

2018

Sistem Pengecekan pH Tanah Otomatis Menggunakan Sensor pH Probe Berbasis Android Dengan Algoritma Binary Search

Meivaldi, Rido

Universitas Sumatera Utara

<http://repositori.usu.ac.id/handle/123456789/5295>

Downloaded from Repositori Institusi USU, Universitas Sumatera Utara

**SISTEM PENGECEKAN PH TANAH OTOMATIS MENGGUNAKAN
SENSOR PH PROBE BERBASIS ANDROID DENGAN
ALGORITMA BINARY SEARCH**

SKRIPSI

RIDO MEIVALDI

141401132



**PROGRAM STUDI S1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA**

MEDAN

2018

SISTEM PENGECEKAN PH TANAH OTOMATIS MENGGUNAKAN SENSOR
PH PROBE BERBASIS ANDROID DENGAN
ALGORITMA BINARY SEARCH

SKRIPSI

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah

Sarjana Ilmu Komputer

RIDO MEIVALDI

141401132



PROGRAM STUDI S1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2018

PERSETUJUAN

Judul : SISTEM PENGECEKAN PH TANAH OTOMATIS
DENGAN SENSOR PH PROBE BERBASIS ANDROID
DENGAN ALGORITMA BINARY SEARCH

Kategori : SKRIPSI

Nama : RIDO MEIVALDI

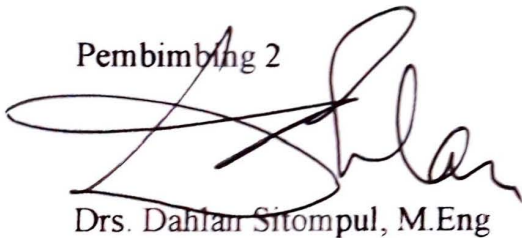
Nomor Induk Mahasiswa : 141401132

Program Studi : SARJANA (S1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

Komisi Pembimbing:


Pembimbing 2



Drs. Dahlan Sitompul, M.Eng

NIP. 196707252005011002

Pembimbing 1



Dr. Poltak Sihombing, M.Kom

NIP. 196203171991031000

Diketahui/disetujui oleh

Program Studi S1 Ilmu Komputer

Ketua



Dr. Poltak Sihombing, M. Kom

NIP. 196203171991031000

PERNYATAAN**SISTEM PENGECEKAN PH TANAH OTOMATIS DENGAN
SENSOR PH PROBE BERBASIS ANDROID DENGAN
ALGORITMA BINARY SEARCH****SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, April 2018

Rido Meivaldi

141401132

PENGHARGAAN

Puji dan syukur saya ucapkan kepada Allah SWT yang telah memberikan rahmat beserta hidayah-Nya sehingga penulis dapat menyelesaikan skripsi ini. Penulis mangat berterima kasih kepada semua pihak yang telah membantu dan mendukung sehingga skripsi ini bisa diselesaikan. Penulis ingin mengucapkan terima kasih kepada:

1. Bapak Prof. Dr. Runtung Sitepu, SH., M.Hum sebagai Rektor Universitas Sumatera Utara.
2. Bapak Prof. Dr. Opim Salim Sitompul, M.Sc sebagai Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Bapak Dr. Poltak Sihombing, M.Kom sebagai Ketua Program Studi S1 Ilmu Komputer dan sekaligus sebagai Dosen Pembimbing I yang telah memberikan banyak masukan kepada saya untuk menyelesaikan skripsi ini.
4. Bapak Drs. Dahlan Sitompul, M.Eng sebagai Dosen Pembimbing II yang telah membimbing dan memberi banyak masukan kepada penulis selama mengerjakan skripsi ini..
5. Seluruh Dosen serta staf Pegawai di Program Studi S1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
6. Orangtua penulis Ibunda dan Ayahanda yang tak kenal lelah memberikan dukungan semangat dan motivasi sehingga penulis dapat menyelesaikan skripsi ini.
7. Istika Suri, seseorang yang selalu memberikan semangat kepada penulis baik secara tersirat maupun tersurat.
8. Teman-Teman stambuk 014, khususnya kom C yang selama ini telah memberikan banyak pelajaran hidup kepada penulis, terkhusus Dicky Febriansah, Nadiya Khairani Nst, Filza Herzy , Ishan Wardhono, Karina Pramudita Kurniadi, Farhan Aulia Rangkuti, Akmad Yusuf, Fattahurahman, Fadly Randa, Hakim Tanjung, Wiro Tirta, Julio Paulus, Mutihandini, Riza Puspita dan teman - teman lainnya.
9. Abang dan kakak senior yang telah memberikan masukan kepada penulis.
10. Adik adik yang juga senantiasa mendukung dan menyemangati kepada penulis terutama Neni dkk dan Moza dkk.

11. Semua pihak yang terlibat langsung ataupun tidak langsung yang tidak dapat penulis ucapkan satu-persatu yang telah membantu penyelesaian skripsi ini.

Semoga Allah SWT memberikan berkah, kesehatan, dan keselamatan bagi semua pihak yang telah mendukung penulis untuk menyelesaikan skripsi ini. Semoga penelitian ini bermanfaat kepada seluruh orang terutama kepada penulis sendiri.

Medan, April 2018

Penulis,

Rido Meivaldi

SISTEM PENGECEKAN PH TANAH OTOMATIS MENGGUNAKAN SENSOR PH PROBE BERBASIS ANDROID DENGAN ALGORITMA BINARY SEARCH

ABSTRAK

Tanah sangat vital peranannya bagi semua kehidupan di bumi karena tanah mendukung kehidupan tumbuhan dengan menyediakan haradan air sekaligus sebagai penopang akar. Struktur tanah yang berongga-rongga juga menjadi tempat yang baik bagi akar untuk bernapas dan tumbuh. Tanah juga menjadi habitat hidup berbagai mikroorganisme. Bagi sebagian besar hewan darat, tanah menjadi lahan untuk hidup dan bergerak. Penelitian ini diharapkan memperoleh sebuah sistem yang dapat menghitung pH (*Potential of Hydrogen*) tanaman dengan mudah dan sesimpel mungkin sehingga para petani dapat dengan mudah mengoperasikannya dan hasil panen menjadi lebih optimal. Dan juga penelitian ini menggunakan maps dari google sehingga para pengguna dapat mengetahui nilai pH tanah di wilayah tertentu. Pengujian sistem dilakukan dengan menggunakan lima jenis sampel tanah, yaitu tanah humus, tanah dengan campuran pupuk kandang, pupuk npk, pupuk kompos, dan pasir. Hasil dari uji sistem yaitu sistem akan menampilkan pH tanah yang diuji pada android beserta tanaman apa saja yang cocok dengan pH tersebut, kemudian nilai tersebut dapat disimpan secara *online* di *firebase database* dimana nilai yang disimpan yaitu nilai pH beserta nama alamat dan koordinatnya. Dalam pengecekan pH tanah, sensor membutuhkan sekitar 2–5 detik pada saat proses pengoperasiannya.

Kata Kunci : Mikrokontroler, Arduino, Sensor PH Tanah, *Bluetooth*, PH, PH Cairan, *Binary Search*

AUTOMATIC PH TESTING SYSTEM USING PH PROBE SENSOR BASED ON ANDROID WITH BINARY SEARCH ALGORITHM

ABSTRACT

The soil is vital for all life on earth because the soil supports plant's life by providing nutrients and water as well as the support of the roots. The hollow ground structure is also a good place for roots to breathe and grow. Soil is also the living habitat of various microorganisms. For most land animals, land becomes a land for living and moving. This study is expected to obtain a system that can calculate the pH (*Potential of Hydrogen*) of the plant easily and asimple as possible so that farmers can easily operate it and the harvest becomes more optimal. And also this research using maps from google so that users can know the value of soil pH in certain area. The system test was conducted using five types of soil samples, namely humus soil, soil with mixture of manure, npk fertilizer, compost, and sand. The result of the system test is that the system will display the tested ground pH on the android along with any crop that matches the pH, then the value can be stored online in firebase database where the stored value is the value of pH along with the name of the address and its coordinates. In checking the soil pH, the sensor takes about 2-5 seconds during the operation process.

Keywords: Microcontroller, Arduino, Soil PH Sensor, Bluetooth, PH, Liquid PH, Binary Search

DAFTAR ISI

Halaman

PERSETUJUAN	ii
PERNYATAAN.....	iii
PENGHARGAAN.....	iv
ABSTRAK	vi
ABSTRACT.....	vii
DAFTAR ISI.....	viii
DAFTAR TABEL	xi
DAFTAR GAMBAR.....	xii

BAB 1. PENDAHULUAN

1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah	3
1.4 Tujuan Penelitian	3
1.5 Manfaat Penelitian	3
1.6 Metodologi Penelitian.....	4
1.7 Sistematika Penulisan	5

BAB 2. LANDASAN TEORI

2.1 Mikrokontroler ATmega328	7
2.2 Tanah.....	7
2.3 Arduino Uno	8
2.4 Derajat Keasaman	10
2.5 <i>Bluetooth</i> HC-05	12
2.6 Sensor PH Probe	14
2.7 Algoritma <i>Binary Search</i>	15
2.8 Sistem Operasi Android.....	16
2.9 Penelitian yang Relevan.....	17

BAB 3. ANALISIS PERANCANGAN

3.1 Analisis Sistem.....	18
3.1.1 Analisis Masalah	18
3.1.2 Analisis Kebutuhan Sistem	20
3.1.2.1 Kebutuhan Fungsional	20
3.1.2.2 Kebutuhan Non-Fungsional	21
3.1.3 Pemodelan Sistem	21
3.1.3.1 <i>Use Case Diagram</i>	22
3.1.3.2 <i>Activity Diagram</i>	25
3.1.3.3 <i>Sequence Diagram</i>	26
3.1.4 <i>Flowchart</i>	27
3.2 Blok Diagram Sistem	30
3.3 Perancangan Sistem	31
3.3.1 Perancangan Perangkat Keras	32
3.3.1.1 <i>Main Board</i> (Arduino)	32
3.3.1.2 Sensor PH Probe	33
3.3.1.3 Konektivitas	34
3.3.2 Perancangan Perangkat Lunak	35
3.3.2.1 Perancangan Antar Muka (<i>Interface</i>)	35

BAB 4. IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Sistem	39
4.1.1 Konstruksi Utama	39
4.1.2 Sensor PH Tanah	41
4.1.3 <i>Bluetooth</i> HC-05	42
4.1.4 Daya Listrik	43
4.2 Penggabungan Perangkat Keras	43
4.2.1 <i>Experimental Setup</i>	44
4.3 Pembuatan Perangkat Lunak	46
4.3.1 Perangkat Lunak Arduino Uno	46
4.3.2 Perangkat Lunak Android	46
4.3.2.1 Panduan Penggunaan	47

4.3.2.2 Menu Utama	49
4.3.2.3 Hasil Perhitungan	50
4.3.2.4 Menu Lihat <i>Map</i>	51
4.4 Pengujian Alat.....	52
4.4.1 Pengujian Sensor PH Tanah	53
4.4.2 Pengujian Pembacaan Sensor pada Sampel Tanah	54
4.4.2.1 Pengujian pada Sampel Tanah Humus	55
4.4.2.2 Pengujian pada Sampel Tanah dengan Pupuk Kompos	56
4.4.2.3 Pengujian pada Sampel Tanah dengan Pupuk NPK	58
4.4.2.4 Pengujian pada Sampel Tanah dengan Pupuk Kandang	59
4.4.2.5 Pengujian pada Sampel Pasir	62
4.4.2.6 Hasil Akhir Pengujian pada Sampel Tanah	64
4.4.3 Pengujian Pembacaan Sensor pada Sampel Cairan Tanah	65
4.4.3.1 Pengujian pada Sampel Cairan Tanah Humus	65
4.4.3.2 Pengujian pada Sampel Cairan Tanah dengan Pupuk NPK	67
4.4.3.3 Pengujian pada Sampel Cairan Tanah Kompos	69
4.4.3.4 Pengujian pada Sampel Cairan Tanah dengan Pupuk Kandang	71
4.4.3.5 Pengujian pada Sampel Cairan Pasir	73
4.4.3.6 Hasil Akhir Pengujian pada Sampel Cairan Tanah	74
4.4.4 Pengujian Pengiriman dan Penyimpanan Nilai PH	76
4.4.5 Pengujian Pengecekan Lokasi di <i>Maps</i>	77
4.5 Perhitungan Manual Algoritma <i>Binary Search</i>	80

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan	81
5.2 Saran	81

DAFTAR PUSTAKA 82

LISTING PROGRAM	A-1
CURRICULUM VITAE	B-1

DAFTAR TABEL

Tabel 2.1	Pin Sensor	14
Tabel 2.2	Karakteristik Sensor	14
Tabel 3.1	Narrative Use Case Proses Menghubungkan SmartpHone dengan Arduino	22
Tabel 3.2	Narrative Use Case Menghitung nilai yang telah dibaca oleh sensor kemudian mengirimnya ke smartpHone pengguna via Bluetooth ..	23
Tabel 4.1	Data Uji Sensor pH Tanah	53
Tabel 4.2	Data Uji Rumus Konversi ADC ke pH	54
Tabel 4.3	Percobaan pH Tanah dengan Sampel Tanah Humus	56
Tabel 4.4	Percobaan pH Tanah dengan Sampel Tanah dengan Pupuk Kompos	57
Tabel 4.5	Percobaan pH Tanah dengan Sampel Tanah dengan Pupuk NPK .	59
Tabel 4.6	Percobaan pH Tanah dengan Sampel Tanah dengan Pupuk Kandang	61
Tabel 4.7	Percobaan pH Tanah dengan Sampel Pasir	63
Tabel 4.8	Hasil Perhitungan Seluruh Sampel	64
Tabel 4.9	Percobaan pH Tanah dengan Sampel Cairan Tanah Humus	67
Tabel 4.10	Percobaan pH Tanah dengan Sampel Cairan Tanah dengan Pupuk NPK	68
Tabel 4.11	Percobaan pH Tanah dengan Sampel Cairan Tanah Kompos	70
Tabel 4.12	Percobaan pH Tanah dengan Sampel Cairan Tanah dengan Pupuk Kandang	72
Tabel 4.13	Percobaan pH Tanah dengan Sampel Cairan Pasir	74
Tabel 4.14	Hasil Perhitungan Seluruh Sampel	75

DAFTAR GAMBAR

Gambar 1.1	Diagram Penggunaan Sistem	5
Gambar 2.1	Bentuk Fisik Arduino Tampak Depan	9
Gambar 2.2	Bentuk Fisik Arduino Tampak Belakang	9
Gambar 2.3	Skala Derajat Keasaman	11
Gambar 2.4	<i>Bluetooth</i> HC-05	13
Gambar 2.5	Bentuk Fisik Sensor PH Tanah	15
Gambar 3.1	Diagram <i>Fishbone</i> Masalah Penelitian	18
Gambar 3.2	<i>Use Case</i> Sistem	21
Gambar 3.3	<i>Activity Diagram</i> Sistem	24
Gambar 3.4	<i>Sequence Diagram</i> Sistem	25
Gambar 3.5	<i>Flowchart</i> Sistem	28
Gambar 3.6	<i>Flowchart</i> Algoritma	29
Gambar 3.7	Blok Diagram Sistem	30
Gambar 3.8	Bentuk Fisik Arduino	33
Gambar 3.9	Bentuk Fisik Sensor pH Tanah	34
Gambar 3.10	Kabel <i>Jumper Male to Male</i>	35
Gambar 3.11	Rancangan <i>Layout</i> Pembuka	36
Gambar 3.12	Rancangan Tampilan Beranda	37
Gambar 3.13	Rancangan Tampilan Hasil Perhitungan	38
Gambar 4.1	Arduino Uno R3	40
Gambar 4.2	Arduino <i>Shield</i>	40
Gambar 4.3	Sensor PH Tanah	41
Gambar 4.4	<i>Bluetooth</i> HC-05	42
Gambar 4.5	<i>Battery Clip</i>	43
Gambar 4.6	Perangkat Keras Sistem	44
Gambar 4.7(a)	<i>Experimental Setup</i> Sampel Tanah	45
Gambar 4.7(b)	<i>Experimental Setup</i> Sampel Cairan Tanah	45
Gambar 4.8	<i>Source Code</i> Arduino	46
Gambar 4.9	<i>Layout</i> Pertama pada Halaman Petunjuk Penggunaan	47
Gambar 4.10	<i>Layout</i> Kedua pada Halaman Petunjuk Penggunaan	48
Gambar 4.11	<i>Layout</i> Ketiga pada Halaman Petunjuk Penggunaan	48
Gambar 4.12	<i>Layout</i> Menu Utama	49

Gambar 4.13	<i>Layout</i> Hasil Perhitungan	51
Gambar 4.14	<i>Layout</i> Menu Lihat <i>Map</i>	52
Gambar 4.15	Hasil Perhitungan pH Tanah Humus	55
Gambar 4.16	<i>Coding Snippet</i> Pembacaan Nilai pH	55
Gambar 4.17	Hasil Perhitungan pH Tanah dengan Campuran Pupuk Kompos ...	57
Gambar 4.18	Hasil Perhitungan pH Tanah dengan Campuran Pupuk NPK	58
Gambar 4.19	Hasil Perhitungan pH Tanah dengan Campuran Pupuk Kandang ..	60
Gambar 4.20	Hasil Perhitungan pH Tanah dengan Sampel Pasir	62
Gambar 4.21	Grafik Hasil Perhitungan pada Sampel Tanah	64
Gambar 4.22	Sampel Tanah	65
Gambar 4.23	Hasil Perhitungan pH Cairan Tanah Humus	66
Gambar 4.24	<i>Coding Snippet</i> Pengambilan Nilai pH	66
Gambar 4.25	Hasil Perhitungan pH Cairan Tanah dengan Pupuk NPK	68
Gambar 4.26	Hasil Perhitungan pH Cairan Tanah Kompos	70
Gambar 4.27	Hasil Perhitungan pH Cairan Tanah dengan Pupuk Kandang	72
Gambar 4.28	Hasil Perhitungan pH Cairan Pasir	73
Gambar 4.29	Grafik Hasil Perhitungan pada Sampel Cairan Tanah	75
Gambar 4.30	Sampel Cairan Tanah	75
Gambar 4.31	Hasil Pengiriman dari Arduino ke Android	76
Gambar 4.32	Database pH Tanah Beserta Lokasi	77
Gambar 4.33	Pengujian Lokasi pada Google Maps	78
Gambar 4.34	<i>Coding Snippet</i> Lokasi Pengecekan	79

BAB I

PENDAHULUAN

1.1. Latar Belakang

Tanah sangat vital peranannya bagi semua kehidupan di bumi karena tanah mendukung kehidupan tumbuhan dengan menyediakan haradan air sekaligus sebagai penopang akar. Struktur tanah yang berongga-rongga juga menjadi tempat yang baik bagi akar untuk bernapas dan tumbuh. Tanah juga menjadi habitat hidup berbagai mikroorganisme. Bagi sebagian besar hewan darat, tanah menjadi lahan untuk hidup dan bergerak.

PH adalah derajat keasaman yang digunakan untuk menyatakan tingkat keasaman atau kebasaan yang dimiliki oleh suatu larutan. Ia didefinisikan sebagai kologaritma aktivitas ion hidrogen (H^+) yang terlarut. Koefisien aktivitas ion hidrogen tidak dapat diukur secara eksperimental, sehingga nilainya didasarkan pada perhitungan teoretis. Skala pH bukanlah skala absolut. Ia bersifat relatif terhadap sekumpulan larutan standar yang pH-nya ditentukan berdasarkan persetujuan internasional.

Cara mengetahui pH tanah yang paling akurat adalah menggunakan sebuah alat pengukur pH yang disebut dengan pH meter. Namun sayangnya, banyak petani yang tidak memiliki alat ini. Mungkin karena harganya yang cukup mahal atau kurangnya pengetahuan tentang pentingnya mengetahui pH tanah. Padahal pengetahuan tentang derajat keasaman tanah sangat berperan dalam keberhasilan suatu budidaya tanaman. Tanaman tidak akan tumbuh dan berproduksi dengan maksimal jika tanah dalam kondisi asam maupun basa. Dengan mengetahui pH tanah, petani bisa menentukan skala pH yang ideal untuk pertumbuhan dan perkembangna tanaman. Sehingga kerugian dapat diminimalisir.

Agar hasil dari suatu tumbuhan optimal, maka pH tanah harus sesuai dengan tumbuhan tersebut. Misalnya pada tanaman jagung, jagung dapat tumbuh

dengan baik pada ketinggian antara 0 – 1 300 m di atas permukaan laut. Menurut Effendi (1985), tanaman jagung akan tumbuh baik pada tanah yang subur, drainase baik, suhu hangat 21-32°C, curah hujan merata sepanjang tahun, serta curah hujan bulanan sekitar 100-125 mm. Tanah yang baik untuk tanaman jagung adalah tanah dengan pH optimum 6.0-7.0.

