawah. Variabel tanah yang mempengaruhi kualitas tanah sawah antara lain, kadar keasaman tanah (pH tanah), tingkat kelembaban tanah, suhu tanah, suhu dan kelembaban udara persawahan. Pengambilan data dari variabel-variabel tersebut akan dilakukan oleh sensor secara *real-time*, dan ditampilkan ke komputer pengguna. Pengguna dari perangkat lunak yang dibangun dapat mengetahui kualitas tanah sawah yang dikelolanya, berdasarkan data yang dihasilkan dari kegiatan pemantauan. Dari data yang dihasilkan oleh perangkat lunak, pengguna diharapkan dapat terbantu dalam menentukan jenis perawatan terhadap tanah sawah yang dikelola.

Secara sederhana proses yang dilakukan dalam pembangunan aplikasi ini adalah sebagai berikut. Sensor-sensor node disebar di area persawahan dan melakukan *sensing* terhadap tanah yang diteliti. Sensor-sensor tersebut akan saling berkomunikasi untuk mengirimkan hasil *sensing* yang didapatkannya ke *base station*. *Base station* akan menerima semua hasil *sensing* sensor yang disebar untuk diteruskan dan ditampilkan ke layar komputer.

3.2 Analisis Arsitektur dan Topologi WSN

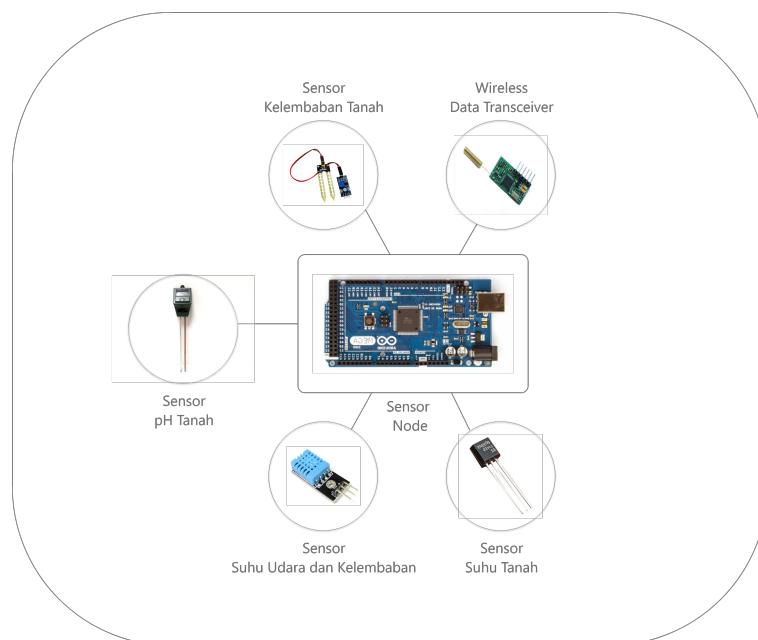
Sensor node yang digunakan pada penelitian ini adalah sensor Arduino Mega 2560. Seperti yang dijelaskan pada subbab 2.4.3, Arduino Mega merupakan pengembangan dari Arduino Uno yang merupakan jenis mikrokontroler paling populer. Jumlah pin yang lebih banyak pada Arduino Mega, memungkinkan lebih banyaknya node *sensing* yang dapat *diattach* pada papan untuk mengambil data tanah persawahan. Selain itu Arduino Mega juga dapat menyimpan data (bersifat sementara) hasil *sensing* lebih banyak dibandingkan Arduino Uno.

Sensor *sensing* yang akan digunakan dalam penelitian ini antara lain, sensor pengukur pH tanah (ETP306), sensor kelembaban (FC28), sensor suhu tanah (DS18B20), sensor suhu udara(DHT11), dan *Wireless Data Transceiver*(NRF24L01). Sensor ETP306 digunakan untuk mengambil data kadar keasaman dari tanah sawah yang diteliti. Sensor FC28 *attach* dengan seri sensor MH untuk dapat terhubung dengan node sensor. Sensor FC28 digunakan untuk melakukan *sensing* kadar kelembaban tanah. Untuk mengetahui suhu tanah, sensor DS18B20 akan dihubungkan dengan node sensor. Sensor FC28 dan DS18B20 harus secara langsung ditancapkan kedalam tanah sawah yang diteliti, untuk mengambil data dan melakukan *sensing*. Sensor DHT11 dapat langsung *attach* pada pin yang berada pada node sensor dan memberikan informasi suhu dan kelembaban udara sekitar persawahan. *Wireless Data Transceiver* (NRF24L01) merupakan modul komunikasi *wireless*

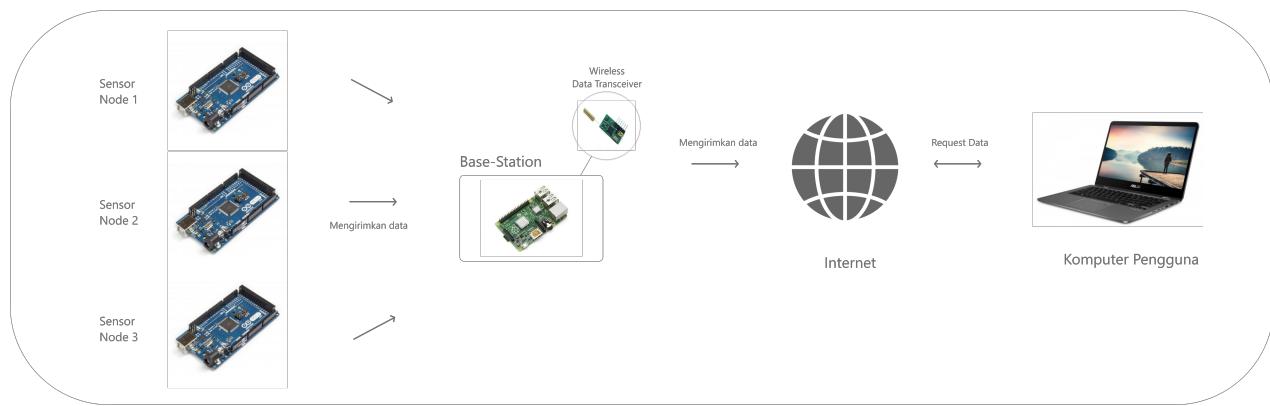
yang dapat digunakan pada perangkat keras Arduino ataupun Raspberry. Seluruh sensor sensing tersebut akan terhubung dalam satu buah node sensor. Untuk *base station* jenis perangkat keras yang akan digunakan adalah Raspberry pi 2 B+ yang memungkinkan pemrograman dilakukan pada sistem operasi Windows, seperti yang dibahas pada subbab 2.5.1.

Perangkat lunak yang dibangun memfokuskan dalam melakukan *sensing* terhadap tanah sawah yang diteliti dan melakukan komunikasi, agar data yang dihasilkan dapat dikirimkan sampai di *base station*, untuk diteruskan ke komputer atau pengguna. Berdasarkan pembahasan pada subbab 2.2.4 mengenai arsitektur WSN, pemilihan jenis arsitektur WSN *flat* akan lebih baik untuk diimplementasikan. Arsitektur WSN flat tersebut dipilih berdasarkan luasnya wilayah penelitian yang tidak terlalu besar, dan jenis *task* dari setiap node yang disebar yang memiliki tugas yang sama (melakukan *sensing*).

Topologi yang akan diuji pada perangkat lunak yang dibangun adalah topologi *star*, topologi *linear*, dan topologi *partially connected mesh*, seperti yang telah dibahas pada subbab 2.2.5. Topologi *linear* dipilih sebagai dasar dari topologi lainnya dan menjadi basis komunikasi antar node. Pada topologi *star*, *central node* akan berperan sebagai *base station* (bersifat aktif) yang merupakan Raspberry pada perangkat lunak yang dibangun. Sedangkan node-node lainnya akan bertugas untuk melakukan *sensing*. Topologi *partially connected mesh* dibangun sebagai penguji diantara jenis topologi jaringan lainnya dan memungkinkan minimalnya gangguan yang terjadi pada topologi lainnya, seperti yang dijelaskan pada subbab 2.2.5.



Gambar 3.1: Node Sensor



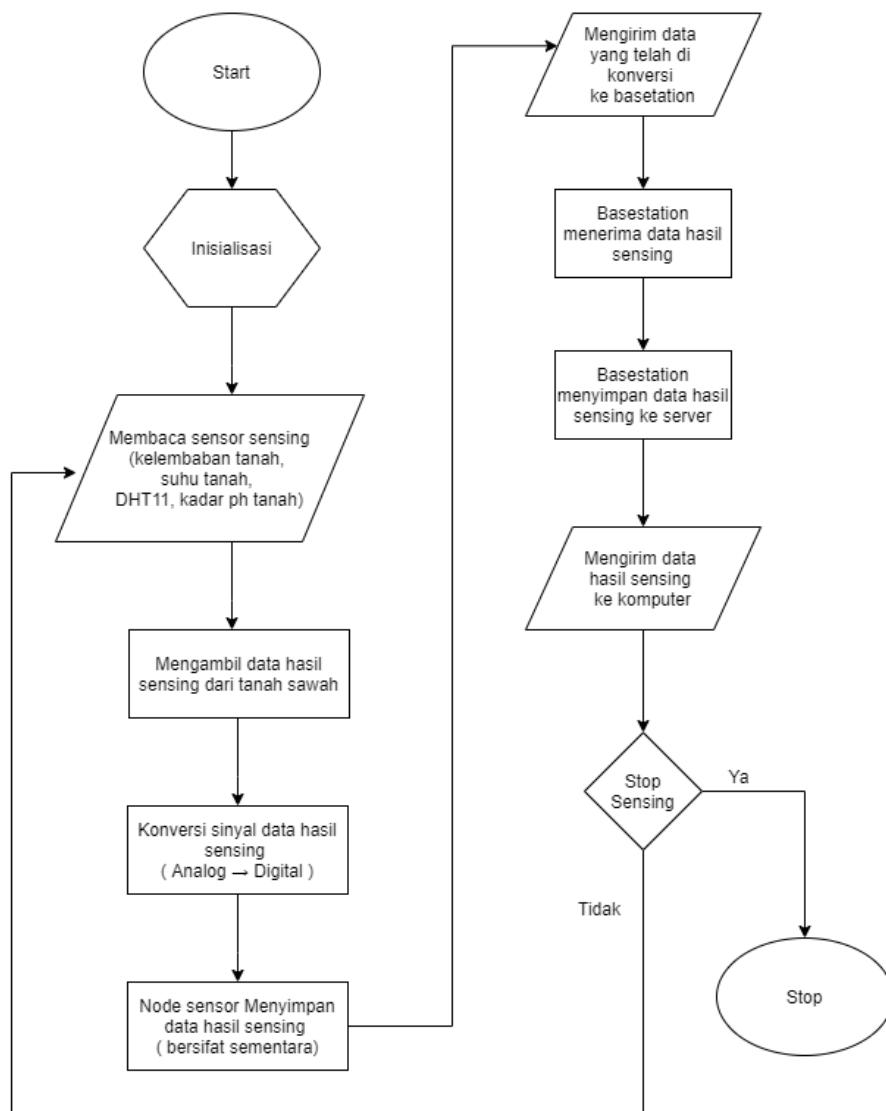
Gambar 3.2: Arsitektur dan Topologi WSN

3.3 Analisis Perangkat lunak

Perangkat keras yang dibutuhkan dalam melakukan pemantauan kondisi tanah padi terdiri dari node sensor dan sensor *sensing*. Kedua perangkat tersebut akan dirakit sebelum disebar pada area yang diteliti. Masing-masing node sensor memiliki sejumlah sensor *sensing* untuk melakukan pengambilan informasi tanah sawah, seperti kadar keasaman tanah (pH), kelembaban tanah, dan suhu tanah. Selain sensor *sensing* untuk tanah, node sensor yang disebar juga memiliki sensor *sensing* udara yang dapat mendeteksi suhu dan kelembaban udara persawahan. Modul terakhir yang terpasang pada node sensor adalah *Wireless Data Transceiver* yang berfungsi untuk mengirimkan dan menerima data antar node sensor maupun dengan *base station*.

Setelah perangkat keras telah siap untuk digunakan, proses yang akan dilakukan adalah menyebarluaskan beberapa node sensor di area persawahan yang diteliti. Letak ataupun bentuk penyebarluasan node sensor yang dilakukan bergantung pada jenis topologi yang akan diuji. Sensor sensing pada setiap node akan mengambil data tanah sawah yang diteliti dan akan diterima oleh node sensor.

Node sensor akan menyimpan informasi hasil *sensing* yang diterima untuk diteruskan ke *base station*. Jika jarak suatu node sensor terlalu jauh dengan *base station*, maka data hasil *sensing* akan dikirimkan ke node sensor yang berdekatan dengannya (dan lebih dekat dengan *base station*) untuk meneruskan data hasil *sensing*nya ke *base station*. Node sensor akan saling berkomunikasi dan saling bertukar data sampai data tersebut sampai ke *base station*. Setelah data sampai di *base station*, maka data tersebut akan disimpan dalam database server. Data-data yang tersimpan akan diteruskan oleh *base station* ke komputer untuk ditampilkan pada pengguna dan/atau admin.



Gambar 3.3: Flowchart keseluruhan sistem

3.3.1 Analisis Fungsi Aplikasi

Aplikasi yang dibangun memiliki fungsi-fungsi sebagai berikut :

1. Memeriksa status pada setiap node sensor
2. Menyamakan waktu pada setiap node sensor
3. Mengirimkan perintah untuk melakukan *sensing* pada seluruh node
4. Mengirimkan perintah untuk menghentikan aktifitas *sensing* pada seluruh node
5. Mengubah data hasil *sensing* dari sinyal analog menjadi sinyal digital
6. Menyimpan hasil konversi hasil *sensing* di node sensor
7. Mengirimkan data hasil *sensing* ke *base station*
8. Menyimpan data hasil *sensing* yang diterima *base station*
9. Menampilkan data hasil *sensing* yang disimpan oleh *base station*

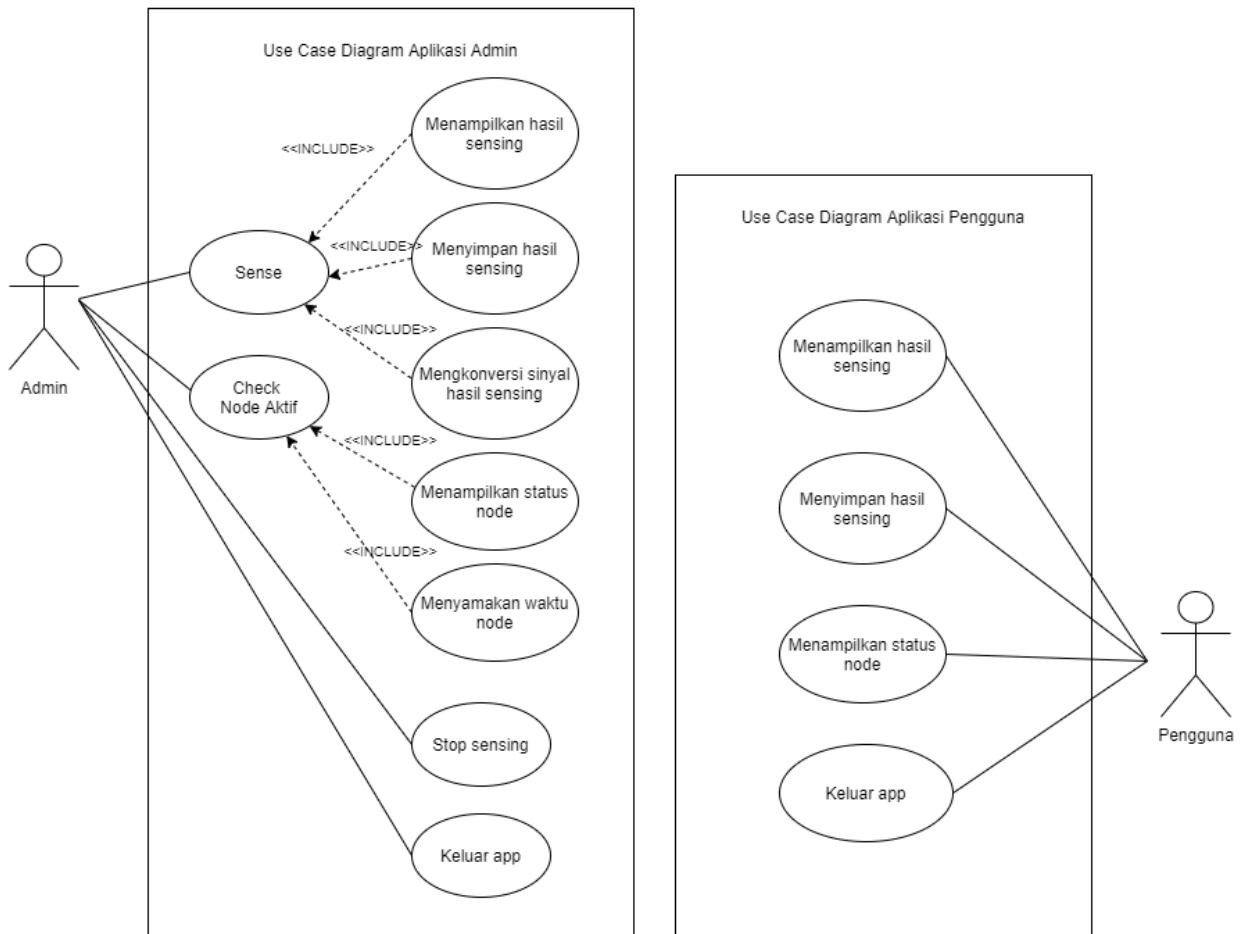
Aplikasi memiliki fungsi untuk memeriksa status untuk setiap node sensor yang disebar. Fungsi ini memastikan sensor dalam jaringan yang ada akan melakukan *sensing* berstatus aktif, sebelum melakukan pengiriman data ke *base station*. Jika terdapat sensor yang tidak dalam status aktif, maka aplikasi akan memberi informasi node yang tidak aktif tersebut dan pengguna dapat melakukan perbaikan terlebih dahulu.

Fungsi menyamakan waktu pada setiap node diperlukan agar setiap node mengirimkan data hasil *sensing* secara bersamaan atau pada waktu yang sama. Selain itu *memory* di setiap node sensor yang bersifat sementara (*volatile*), juga menjadi alasan perlunya penyamaan waktu. Karena aplikasi perlu mencatat waktu terakhir setiap node sensor tersebut mati, agar dapat kembali melakukan konfigurasi waktu, dan node dapat kembali melakukan *sensing* secara bersamaan.

Aplikasi yang dibangun juga memiliki fungsi untuk memberikan perintah kepada node sensor yang disebar untuk melakukan *sensing*. Ketika proses pemantauan selesai maka aplikasi juga dapat memberikan perintah untuk menghentikan kegiatan *sensing* pada setiap node sensor.

Fungsi konversi data hasil *sensing* digunakan untuk mengubah sinyal hasil *sensing* yang berupa sinyal analog menjadi sinyal digital. Hasil konversi ini akan langsung dikirimkan ke *base station* dan disimpan di server atau *localhost*. *Base station* akan meneruskan data hasil *sensing* yang diterimanya ke komputer.

Fungsi yang terakhir merupakan fungsi antarmuka. Antarmuka pada aplikasi yang dibangun dapat menampilkan data hasil *sensing* di layar komputer pengguna. Selain itu, antarmuka juga menjadi perantara untuk pengguna, untuk berinteraksi dengan *base station* dan node sensor.



Gambar 3.4: Use Case Diagram Aplikasi

Tabel 3.1: Tabel Skenario Memeriksa Status Node Sensor Aplikasi Admin

Nama	<i>Check Status Node</i>
Deskripsi	Memeriksa status node sensor agar siap untuk melakukan <i>sensing</i> . Base-station akan mengirim sinyal kepada seluruh node yang terhubung dalam jaringan. Jika node aktif, maka node akan mengirim data status node untuk ditampilkan pada aplikasi Admin. Jika sensor tidak aktif maka tidak ada sinyal atau data yang dapat dikirimkan. Aplikasi juga menyamakan waktu pada setiap node sensor dengan komputer pengguna
Aktor	Admin
Pre-kondisi	Aplikasi Admin sudah dibuka
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Sistem menampilkan opsi fitur (menu utama) aplikasi admin yang dapat dipilih oleh admin. 2. Admin memilih opsi "<i>Check Status Node</i>" 3. Admin menginput opsi untuk mengeksekusi "Check Status Node" 4. Sistem mengirim sinyal ke seluruh Node yang terhubung dalam jaringan untuk mengirim status node 5. Sistem menampilkan dialog "Mengirim perintah check status" dan "Silakan tunggu respon" 6. Sistem menunggu waktu <i>time-out</i> dan menerima status node yang dikirimkan oleh node sensor yang aktif dan terhubung dalam jaringan 7. Sistem menampilkan node yang berstatus aktif 8. Sistem menampilkan dialog "Check Node Selesai" bila terdapat node yang merespon perintah 'check status node' 9. Sistem menampilkan dialog "Tidak ada respon" dan "Silakan Check Perangkat", jika tidak ada node yang merespon perintah 'check status node' 10. Sistem kembali menampilkan menu utama aplikasi admin

Tabel 3.2: Tabel Skenario Perintah *Sensing* Aplikasi Admin

Nama	<i>Mulai Sensing</i>
Deskripsi	Memberikan perintah untuk mengeksekusi proses <i>sensing</i> pada seluruh node sensor, mengubah sinyal hasil <i>sensing</i> , dan melakukan pengiriman data oleh <i>base station</i> , serta menampilkannya pada komputer
Aktor	Admin
Pre-kondisi	Aplikasi sudah aktif dan seluruh status node telah " <i>online</i> "
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Sistem menampilkan opsi fitur (menu utama) aplikasi yang dapat dipilih oleh admin. 2. Admin memilih opsi "<i>Mulai Sensing</i>" 3. Admin menginput opsi untuk mengeksekusi "Mulai Sensing" 4. Sistem menampilkan dialog "Sensing dimulai.." 5. <i>Base station</i> mengirimkan perintah sensing pada setiap node yang terhubung dalam jaringan 6. Node yang terhubung dalam jaringan menerima perintah <i>sensing</i> dan melakukan proses <i>sensing</i> dan pemantauan 7. Sistem pada node mengubah sinyal hasil <i>sensing</i> dari analog ke digital 8. Sistem pada node mengirim hasil sensing ke base-station 9. Aplikasi admin menerima hasil sensing yang dikirimkan oleh node sensor 10. Aplikasi admin menampilkan hasil sensing yang diterima dan menyimpan hasil <i>sensing</i> tersebut

Tabel 3.3: Tabel Skenario Memeriksa Status Node Sensor Aplikasi Pengguna

Nama	<i>Check Status</i>
Deskripsi	Memeriksa status node sensor agar siap untuk melakukan <i>sensing</i> , jika sensor aktif maka akan menampilkan status " <i>online</i> ", sebaliknya jika sensor tidak aktif maka akan ditampilkan status " <i>offline</i> ". Aplikasi juga menyamakan waktu pada setiap node sensor dengan komputer pengguna
Aktor	Pengguna dan Admin
Pre-kondisi	Aplikasi Pengguna sudah dibuka
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Aplikasi menampilkan opsi fitur aplikasi yang dapat dipilih oleh pengguna. 2. Pengguna memilih opsi "<i>Check Status Node</i>" 3. Aplikasi menampilkan halaman check status node 4. Aplikasi menampilkan status node di komputer Pengguna

Tabel 3.4: Tabel Skenario Perintah *Sensing* Aplikasi Pengguna

Nama	<i>Mulai Sensing</i>
Deskripsi	Menampilkan data hasil sensing yang tersimpan di basis data. Proses <i>sensing</i> pada seluruh node sensor yang yang telah dikirimkan ke base-station akan disimpan di basis data, dan akan ditampilkan pada aplikasi pengguna
Aktor	Pengguna dan Admin
Pre-kondisi	Aplikasi sudah aktif dan seluruh status node telah " <i>online</i> "
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Aplikasi menampilkan opsi fitur aplikasi yang dapat dipilih oleh pengguna. 2. Pengguna menekan tombol '<i>Let's get Started</i>' pada halaman utama atau menekan link '<i>Sensing</i>' yang berada di header 3. Aplikasi menampilkan halaman sensing dan menampilkan tabel data hasil sensing yang didapatkan secara <i>real-time</i> 4. Aplikasi melakukan <i>refresh</i> halaman <i>sensing</i> secara berkala untuk memperbarui tabel data hasil <i>sensing</i>

Tabel 3.5: Tabel Skenario Menghentikan *Sensing* Aplikasi Admin

Nama	<i>Stop Sensing</i>
Deskripsi	Memberikan perintah untuk menghentikan proses <i>sensing</i> pada seluruh node sensor dan pengiriman data ke <i>base station</i>
Aktor	Admin
Pre-kondisi	<i>Base station</i> belum mengirimkan perintah sensing pada node sensor yang terhubung dalam jaringan
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Sistem menampilkan data hasil sensing yang diterima oleh base station 2. Admin memilih opsi "Stop Sensing" 3. Admin menginput opsi untuk mengeksekusi "Stop Sensing" 4. Base Station mengirimkan perintah hentikan sensing pada setiap node yang terhubung dalam jaringan 5. Node yang terhubung dalam jaringan menerima perintah hentikan sensing dan menghentikan seluruh aktifitas sensing 6. Aplikasi Admin menampilkan dialog "Sensing dihentikan !" 7. Aplikasi menampilkan kembali menu utama aplikasi Admin

Tabel 3.6: Tabel Skenario Keluar Aplikasi Aplikasi Admin

Nama	Matikan Aplikasi Base Station
Deskripsi	Memberikan perintah untuk menghentikan proses <i>sensing</i> pada seluruh node sensor dan pengiriman data oleh <i>base station</i> , serta keluar dari aplikasi
Aktor	Admin
Pre-kondisi	Aplikasi sedang melakukan <i>sensing</i> dan <i>transfer</i> data, atau pun aplikasi tidak sedang melakukan kegiatan pemantauan
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Pengguna memilih opsi "Matikan Aplikasi Base Station" 2. Aplikasi menghentikan segala proses <i>sensing</i> dan pemantauan, jika sedang dilakukan. 3. Aplikasi menampilkan pop-up dialog konfirmasi keluar dari aplikasi 4. Keluar dari aplikasi

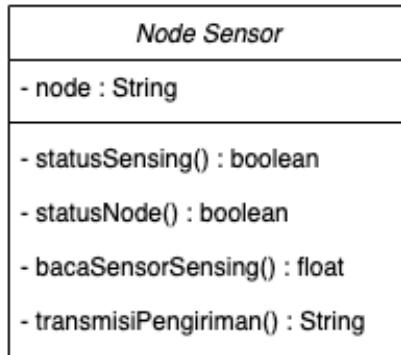
Tabel 3.7: Tabel Skenario Keluar Aplikasi Aplikasi Pengguna

Nama	Keluar Program
Deskripsi	Memberikan perintah untuk menghentikan proses <i>sensing</i> pada seluruh node sensor dan pengiriman data oleh <i>base station</i> , serta keluar dari aplikasi
Aktor	Pengguna dan Admin
Pre-kondisi	Aplikasi sedang melakukan <i>sensing</i> dan <i>transfer</i> data, ataupun aplikasi tidak sedang melakukan kegiatan pemantauan
Alur Skenario Utama	<ol style="list-style-type: none"> 1. Pengguna memilih opsi "<i>Keluar Aplikasi</i>" yang disediakan oleh browser 2. Keluar dari aplikasi

3.3.2 Analisis Kelas

Perangkat lunak yang dibangun memiliki pemodelan konseptual umum dari suatu struktur aplikasi, menggunakan kelas diagram. Dengan diagram kelas, objek-objek yang akan dibangun pada perangkat lunak akan lebih mudah untuk dirinci dan diterjemahkan, ataupun dijelaskan. Kelas-kelas yang dirancang ditujukan untuk memudahkan mengetahui fungsi-fungsi yang dimiliki oleh node sensor dan *base station*.

- Kelas Diagram Node Sensor



Gambar 3.5: Kelas Diagram Node Sensor

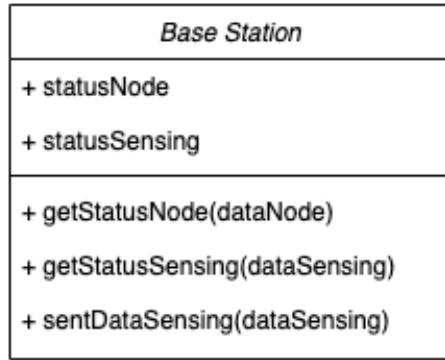
Keterangan

– Kelas Node Sensor

Kelas node sensor mempresentasikan tentang atribut yang dimiliki oleh node sensor dan fungsi-fungsi yang dapat dilakukan oleh node sensor. Node sensor memiliki atribut node untuk memberikan identitas node.

