TUGAS AKHIR – KI1502

RANCANG BANGUN SISTEM MONITORING KUALITAS AIR TERDISTRIBUSI BERBASIS SENSOR-CLOUD DENGAN VIRTUALISASI BERBASIS DOCKER

Adiwinoto Saptorenggo
NRP 05111440000164

Dosen Pembimbing I
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

Dosen Pembimbing II
Ir. F.X. Arunanto, M.Sc

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – KI1502

RANCANG BANGUN SISTEM MONITORING KUALITAS AIR TERDISTRIBUSI BERBASIS SENSOR-CLOUD DENGAN VIRTUALISASI BERBASIS DOCKER

ADIWINOTO SAPTORENGGO
NRP 05111440000164

Dosen Pembimbing I
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

Dosen Pembimbing II
Ir. F.X. Arunanto, M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESES – KI1502

DISTRIBUTED SENSOR-CLOUD BASED WATER QUALITY MONITORING SYSTEM WITH DOCKER BASED VIRTUALISATION

ADIWINOTO SAPTORENGGO
NRP 05111440000164

Supervisor I
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

Supervisor II
Ir. F.X. Arunanto, M.Sc.

DEPARTMENT OF INFORMATICS
Faculty of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN SISTEM MONITORING KUALITAS AIR TERDISTRIBUSI BERBASIS SENSOR-CLOUD DENGAN VIRTUALISASI BERBASIS DOCKER

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Komputasi Berbasis Jaringan
Sistem Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh
ADIWINOTO SAPTORENGGO
NRP. 05111440000164

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Waskitho Wibisono,
S.Kom., M.Eng., Ph.D.
NIP: 19741022 200003 1 001 ..... (pembimbing 1)

Ir. F.X. ARUNANTO,
M.Sc.
NIP: 19570101 198303 1 004 ..... (pembimbing 2)

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI
DEPARTEMEN TEKNIK INFORMATIKA

SURABAYA
MARET, 2018

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN SISTEM MONITORING KUALITAS AIR TERDISTRIBUSI BERBASIS SENSOR-CLOUD DENGAN VIRTUALISASI BERBASIS DOCKER

Nama : **Adiwinoto Saptorenggo**
NRP : **05111440000164**
Departemen : **Informatika – FTIK**
Dosen Pembimbing I : **Waskitho Wibisono, S.Kom., M.Eng., Ph.D.**
Dosen Pembimbing II : **Ir. F.X. ARUNANTO, M.Sc.**

Abstrak

Lembaga Ilmu Pengetahuan Indonesia menyebutkan kualitas air di Indonesia terus menurun. Kepala LIPI, Iskandar Zulkarnain menyatakan menurunnya kualitas air terutama terjadi di kota-kota besar. Air permukaan sungai terus mengalami pencemaran dan rusak. Penyebabnya adalah eksploitasi air oleh manusia, limbah rumah tangga, dan kegiatan industry.

Akan dibuat juga aplikasi website yang akan memonitoring kualitas air, berdasarkan data – data yang di dapat dari node – node sensor yang disebar. Hasil yang didapat akan menjaga kualitas air agar segera dapat di tanggulangi jikalau ada berbagai macam gangguan, seperti polutan, yang akan mencemarkan air.

Monitoring Kualitas Air dapat memberikan layanan monitoring secara detail, real-time, dan tepat. Database dari masing-masing cluster air akan terisolir satu sama lain (keamanan terjaga). Jika terjadi gangguan di salah satu container, container lain tetap tersedia (ketersediaan terjaga). User bisa mengakses website cukup baik dengan error sebesar 1.00%. Pengiriman data ke server sangat mempengaruhi daya yang dikeluarkan alat sensor.

Kata kunci: ***Docker, Sensor-cloud, Virtualisasi.***

[Halaman ini sengaja dikosongkan]

DESIGN AND IMPLEMENTATION OF DISTRIBUTED SENSOR-CLOUD BASED WATER QUALITY MONITORING SYSTEM USING DOCKER VIRTUALIZATION

Student Name	**: Adiwinoto Saptorenggo**
NRP	**: 05111440000164**
Major	**: Informatics – FTIK**
Supervisor I	**: Waskitho Wibisono, S.Kom., M.Eng., Ph.D.**
Supervisor II	**: Ir. F.X. Arunanto, M.Sc.**

Abstract

The Indonesian Institute of Sciences says air quality in Indonesia continues to decline. The Head of LIPI, Iskandar Zulkarnain stated that the decline in air quality occurred in big cities. The river's water effect continues to be contaminated and damaged. The cause is the exploitation of air by humans, household waste, and industrial activities.