Pada penelitian terdahulu, telah dibuat sebuah alat untuk mengukur pH, suhu, dan kelembaban pada tanah berbasis mikrokontroler ATmega 328P. Pengukuran kelembaban tanah menggunakan sensor *soil moisture*, Pengukuran pH tanah menggunakan elektroda dan pengukuran suhu tanah menggunakan sensor DS18B20. Mikrokontroler ATmega328P digunakan sebagai pengendali dan pemroses sinyal. Alat tersebut dilengkapi dengan sistem penyimpanan data menggunakan *SD Card* sehingga mudah untuk pengambilan data hasil pengukuran. Hasil penelitian menunjukkan bahwa kinerja alat ini mampu mengukur pH, kelembaban dan suhu tanah dengan error berturut-turut sebesar 2,68%, 1,53% dan 0,22%.

1.2. Rumusan Masalah

Berdasarkan latar belakang diatas, rumusan masalah dari penelitian ini adalah bagaimana caranya mengetahui pH dari beberapa sampel tanah menggunakan sensor pH berbasis android dan bagaimana mengirim data dari arduino ke android menggunakan modul *bluetooth*, dan bagaimana menyimpan data secara *real time* menggunakan google maps

1.3. Batasan Masalah

Batasan masalah penelitian sebagai berikut:

1. Medium yang digunakan adalah tanah humus, pupuk kompos, dan tanah biasa.
2. Hanya menghitung pH pada medium tanah saja.
3. Pengiriman data hanya menggunakan perangkat Bluetooth.
4. Penelitian ini menggunakan sebuah sistem minimum berupa Arduino Uno R3.
5. Menggunakan bahasa pemrograman Java berbasis Android.
6. Nilai pH tanah disimpan pada maps yang ada pada aplikasi Android.

1.4. Tujuan Penelitian

Tujuan penelitian ini adalah:

1. Menghitung pH tanah yang digunakan untuk menanam semua tanaman yang mencakup dalam bidang pertanian.
2. Mengirimkan nilai pH ke *device* android dengan menggunakan modul *Bluetooth*.
3. Memberikan informasi kepada pengguna melalui *maps* yang ada pada aplikasi android tentang pH tanah di beberapa wilayah.

1.5. Manfaat Penelitian

Penelitian ini diharapkan memperoleh sebuah sistem yang dapat menghitung pH tanaman dengan mudah dan sesimpel mungkin sehingga para petani dapat dengan mudah mengoperasikannya dan hasil panen menjadi lebih optimal. Dan juga penelitian ini menggunakan *maps* dari *google* sehingga para pengguna dapat mengetahui nilai pH tanah di wilayah tertentu.

1.6. Metodologi Penelitian

Metode penelitian yang dilakukan dalam penelitian ini adalah:

1. Studi Pustaka

Tahap ini penelitian dimulai dengan cara mencari referensi dari berbagai sumber yang terpercaya dan melakukan peninjauan pustaka melalui buku-buku, artikel ilmiah, dan penelitian-penelitian lainnya dalam bentuk jurnal yang berhubungan dengan Arduino dan pH tanah.

2. Analisa dan Perancangan

Pada tahap ini, dilakukan Analisa apa saja yang diperlukan dalam penelitian sehingga dapat dirancang diagram alir (*flowchart*).

3. Implementasi

Pada tahap ini, perancangan sistem dibuat menggunakan aplikasi berbasis android dengan melihat diagram alir yang telah dibuat.

4. Pengujian

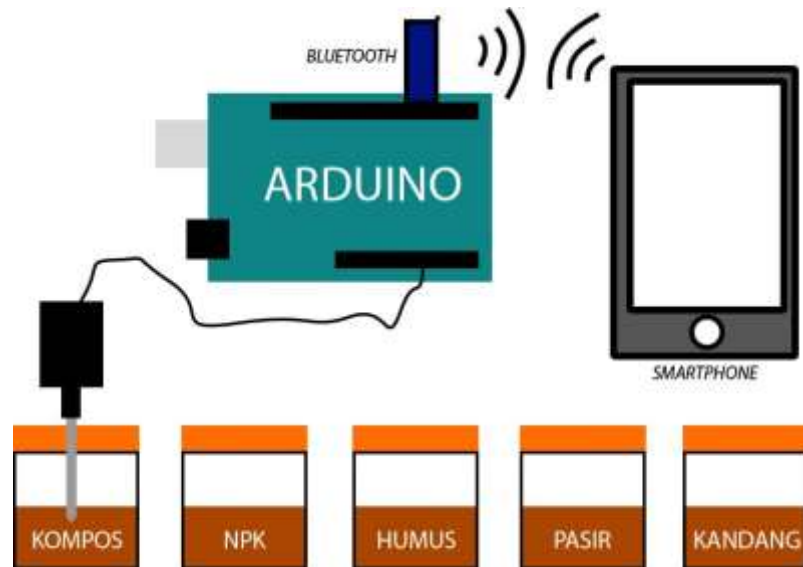
Pada tahap ini, *prototype* sistem yang telah dirancang dilakukan uji coba menggunakan medium tanah.

5. Dokumentasi

Pada tahap ini, penelitian yang telah dilakukan, didokumentasikan mulai dari tahap analisa sampai kepada pengujian dalam bentuk skripsi.

6. Skema sistem

Skema dari gambaran umum sistem dapat digambarkan pada gambar 1.1 :



Gambar 1.1: Diagram Penggunaan Sistem

1.7. Sistematika Penulisan

Sistematika penulisan skripsi ini terdiri dari beberapa bagian utama, yang terdiri dari beberapa bab-bab berikut:

BAB 1 PENDAHULUAN

Bab ini berisi latar belakang dari penelitian yang akan dilakukan yang berjudul “Sistem Pengecekan pH Tanah Otomatis menggunakan Sensor *PH Probe* dengan Arduino berbasis Android”, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian dan sistematika penulisan.

BAB 2 LANDASAN TEORI

Bab ini berisi penjelasan secara umum mengenai mikrokontroller, tanah, arduino uno, derajat keasaman (pH), modul *Bluetooth*, sensor pH *probe*, sistem operasi android, dan penelitian yang relevan dan beberapa teori yang mendukung dalam penelitian.

BAB 3 ANALISIS DAN PERANCANGAN

Bab ini berisi analisis terhadap masalah penelitian dan perancangan sistem yang akan dibangun sebagai solusi permasalahan tersebut.

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi implementasi dari pengecekan pH tanah menggunakan sensor pH tanah dengan arduino, selanjutnya pengujian terhadap sistem yang telah dibangun dengan beberapa jenis tanah serta pembahasan dan hasil pengujiannya.

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dari uraian setiap bab sebelumnya dan saran berdasarkan hasil pengujian agar bermanfaat bagi penelitian kedepannya.

BAB II

LANDASAN TEORI

2.1. Mikrokontroller ATmega328

Mikrokontroler adalah sebuah sistem komputer fungsional dalam sebuah chip. Di dalamnya terkandung sebuah inti prosesor, memori (sejumlah kecil RAM, memori program, atau keduanya), dan perlengkapan input output. Dengan kata lain, mikrokontroler adalah suatu alat elektronika digital yang mempunyai masukan dan keluaran serta kendali dengan program yang bisa ditulis dan dihapus dengan cara khusus. (Amos, 1996).

Mikrokontroler merupakan komputer didalam chip yang digunakan untuk mengontrol peralatan elektronik, yang menekankan efisiensi dan efektifitas biaya. Secara garis besar bisa disebut “pengendali kecil” dimana sebuah system elektronik yang sebelumnya banyak memerlukan komponen-komponen pendukung seperti IC TTL dan CMOS dapat direduksi/diperkecil dan akhirnya terpusat serta dikendalikan oleh mikrokontroler ini. (Amos, 1996).

2.2. Tanah

Tanah sangat vital peranannya bagi semua kehidupan di bumi karena tanah mendukung kehidupan tumbuhan dengan menyediakan hara dan air sekaligus sebagai penopang akar. Struktur tanah yang berongga-rongga juga menjadi tempat yang baik bagi akar untuk bernapas dan tumbuhan. Tanah juga menjadi habitat hidup berbagai mikroorganisme. Bagi sebagian hewan darat, tanah menjadi lahan untuk hidup dan bergerak. Ilmu yang mempelajari berbagai aspek mengenai tanah dikenal sebagai ilmu tanah. Dari segi klimatologi, tanah memegang peranan

penting sebagai penyimpan air dan menekan erosi, meskipun tanah sendiri juga dapat erosi. (Gultom, 2002).

Komposisi tanah yang berbeda-beda pada satu lokasi dengan lokasi yang lain. Air dan udara merupakan bagian dari tanah. Pedologi Tanah berasal dari pelapukan batuan dengan bantuan organism, membentuk tubuh unik yang menutupi batuan. Proses pembentukan tanah dikenal sebagai “pedogenesis”. Proses yang unik ini membentuk tanah sebagai tubuh alam yang terdiri atas lapisan-lapisan atau disebut sebagai horizon tanah. Setiap horizon menceritakan mengenai asal dan proses-proses fisika, kimia, dan biologi yang telah dilalui tubuh tanah tersebut. (Gultom, 2002).

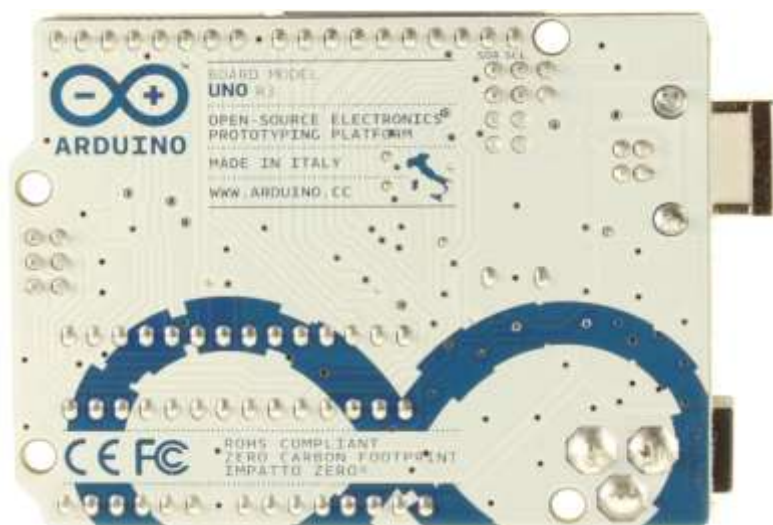
2.3. Arduino Uno

Arduino Uno adalah salah satu kit mikrokontroler yang berbasis mikrokontroller ATmega328. Modul ini sudah dilengkapi dengan berbagai hal yang dibutuhkan untuk mendukung mikrokontroler untuk bekerja, hanya sambungkan ke power suply atau sambungkan melalui kabel USB ke PC Arduino Uno ini sudah siap sedia. Arduino Uno ini memiliki 14 pin *digital input/output*, 6 *analog input*, sebuah resonator keramik 16MHz, koneksi USB, colokan *power input*, ICSP header, dan sebuah tombol *reset*. (Andrianto, et al. 2016).

Arduino memiliki kelebihan tersendiri dibanding *board* mikrokontroler yang lain selain bersifat *open source*, arduino juga mempunyai bahasa pemrogramanya sendiri yang berupa bahasa C. Selain itu dalam *board* arduino sendiri sudah terdapat *loader* yang berupa USB sehingga memudahkan kita ketika kita memprogram mikrokontroler didalam arduino. Sedangkan pada kebanyakan *board* mikrokontroler yang lain yang masih membutuhkan rangkaian *loader* terpisah untuk memasukkan program ketika kita memprogram mikrokontroler. Port USB tersebut selain untuk *loader* ketika memprogram, bisa juga difungsikan sebagai *port* komunikasi serial. (Andrianto, et al. 2016).



Gambar 2.1. Bentuk Fisik Arduino Tampak Depan
(Sumber: Sumber: <http://arduino.org>)



Gambar 2.2. Bentuk Fisik Arduino Tampak Belakang
(Sumber: Sumber: <http://arduino.org>)

Arduino menyediakan 20 pin I/O, yang terdiri dari 6 pin *input analog* dan 14 pin *digital input/output*. Untuk 6 pin analog sendiri bisa juga difungsikan sebagai *output digital* jika diperlukan *output digital* tambahan selain 14 pin yang sudah tersedia. Untuk mengubah pin *analog*

menjadi *digital* cukup mengubah konfigurasi pin pada program. Dalam *board* kita bisa lihat pin *digital* diberi keterangan 0-13, jadi untuk menggunakan pin *analog* menjadi *output digital*, pin *analog* yang pada keterangan *board* 0-5 kita ubah menjadi pin 14-19. dengan kata lain pin *analog* 0-5 berfungsi juga sebagai pin *output digital* 14-16. (Andrianto, et al. 2016).

Sifat *open source* arduino juga banyak memberikan keuntungan tersendiri untuk kita dalam menggunakan *board* ini, karena dengan sifat *open source* komponen yang kita pakai tidak hanya tergantung pada satu merek, namun memungkinkan kita bisa memakai semua komponen yang ada dipasaran. (Andrianto, et al. 2016).

Bahasa pemrograman arduino merupakan bahasa C yang sudah disederhanakan sintaks bahasa pemrogramannya sehingga mempermudah kita dalam mempelajari dan mendalami mikrokontroller. (Andrianto, et al. 2016).

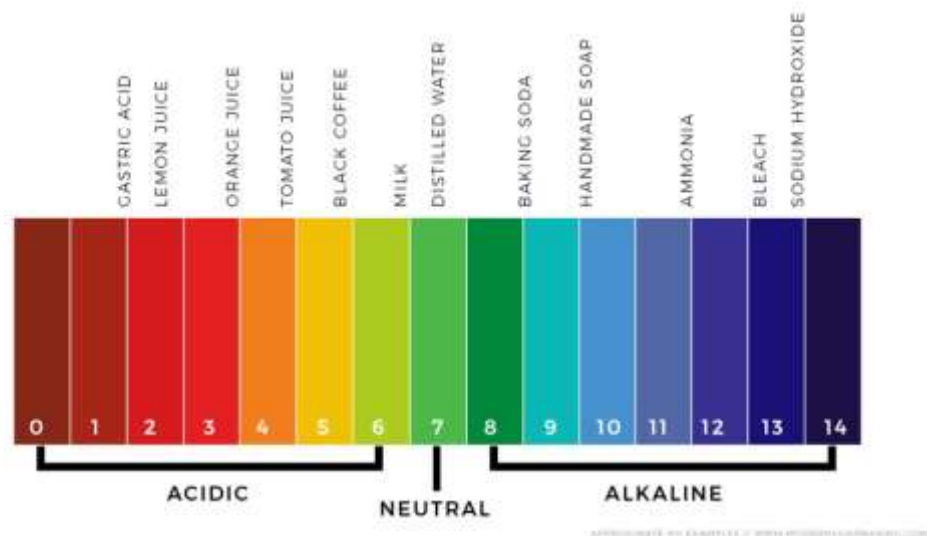
2.4. Derajat Keasaman (pH)

pH atau singkatan dari *potential of Hydrogen* merupakan derajat keasaman yang digunakan untuk menyatakan tingkat keasaman atau kebasaan yang dimiliki oleh suatu larutan. PH didefinisikan sebagai *kologaritmaaktivitas* hidrogen (H^+) yang terlarut. Koefisien aktivitas ion hidrogen tidak dapat diukur secara eksperimental, sehingga nilainya didasarkan pada perhitungan teoritis. Skala pH bukanlah skala absolut. Ia bersifat relatif terhadap sekumpulan larutan standar yang pH-nya ditentukan berdasarkan persetujuan internasional. (Lilisma, 2001).

Konsep pH pertama kali diperkenalkan oleh kimiawan Denmark Søren Peder Lauritz Sørensen pada tahun 1909. Tidaklah diketahui dengan pasti makna singkatan "p" pada "pH". Beberapa rujukan mengisyaratkan bahwa p berasal dari singkatan untuk power (pangkat), yang lainnya merujuk kata bahasa Jerman Potenz (yang juga berarti pangkat), dan ada pula yang merujuk pada kata potential. (Jens Norby,

2000) berargumen bahwa p adalah sebuah tetapan yang berarti "logaritma negatif". (Lilisma, 2001).

Air murni bersifat netral, dengan pH-nya pada suhu 25 °C ditetapkan sebagai 7,0. Larutan dengan pH kurang daripada tujuh disebut bersifat asam, dan larutan dengan pH lebih daripada tujuh dikatakan bersifat basa atau alkali. Pengukuran pH sangatlah penting dalam bidang yang terkait dengan kehidupan atau industri pengolahan kimia seperti kimia, biologi, kedokteran, pertanian, ilmu pangan, rekayasa (keteknikan), dan oseanografi. Tentu saja bidang-bidang sains dan teknologi lainnya juga memakai meskipun dalam frekuensi yang lebih rendah. (Lilisma, 2001).



Gambar 2.3. Skala Derajat Keasaman (pH)
(Sumber: <https://www.modernsoapmaking.com>)

PH adalah tingkat keasaman atau kebasaan suatu benda yang diukur dengan menggunakan skala pH antara 0 hingga 14. Sifat asam mempunyai pH antara 0 hingga 7 dan sifat basa mempunyai nilai pH 7 hingga 14. (Lilisma, 2001).

Jus jeruk dan air aki mempunyai pH antara 0 hingga 7, sedangkan air laut dan cairan pemutih mempunyai sifat basa (yang juga disebut sebagai alkaline) dengan nilai pH 7 – 14. Air murni adalah netral atau mempunyai nilai pH 7. Di dalam air minum PH meter adalah suatu alat yang digunakan untuk mengukur tingkat keasaman dan kebasaan. Keasaman dalam larutan itu dinyatakan sebagai kadar ion hidrogen disingkat dengan $[H^+]$, atau sebagai pH yang artinya $-\log [H^+]$. Dengan kata lain pH merupakan ukuran kekuatan suatu asam. pH suatu larutan dapat ditera dengan beberapa cara antara lain dengan jalan menitrasi larutan dengan asam dengan indikator atau yang lebih teliti lagi dengan pH meter. PH air disebut asam bila kurang dari 7, PH air disebut basa (*alkaline*) bila lebih dari 7 dan PH air disebut netral bila pH sama dengan 7. PH air minum ideal menurut standar Departemen Kesehatan RI adalah berkisar antara 6,5 sampai 8,5. (Lilisma, 2001).

2.5. **Bluetooth HC-05**

Bluetooth adalah protokol komunikasi *wireless* yang bekerja pada frekuensi radio 2.4 GHz untuk pertukaran data pada perangkat bergerak seperti pada, laptop, HP, dan lain-lain. Salah satu hasil contoh modul *Bluetooth* yang paling banyak digunakan adalah tipe HC-05. Modul *Bluetooth* HC-05 merupakan modul *Bluetooth* yang bisa menjadi *slave* ataupun *master*, hal ini dibuktikan dengan bisa memberikan notifikasi untuk melakukan *pairing* keperangkat lain, maupun perangkat lain tersebut yang melakukan *pairing* ke module *Bluetooth* HC-05. Untuk mengeset perangkat *Bluetooth* dibutuhkan perintah-perintah AT *Command* yang mana perintah AT *Command* tersebut akan di respon oleh perangkat *Bluetooth* jika modul *Bluetooth* tidak dalam keadaan terkoneksi dengan perangkat lain. (Tindaon, 2017).



Gambar 2.4. Bluetooth HC-05

(Sumber: <http://artofcircuits.com/product/hc-05-bluetooth-serial-pass-through-master-slave-module>)

Seperti dijelaskan di atas, modul HC-05 memiliki dua mode kerja yaitu *mode AT Command* dan *mode Data*. Modul HC-05 menggunakan *mode Data* secara default. Berikut ini adalah keterangan untuk kedua *mode* tersebut:

1. *AT Command*. Pada *mode* ini, modul HC-05 akan menerima instruksi berupa perintah *AT Command*. *Mode* ini dapat digunakan untuk mengatur konfigurasi modul HC05 .Perintah *AT Command* yang dikirimkan ke modul HC-05 menggunakan huruf kapital dan diakhiri dengan karakter CRLF (atau 0x0d 0x0a dalam heksadesimal).
2. *Mode Data*. Pada *mode* ini, modul HC-05 dapat terhubung dengan perangkat *bluetooth* lain dan mengirimkan serta menerima data melalui pin TX dan RX. Konfigurasi koneksi *serial* pada *mode* ini menggunakan *baudrate*: 9600 bps, data: 8 bit, stop bits: 1 bit, *parity*: None, handshake: None. Adapun *password default* untuk terhubung dengan modul HC-05 pada *mode Data* adalah 0000 atau 1234.

2.6. Sensor pH *Probe*

Sensor pH Tanah merupakan sensor pendeteksi tingkat keasaman (*acid*) atau kebasaan (alkali) tanah. Skala pH yang dapat diukur oleh sensor pH Tanah ini memiliki *range* 3.5 hingga 8. Sensor ini dapat langsung disambungkan dengan pin *analog* Arduino maupun pin *analog* mikrokontroller lainnya, tanpa harus memakai modul penguat tambahan.