Method-method yang dimiliki oleh kelas ini antarlain method bacaSensorsSnsing dan method transmisiPengiriman. Method bacaSensorSensing berfungsi untuk mengambil data sensing yang dikirimkan oleh sensor sensing. Method lainnya adalah method statusNode dan statusSensing untuk mengetahui status keaktifan node dan status *sensing* node. Method yang digunakan untuk mengirim data hasil sensing ke *base station* adalah transmisiPengiriman yang memiliki paramater untuk memasukan data yang akan dikirimkan oleh node sensor.

- Kelas Diagram *Base station*



Gambar 3.6: Kelas Diagram *Base station*

Keterangan

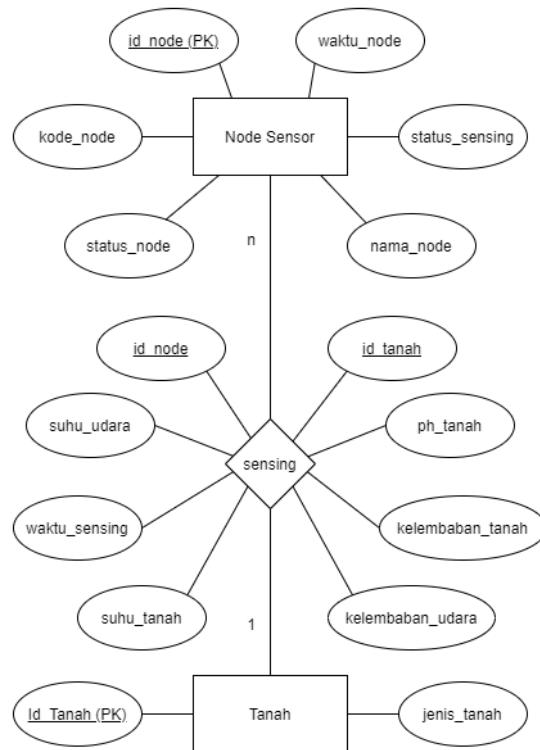
- Kelas *Base station*

Kelas base stasion memiliki lima atribut dan lima methods. Atribut yang terdapat dikelas ini adalah atribut `statusNode` dan `statusSensing`, yang berperan untuk menampung value dari status node dan status sensing yang dikirimkan oleh node sensor.

Kelas ini memiliki method `getDataSensing` dan method `sendDataSensing`. Method `getDataSensing` berperan untuk menerima data yang dikirimkan oleh node sensor. Method `sendDataSensing` digunakan untuk mengirimkan data hasil *sensing* yang dikirim oleh node sensor dan telah diterima oleh *base station*, ke *server* untuk ditampilkan ke komputer pengguna.

3.3.3 Analisis Basis Data

Basis data yang dirancang, ditujukan untuk menyimpan data hasil sensing di komputer pengguna.



Gambar 3.7: ERD Perangkat lunak

- Rancangan Konseptual Basis Data

- Entitas

- * Node Sensor

Memiliki Atribut :

- `id node`
- `kode node`
- `nama node`
- `status node`
- `status sensing`
- `waktu node`

Keterangan :

Setiap node sensor memiliki id sensor yang berbeda-beda sesuai dengan urutan terhubungnya node sensor dengan *base station*. Atribut kode node dan nama node pada node sensor, didapatkan berdasarkan atribut nama yang dikirimkan oleh node sensor. Atribut status node dan status sensing, menunjukkan kesiapan sensor untuk melakukan penerimaan data hasil *sensing* dan pengiriman data. Atribut waktu node menunjukkan waktu node dalam melakukan *sensing*.

- * Tanah

Memiliki Atribut :

- `id tanah`
- `jenis tanah`

Keterangan :

Entitas memiliki atribut id tanah yang merupakan kode petak sawah yang dilakukan penelitian. Atribut jenis tanah menunjukkan cara pengelolaan air pada tanah sawah yang diteliti.

– Relasi

- * Node Sensor – Tanah

Relasi : *One to Many*

Keterangan :

Sebuah node sensor hanya dapat melakukan *sensing* pada sebuah petak tanah sawah. Namun sebuah petak tanah sawah dapat dilakukan penelitian atau pemantauan oleh lebih dari satu node sensor.

– Tabel Relasi

- * Sensing

Keterangan : Memiliki Atribut :

- id tanah
- id node
- ph tanah
- suhu tanah
- kelembaban tanah
- suhu udara
- kelembaban udara

Atribut id tanah dan id node berperan untuk menghubungkan entitas node sensor dengan entitas tanah. Atribut pH tanah, suhu tanah, kelembaban tanah, suhu udara, dan kelembaban udara, merupakan informasi yang dimiliki oleh tanah sawah yang digunakan untuk menentukan kualitas tanah sawah.

3.3.4 Analisis Paket/Pesan

Pertama-tama admin akan menkonfigurasi node tetangga terdekat dari setiap node, agar paket dapat dikirimkan antar node jika jarak suatu node terlalu jauh untuk dikirimkan ke *base station*. Setelah itu, admin mengonfigurasi waktu pada setiap node, agar proses pengiriman data dilakukan secara bersamaan. Ketika kedua tahap tersebut telah dilakukan, saat node sensor pertama kali diaktifkan, maka node sensor akan mengganti statusnya dari *offline* menjadi *online*, lalu akan diketahui tujuan data hasil *sensing* yang akan dikirimkan, dan seluruh node memiliki waktu yang sama.

Seluruh node menunggu perintah *sensing* dari komputer pengguna. Ketika pengguna memberikan perintah *sensing*, maka seluruh node sensor akan melakukan pengambilan data. Data-data yang diambil dari tanah yang diuji, akan dikonversi menjadi sinyal digital dan dikirimkan menuju *base station*. *base station* akan menerima data hasil *sensing* secara terus menerus dan menyimpannya ke server, untuk ditampilkan di komputer pengguna.

BAB 4

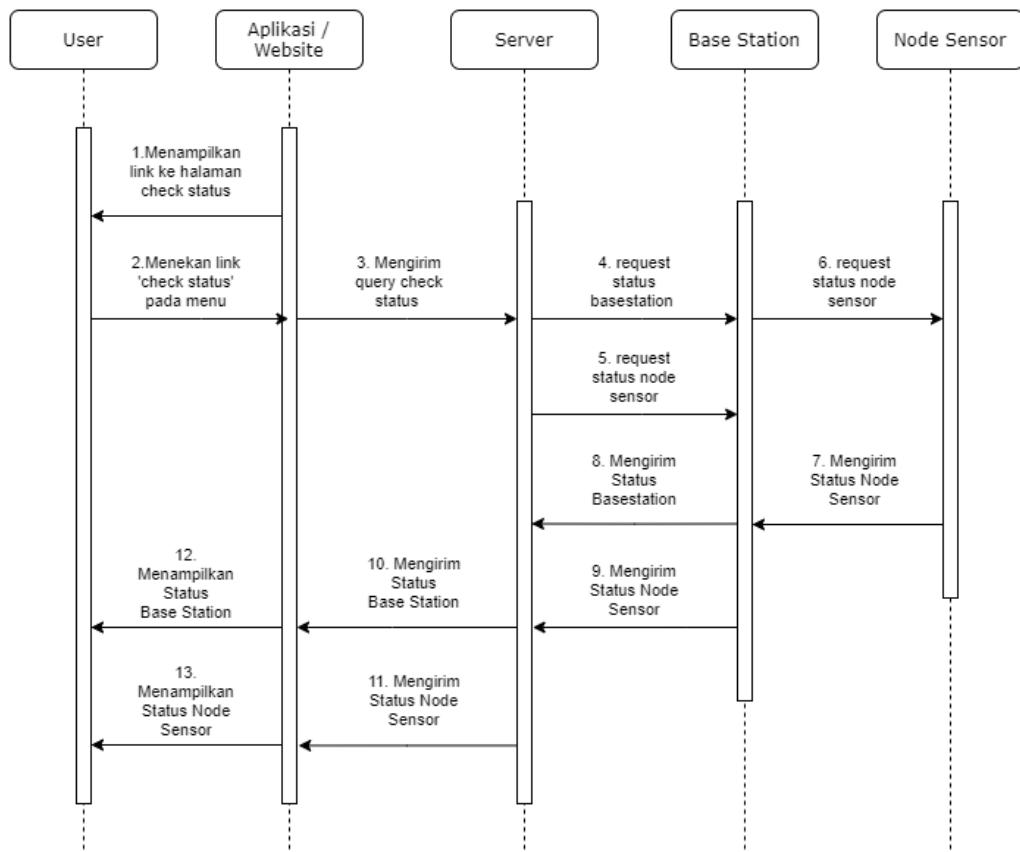
PERANCANGAN

Bab ini akan menjelaskan mengenai perancangan aplikasi yang dibangun yang merujuk pada hasil studi dan analisis yang telah dibahas pada bab sebelumnya. Pada bab ini juga akan dibahas perancangan interaksi antar node, perancangan antarmuka, perancangan kelas, perancangan masukan dan keluaran aplikasi yang dibangun.

4.1 Perancangan Interaksi Antar Node

Perancangan Interaksi antar node akan menginformasikan proses interaksi antar node dengan base station dan perangkat lunak yang dibangun untuk pengguna dan admin.

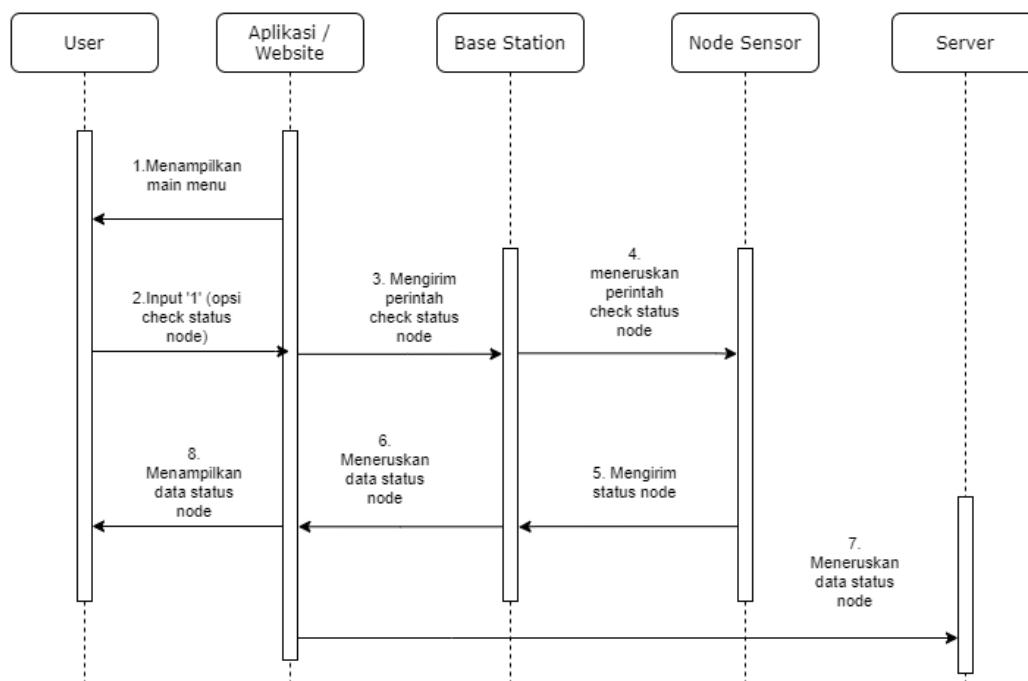
4.1.1 Diagram Sequence "Check Status Node" Aplikasi Pengguna



Gambar 4.1: Sequence Diagram Check Status Aplikasi Pengguna

Pada Check Status Node, pertama aplikasi/website akan menampilkan link ke halaman check status yang berada di header. Selanjutkan *user* dapat menekan link yang ditampilkan tersebut. Aplikasi akan mengirim perintah query yang dikirimkan ketika *user* menekan link check status node, ke server. Server akan menerima query request yang dikirimkan untuk dilanjutkan ke *base station*. *Base station* akan mengirimkan informasi status *base station* dan mengambil status seluruh node yang terhubung dalam jaringan. Ketika seluruh informasi status node diterima oleh *base station*, *base station* akan mengirim kembali informasi tersebut ke server. Server akan menerima seluruh informasi yang dikirimkan oleh *base station* untuk ditampilkan ke aplikasi. Terakhir, aplikasi akan menampilkan status terbaru dari setiap node yang terhubung dalam jaringan kepada *user*.

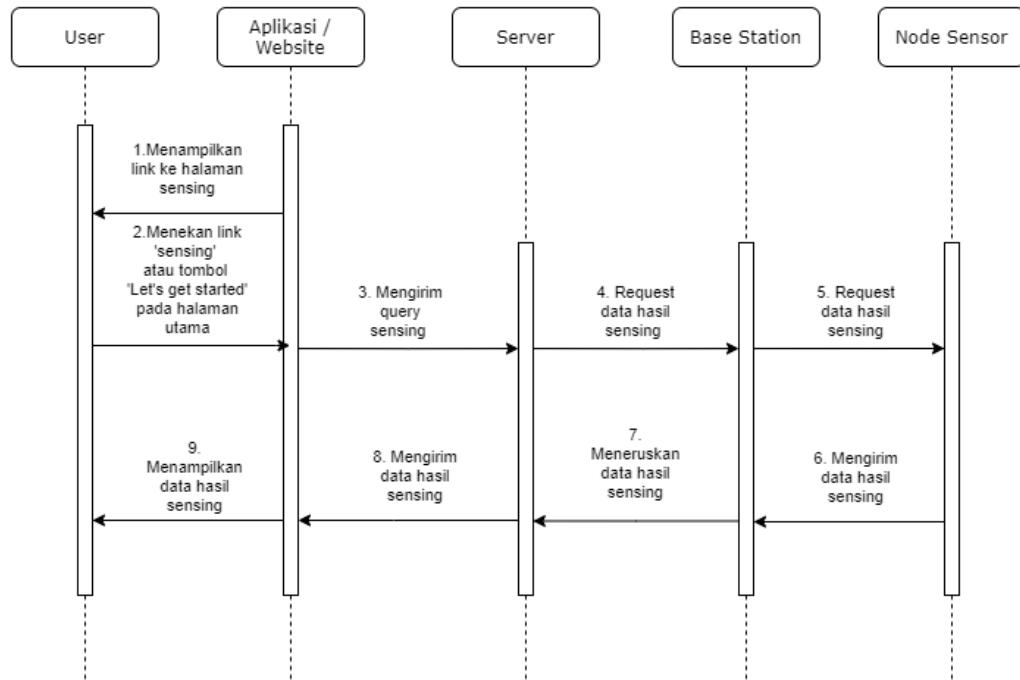
4.1.2 Diagram Sequence "Check Status Node" Aplikasi Admin



Gambar 4.2: Sequence Diagram Check Status Aplikasi Admin

Aplikasi admin akan menampilkan opsi fitur yang dapat dipilih oleh pengguna (admin) ketika program pertama kali dieksekusi. Pengguna dapat memilih opsi 'Check Status Node' dengan cara melakukan input '1' pada shell. Setelah pengguna melakukan input, aplikasi admin akan mengirim perintah check status node dari base station ke seluruh node yang terhubung dalam jaringan. Node sensor akan mulai mengirim data status nodenya masing-masing, ke base station. Base station akan menerima data status node tersebut dan meneruskannya ke aplikasi admin untuk ditampilkan ke pengguna. Base station juga akan meneruskan data status node tersebut ke server untuk disimpan.

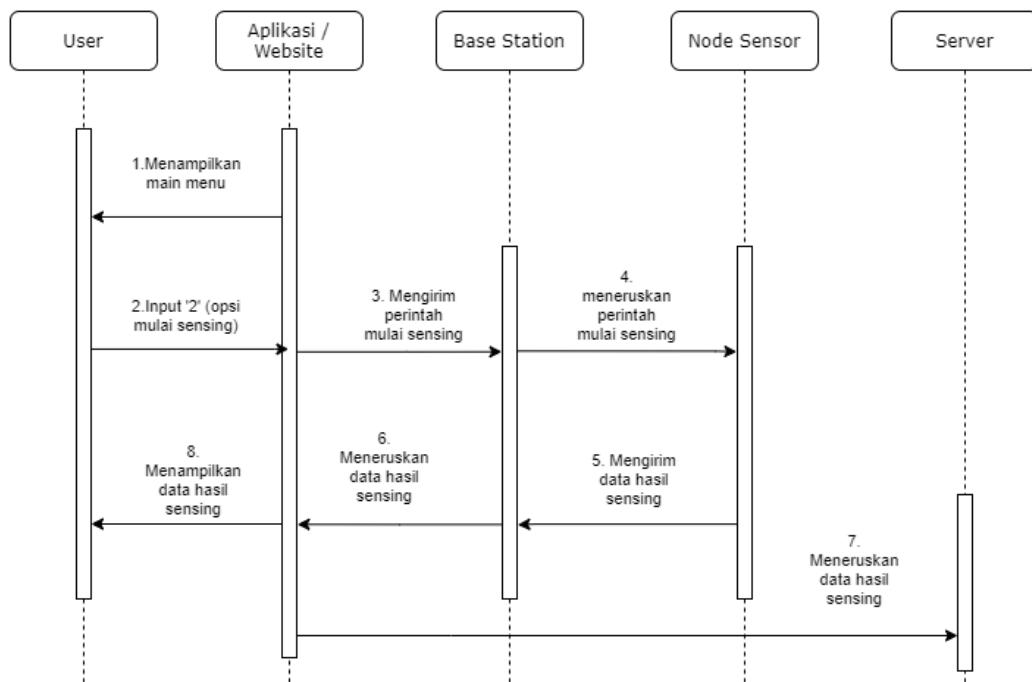
4.1.3 Diagram Sequence "Sensing" Aplikasi Pengguna



Gambar 4.3: Sequence Diagram Sensing Aplikasi Pengguna

Untuk melakukan *sensing*, pertama aplikasi/website akan menampilkan link untuk ke halaman *sensing* yang berada header atau tombol yang berada di halaman utama dengan nama '*Lets get started*'. *User* menekan link atau tombol yang berada di halaman utama tersebut untuk berpindah halaman ke halaman *sensing*. Aplikasi akan mengirim query ke server yang berfungsi untuk memberikan perintah *sensing*. Server yang menerima query tersebut akan melakukan request ke *base station* untuk melakukan *sensing*. *Base station* akan melanjukan *request* tersebut pada setiap node sensor yang terhubung dalam jaringan. Seluruh node akan melakukan *sensing*, dan mengirimkan data hasil *sensing* ke *base station* secara real time. *Base station* akan mengambil seruruh data hasil *sensing* dari setiap node dan diteruskan ke server. Server akan menyimpan data hasil *sensing* yang dikirimkan oleh *base station* untuk ditampilkan di halaman aplikasi. Aplikasi akan menampilkan data hasil *sensing* yang dikirimkan oleh node sensor dan *base station*.

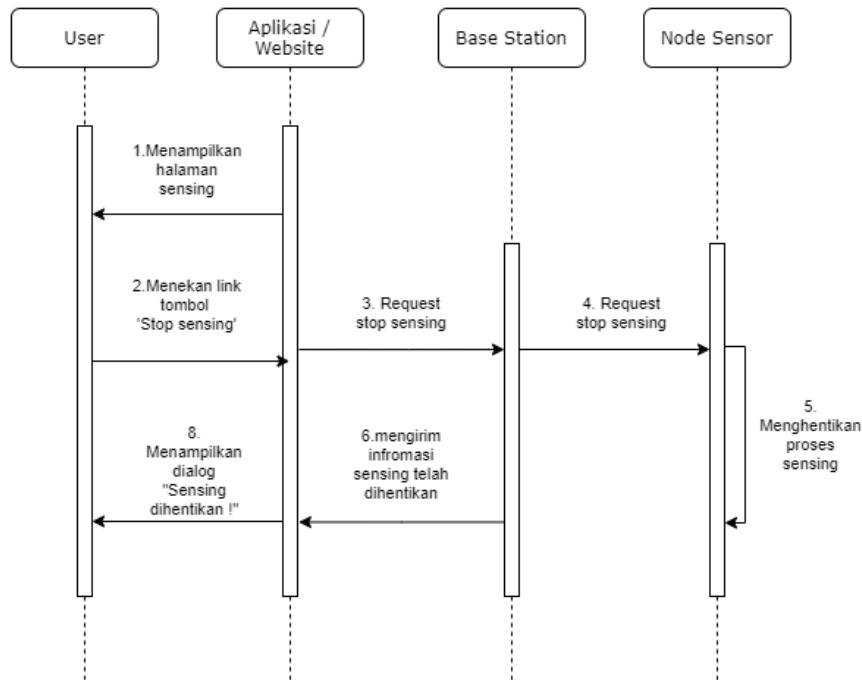
4.1.4 Diagram Sequence "Mulai Sensing" Aplikasi Admin



Gambar 4.4: Sequence Diagram Sensing Aplikasi Admin

Aplikasi admin akan menampilkan opsi fitur yang dapat dipilih oleh pengguna (admin) ketika program pertama kali dieksekusi. Pengguna dapat memilih opsi 'mulai sensing' dengan cara melakukan input '2' pada shell. Setelah pengguna melakukan input, aplikasi admin akan mengirim perintah mulai *sensing* dari base station ke seluruh node yang terhubung dalam jaringan. Node sensor akan mulai mengirim data hasil *sensing*nya ke base station. Base station akan menerima data hasil *sensing* tersebut dan meneruskannya ke aplikasi admin untuk ditampilkan ke pengguna. Base station juga akan meneruskan data hasil *sensing* tersebut ke server untuk disimpan.

4.1.5 Diagram Sequence "Stop Sensing" Aplikasi Admin



Gambar 4.5: Sequence Diagram Stop Sensing Aplikasi Admin

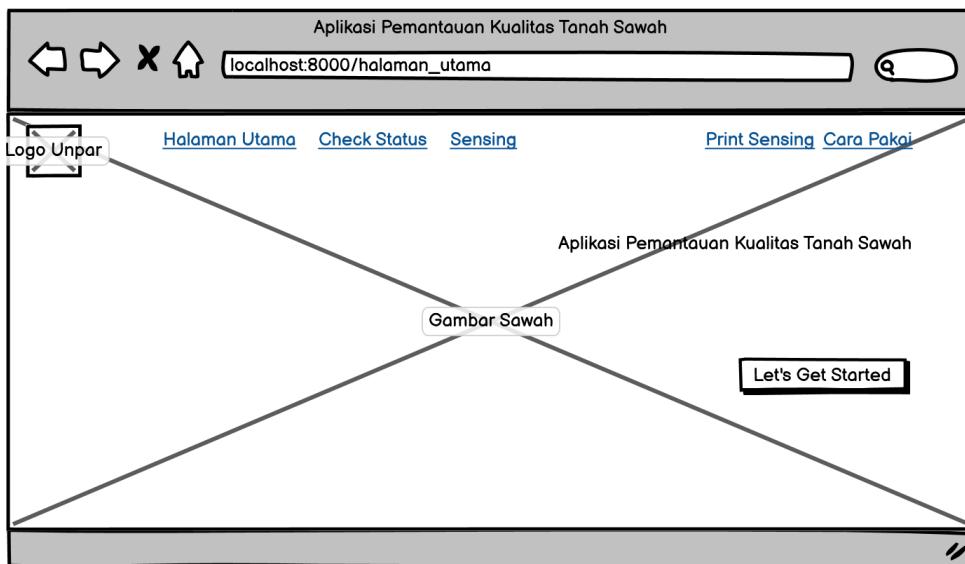
Untuk melakukan stop *sensing*, pengguna cukup memilih opsi stop *sensing* yang berada di opsi fitur (main menu). Ketika opsi stop *sensing* dipilih oleh *user* dengan cara melakukan input '3', aplikasi akan mengirim perintah dari base station ke seluruh node yang terhubung dalam jaringan. *Base station* akan mengirim perintah untuk menghentikan seluruh proses *sensing* yang dilakukan oleh node sensor yang terhubung dalam jaringan. Setelah seluruh node sensor tidak melakukan *sensing*, *base station* akan mengirimkan informasi pada server bahwa seluruh *sensing* telah dihentikan. Server akan melanjutkan informasi tersebut ke aplikasi, dan aplikasi akan menampilkan dialog "Sensing Dihentikan!"

4.2 Perancangan Antarmuka Aplikasi

Terdapat beberapa fungsi pada aplikasi yang dibangun yang memerlukan interaksi pada antarmuka seperti yang disebutkan di-subbab 3.3.1. Beberapa fungsi yang memerlukan interaksi pada antarmuka diantaranya adalah fungsi check status node, mulai *sensing*, dan stop *sensing*. Design antarmuka Aplikasi yang dibangun berdasarkan pada antarmuka website. Berikut adalah perancangan antarmuka halaman-halaman untuk aplikasi yang dibangun.

4.2.1 Mockup Halaman Utama

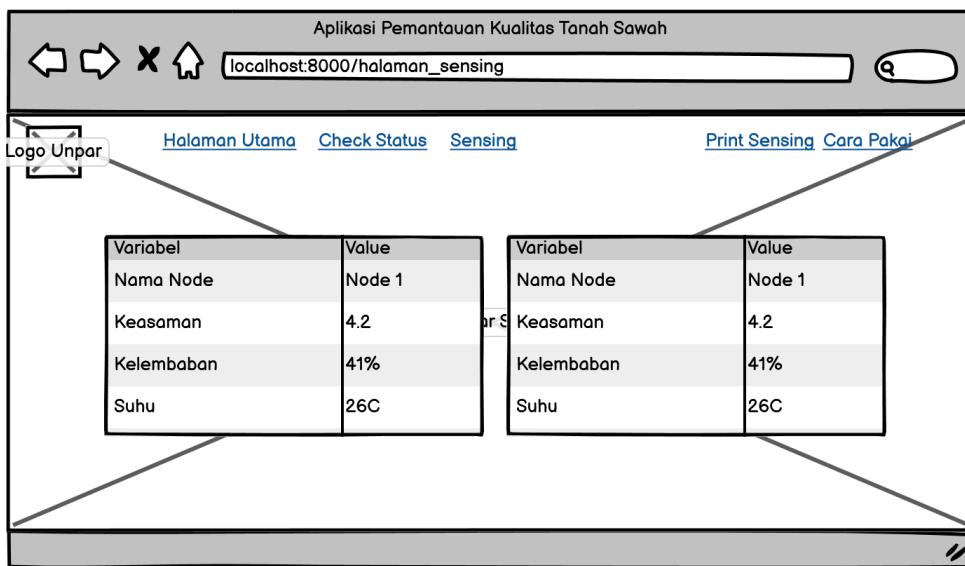
Pada halaman utama terdapat judul website dan tombol "*Let's get started*". Jika *user* menekan tombol tersebut maka antarmuka website akan diarahkan ke halaman *sensing* secara otomatis, dan *sensing* akan langsung dilakukan. Pada halaman utama juga *user* dapat memilih halaman yang ingin dikunjungi berdasarkan *link* yang tertera pada *header*.



Gambar 4.6: Mockup Halaman Utama

4.2.2 Mockup Halaman "Sensing"

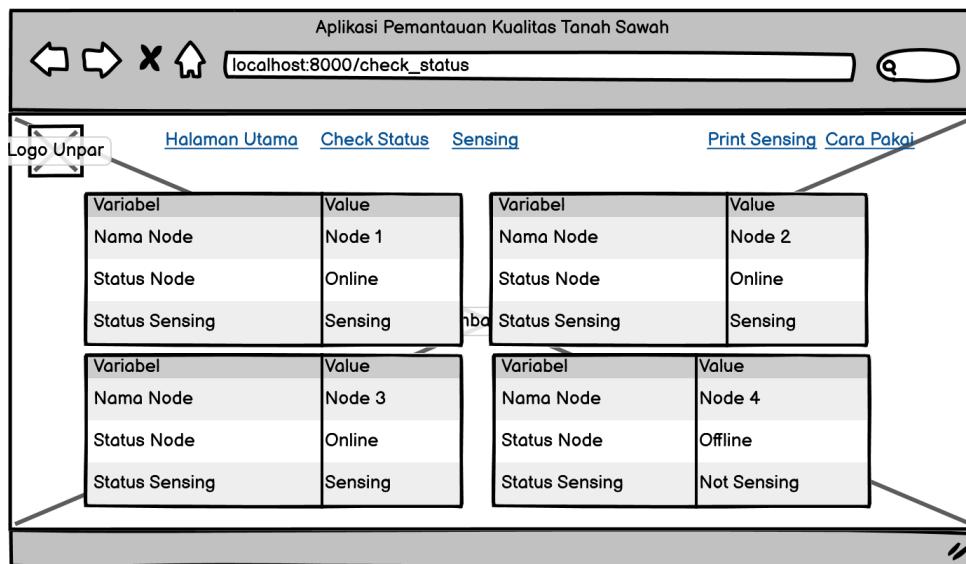
Halaman *Sensing* memiliki fungsi untuk menampilkan hasil *sensing* yang telah disimpan di basis data secara *real-time*. Sensing secara otomatis akan dilakukan ketika halaman ini dibuka, dan halaman ini akan melakukan *refresh* secara berkala untuk mendapatkan data hasil *sensing* terbaru. Pada halaman ini juga *user* dapat memulai dan menghentikan *sensing*, dengan cara menekan tombol 'mulai sensing' untuk memulai *sensing* atau 'stop sensing' untuk menghentikan *sensing*.