There will also be a website application that will monitor air quality, based on data that can be obtained from sensor nodes that are deployed. The results obtained will maintain air quality so that it can be captured immediately in various types of disturbances, such as pollutants, which pollute the air.

Water Quality Monitoring can provide detailed, real-time, and precise monitoring services. The database of each air cluster will be isolated from each other (confidential). In case of interference in one container, another container is still available (director). Users can access the website quite well with an error of 1.00%. Delivery of data to the server greatly affects the power of the device's sensor.

Keywords: ***Docker, Sensor-cloud, Virtualization.***

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Segala puji syukur penulis kepada Allah SWT atas segala nikmat dan karunia-Nya, sehingga tugas akhir berjudul "Rancang Bangun Sistem Monitoring Kualitas Air Terdistribusi Berbasis Sensor-Cloud dengan Virtualisasi Berbasis Docker" ini dapat selesai sesuai dengan waktu yang telah ditentukan.

Pengerjaan tugas akhir ini menjadi sebuah sarana untuk penulis memperdalam ilmu yang telah didapatkan di Institut Teknologi Sepuluh Nopember Surabaya, khususnya dalam disiplin ilmu Teknik Informatika. Terselesaikannya buku tugas akhir ini tidak terlepas dari bantuan dan dukungan semua pihak. Pada kesempatan kali ini penulis ingin mengucapkan terima kasih kepada:

1. Bapak, Ibu dan keluarga yang selalu memberikan dukungan berupa doa dan nutrisi selama proses pengerjaan Tugas Akhir.
2. Bapak Waskitho dan Bapak Arun selaku dosen pembimbing yang telah bersedia meluangkan waktu selama proses pengerjaan Tugas Akhir.
3. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS yang banyak memberikan ilmu dan bimbingan bagi penulis.
4. Seluruh staf dan karyawan FTIf ITS yang banyak memberikan kelancaran administrasi akademik kepada penulis.
5. Teman-teman 'Rantau' dan TC Angkatan 2014 yang selalu mendukung selama proses pengerjaan tugas akhir.
6. Serta semua pihak yang turut membantu penulis dalam menyelesaikan tugas akhir ini.

Penulis menyadari bahwa tugas akhir ini masih memiliki banyak sekali kekurangan. Dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Januari 2018

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pengerjaan Tugas Akhir, dan sistematika penulisan Tugas Akhir.

1.1 Latar Belakang

Lembaga Ilmu Pengetahuan Indonesia menyebutkan kualitas air di Indonesia terus menurun. Kepala LIPI, Iskandar Zulkarnain menyatakan menurunnya kualitas air terutama terjadi di kota-kota besar. Air permukaan sungai terus mengalami pencemaran dan rusak. Penyebabnya adalah eksploitasi air oleh manusia, limbah rumah tangga, dan kegiatan industry.

Salah satu cara untuk menanggulangi masalah tersebut, dengan cara memonitoring kualitas air di daerah tertentu. Monitoring dilakukan dengan pemasangan node-node sensor di daerah yang diamati. Data – data yang didapat dari node – node sensor tersebut, nantinya dikirim ke dalam web server. Setelah itu akan dilakukan virtualisasi berbasis docker agar lebih cepat dan efisien.

Akan dibuat juga aplikasi website yang akan memonitoring kualitas air, berdasarkan data – data yang di dapat dari node – node sensor yang disebar. Hasil yang didapat akan menjaga kualitas air agar segera dapat di tanggulangi jikalau ada berbagai macam gangguan, seperti polutan, yang akan mencemarkan air.

1.2 Rumusan Permasalahan

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana cara mengambil data sensor?
2. Bagaimana cara mengirim data sensor ke server?

3. Bagaimana menerapkan virtualiasi docker pada masing2 kolam?
4. Bagaimana membangun aplikasi web monitoring pada masing - masing kolam?
5. Bagaimana membangun aplikasi web monitoring air sebagai wahana untuk memonitor kualitas air?