Tabel 2.1. Pin Sensor

PIN	Warna Kabel	Deskripsi
Output	Hitam	Output ke pin A0 arduino
Gnd	Putih	GND arduino

Tabel 2.2. Karakteristik Sensor

Parameter	Simbol	Min	Max	Units
Tegangan Masukan	Vcc	3.0	4.7	V
Tegangan Keluaran	Δ Volt	4	45	ADC
Respon Waktu	T	0.1	0.3	S
Sensivitas	Vcc	0.036	0.234	V



Gambar 2.6. Bentuk Fisik Sensor pH Tanah
(Sumber: Datasheet Sensor pH Tanah)

2.7. Algoritma *Binary Search*

Binary Search adalah sebuah algoritma pencarian yang digunakan untuk mencari data tertentu pada sebuah *array* dimana pada langkah nya *array* tersebut dihilangkan setengah pada setiap langkahnya, dengan syarat *array* tersebut telah diurutkan baik dari yang terkecil ke terbesar maupun sebaliknya. Algoritma ini mencari nilai tengah (median), melakukan sebuah perbandingan untuk menentukan apakah nilai yang dicari ada sebelum atau sesudahnya, kemudian mencari setengah sisanya dengan cara yang sama. Sebuah pencarian biner adalah salah satu contoh dari algoritma *divide and conquer*. (Honggo, 2009).

Dibawah ini merupakan *pseudocode* dari algoritma *binary search*:

```

function binarySearch(a, value, left, right)
  if right < left
    return not found
  mid := floor((right-left)/2)+left
  if a[mid] = value
    return mid
  if value < a[mid]
    return binarySearch(a, value, left, mid-1)
  else
    return binarySearch(a, value, mid+1, right)

```

Contoh :

Terdapat sebuah *array* yang berisikan angka 2, 7, 8, 15, 22, 23, 37, 50, 52, 81, 100. Kita ingin mencari angka 22, maka langkah-langkah dari algoritma *binary search* adalah sebagai berikut :

1. Ambil nilai tengah dari *array* tersebut. Nilai tengah dari *array* diatas adalah 23. Kemudian pecah *array* tersebut menjadi dua sub-*array*, dimana sub-*array* yang pertama berisi nilai yang lebih kecil dari 23, sedangkan sub-*array* yang kedua berisi nilai yang lebih besar dari 23.
2. Kemudian bandingkan nilai yang dicari dengan nilai tengahnya, jika lebih kecil dari 23 maka pencarian dilakukan di sub-*array* nilainya lebih kecil dari 23, yaitu sub-*array* pertama.
3. Kemudian ulangi langkah 1-2 diatas sampai panjang sub-*array* tersebut sebesar 1 atau indeks awal dan akhir dari sub-*array* tersebut sama. Jika nilai yang dicari terdapat pada sub-*array* tersebut, maka pencarian berhasil, jika tidak maka pencarian gagal.

2.8. Sistem Operasi Android

Android adalah sistem operasi berbasis Linux untuk telepon seluler seperti *smartphone* dan komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh macam-macam perangkat. (Fitri, 2014).

2.9. Penelitian yang Relevan

Beberapa penelitian yang relevan dengan penelitian yang akan dilakukan oleh penulis adalah sebagai berikut:

1. Alat ukur keasaman dan kekeruhan air menggunakan Arduino dapat bekerja dengan tegangan supply 9 volt dan tetap relatif akurat. Alat yang dibuat ini mempunyai batas pengukuran kekeruhan air antara 0 – 20 NTU serta tingkat keasaman air (pH) 0 – 14(Akip,2016).
2. Mikrokontroller AT89S51 dapat digunakan sebagai kontroler proporsional pada pengaturan pH pada akuarium yang berisi air dengan volume 5 liter, dengan cara menginjeksi larutan AsamBasa pada setiap perubahan PH yang terjadi. Stabilitas pH dipertahankan pada nilai 8.0(Lazuardi,2011).
3. Arduino uno dapat mengendalikan kadar keasaman hidroponik stroberi, namun kemampuan sensor pH untuk menerima data cukup lambat disebabkan perubahan pH dalam air membutuhkan waktu pencampuran(Kustanti,2014).
4. Dalam penelitian penghitungan suhu, kelembapan, dan pH tanah dapat disimpulkan bahwa alat ukur suhu, kelembapan dan pH yang terintegrasi dalam satu alat dapat dibuat menggunakan sensor suhu DS18B20, sensor *soil moisture* YL-69, elektroda dan Arduino Uno. Alat ini memiliki *error* sebesar 0,22% 1,58% dan 2,68% berturut-turut untuk suhu, kelembapan dan pH tanah.(Jupri dkk,2017).

BAB III

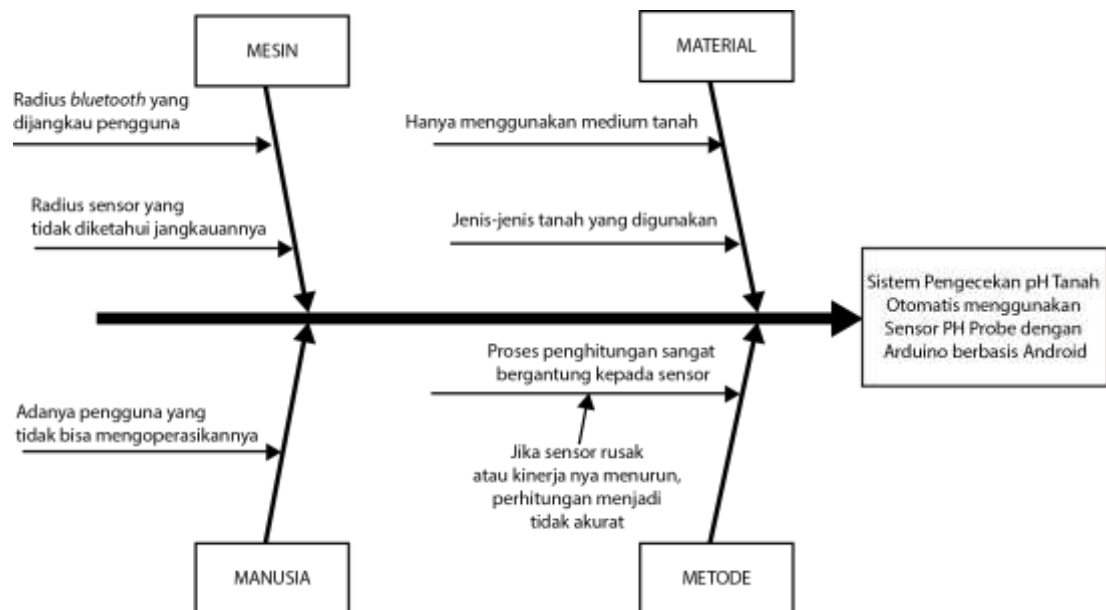
ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Sistem

Dalam pengerjaan suatu sistem, hal yang harus dimulai pertama sekali ialah analisis. Di dalam tahap analisis ini akan dilakukan identifikasi masalah, sehingga diketahui apa-apa saja hambatan yang akan terjadi beserta penanggulangannya. Setelah melakukan identifikasi masalah, tahap selanjutnya adalah analisis kebutuhan sistem, dimana pada tahap ini akan dianalisis apa-apa saja yang dibutuhkan pada sistem. Kemudian tahap terakhir adalah pembuatan model dan spesifikasi sistem agar diketahui gambaran yang akan dilakukan kedepannya.

3.1.1. Analisis Masalah

Didalam tahap ini akan dilakukan proses penentuan masalah apa yang akan terjadi, penyebab masalah itu terjadi, dan solusi untuk mengatasinya. Adapun pada tahapan analisis masalah dapat digambarkan dengan menggunakan Diagram Ishikawa atau biasa disebut dengan Diagram *fishbone*. Diagram *fishbone* pada masalah dalam penelitian penulis dapat dilihat pada gambar dibawah ini:



Gambar 3.1. Diagram *Fishbone* Masalah Penelitian

Pada gambar diagram *fishbone* diatas, dijelaskan masalah utama yang terdapat pada penelitian penulis yang akan diselesaikan. Diagram *fishbone* sendiri pada umumnya terbagi menjadi dua, yaitu bagian *head* dan bagian *bone*. Bagian *head* merupakan masalah yang terjadi pada penelitian. Sedangkan *bone* adalah penyebab masalah tersebut, yang terdiri dari 4 kategori yaitu manusia, metode, mesin dan material. Didalam 4 kategori ini juga mempunyai sub kategori yang ditunjukkan oleh anak panah yang mengarah ke masing-masing kategori.

3.1.2. Analisis Kebutuhan Sistem

Setelah dilakukan analisis masalah, tahap selanjutnya adalah analisis kebutuhan sistem. Pada tahap ini akan dilakukan analisis berupa apa-apa saja yang diperlukan sistem ini agar tujuan sistem tersebut tercapai. Analisis kebutuhan sistem dibagi menjadi dua kategori yang umum, yaitu analisis kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional yaitu kebutuhan yang harus dipenuhi agar tujuan dari sistem tercapai, sedangkan kebutuhan non-fungsional adalah kebutuhan yang dapat membuat kinerja sistem menjadi lebih baik.

3.1.2.1. Kebutuhan Fungsional

Supaya proses penghitungan pH tanah dengan sensor pH *probe* ini berjalan dengan lancar, kebutuhan fungsional yang harus dipenuhi adalah sebagai berikut:

1. Sistem harus memiliki sensor pH *probe* yang bertujuan untuk membaca nilai analog yang berupa pH tanah menjadi nilai digital agar dapat diproses oleh Arduino.
2. Sistem harus memiliki sebuah *board* Arduino dimana komponen ini bertugas sebagai unit pemrosesan inti yang melakukan pembacaan, perhitungan, dan sebagainya.
3. Sistem harus memiliki sebuah modul *Bluetooth* agar nilai yang sudah dihitung oleh Arduino dapat dikirim ke *device* lain.
4. Sistem harus memiliki sebuah *smartphone* agar nilai yang dikirim oleh Arduino melalui *Bluetooth* dapat dibaca oleh pengguna.

3.1.2.2. Kebutuhan Non-Fungsional

Adapun kebutuhan non-fungsional dari sistem ini agar kinerjanya menjadi lebih baik adalah sebagai berikut:

1. *User friendly*

Agar sistem dapat dengan mudah dioperasikan oleh para pengguna, maka *user interface* sistem haruslah sesederhana mungkin.

2. Kualitas

Sistem memiliki output yang bervariasi, tidak hanya berupa sebuah bilangan saja.

3. Efektif dan Efisien

Sistem yang dibangun haruslah cepat dalam proses pengeksekusiannya karena nilainya sudah diberikan oleh sensor.

4. Dokumentasi

Sistem yang dibangun memiliki petunjuk penggunaan sistem.

5. Kinerja

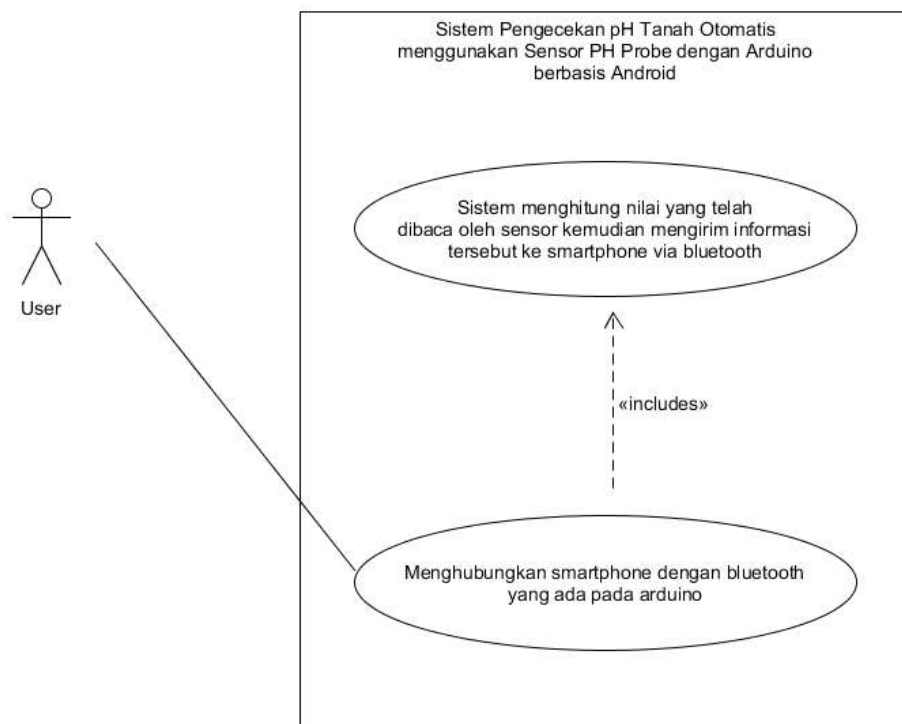
Sistem dapat melakukan perhitungan nilai pH tanah berdasarkan input yang diberikan oleh sensor dan menampilkan hasilnya beserta saran dari tanaman yang cocok untuk ditanam.

3.1.3. Pemodelan Sistem

Pemodelan sistem merupakan sebuah proses yang bertujuan untuk merancang sistem agar didapatkan gambaran umum beserta fungsi dan tujuan utama dari sistem yang akan dibangun. Pemodelan sistem ini meliputi *Use Case Diagram*, *Activity Diagram* dan *Sequence Diagram*.

3.1.3.1. Use Case Diagram

Use Case Diagram adalah sebuah diagram yang menggambarkan apa-apa saja yang dapat dilakukan oleh *user* beserta interaksi antara *user* dengan sistem. *Use Case diagram* dari sistem yang akan dibangun dapat dilihat pada gambar dibawah ini:



Gambar 3.2. Use Case Sistem

Berdasarkan diagram *Use case* diatas, *user* akan menghubungkan perangkat *smartphone* nya dengan sistem menggunakan *Bluetooth*, kemudian sistem akan melakukan proses penghitungan sesuai dengan nilai yang diberikan oleh sensor. Adapun penjelasan lebih singkat dapat dilihat pada *narrative use case* dibawah ini :

Tabel 3.1. Narrative Use Case Proses Menghubungkan Smartphone dengan Arduino

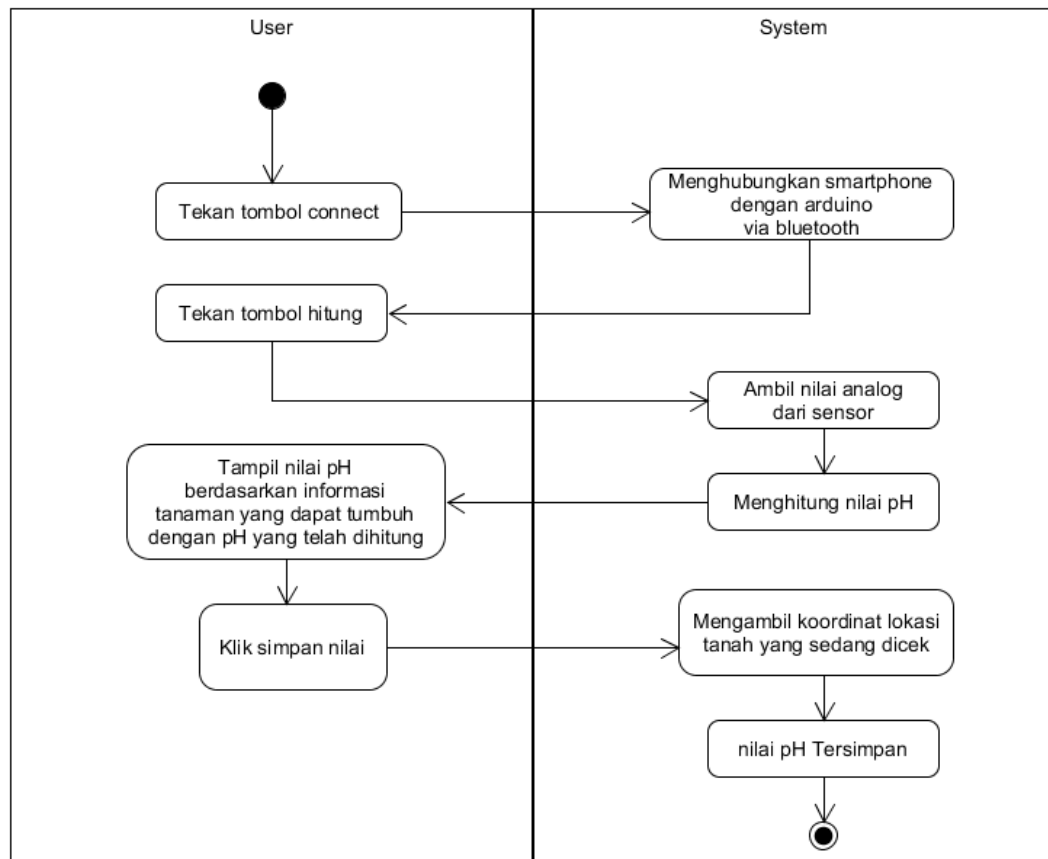
<i>Nama Use Case</i>	Menghubungkan <i>smartphone</i> dengan <i>Bluetooth</i> pada arduino	
<i>Actor</i>	Petani	
<i>Description</i>	<i>Use case</i> ini menghubungkan <i>smartphone</i> dengan Arduino via <i>Bluetooth</i>	
<i>Pre-Condition</i>	<i>Smartphone</i> sudah <i>dipairing</i> dengan <i>Bluetooth</i> pada arduino	
<i>Typical course of Event</i>	Kegiatan Pengguna	Respon Sistem
	1. Menekan tombol <i>connect</i>	2. Mengambil <i>address smartphone</i> dan membuat <i>socket</i> antara <i>smartphone</i> dengan <i>Bluetooth</i> .
<i>Alternate Course</i>	Kegiatan Pengguna	Respon Sistem
<i>Post-Condition</i>	Sistem terhubung dengan <i>smartphone</i> pengguna	

Tabel 3.2. Narrative Use Case Menghitung nilai yang telah dibaca oleh sensor kemudian mengirimnya ke *smartphone* pengguna via *Bluetooth*

<i>Nama Use Case</i>	Menghitung nilai yang telah dibaca oleh sensor kemudian mengirimnya ke <i>smartphone</i> pengguna via <i>Bluetooth</i> .	
<i>Actor</i>	Petani	
<i>Description</i>	<i>Use case</i> ini menghitung nilai dari sensor dan mengirim hasilnya ke <i>smartphone</i> melalui <i>Bluetooth</i> .	
<i>Pre-Condition</i>	Sensor sudah memberikan nilai yang dibacanya	
<i>Typical course of Event</i>	Kegiatan Pengguna	Respon Sistem
	1. Menekan tombol hitung	2. Mengambil nilai yang diberikan sensor, menghitungnya, mengirimnya, kemudian ditampilkan di layar <i>smartphone</i> pengguna.
<i>Alternate Course</i>	Kegiatan Pengguna	Respon Sistem
<i>Post-Condition</i>	Sistem menampilkan hasil perhitungan pH tanah beserta informasi lainnya.	

3.1.3.2. Activity Diagram

Activity diagram adalah sebuah diagram yang menggambarkan alur aktifitas antara *user* dengan sistem. Didalam diagram ini dijelaskan proses kerja sistem dari awal sampai akhir terhadap aktifitas yang dilakukan oleh pengguna. *Activity diagram* dari sistem yang akan dibangun dapat dilihat pada gambar dibawah ini :



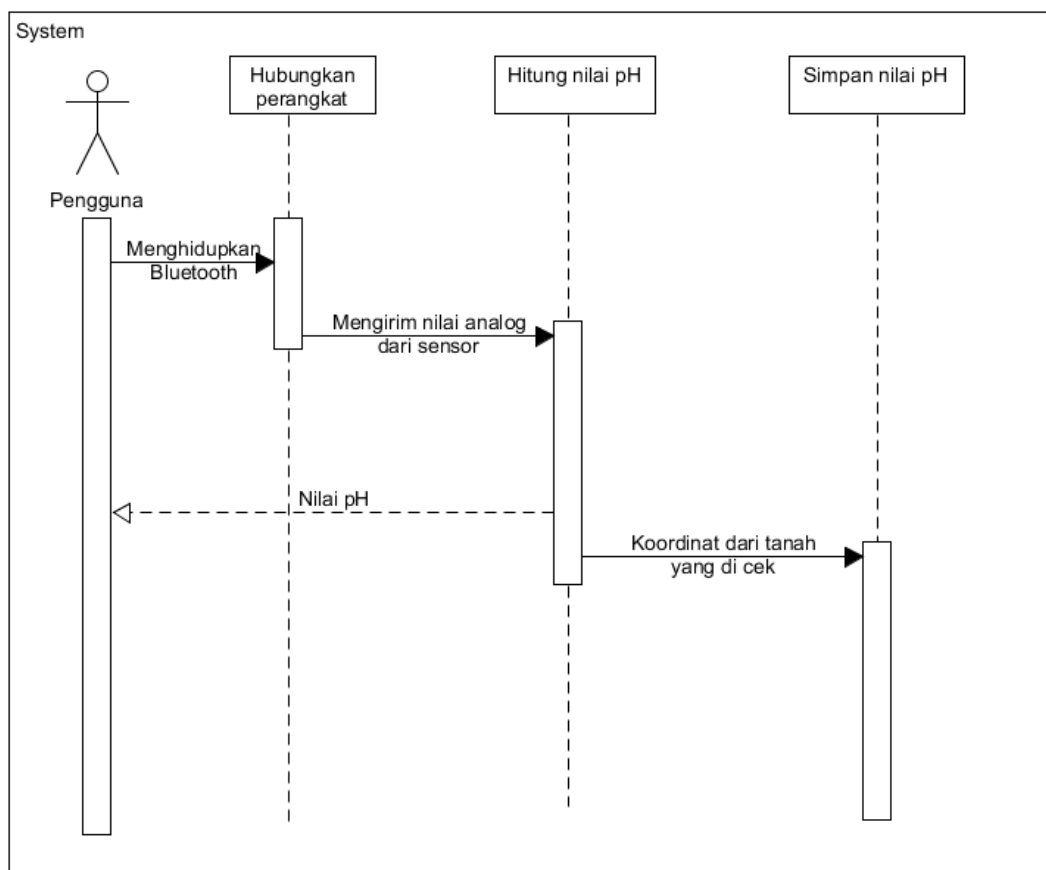
Gambar 3.3. Activity Diagram Sistem

Pada gambar 3.2 diatas, dijelaskan bahwa sistem dimulai dengan aktifitas *user* mengklik tombol *connect* untuk menghubungkan antara *smartphone user* dengan Arduino. Setelah terhubung maka selanjutnya *user* mengklik tombol hitung, kemudian sistem akan mengambil nilai yang

dibaca oleh sensor untuk diproses menjadi nilai pH tanah. Kemudian nilai tersebut ditampilkan di layar *smartphone*. Kemudian *user* mengklik tombol simpan nilai untuk menyimpan nilai pH tanah di *google maps* dengan lokasi yang sesuai dengan lokasi pengecekan tanah tersebut.