Gambar 4.7: Mockup Halaman Sensing Online

4.2.3 Mockup Halaman "Check Status"

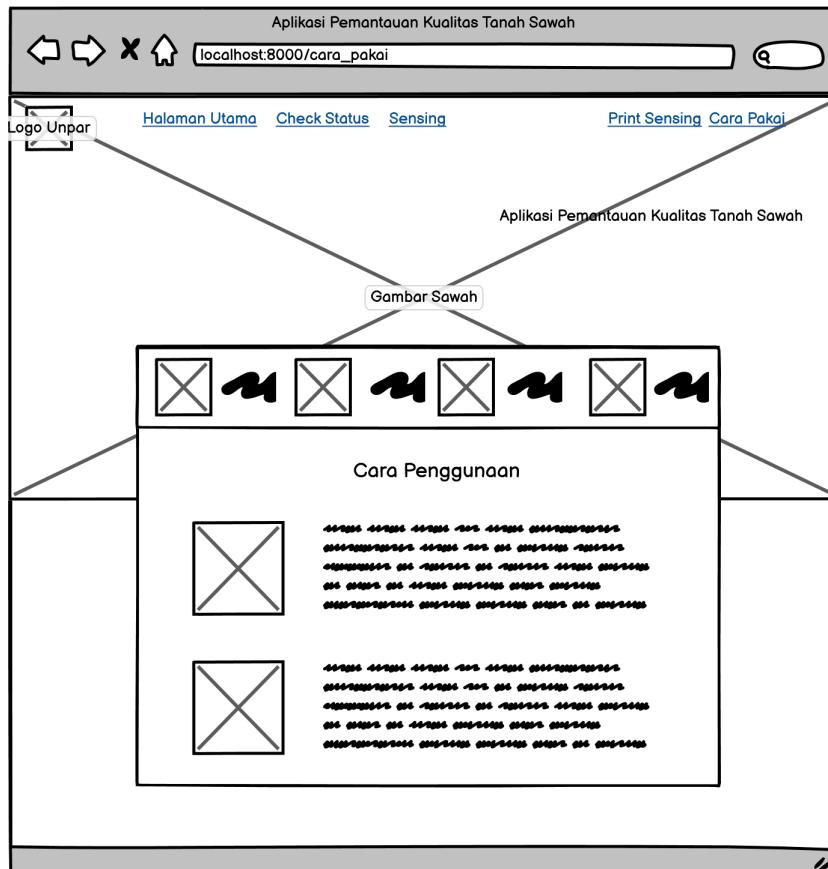
Seperti namanya, halaman check status memiliki peran untuk menampilkan status dari setiap node yang disebar. Jika status node aktif maka *value* node pada kolom node tersebut adalah '*Online*'. Sebaliknya jika status node non-aktif maka *value* node pada kolom node tersebut adalah '*Offline*'.



Gambar 4.8: Mockup Halaman Check Status

4.2.4 Mockup Halaman "Cara Penggunaan"

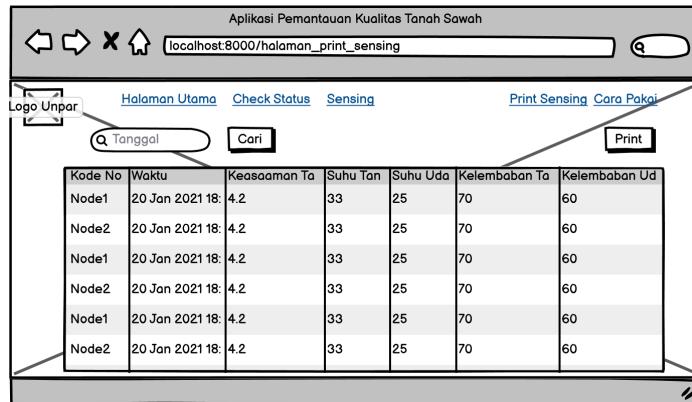
Halaman cara penggunaan adalah halaman informasi yang menunjukkan cara kerja atau pemakaian sistem pemantauan kualitas tanah sawah yang dibangun. Halaman ini meringkas tatacara penggunaan aplikasi agar mudah dimengerti oleh *user*.



Gambar 4.9: Mockup Halaman Cara Penggunaan

4.2.5 Mockup Halaman "Print Sensing"

Halaman "Print Sensing" adalah halaman yang digunakan untuk menampilkan seluruh riwayat *sensing* yang pernah dilakukan. Pengguna juga dapat mencari data riwayat *sensing* tertentu berdasarkan waktu *sensing* maupun kode petak tanah yang dilakukan pengamatan.



The mockup shows a web browser window titled "Aplikasi Pemantauan Kualitas Tanah Sawah" with the URL "localhost:8000/halaman_print_sensing". The page has a header with "Logo Unpar", "Halaman Utama", "Check Status", "Sensing", and "Print Sensing Cara Pakai". Below the header is a search bar with "Tangggol" and "Cari" buttons, and a "Print" button. The main content is a table with the following data:

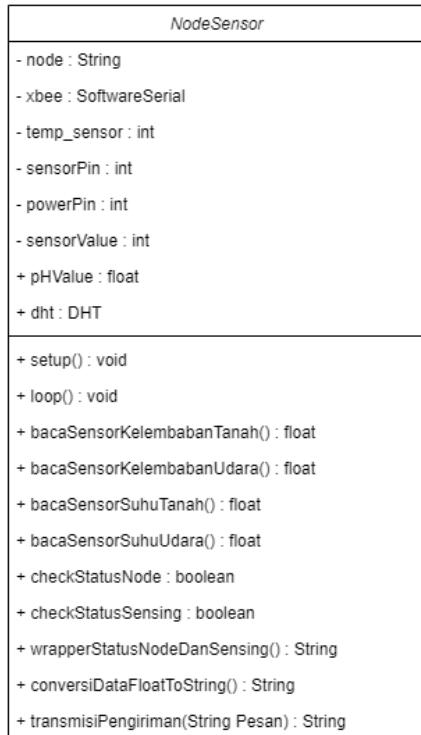
Kode No	Waktu	Keasaman Ta	Suhu Tan	Suhu Uda	Kelembaban Ta	Kelembaban Ud
Node1	20 Jan 2021 18:4.2	33	25	70	60	
Node2	20 Jan 2021 18:4.2	33	25	70	60	
Node1	20 Jan 2021 18:4.2	33	25	70	60	
Node2	20 Jan 2021 18:4.2	33	25	70	60	
Node1	20 Jan 2021 18:4.2	33	25	70	60	
Node2	20 Jan 2021 18:4.2	33	25	70	60	

Gambar 4.10: Mockup Halaman Print Sensing

4.3 Perancangan Kelas Aplikasi

Pada subbab ini akan dijelaskan kembali kelas diagram yang terdapat pada aplikasi yang dibangun secara lebih mendetil, yang telah dibahas pada subbab 3.3.2. Aplikasi yang dibangun memiliki tiga kelas utama yaitu kelas node sensor, kelas sensor *sensing*, dan kelas *base station*.

4.3.1 Kelas Node Sensor



Gambar 4.11: Kelas Diagram Node Sensor

Kelas node sensor memodelkan atribut-atribut yang dimiliki oleh node sensor dan fungsi-fungsi yang dapat dilakukan oleh node sensor. Atribut- atribut yang ada pada kelas ini adalah sebagai berikut:

- private String node

Atribut node berperan untuk mengidentifikasi node yang melakukan *sensing* berdasarkan namanya

- private SoftwareSerial xbee

Atribut xbee berperan sebagai penghubung antara perangkat xbee dengan pin yang terdapat pada perangkat arduino. Atribut ini juga berfungsi sebagai instansiasi pin yang terhubung sebagai recevier dan transivier.

- private int temp_sensor

Atribut temp_sensor memiliki peran untuk menginformasikan letak pin yang digunakan untuk melakukan pengiriman data hasil *sensing* temperatur dan kelembaban tanah.

- private int sensorPin

Atribut sensorPin merupakan atribut yang berfungsi untuk menginformasikan pin analog yang akan digunakan oleh sensor *sensing* kelembaban tanah.

- private int powerPin

Atribut powerPin berfungsi untuk mengubah pin tertentu menjadi VCC atau pengganti daya 5V.

- private int sensorValue

Atribut sensorValue berperan sebagai inialisasi data hasil *sensing* yang akan dikonversi dari Analog menjadi Digital untuk sensor *sensing* keasaman tanah.

- public float pHValue

Atribut pHValue merupakan atribut yang berfungsi sebagai instansiasi nilai pH tanah dan juga atribut penyimpan data hasil *sensing* keasaman tanah.

- DHT dht

Atribut dht adalah atribut intansiasi kelas DHT yang berfungsi untuk menginformasikan pin yang digunakan untuk mengaktifasi sensor *sensing* suhu dan kelembaban udara.

Methods yang ada pada kelas node sensor adalah sebagai berikut:

- void setup()

Seperti yang telah dibahas pada subbab 2.4.3, method setup merupakan method default yang ada ketika membangun program berbasis arduino. Method ini digunakan untuk menginisiasi seluruh atribut termasuk frekuensi komunikasi serial yang akan digunakan. Method ini juga merupakan method pertama yang akan dieksekusi ketika program dijalankan.

- void loop()

Method loop juga merupakan method default untuk pembangunan program berbasis arduino. Method loop akan dieksekusi secara terus menerus sampai perangkat akhirnya kehabisan daya atau tidak menerima daya. Method loop pada pembangunan aplikasi ini berperan sebagai wrapper yang membungkus method-method lain sebelum dikirimkan ke *base station*. Method loop juga akan menampilkan data hasil *sensing* apabila perangkat arduino terhubung dengan laptop.

- public float bacaSensorKelembabanTanah()

Method bacaSensorKelembabanTanah adalah method yang memiliki fungsi untuk melakukan pengambilan data hasil *sensing* yang dilakukan oleh sensor *sensing* kelembaban tanah. Method ini juga melakukan konversi data dari Analog menjadi Digital. Konversi dilakukan karena perangkat sensor *sensing* menggunakan arus daya untuk mengukur kelembaban suatu tanah. Sehingga data yang diterima harus dikonversi terlebih dahulu agar mendapatkan data yang valid.

- public float bacaSensorSuhuUdara()

Method bacaSensorSuhuUdara berfungsi untuk mengambil dan mengembalikan data suhu udara, hasil *sensing* sensor *sensing* yang terhubung dengan perangkat arduino (node sensor).

- public float bacaSensorKelembabanUdara()

Method bacaSensorSuhuUdara memiliki fungsi yang sama dengan method bacaSensorSuhuUdara. Namun data yang diambil dan dikembalikan oleh method ini adalah data kelembaban udara.

- public float bacaSensorSuhuTanah()

Method bacaSensorSuhuTanah berperan untuk melakukan *sensing* terhadap suhu tanah yang dilakukan pengamatan. Method ini juga akan mengembalikan nilai suhu tanah dari hasil *sensing* yang dilakukan.

- public float bacaSensorPHTanah()

Seperti method bacaSensorKelembabanTanah, method bacaSensorPHTanah perlu melakukan konversi data hasil *sensing*. Namun, untuk method ini data yang perlu dikonversi adalah data keasaman tanah yang dilakukan oleh perangkat *sensing* pH tanah.

- public boolean checkStatusNode()

Method checkStatusNode berfungsi untuk menginformasikan status keaktifan node. Method akan mengembalikan nilai *true* apabila node aktif (*online*), dan akan mengembalikan nilai *false* apabila node tidak aktif (*offline*).

- public boolean checkStatusSensing()

Method checkStatusSensing memiliki fungsi untuk menginformasikan status *sensing* dari node sensor. Apabila sensor sedang melakukan *sensing* maka method akan mengembalikan nilai *true*. Namun jika sensor tidak melakukan *sensing* maka method akan mengembalikan nilai *false*.

- public String wrapperStatusNodeDanSensing()

Method wrapperStatusNodeDanSensing berfungsi sebagai method pembungkus antara method checkStatusNode dan checkStatusSensing. Method ini dibuat agar program dapat ditulis dengan rapi dan mudah untuk dimengerti. Selain itu method checkStatusNode dan checkStatusSensing merupakan method yang berhubungan langsung dengan perangkat node sensor.

- public String conversiDataFloatToString()

Untuk dapat mengirimkan data tersebut menggunakan xbee, maka seluruh tipe data hasil *sensing* harus dikonversi terlebih dahulu dari float menjadi string. Setelah data telah dikonversi maka hasil konversi akan dimasukan kedalam parameter method transmisiPengiriman untuk dikirimkan ke *base station*.

- public String transmisiPengiriman(String pesan)

Method transmisiPengiriman merupakan method terakhir yang akan dieksekusi didalam method loop. Method ini bertanggung jawab dalam pengiriman hasil pesan yang telah dikonversi ke *base station*. Setelah method ini dieksekusi, maka proses loop (melakukan *sensing* berikutnya) baru kembali dilakukan.

4.3.2 Kelas Base Station



Gambar 4.12: Kelas Diagram *Base station*

Kelas *Base station* mempresentasikan atribut dan fungsi SINK terhubung dengan node sensor dalam jaringan. Atribut- atribut yang ada pada kelas ini adalah sebagai berikut:

- ser

Atribut ser berfungsi sebagai inialisasi pin yang menjadi penghubung antar perangkat Raspberry Pi dengan perangkat XBee. Atribut ini juga menginstansiasi *port* (pin) yang terhubung dengan xbee, berserta frekuensi komunikasi serial yang digunakan agar dapat bertukar pesan dengan node sensor.

- POOL_SIZE

Atribut POOL_SIZE berperan sebagai atribut penampung yang menginformasikan banyaknya node yang terhubung dalam jaringan. Atribut ini akan digunakan dalam *thread* agar data yang dikirim oleh node dapat diterima, walaupun data diterima dalam waktu bersamaan.

- appRunning

Atribut appRunning digunakan untuk menginisiasi kondisi *loop* yang dieksekusi pertama kali ketika program dieksekusi. Atribut appRunning akan terus bernilai *True* selama tidak ada interrupt dari keyboard. Selama appRunning bernilai *True* maka program akan terus dieksekusi didalam *loop*.

- menuShow

Sama seperti appRunning, menuShow berperan untuk menginisiasi kondisi *loop*. Namun menuShow digunakan didalam *loop* appRunning, yang memiliki fungsi untuk terus menampilkan opsi fitur aplikasi *base station* (*main menu*) ketika suatu perintah telah di eksekusi.

- sensingRunning

SensingRunning juga berperan sebagai inialisasi dari suatu kondisi loop. Namun, SensingRunning digunakan khusus untuk melakukan *loop sensing* secara terus menerus, sehingga data hasil *sensing* yang dikirimkan oleh node sensor dapat diterima dan tampilan pada *shell* aplikasi.

- statusSensing

Atribut statusSensing digunakan untuk menampung nilai status pengamatan yang dikirimkan oleh node sensor. Status bernilai '*True*' jika sensor sedang melakukan *sensing*, dan bernilai '*False*' jika sebaliknya.

- statusNode

Seperti statusSensing, status node juga digunakan untuk menampung nilai. Namun nilai yang ditampung dalam atribut ini adalah status keaktifan node. Jika node berstatus '*Online*' maka nilai dari status node adalah '*True*', begitupula sebaliknya jika node berstatus '*Offline*' maka nilai dari status node adalah '*False*'.

- counter

Atribut counter digunakan sebagai nilai penghitung batas waktu tunggu respon dari node ketika *base station* telah mengirim perintah.

- respon

Atribut respon digunakan sebagai atribut yang menampung jumlah node yang merespon ketika *base station* mengirim perintah.

- inputNum

Atribut inputNum berfungsi untuk menampung nilai dari *input* yang dimasukan oleh Admin. Nilai dari atribut ini yang akan digunakan untuk memilih opsi instruksi dari fitur yang akan dieksekusi.

- finding

Atribut finding digunakan untuk menampung nilai dari status sensing yang didapatkan dari respon node sensor dari perintah 'stop sensing' oleh base station.

Methods yang ada pada kelas node sensor adalah sebagai berikut:

- getDataSensing(dataSensing)

Method getDataSensing berfungsi untuk mendapatkan seluruh data hasil *sensing* yang dikirimkan oleh masing-masing node yang terhubung dalam jaringan

- sendDataSensing(dataSensing)

Method sendDataSensing berfungsi untuk mengirim seluruh data yang telah diterima oleh *base station* ke server

- mainMenu

Method mainMenu berfungsi untuk menampilkan daftar opsi fitur yang dapat dipilih oleh admin untuk mengirim perintah ke node sensor dari *base station*

- counterAdd

Method counterAdd berfungsi untuk menambahkan nilai atribut counter juga untuk membatasi waktu tunggu respon node sensor setelah *base station* mengirim perintah

- counterSet

Method counterSet berfungsi untuk melakukan *reset* pada nilai atribut counter setelah batas waktu tunggu respon node sensor habis.

- updateStatusSensing

Method updateStatusSensing berperan untuk melakukan *update* terhadap basis data ketika status sensing diterima oleh *base station*.

- goingOffline

Method goingOffline digunakan untuk melakukan *update* terhadap basis data ketika aplikasi *base station* dimatikan.

- terimaRespon

Method terimaRespon berperan untuk menghitung jumlah node yang membalas atau merespon ketika *base station* telah mengirim perintah

4.4 Perancangan *Input* dan *Output*

Aplikasi ini berbasis website sehingga menggunakan *button* pada halaman untuk menerima *input* dan tabel pada halaman untuk menampilkan data. Aplikasi yang dibangun juga memiliki dua halaman dengan fungsi yang berbeda-beda sebagai fungsi utama. Halaman yang dimaksud adalah halaman *sensing* dan halaman check status node. Pada halaman 'Sensing' *user* dapat menekan tombol mulai *sensing* untuk memulai *sensing* pada tanah yang diuji. Pada halaman *sensing* juga, *user* dapat melakukan aktifitas hentikan *sensing* dengan menekan tombol 'stop *sensing*'. Untuk menampilkan status setiap node, *user* dapat menekan link yang merujuk pada halaman check status node, dan aplikasi akan menampilkan status dari setiap node yang terhubung dalam jaringan.

BAB 5

IMPLEMENTASI DAN PENGUJIAN

Bab ini terdiri dari implementasi perangkat keras maupun lunak yang telah dibahas pada bab 4. Pada bab ini juga akan dibahas pengujian yang dilakukan atas aplikasi yang dibangun, dipandang dari hasil pengujian perangkat keras maupun pengujian perangkat lunak yang digunakan. Selain hal yang telah disebutkan diatas, bab ini juga akan membahas kendala dan kesulitan yang dialami selama perancangan sampai pengujian aplikasi dilakukan.

5.1 Implementasi

Implementasi merupakan kegiatan atau tindakan yang dilakukan berdasarkan hasil perancangan tertentu, yang bertujuan untuk mencapai tujuan tertentu. Pada aplikasi yang dibangun, implementasi dilakukan di tanah persawahan dengan tujuan untuk menguji aplikasi yang dibangun serta mendapatkan data unsur-hara tanah dari hasil *sensing* yang dilakukan selama implementasi.

5.1.1 Lingkungan Implementasi

Berikut adalah perangkat keras yang digunakan untuk melakukan implementasi, juga perangkat lunak yang dioperasikan pada perangkat keras tersebut.

- Laptop

Spesifikasi laptop yang digunakan dalam implementasi aplikasi adalah sebagai berikut.

- Perangkat Keras

- * *Processor* : Intel(R) Core(TM) i5-7200
 - * *Memory* : 12288MB

- Perangkat Lunak

- * *Operation System* : Windows 10 Home 64-bit
 - * *Remote access (CLI)* : PuTTy
 - * *Remote access (GUI)* : VNC
 - * IDE : Arduino IDE

- Raspberry Pi 3 B+

Spesifikasi Raspberry Pi yang berperan sebagai *base station* dalam implementasi aplikasi adalah sebagai berikut.

- Perangkat Keras

- * *Processor* : 1.4GHz 64-bit quad-core
 - * *Memory* : 1000MB
 - * *Power* : 2.5A

- Perangkat Lunak
 - * *Operation System* : Raspbian Pi
 - * *Remote access (GUI)* : VNC
 - * *Browser* : Chromium
 - * *PHP Version* : 7.0.1
 - * *Framework* : Laravel 8.x
- *Powerbank*

Powerbank yang berperan sebagai pemberi daya *base station* untuk kebutuhan implementasi atau pengujian diluar ruangan. Spesifikasi *powerbank* yang digunakan adalah sebagai berikut.

 - Model : Anker Astro E1
 - Capacity : 6700mAh
 - Input/Output : 5V=1A / 5V=2A
- Arduino Mega 2560
 - *Chip* : ATMEGA2560
 - *Memory* : 256KB
 - Jumlah I/O : 54
- Sensor *sensing*

Sensor *sensing* yang terhubung dengan perangkat Arduino terdiri dari sensor kelembaban tanah, sensor keasaman tanah (pH tanah), sensor suhu dan kelembaban udara, sensor suhu tanah, dan perangkat X-Bee untuk komunikasi.

5.1.2 Hasil Implementasi

Berdasarkan pembahasan pada bab 4, terdapat dua buah kelas yang dilakukan secara terpisah, yaitu pemograman pada node sensor (Kelas Node Sensor) yang dilakukan menggunakan bahasa Arduino dan pemrograman pada *base station* (Kelas Base-Station) yang dilakukan menggunakan bahasa Python.

Kelas Node Sensor

Pada kelas Node sensor terdapat barisan code program '#include' yang berfungsi untuk mengimport library-library. Barisan kode tersebut diperlukan agar sensor *sensing* dapat melakukan pengambilan data. Setelah kode program untuk mengimport library, maka terdapat banyak inialisasi untuk setiap sensor *sensing* yang terhubung dengan perangkat Arduino.

Pada kelas ini terdapat dua buah methods default Arduino yaitu setup dan loop. Method setup merupakan method yang pertama kali dieksekusi oleh program ketika mendapat power up. Method setup akan menginformasikan serial komunikasi (dalam aplikasi ini baut 9600) juga menginisialisasi pin-pin yang akan digunakan untuk melakukan transfer data hasil *sensing* maupun pengiriman pesan.

Kode 5.1: Metode setup()

```
void setup(){
  Serial.begin(9600);
  dht.begin();

  // jadikan pin power sebagai output
  pinMode(powerPin, OUTPUT);

  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
```

```
// default bernilai LOW
digitalWrite(powerPin, LOW);

// mulai komunikasi serial
xbee.begin(9600);
}
```

- Melakukan *sensing*

Perangkat Arduino yang berperan sebagai node sensor akan mengambil data yang dilakukan oleh sensor *sensing* dan disimpan sementara sebelum dikirimkan ke *base station*.

Pada perangkat ini terdapat beberapa pemrograman method untuk mengambil hasil *sensing* dari setiap sensor *sensing* yang terhubung dengan perangkat Arduino.

- Method loop

Method loop yang akan terus dieksekusi selama Arduino memiliki daya untuk tetap menyala. Method loop pada kelas ini dijadikan sebagai wrapper, yang merupakan method penampung hasil method-method lainnya. Method loop juga berperan untuk menampilkan data hasil *sensing* ke aplikasi sebelum paket dikirimkan ke *base station*.

Kode 5.2: Metode loop()

```
void loop(){
    Serial.println("");
    Serial.print("Node_pengiriman:_");
    Serial.println(nodeID);
    Serial.print("Kelembaban_Tanah:_");
    Serial.print(bacaSensorKelembabanTanah());
    Serial.println("-_10_%");
    Serial.print("Kelembaban_Udara:_");
    Serial.print(bacaSensorKelembabanUdara());
    Serial.println("_%");
    Serial.print("Suhu_Tanah:_");
    Serial.print(bacaSensorSuhuTanah());
    Serial.println("_C");
    Serial.print("Suhu_Udara:_");
    Serial.print(bacaSensorSuhuUdara());
    Serial.println("_C");
    Serial.print("Kadar_Keasaman_Tanah_(pH):_");
    Serial.println(bacaSensorPhTanah());
    Serial.println("");
    Serial.println(WrapperStatusNodeDanSensing());
```

- Method bacaSensorKelembabanTanah

Method ini berfungsi untuk mendapatkan data hasil *sensing* kelembaban tanah, dari sensor *sensing* yang tertancap di tanah yang diuji.

Kode 5.3: Metode bacaSensorKelembabanTanah()

```
float bacaSensorKelembabanTanah() {
    // hidupkan power
    digitalWrite(powerPin, HIGH);
    delay(2000);

    // baca nilai analog dari sensor
    int nilaiSensor = analogRead(sensorPin);
    digitalWrite(powerPin, LOW);

    // makin lembab maka makin tinggi nilai outputnya
    // konversi sensor kelembaban tanah (analog ke digital)
    return (1023 - nilaiSensor)/10;
}
```

- Method bacaSensorKelembabanUdara

Method bacaSensorKelembabanUdara memiliki peran untuk mengambil data kelembaban udara dari area persawahan yang diuji.

Kode 5.4: Metode bacaSensorKelembabanUdara()

```
float bacaSensorKelembabanUdara(){
    float kelembaban = dht.readHumidity();
    return kelembaban;
}
```

- Method bacaSensorSuhuTanah

Method ini berfungsi untuk mengetahui suhu tanah yang diuji, dengan cara mengambil data dari hasil *sensing* yang dilakukan sensor *sensing*, yang ditancapkan ke tanah yang dilakukan pengamatan.

Kode 5.5: Metode bacaSensorSuhuTanah()

```
float bacaSensorSuhuTanah(){
    sensorSuhuTanah.requestTemperatures();
    float suhuTanah = sensorSuhuTanah.getTempCByIndex(0);
    return suhuTanah;
}
```

- Method bacaSensorSuhuUdara

Seperti method bacaSensorSuhuTanah, method bacaSensorSuhuUdara juga mengambil data suhu. Namun data suhu yang diambil adalah suhu udara dari area yang dilakukan pengamatan.

Kode 5.6: Metode bacaSensorSuhuUdara()

```
float bacaSensorSuhuUdara(){
    float suhu = dht.readTemperature();
    return suhu;
}
```

- Method bacaSensorPhTanah

Method bacaSensorPhTanah berfungsi untuk mendapatkan data tingkat keasaman tanah yang diuji. Method ini akan menerima data dari sensor *sensing* yang ditancapkan ditanah dan melakukan konversi sinyal sebelum dikirimkan ke *base station*.

Kode 5.7: Metode bacaSensorPhTanah()

```
float bacaSensorPhTanah(){
    //mengambil data \textit{sensing} sensor
    sensorValue = analogRead(analogInPin);

    //Mathematical conversion from ADC to pH
    //Konversi analog menjadi digital
    //rumus didapat berdasarkan datasheet
    phValue = (-0.0693*sensorValue)+7.3855;

    return phValue*-1;
}
```

- Method conversiDataFloatToString

Method conversiDataFloatToString berfungsi untuk mengubah seluruh hasil data *sensing* yang umumnya berjenis data float menjadi string agar dapat dikirimkan ke *baseStation* melalui method transmisiPengiriman.

Kode 5.8: Metode conversiDataFloatToString()

```
String conversiDataFloatToString(){
    String kelembabanTanah =
        String(bacaSensorKelembabanTanah());
    String kelembabanUdara =
        String(bacaSensorKelembabanUdara());
    String suhuTanah = String(bacaSensorSuhuTanah());
    String suhuUdara = String(bacaSensorSuhuUdara());
    String phTanah = String(bacaSensorPhTanah());
    String res = node2+"|"+kelembabanTanah+"|"+
    kelembabanUdara+"|"+suhuTanah+"|"+
    suhuUdara+"|"+phTanah+"|"+
    WrapperStatusNodeDanSensing();
    return res;
}
```

- Melakukan Respon terhadap perintah *Base Station*

Node dapat melakukan instruksi yang dikirim oleh *base station*, dan dapat mengirim respon untuk membalas perintah tertentu yang dikirimkan. Barisan kode dibawah ini terdapat didalam void loop, sehingga aplikasi akan menunggu input instruksi secara berkala.