1.3 Batasan Permasalahan

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Air yang diamati adalah air yan ada pada kolam/danau.
2. Microcontroller yang digunakan adalah Arduino Yun.

1.4 Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah sebagai berikut:

1. Mengetahui kualitas air dari data – data yang di dapat dari sensor.
2. Memberikan efek isolasi dan ketersediaan kepada masing-masing container pada virtualisasi.
3. Merancang website Monitoring Air sebagai aplikasi untuk memonitor kualitas air.

1.5 Manfaat

Penelitian ini diharapkan dapat memiliki manfaat untuk menghasilkan sebuah sistem monitoring kualitas air terdistribusi berbasis sensor-cloud dengan virtualisasi berbasis docker.

1.6 Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu sebagai berikut:

1.6.1 Penyusunan Proposal

Tahap awal pengerjaan tugas akhir ini dimulai dengan penyusunan Proposal Tugas Akhir. Proposal Tugas Akhir ini berisi tentang perencanaan perancangan website monitoring

beserta virtualisasi berbasis docker sebagai media penyimpan database dari masing-masing *cluster* dan juga sensor yang digunakan untuk mendeteksi kualitas air beserta mikrokontroller arduino.

Proposal Tugas Akhir ini terdiri dari deskripsi pendahuluan yang berisi penjabaran latar belakang dan rumusan masalah yang menjadi dasar pengerjaan tugas akhir, batasan masalah dalam perancangan jaringan, serta tujuan dan manfaat yang diharapkan dapat tercapai dengan perancangan jaringan nirkabel ini. Pada proposal Tugas Akhir ini juga menyertakan tinjauan pustaka yang menjelaskan berbagai teori yang menjadi dasar pembuatan tugas akhir ini, yaitu membuat sistem monitoring kualitas air berbasis *sensor-cloud* dengan *virtualisasi* berbasis *docker*.

1.6.2 Studi Literatur

Studi literatur yang dilakukan dalam pengerjaan Tugas Akhir ini dilakukan dengan mencari informasi dari sumber-sumber terkait yang membahas mengenai virtualisasi berbasis docker dan modul-modul sensor Arduino. Selain itu, juga dilakukan studi literatur mengenai proses pengiriman data dari Arduino ke server menggunakan web servis.

Hal-hal lain yang dipelajari dalam studi literatur meliputi mikrokontroler Arduino, pencarian modul sensor yang cocok, pemrograman Arduino, pemrograman Python, basis data, pemrograman web dan lain sebagainya.

1.6.3 Analisis dan Desain Perangkat Lunak

Tahap ini meliputi perancangan dan analisis sistem berdasarkan studi literatur. Berdasarkan konsep teknologi dari perangkat lunak yang ada saat ini, dilakukan perancangan sistem yang akan dibangun. Langkah-langkah pengerjaan juga didefinisikan pada tahap ini.

Pada tahapan ini dibuat prototipe dari sistem, yang merupakan rancangan dasar dari sistem yang akan dibuat. Selain itu, dalam tahap ini juga dilakukan desain sistem dan proses-proses yang ada.

1.6.4 Implementasi Perangkat Lunak

Implementasi Perangkat Lunak merupakan tahapan untuk membangun rancangan program yang telah dibuat. Pada tahapan ini merealisasikan apa yang terdapat pada tahapan sebelumnya, sehingga menjadi sebuah program yang sesuai dengan apa yang telah direncanakan.

1.6.5 Pengujian dan Evaluasi

Pada tahapan ini dilakukan uji coba pada alat yang telah dibuat. Tahapan ini dimaksudkan untuk mengevaluasi tingkat akurasi dan performa dari alat tersebut serta mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

1.6.6 Penyusunan Buku

Pada tahap ini disusun buku sebagai dokumentasi dari pelaksanaan tugas akhir yang mencakup seluruh konsep, dasar teori, implementasi, serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat

1.7 Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan yang menjadi dasar dari pembuatan Tugas Akhir ini.

Bab III Perancangan Perangkat Lunak

Bab ini berisi implementasi dari perancangan perangkat lunak yang telah dibuat pada bab sebelumnya. Implementasi berupa pseudocode dari fungsi utama dan screenshot perangkat lunak.