3.1.3.3. Sequence Diagram

Sequence diagram merupakan diagram yang menggambarkan interaksi antara pengguna dengan objek yang terkait pada sistem. *Sequence diagram* dari penelitian ini dapat dilihat pada gambar dibawah ini:

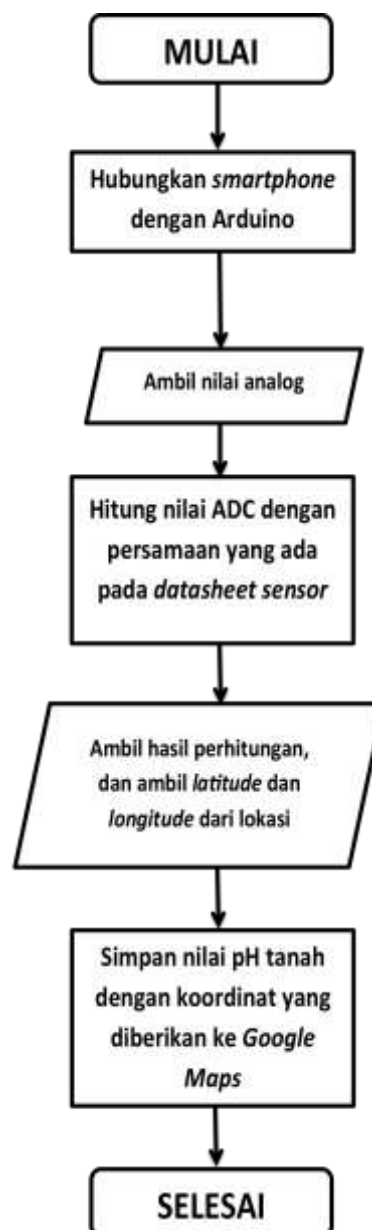


Gambar 3.4. Sequence Diagram Sistem

Seperti yang digambarkan *Sequence Diagram* sistem diatas,terdiri dari beberapa aktifitas yaitu menghubungkan perangkat, menghitung nilai pH dan menyimpan nilai pH. Pengguna pertama-tama menghidupkan *Bluetooth* kemudian perangkat *smartphone* dan sistem akan saling terhubung. Kemudian Arduino mengambil nilai dari sensor yang berupa besaran *analog* untuk dihitung nilai pH nya. Kemudian sistem memberikan nilai pH tersebut ke pengguna. Kemudian sistem akan menyimpan nilai pH tersebut di *maps* dengan koordinat yang sesuai dengan lokasi pengecekan.

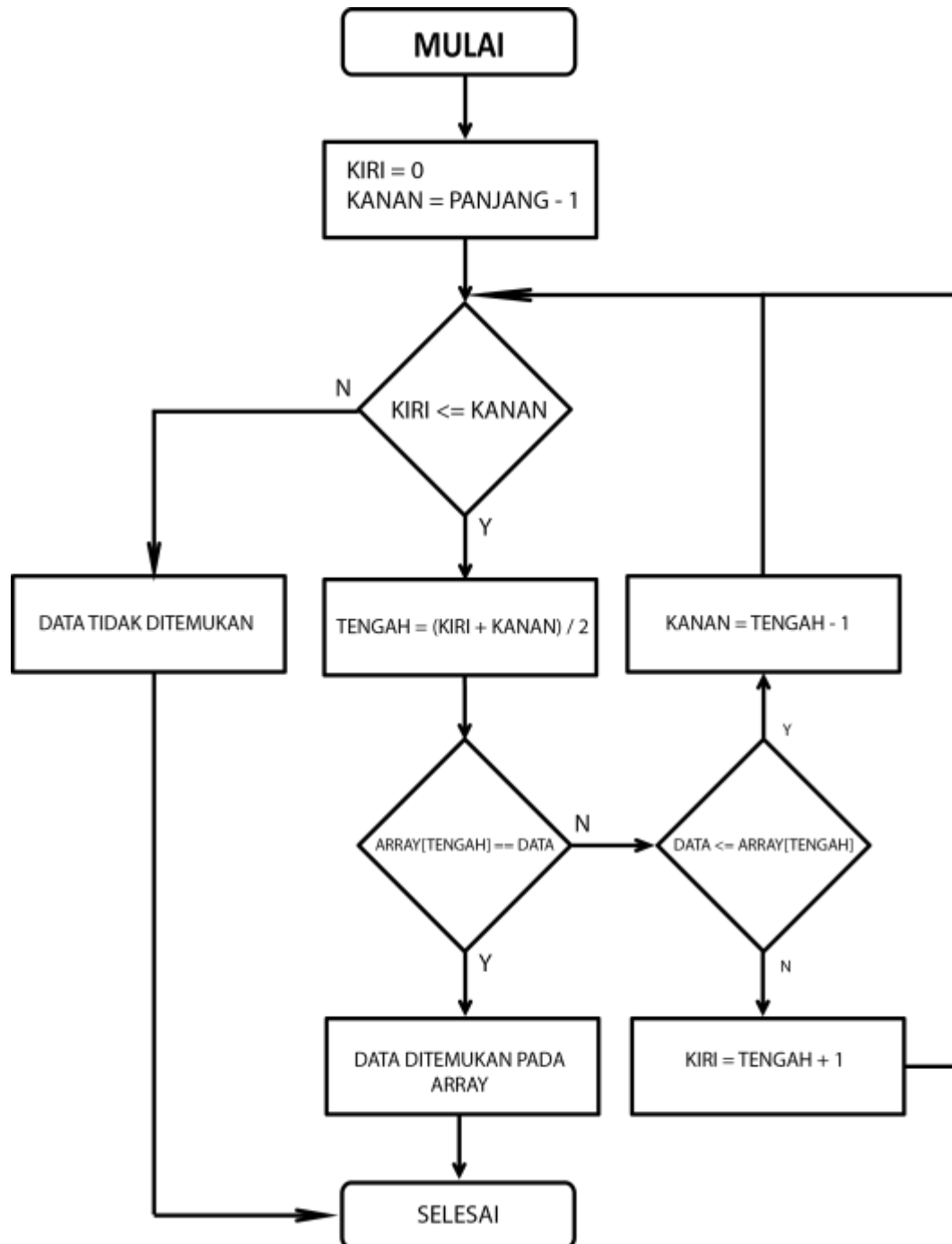
3.1.4. Flowchart

Flowchart adalah sebuah diagram yang menggambarkan urutan langkah-langkah yang secara logis yang digambarkan dengan symbol-simbol. *Flowchart* dari sistem yang akan dibangun ditunjukkan pada gambar dibawah ini:



Gambar 3.5. *Flowchart* Sistem

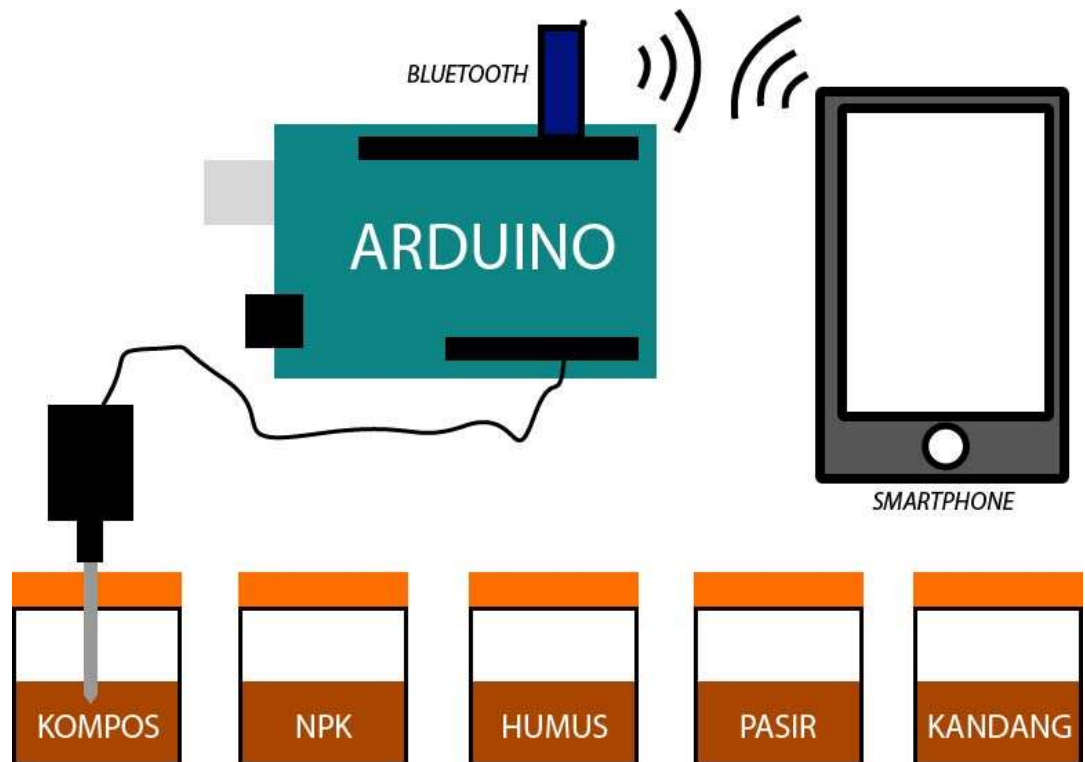
Sedangkan *flowchart* algoritma bisa dilihat pada gambar 3.6 dibawah ini:



Gambar 3.6. Flowchart Algoritma Binary Search

3.2. Blok Diagram Sistem

Dibawah ini merupakan gambar dari blok diagram dari sistem yang akan dibangun:



Gambar 3.7 Blok Diagram Sistem

Berikut ini merupakan penjelasan dari blok diagram sistem diatas:

1. Pengguna menghubungkan *smartphone* miliknya dengan Arduino menggunakan *Bluetooth* dengan mengklik tombol yang ada pada aplikasi.
2. Pengguna menancapkan sensor *pH probe* ke tanah yang akan dihitung nilai *pH* nya.
3. Pengguna mengklik tombol hitung yang ada pada aplikasi guna untuk menghitung nilai *pH* tanah tersebut.
4. Arduino akan memproses dan melakukan perhitungan berdasarkan nilai *analog* yang diterima dari sensor.

5. Arduino mengirim nilai dari hasil perhitungan tersebut, yang berupa pH tanah, ke *smartphone* pengguna menggunakan *Bluetooth*.
6. Pengguna mengklik tombol simpan yang ada pada aplikasi yang bertujuan untuk menyimpan nilai pH tanah pada *maps* dengan koordinat yang sesuai pada lokasi pengecekan tersebut agar pengguna lainnya juga mengetahui nilai pH pada tanah yang berada di lokasi tertentu.

3.3. Perancangan Sistem

Pada penelitian ini, perancangan meliputi 2 bagian utama yaitu perancangan perangkat keras dan perancangan perangkat lunak. Yang dimaksud perangkat keras disini adalah komponen-komponen fisik yang digunakan yang membentuk sistem elektronika. Sedangkan perangkat lunak dalam penelitian ini merupakan program yang dijalankan *smartphone* yang bekerja sama dengan sistem.

Perangkat keras dari sistem ini terdiri dari beberapa komponen utama yaitu *main board*, sensor, dan konektifitas lainnya. Perancangan perangkat keras haruslah dirancang sesimpel mungkin agar tidak mempersulit pengguna pada saat pengoperasiannya. Kemudian rancangan sistem juga harus fleksibel agar dapat dibawa kemana saja.

Perangkat lunak dari sistem yang dibangun memiliki beberapa kemampuan seperti menghubungkan perangkat *smartphone* pengguna dengan *main board* (Arduino) menggunakan modul *Bluetooth*, mengambil data yang telah diproses oleh Arduino, dan dapat menyimpan nilai tersebut pada *maps*.

Dalam perancangan sistem ini seperti yang telah disebutkan diatas, terdiri dari 2 bagian utama yaitu perancangan perangkat keras dan perangkat lunak, yang akan dirincikan lagi pada subbab berikut.

3.3.1. Perancangan Perangkat Keras

Pada sistem pengecekan pH tanah otomatis dengan menggunakan sensor pH *probe*, terdapat beberapa komponen utama dalam sistem ini agar dapat berjalan dengan lancar sesuai yang diharapkan, yaitu *main board* (Arduino), sensor, dan konektivitasnya.

3.3.1.1. Main Board (Arduino)

Komponen utama dalam sistem ini yaitu Arduino itu sendiri sebagai *main board* (papan utama), karena alat inilah yang menjadi unit pemrosesan utama. Arduino pada sistem ini juga menggunakan *shield* Arduino, agar ukuran sistem ini tidak terlalu memakan tempat dan menjadi lebih fleksibel.



Gambar 3.8. Bentuk Fisik Arduino
(Sumber: Sumber: <http://arduino.org>)

3.3.1.2.Sensor PH *Probe*

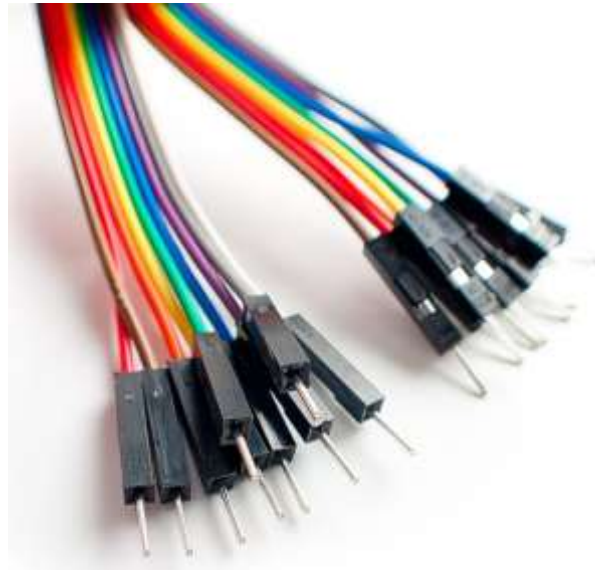
Agar derajat keasaman (pH) tanah dapat dibaca oleh Arduino, maka kita harus membutuhkan sensor yang mengubah besaran *analog*, yaitu nilai pH tanah itu sendiri, menjadi besaran digital agar dapat diproses oleh arduino dan sistem operasi android. Sensor yang digunakan pada sistem ini adalah sensor pH *probe*.



Gambar 3.9. Bentuk Fisik Sensor pH Tanah
(Sumber: Datasheet Sensor pH Tanah)

3.3.1.3.Konektivitas

Maksud dari konektivitas disini adalah bagaimana perangkat-perangkat yang digunakan dapat dihubungkan semuanya. Penulis menggunakan kabel *jumper male to male* sebagai konektivitas antara perangkat-perangkat elektronika seperti sensor dan modul *Bluetooth* dikarenakan mudah dipakai.



Gambar 3.10. Kabel *Jumper Male to Male*

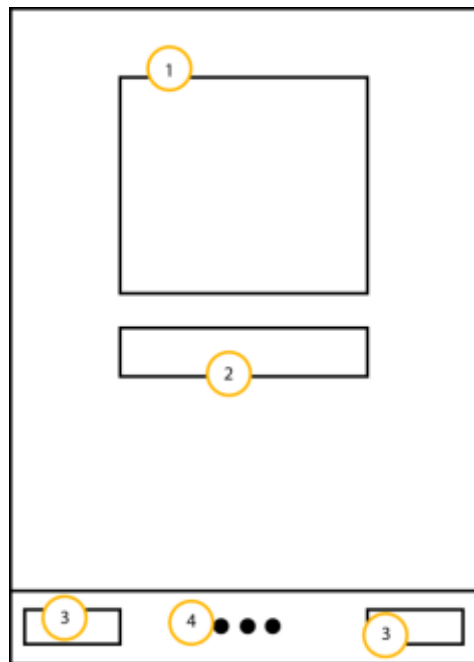
(Sumber: tokopedia.com)

3.3.2. Perancangan Perangkat Lunak

Perangkat lunak yang digunakan pada sistem ini adalah sebuah program dari sistem operasi android. Program aplikasi ini seperti yang sudah dijelaskan diatas memiliki kemampuan untuk menghubungkan *smartphone* dan Arduino.

3.3.2.1. Perancangan Antar Muka (*Interface*)

Aplikasi android ini memiliki 4 halaman yaitu halaman petunjuk penggunaan, halaman menghubungkan *Bluetooth*, halaman tampilan nilai pH tanah dan yang terakhir halaman tampilan *google maps*.



Gambar 3.11. Rancangan *Layout* Pembuka

Keterangan:

1. *Image View*

Merupakan sebuah gambar yang menunjukkan ilustrasi dari instruksi.

2. *Text View*

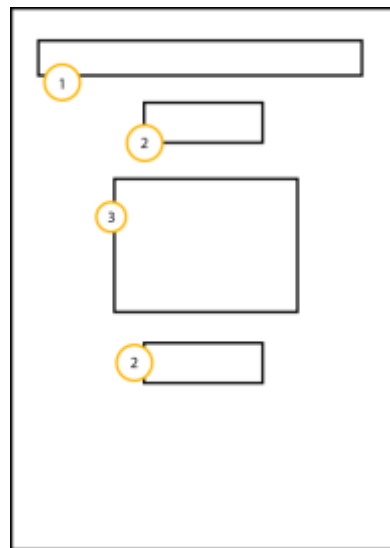
Berisi tentang petunjuk penggunaan sistem.

3. *Button*

Tombol yang melakukan aksi atau *procedure* jika di klik.

4. *Scroll Tab*

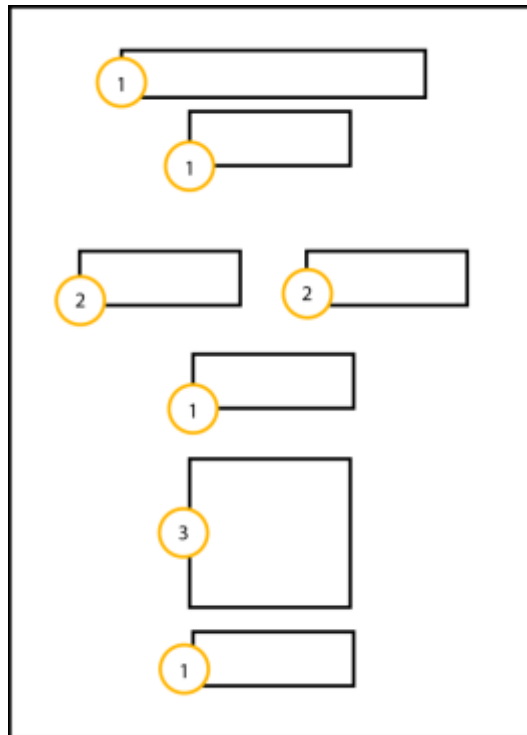
Berguna untuk men-*swipe layout* petunjuk penggunaan.



Gambar 3.12. Rancangan Tampilan Beranda

Keterangan:

1. *Layout Box*
Berfungsi sebagai indikator Koneksi *Bluetooth*.
2. *Button*
Tombol yang melakukan aksi atau *procedure* jika di klik.
3. *Image View*
Merupakan sebuah gambar yang menunjukkan ilustrasi dari instruksi.



Gambar 3.13. Rancangan Tampilan Hasil Perhitungan

Keterangan:

1. *Text View*
Berisi tentang informasi yang telah dijalankan oleh sistem.
2. *Button*
Tombol yang melakukan aksi atau *procedure* jika di klik.
3. *Image View*
Merupakan sebuah gambar yang menunjukkan ilustrasi dari instruksi.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

Pada tahap ini dilakukan implementasi sistem sesuai dengan analisis dan perancangan yang sudah ditentukan dan kemudian melakukan pengujian sistem.

4.1. Implementasi Sistem

Pada tahap implementasi sistem dilakukan pembuatan sistem sesuai dengan rancangan. Tahap ini dibagi menjadi dua sub bagian, yaitu konstruksi perangkat keras dan konstruksi perangkat lunak.