Kode 5.9: Respon Perintah Base Station

```
/* ===== Perintah dari Base Station===== */
//terdapat input perintah dari base station
if (xbee.available()) {
    boolean adaPerintah = true;
    char temp= xbee.read();
    if(temp=='a'){ //mulai sensing
        String pesan = conversiDataFloatToString();
        transmisiPengiriman(pesan);
    }
    else if(temp == 'b'){ //check status
        String pesan = WrapperStatusNodeDanSensing();
        transmisiPengiriman(pesan);
    }
    else if(temp == 'c'){ //stop sensing
        adaPerintah = false;
        while(xbee.available()){ //cek perintah base station
            if(adaPerintah == false){
                //selama ga ada perintah >> kunci loop
                delay(5000);
            }
            else{
                //keluar kunci loop dan kembali ke void loop
                break;
            }
        }
    }
}
```

- Konversi data Analog menjadi Digital

Beberapa perangkat sensor *sensing* memiliki spesifikasi pengambilan data secara analog. Perangkat sensor *sensing* dengan pengambilan data analog adalah sensor keasaman tanah (pH Tanah) dan kelembaban tanah. Oleh karena itu, diperlukan konversi tipe data dari analog menjadi digital sebelum dikirimkan ke *base station*.

Kode 5.10: Metode bacaSensorPhTanah()

```
float bacaSensorPhTanah(){
    //mengambil data sensing sensor
    sensorValue = analogRead(analogInPin);

    //Mathematical conversion from ADC to pH
    //Konversi analog menjadi digital
    //rumus didapat berdasarkan datasheet
    phValue = (-0.0693*sensorValue)+7.3855;

    return phValue*-1;
}
```

Kode 5.11: Metode bacaSensorKelembabanTanah()

```
float bacaSensorKelembabanTanah() {
    // hidupkan power
    digitalWrite(powerPin, HIGH);
    delay(2000);

    // baca nilai analog dari sensor
    int nilaiSensor = analogRead(sensorPin);
    digitalWrite(powerPin, LOW);

    // makin lembab maka makin tinggi nilai outputnya
    // konversi sensor kelembaban tanah (analog ke digital)
    return (1023 - nilaiSensor)/10;
}
```

- Mengirim status node dan status *sensing*

Selain mengirimkan data hasil *sensing*, node sensor juga mengirim status aktif dari node dan status *sensing*. Untuk mengetahui nilai status sensor node, method checkstatusnode akan mengembalikan nilai *true* jika *online*, dan *false* jika *offline*.

Kode 5.12: Metode checkStatusNode()

```
boolean checkStatusNode(){
    if(checkStatusSensing()==true){
        return true;
    }
    else{ //check lampu indikator 'on' Arduino
```

```

    if(digitalRead(LED_BUILTIN == HIGH)){
        return true;
    }
    else{
        return false;
    }
}

```

Untuk mengetahui nilai status *sensing*, method checkStatusSensing akan dieksekusi. Seperti method checkStatusNode, checkStatusSensing akan mengembalikan nilai *true* apabila node sendang melakukan pengamatan atau *sensing*, dan mengembalikan nilai *false* apabila tidak melakukan pengamatan atau *sensing*.

Kode 5.13: Metode checkStatusSensing()

```

boolean checkStatusSensing(){
    boolean kelembabanTanah = false;
    boolean kelembabanUdara = false;
    boolean suhuTanah = false;
    boolean suhuUdara = false;

    // ph Tanah tidak memiliki value pasti-
    // ketika sensor bermasalah
    boolean phTanah = true;

    boolean isSensing = false;

    if(bacaSensorKelembabanTanah()!=1023.0){
        //1023.00 adalah value yang didapatkan-
        //jika sensor suhu tanah bermasalah
        kelembabanTanah = true;
    }
    if(isnan(bacaSensorKelembabanUdara())==false){
        //bukan nan (mengembalikan value),
        //jika bermasalah value = nan
        kelembabanUdara = true;
    }
    if(bacaSensorSuhuTanah()!=-127.00){
        //-127.00 adalah value yang didapatkan-
        //jika sensor suhu tanah bermasalah
        suhuTanah = true;
    }
    if(isnan(bacaSensorSuhuUdara())==false){
        //bukan nan (mengembalikan value),
        //jika bermasalah value = nan
        suhuUdara = true;
    }

    if((kelembabanTanah&&kelembabanUdara&&
    suhuTanah&&suhuUdara&&phTanah)==true){
        isSensing = true;
    }
    return isSensing;
}

```

- Mengirim data hasil *sensing*

Setelah data hasil *sensing* didapatkan dan di konversi dengan baik, maka node sensor harus mengirimkan data hasil *sensing* tersebut sebelum melakukan pengambilan data *sensing* berikutnya.

Kode 5.14: Metode transmisiPengiriman()

```

String transmisiPengiriman(String pesan){
    xbee.print(pesan);
}

```

Kode program kelas ini secara lengkap dapat dilihat pada lampiran A.

Kelas Base Station

Pada kelas *base station* terdapat dua buah method utama yang akan selalu dieksekusi yaitu getDataSensing dan sentDataSensing.

- Menerima data hasil *sensing*

Method `getDataSensing` berperan untuk mengambil data yang dikirimkan oleh node sensor yang berkomunikasi menggunakan perangkat X-bee dari receiver (node sensor) ke coordinator (*base station*). Method `getDataSensing` juga berfungsi untuk menampilkan hasil data *sensing* yang dikirimkan oleh node untuk menunjukan bahwa paket telah diterima dan siap untuk dikirimkan dan disimpan di internet (localhost).

Kode 5.15: Metode `getDataSensing()`

```
def getDataSensing(dataSensing):
    # Data yang dikirim Arduino (String)
    # "Node 1"+|"kelembabanTanah+"|"kelembabanUdara+"|"suhuTanah+"| "
    #|"suhuUdara+"|"phTanah+"|"statusNode+"|"statusSensing";

    potongData = dataSensing.split("|")

    if len(potongData) > 1:
        nodeSensing = potongData[0]
        kelembabanTanah = potongData[1]
        kelembabanUdara = potongData[2]
        suhuTanah = potongData[3]
        suhuUdara = potongData[4]
        phTanah = potongData[5]
        statusNode = potongData[6]
        statusSensing = potongData[7]

        waktuSensing = datetime.datetime.now()
        waktuSensing = waktuSensing.strftime('%Y-%m-%d %H:%M:%S')

    return nodeSensing,kelembabanTanah,kelembabanUdara,
           suhuTanah,suhuUdara,phTanah,statusNode,statusSensing,
           waktuSensing
```

- Menyamakan waktu node

Setiap node akan mengirimkan hasil *sensing* beserta waktu data hasil *sensing* diterima oleh *base station*.

Kode 5.16: Metode `getDataSensing()`

```
waktuSensing = datetime.datetime.now()
waktuSensing = waktuSensing.strftime('%Y-%m-%d %H:%M:%S')
```

- Menyimpan data hasil *sensing*

Setelah data diterima dengan baik oleh *base station*, maka *base station* akan meneruskan data hasil *sensing* tersebut ke basisdata (internet) yang dalam pembangunan aplikasi ini tersimpan di localhost.

Kode 5.17: Metode `sentDataSensing()`

```
def sentDataSensing(dataSensing):
    db = mysql.connector.connect(
        host = 'localhost',
        database = 'Skripsi_II',
        user = 'admin',
        password = 'root',
        pool_name = 'mypool',
        pool_size = POOL_SIZE+1
    )

    nodeSensing = dataSensing[0]
    kelembabanTanah = dataSensing[1]
    kelembabanUdara = dataSensing[2]
    suhuTanah = dataSensing[3]
    suhuUdara = dataSensing[4]
    phTanah = dataSensing[5]
    statusNode = dataSensing[6]
    statusSensing = dataSensing[7]

    kelembabanTanahFlt = float(kelembabanTanah)
    kelembabanUdaraFlt = float(kelembabanUdara)/10.0
    suhuTanahFlt = float(suhuTanah)
    suhuUdaraFlt = float(suhuUdara)
    phTanahFlt = float(phTanah)

    getNodeNumber = nodeSensing[len(nodeSensing)-1]
    getNodeNumberInt = int(getNodeNumber)

    waktuSensing = datetime.datetime.now()
    waktuSensing = waktuSensing.strftime('%Y-%m-%d %H:%M:%S')
```

```

jenisTanah = "irigasi"

#cursor digunakan untuk memasukan data ke mysql, dan eksekusi query
cursor = db.cursor(buffered=True)

queryTanah = ("INSERT INTO_sensing(waktu_sensing,ph_tanah,
suhu_tanah,kelembaban_tanah,suhu_udara,
kelembaban_udara,kode_petak,jenis_tanah)
VALUES_(%,%,%,%,%,%,%,%)")

queryInputTanah = (waktuSensing,phTanahFlt,suhuTanahFlt,
kelembabanTanahFlt,suhuUdaraFlt,
kelembabanUdaraFlt,getNodeNumberInt,
jenisTanah)

cursor.execute(queryTanah,queryInputTanah)

queryNode2 = ("UPDATE_nodesensor_SET_status_node=%s,
status_sensing=%s,waktu_node=%s
WHERE_kode_node=%s")
queryInputNode2 = (statusNode,statusSensing,waktuSensing,
getNodeNumber)

cursor.execute(queryNode2,queryInputNode2)

db.commit()
cursor.close()
db.close()

```

- Menampilkan opsi fitur aplikasi (main menu)

Ketika aplikasi admin pertama kali buka, aplikasi akan menampilkan opsi fitur aplikasi yang dapat dipilih oleh Admin. Opsi fitur yang dapat dipilih admin antara lain, check status node, mulai *sensing*, stop *sensing*, dan matikan aplikasi *base station*. Untuk mengeksekusi opsi yang diberikan admin dapat menginput nomor opsi yang ingin dieksekusi pada aplikasi

Kode 5.18: Metode mainMenu()

```

def mainMenu():
    print("=====")
    print("Pilihan_Eksekusi_Program")
    print("1._Check_Status_Node")
    print("2._Mulai_Sensing")
    print("3._Stop_Sensing")
    print("4._Matikan_Aplikasi_Base_Station")
    print("=====")
    print("Masukan_Nomor_Instruksi=")

```

- Melakukan check status node

Aplikasi admin dapat melakukan check status keaktifan node yang terhubung dalam jaringan. Sistem akan mengirimkan perintah pada setiap node untuk mengirimkan status node masing masing dalam jangka waktu tertentu

Kode 5.19: Metode getPingArduino()

```

def getPingArduino(dataNode):
    potongData = dataNode.split(" | ")

    if len(potongData) > 1:
        nodeSensing = potongData[0]
        kelembabanTanah = potongData[1]
        kelembabanUdara = potongData[2]
        suhuTanah = potongData[3]
        suhuUdara = potongData[4]
        phTanah = potongData[5]
        statusNode = potongData[6]
        statusSensing = potongData[7]

    return nodeSensing,statusNode

```

Method counterAdd digunakan untuk menghitung dan membatasi waktu tunggu aplikasi terhadap respon balasan node, setelah perintah check status node dieksekusi

Kode 5.20: Metode counterAdd()

```

def counterAdd():
    global statusNode

```

```

enter="try__":
#jika tidak mendapat respon selama 28 detik artinya node mati
if(counter>28):
    print("Node_Offline")
    print("")
statusNode = False

return enter

```

Method counterSet akan dieksekusi setelah counterAdd selesai dieksekusi, dan melakukan reset terhadap nilai dari atribut counter.

Kode 5.21: Metode counterSet()

```

def counterSet():
    global counter
    counter=0;

```

Method terimaRespon akan menampilkan dialog apabila tidak ada node yang merespon dari perintah yang dikirimkan oleh aplikasi admin (*base station*)

Kode 5.22: Metode terimaRespon()

```

def terimaRespon():
    global statusNode
    output = ""

    if(respon==0): #jika tidak mendapat respon
        output = "Seluruh_Node_Offline"
        statusNode = False

    return output

```

- Tester

Selain fitur melakukan check status node dan mulai *sensing*, Aplikasi admin dapat mengirimkan perintah stop *sensing* pada node yang terhubung dalam jaringan. Fitur ini terdapat di tester dan setelah perintah dieksekusi, maka akan muncul dialog yang menginformasikan bahwa *sensing* telah berhasil dihentikan. Pada tester juga terdapat fitur untuk keluar dari aplikasi admin.

Kode 5.23: Metode tester()

```

#tester
while appRunning:
    finding = False
    while menuShow:
        print("_")
        if(inputNum=="2"):
            print("Sensing_dimulai...")
            while sensingRunning:
                # kirim perintah sensing arduino
                ser.write(str.encode("a"))

                # ambil hasil komunikasi arduino
                decoder=ser.readline().decode("ascii").strip()
                with concurrent.futures.ThreadPoolExecutor() as executor:
                    future = executor.submit(getDataSensing,decoder)
                    #print(future.result())
                    if future.result() != None:
                        future2 =
                            executor.submit(sentDataSensing,
                            future.result())
                        print(future.result())

    elif(inputNum == "1"):
        print("Mengirim_perintah_check_status")
        print("Silakan_tunggu_respon...")
        #28 detik untuk cek seluruh node yang disebar
        while (counter<28):
            # kirim perintah check arduino
            ser.write(str.encode("b").strip())

            # ambil hasil komunikasi arduino
            decoder=ser.readline().decode("ascii").strip()
            with concurrent.futures.ThreadPoolExecutor() as executor:

```

```

        counterAdd()
        counter+=1
        future3 = executor.submit(getPingArduino,decoder)

        if future3.result()!=None:
            print("")
            print("Ping_Node_Diterima")
            print(future3.result())
            future4 =
            executor.submit(sentDataSensing,
            future3.result())
            global statusNode
            statusNode = True
            respon+=1

        if(respon==0):
            print("")
            print("Tidak_ada_Respon")
            print("Silakan_Check_Perangkat..")
            print("")
        else:
            print("")
            print("Check_Node_Selesai")
            print("")
        respon=0
        mainMenu()

    elif(inputNum == "4"):
        # kirim perintah matikan sensing
        ser.write(str.encode("c").strip())
        finding = False

        #Stop App Base Station
        print("Eksekusi_: Matikan_Aplikasi_Base_Station")
        os.system("skripsi laravel_BS_Rev1.py")
        print("====")
        print("Sensing_Dihentikan_!")
        print("Base_Station_Offline")
        exit()

    elif(inputNum == "3"):
        # kirim perintah matikan sensing
        ser.write(str.encode("c").strip())
        finding = False

        # kirim perintah stop sensing
        print("Sensing_Dihentikan_!")
        statusSensing = False
        mainMenu()

    else:
        print("Pilihan_Eksekusi_Salah")
        print("Restart_Aplikasi!")
        exit()

    inputNum=input()

```

Kode program kelas ini secara lengkap dapat dilihat pada lampiran B.

5.2 Pengujian

Subbab pengujian akan terdiri dari hasil pengujian yang dilakukan oleh aplikasi yang dibangun. Terdapat dua pengujian yang dilakukan dalam pembangunan aplikasi yaitu pengujian fungsional dan pengujian eksperimental.

5.2.1 Pengujian Fungsional

Pengujian fungsional dilakukan memastikan aplikasi yang dibangun dapat melakukan fitur-fitur yang telah ditentukan pada tahap perancangan.

Perangkat Arduino

Perangkat node sensor akan melakukan pengamatan terhadap tanah yang diuji menggunakan sensor *sensing* yang terhubung dengan perangkat Arduino. Node sensor akan mengolah data yang dikirim dari sensor *sensing* dari sinyal analog menjadi sinyal digital. Setelah seluruh data telah dikonversi dengan baik, node sensor akan manampilkan data ke monitor (jika terhubung dengan komputer),

atau langsung mengirim data tersebut ke *base station*. Setelah data dikirimkan maka kegiatan *sensing* berikutnya akan terus berlangsung.

```

11:56:02.516 -> Node pengiriman: Node 1
11:56:02.516 -> Kelembaban Tanah: 675.00^-10 %
11:56:04.533 -> Kelembaban Udara: 76.00 %
11:56:04.800 -> Suhu Tanah: 23.81 C
11:56:04.934 -> Suhu Udara: 27.00 C
11:56:04.934 -> Kadar Keasaman Tanah (pH): 3.15
11:56:04.983 ->
11:56:09.753 -> Online|Sensing
11:56:17.054 ->
11:56:17.054 -> Node pengiriman: Node 1
11:56:17.054 -> Kelembaban Tanah: 669.00^-10 %
11:56:19.073 -> Kelembaban Udara: 76.00 %
11:56:19.355 -> Suhu Tanah: 23.75 C
11:56:19.477 -> Suhu Udara: 27.00 C
11:56:19.477 -> Kadar Keasaman Tanah (pH): 3.43
11:56:19.522 ->
11:56:24.307 -> Online|Sensing
11:56:32.430 ->
11:56:32.430 -> Node pengiriman: Node 1
11:56:32.430 -> Kelembaban Tanah: 675.00^-10 %
11:56:34.444 -> Kelembaban Udara: 76.00 %
11:56:34.705 -> Suhu Tanah: 23.81 C
11:56:34.840 -> Suhu Udara: 27.00 C
11:56:34.840 -> Kadar Keasaman Tanah (pH): 3.15
11:56:34.891 ->
11:56:39.678 -> Online|Sensing

```

Gambar 5.1: Arduino menampilkan hasil sensing

Perangkat Raspberry Pi

- Menampilkan opsi pilihan fitur

Saat pertama kali aplikasi admin dijalankan, maka aplikasi akan menampilkan opsi pilihan fitur (*main menu*) yang dapat dipilih dan dieksekusi oleh admin. Pilihan fitur yang ditampilkan pada main menu adalah check status node, mulai *sensing*, stop *sensing*, dan matikan aplikasi *base station*.

```

Python 3.5.3 (default, Jul 9 2020, 13:00:10)
[GCC 6.3.0 20170516] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/projects/skripsiLaravel_BS_Rev2.py =====
Base Station Online
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:

```

```

len(potongData) > 1:
    nodeSensing = potongData[0]
    kelembabanTanah = potongData[1]
    kelembabanUdara = potongData[2]
    suhuTanah = potongData[3]
    suhuUdara = potongData[4]
    phTanah = potongData[5]
    statusNode = potongData[6]
    statusSensing = potongData[7]

    waktuSensing = datetime.datetime.now()
    waktuSensing = waktuSensing.strftime('%Y-%m-%d %H:%M:%S')

    return nodeSensing,kelembabanTanah,kelembabanUdara,suhuTanah,suhuUdara,p

def PingArduino(dataNode):
    potongData = dataNode.split("|")

    len(potongData) > 1:
        nodeSensing = potongData[0]
        kelembabanTanah = potongData[1]
        kelembabanUdara = potongData[2]
        suhuTanah = potongData[3]
        suhuUdara = potongData[4]
        phTanah = potongData[5]
        statusNode = potongData[6]
        statusSensing = potongData[7]

        return nodeSensing,statusNode

    h node yang disebar (thread)
    IZE = 3

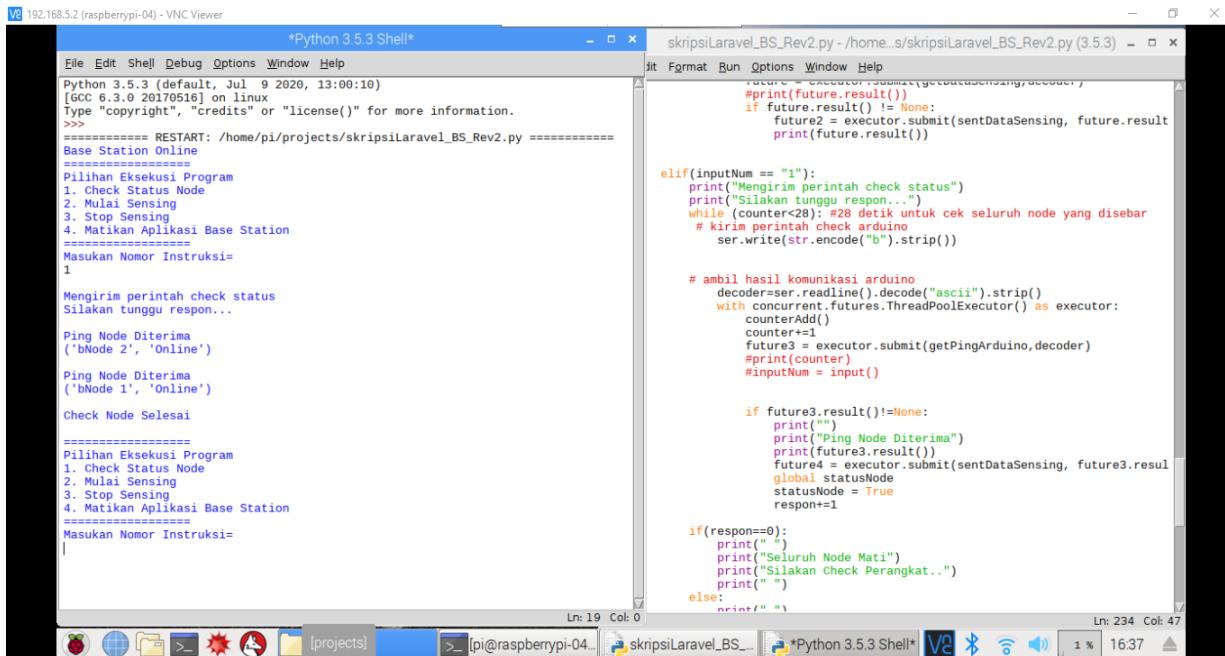
    ntDataSensing(dataSensing):
        mysql.connector.connect(
            host = 'localhost',
            database = 'Skripsi IT'.

```

Gambar 5.2: Raspberry Pi menampilkan opsi fitur

- Mengirim perintah check status node

Aplikasi admin memiliki opsi fitur check status yang berfungsi untuk mengirimkan perintah kepada seluruh node yang terhubung dalam jaringan untuk mengirimkan status nodenya masing-masing.



```

192.168.5.2 (raspberrypi-04) - VNC Viewer
*Python 3.5.3 Shell*
File Edit Shell Debug Options Window Help
Python 3.5.3 (default, Jul 9 2020, 13:00:10)
[GCC 6.3.0 20170516] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/projects/skripsiLaravel_BS_Rev2.py =====
Base Station Online
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
1

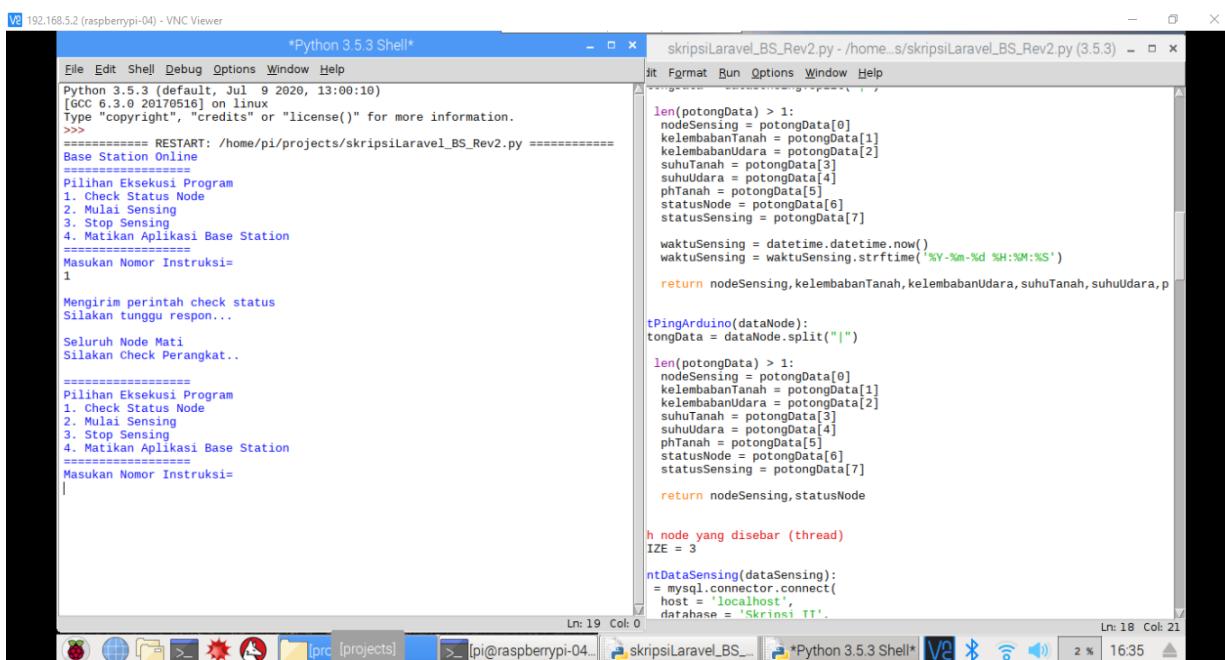
Mengirim perintah check status
Silakan tunggu respon...
Ping Node Diterima
('bNode 2', 'Online')
Ping Node Diterima
('bNode 1', 'Online')
Check Node Selesai
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
|
```

```

Ln: 19 Col: 0
Ln: 234 Col: 47

```

Gambar 5.3: Raspberry Pi menerima respon status node



```

192.168.5.2 (raspberrypi-04) - VNC Viewer
*Python 3.5.3 Shell*
File Edit Shell Debug Options Window Help
Python 3.5.3 (default, Jul 9 2020, 13:00:10)
[GCC 6.3.0 20170516] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/projects/skripsiLaravel_BS_Rev2.py =====
Base Station Online
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
1

Mengirim perintah check status
Silakan tunggu respon...
Seluruh Node Mati
Silakan Check Perangkat...
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
|
```

```

Ln: 19 Col: 0
Ln: 18 Col: 21

```

Gambar 5.4: Raspberry Pi tidak menerima respon status node

- Menerima hasil *sensing* yang dikirimkan oleh node sensor

Perangkat Raspberry Pi yang berperan sebagai *base station* memiliki fitur untuk menerima hasil *sensing* yang dikirimkan oleh node sensor, dan menampilkannya di aplikasi admin.

```

4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
1
Mengirim perintah check status
Silakan tunggu respon...
Ping Node Diterima
('bNode 2', 'Online')
Ping Node Diterima
('bNode 1', 'Online')
Check Node Selesai
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
2
Sensing dimulai...
('Node 2', '17.00', '74.00', '26.06', '27.00', '-2.88', 'Online', 'Sensinga', '2
021-01-08 16:38:11')
('aNode 2', '17.00', '73.00', '26.13', '27.00', '-2.88', 'Online', 'Sensing', '2
021-01-08 16:38:22')
('aNode 1', '19.00', '73.00', '26.00', '27.00', '4.81', 'Online', 'Sensing', '20
21-01-08 16:38:50')
('aNode 2', '17.00', '73.00', '26.13', '27.00', '-2.74', 'Online', 'Sensing', '2
021-01-08 16:38:52')
('aNode 1', '14.00', '73.00', '26.00', '27.00', '4.19', 'Online', 'Sensing', '20
21-01-08 16:38:53')
('aNode 2', '17.00', '73.00', '26.13', '27.00', '-2.81', 'Online', 'Sensing', '2
021-01-08 16:39:21')

```

```

#future = executor.submit(getDataSensing, decoder)
#if future.result() != None:
#    future2 = executor.submit(sentDataSensing, future.result())
#    print(future.result())

#elif(inputNum == "1"):
#    print("Mengirim perintah check status")
#    print("Silakan tunggu respon...")
#    while (counter<28): #28 detik untuk cek seluruh node yang disebar
#        # kirim perintah check arduino
#        ser.write(str.encode("b").strip())

#        # ambil hasil komunikasi arduino
#        decoder=ser.readline().decode("ascii").strip()
#        with concurrent.futures.ThreadPoolExecutor() as executor:
#            counterAdd()
#            counter+=1
#            future3 = executor.submit(getPingArduino,decoder)
#            #print(counter)
#            #inputNum = input()

#            if future3.result()!=None:
#                print("")
#                print("Ping Node Diterima")
#                print(future3.result())
#                future4 = executor.submit(sentDataSensing, future3.result)
#                global statusNode
#                statusNode = True
#                respon+=1

#                if(respon==0):
#                    print(" ")
#                    print("Seluruh Node Mati")
#                    print("Silakan Check Perangkat..")
#                    print(" ")
#                else:
#                    print(" ")

```

Gambar 5.5: Raspberry Pi menerima hasil *sensing*

- Menyimpan hasil *sensing* yang diterima

Setelah data yang dikirimkan oleh node sensor dapat diterima dan ditampilkan di aplikasi admin, Raspberry melakukan fitur menyimpan data. Data yang dikirimkan node sensor akan disimpan ke internet. Dalam pembangunan aplikasi ini, localhost dibangun diperangkat Raspberry Pi untuk menggantikan internet sebagai penyimpanan basis data.