Bab IV Implementasi

Bab ini membahas implementasi dari desain yang telah dibuat pada bab sebelumnya. Penjelasan berupa kode yang digunakan untuk proses implementasi.

Bab V Hasil Uji Coba dan Evaluasi

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

Bab VI Kesimpulan

Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

Lampiran

Lampiran yang ada berisi kelengkapan – kelengkapan yang diperlukan dalam menyusun buku tugas akhir.

[Halaman ini sengaja dikosongkan]

BAB II
DASAR TEORI

Pada bab ini, dijabarkan tentang penjelasan teori-teori yang berkaitan dengan pokok bahasan tugas akhir. Bab ini juga menjelaskan modul dan alat yang nantinya akan digunakan pada tahap implementasi program. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap alat yang digunakan dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 Jaringan Nirkabel

Jaringan nirkabel atau yang lebih dikenal dengan nama *wireless network* merupakan bidang ilmu yang berkaita dengan komunikasi antar perangkat komputer tanpa menggunakan kabel. Jaringan nirkabel dapat diterapkan untuk komunikasi antar sitem komputer baik dengan jarak dekat (dengan menggunakan *bluetooth*) maupun dengan jarak yang jauh (dengan menggunakan satelit).

Saat ini, ponsel mungkin merupakan bentuk paling umum dari jaringan nirkabel. Setiap orang memiliki satu hari ini, beberapa bahkan anak-anak yang bahkan belum mencapai usia remaja. Telepon yang digunakan tidak hanya sebagai bentuk komunikasi hari ini tetapi juga sebagai metode memperoleh update dan informasi dan sebagai alat hiburan. Satelit adalah bentuk lain dari jaringan nirkabel, memungkinkan kita untuk menonton tv kabel dan untuk menerima program yang sedang ditampilkan langsung di bagian lain dari dunia [1].

Jaringan nirkabel secara umum diatur dalam dokumen IEEE 802.11 dan termasuk di dalamnya Wi-Fi (*Wireless Fidelity)* yang diatur dalam IEEE 802.11b. Protokol IEEE 802.11g adalah versi lain yang memiliki performa lebih cepat [2]. Pada perkembangannya diatur juga teknologi lain dalam IEEE 802.15 untuk WPAN (*Wireless Personal Area Network*). Beberapa diantaranya adalah Bluetooth (diatur dalam IEEE 802.15.1), Zigbee (802.15.4) dan sebagainya.

2.2 Sensor Cloud

Sensor Cloud merupakan istilah yang sering digunakan dengan menerapkan konsep virtualisasi sensor. Virtualisasi sensor merupakan metode untuk menyediakan layanan sensor secara virtual. Sensor virtual yang dibuat juga telah dikelompokkan (cluster) sedemikian rupa sehingga pengguna dapat lebih mudah dalam melakukan pengelolaan data. Perancangan sensor dan kelompok sensor virtual berguna agar pengguna dapat melakukan pemantauan tanpa perlu mengkhawatirkan tentang faktor lokasi dan spesifikasi sensor fisik [3].

2.3 Virtualisasi Data

Dalam ilmu komputer, virtualisasi (bahasa Inggris: virtualization) adalah istilah umum yang mengacu kepada abstraksi dari sumber daya komputer. Definisi lainnya adalah "sebuah teknik untuk menyembunyikan karakteristik fisik dari sumber daya komputer dari bagaimana cara sistem lain, aplikasi atau pengguna berinteraksi dengan sumber daya tersebut. Hal ini termasuk membuat sebuah sumber daya tunggal (seperti server, sebuah sistem operasi, sebuah aplikasi, atau peralatan penyimpanan terlihat berfungsi sebagai beberapa sumber daya logikal; atau dapat juga termasuk definisi untuk membuat beberapa sumber daya fisik (seperti beberapa peralatan penyimpanan atau server) terlihat sebagai satu sumber daya logikal [7]."

Istilah virtualisasi sudah digunakan secara luas sejak 1960-an, dan telah diaplikasikan kepada beberapa aspek komputer—dari keseluruhan sistem komputer sampai sebuah kemampuan atau komponen individu. Secara umum semua teknologi virtualisasi mengacu kepada "menyembunyikan detail teknis" melalui enkapsulasi [7].