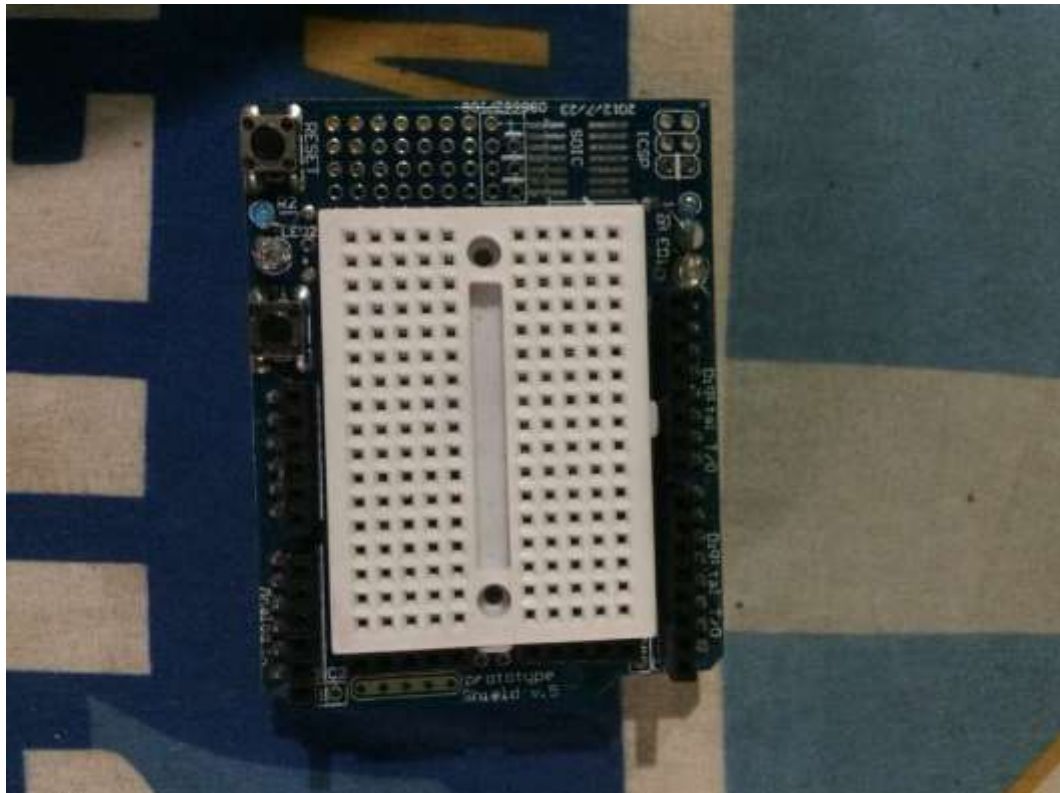
4.1.1. Konstruksi Utama

Kerangka utama dari sistem ini berupa sebuah papan Arduino Uno R3 karena mudah dioperasikan dan lebih simpel. Arduino pada sistem ini bertindak sebagai unit pemrosesan utama, dimana sensor serta perangkat-perangkat lainnya akan terhubung pada papan ini.

Kemudian papan ini terhubung dengan Arduino *Shield* agar unit-unit seperti input/output bisa diperluas. Gambar 4.1 menunjukkan bentuk fisik Arduino uno, sedangkan Gambar 4.2 menunjukkan bentuk fisik dari Arduino *shield*.



Gambar 4.1. Arduino Uno R3



Gambar 4.2. Arduino Shield

Fungsi dari Arduino *shield* ini salah satunya sebagai *expansion board*, dimana papan ini menyediakan *port-port* input output yang lebih banyak lagi, sehingga menghemat ruang dari sistem tersebut.

4.1.2. Sensor pH Tanah

Alat ini menggunakan sensor pH tanah yang akan diletakkan pada kerangka utama melalui papan *shield*. Sensor ini dihubungkan melalui dua kabel, yaitu kabel berwarna hitam sebagai nilai keluaran dari sensor (output) dan kabel putih yang dihubungkan ke *ground*. Gambar 4.3 dibawah ini menunjukkan bentuk fisik dari sensor pH tanah.

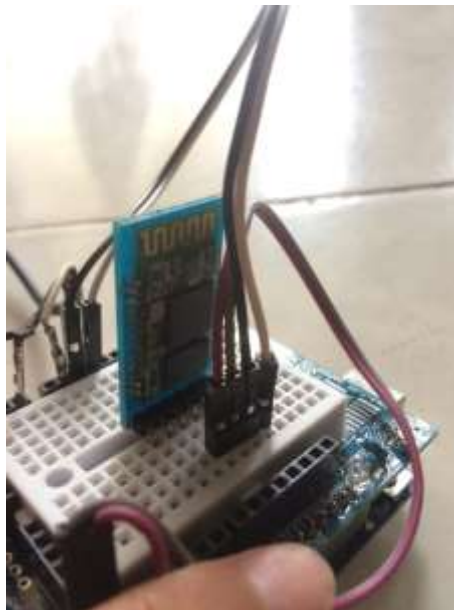


Gambar 4.3. Sensor pH Tanah

Pada implementasinya, sensor ini akan ditancapkan langsung pada sampel tanah yang digunakan, karena sensor inilah yang berinteraksi langsung dengan sampel tanah yang akan dihitung nilai derajat keasamannya.

4.1.3. *Bluetooth* HC-05

Sistem yang dibuat menggunakan *Bluetooth* HC-05 untuk pengiriman data dari Arduino ke *smartphone* pengguna. Perangkat ini dihubungkan ke Arduino dengan cara menghubungkan pin RX pada *Bluetooth* ke pin ke TX pada Arduino, dan pin TX pada Arduino ke pin RX pada Arduino. Gambar 4.4 dibawah ini menunjukkan hubungan antara Arduino dengan *Bluetooth* beserta konektivitasnya.



Gambar 4.4. *Bluetooth* HC-05

4.1.4. Daya listrik

Alat ini menggunakan baterai 9V sebagai sumber tenaganya dimana tegangan ini akan dialirkan menggunakan *socket* baterai. Jadi, pengguna tidak memerlukan sumber tenaga dari luar. Gambar 4.5 dibawah ini menunjukkan bentuk fisik dari *socket* baterai.



Gambar 4.5. Battery Clip

4.2. Penggabungan Perangkat Keras

Perangkat keras diimplementasikan dengan papan Arduino dan perangkat lainnya yang sudah dipaparkan diatas. Papan Arduino bertindak sebagai komponen sistem utama, karena pada komponen inilah semua data akan diproses dan proses input/output terjadi di unit ini. Gambar 4.6 dibawah ini menunjukkan penggabungan seluruh komponen perangkat keras yang dibutuhkan oleh sistem.



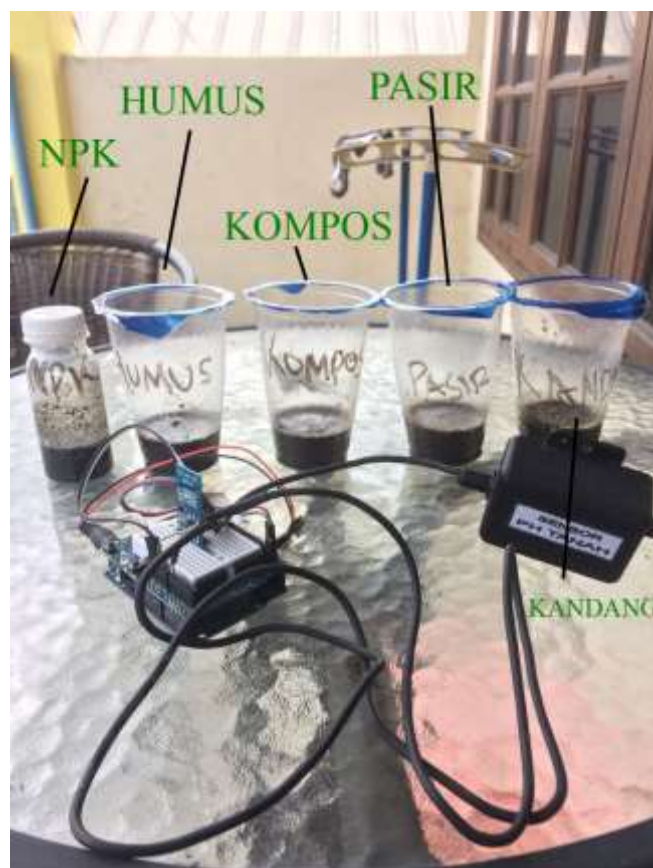
Gambar 4.6. Perangkat Keras Sistem

4.2.1 *Experimental Setup*

Gambar 4.7 dibawah ini menunjukkan *experimental setup*, yaitu alat dan bahan yang diperlukan dalam percobaan dibawah ini



Gambar 4.7 (a). *Experimental Setup Sampel Tanah*



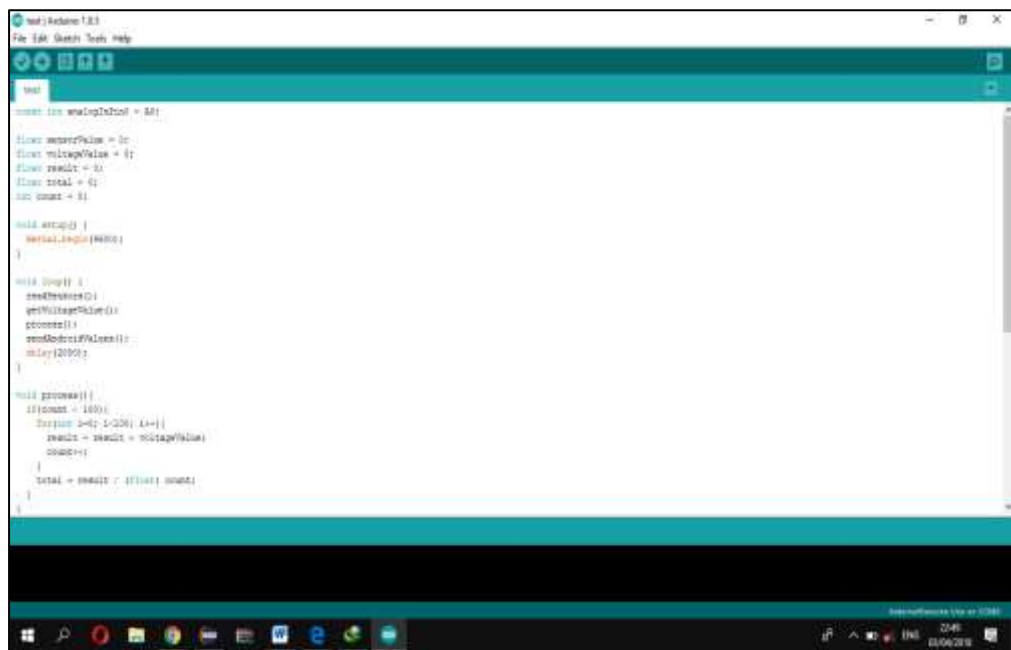
Gambar 4.7 (b). *Experimental Setup Sampel Cairan Tanah*

4.3. Pembuatan Perangkat Lunak

Pada tahap pembuatan perangkat lunak, tahap ini dibagi menjadi dua yaitu perangkat lunak Arduino dan perangkat lunak android.

4.3.1. Perangkat Lunak Arduino Uno

Papan Arduino uno diprogram menggunakan Bahasa pemrograman C dan aplikasi Arduino CC sebagai *compiler*-nya. File program dari *compiler* nya berekstensi .ino yang kemudian ditanamkan pada papan Arduino melalui kabel USB khusus papan Arduino. Gambar 4.8 dibawah ini merupakan tampilan dari Arduino IDE.



Gambar 4.8. Source code Arduino

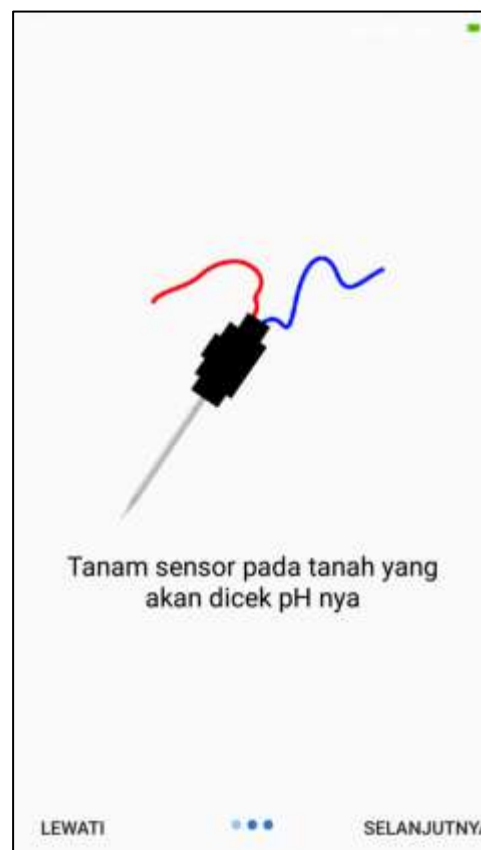
4.3.2. Perangkat Lunak Android

Aplikasi android pada sistem ini berfungsi untuk menampilkan informasi dari Arduino yang telah melakukan proses perhitungan pH tanah. Implementasi dari aplikasi android terdapat 3 bagian halaman *activity*. Berikut adalah rinciannya:

4.3.2.1. Panduan Penggunaan

Pada saat aplikasi pertama kali dijalankan, akan muncul tampilan panduan penggunaan yang dapat di *swipe*. Tampilan ini memiliki tiga *layout* yang dapat di *swipe*, dimana *layout* pertama berisikan instruksi untuk menghubungkan perangkat *smartphone* dengan sistem. Sedangkan *layout* kedua berisikan instruksi untuk menancapkan sensor ke sampel tanah yang akan dicek.

Sedangkan *layout* yang terakhir merupakan pemberitahuan dari hasil proses perhitungan pH tanah beserta informasi lainnya seperti tanaman yang cocok ditanam sesuai dengan pH yang telah dihitung. Gambar 4.9, gambar 4.10 dan gambar 4.11 dibawah ini merupakan *layout* dari halaman petunjuk penggunaan.



Gambar 4.9. Layout Pertama pada Halaman Petunjuk Penggunaan



Gambar 4.10. *Layout* Kedua pada Halaman Petunjuk Penggunaan



Gambar 4.11. *Layout* Ketiga pada Halaman Petunjuk Penggunaan

4.3.2.2. Menu Utama

Menu utama pada aplikasi android ini merupakan tampilan yang berisikan perintah untuk menghubungkan *smartphone* dengan sistem yang telah dihubungkan dengan *Bluetooth*. Jika pada *smartphone* pengguna *Bluetooth* belum dinyalakan, maka akan ada perintah untuk mengaktifkan *Bluetooth* terlebih dahulu. Dan tampilan ini hanya terdiri dari satu tombol untuk menghubungkan *smartphone* dengan sistem. Gambar 4.12 menunjukkan *layout* dari menu utama pada sistem.



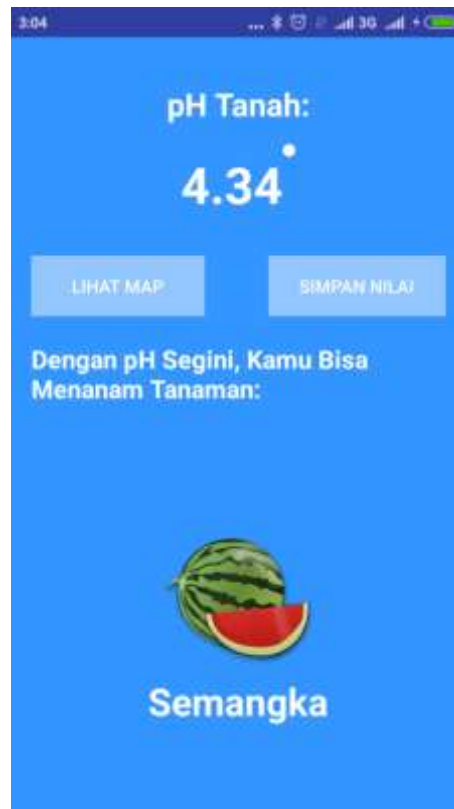
Gambar 4.12. Layout Menu Utama

4.3.2.3. Hasil Perhitungan

Menu hasil perhitungan merupakan tampilan yang berisikan informasi dari proses perhitungan pH tanah yang dilakukan oleh sensor. Selain menampilkan nilai pH nya, tampilan ini memiliki dua tombol, yaitu tombol simpan nilai dan tombol lihat map.

Tombol simpan nilai merupakan perintah untuk menyimpan nilai pH ke database menggunakan fasilitas *google firebase* dimana data yang disimpan berupa koordinat dari tempat pengecekan beserta nilai pH dan nama lokasinya. Sedangkan tombol lihat *map* merupakan perintah untuk membuka *layout* yang berisikan peta dimana peta tersebut terdapat *marker* yang berisi nilai pH tanah yang sudah diperiksa.

Selain itu, tampilan hasil perhitungan memiliki informasi berupa tanaman yang cocok untuk ditanam sesuai dengan pH yang telah dihitung. Tampilan ini dapat di *swipe*, sehingga pengguna dapat melihat tanaman apa saja yang sesuai untuk ditanam. Gambar 4.13 menunjukkan hasil dari *layout* proses perhitungan.



Gambar 4.13. Layout Hasil Perhitungan

4.3.2.4. Menu Lihat Map

Tampilan ini tidak jauh beda dengan tampilan *google maps*, dikarenakan aplikasi android ini menggunakan *google maps API* yang disediakan oleh *google*. Tampilan ini menampilkan peta lokasi dimana pengguna berada, dan juga terdapat *marker* yang sesuai dengan koordinat dari setiap pengecekan nilai pH tanah yang telah dilakukan. *Marker* ini menampilkan nama lokasi dan nilai pH yang telah dihitung. Gambar 4.14 menunjukkan lokasi pengecekan yang telah ditandai dengan *marker*.



Gambar 4.14. Layout Lihat Map

4.4. Pengujian Alat

Pengujian alat dilakukan untuk mengetahui apakah alat yang telah dibuat sesuai dengan analisis dan perancangan sistem yang telah dilakukan sebelumnya dan untuk mengetahui apakah alat dapat bekerja dengan baik atau tidak. Pada tahap ini, pengujian yang dilakukan adalah pengujian sensor pH tanah, membaca nilai pH dengan sensor, membaca nilai pH secara manual, proses pengiriman dan penyimpanan nilai pH tanah, dan yang terakhir pengecekan lokasi-lokasi di *maps*.

4.4.1. Pengujian Sensor pH Tanah

Pengujian sensor pH tanah ini adalah hal yang sangat penting dilakukan, mengingat proses perhitungan pH tanah ini hanya mengandalkan sensor ini. Sensor pH tanah menggunakan sinyal analog sebagai transimisinya. Rentang dari sinyal analog ini terdiri dari 10 bit dengan rentang 0-1023. Perlu diketahui bahwa nilai yang dikeluarkan oleh sensor merupakan nilai ADC (*Analog to Digital Converter*) yang akan diproses lebih lanjut. Pengujian ini dilakukan untuk membuktikan apakah sensor dapat bekerja dengan baik atau tidak. Tabel 4.1 dibawah ini merupakan tabel pengujian sensor pH tanah dengan tanah yang diberi larutan pH *Buffer* Asam-Basa berdasarkan *datasheet* sensor pH tanah.

Tabel 4.1. Data Uji Sensor pH Tanah

Tanah Asam				Tanah Basa			
Cairan Asam (ml)	pH	AVO Meter (mV)	ADC (Analog to Digital Converter)	Cairan Basa (ml)	pH	AVO Meter (mV)	ADC (Analog to Digital Converter)
0	7	49.7	7	0	7	41.5	6
6	6	117.9	20	6	7	36	4
12	4.9	204	35	12	-	-	-
18	4.3	234	45	18	-	-	-

Sehingga dari tabel diatas didapatkan persamaan :

$$y = -0.0693x + 7.3855$$

Dimana :

y = nilai pH

x = nilai ADC

nilai konstan = 7.3855

Contoh:

Didapat nilai ADC (*Analog to Digital Converter*) setelah pembacaan sampel tanah kompos sebesar 5. Jadi, nilai pH nya yaitu :

$$\begin{aligned} y &= -(0.0693 * x) + 7.3855 \\ &= -(0.0693 * 5) + 7.3855 \\ &= -0.3465 + 7.3855 \\ &= 7.039 \end{aligned}$$

Jadi, nilai pH yang didapat setelah melewati proses pembacaan sensor adalah 7.039.

Persamaan inilah yang akan digunakan oleh Arduino untuk menghitung segala macam jenis tanah. Tabel 4.2 dibawah menunjukkan hasil perhitungan dari beberapa nilai ADC yang berbeda-beda.