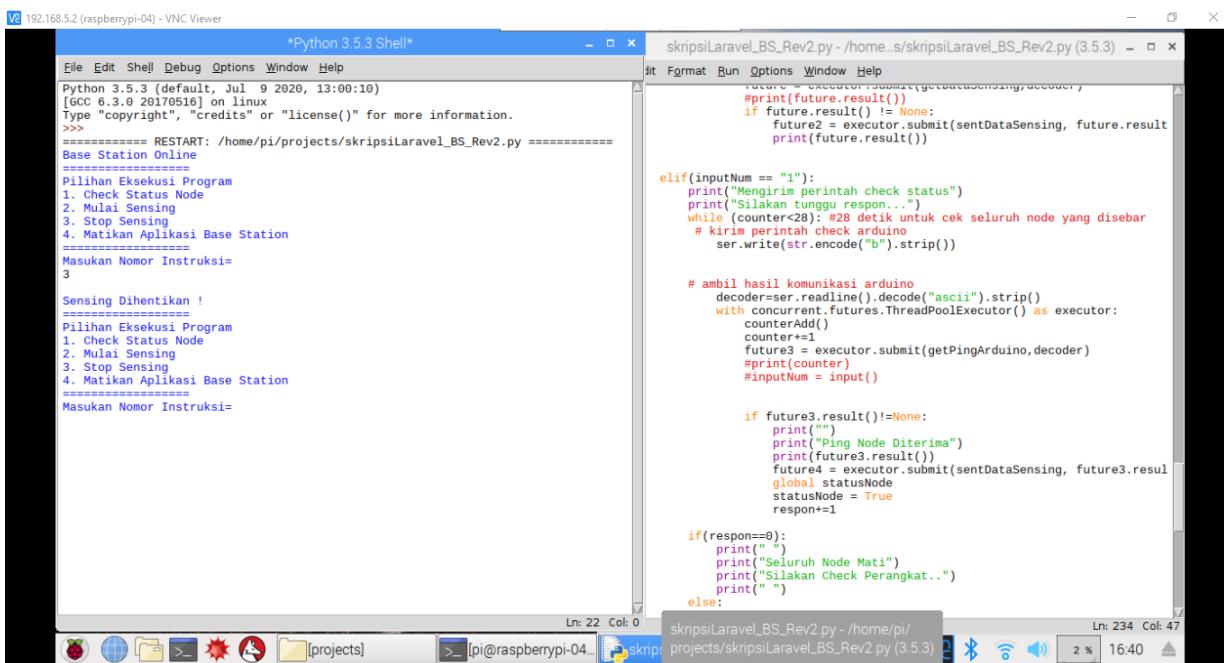
	id_tanah	jenis_tanah	ph_tanah	suhu_tanah	kelembaban_tanah	suhu_udara	kelembaban_udara	kode_petak	waktu
<input type="checkbox"/>	82	irigasi	7.79	27.13	15	27	66	2	2020-01-08 16:38:11
<input type="checkbox"/>	83	irigasi	7.93	27.19	15	27	65	2	2020-01-08 16:38:22
<input type="checkbox"/>	84	irigasi	8.07	27.13	15	27	66	2	2020-01-08 16:38:50
<input type="checkbox"/>	85	irigasi	8.07	27.19	15	27	65	2	2020-01-08 16:38:52
<input type="checkbox"/>	86	irigasi	10.91	26.69	19	27	65	1	2020-01-08 16:39:21

Gambar 5.6: Raspberry Pi menyimpan data *sensing*

- Mengirim perintah stop *sensing*

Base station dapat mengirim perintah stop *sensing* pada node yang terhubung dalam jaringan. Admin hanya perlu memilih opsi fitur stop *sensing* dengan cara melakukan input '3' pada

aplikasi saat main menu ditampilkan. Setelah opsi stop *sensing* dipilih, maka perintah untuk menghentikan *sensing* akan dieksekusi dan akan ditampilkan dialog "Sensing dihentikan!" pada aplikasi admin



```

Python 3.5.3 (default, Jul 9 2020, 13:00:10)
[GCC 6.3.0 20170516] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ====== RESTART: /home/pi/projects/skripsiLaravel_BS_Rev2.py ======
Base Station Online
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
3

Sensing Dihentikan !
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:

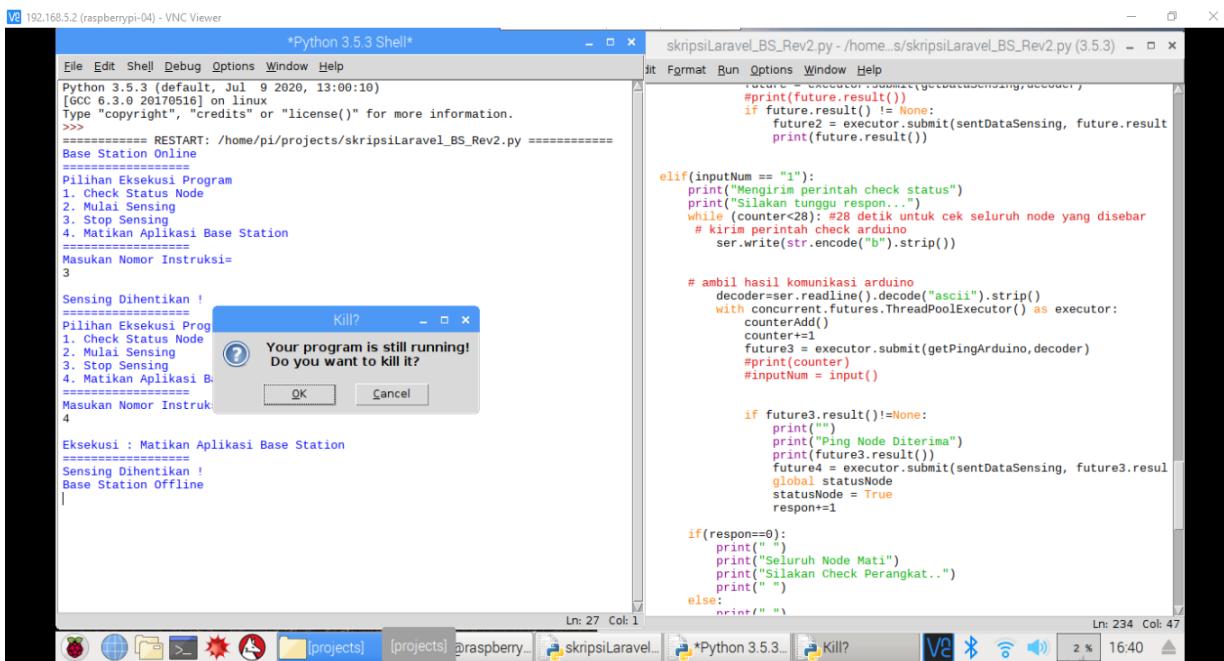
Ln: 22 Col: 0
skripsiLaravel_BS_Rev2.py - /home/pi/projects/skripsiLaravel_BS_Rev2.py (3.5.3) Ln: 234 Col: 47

```

Gambar 5.7: Raspberry Pi mengirim perintah stop *sensing*

- Keluar dari aplikasi

Jika proses pemantauan telah selesai dilakukan maka admin dapat keluar dari aplikasi dengan cara memilih opsi "Matikan Aplikasi Base Station" di main menu. Jika Opsi ini dipilih, secara otomatis *base station* akan mengirimkan perintah stop *sensing* sebelum aplikasi ditutup.



```

Python 3.5.3 (default, Jul 9 2020, 13:00:10)
[GCC 6.3.0 20170516] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ====== RESTART: /home/pi/projects/skripsiLaravel_BS_Rev2.py ======
Base Station Online
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi Base Station
=====
Masukan Nomor Instruksi:
3

Sensing Dihentikan !
=====
Pilihan Eksekusi Program
1. Check Status Node
2. Mulai Sensing
3. Stop Sensing
4. Matikan Aplikasi B
=====
Masukan Nomor Instruk:
4

Eksekusi : Matikan Aplikasi Base Station
=====
Sensing Dihentikan !
Base Station Offline
| Kill? - x
Your program is still running!
Do you want to kill it?
OK Cancel
Ln: 27 Col: 1
skripsiLaravel_BS_Rev2.py - /home/pi/projects/skripsiLaravel_BS_Rev2.py (3.5.3) Ln: 234 Col: 47

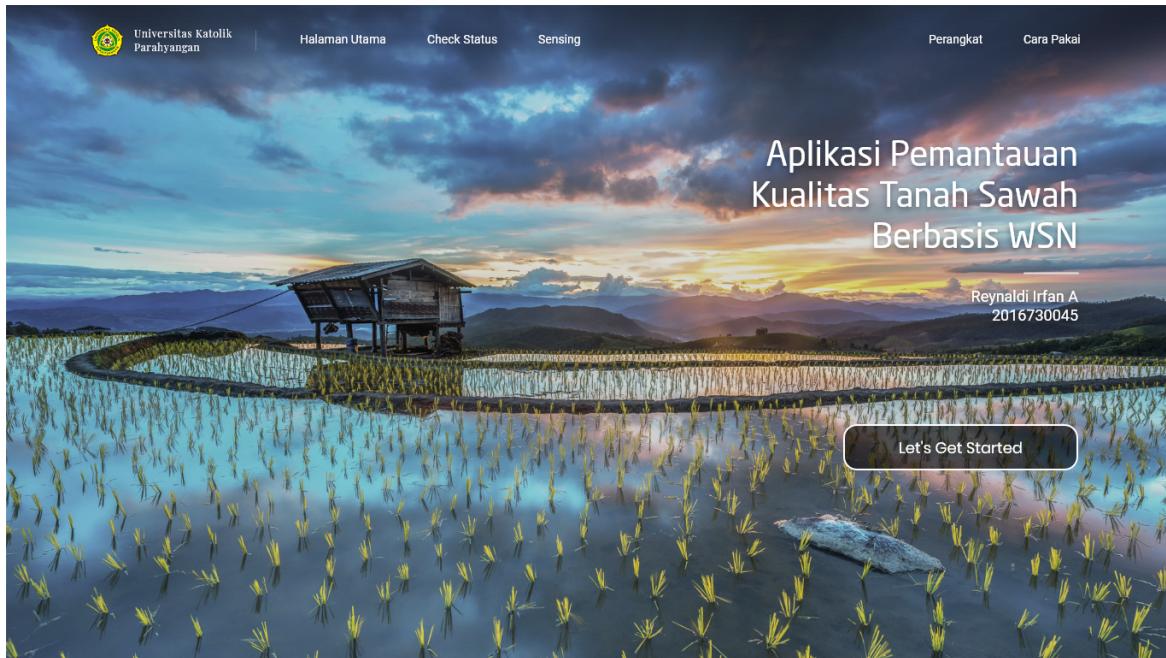
```

Gambar 5.8: Raspberry Pi mengirim perintah keluar dari aplikasi

Website / Aplikasi

- Fitur Check Status

Fitur check status akan menampilkan status aktif node yang terhubung dalam jaringan, juga menampilkan status *sensing* yang dilakukan oleh node tersebut. Fitur check status juga akan menampilkan status *base station*.



Gambar 5.9: Halaman Utama

Variable	Value
Nama Node	Node 1
Status Node	Offline
Status Sensing	Not Sensing

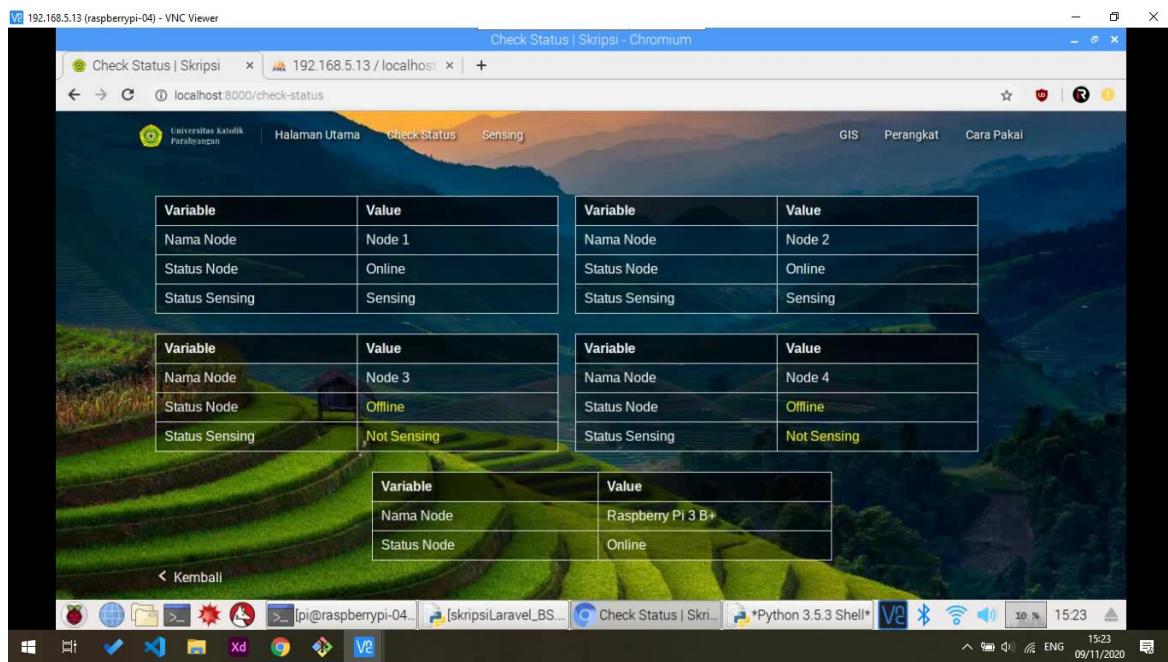
Variable	Value
Nama Node	Node 2
Status Node	Offline
Status Sensing	Not Sensing

Variable	Value
Nama Node	Node 3
Status Node	Offline
Status Sensing	Not Sensing

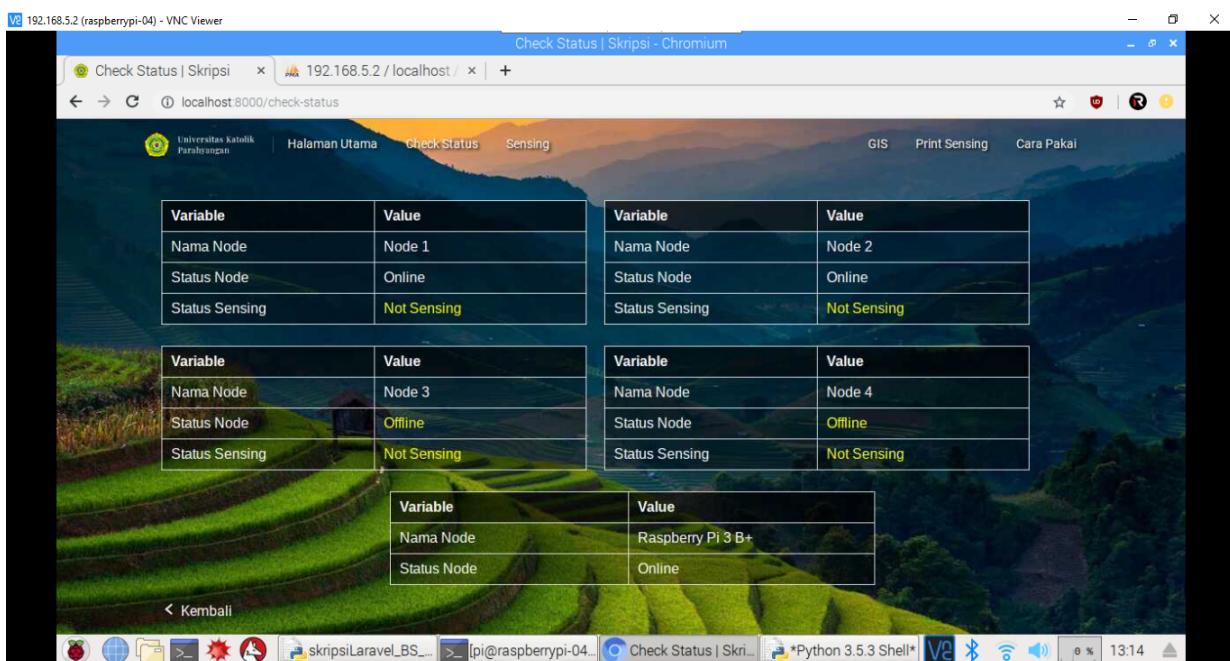
Variable	Value
Nama Node	Node 4
Status Node	Offline
Status Sensing	Not Sensing

Variable	Value
Nama Node	Raspberry Pi 3 B+
Status Node	Online

Gambar 5.10: Halaman Check Status sebelum mendapat pengiriman paket



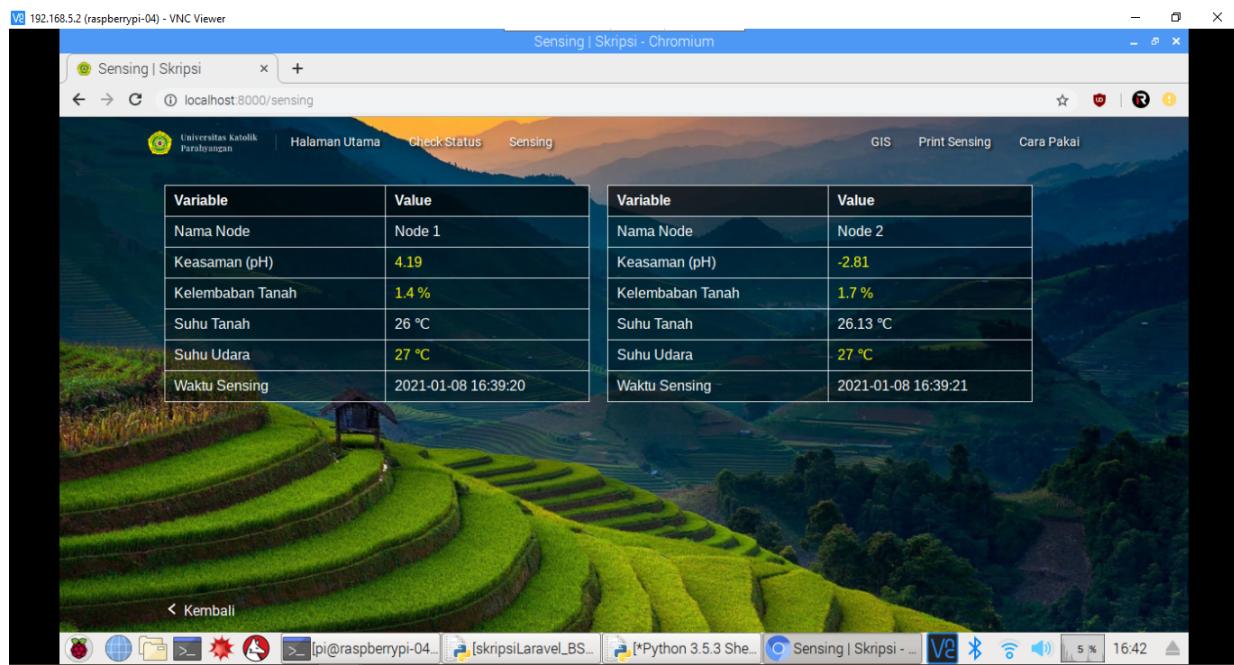
Gambar 5.11: Halaman Check Status setelah mendapat pengiriman paket



Gambar 5.12: Halaman Check Status setelah opsi perintah stop *sensing* dieksekusi

- Fitur Sensing

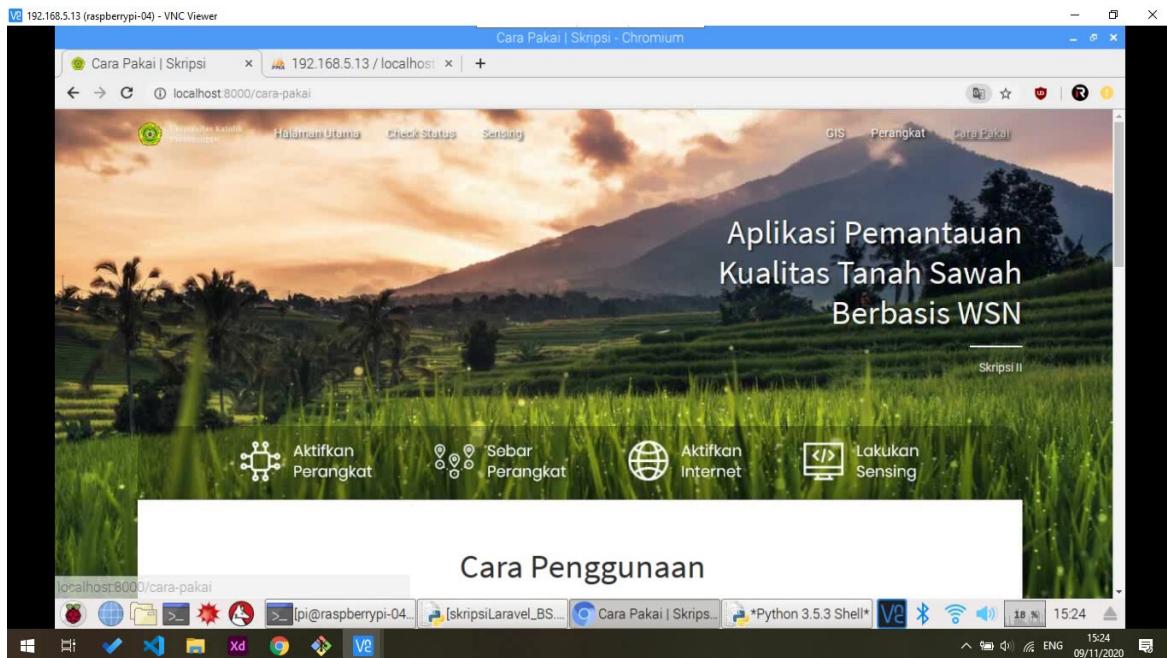
Fitur *sensing* merupakan fitur utama pada aplikasi yang bangun. Fitur ini akan menampilkan data hasil *sensing* yang dilakukan oleh node sensor yang telah terekam dan disimpan di basis data. Tabel pada halaman fitur ini akan diperbarui secara terus menerus dalam tempo waktu tertentu, untuk memperbarui data *sensing* yang diterima oleh node sensor.



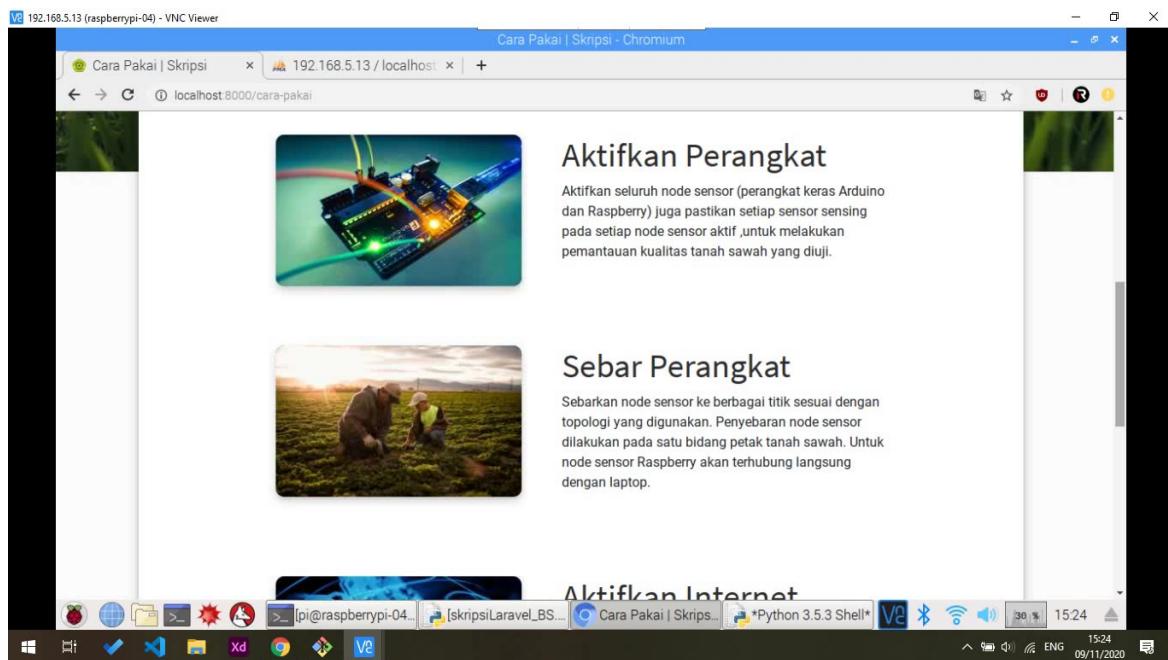
Gambar 5.13: Halaman Sensing setelah mendapat pengiriman paket

- Fitur Cara Penggunaan

Fitur cara penggunaan merupakan fitur tambahan untuk menampilkan proses dan tata-cara untuk melakukan pengamatan kualitas tanah sawah secara sederhana, menggunakan aplikasi yang dibangun. Fitur ini dibangun khusus untuk pengguna, agar pengguna dapat mengetahui cara penggunaan aplikasi yang dibangun.



Gambar 5.14: Halaman Cara Pakai



Gambar 5.15: Halaman Cara Pakai (2)

- Fitur Print Sensing

Fitur print *sensing* memungkinkan pengguna untuk melihat riwayat *sensing* yang dilakukan dan menyimpan riwayat *sensing* tersebut dengan cara melakukan print halaman. Fitur ini juga memungkinkan pengguna untuk melakukan filter terhadap tabel riwayat *sensing* berdasarkan waktu tertentu atau berdasarkan kode petak tertentu.

Kode Petak	Waktu Sensing	Keasaman (pH)	Kelembaban Tanah	Kelembaban Udara	Suhu Tanah	Suhu Udara
2	2021-01-08 16:39:21	-2.81	1.7	73	26.13	27
1	2021-01-08 16:39:20	4.19	1.4	73	26	27
2	2021-01-08 16:38:52	-2.74	1.7	73	26.13	27
1	2021-01-08 16:38:50	4.81	1.9	73	26	27
2	2021-01-08 16:38:22	-2.88	1.7	73	26.13	27
2	2021-01-08 16:38:11	-2.88	1.7	74	26.06	27
2	2021-01-05 16:47	-3.99	1.7	76	25.56	25

Gambar 5.16: Halaman Print Sensing

5.2.2 Pengujian Eksperimental

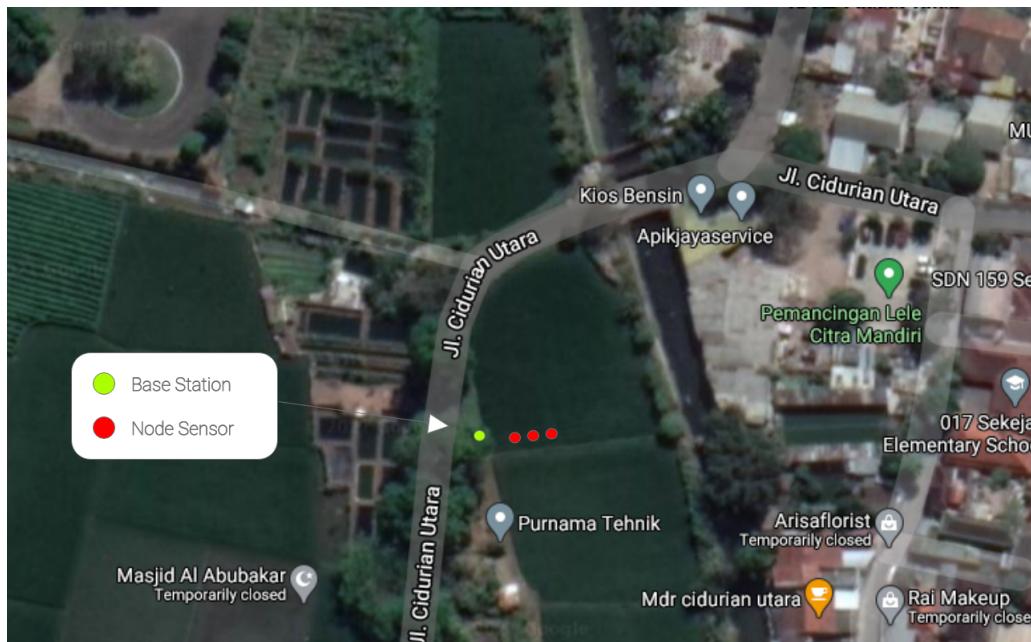
Pengujian eksperimental dilakukan dirumah menggunakan tanah yang ditanam dengan tanaman cabai untuk mengetahui efektifitas perangkat dalam mengambil data dari tanah tersebut. Pengujian eksperimental ini juga menguji pengiriman data hasil *sensing* ke *base station*, untuk disimpan dan ditampilkan ke pengguna.



Gambar 5.17: Pengujian Eksperimental

5.2.3 Pengujian Lapangan

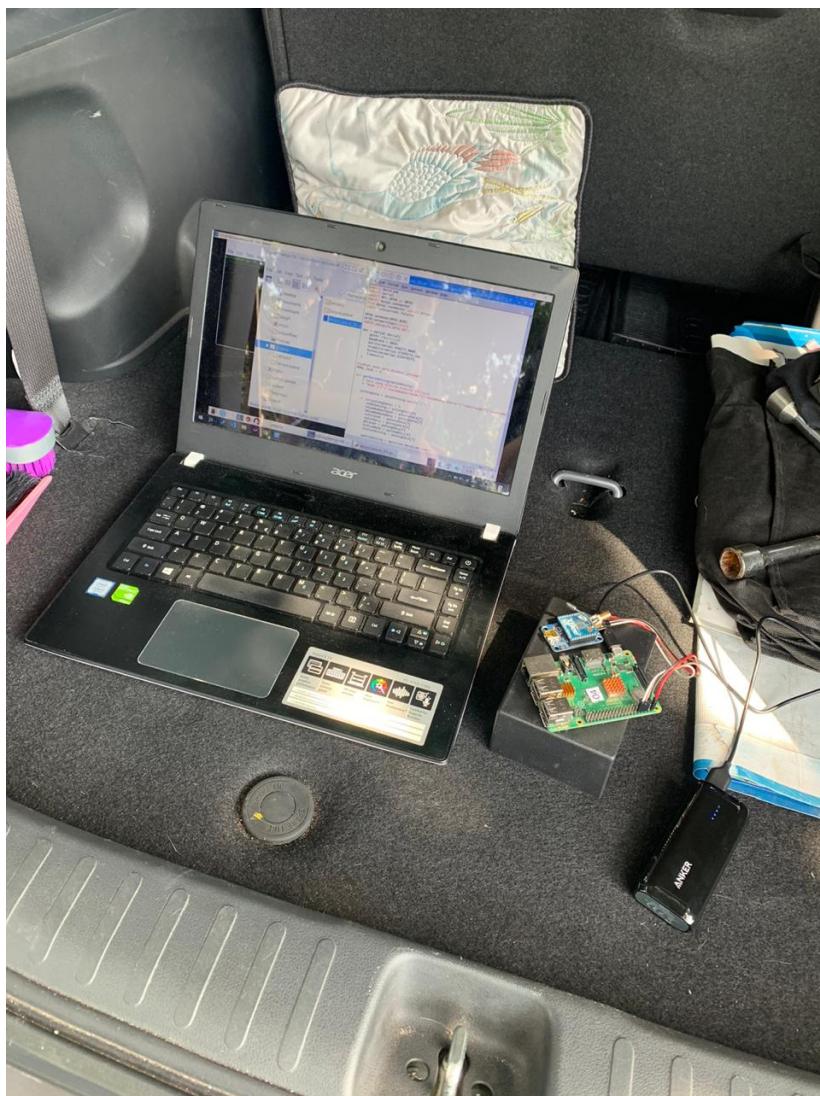
Pengujian lapangan dilakukan ditanah persawahan pindad. Pada pengujian lapangan terdapat dua petak sawah yang diuji dalam satu lokasi yang berdekatan. Petak sawah pertama merupakan petak tanah sawah dengan luas yang tidak terlalu besar. Sedangkan petak tanah sawah kedua merupakan bagian dari persawahan besar milik PT.Pindad Persero. Pada pengujian ini, node disebar di beberapa titik tertentu di area persawahan yang akan dilakukan pemantauan.



Gambar 5.18: Peta Sebaran Node



Gambar 5.19: Pengujian Lapangan - Node Disebar



Gambar 5.20: Pengujian Lapangan - Base Station

5.2.4 Skala Penilaian Kualitas Tanah Hasil Pengujian

Untuk mengetahui kualitas tanah sawah yang diuji memiliki kategori baik atau kurang baik. Dapat dilakukan penilaian yang terdiri atas variabel-variabel, yang mempengaruhi kualitas tanah sawah. Variabel-Variabel tersebut antarlain, pH tanah, suhu tanah, kelembaban tanah, dan suhu udara persawahan. Masing-masing variabel memiliki nilai 25, sehingga jika tanah dalam keadaan sangat baik (seluruh variable ideal) akan mendapatkan nilai 100. Nilai 75 untuk tanah dalam keadaan baik (satu variable tidak ideal). Nilai 50 untuk tanah berkualitas cukup baik (dua nilai variable tidak ideal), Nilai 25 untuk tanah berkualitas kurang baik, dan nilai 0 untuk tanah berkualitas buruk.

Dari hasil pengujian lapangan yang dilakukan di persawahan milik PT.Pindad Persero, didapatkan nilai hasil pemantauan sebagai berikut.

- Pemantauan pada Siang Hari

Berikut adalah beberapa nilai dari tabel pemantauan yang dilakukan di persawahan PT.Pindad Persero pada siang hari, dengan kondisi cuaca cukup cerah.

Tabel 5.1: Tabel Pemantauan di Siang Hari

Kode Node	Waktu	pH Tanah	Suhu Tanah	Kel.Tanah	Suhu Udara
3	2021-01-13 13:13:59	5.43	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:14:05	4.39	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:14:09	3.69	26.56 °C	69 %	33 °C
3	2021-01-13 13:14:29	5.43	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:14:35	4.46	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:14:39	3.97	26.56 °C	68.6 %	33 °C
3	2021-01-13 13:14:59	5.43	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:15:04	4.46	26.56 °C	78.5 %	29 °C
1	2021-01-13 13:15:09	3.76	26.63 °C	68.8 %	33 °C
3	2021-01-13 13:15:29	5.43	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:15:34	4.46	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:15:38	3.83	26.56 °C	68.8 %	33 °C
3	2021-01-13 13:15:59	5.43	26.38 °C	77.1 %	29 °C
2	2021-01-13 13:16:04	4.39	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:16:08	3.9	26.56 °C	68.8 %	34 °C
3	2021-01-13 13:16:29	5.43	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:16:34	4.39	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:16:38	3.83	26.56 °C	68.7 %	33 °C
3	2021-01-13 13:16:59	5.49	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:17:03	4.39	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:17:08	3.9	26.56 °C	68.6 %	33 °C
3	2021-01-13 13:17:28	5.49	26.31 °C	77 %	29 °C
2	2021-01-13 13:17:33	4.46	26.44 °C	78.5 %	29 °C
1	2021-01-13 13:17:38	3.76	26.56 °C	68.7 %	34 °C
3	2021-01-13 13:17:58	5.49	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:18:03	4.46	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:18:08	3.9	26.56 °C	68.5 %	33 °C
3	2021-01-13 13:18:28	5.49	26.31 °C	77.1 %	29 °C
2	2021-01-13 13:18:32	4.46	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:18:38	3.83	26.5 °C	68.6 %	33 °C
3	2021-01-13 13:18:58	5.49	26.25 °C	77.1 %	29 °C
2	2021-01-13 13:19:02	4.39	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:19:08	3.83	26.56 °C	67.8 %	34 °C
3	2021-01-13 13:19:28	5.49	26.31 °C	77 %	29 °C
2	2021-01-13 13:19:32	4.46	26.5 °C	78.5 %	29 °C
1	2021-01-13 13:19:38	4.11	26.56 °C	68.1 %	34 °C
3	2021-01-13 13:19:58	5.49	26.31 °C	77 %	29 °C
2	2021-01-13 13:20:01	4.46	26.5 °C	78.4 %	29 °C
1	2021-01-13 13:20:08	3.9	26.56 °C	68.5 %	34 °C
1	2021-01-13 13:20:38	3.83	26.56 °C	68.5 %	34 °C
3	2021-01-13 13:20:58	5.49	26.31 °C	77.1 %	29 °C
1	2021-01-13 13:21:08	3.83	26.56 °C	68.5 %	34 °C
3	2021-01-13 13:21:28	5.49	26.25 °C	77 %	29 °C
2	2021-01-13 13:21:31	4.46	26.44 °C	78.4 %	29 °C
1	2021-01-13 13:21:38	4.25	26.56 °C	68.5 %	33 °C
3	2021-01-13 13:21:57	5.49	26.25 °C	77 %	29 °C
2	2021-01-13 13:22:00	4.46	26.44 °C	78.4 %	29 °C
1	2021-01-13 13:22:07	3.9	26.56 °C	68.4 %	34 °C
3	2021-01-13 13:22:27	5.49	26.25 °C	77 %	29 °C
2	2021-01-13 13:22:30	4.46	26.44 °C	78.4 %	29 °C

Berdasarkan tabel 5.1 didapatkan data rata-rata nilai pH tanah untuk kode node satu adalah 3.8, suhu tanah adalah 26.5°C, kelembaban tanah adalah 68.63%, dan suhu udara adalah 33°C. Sedangkan untuk kode petak dua rata-rata nilai untuk pH tanah adalah 4.43, suhu tanah adalah 26.5°C, kelembaban tanah adalah 78.5%, dan suhu udara adalah 29°C. Terakhir kode petak tiga dengan rata-rata nilai untuk pH tanah adalah 5.43, suhu tanah adalah 26.31 °C, kelembaban tanah adalah 71.1%, dan suhu udara adalah 29°C.

- Pemantauan pada Pagi Hari

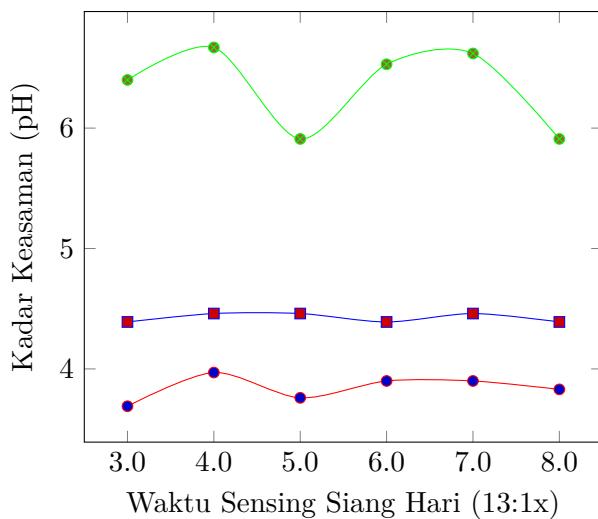
Berikut adalah beberapa nilai dari tabel pemantauan yang dilakukan di persawahan PT.Pindad Persero pada pagi hari, dengan kondisi cuaca cukup mendung. Pemantauan yang dilakukan di pagi hari, dilakukan setelah hujan reda beberapa saat sebelum pemantauan dilakukan.

Tabel 5.2: Tabel Pemantauan di Pagi Hari

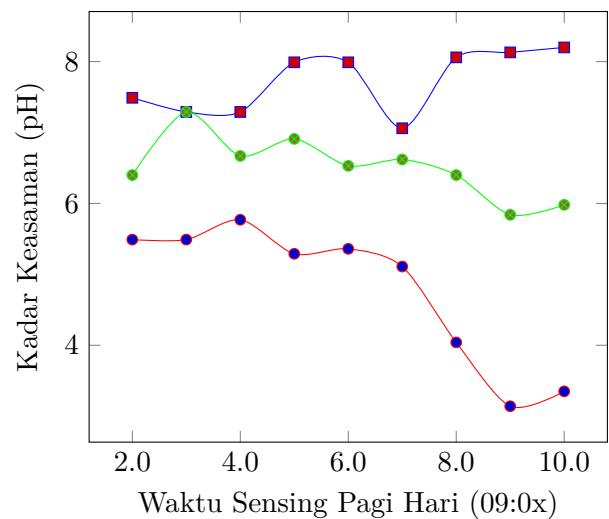
Kode Node	Waktu	pH Tanah	Suhu Tanah	Kel.Tanah	Suhu Udara
1	2021-01-14 09:02:53	5.49	23.63 °C	72.9 %	20 °C
3	2021-01-14 09:03:06	6.4	23.19 °C	77.6 %	20 °C
2	2021-01-14 09:03:18	7.92	23.75 °C	73.3 %	21 °C
1	2021-01-14 09:03:23	5.49	23.63 °C	72.8 %	20 °C
3	2021-01-14 09:03:36	6.67	23.19 °C	77.5 %	21 °C
2	2021-01-14 09:03:47	7.92	23.69 °C	73.3 %	22 °C
1	2021-01-14 09:03:53	5.7	23.63 °C	72.7 %	21 °C
3	2021-01-14 09:04:06	6.26	23.19 °C	77.5 %	22 °C
2	2021-01-14 09:04:17	7.92	23.69 °C	73.3 %	23 °C
1	2021-01-14 09:04:23	5.77	23.56 °C	72.8 %	21 °C
3	2021-01-14 09:04:36	6.26	23.19 °C	77.5 %	22 °C
2	2021-01-14 09:04:47	7.99	23.75 °C	73.3 %	23 °C
1	2021-01-14 09:04:52	5.43	23.69 °C	72.7 %	22 °C
3	2021-01-14 09:05:06	8.41	23.25 °C	77.5 %	23 °C
2	2021-01-14 09:05:16	7.99	23.69 °C	73.3 %	24 °C
1	2021-01-14 09:05:22	5.91	23.63 °C	72 %	23 °C
3	2021-01-14 09:05:36	5.91	23.25 °C	77.5 %	24 °C
2	2021-01-14 09:05:46	7.99	23.75 °C	73.3 %	25 °C
1	2021-01-14 09:05:52	5.29	23.69 °C	72.8 %	24 °C
3	2021-01-14 09:06:05	6.33	23.25 °C	77.5 %	25 °C
2	2021-01-14 09:06:16	7.99	23.75 °C	73.3 %	24 °C
1	2021-01-14 09:06:22	5.22	23.69 °C	72.9 %	24 °C
3	2021-01-14 09:06:35	6.53	23.25 °C	77.5 %	25 °C
2	2021-01-14 09:06:45	7.99	23.75 °C	73.3 %	24 °C
1	2021-01-14 09:06:52	5.36	23.69 °C	72.8 %	24 °C
3	2021-01-14 09:07:05	6.74	23.25 °C	77.5 %	25 °C
2	2021-01-14 09:07:15	7.99	23.81 °C	73.3 %	24 °C
1	2021-01-14 09:07:22	5.32	23.63 °C	72.3 %	24 °C
3	2021-01-14 09:07:35	6.62	23.25 °C	77.5 %	25 °C
2	2021-01-14 09:07:45	7.06	23.75 °C	73.3 %	24 °C
1	2021-01-14 09:07:52	5.11	23.69 °C	71.8 %	24 °C
3	2021-01-14 09:08:05	6.4	23.25 °C	77.5 %	25 °C
2	2021-01-14 09:08:15	8.06	23.81 °C	73.3 %	24 °C
1	2021-01-14 09:08:22	3.55	23.69 °C	70.8 %	24 °C
3	2021-01-14 09:08:35	6.4	23.25 °C	77.5 %	25 °C
2	2021-01-14 09:08:44	8.06	23.81 °C	73.3 %	24 °C
1	2021-01-14 09:08:52	4.04	23.69 °C	71.7 %	24 °C
3	2021-01-14 09:09:05	5.91	23.31 °C	77.5 %	25 °C
2	2021-01-14 09:09:14	8.06	23.88 °C	73.3 %	24 °C
1	2021-01-14 09:09:22	2.93	23.69 °C	70.5 %	24 °C
3	2021-01-14 09:09:35	6.19	23.31 °C	77.5 %	25 °C
2	2021-01-14 09:09:44	8.13	23.81 °C	73.2 %	25 °C
1	2021-01-14 09:09:52	3.14	23.69 °C	69.8 %	24 °C
3	2021-01-14 09:10:05	5.84	23.31 °C	77.5 %	24 °C
2	2021-01-14 09:10:13	8.13	23.81 °C	73.2 %	25 °C
1	2021-01-14 09:10:22	3	23.75 °C	70.7 %	24 °C
3	2021-01-14 09:10:34	5.98	23.31 °C	77.5 %	24 °C
2	2021-01-14 09:10:43	8.2	23.81 °C	73.2 %	25 °C
1	2021-01-14 09:10:52	3.35	23.75 °C	71.1 %	24 °C

Dari tabel 5.2 Didapatkan data rata-rata nilai untuk pH tanah untuk kode node satu adalah 5.26, suhu tanah adalah 23.67°C, kelembaban tanah adalah 72,3%, dan suhu udara adalah 24°C. Sedangkan untuk kode petak dua rata-rata nilai untuk pH tanah adalah 7.68, suhu tanah adalah 23.75°C, kelembaban tanah adalah 72,3%, dan suhu udara adalah 24°C. Terakhir kode petak tiga dengan rata-rata nilai untuk pH tanah adalah 6.63, suhu tanah adalah 23.25°C, kelembaban tanah adalah 77.5%, dan suhu udara adalah 25°C.

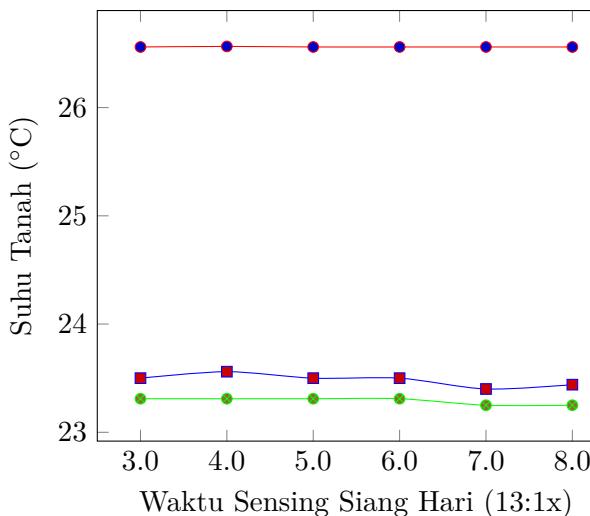
Dari kedua tabel diatas didapatkan informasi, bahwa waktu pemantauan berpengaruh terhadap suhu udara, yang merupakan variabel yang mempengaruhi kualitas tanah sawah. Juga didapatkan kesimpulan bahwa air hujan mempengaruhi kualitas tanah sawah dengan meningkatkan nilai keasaman (pH) Tanah, suhu udara, dan kelembaban tanah. Dari kedua tabel diatas juga dapat disimpulkan bahwa tanah persawahan PT.Pindad persero memiliki kualitas tanah sawah dengan katergori cukup baik dengan nilai 50. Nilai 50 dihasilkan dari skala penilaian kualitas tanah sawah. Berdasarkan hasil pemantauan, diketahui bahwa tanah milik PT.Pindad Persero memiliki dua variable yang memenuhi kategori ideal yaitu suhu udara dan suhu tanah, dan dua variable lain yang tidak memenuhi kategori ideal yaitu pH tanah (kadar keasaman tanah) dan kelembaban tanah.



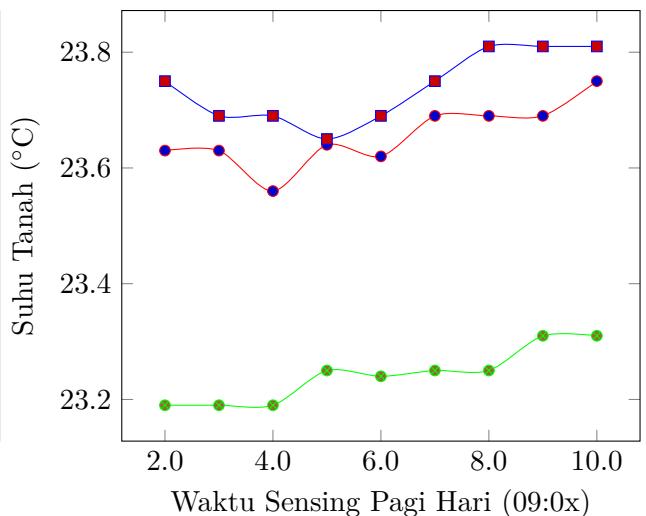
Gambar 5.21: Kadar Ph Tanah Siang Hari



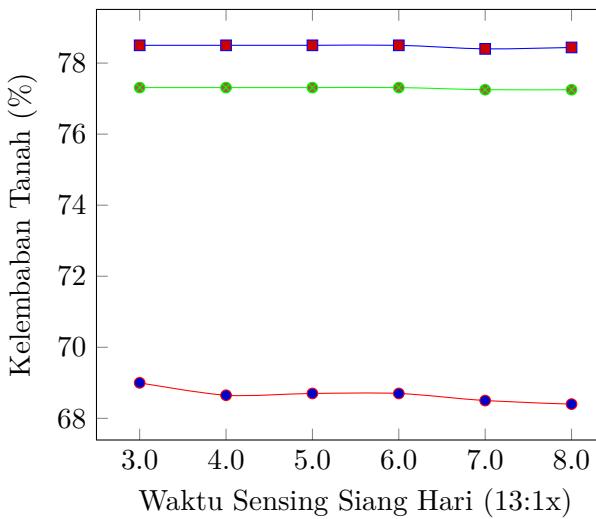
Gambar 5.22: Kadar Ph Tanah Pagi Hari



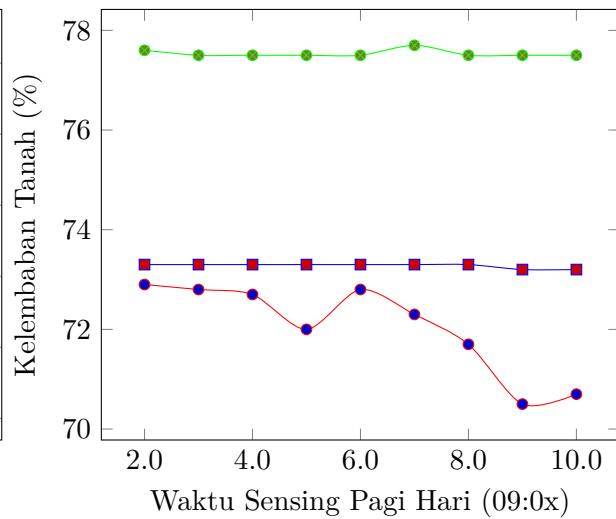
Gambar 5.23: Suhu Tanah di Siang Hari



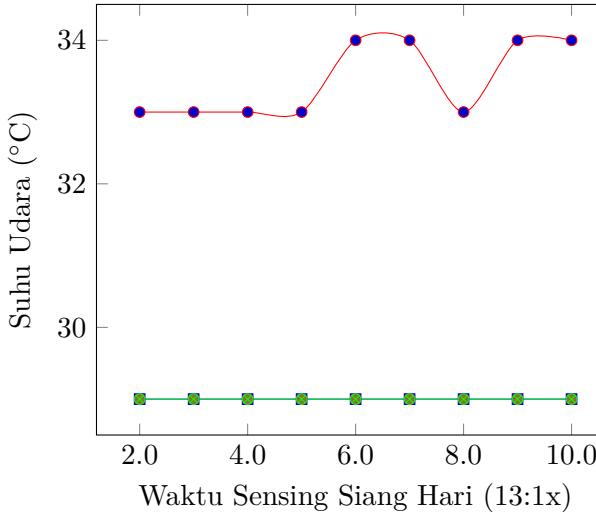
Gambar 5.24: Suhu Tanah di Pagi Hari



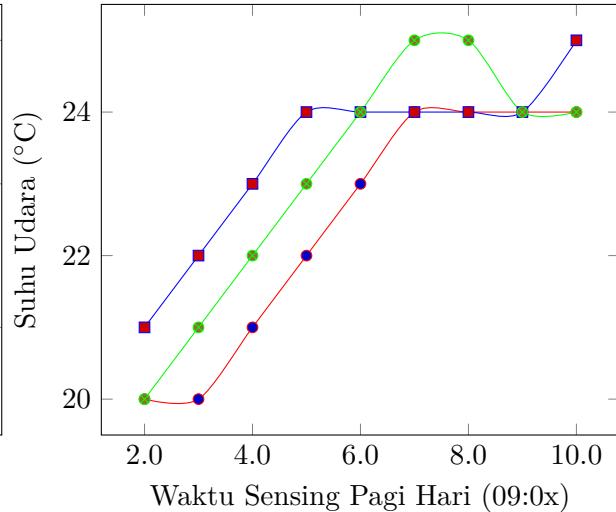
Gambar 5.25: Kelembaban Tanah di Siang Hari



Gambar 5.26: Kelembaban Tanah di Pagi Hari



Gambar 5.27: Suhu Udara di Siang Hari



Gambar 5.28: Suhu Udara di Pagi Hari

Grafik diatas merupakan grafik hasil pemantauan yang memiliki value yang sama dengan tabel 5.1 dan tabel 5.2. Grafik berwarna merah merepresentasikan kode node satu, sedangkan grafik berwarna biru dan hijau merepresentasikan kode node dua dan tiga. Dari tabel dan grafik diatas, yang merupakan hasil pemantauan yang dilakukan selama dua hari di persawahan milik PT.Pindad persero juga menghasilkan informasi bahwa hujan mempengaruhi kadar keasaman tanah sawah. Hal ini dapat dibuktikan dengan drastisnya nilai pH tanah yang berada di tabel 5.1 yang merupakan data tanah sebelum hujan dengan tabel 5.2 yang merupakan data tanah setelah hujan. Selain itu waktu dilakukannya pemantauan juga mempengaruhi nilai suhu ideal dari tanah yang dilakukan pemantauan.

5.3 Kendala

Pada pembangunan suatu aplikasi, umumnya akan ditemukan beberapa kendala, baik dari segi pemrograman maupun dari segi perangkat keras yang digunakan. Begitupula pada pembangunan aplikasi pemantauan kualitas tanah sawah yang dibangun.

5.3.1 Kendala Pemrograman

Berikut adalah kendala atau masalah yang dihadapi selama perancangan dan pembangunan aplikasi dari segi pemrograman.

- Pemrograman Node Sensor

Kendala yang dihadapi pada tahap pemrograman adalah bug yang menyebabkan sensor hanya dapat melakukan *sensing* dalam jangka waktu tertentu lalu terhenti. Namun kendala ini telah dituntaskan dengan cara memastikan hasil *sensing* telah dikirim sebelum melakukan *sensing* kembali. Selain itu juga library yang dimiliki oleh Arduino bersifat *fixed* sehingga sulit mengetahui apakah error terjadi di perangkat keras atau kode program.

- Pemrograman *Base station*

Kendala lain yang dihadapi pada pemrograman pada *base station* adalah banyak mendapatkan kendala untuk interfacing perangkat keras, yang akan dibahas di subbab 5.3.2.

5.3.2 Kendala Perangkat Keras

Berikut adalah kendala atau masalah yang dihadapi selama perancangan dan pembangunan aplikasi dari segi perangkat keras.

- Laptop

Ada beberapa kendala yang dihadapi pada perangkat keras laptop, diantaranya adalah kendala dalam melakukan instalasi aplikasi putty dan VCN untuk melakukan *interfacing* dengan perangkat Raspberry Pi berdasarkan jaringan yang terhubung. Selain itu, kendala juga terjadi dalam mencari ip Raspberry Pi yang terhubung dalam jaringan.

- Raspberry Pi 3 B+

Kendala yang dihadapi pada perangkat Raspberry Pi 3 B+ adalah komunikasi antara Raspberry Pi dengan laptop yang akan menjadi monitor menggunakan perangkat lunak VNC (untuk GUI) atau putty (untuk CLI). Selain itu juga mengakses izin port-port agar perangkat xbee dapat berkomunikasi, serta mengupdate versi php dan menginstall laravel pada perangkat Raspberry Pi yang lebih rumit dibandingkan perangkat laptop.

- Node Sensor

Hampir tidak ada kendala yang didapat dari node sensor, namun perbedaan kecepatan pengiriman paket hasil *sensing* terjadi antara Arduino *original* dengan yang *non-original*.

- Sensor *Sensing*

Kendala pada sensor *sensing* adalah adanya beberapa sensor yang memerlukan kegiatan khusus seperti merakit atau menggunakan las untuk menghubungkan kabel.

BAB 6

KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil penelitian yang dilakukan, diperoleh kesimpulan - kesimpulan sebagai berikut:

1. Aplikasi pemantauan kualitas tanah sawah berbasis *Wireless Sensor Network* telah berhasil dibangun baik dari sisi aplikasi admin maupun aplikasi pengguna.
2. Berdasarkan pengujian, aplikasi yang dibangun berhasil melakukan *sensing* terhadap tanah sawah yang diuji, juga berhasil dalam mengirimkan data tersebut ke *base station* dengan jumlah sensor node sebanyak dua. Hasil pengujian juga menunjukkan bahwa *base station* telah berhasil menerima data hasil *sensing* yang dikirimkan, serta menyimpan data tersebut dan menampilkannya ke *browser*.
3. Hasil pengujian juga memperlihatkan bahwa aplikasi admin di *base station* telah berhasil mengirim beberapa opsi instruksi ke node sensor dan mendapatkan respon.
4. Hasil pengujian lapangan juga menunjukkan bahwa aplikasi yang dibangun mampu memperkirakan kualitas tanah sawah yang dilakukan pengamatan

6.2 Saran

Berdasarkan hasil penelitian yang dilakukan, ada beberapa saran untuk pengembangan aplikasi sebagai berikut:

1. Aplikasi ini menyimpan data hasil *sensing* di basis data yang bersifat *local*. Akan lebih baik jika hasil *sensing* disimpan di internet (*cloud*). Dengan menyimpan data di *cloud*, akses data hasil *sensing* dapat ditampilkan diberbagai jenis perangkat yang terhubung dengan internet.
2. Perangkat keras dari aplikasi yang dibangun disebar menggunakan rangkaian kayu sederhan untuk meletakan node. Akan lebih baik jika rangkaian kayu dapat lebih tinggi dan kabel sensor *sensing* diperpanjang sampai mencapai tanah, sehingga proses penyebaran antar node dapat saling berjauhan tanpa adanya kendala sinyal yang terhalang oleh tanaman padi dalam proses pengiriman hasil sensing.
3. Pemantauan kualitas tanah sawah yang dilakukan di skripsi ini dilakukan pada waktu siang hari dan pagi hari. Proses pemantauan kedepannya dapat dilakukan di waktu yang lebih variatif dengan kondisi cuaca yang berbeda-beda, seperti pada waktu subuh, sore, atau malam hari.