Tabel 4.2. Data Uji Rumus Konversi ADC ke pH

pH Tanah	AVO Meter (mV)	ADC	Hasil Perhitungan (pH)
7	36	4	7.1083
7	41.5	6	6.9697
7	49.7	7	6.9004
6	117.9	20	5.9995
4.9	204	35	4.96
4.3	234	45	4.267

4.4.2. Pengujian Pembacaan Sensor pada Sampel Tanah

Pada tahap ini dilakukan pengujian terhadap sampel tanah yang telah ditentukan, yaitu tanah humus, tanah dengan pupuk NPK, tanah kompos, tanah yang diberi pupuk kandang, dan pasir.

4.4.2.1. Pengujian pada Sampel Tanah Humus

Gambar 4.15 menunjukkan gambar dari proses pengecekan pH pada tanah humus. Tabel 4.3 menunjukkan hasil pengecekan pH tanah humus dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.15. Hasil Perhitungan pH Tanah Humus

```
void readSensors()
{
  sensorValue = analogRead(analogInPin0);
}
```

Gambar 4.16. Coding Snippet Pembacaan Nilai pH

Tabel 4.3. Percobaan pH Tanah dengan Sampel Tanah Humus

Percobaan	Nilai pH
1	4,47
2	4,47
3	4,82
4	5,58
5	5,58
6	5,58
7	5,58
8	5,58
9	5,58
10	5,58
Rata-rata	5,28

Berdasarkan data pada tabel 4.3 diatas menunjukkan bahwa sampel tanah humus bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.2.2. Pengujian pada Sampel Tanah dengan Pupuk Kompos

Gambar 4.17 menunjukkan gambar dari proses pengecekan pH pada tanah dengan campuran pupuk kompos. Tabel 4.4 menunjukkan hasil pengecekan pH tanah dengan pupuk kompos dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.17. Hasil Perhitungan pH Tanah dengan Campuran Pupuk Kompos

Tabel 4.4. Percobaan pH Tanah dengan Sampel Tanah dengan Pupuk Kompos

Percobaan	Nilai pH
1	5,39
2	5,38
3	4,82
4	5,10
5	5,03
6	4,96
7	4,68
8	4,82

9	4,82
10	4.06
Rata-rata	4,9

Berdasarkan data pada tabel 4.4 diatas menunjukkan bahwa sampel tanah dengan pupuk kompos bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.2.3. Pengujian pada Sampel Tanah dengan Pupuk NPK

Gambar 4.18 menunjukkan gambar dari proses pengecekan pH pada tanah dengan campuran pupuk NPK. Tabel 4.5 menunjukkan hasil pengecekan pH tanah dengan pupuk NPK dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.18. Hasil Perhitungan pH Tanah dengan Campuran Pupuk NPK

Tabel 4.5. Percobaan pH Tanah dengan Sampel Tanah dengan Pupuk NPK

Percobaan	Nilai pH
1	1,77
2	2,12
3	1,84
4	1,77
5	1,98
6	2,12
7	2,26
8	1,98
9	2,19
10	2,12
Rata-rata	2,015

Berdasarkan data pada tabel 4.5 diatas menunjukkan bahwa sampel tanah dengan pupuk kompos bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.2.4. Pengujian pada Sampel Tanah dengan Pupuk Kandang

Gambar 4.19 menunjukkan gambar dari proses pengecekan pH pada tanah dengan campuran pupuk Kandang. Tabel 4.6 menunjukkan hasil pengecekan pH tanah dengan pupuk kandang dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.19. Hasil Perhitungan pH Tanah dengan Campuran Pupuk Kandang

Tabel 4.6. Percobaan pH Tanah dengan Sampel Tanah dengan Pupuk Kandang

Percobaan	Nilai pH
1	3,99
2	4,89
3	4,41
4	4,47
5	4,34
6	4,27
7	4,20
8	4,13
9	4,13
10	4,06
Rata-rata	4,28

Berdasarkan data pada tabel 4.6 diatas menunjukkan bahwa sampel tanah dengan pupuk kandang bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.2.5. Pengujian pada Sampel Pasir

Gambar 4.20 menunjukkan gambar dari proses pengecekan pH pada pasir. Tabel 4.7 menunjukkan hasil pengecekan pH pasir dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.20. Hasil Perhitungan pH Tanah dengan Sampel Pasir

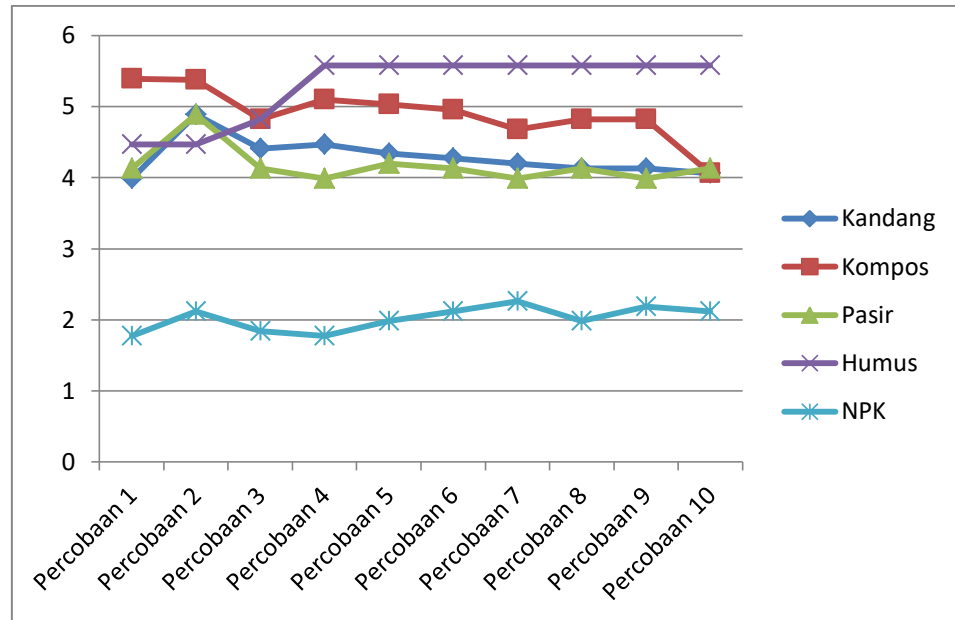
Tabel 4.7. Percobaan pH Tanah dengan Sampel Pasir

Percobaan	Nilai pH
1	4,13
2	4,89
3	4,13
4	3,99
5	4,20
6	4,13
7	3,99
8	4,13
9	3,99
10	4,13
Rata-rata	4,17

Berdasarkan data pada tabel 4.7 diatas menunjukkan bahwa pasir bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.2.6 Hasil Akhir Pengujian pada Sampel Tanah

Gambar 4.21 menunjukkan grafik dari masing-masing percobaan pada sampel tanah.



Gambar 4.21. Grafik Hasil Perhitungan pada Sampel Tanah

Tabel 4.8. Hasil Perhitungan Seluruh Sampel

Sampel	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	\bar{x}
Kandang	3,99	4,89	4,41	4,47	4,34	4,27	4,20	4,13	4,13	4,06	4,28
Kompos	5,39	5,38	4,82	5,10	5,03	4,96	4,68	4,82	4,82	4,06	4,9
Pasir	4,13	4,89	4,13	3,99	4,20	4,13	3,99	4,13	3,99	4,13	4,17
Humus	4,47	4,47	4,82	5,58	5,58	5,58	5,58	5,58	5,58	5,58	5,28
NPK	1,77	2,12	1,84	1,77	1,98	2,12	2,26	1,98	2,19	2,12	2,01

Dimana X_n = Nilai pH



Gambar 4.22. Sampel Tanah

4.4.3. Pengujian Pembacaan Sensor pada Sampel Cairan Tanah

Pada tahap ini dilakukan pengujian terhadap sampel cairan tanah yang telah ditentukan, yaitu cairan dari tanah humus, tanah dengan pupuk NPK, tanah kompos, tanah yang diberi pupuk kandang, dan pasir.

4.4.3.1. Pengujian pada Sampel Cairan Tanah Humus

Gambar 4.22 menunjukkan gambar dari proses pengecekan pH pada cairan tanah humus. Tabel 4.9 menunjukkan hasil pengecekan pH cairan tanah humus dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.23. Hasil Perhitungan pH Cairan Tanah Humus

```
void readSensors()
{
  sensorValue = analogRead(analogInPin0);
}
```

Gambar 4.24. Coding Snippet Pengambilan Nilai PH

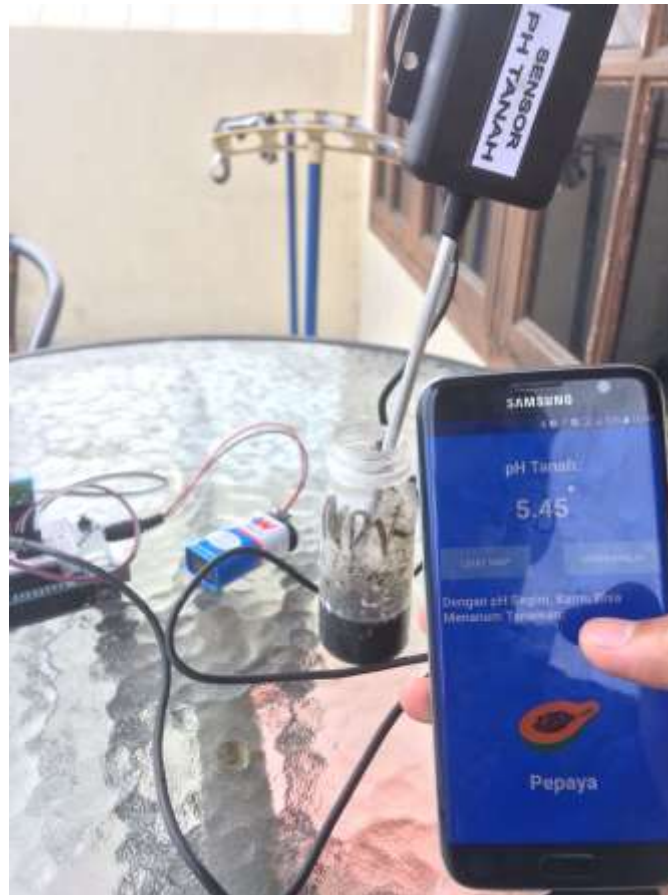
Tabel 4.9. Percobaan pH Tanah dengan Sampel Cairan Tanah Humus

Percobaan	Nilai pH
1	2,26
2	2,19
3	2,33
4	2,40
5	2,26
6	2,53
7	2,53
8	2,47
9	2,60
10	2,74
Rata-rata	2,43

Berdasarkan data pada tabel 4.9 diatas menunjukkan bahwa sampel cairan tanah humus bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.3.2. Pengujian pada Sampel Cairan Tanah dengan Pupuk NPK

Gambar 4.24 menunjukkan gambar dari proses pengecekan pH pada cairan tanah dengan pupuk NPK. Tabel 4.10 menunjukkan hasil pengecekan pH cairan tanah dengan pupuk NPK dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.25. Hasil Perhitungan pH Cairan Tanah dengan Pupuk NPK

Tabel 4.10. Percobaan pH Tanah dengan Sampel Cairan Tanah dengan Pupuk NPK

Percobaan	Nilai pH
1	4,41
2	4,20
3	4,54
4	4,54
5	4,89
6	4,75
7	4,96
8	5,03

9	5,31
10	5,45
Rata-rata	4,8

Berdasarkan data pada tabel 4.9 diatas menunjukkan bahwa sampel cairan tanah dengan pupuk NPK bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.3.3. Pengujian pada Sampel Cairan Tanah Kompos

Gambar 4.25 menunjukkan gambar dari proses pengecekan pH pada cairan tanah kompos. Tabel 4.11 menunjukkan hasil pengecekan pH cairan tanah kompos dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.26. Hasil Perhitungan pH Cairan Tanah Kompos

Tabel 4.11. Percobaan pH Tanah dengan Sampel Cairan Tanah Kompos

Percobaan	Nilai pH
1	1,84
2	1,91
3	2,12
4	2,19
5	2,19
6	2,12
7	2,12
8	2,53

9	2,6
10	2,67
Rata-rata	2,29

Berdasarkan data pada tabel 4.10 diatas menunjukkan bahwa sampel cairan tanah kompos bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.3.4. Pengujian pada Sampel Cairan Tanah dengan Pupuk Kandang

Gambar 4.26 menunjukkan gambar dari proses pengecekan pH pada cairan tanah dengan pupuk kandang. Tabel 4.12 menunjukkan hasil pengecekan pH cairan tanah dengan pupuk kandang dengan banyaknya percobaan sebanyak 10 kali.



Gambar 4.27. Hasil Perhitungan pH Cairan Tanah dengan Pupuk Kandang

Tabel 4.12. Percobaan pH Tanah dengan Sampel Cairan Tanah dengan Pupuk Kandang

Percobaan	Nilai pH
1	4,13
2	4,61
3	4,47
4	4,75
5	4,89
6	4,47
7	4,89
8	4,96

9	4,68
10	4,61
Rata-rata	4,64

Berdasarkan data pada tabel 4.12 diatas menunjukkan bahwa sampel cairan tanah dengan pupuk kandang bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.3.5. Pengujian pada Sampel Cairan Pasir

Gambar 4.27 menunjukkan gambar dari proses pengecekan pH pada cairan pasir. Tabel 4.13 menunjukkan hasil pengecekan pH cairan tanah dengan pupuk kandang dengan banyaknya percobaan sebanyak 10 kali



Gambar 4.28. Hasil Perhitungan pH Cairan Pasir

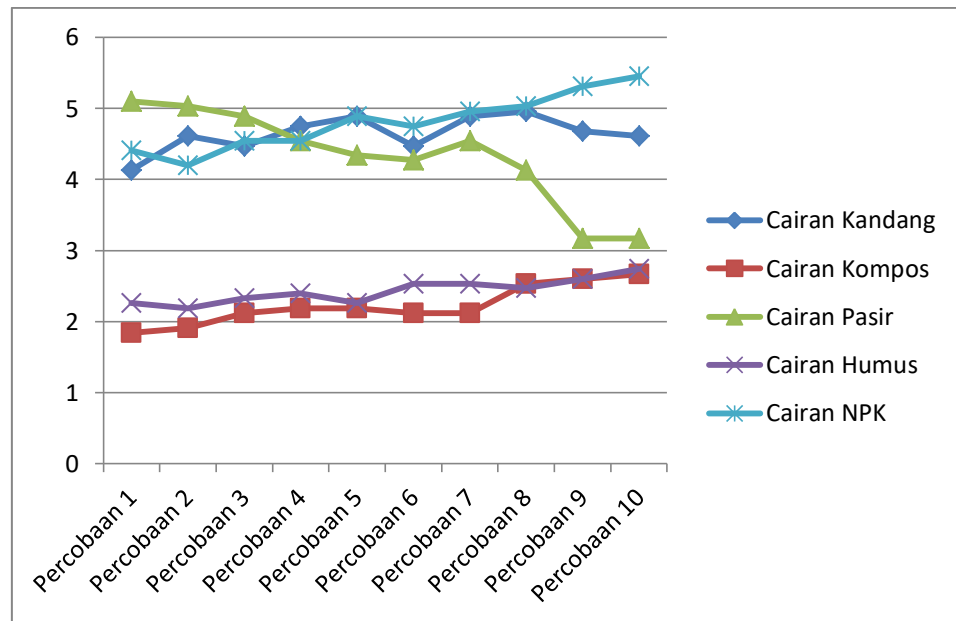
Tabel 4.13. Percobaan pH Tanah dengan Sampel Cairan Pasir

Percobaan	Nilai pH
1	5,1
2	5,03
3	4,89
4	4,54
5	4,34
6	4,27
7	4,54
8	4,13
9	3,17
10	3,17
Rata-rata	4,31

Berdasarkan data pada tabel 4.13 diatas menunjukkan bahwa sampel cairan pasir bersifat asam dikarenakan nilainya berada dibawah 7 (<7).

4.4.3.6 Hasil Akhir Pengujian pada Sampel Cairan Tanah

Gambar 4.28 menunjukkan grafik dari masing-masing percobaan pada sampel tanah.



Gambar 4.29. Grafik Hasil Perhitungan pada Sampel Cairan Tanah

Tabel 4.14. Hasil Perhitungan Seluruh Sampel

Sampel	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	\bar{x}
Cairan Kandang	4,13	4,61	4,47	4,75	4,89	4,47	4,89	4,96	4,68	4,61	4,64
Cairan Kompos	1,84	1,91	2,12	2,19	2,19	2,12	2,12	2,53	2,6	2,67	2,29
Cairan Pasir	5,1	5,03	4,89	4,54	4,34	4,27	4,54	4,13	3,17	3,17	4,31
Cairan Humus	2,26	2,19	2,33	2,40	2,26	2,53	2,53	2,47	2,60	2,74	2,43
Cairan NPK	4,41	4,20	4,54	4,54	4,89	4,75	4,96	5,03	5,31	5,45	4,8

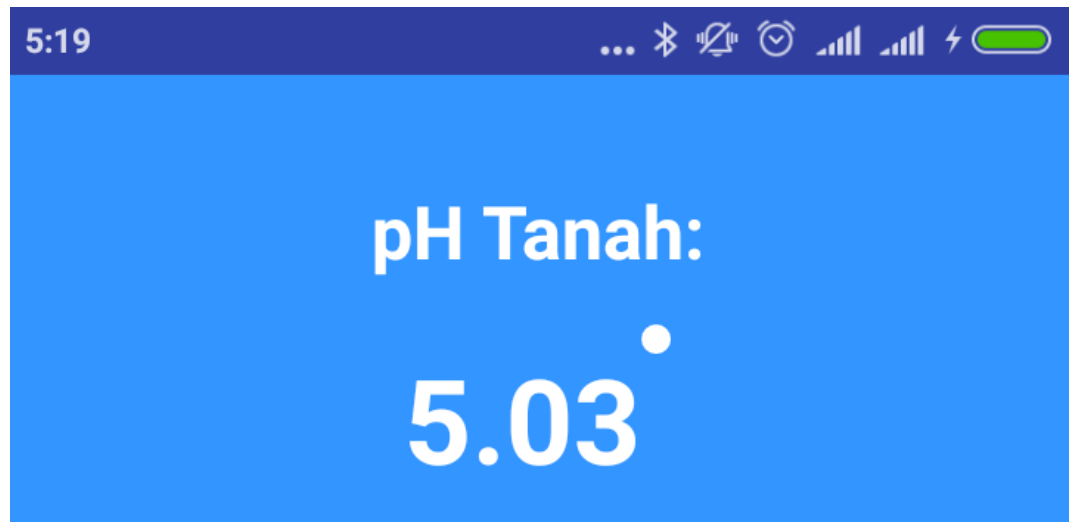
Dimana X_n = Nilai pH



Gambar 4.30. Sampel Cairan Tanah

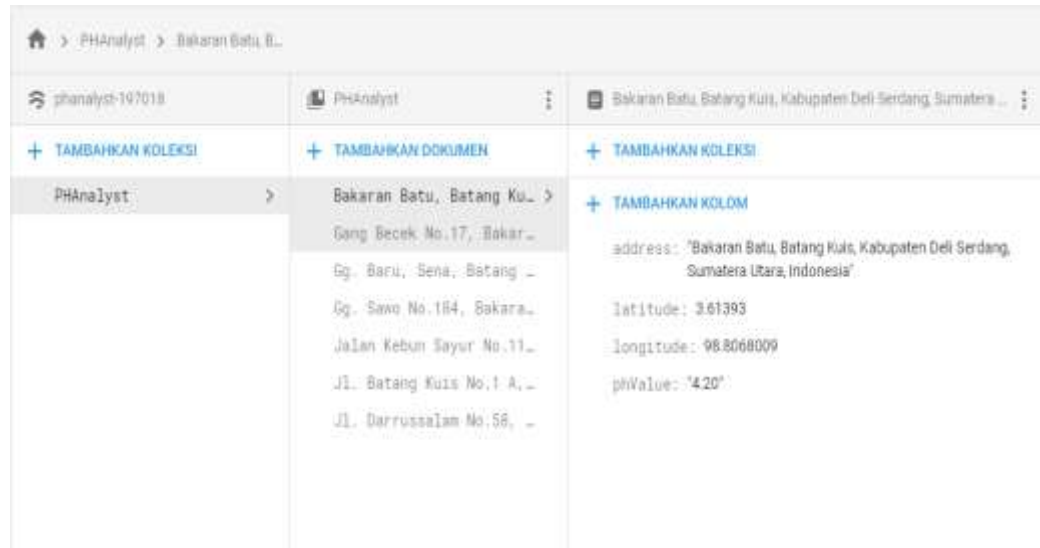
4.4.4. Pengujian Pengiriman dan Penyimpanan Nilai pH

Setelah sensor membaca dan menghitung nilai pH dari beberapa sampel tanah, kemudian aplikasi android akan menampilkan nilai yang telah diproses oleh Arduino. Gambar 4.29 menunjukkan hasil dari pembacaan data yang dikirim oleh Arduino.



Gambar 4.31. Hasil Pengiriman dari Arduino ke Android

Jika pengguna menekan tombol simpan nilai, maka nilai pH tersebut akan disimpan di *database* pada *google firebase* dan juga nama lokasi dan koordinat akan ikut disimpan. Gambar 4.30 menunjukkan nilai yang sudah tersimpan ke *database*.