DAFTAR REFERENSI

- [1] Pujiarti (2018) Implementasi real time *Monitoring* lahan pertanian pada tanaman padi menggunakan *Smart Sensor*. Skripsi. Universitas Sumatera Utara, Indonesia.
- [2] Padmawati, N. L. A., Arthagama, I. D. M., dan Susila, K. D. (2014) Evaluasi kualitas tanah di lahan sawah simantri dan non simantri di subak riang desa riang gede, kecamatan penebel. Technical report. Universitas Udayana, Jl. PB. Sudirman Denpasar 80232 Bali.
- [3] Wahyudianto, D. K., Rahmat, G. S., Masrur, M., Permana, R. I., dan Abidin, L. (2013) Perancangan alat bantu indikator kualitas tanah dengan parameter resistivitas tanah dan ph tanah untuk tanaman padi. Technical report. Institut Teknologi Sepuluh Nopember Surabaya, Indonesia.
- [4] Arif, C., Setiawan, B. I., dan Mizoguchi, M. (2014) Penentuan kelembaban tanah optimum untuk budidaya padi sawah sri (system of rice intensification) menggunakan algoritma genetika. Technical Report JI2014-Vol9,No1. IPB Bogor, Indonesia.
- [5] Kristanto, I. (2020) Pengembangan aplikasi ekstraksi fitur domain waktu/frekuensi untuk data akeselerometer di wsn. Skripsi. Universitas Katolik Parahyangan, Indonesia.
- [6] Sabiq, A., Nurmaya, dan Alfarisi, T. (2017) Sistem wireless sensor network berbasis arduino uno dan raspberry pi untuk pemantauan kualitas udara di cempaka putih timur, jakarta pusat. *Raspberry*, **63**, 301–305.

LAMPIRAN A

KODE PROGRAM NODE SENSOR

Kode A.1: Node.ino

```
1 /* ===== library ===== */
2 #include <OneWire.h>
3 #include <DallasTemperature.h>
4 #include <DHT.h>
5 #include <SPI.h>
6 #include <SoftwareSerial.h>
7
8 /* ===== WDT / Transmisi ===== */
9 #define rxPin 18
10 #define txPin 19
11 SoftwareSerial xbee = SoftwareSerial(rxPin, txPin);
12
13 /* ===== suhu & kelembaban tanah ===== */
14 /* suhu tanah */
15
16 // sensor suhu tanah diletakkan di pin 4
17 int temp_sensor = 4;
18
19 OneWire oneWire(temp_sensor);
20 DallasTemperature sensorSuhuTanah(&oneWire);
21
22 /* kelembabab tanah */
23
24 // pin sensor
25 int sensorPin = A0;
26
27 // untuk penganti VCC
28 int powerPin = 2;
29
30
31 /* sensor pH tanah */
32 #define analogInPin A1
33
34 //variable
35 int sensorValue = 0;           //ADC value from sensor
36 float phValue = 0.0;          //pH value after conversion
37
38
39
40 /* ===== suhu & kelembaban udara ===== */
41 DHT dht(3, DHT11); //Pin, Jenis DHT
42
43
44 /* ===== Keterangan Node ===== */
45 String node1 = "Node_1";
46 String node2 = "Node_2";
47 String node3 = "Node_3";
48
49
50 void setup(){
51   Serial.begin(9600);
52
53   dht.begin();
54
55   // jadikan pin power sebagai output
56   pinMode(powerPin, OUTPUT);
57
58   // initialize digital pin LED_BUILTIN as an output.
59   pinMode(LED_BUILTIN, OUTPUT);
60
61   // default bernilai LOW
62   digitalWrite(powerPin, LOW);
63
64   // mulai komunikasi serial
65   xbee.begin(9600);
66 }
67
68 void loop(){
69   Serial.println("");
70   Serial.print("Node_pengiriman:_");
71   Serial.println(node3);
72   Serial.print("Kelembaban_Tanah:_");
73   Serial.print(bacaSensorKelembabanTanah());
74   Serial.println("%");
75   Serial.print("Kelembaban_Udara:_");
```

```

76|     Serial.print(bacaSensorKelembabanUdara());
77|     Serial.println(".");
78|     Serial.print("Suhu_Tanah:_");
79|     Serial.print(bacaSensorSuhuTanah());
80|     Serial.println(".");
81|     Serial.print("Suhu_Udara:_");
82|     Serial.print(bacaSensorSuhuUdara() );
83|     Serial.println(".");
84|     Serial.print("Kadar_Keasaman_Tanah_(pH):_");
85|     Serial.println(bacaSensorPhTanah());
86|     Serial.println(" ");
87|     Serial.println(WrapperStatusNodeDanSensing());
88|
89/* ===== Perintah dari Base Station===== */
90
91 if (xbee.available()) { //ada input perintah dari base station
92     boolean adaPerintah = true;
93     char temp= xbee.read();
94     if(temp=='a'){ //mulai sensing
95         String pesan = conversiDataFloatToString();
96         transmisiPengiriman(pesan);
97     }
98     else if(temp == 'b'){ //check status
99         String pesan = WrapperStatusNodeDanSensing();
100        transmisiPengiriman(pesan);
101    }
102    else if(temp == 'c'){ //stop sensing
103        adaPerintah = false;
104        while(xbee.available()){ //cek perintah base station
105            if(adaPerintah == false){ //selama ga ada perintah >> kunci loop
106                delay(5000);
107            }
108            else{
109                break; //keluar kunci loop dan balik ke void loop
110            }
111        }
112    }
113}
114}
115}
116}
117}
118}
119/* ===== Koding ===== */
120
121 float bacaSensorKelembabanTanah() {
122     // hidupkan power
123     digitalWrite(powerPin, HIGH);
124     delay(2000);
125
126     // baca nilai analog dari sensor
127     int nilaiSensor = analogRead(sensorPin);
128     digitalWrite(powerPin, LOW);
129
130     // makin lembab maka makin tinggi nilai outputnya
131     // konversi sensor kelembaban tanah (analog ke digital)
132     return (1023 - nilaiSensor);
133 }
134
135 float bacaSensorSuhuUdara(){
136     float suhu = dht.readTemperature();
137     return suhu;
138 }
139
140 float bacaSensorKelembabanUdara(){
141     float kelembaban = dht.readHumidity();
142     return kelembaban;
143 }
144
145 float bacaSensorSuhuTanah()
146 {
147     sensorSuhuTanah.requestTemperatures();
148     float suhuTanah = sensorSuhuTanah.getTempCByIndex(0);
149     return suhuTanah;
150 }
151
152
153 float bacaSensorPhTanah(){
154     //mengambil data sensing sensor
155     sensorValue = analogRead(analogInPin);
156
157     //Mathematical conversion from ADC to pH
158     //Konversi analog menjadi digital
159     //rumus didapat berdasarkan datasheet
160     phValue = (-0.0693*sensorValue)+7.3855;
161
162     return phValue*-1;
163 }
164
165 String conversiDataFloatToString(){
166     String kelembabanTanah = String(bacaSensorKelembabanTanah());
167     String kelembabanUdara = String(bacaSensorKelembabanUdara());
168     String suhuTanah = String(bacaSensorSuhuTanah());
169     String suhuUdara = String(bacaSensorSuhuUdara());
170     String phTanah = String(bacaSensorPhTanah());
171     String res = node3+"|"+kelembabanTanah+"|"+kelembabanUdara+"|"+suhuTanah+"|"+suhuUdara+"|"+phTanah+"|"+
172         WrapperStatusNodeDanSensing();
173     return res;
174 }
```

```

174
175 String WrapperStatusNodeDanSensing(){
176     boolean isActive = checkStatusNode();
177     boolean isSensing = checkStatusSensing();
178     String isActiveStr = "";
179     String isSensingStr = "";
180     String res = "";
181
182     if(isActive==true){
183         isActiveStr = "Online";
184     }else{
185         isActiveStr = "Offline";
186     }
187
188
189     if(isSensing==true){
190         isSensingStr = "Sensing";
191     }else{
192         isSensingStr = "Not_Sensing";
193     }
194
195     res = isActiveStr+" | "+isSensingStr;
196
197     return res;
198 }
199
200 String transmisiPengiriman(String pesan){
201     xbee.print(pesan);
202 }
203
204 boolean checkStatusNode(){
205     if(checkStatusSensing()==true){
206         return true;
207     }
208     else{ //check lampu indikator 'on' arduino
209         if(digitalRead(LED_BUILTIN == HIGH)){
210             return true;
211         }
212         else{
213             return false;
214         }
215     }
216 }
217
218 boolean checkStatusSensing(){
219     boolean kelembabanTanah = false;
220     boolean kelembabanUdara = false;
221     boolean suhuTanah = false;
222     boolean suhuUdara = false;
223     boolean pHTanah = true; //tidak memiliki value pasti ketika sensor bermasalah
224     boolean isSensing = false;
225
226     if(bacaSensorKelembabanTanah()!=1023.0){ //1023.0 adalah value yang didapatkan jika sensor suhu tanah bermasalah
227         kelembabanTanah = true;
228     }
229     if(isnan(bacaSensorKelembabanUdara())==false){ //bukan nan (mengembalikan value), jika bermasalah value = nan
230         kelembabanUdara = true;
231     }
232     if(bacaSensorSuhuTanah()!=-127.00){ // -127.00 adalah value yang didapatkan jika sensor suhu tanah bermasalah
233         suhuTanah = true;
234     }
235     if(isnan(bacaSensorSuhuUdara())==false){ //bukan nan (mengembalikan value), jika bermasalah value = nan
236         suhuUdara = true;
237     }
238
239     if((kelembabanTanah&&kelembabanUdara&&suhuTanah&&suhuUdara&&pHTanah)==true){
240         isSensing = true;
241     }
242
243     return isSensing;
244 }

```


LAMPIRAN B

KODE PROGRAM BASE STATION

Kode B.1: BaseStation.py

```
1 import datetime
2 import serial
3 import RPi.GPIO as GPIO
4 import mysql.connector
5 from mysql.connector import Error
6 import concurrent.futures
7 import os
8
9 GPIO.setmode(GPIO.BCM)
10 GPIO.setwarnings(False)
11 GPIO.setup(23,GPIO.OUT)
12
13
14 appRunning = True
15 menuShow = True
16 sensingRunning = True
17 statusSensing = False
18 global statusNode
19 statusNum = -1
20 getPing = True
21 respon = 0
22
23 print("Base_Station_Online")
24 print("====")
25 print("Pilihan_Eksekusi_Program")
26 print("1._Check_Status_Node")
27 print("2._Mulai_Sensing")
28 print("3._Stop_Sensing")
29 print("4._Matikan_Aplikasi_Base_Station")
30 print("====")
31 print("Masukan_Nomor_Instruksi=")
32
33
34 inputNum = input()
35 #print(inputNum)
36 counter = 0
37
38
39
40 def mainMenu():
41     print("====")
42     print("Pilihan_Eksekusi_Program")
43     print("1._Check_Status_Node")
44     print("2._Mulai_Sensing")
45     print("3._Stop_Sensing")
46     print("4._Matikan_Aplikasi_Base_Station")
47     print("====")
48     print("Masukan_Nomor_Instruksi=")
49
50
51 ser = serial.Serial(
52     port="/dev/ttyS0",
53     baudrate = 9600,
54     parity=serial.PARITY_NONE,
55     stopbits=serial.STOPBITS_ONE,
56     bytesize=serial.EIGHTBITS,
57     timeout=1
58 )
59
60
61
62 def getDataSensing(dataSensing):
63     # Data yang dikirim Arduino (String)
64     # "Node 1"+ "+kelembabanTanah+" "+kelembabanUdara+" "+suhuTanah+" "+suhuUdara+" "+phTanah+" "+statusNode+" "+statusSensing";
65
66     potongData = dataSensing.split("|")
67
68     if len(potongData) > 1:
69         nodeSensing = potongData[0]
70         kelembabanTanah = potongData[1]
71         kelembabanUdara = potongData[2]
72         suhuTanah = potongData[3]
73         suhuUdara = potongData[4]
74         phTanah = potongData[5]
75         statusNode = potongData[6]
```

```

76     statusSensing = potongData[7]
77
78     waktuSensing = datetime.datetime.now()
79     waktuSensing = waktuSensing.strftime('%Y-%m-%d_%H:%M:%S')
80
81     return nodeSensing,kelembabanTanah,kelembabanUdara,suhuTanah,suhuUdara,phTanah,statusNode,statusSensing,waktuSensing
82
83
84 def getPingArduino(dataNode):
85     potongData = dataNode.split(" | ")
86
87     if len(potongData) > 1:
88         nodeSensing = potongData[0]
89         kelembabanTanah = potongData[1]
90         kelembabanUdara = potongData[2]
91         suhuTanah = potongData[3]
92         suhuUdara = potongData[4]
93         phTanah = potongData[5]
94         statusNode = potongData[6]
95         statusSensing = potongData[7]
96
97     return nodeSensing,statusNode
98
99
100 #jumlah node yang disebar (thread)
101 POOL_SIZE = 3
102
103 def updateStatusSensing(dataSensing):
104     db = mysql.connector.connect(
105         host = 'localhost',
106         database = 'Skripsi_II',
107         user = 'admin',
108         password = 'root',
109         pool_name = 'mypool',
110         pool_size = POOL_SIZE+1
111     )
112
113     waktuSensing = datetime.datetime.now()
114     waktuSensing = waktuSensing.strftime('%Y-%m-%d_%H:%M:%S')
115     statusSensing = "Not_Sensing"
116
117     #cursor digunakan untuk memasukan data ke mysql, dan eksekusi query
118     cursor = db.cursor(buffered=True)
119
120
121     queryNode2 = ("UPDATE_nodesensor_SET_status_sensing=%s,_waktu_node=%s")
122     queryInputNode2 = (statusSensing,waktuSensing)
123
124
125     cursor.execute(queryNode2,queryInputNode2)
126     #cursor.execute(queryNode3,queryInputNode3)
127
128     db.commit()
129     cursor.close()
130     db.close()
131
132
133 def goingOffline(dataSensing):
134     db = mysql.connector.connect(
135         host = 'localhost',
136         database = 'Skripsi_II',
137         user = 'admin',
138         password = 'root',
139         pool_name = 'mypool',
140         pool_size = POOL_SIZE+1
141     )
142
143     waktuSensing = datetime.datetime.now()
144     waktuSensing = waktuSensing.strftime('%Y-%m-%d_%H:%M:%S')
145     statusNode = "Offline"
146     statusSensing = "Not_Sensing"
147
148     #cursor digunakan untuk memasukan data ke mysql, dan eksekusi query
149     cursor = db.cursor(buffered=True)
150
151
152     queryNode2 = ("UPDATE_nodesensor_SET_status_node=%s_,status_sensing=%s,_waktu_node=%s,_WHERE_kode_node!=5")
153     queryInputNode2 = (statusNode,statusSensing,waktuSensing)
154
155
156     cursor.execute(queryNode2,queryInputNode2)
157
158     db.commit()
159     cursor.close()
160     db.close()
161
162
163 def sendDataSensing(dataSensing):
164     db = mysql.connector.connect(
165         host = 'localhost',
166         database = 'Skripsi_II',
167         user = 'admin',
168         password = 'root',
169         pool_name = 'mypool',
170         pool_size = POOL_SIZE+1
171     )
172
173     #global statusNode
174

```



```

271
272     elif(inputNum == "1"):
273         print("Mengirim_perintah_check_status")
274         print("Silakan_tunggu_respon...")
275         while (counter<28): #28 detik untuk cek seluruh node yang disebar
276             # kirim perintah check arduino
277             ser.write(str.encode("b").strip())
278
279
280         # ambil hasil komunikasi arduino
281         decoder=ser.readline().decode("ascii").strip()
282         with concurrent.futures.ThreadPoolExecutor() as executor:
283             counterAdd()
284             counter+=1
285             future3 = executor.submit(getPingArduino,decoder)
286
287             if future3.result()!=None:
288                 print("")
289                 print("Ping_Node_Diterima")
290                 print(future3.result())
291                 future4 = executor.submit(sentDataSensing, future3.result())
292                 global statusNode
293                 statusNode = True
294                 respon+=1
295
296             if(respon==0):
297                 print(" ")
298                 print("Tidak Ada Respon")
299                 print("Silakan_Check_Perangkat..")
300                 print(" ")
301             else:
302                 print(" ")
303                 print("Check_Node_Selesai")
304                 print(" ")
305             counterSet()
306             respon=0
307             mainMenu()
308
309     elif(inputNum == "4"):
310         # kirim perintah matikan sensing
311         ser.write(str.encode("c").strip())
312         finding = False
313
314         decoder=ser.readline().decode("ascii").strip()
315         with concurrent.futures.ThreadPoolExecutor() as executor:
316             future = executor.submit(goingOffline,decoder)
317
318             #Stop App Base Station
319             print("Eksekusi : Matikan_Aplikasi_Base_Station")
320             os.system("skripsi_laravel_BS_Rev1.py")
321             print("====")
322             print("Sensing_Dihentikan!")
323             print("Base_Station_Offline")
324             exit()
325
326
327     elif(inputNum == "3"):
328         # kirim perintah matikan sensing
329         ser.write(str.encode("c").strip())
330         finding = False
331
332
333         # kirim perintah stop sensing
334         print("Sensing_Dihentikan!")
335         statusSensing = False
336
337         decoder=ser.readline().decode("ascii").strip()
338         with concurrent.futures.ThreadPoolExecutor() as executor:
339             future = executor.submit(updateStatusSensing,decoder)
340
341
342         mainMenu()
343
344     else:
345         print("Pilihan_Eksekusi_Salah")
346         print("Restart_Aplikasi!")
347         exit()
348
349
350     inputNum=input()
351

```

LAMPIRAN C

KODE PROGRAM APLIKASI PENGGUNA (WEBSITE)

Kode C.1: skripsi_header.blade.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
6
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
8
9   <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KKN" crossorigin="anonymous"></script>
10  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPsvkXusvfa0b4Q" crossorigin="anonymous"></script>
11  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyJuar5+76PVCmYl" crossorigin="anonymous"></script>
12
13 <title>Home</title>
14 <link href="{{ asset('css/skripsi_mainstyle.css') }}" rel="stylesheet" type="text/css" >
15 </head>
16 <style>
17 </style>
18
19 <body>
20
21   <ul style="margin-top: .4px;">
22     <li></li>
23     <li><a href="/">Halaman Utama</a></li>
24     <li><a href="/check-status">Check Status</a></li>
25     <li><a href="/sensing">Sensing</a></li>
26
27     <li style="float: right;"><a href="/cara-pakai">Cara Pakai</a></li>
28     <li style="float: right;"><a href="/print-sensing">Print Sensing</a></li>
29
30   </ul>
31
32 </body>
33 </html>
```

Kode C.2: skripsi_home.blade.php

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
6   <link rel="icon" href="{{ asset('assets/unpar.png') }}">
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
8
9   <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KKN" crossorigin="anonymous"></script>
10  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPsvkXusvfa0b4Q" crossorigin="anonymous"></script>
11  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyJuar5+76PVCmYl" crossorigin="anonymous"></script>
12
13 <title>Halaman Utama | Skripsi</title>
14 <link href="{{ asset('css/skripsi_mainstyle.css') }}" rel="stylesheet" type="text/css" >
15 </head>
16 <style>
17   body{
18     background-image: url("assets/home_bg.jpg");
19     background-size: cover;
20     padding-left: 99px;
21     padding-right: 108px;
22   }
23 </style>
24 <body>
25   @include('skripsi_header')
26
27   <h1>Aplikasi Pemantauan<br>
28     Kualitas Tanah Sawah<br>
```

```

29      Berbasis WSN
30  </h1>
31
32  <hr>
33
34  <h2>
35      Reynaldi Irfan A <br>
36      2016730045
37  </h2>
38
39  <button class="button_button4" onclick="location.href='{{url('sensing')}}'">Let's Get Started</button>
40
41 </body>
</html>

```

Kode C.3: skripsi_checkstatus.blade.php

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
6      <link rel="icon" href="{{asset('assets/unpar.png')}}">
7      <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQiaoWXA+058RXPxPgfy4IWvTNN0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
8
9      <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkYIK3UENzm7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGF93hXpG5KKN" crossorigin="anonymous"></script>
10     <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtvzRN7W3mgPxhU9K/Sc0sAP7hUiX39j7fakFPsvXusvfa0b40" crossorigin="anonymous"></script>
11     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQxSFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>
12
13     <title>Check Status | Skripsi</title>
14     <link href="{{asset('css/skripsi_mainstyle.css')}}" rel="stylesheet" type="text/css" >
15 </head>
16 <style>
17     body{
18         background-image: url("assets/sensing_bg.jpg") ;
19         background-size: 100%;
20         padding-left: 99px;
21         padding-right: 108px;
22         background-repeat: no-repeat;
23     }
24 </style>
25 <body>
26     @include('skripsi_header')
27     <br>
28     <table style="margin-top:15px;float:left; width:45%;" id="table1">
29         @foreach ($check1 as $check1)
30             <tr style="background-color: #rgba(0,0,0,0.5)">
31                 <th>Variable</th>
32                 <th>Value</th>
33             </tr>
34             <tr>
35                 <td>Nama Node</td>
36                 <td>{{ $check1->nama_node }}</td>
37             </tr>
38             <tr>
39                 <td>Status Node</td>
40                 <td id="n1_aktif">{{ $check1->status_node }}</td>
41             </tr>
42             <tr>
43                 <td>Status Sensing</td>
44                 <td id="n1_sensing">{{ $check1->status_sensing }}</td>
45             </tr>
46         @endforeach
47     </table>
48
49     <table style="margin-top:15px;float:left; width:45%;" id="table2">
50         @foreach ($check2 as $check2)
51             <tr style="background-color: #rgba(0,0,0,0.5)">
52                 <th>Variable</th>
53                 <th>Value</th>
54             </tr>
55             <tr>
56                 <td>Nama Node</td>
57                 <td>{{ $check2->nama_node }}</td>
58             </tr>
59             <tr>
60                 <td>Status Node</td>
61                 <td id="n2_aktif">{{ $check2->status_node }}</td>
62             </tr>
63             <tr>
64                 <td>Status Sensing</td>
65                 <td id="n2_sensing">{{ $check2->status_sensing }}</td>
66             </tr>
67         @endforeach
68     </table>
69
70     <br>
71
72     <table style="margin-top:24px;float:left; width:45%;" id="table3">
73         @foreach ($check3 as $check3)
74             <tr style="background-color: #rgba(0,0,0,0.5)">
75                 <th>Variable</th>
76                 <th>Value</th>
77             </tr>
78             <tr>

```

```

79        <td>Nama Node</td>
80        <td>{{ $check3->nama_node }}</td>
81    </tr>
82    <tr>
83        <td>Status Node</td>
84        <td id="n3_aktif">{{ $check3->status_node }}</td>
85    </tr>
86    <tr>
87        <td>Status Sensing</td>
88        <td id="n3_sensing">{{ $check3->status_sensing }}</td>
89    </tr>
90    @endforeach
91 </table>
92
93 <table style="margin-top: _24px; float: _left; width: 45%;" id="table4">
94    @foreach ($check4 as $check4)
95    <tr style="background-color: _rgba(0, _0, _0, _0.5)">
96        <th>Variable</th>
97        <th>Value</th>
98    </tr>
99    <tr>
100        <td>Nama Node</td>
101        <td>{{ $check4->nama_node }}</td>
102    </tr>
103    <tr>
104        <td>Status Node</td>
105        <td id="n4_aktif">{{ $check4->status_node }}</td>
106    </tr>
107    <tr>
108        <td>Status Sensing</td>
109        <td id="n4_sensing">{{ $check4->status_sensing }}</td>
110    </tr>
111    @endforeach
112 </table>
113
114 <br>
115
116 <table style="margin-top: _24px; float: _left; margin-left: 26%; " id="table5">
117    @foreach ($check5 as $check5)
118    <tr style="background-color: _rgba(0, _0, _0, _0.5)">
119        <th>Variable</th>
120        <th>Value</th>
121    </tr>
122    <tr>
123        <td>Nama Node</td>
124        <td>{{ $check5->nama_node }}</td>
125    </tr>
126    <tr>
127        <td>Status Node</td>
128        <td id="n5_aktif">{{ $check5->status_node }}</td>
129    </tr>
130    @endforeach
131 </table>
132
133 <div style="margin-left: 15px; margin-top: _418px;">
134     <a href="/" style="color: _white;">Kembali</a>
135 </div>
136
137 <form method="post" action="/check-status/update/">
138     {{ csrf_field() }}
139     {{ method_field('PUT') }}
140     <button class="button_button4" style="visibility:hidden;" onclick="changeButtonText()" value="online" id="sensing_button"
141         style="margin-top: _140px; margin-right: _15px;">Update</button>
142 </form>
143
144 </body>
145
<script>
    var buttonSensing = document.getElementById('sensing_button').value;
    var statusSensing = "true";
146
147
    var node1Stat = document.getElementById('n1_aktif').textContent;
    var node1Sens = document.getElementById('n1_sensing').textContent;
148
149
    var node2Stat = document.getElementById('n2_aktif').textContent;
    var node2Sens = document.getElementById('n2_sensing').textContent;
150
151
    var node3Stat = document.getElementById('n3_aktif').textContent;
    var node3Sens = document.getElementById('n3_sensing').textContent;
152
153
    var node4Stat = document.getElementById('n4_aktif').textContent;
    var node4Sens = document.getElementById('n4_sensing').textContent;
154
155
    var node5Stat = document.getElementById('n5_aktif').textContent;
156
157
    //CHECK STATUS NODE
158    if(node1Stat=='Online'){
159        document.getElementById('n1_aktif').style.color="yellow";
160        document.getElementById('n1_sensing').style.color="yellow";
161    }
162    if(node2Stat=='Online'){
163        document.getElementById('n2_aktif').style.color="yellow";
164        document.getElementById('n2_sensing').style.color="yellow";
165    }
166    if(node3Stat!='Online'){
167        document.getElementById('n3_aktif').style.color="yellow";
168        document.getElementById('n3_sensing').style.color="yellow";
169    }
170
171
172
173
174
175
176
}

```

```

177 | if(node4Stat!=='Online'){
178 |   document.getElementById('n4_aktif').style.color="yellow";
179 |   document.getElementById('n4_sensing').style.color="yellow";
180 |
181 | if(node5Stat!=='Online'){
182 |   document.getElementById('n5_aktif').style.color="yellow";
183 | }
184 |
185 //CHECK STATUS SENSING
186 if(node1Sens!=='Sensing'){
187   document.getElementById('n1_sensing').style.color="yellow";
188 }
189 if(node2Sens!=='Sensing'){
190   document.getElementById('n2_sensing').style.color="yellow";
191 }
192 if(node3Sens!=='Sensing'){
193   document.getElementById('n3_sensing').style.color="yellow";
194 }
195 if(node4Sens!=='Sensing'){
196   document.getElementById('n4_sensing').style.color="yellow";
197 }
198
199
200 //TIMER UPDATE 30 sec STATUS NODE
201 if(node1Stat=='Online'){
202   setTimeout(setStatusN1, 30000);
203 }
204 if(node2Stat=='Online'){
205   setTimeout(setStatusN2, 30000);
206 }
207 if(node3Stat=='Online'){
208   setTimeout(setStatusN3, 30000);
209 }
210 if(node4Stat=='Online'){
211   setTimeout(setStatusN4, 30000);
212 }
213
214
215 //GANTI STATUS
216 function setStatusN1() {
217   node1Stat='Offline'
218   node1Sens!='Not_Sensing'
219 }
220 function setStatusN2() {
221   node2Stat='Offline'
222   node1Sens!='Not_Sensing'
223 }
224 function setStatusN3() {
225   node3Stat='Offline'
226   node1Sens!='Not_Sensing'
227 }
228 function setStatusN4() {
229   node4Stat='Offline'
230   node1Sens!='Not_Sensing'
231 }
232
233 </script>

```

Kode C.4: skripsi_sensing.blade.php

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
6   <link rel="icon" href="{{asset('assets/unpar.png')}}">
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXpPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
8
9   <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkYIK3UENzm7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGF93hXpG5KKN" crossorigin="anonymous"></script>
10  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y10Ktv3Rn/W3mgPxhU9K/ScQsAP7UiBx39j7fakFPsvXusvfa0b40" crossorigin="anonymous"></script>
11  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQxSfFWpi1MquVdAyJuar5+76PVcmYL" crossorigin="anonymous"></script>
12
13  <title>Sensing | Skripsi</title>
14  <link href="{{asset('css/skripsi_mainstyle.css')}}" rel="stylesheet" type="text/css" >
15 </head>
16 {{-- <script src="https://js.pusher.com/6.0/pusher.min.js"></script>
17 <script>
18   // Enable pusher logging - don't include this in production
19   Pusher.logToConsole = true;
20
21   var pusher = new Pusher('c559846b0d74d3af1611', {
22     cluster: 'ap1'
23   });
24
25   var channel = pusher.subscribe('my-channel');
26   channel.bind('my_event', function(data) {
27     alert(JSON.stringify(data));
28   });
29 </script> TADINYA UNTUK WEBSOCKET--}}
30 <style>
31   body{
32     background-image: url("assets/sensing_bg.jpg");
33     background-size: cover;
34

```

```

35         padding-left: 99px;
36         padding-right: 108px;
37     }
38 
```

```

39 </style>
40 <body>
41     @include('skripsi_header')
42     <table style="margin-top:_15px;float:_left;_width:45%;" id="table1">
43         @foreach ($nodes1 as $node1)
44             <tr style="background-color:_rgba(0,_0,_0,_0.5)">
45                 <th>Variable</th>
46                 <th>Value</th>
47             </tr>
48             <tr>
49                 <td>Nama Node</td>
50                 <td id="namanode1">{{ $node1->nama_node }}</td>
51             </tr>
52             <tr>
53                 <td>Keasaman (pH)</td>
54                 <td id="node1_ph">{{ $node1->ph_tanah }}</td>
55             </tr>
56             <tr>
57                 <td>Kelembaban Tanah</td>
58                 <td id="node1_lembab">{{ $node1->kelembaban_tanah }} %</td>
59             </tr>
60             <tr>
61                 <td>Suhu Tanah</td>
62                 <td id="node1_suhuTh">{{ $node1->suhu_tanah }} <span>#8451;</span></td>
63             </tr>
64             <tr>
65                 <td>Suhu Udara</td>
66                 <td id="node1_suhu">{{ $node1->suhu_udara }} <span>#8451;</span></td>
67             </tr>
68             <tr>
69                 <td>Waktu Sensing</td>
70                 <td id="node1_waktu">{{ $node1->waktu_sensing }}</td>
71             </tr>
72         @endforeach
73     </table>
74
75     <table style="margin-top:_15px;float:_left;_width:45%;" id="table2">
76         @foreach ($nodes2 as $node2)
77             <tr style="background-color:_rgba(0,_0,_0,_0.5)">
78                 <th>Variable</th>
79                 <th>Value</th>
80             </tr>
81             <tr>
82                 <td>Nama Node</td>
83                 <td>{{ $node2->nama_node }}</td>
84             </tr>
85             <tr>
86                 <td>Keasaman (pH)</td>
87                 <td id="node2_ph">{{ $node2->ph_tanah }}</td>
88             </tr>
89             <tr>
90                 <td>Kelembaban Tanah</td>
91                 <td id="node2_lembab">{{ $node2->kelembaban_tanah }} %</td>
92             </tr>
93             <tr>
94                 <td>Suhu Tanah</td>
95                 <td id="node2_suhuTh">{{ $node2->suhu_tanah }} <span>#8451;</span></td>
96             </tr>
97             <tr>
98                 <td>Suhu Udara</td>
99                 <td id="node2_suhu">{{ $node2->suhu_udara }} <span>#8451;</span></td>
100            </tr>
101            <tr>
102                <td>Waktu Sensing</td>
103                <td id="node2_waktu">{{ $node2->waktu_sensing }}</td>
104            </tr>
105        @endforeach
106     </table>
107
108     <br>
109
110     <table style="margin-top:_16px;float:_left;_width:45%;" id="table3">
111         @foreach ($nodes3 as $node3)
112             <tr style="background-color:_rgba(0,_0,_0,_0.5)">
113                 <th>Variable</th>
114                 <th>Value</th>
115             </tr>
116             <tr>
117                 <td>Nama Node</td>
118                 <td>{{ $node3->nama_node }}</td>
119             </tr>
120             <tr>
121                 <td>Keasaman (pH)</td>
122                 <td id="node3_ph">{{ $node3->ph_tanah }}</td>
123             </tr>
124             <tr>
125                 <td>Kelembaban Tanah</td>
126                 <td id="node3_lembab">{{ $node3->kelembaban_tanah }} %</td>
127             </tr>
128             <tr>
129                 <td>Suhu Tanah</td>
130                 <td id="node3_suhuTh">{{ $node3->suhu_tanah }} <span>#8451;</span></td>
131             </tr>
132             <tr>
133                 <td>Suhu Udara</td>

```

```

134     <td id="node3_suhu">{{ $node3->suhu_udara }} <span>{{ $node3->suhu_udara }} </span></td>
135   </tr>
136   <tr>
137     <td>Waktu Sensing</td>
138     <td id="node3_waktu">{{ $node3->waktu_sensing }} </td>
139   </tr>
140 @endforeach
141 </table>
142
143 <table style="margin-top:20px;float:left; width:45%;" id="table4" >
144   @foreach ($nodes4 as $node4)
145   <tr style="background-color:rgba(0,0,0,0.5)">
146     <th>Variable</th>
147     <th>Value</th>
148   </tr>
149   <tr>
150     <td>Nama Node</td>
151     <td>{{ $node4->nama_node }}</td>
152   </tr>
153   <tr>
154     <td>Keasaman (pH)</td>
155     <td id="node4_ph">{{ $node4->ph_tanah }}</td>
156   </tr>
157   <tr>
158     <td>Kelembaban Tanah</td>
159     <td id="node4_lembab">{{ $node4->kelembaban_tanah }} %</td>
160   </tr>
161   <tr>
162     <td>Suhu Tanah</td>
163     <td id="node4_suhuTh">{{ $node4->suhu_tanah }} <span>{{ $node4->suhu_tanah }} </span></td>
164   </tr>
165   <tr>
166     <td>Suhu Udara</td>
167     <td id="node4_suhu">{{ $node4->suhu_udara }} <span>{{ $node4->suhu_udara }} </span></td>
168   </tr>
169   <tr>
170     <td>Waktu Sensing</td>
171     <td id="node4_waktu">{{ $node4->waktu_sensing }} </td>
172   </tr>
173 @endforeach
174 </table>
175
176 <button style="visibility:hidden;" class="button_button4" onclick="changeButtonText()" value="online" id="sensing_button"
177   style="margin-top:140px; margin-right:15px;">Pause Alert</button>
178
179 <!-- <div style="margin-left:15px; margin-top:468px;">
180   <a href="/" style="color:white;">Kembali</a>
181 </div> -->
182
183 <script>
184
185 var buttonSensing = document.getElementById('sensing_button').value;
186 var statusSensing = "true";
187
188
189 function changeButtonText(){
190   if(buttonSensing=="offline"){
191     document.getElementById('sensing_button').innerHTML = 'Stop Sensing';
192     buttonSensing = "online";
193     alert("Sensing akan dimulai!");
194     statusSensing = "true";
195     location.reload(),6000;
196     return false;
197
198   }
199   else{
200     document.getElementById('sensing_button').innerHTML = 'Mulai Sensing';
201     alert("Sensing dihentikan! Klik OK untuk melanjutkan sensing");
202
203
204     var table1 = document.getElementById('table1');
205     var table2 = document.getElementById('table2');
206     var table3 = document.getElementById('table3');
207     var table = document.getElementById('table4');
208     var lengthTable = table.rows.length;
209
210     for(i=1;i<lengthTable;i++){
211       var cells = table.rows.item(i).cells;
212       var cells1 = table1.rows.item(i).cells;
213       var cells2 = table2.rows.item(i).cells;
214       var cells3 = table3.rows.item(i).cells;
215
216       //gets amount of cells of current row
217       var cellLength = cells.length;
218
219       //loops through each cell in current row
220       for(var j = 1; j < cellLength; j++){
221         /* get your cell info here */
222         cells.item(j).innerHTML= '-';
223         cells1.item(j).innerHTML= '-';
224         cells2.item(j).innerHTML= '-';
225         cells3.item(j).innerHTML= '-';
226       }
227     }
228
229     buttonSensing = "offline";
230     statusSensing = "false";
231   }
}

```

```

232}
233
234 if(statusSensing=="true"){
235     setTimeout(function() {
236         location.reload();
237     }, 5000);
238 }
239
240 //CHECK KLASIFIKASI PH TANAH
241 var nPh1 = parseFloat(document.getElementById('node1_ph').textContent);
242 var nPh2 = parseFloat(document.getElementById('node2_ph').textContent);
243 var nPh3 = parseFloat(document.getElementById('node3_ph').textContent);
244 //var nPh4 = parseFloat(document.getElementById('node4_ph').textContent);
245
246 if(nPh1!=7){
247     document.getElementById('node1_ph').style.color = "yellow";
248 }
249 if(nPh2!=7){
250     document.getElementById('node2_ph').style.color = "yellow";
251 }
252
253 if(nPh3!=7){
254     document.getElementById('node3_ph').style.color = "yellow";
255 }
256 /**
257 if(nPh4!=7){
258     document.getElementById('node4_ph').style.color = "yellow";
259 }
260 */
261
262 //CHECK KLASIFIKASI KELEMBABAN TANAH
263 var nLembab1 = parseFloat(document.getElementById('node1_lembab').textContent);
264 var nLembab2 = parseFloat(document.getElementById('node2_lembab').textContent);
265 var nLembab3 = parseFloat(document.getElementById('node3_lembab').textContent);
266 //var nLembab4 = parseFloat(document.getElementById('node2_lembab').textContent);
267
268 if(nLembab1<40 || nLembab1>62){
269     document.getElementById('node1_lembab').style.color = "yellow";
270 }
271 if(nLembab2<40 || nLembab2>62){
272     document.getElementById('node2_lembab').style.color = "yellow";
273 }
274
275 if(nLembab3<40 || nLembab3>62){
276     document.getElementById('node3_lembab').style.color = "yellow";
277 }
278 /**
279 if(nLembab4<40 || nLembab4>62){
280     document.getElementById('node4_lembab').style.color = "yellow";
281 }
282 */
283
284 //CHECK KLASIFIKASI SUHU TANAH
285 var nSuhuTh1 = parseFloat(document.getElementById('node1_suhuTh').textContent);
286 var nSuhuTh2 = parseFloat(document.getElementById('node2_suhuTh').textContent);
287 var nSuhuTh3 = parseFloat(document.getElementById('node3_suhuTh').textContent);
288 //var nSuhuTh4 = parseFloat(document.getElementById('node4_suhuTh').textContent);
289
290
291 if(nSuhuTh1<10 || nSuhuTh1>30){
292     document.getElementById('node1_suhuTh').style.color = "yellow";
293 }
294 if(nSuhuTh2<10 || nSuhuTh2>30){
295     document.getElementById('node2_suhuTh').style.color = "yellow";
296 }
297
298 if(nSuhuTh3<10 || nSuhuTh3>30){
299     document.getElementById('node3_suhuTh').style.color = "yellow";
300 }
301 /**
302 if(nSuhuTh4<10 || nSuhuTh4>30){
303     document.getElementById('node4_suhuTh').style.color = "yellow";
304 }
305 */
306
307 //CHECK KLASIFIKASI SUHU UDARA
308 var nSuhu1 = parseFloat(document.getElementById('node1_suhu').textContent);
309 var nSuhu2 = parseFloat(document.getElementById('node2_suhu').textContent);
310 var nSuhu3 = parseFloat(document.getElementById('node3_suhu').textContent);
311 //var nSuhu4 = parseFloat(document.getElementById('node4_suhu').textContent);
312
313
314 if(nSuhu1<18 || nSuhu1>26){
315     document.getElementById('node1_suhu').style.color = "yellow";
316 }
317 if(nSuhu2<18 || nSuhu2>26){
318     document.getElementById('node2_suhu').style.color = "yellow";
319 }
320
321 if(nSuhu3<18 || nSuhu3>26){
322     document.getElementById('node3_suhu').style.color = "yellow";
323 }
324 /**
325 if(nSuhu4<18 || nSuhu4>26){
326     document.getElementById('node4_suhu').style.color = "yellow";
327 }
328 */
329
330 </script>

```

331 </body>
 332 </html>

Kode C.5: skripsi_perangkat.blade.php

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, _initial-scale=1.0, _shrink-to-fit=no">
6   <link rel="icon" href="{{ asset('assets/unpar.png') }}>
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xq0IaoWXA+058XXPg6fy4IWvTNh0E263XmFcJlSAwiGFAW/daiS6JXm" crossorigin="anonymous">
8
9   <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkYIK3UEzmM7KCkR/rE9/Opg6aAZGJwFDMVNA/GpGF93hXpGSKh" crossorigin="anonymous"></script>
10  <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@1.12.9/dist/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7HuiB39j7fakFPsvXusvfa0b40" crossorigin="anonymous"></script>
11  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQx5ffWp11MquVdAyjUar5+76PVcmYL" crossorigin="anonymous"></script>
12  <script src="https://kit.fontawesome.com/a076d05399.js"></script>
13
14 <title>Print Sensing | Skripsi</title>
15 <link href="{{ asset('css/skripsi_mainstyle.css') }}" rel="stylesheet" type="text/css" >
16 </head>
17 <style>
18 body{
19   background-image: url("assets/under_maintenance.jpg") ;
20   background-repeat: repeat-y;
21   background-size: 100%;
22   padding-left: 99px;
23   padding-right: 108px;
24
25 .dropbtn {
26   background-color: lightgrey;
27   color: black;
28   padding: 4px;
29   font-size: 15px;
30   border: none;
31   cursor: pointer;
32 }
33
34 .dropbtn:hover, .dropbtn:focus {
35   background-color: #EEEEEDED;
36 }
37
38 .dropdown {
39   position: relative;
40   display: inline-block;
41 }
42
43 .dropdown-content {
44   display: none;
45   position: absolute;
46   background-color: #EEEEEDED;
47   min-width: 160px;
48   overflow: auto;
49   box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
50   z-index: 1;
51 }
52
53 .dropdown-content a {
54   color: black;
55   padding: 12px 16px;
56   text-decoration: none;
57   display: block;
58 }
59
60 .dropdown a:hover {background-color: #ddd;}
61
62 .show {display: block;}
63 </style>
64 <body>
65   @include('skripsi_header')
66
67   <div style="position:center; margin-top:20px">
68     <form action="/print-sensing/cari" method="GET">
69
70       <a style="color:white">Mulai :</a>
71       <input placeholder="Date" name="cariAwal" class="textbox-n" type="text" onfocus="(this.type='date')" id="date" value="{{ old('cariAwal') }}>
72
73       <a style="color:white; padding-left:10px">Sampai :</a>
74       <input placeholder="Date" name="cariAkhir" class="textbox-n" type="text" onfocus="(this.type='date')" id="date" value="{{ old('cariAkhir') }}>
75
76       <input type="submit" value="cari">
77     </form>
78
79
80
81     <button onclick="window.print()" style="float:right; border-radius:5px; ">Print</button>
82
83     <div class="dropdown" style="float:right; margin-right:10px">
84       <button onclick="myFunction()" class="dropbtn">Urutkan <i class="fas fa-angle-down"></i> </button>
85       <div id="myDropdown" class="dropdown-content">
86         <a href="/print-sensing/sortkode">Kode Petak</a>
87

```

```

88         <a href="/print-sensing">Waktu Sensing</a>
89     </div>
90   </div>
91 </div>
92
93 <table style="margin-top: 24px; float: left; width: 100%;" id="table1">
94
95   <tr style="background-color: #rgba(0, 0, 0, 0.5)">
96     <!--<th>ID Sensing</th> -->
97     <th>Jenis Tanah</th>
98     <th>Kode Petak</th>
99     <th>Waktu Sensing</th>
100    <th>Keasaman (pH)</th>
101    <th>Kelembaban Tanah</th>
102    <th>Kelembaban Udara</th>
103    <th>Suhu Tanah</th>
104    <th>Suhu Udara</th>
105  </tr>
106  @foreach ($tanah1 as $tanah1)
107  <tr>
108    <!--<td>{$tanah1->id_tanah}</td>-->
109    <!--<td>{$tanah1->jenis_tanah}</td>-->
110    <td>{$tanah1->kode_node}</td>
111    <td>{$tanah1->waktu_sensing}</td>
112    <td>{$tanah1->ph_tanah}</td>
113    <td>{$tanah1->kelembaban_tanah}</td>
114    <td>{$tanah1->kelembaban_udara}</td>
115    <td>{$tanah1->suhu_tanah}</td>
116    <td>{$tanah1->suhu_udara}</td>
117  </tr>
118
119  </tr>
120  @endforeach
121 </table>
122
123
124 <script>
125   /* When the user clicks on the button,
126    toggle between hiding and showing the dropdown content */
127   function myFunction() {
128     document.getElementById("myDropdown").classList.toggle("show");
129   }
130
131   // Close the dropdown if the user clicks outside of it
132   window.onclick = function(event) {
133     if (!event.target.matches('.dropbtn')) {
134       var dropdowns = document.getElementsByClassName("dropdown-content");
135       var i;
136       for (i = 0; i < dropdowns.length; i++) {
137         var openDropdown = dropdowns[i];
138         if (openDropdown.classList.contains('show')) {
139           openDropdown.classList.remove('show');
140         }
141       }
142     }
143   }
144 </script>
145
146 </body>
147
148 </html>

```

Kode C.6: skripsi_carapakai.blade.php

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-fit=no">
6   <link rel="icon" href="{{asset('assets/unpar.png')}}"/>
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQlaoWXA+058RXpPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
8
9   <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzm7KCkRr/rE9/Qpg6aAZGjwMDMVNA/GpGFF93XpGKKN" crossorigin="anonymous"></script>
10  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y10Ktv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPsvXusvfa0b4Q" crossorigin="anonymous"></script>
11  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQxSffWpi1MquVdAyjUar5+76PVcmYl" crossorigin="anonymous"></script>
12
13  <title>Cara Pakai | Skripsi</title>
14  <link href="{{asset('css/skripsi_mainstyle.css')}}" rel="stylesheet" type="text/css" >
15 </head>
16 <style>
17   body{
18     background-image: url("assets/carapakai_bg.jpg") ;
19     background-size: 100%;
20     padding-left: 99px;
21     padding-right: 108px;
22     background-repeat: no-repeat;
23     /* box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19); */
24   }
25 </style>
26 <body style="background-color: #ecececc4;">
27   @include('skripsi_header')
28
29   <h1>Aplikasi Pemantauan<br>
30     Kualitas Tanah Sawah<br>

```

```

31     Berbasis WSN
32 </h1>
33
34 <hr>
35
36 <h2 style="font-size:_14px;">
37 Skripsi II<br>
38 </h2>
39
40 <div class="carapakaihead">
41   
42 </div>
43 <div class="carapakaicontent">
44   <h1>Cara Penggunaan</h1>
45   <hr>
46   <div class="cpisi" style="margin-top:_80px;">
47     
48     <h1>Aktifkan Perangkat</h1>
49     <a>Aktifkan seluruh node sensor (perangkat keras Arduino dan Raspberry)
50       juga pastikan setiap sensor sensing pada setiap node sensor aktif
51       ,untuk melakukan pemantauan kualitas tanah sawah yang diuji.
52     </a>
53   </div>
54
55   <div class="cpisi">
56     
57     <h1>Sebar Perangkat</h1>
58     <a>Sebarakan node sensor ke berbagai titik sesuai dengan topologi yang digunakan.
59       Penyebaran node sensor dilakukan pada satu bidang petak tanah sawah.
60       Untuk node sensor Raspberry akan terhubung langsung dengan laptop.
61     </a>
62   </div>
63
64   <div class="cpisi">
65     
66     <h1>Aktifkan Internet</h1>
67     <a>Pastikan laptop yang terhubung dengan perangkat Rapsberry terhubung dengan internet.
68       Hasil sensing yang didapatkan oleh node sensor akan dikirimkan ke Raspberry lalu disimpan
69       di internet (local-host).
70     </a>
71   </div>
72
73   <div class="cpisi">
74     
75     <h1>Lakukan Sensing</h1>
76     <a>Untuk melakukan sensing, masuk ke halaman sensing pada website. Hasil sensing yang
77       diambil oleh node sensor secara otomatis akan ditampilkan pada halaman tersebut.
78       Pengguna juga dapat menghentikan sensing ataupun memulai kembali sensing pada halaman tersebut.
79     </a>
80   </div>
81 </div>
82
83 <a class="trademark">Pengembangan Aplikasi Pemantauan Kualitas Tanah Sawah Berbasis WSN</a>
84
85
86 </body>
87 </html>

```

Kode C.7: web.php

```

1 <?php
2 use App\Event\TaskEvent;
3
4 use Illuminate\Support\Facades\Route;
5
6 /*
7 |-----|
8 | Web Routes
9 |-----|
10 |
11 | Here is where you can register web routes for your application. These
12 | routes are loaded by the RouteServiceProvider within a group which
13 | contains the "web" middleware group. Now create something great!
14 |
15 */
16
17 Route::get('/', function () {
18   return view('skripsi_home');
19 });
20
21 Route::get('/cara-pakai', function () {
22   return view('skripsi_carapakai');
23 });
24
25 Route::get('/print-sensing', function () {
26   return view('skripsi_perangkat');
27 });
28
29 Route::get('/GIS', function () {
30   return view('skripsi_gis');
31 });
32
33
34 Route::get('/print-sensing', 'PrintViewController@index');
35 Route::get('/print-sensing/cari', 'PrintViewController@cari');
36 Route::get('/print-sensing/sortkode', 'PrintViewController@sortkode');
37
38 Route::get('/GIS', 'GISViewController@index');

```

```

39| Route::get('/sensing', 'SensingViewController@index');
40|
41| Route::get('/check-status', 'StatusViewController@index');
42| Route::get('/check-status/update', 'StatusViewController@update');
43| Route::put('/check-status/update', 'StatusViewController@update');
44|
45| Route::get('event', function () {
46|     event(new TaskEvent('Hey_How_Are_You'));
47| });
48|

```

Kode C.8: controller.php

```

1 <?php
2 namespace App\Http\Controllers;
3
4 use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
5 use Illuminate\Foundation\Bus\DispatchesJobs;
6 use Illuminate\Foundation\Validation\ValidatesRequests;
7 use Illuminate\Routing\Controller as BaseController;
8
9 class Controller extends BaseController
10{
11    use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
12}
13

```

Kode C.9: StatusViewController.php

```

1 <?php
2 namespace App\Http\Controllers;
3 use Illuminate\Http\Request;
4 use DB;
5 use App\Http\Requests;
6 use App\Http\Controllers\Controller;
7
8 class StatusViewController extends Controller {
9     public function index(){
10         $check1 = DB::select(
11             'SELECT_nama_node,status_node,status_sensing
12             FROM_nodesensor'
13             WHERE_kode_node=_1'
14         );
15         $check2 = DB::select(
16             'SELECT_nama_node,status_node,status_sensing
17             FROM_nodesensor'
18             WHERE_kode_node=_2'
19         );
20         $check3 = DB::select(
21             'SELECT_nama_node,status_node,status_sensing
22             FROM_nodesensor'
23             WHERE_kode_node=_3'
24         );
25         $check4 = DB::select(
26             'SELECT_nama_node,status_node,status_sensing
27             FROM_nodesensor'
28             WHERE_kode_node=_4'
29         );
30         $check5 = DB::select(
31             'SELECT_nama_node,status_node,status_sensing
32             FROM_nodesensor'
33             WHERE_kode_node=_5'
34         );
35         return view('skripsi_checkstatus',[ 'check1'=>$check1, 'check2'=>$check2, 'check3'=>$check3, 'check4'=>$check4,
36         'check5'=>$check5]);
37     }
38
39     public function update(){
40         DB::select(
41             'UPDATE
42             nodesensor
43             SET
44             status_node=_Offline_,status_sensing=_Not_Sensing"
45             WHERE
46             kode_node_!=_5');
47
48         return redirect('/check-status');
49     }
50 }
51

```

Kode C.10: PrintViewController.php

```

1 <?php
2 namespace App\Http\Controllers;
3 use Illuminate\Http\Request;
4 use DB;
5 use App\Http\Requests;
6 use App\Http\Controllers\Controller;
7
8 class PrintViewController extends Controller {
9     public function index(){
10         $stanah1 = DB::select
11

```

```

11         ('SELECT
12             *
13             FROM
14             tanah
15             ORDER_BY
16             waktu_sensing_DESC
17             LIMIT
18             30');
19
20     return view('skripsi_perangkat',[ 'tanah1'=>$tanah1]);
21 }
22
23 public function cari(Request $request)
24 {
25     // menangkap data pencarian
26     $cariAwal = $request->cariAwal;
27     $cariAkhir = $request->cariAkhir;
28
29     // mengambil data dari table sensing sesuai pencarian data
30     $tanah1 = DB::table('sensing')
31     ->where('waktu_sensing','>',$cariAwal)
32     ->where('waktu_sensing','<',$cariAkhir)
33     ->paginate();
34
35     // mengirim data pegawai ke view index
36     return view('skripsi_perangkat',[ 'tanah1' => $tanah1]);
37 }
38
39
40 public function sortkode(){
41     $tanah1 = DB::select
42         ('SELECT
43             *
44             FROM
45             sensing
46             ORDER_BY
47             kode_node_ASC,waktu_sensing_DESC
48         ');
49
50     return view('skripsi_perangkat',[ 'tanah1'=>$tanah1]);
51 }
52 }
53 }
```

Kode C.11: SensingViewController.php

```

1 <?php
2 namespace App\Http\Controllers;
3 use Illuminate\Http\Request;
4 use DB;
5 use App\Http\Requests;
6 use App\Http\Controllers\Controller;
7
8 class SensingViewController extends Controller {
9     public function index(){
10         $nodes1 = DB::select(
11             'SELECT
12                 _nodesensor.kode_node,_,waktu_sensing,_,nodesensor.nama_node,ph_tanah,kelembaban_tanah,suhu_tanah,suhu_udara
13                 FROM
14                 _nodesensor JOIN sensing ON sensing.kode_node=_nodesensor.kode_node JOIN tanah ON tanah.id_Tanah=_nodesensor.
15                 kode_node
16                 WHERE
17                 waktu_sensing IN_(SELECT
18                     max(waktu_sensing)
19                     FROM
20                     sensing
21                     WHERE
22                     kode_node=_1'
23     );
24     $nodes2 = DB::select(
25         'SELECT
26             _nodesensor.kode_node,_,waktu_sensing,_,nodesensor.nama_node,ph_tanah,kelembaban_tanah,suhu_tanah,suhu_udara
27             FROM
28             _nodesensor JOIN sensing ON sensing.kode_node=_nodesensor.kode_node JOIN tanah ON tanah.id_Tanah=_nodesensor.
29             kode_node
30             WHERE
31             waktu_sensing IN_(SELECT
32                 max(waktu_sensing)
33                 FROM
34                 sensing
35                 WHERE
36                 kode_node=_2'
37     );
38     $nodes3 = DB::select(
39         'SELECT
40             _nodesensor.kode_node,_,waktu_sensing,_,nodesensor.nama_node,ph_tanah,kelembaban_tanah,suhu_tanah,suhu_udara
41             FROM
42             _nodesensor JOIN sensing ON sensing.kode_node=_nodesensor.kode_node JOIN tanah ON tanah.id_Tanah=_nodesensor.
43             kode_node
44             WHERE
45             waktu_sensing IN_(SELECT
46                 max(waktu_sensing)
47                 FROM
48                 sensing
49                 WHERE
50                 kode_node=_3'
51     );
52     $nodes4 = DB::select(
```

```
50         'SELECT
51             nodesensor.kode_node, waktu_sensing, nodesensor.nama_node, ph_tanah, kelembaban_tanah, suhu_tanah, suhu_udara
52         FROM
53             nodesensor JOIN_sensing ON sensing.kode_node = nodesensor.kode_node JOIN_tanah ON tanah.id_Tanah = nodesensor.
54             kode_node
55             WHERE
56                 waktu_sensing IN (SELECT
57                     max(waktu_sensing)
58                 FROM
59                     sensing
60                     WHERE
61                         kode_node = 4)
62     );
63     return view('skripsi_sensing',[ 'nodes1'=>$nodes1, 'nodes2'=>$nodes2, 'nodes3'=>$nodes3, 'nodes4'=>$nodes4]);
64 }
```