Gambar 4.32. Database pH Tanah Beserta Lokasi

4.4.5. Pengujian Pengecekan Lokasi di Maps

Setelah nilai pH tersimpan di *database*, pengguna dapat melihat lokasi mana saja yang sudah dicek nilai pH tanahnya. Lokasi ini ditandai dengan *marker-marker* yang ada pada *google maps*. Marker ini jika diklik akan memunculkan nama lokasi beserta nilai pH yang telah dihitung. Gambar 4.31 dibawah ini menunjukkan lokasi dimana tempat pengecekan pH tanah telah dilaksanakan.



Gambar 4.33. Pengujian Lokasi pada *Google Maps*

```

private void getDeviceLocation() {
    Log.d(TAG, "getDeviceLocation: getting current device location");

    mFusedLocationProviderClient =
        LocationServices.getFusedLocationProviderClient(this);

    try {
        if (mLocationPermissionGranted) {
            final Task location = mFusedLocationProviderClient.getLastLocation();
            location.addOnCompleteListener(new OnCompleteListener() {
                @Override
                public void onComplete(@NonNull Task task) {
                    if (task.isSuccessful()) {
                        Log.d(TAG, "onComplete: found location!");
                        Location currentLocation = (Location) task.getResult();

                        /*geocoder = new Geocoder(getApplicationContext(),
                        Locale.getDefault());

                        try {
                            addresses =
                                geocoder.getFromLocation(currentLocation.getLatitude(),
                                currentLocation.getLongitude(), 1);
                        } catch (IOException e) {
                            e.printStackTrace();
                        }

                        String myAddress = addresses.get(0).getAddressLine(0);*/

                        moveCamera(new LatLng(currentLocation.getLatitude(),
                        currentLocation.getLongitude()), 25f);
                    } else {
                        Log.d(TAG, "onComplete: current location is null!");
                        Toast.makeText(Maps.this, "Unable to get current location",
                        Toast.LENGTH_SHORT).show();
                    }
                }
            });
        }
    } catch (SecurityException e) {
        Log.d(TAG, "getDeviceLocation: Security Exception: " + e.getMessage());
    }
}

```

Gambar 4.34. Coding Snippet Lokasi Pengecekan

4.5. Perhitungan Manual Algoritma *Binary Search*

Array = [4.0, 4.5, 5.0, 5.5, 6.0]

Left = 0, Right = 4;

Nilai pH = 4.63

1. Pertama nilai pH dibulatkan ke atas/ke bawah dengan fungsi *ceil* atau *floor*. Jadi, nilai pH = 5.0 jika digunakan fungsi *ceil*.
2. Kemudian dilakukan pengecekan apakah indeks terkecil yang sekarang lebih kecil dari indeks terbesar yang sekarang. Jika benar, maka proses akan berlanjut. Jika tidak, proses akan berhenti.
3. Kemudian dicari nilai tengah dari *array* tersebut dengan cara menjumlahkan indeks terkecil dengan indeks terbesar kemudian dibagi dengan dua. Dalam kasus diatas, nilai tengahnya yaitu $(0+4)/2 = 2$.
4. Kemudian dilakukan pengecekan, apakah nilai pH diatas sama nilainya dengan *array* dengan indeks ke 2 (nilai tengah). Jika iya maka proses berhenti. Jika tidak maka akan dilakukan pengulangan proses selanjutnya. Dalam kasus diatas, 5.0 sama dengan nilai array yang indeksnya nilai tengah. Oleh karena itu, proses perhitungan selesai.
5. Jika nilai yang dicari tidak sama, maka nilai pH tersebut dicek apakah lebih kecil dari nilai *array* dengan indeks ke 2 (nilai tengah). Jika iya, maka proses pencarian dilakukan sama seperti diatas, hanya saja indeks terkecil dimulai dari 0, dan indeks terbesar dimulai dari indeks yang bernilai nilai tengah dikurangi satu, yaitu 1. Jika tidak, proses pencarian dilakukan seperti diatas dengan indeks terkecil dimulai dari nilai tengah ditambah satu, yaitu 3, dan indeks terbesar bernilai panjang array dikurangi satu, yaitu 4.
6. Jika pada proses diatas tidak ada nilai yang ditentukan, maka proses perhitungan akan memberikan nilai -1, yang artinya tidak ada ditemukan.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan pembahasan dan hasil dari penelitian, maka diperoleh beberapa kesimpulan sebagai berikut:

1. Hasil penggunaan sensor pH tanah tidak jauh beda dengan perhitungan pH tanah secara manual.
2. Tegangan yang dikeluarkan sensor berbeda-beda, sesuai dengan derajat pH dari beberapa sampel tanah.
3. Sensor membutuhkan waktu sekitar 2 s.d 5 detik untuk menghitung nilai pH dari sampel tanah.
4. *Bluetooth* akan terus mengirimkan nilai pH tersebut ke *smartphone* dengan interval 2 detik agar *smartphone* dapat membaca data yang dikirim tersebut secara optimal.

5.2. Saran

Adapun saran-saran yang dapat dipertimbangkan dari hasil penelitian ini agar penelitian ini dapat dikembangkan lebih lanjut yaitu:

1. Karena dalam penelitian ini penulis hanya menggunakan sensor pH tanah saja, diharapkan penelitian selanjutnya dapat menggunakan sensor-sensor lainnya seperti sensor kelembapan tanah, dll.
2. Untuk penggunaan sensor diharapkan penelitian selanjutnya memilih sensor yang memiliki kualitas lebih bagus lagi dikarenakan sensor yang digunakan penulis kinerja nya kurang bagus dan terkadang tidak stabil.

DAFTAR PUSTAKA

- Fitri,Tiya Fadila.2014.*Aplikasi Tuntunan Shalat Lima Waktu Berbasis Android*.Skripsi.Medan: Universitas Sumatera Utara.
- Gupta, Gaytri dkk.*Microcontroller based IFSET pH Measurement System with Wireless Communication.Procedure of International Conference on Control, Communication and Power Engineering*.2010
- Jupri,Achmad,Abdul Muid, Muliadi.Rancang Bangun Alat Ukur Suhu, Kelembapan, dan pH pada Tanah Berbasis Mikrokontroller ATMega328P.*Jurnal Edukasi dan Penelitian Informatika (JEPIN) Vol. 3,No. 2*.2017.
- Kustanti,Ika.Pengendalian Kadar Keasaman (pH) Pada Sistem Hidroponik Stroberi Menggunakan Kontroler PID Berbasis Arduino Uno.*Jurnal Ika Kustanti*.2014.
- Lazuardi,Muhammad.*Aplikasi Mikrokontroller sebagai Kontroler Proporsional pada Pengaturan PH*.Skripsi.Semarang: Universitas Diponegoro.
- Prayoga,Dwi.2015.*Rancangan Alat Ukur PH Meter Berbasis Mikrokontroller ATMega8*.Skripsi.Medan: Universitas Sumatera Utara.
- Saputra,Akip.*Pengukur Kadar Keasaman dan Kekeruhan Air berbasis Arduino*.Skripsi.Surakarta: Universitas Muhammadiyah Surakarta.
- Tindaon,Afif Yumna.2017. *Rancang Bangun Penjaga Jarak Memanfaatkan Bluetooth HC-05 dan Bluetooth HC-06 Berbasis Mikrokontroller ATMega328*.Skripsi.Medan: Universitas Sumatera Utara.

LISTING PROGRAM

1. *Source Code* Arduino

```

const int analogInPin0 = A0;

float sensorValue = 0;
float voltageValue = 0;
float result = 0;
float total = 0;
int count = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  readSensors();
  getVoltageValue();
  process();
  sendAndroidValues();
  delay(2000);
}

void process(){
  if(count < 100){
    for(int i=0; i<100; i++){
      result = result + voltageValue;
      count++;
    }
    total = result / (float) count;
  }
}

void readSensors()
{
  sensorValue = analogRead(analogInPin0);
}

void sendAndroidValues()
{
  Serial.print('#');
  Serial.print(total);
  Serial.print('+');
  Serial.print('~');
  Serial.println();
}

```

```
    delay(10);  
}  
  
void getVoltageValue()  
{  
    voltageValue = (-0.0693 * sensorValue) + 7.3855;  
}
```

2. Listing Program Android

2.1 MainActivity.java

```

package com.meivaldi.phanalyst;

import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.Toast;

import java.io.IOException;
import java.util.Set;
import java.util.UUID;

public class MainActivity extends AppCompatActivity {

    Button btConnect;

    private ProgressDialog progressDialog;

    public static String EXTRA_DEVICE_ADDRESS = "device_address";

    private BluetoothAdapter mBtAdapter;
    private BluetoothDevice bluetoothDevice = null;
    private BluetoothSocket socket = null;
    private static final UUID BTMODULEUUID =
        UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
    }

```

```

getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
    WindowManager.LayoutParams.FLAG_FULLSCREEN);
setContentView(R.layout.activity_main);

btConnect = (Button) findViewById(R.id.connect);

mBtAdapter = BluetoothAdapter.getDefaultAdapter();

progressDialog = new ProgressDialog(this);
progressDialog.setMessage("Menyambungkan...");
progressDialog.setTitle("Tunggu Sebentar");
progressDialog.setIcon(R.drawable.sensor);

if(mBtAdapter == null)
{
    Toast.makeText(getApplicationContext(), "Bluetooth Tidak Tersedia", Toast.LENGTH_LONG).show();

    finish();
}
else if(!mBtAdapter.isEnabled())
{
    Intent turnBTon = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(turnBTon,1);
}

btConnect.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        progressDialog.show();
        connectBlueTooth();
        Toast.makeText(getApplicationContext(), "Connected",
Toast.LENGTH_LONG).show();
        String address = bluetoothDevice.getAddress();

        Log.d("Debug", address);

        if(!address.isEmpty()){
            progressDialog.hide();
            Intent intent = new Intent(getApplicationContext(),
ResultActivity.class);
            intent.putExtra(EXTRA_DEVICE_ADDRESS, address);
            startActivity(intent);
        } else {
            Toast.makeText(getApplicationContext(), "Perangkat tidak ditemukan!", Toast.LENGTH_LONG).show();

```

```

        }
    }
});

}

private void connectBlueTooth() {
    Set<BluetoothDevice> pairedDevices =
mBtAdapter.getBondedDevices();

    if (pairedDevices.size()>0)
    {
        for(BluetoothDevice bt : pairedDevices)
        {
            if(bt.getName().equals("HC-05")){
                bluetoothDevice = bt;
                break;
            }
        }
    } else {
        Toast.makeText(getApplicationContext(), "Tidak ada perangkat
yang ditemukan", Toast.LENGTH_LONG).show();
    }

}

private final BroadcastReceiver mReceiver = new BroadcastReceiver()
{
    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();

        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            Toast.makeText(getApplicationContext(), "Found Device",
Toast.LENGTH_LONG).show();
        }
        else if
(BluetoothDevice.ACTION_ACL_CONNECTED.equals(action)) {
            Toast.makeText(getApplicationContext(), "Device is now
connected!", Toast.LENGTH_LONG).show();
        }
        else if
(BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(action)) {
            Toast.makeText(getApplicationContext(), "Done searching",
Toast.LENGTH_LONG).show();
        }
    }
}

```

```

        else if
        (BluetoothDevice.ACTION_ACL_DISCONNECT_REQUESTED.equals(
        action)) {
            Toast.makeText(getApplicationContext(), "Device is about to
            disconnect", Toast.LENGTH_LONG).show();
        }
        else if
        (BluetoothDevice.ACTION_ACL_DISCONNECTED.equals(action)) {

            Toast.makeText(getApplicationContext(), "Device has
            disconnected", Toast.LENGTH_LONG).show();
        }
    };

    private void checkConnectivity(){
        try {
            socket =
            bluetoothDevice.createRfcommSocketToServiceRecord(BTMODULEUU
            ID);
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            socket.connect();
        } catch (IOException e) {
            e.printStackTrace();
        }

        IntentFilter filter = new IntentFilter();
        filter.addAction(BluetoothDevice.ACTION_ACL_CONNECTED);

        filter.addAction(BluetoothDevice.ACTION_ACL_DISCONNECT_REQU
        ESTED);

        filter.addAction(BluetoothDevice.ACTION_ACL_DISCONNECTED);
        this.registerReceiver(mReceiver, filter);
    }
}

```

2.2 Maps.java

```

package com.meivaldi.phanalyst;

import android.*;
import android.Manifest;
import android.app.Dialog;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GoogleApiAvailability;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.firestore.CollectionReference;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QuerySnapshot;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Locale;

public class Maps extends AppCompatActivity implements
OnMapReadyCallback{

```

```

private static final String TAG = "MapActivity";

private static final String FINE_LOCATION =
Manifest.permission.ACCESS_FINE_LOCATION;
private static final String COARSE_LOCATION =
Manifest.permission.ACCESS_COARSE_LOCATION;
private static final int LOCATION_PERMISSION_REQUEST_CODE
= 1234;

private Boolean mLocationPermissionGranted = false;
private GoogleMap mMap;
private FusedLocationProviderClient mFusedLocationProviderClient;

private Geocoder geoCoder;
private List<Address> addresses;

private FirebaseFirestore mFirestore = FirebaseFirestore.getInstance();
private CollectionReference mCollectionReference =
mFirestore.collection("PHAnalyst");
public static ArrayList<PlaceInfo> arrayList = new ArrayList<>();

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_map);

    getLocationPermission();
}

private void getDeviceLocation() {
    Log.d(TAG, "getDeviceLocation: getting current device location");

    mFusedLocationProviderClient =
    LocationServices.getFusedLocationProviderClient(this);

    try {
        if (mLocationPermissionGranted) {
            final Task location =
mFusedLocationProviderClient.getLastLocation();
            location.addOnCompleteListener(new OnCompleteListener() {
                @Override
                public void onComplete(@NonNull Task task) {
                    if (task.isSuccessful()) {
                        Log.d(TAG, "onComplete: found location!");
                        Location currentLocation = (Location) task.getResult();

                        /*geoCoder = new Geocoder(getApplicationContext(),
Locale.getDefault());

```



```

        try {
            addresses =
geoCoder.getFromLocation(currentLocation.getLatitude(),
currentLocation.getLongitude(), 1);
        } catch (IOException e) {
            e.printStackTrace();
        }

        String myAddress =
addresses.get(0).getAddressLine(0);*/

        moveCamera(new LatLng(currentLocation.getLatitude(),
currentLocation.getLongitude()), 25f);
    } else {
        Log.d(TAG, "onComplete: current location is null!");
        Toast.makeText(Maps.this, "Unable to get current
location", Toast.LENGTH_SHORT).show();
    }
}
});
}
} catch (SecurityException e) {
    Log.d(TAG, "getDeviceLocation: Security Exception: " +
e.getMessage());
}
}

private void moveCamera(LatLng latLng, float zoom) {
    Log.d(TAG, "moveCamera: Moving camera to: lat: " +
latLng.latitude + " , lng: " + latLng.longitude);
    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng,
zoom));
}

private void getLocationPermission() {
    String[] permissions = {
        Manifest.permission.ACCESS_COARSE_LOCATION,
        Manifest.permission.ACCESS_FINE_LOCATION
    };

    if
(ContextCompat.checkSelfPermission(this.getApplicationContext(),
        FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
        if
(ContextCompat.checkSelfPermission(this.getApplicationContext(),

```

```

        COARSE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
            mLocationPermissionGranted = true;
            initMap();
        } else {
            ActivityCompat.requestPermissions(this,
                permissions,
                LOCATION_PERMISSION_REQUEST_CODE);
        }
    } else {
        ActivityCompat.requestPermissions(this,
            permissions,
            LOCATION_PERMISSION_REQUEST_CODE);
    }
}

private void initMap() {
    SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.map);

    mapFragment.getMapAsync(Maps.this);
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull
String[] permissions, @NonNull int[] grantResults) {
    mLocationPermissionGranted = false;

    switch (requestCode) {
        case LOCATION_PERMISSION_REQUEST_CODE: {
            if (grantResults.length > 0) {
                for (int i = 0; i < grantResults.length; i++) {
                    if (grantResults[i] !=
PackageManager.PERMISSION_GRANTED) {
                        mLocationPermissionGranted = false;
                        return;
                    }
                }
                mLocationPermissionGranted = true;
                initMap();
            }
        }
    }
}

@Override
public void onMapReady(GoogleMap googleMap) {

```

```

        Toast.makeText(getApplicationContext(), "Maps is Ready",
        Toast.LENGTH_SHORT).show();
        mMap = googleMap;

        if (mLocationPermissionGranted) {
            getDeviceLocation();

            if (ActivityCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_FINE_LOCATION)
                != PackageManager.PERMISSION_GRANTED &&
            ActivityCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_COARSE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
                return;
            }
            mMap.setMyLocationEnabled(true);

            getAllData();
            //addAllMarker();
        }
    }

    private void addAllMarker() {
        for(PlaceInfo place: arrayList){
            MarkerOptions marker = new MarkerOptions()
                .position(new LatLng(place.getLatitude(),
            place.getLongitude()))
                .title(place.getAddress())
                .snippet("PH Tanah: " + place.getPhValue());

            mMap.addMarker(marker);
        }

        Log.d(TAG, "addAllMarker: Success!");
    }

    private void getAllData(){
        mCollectionReference.get()
            .addOnSuccessListener(this, new
            OnSuccessListener<QuerySnapshot>() {
                @Override
                public void onSuccess(QuerySnapshot documentSnapshots) {
                    Log.d(TAG, "onSuccess: Success to retrieve data");

                    if(documentSnapshots.isEmpty()){
                        Log.d(TAG, "onSuccess: List is Empty");
                    } else {

```

```

        List<PlaceInfo> places =
documentSnapshots.toObject(PlaceInfo.class);
        arrayList.addAll(places);

        Log.d(TAG, "onSuccess: " + arrayList);
        addAllMarker();
    }
}
}).addOnFailureListener(this, new OnFailureListener() {
@Override
public void onFailure(@NonNull Exception e) {
    Log.d(TAG, "Failure to get data");
}
});
/*mCollectionReference.get()
.addOnCompleteListener(this, new
OnCompleteListener<QuerySnapshot>() {
@Override
public void onComplete(@NonNull Task<QuerySnapshot>
task) {
    if(task.isSuccessful()){
        DocumentSnapshot documentSnapshot =
task.getResult().getDocuments().get(0);
        PlaceInfo placeInfo =
documentSnapshot.toObject(PlaceInfo.class);

        Toast.makeText(getApplicationContext(),
placeInfo.getAddress(), Toast.LENGTH_LONG).show();
    }
}
});
*/
/*mFirestore.collection("PHAnalyst").get()
.addOnSuccessListener(new
OnSuccessListener<QuerySnapshot>() {
@Override
public void onSuccess(QuerySnapshot documentSnapshots) {
    if(documentSnapshots.isEmpty()){
        Toast.makeText(getApplicationContext(), "Data
kosong", Toast.LENGTH_LONG).show();
    } else {
        arrayList = (ArrayList<PlaceInfo>)
documentSnapshots.toObject(PlaceInfo.class);
        PlaceInfo place = arrayList.get(0);

        Toast.makeText(getApplicationContext(), "Place: " +
place.getAddress(), Toast.LENGTH_LONG).show();
    }
}
}

```

```

    }
    }).addOnFailureListener(this, new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(getApplicationContext(), "Gagal
mendapatkan data!", Toast.LENGTH_LONG).show();
        }
    });*/
}
}

```

2.3 MyPagerAdapter.java

```

package com.meivaldi.phanalyst;

import android.support.v4.view.PagerAdapter;
import android.support.v4.view.ViewPager;
import android.view.View;
import android.view.ViewGroup;

import java.util.ArrayList;

/**
 * Created by root on 26/03/18.
 */

public class MyPagerAdapter extends PagerAdapter {

    private ArrayList<View> views = new ArrayList<View>();

    @Override
    public int getItemPosition(Object object) {
        int index = views.indexOf (object);
        if (index == -1)
            return POSITION_NONE;
        else
            return index;
    }

    @Override
    public Object instantiateItem(ViewGroup container, int position) {
        View v = views.get (position);
        container.addView (v);
        return v;
    }

    @Override

```

```

public void destroyItem(ViewGroup container, int position, Object object) {
    container.removeView (views.get (position));
}

@Override
public int getCount() {
    return views.size();
}

@Override
public boolean isViewFromObject(View view, Object object) {
    return view == object;
}

public int addView(View v){
    return addView(v, views.size());
}

public int addView(View v, int position) {
    views.add(position, v);
    return position;
}

public int removeView (ViewPager pager, View v)
{
    return removeView (pager, views.indexOf(v));
}

public int removeView (ViewPager pager, int position)
{
    pager.setAdapter (null);
    views.remove (position);
    pager.setAdapter (this);

    return position;
}

public View getView(int position){
    return views.get(position);
}
}

```

2.4 PlaceInfo.java

```
package com.meivaldi.phanalyst;

/**
 * Created by root on 25/03/18.
 */

import android.net.Uri;

import com.google.android.gms.maps.model.LatLng;

/**
 * Created by User on 10/2/2017.
 */

public class PlaceInfo {

    private String address;
    private double latitude;
    private double longitude;
    private String phValue;

    public PlaceInfo(String address, double latitude,
                     double longitude, String value) {
        this.address = address;
        this.latitude = latitude;
        this.longitude = longitude;
        this.phValue = value;
    }

    public PlaceInfo() {

    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }

    public double getLatitude() {
        return latitude;
    }

    public void setLatitude(double latitude) {
```

```

        this.latitude = latitude;
    }

    public double getLongitude() {
        return longitude;
    }

    public void setLongitude(double longitude) {
        this.longitude = longitude;
    }

    public String getPhValue() {
        return phValue;
    }

    public void setPhValue(String phValue) {
        this.phValue = phValue;
    }

    @Override
    public String toString() {
        return "address=" + address + "\" +
            ", latitude=" + latitude + "\" +
            ", longitude=" + longitude + "\" +
            ", phValue=" + phValue + "\";
    }
}

```

2.5 ResultActivity.java

```

package com.meivaldi.phanalyst;

import android.Manifest;
import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.os.Handler;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;

```



```

import android.support.v4.view.ViewPager;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.firestore.FirebaseFirestore;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.UUID;

public class ResultActivity extends AppCompatActivity {

    private MyPagerAdapter pagerAdapter = null;

    private static final String LATITUDE_KEY = "latitude";
    private static final String LONGITUDE_KEY = "longitude";
    private static final String VALUE_KEY = "phValue";
    private static final String ADDRESS_KEY = "address";

    private Button map, saveValue;

    private TextView pHLabel;
    private ViewPager suggestionPlant = null;

```

```

private Handler bluetoothIn;

private final int handlerState = 0;
private BluetoothAdapter btAdapter = null;
private BluetoothSocket btSocket = null;
private StringBuilder recDataString = new StringBuilder();

private static String address;
private ConnectedThread mConnectedThread;
private static final UUID BTMODULEUUID =
UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

private Boolean mLocationPermissionGranted = false;
private FusedLocationProviderClient mFusedLocationProviderClient;

private static final String FINE_LOCATION =
android.Manifest.permission.ACCESS_FINE_LOCATION;
private static final String COARSE_LOCATION =
android.Manifest.permission.ACCESS_COARSE_LOCATION;
private static final int LOCATION_PERMISSION_REQUEST_CODE
= 1234;

private static final String TAG = "ResultActivity";

private FirebaseFirestore mFirestore = FirebaseFirestore.getInstance();

private Geocoder geoCoder;
private List<Address> addresses;
private ProgressDialog progressDialog;

private RelativeLayout baseLayout;
private LinearLayout dot;
private LayoutInflater inflater;
private List<RelativeLayout> previousLayout = new ArrayList<>();

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_result);

    getLocationPermission();

    suggestionPlant = (ViewPager) findViewById(R.id.suggestionPlant);
    pHLabel = (TextView) findViewById(R.id.pHValue);
    map = (Button) findViewById(R.id.seeMap);
    saveValue = (Button) findViewById(R.id.simpanNilai);
    dot = (LinearLayout) findViewById(R.id.dot_menu);

```

```

progressDialog = new ProgressDialog(this);
progressDialog.setMessage("Tunggu Sebentar");
progressDialog.setIcon(R.drawable.sensor);
progressDialog.setTitle("Sedang Memroses...");

map.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(getApplicationContext(), Maps.class));
    }
});

saveValue.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        progressDialog.show();
        getLatLng();
    }
});

bluetoothIn = new Handler() {
    public void handleMessage(android.os.Message msg) {
        if (msg.what == handlerState) {
            String readMessage = (String) msg.obj;
            recDataString.append(readMessage);

            int endOfLineIndex = recDataString.indexOf("~");
            if (endOfLineIndex > 0) {
                if (recDataString.charAt(0) == '#') {
                    String sensor = recDataString.substring(1, 5);
                    pHLabel.setText(sensor);
                    dot.setVisibility(View.VISIBLE);
                }
                recDataString.delete(0, recDataString.length());
            }
        }
    }
};

pHLabel.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        dot.setVisibility(View.INVISIBLE);
        String text = pHLabel.getText().toString();
        float ph = Float.parseFloat(text);
        checkLayout(ph);
    }
}

```

```

});

btAdapter = BluetoothAdapter.getDefaultAdapter();
checkBTState();

pagerAdapter = new MyPagerAdapter();
suggestionPlant = (ViewPager) findViewById(R.id.suggestionPlant);
suggestionPlant.setAdapter(pagerAdapter);

inflater = this.getLayoutInflater();
baseLayout = (RelativeLayout) inflater.inflate(R.layout.plant_default,
null);
previousLayout.add(baseLayout);
pagerAdapter.addView(baseLayout, 0);
pagerAdapter.notifyDataSetChanged();
}

private void checkLayout(float pH) {
    removeLayout();
    if(pH == 6.0){
        previousLayout.clear();
        RelativeLayout firstLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_cabbage, null);
        RelativeLayout secondLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_banana, null);
        RelativeLayout thirdLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_broccoli, null);
        previousLayout.add(firstLayout);
        previousLayout.add(secondLayout);
        previousLayout.add(thirdLayout);
        pagerAdapter.addView(firstLayout, 0);
        pagerAdapter.addView(secondLayout, 1);
        pagerAdapter.addView(thirdLayout, 2);
        pagerAdapter.notifyDataSetChanged();
    } else if(pH >= 5.5 && pH < 6.0){
        previousLayout.clear();
        RelativeLayout firstLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_carrot, null);
        RelativeLayout secondLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_melon, null);
        RelativeLayout thirdLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_mint, null);
        previousLayout.add(firstLayout);
        previousLayout.add(secondLayout);
        previousLayout.add(thirdLayout);
        pagerAdapter.addView(firstLayout, 0);
        pagerAdapter.addView(secondLayout, 1);
        pagerAdapter.addView(thirdLayout, 2);
    }
}

```

```

        pagerAdapter.notifyDataSetChanged();
    } else if(pH >= 5.0 && pH < 5.5){
        previousLayout.clear();
        RelativeLayout firstLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_garlic, null);
        RelativeLayout secondLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_pakcoy, null);
        RelativeLayout thirdLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_papaya, null);
        previousLayout.add(firstLayout);
        previousLayout.add(secondLayout);
        previousLayout.add(thirdLayout);
        pagerAdapter.addView(firstLayout, 0);
        pagerAdapter.addView(secondLayout, 1);
        pagerAdapter.addView(thirdLayout, 2);
        pagerAdapter.notifyDataSetChanged();
    } else if(pH >= 4.5 && pH < 5.0){
        previousLayout.clear();
        RelativeLayout firstLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_onion, null);
        RelativeLayout secondLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_pineapple, null);
        RelativeLayout thirdLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_potato, null);
        previousLayout.add(firstLayout);
        previousLayout.add(secondLayout);
        previousLayout.add(thirdLayout);
        pagerAdapter.addView(firstLayout, 0);
        pagerAdapter.addView(secondLayout, 1);
        pagerAdapter.addView(thirdLayout, 2);
        pagerAdapter.notifyDataSetChanged();
    } else if(pH >= 4.0 && pH < 4.5){
        previousLayout.clear();
        RelativeLayout firstLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_watermelon, null);
        RelativeLayout secondLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_spinach, null);
        RelativeLayout thirdLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_radish, null);
        RelativeLayout fourthLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_strawberry, null);
        previousLayout.add(firstLayout);
        previousLayout.add(secondLayout);
        previousLayout.add(thirdLayout);
        previousLayout.add(fourthLayout);
        pagerAdapter.addView(firstLayout, 0);
        pagerAdapter.addView(secondLayout, 1);
        pagerAdapter.addView(thirdLayout, 2);

```

```

        pagerAdapter.addView(fourthLayout, 3);
        pagerAdapter.notifyDataSetChanged();
    } else if(pH != 0.0){
        previousLayout.clear();
        Toast.makeText(getApplicationContext(), "Nilai pH diluar
jangkauan!", Toast.LENGTH_LONG).show();
        RelativeLayout defaultLayout = (RelativeLayout)
inflater.inflate(R.layout.plant_default, null);
        pagerAdapter.addView(defaultLayout, 0);
        previousLayout.add(defaultLayout);
        pagerAdapter.notifyDataSetChanged();
    }
}

private void removeLayout() {
    for(RelativeLayout layout: previousLayout){
        pagerAdapter.removeView(suggestionPlant, layout);
        pagerAdapter.notifyDataSetChanged();
    }
}

private void getLatLng() {
    mFusedLocationProviderClient =
LocationServices.getFusedLocationProviderClient(ResultActivity.this);

    try {
        if (mLocationPermissionGranted) {
            final Task location =
mFusedLocationProviderClient.getLastLocation();
            location.addOnCompleteListener(new OnCompleteListener() {
                @Override
                public void onComplete(@NonNull Task task) {
                    if (task.isSuccessful()) {
                        Log.d(TAG, "onComplete: found location!");
                        Location currentLocation = (Location) task.getResult();

                        geoCoder = new Geocoder(getApplicationContext(),
Locale.getDefault());

                        try {
                            addresses =
geoCoder.getFromLocation(currentLocation.getLatitude(),
currentLocation.getLongitude(), 1);
                        } catch (IOException e) {
                            e.printStackTrace();
                        }

                        String myAddress = addresses.get(0).getAddressLine(0);

```

```

        Map<String, Object> dataToSave = new
HashMap<String, Object>();
        dataToSave.put(ADDRESS_KEY, myAddress);
        dataToSave.put(LATITUDE_KEY,
currentLocation.getLatitude());
        dataToSave.put(LONGITUDE_KEY,
currentLocation.getLongitude());
        dataToSave.put(VALUE_KEY,
pHLabel.getText().toString());

mFirestore.collection("PHAnalyst").document(myAddress).set(dataToSave)
e)
        .addOnCompleteListener(ResultActivity.this, new
OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void>
task) {
                if(task.isSuccessful()){
                    progressDialog.hide();
                    Toast.makeText(getApplicationContext(),
"Berhasil Menyimpan data", Toast.LENGTH_SHORT).show();
                } else {
                    progressDialog.hide();
                    Toast.makeText(getApplicationContext(),
"Gagal Menyimpan data", Toast.LENGTH_SHORT).show();
                }
            }
        });
    } else {
        Log.d(TAG, "onComplete: current location is null!");
        Toast.makeText(ResultActivity.this, "Unable to get
current location", Toast.LENGTH_SHORT).show();
    }
}
});
}
} catch (SecurityException e) {
    Log.d(TAG, "getDeviceLocation: Security Exception: " +
e.getMessage());
}
}

private void getLocationPermission() {
    String[] permissions = {
        android.Manifest.permission.ACCESS_COARSE_LOCATION,
        Manifest.permission.ACCESS_FINE_LOCATION
    }
}

```

```

    };

    if
    (ContextCompat.checkSelfPermission(this.getApplicationContext(),
        FINE_LOCATION) ==
    PackageManager.PERMISSION_GRANTED) {
        if
        (ContextCompat.checkSelfPermission(this.getApplicationContext(),
            COARSE_LOCATION) ==
        PackageManager.PERMISSION_GRANTED) {
            mLocationPermissionGranted = true;
        } else {
            ActivityCompat.requestPermissions(this,
                permissions,
                LOCATION_PERMISSION_REQUEST_CODE);
        }
    } else {
        ActivityCompat.requestPermissions(this,
            permissions,
            LOCATION_PERMISSION_REQUEST_CODE);
    }
}

@Override
protected void onResume() {
    super.onResume();

    Intent intent = getIntent();

    address =
    intent.getStringExtra(MainActivity.EXTRA_DEVICE_ADDRESS);

    BluetoothDevice device = btAdapter.getRemoteDevice(address);

    try {
        btSocket = createBluetoothSocket(device);
    } catch (IOException e) {
        Toast.makeText(getBaseContext(), "Socket creation failed",
            Toast.LENGTH_LONG).show();
    }

    try {
        btSocket.connect();
    } catch (IOException e) {
        try {
            btSocket.close();
        } catch (IOException e2) {

```



```

    }
}
mConnectedThread = new ConnectedThread(btSocket);
mConnectedThread.start();

}

private void checkBTState() {

    if (btAdapter == null) {
        Toast.makeText(getApplicationContext(), "Device does not support
bluetooth", Toast.LENGTH_LONG).show();
    } else {
        if (btAdapter.isEnabled()) {
        } else {
            Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, 1);
        }
    }
}

private class ConnectedThread extends Thread {
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;

    public ConnectedThread(BluetoothSocket socket) {
        InputStream tmpIn = null;
        OutputStream tmpOut = null;

        try {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {
        }

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }

    public void run() {
        byte[] buffer = new byte[256];
        int bytes;

        while (true) {
            try {

```

```

        bytes = mmInStream.read(buffer);
        String readMessage = new String(buffer, 0, bytes);
        bluetoothIn.obtainMessage(handlerState, bytes, -1,
readMessage).sendToTarget();
    } catch (IOException e) {
        break;
    }
}
}

private BluetoothSocket createBluetoothSocket(BluetoothDevice
device) throws IOException {

    return
device.createRfcommSocketToServiceRecord(BTMODULEUUID);
}

@Override
protected void onPause() {
    super.onPause();
    try {
        btSocket.close();
    } catch (IOException e2) {

    }
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull
String[] permissions, @NonNull int[] grantResults) {
    mLocationPermissionGranted = false;

    switch (requestCode) {
        case LOCATION_PERMISSION_REQUEST_CODE: {
            if (grantResults.length > 0) {
                for (int i = 0; i < grantResults.length; i++) {
                    if (grantResults[i] !=
PackageManager.PERMISSION_GRANTED) {
                        mLocationPermissionGranted = false;
                        return;
                    }
                }
                mLocationPermissionGranted = true;
            }
        }
    }
}
}

```

```

    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_main, menu);

        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.exit:
                finish();
            default:
                return super.onOptionsItemSelected(item);
        }
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();

        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setMessage("Do You Want To
Disconnect?").setCancelable(false)
            .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    try {
                        btSocket.close();
                        previousLayout.clear();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                    finish();
                }
            }).setNegativeButton("No", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    dialog.cancel();
                }
            }).show();
    }
}

```

2.6 Splash.java

```

package com.meivaldi.phanalyst;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Window;
import android.view.WindowManager;

public class Splash extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);

        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
            WindowManager.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_splash);

        Thread thread = new Thread(new Runnable() {
            @Override
            public void run() {
                try{
                    Thread.sleep(3000);
                } catch (InterruptedException e){
                    e.printStackTrace();
                } finally {
                    Intent intent = new
Intent("com.meivaldi.phanalyst.WELCOMEACTIVITY");
                    startActivity(intent);
                }
            }
        });

        thread.start();
    }

    @Override
    protected void onPause() {
        super.onPause();
        finish();
    }
}

```

2.7 WelcomeActivity.java

```

package com.meivaldi.phanalyst;

import android.content.Context;
import android.content.Intent;
import android.graphics.Color;
import android.os.Build;
import android.support.v4.view.PagerAdapter;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.Html;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.TextView;

public class WelcomeActivity extends AppCompatActivity {

    private ViewPager viewPager;
    private MyViewPagerAdapter myViewPagerAdapter;
    private LinearLayout dotsLayout;
    private TextView[] dots;
    private int[] layouts;
    private Button btnSkip, btnNext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_welcome);

        // Making notification bar transparent
        if (Build.VERSION.SDK_INT >= 21) {

getWindow().getDecorView().setSystemUiVisibility(View.SYSTEM_UI_
FLAG_LAYOUT_STABLE |
View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN);
        }

        setContentView(R.layout.activity_welcome);

        viewPager = (ViewPager) findViewById(R.id.view_pager);
        dotsLayout = (LinearLayout) findViewById(R.id.layoutDots);

```

```

btnSkip = (Button) findViewById(R.id.btn_skip);
btnNext = (Button) findViewById(R.id.btn_next);

// layouts of all welcome sliders
// add few more layouts if you want
layouts = new int[]{
    R.layout.intro_slide1,
    R.layout.intro_slide2,
    R.layout.intro_slide3};

// adding bottom dots
addBottomDots(0);

// making notification bar transparent
changeStatusBarColor();

myViewPagerAdapter = new MyViewPagerAdapter();
viewPager.setAdapter(myViewPagerAdapter);

viewPager.addOnPageChangeListener(viewPagerPageChangeListener);

btnSkip.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        launchActivity();
    }
});

btnNext.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // checking for last page
        // if last page home screen will be launched
        int current = getItem(+1);
        if (current < layouts.length) {
            // move to next screen
            viewPager.setCurrentItem(current);
        } else {
            launchActivity();
        }
    }
});
}

private void launchActivity() {
    Intent intent = new Intent(getApplicationContext(),
MainActivity.class);

```

```

        startActivity(intent);
    }

    private void addBottomDots(int currentPage) {
        dots = new TextView[layouts.length];

        int[] colorsActive =
        getResources().getIntArray(R.array.array_dot_active);
        int[] colorsInactive =
        getResources().getIntArray(R.array.array_dot_inactive);

        dotsLayout.removeAllViews();
        for (int i = 0; i < dots.length; i++) {
            dots[i] = new TextView(this);
            dots[i].setText(Html.fromHtml("&#8226;"));
            dots[i].setTextSize(35);
            dots[i].setTextColor(colorsInactive[currentPage]);
            dotsLayout.addView(dots[i]);
        }

        if (dots.length > 0)
            dots[currentPage].setTextColor(colorsActive[currentPage]);
    }

    private int getItem(int i) {
        return viewPager.getCurrentItem() + i;
    }

    // viewpager change listener
    ViewPager.OnPageChangeListener viewPagerPageChangeListener =
    new ViewPager.OnPageChangeListener() {

        @Override
        public void onPageSelected(int position) {
            addBottomDots(position);

            // changing the next button text 'NEXT' / 'GOT IT'
            if (position == layouts.length - 1) {
                // last page. make button text to GOT IT
                btnNext.setText(getString(R.string.start));
                btnSkip.setVisibility(View.GONE);
            } else {
                // still pages are left
                btnNext.setText(getString(R.string.next));
                btnSkip.setVisibility(View.VISIBLE);
            }
        }
    }
}

```

```

@Override
public void onPageScrolled(int arg0, float arg1, int arg2) {

}

@Override
public void onPageScrollStateChanged(int arg0) {

}
};

/**
 * Making notification bar transparent
 */
private void changeStatusBarColor() {
    if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.LOLLIPOP) {
        Window window = getWindow();

window.addFlags(WindowManager.LayoutParams.FLAG_DRAWS_SYSTEM_BAR_BACKGROUNDS);
        window.setStatusBarColor(Color.TRANSPARENT);
    }
}

/**
 * View pager adapter
 */
public class MyViewPagerAdapter extends PagerAdapter {
    private LayoutInflater inflater;

    public MyViewPagerAdapter() {
    }

    @Override
    public Object instantiateItem(ViewGroup container, int position) {
        inflater = (LayoutInflater)
getService(Context.LAYOUT_INFLATER_SERVICE);

        View view = inflater.inflate(layouts[position], container,
false);
        container.addView(view);

        return view;
    }

    @Override

```



```

    public int getCount() {
        return layouts.length;
    }

    @Override
    public boolean isViewFromObject(View view, Object obj) {
        return view == obj;
    }

    @Override
    public void destroyItem(ViewGroup container, int position, Object
object) {
        View view = (View) object;
        container.removeView(view);
    }

    @Override
    protected void onPause() {
        super.onPause();
        finish();
    }
}

```

CURRICULUM VITAE

DATA DIRI



Nama Lengkap : Rido Meivaldi
 Nama Panggilan : Rido
 Tempat/Tanggal Lahir : Batang Kuis, 31 Mei 1996
 Jenis Kelamin : Laki - laki
 Agama : Islam
 Kebangsaan : Indonesia
 Alamat : Dsn I Jl. Kebun Sayur Desa Tanjung Sari Kecamatan Batang Kuis
 Nomor HP : +6285761806490
 E-mail : meivaldi@gmail.com

RIWAYAT PENDIDIKAN

S1 Ilmu Komputer
Universitas Sumatera Utara, Medan
2014-2018

Sekolah Menengah Atas
MA NEGERI 2 MODEL MEDAN
2011-2014

Sekolah Menengah Pertama
SMP NEGERI 1 BATANG KUIS
2008-2011

Sekolah Dasar
SD NEGERI 104230 BATANG KUIS
2002-2008

SEMINAR /WORKSHOP /PELATIHAN

No	Nama Kegiatan	Jenis Kegiatan	Tahun
1	What Will You be?	Seminar	2014
2	Pelatihan Dasar Organisasi (PDO)	Pelatihan	2015
3	Seminar Kewirausahaan	Seminar	2015

KEAHLIAN

Bahasa Pemrograman : Java,C++,C,Android, PHP, Laravel, Vue.JS, React JS
DMBS : MySQL
Software : Android Studio,Office Application, Proteus

PENGALAMAN ORGANISASI & KEPANITIAAN

No	Organisasi	Posisi	Tahun
1	UKM Robotika Sikonek USU	Anggota	2015 - 2016
2	IKLC USU	Anggota	2016
3	IMILKOM USU	Anggota Departemen Wawasan Kontemporer	2016 - 2017
4	UKM Beladiri USU	Anggota	2017

PROJECT YANG PERNAH DIKERJAKAN

1. PH Analyst
2. Lock PDF
3. Combur Chat Messenger
4. StegoImage Creator
5. Image Secure
6. Plant Watcher