No Structured Query Language merupakan istilah yang digunakan pertama kali oleh Carlo Strozzi pada tahun 1998 untuk basis data relasional Strozzi NoSQL opensource-nya. Database ini bersifat relasional, namun istilahnya telah berevolusi dan sejak 2009

lebih dikenal dengan database non-relasional atau non-ACID. Database NoSQL bervariasi dari toko dengan nilai kunci sederhana hingga sistem seperti SQL yang kompleks.

2.4 Docker

Docker adalah salah satu platform yang dibangun berdasarkan teknologi container. Docker merupakan sebuah project open-source yang menyediakan platform terbuka untuk developer maupun sysadmin untuk dapat membangun, mengemas, dan menjalankan aplikasi dimanapun sebagai sebuah wadah (container) yang ringan. Dengan sangat populernya docker, sebagian orang sering menganggap docker adalah sebutan lain untuk container [8].

Docker pertama kali dikembangkan oleh Solomon Hykes sebagai project internal di dotCloud bersama dengan beberapa koleganya seperti Andrea Luzzardi dan Francois-Xavier Bourlet. Perilisan platform ini secara open-source dilakukan pada mei 2013 silam. Docker terus berkembang hingga memiliki ribuan orang yang berkontribusi membuatnya menjadi lebih baik.Berbeda dengan virtualisasi yang mana aplikasi berjalan di atas hypervisor dan guest OS, docker dapat menjalankan aplikasi langsung tanpa kedua hal tadi. Docker juga dilengkapi dengan fitur sandbox yang menjamin pengerjaan pengembang dan sysadmin tidak terganggu. Sandbox pada istilah keamanan komputer adalah mekanisme pemisahan aplikasi atau program tanpa mengganggu host (isolasi). Bagi pengembang, sandbox Menjamin aplikasinya dapat berjalan tanpa ada gangguan atas perubahan lingkungan host. Sedangkan bagi sysadmin, menjamin host server yang dikelola tidak terganggu dan dapat melakukan update tanpa takut mengganggu aplikasi [8].

Dengan adanya Docker, hal tersebut dapat diminimalisir cukup baik. Docker adalah sebuah proyek yang bersifat open source dibawah lisensi Apache Versi 2.0 yang bisa dipergunakan secara gratis oleh developer dan berfungsi sebagai wadah atau container untuk memasukkan sebuah aplikasi secara lengkap beserta semua hal lainnya yang dibutuhkan sehingga dapat berjalan dimana saja. Dalam hal ini, developer atau sysadmin dapat menjalankan aplikasi

di mana pun misalnya di laptop, data center, virtual machine dan cloud [4].

2.5 Arduino

Arduino adalah pengendali mikro single-board yang bersifat open-source, diturunkan dari Wiring platform, dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. Hardwarenya memiliki prosesor Atmel AVR dan softwarenya memiliki bahasa pemrograman sendiri [5].

Arduino juga merupakan platform hardware terbuka yang ditujukan kepada siapa saja yang ingin membuat purwarupa peralatan elektronik interaktif berdasarkan hardware dan software yang fleksibel dan mudah digunakan. Mikrokontroler diprogram menggunakan bahasa pemrograman arduino yang memiliki kemiripan syntax dengan bahasa pemrograman C. Karena sifatnya yang terbuka maka siapa saja dapat mengunduh skema hardware arduino dan membangunnya [5].

Arduino menggunakan keluarga mikrokontroler ATMega yang dirilis oleh Atmel sebagai basis, namun ada individu/perusahaan yang membuat clone arduino dengan menggunakan mikrokontroler lain dan tetap kompatibel dengan arduino pada level hardware. Untuk fleksibilitas, program dimasukkan melalui bootloader meskipun ada opsi untuk mem-bypass bootloader dan menggunakan downloader untuk memprogram mikrokontroler secara langsung melalui port ISP [5].

2.6 Arduino Yun

Gambar 2.1 Arduino Yun

Arduino Yun adalah sebuah Mikrokontroler ATmega32u4 dan Atheros AR9331 . Prosesor Atheros mendukung distribusi Linux berbasis OpenWRT OpenWRT [6].

Arduino Yun built-in Ethernet dan dukungan WiFi , USB - Sebuah port , slot kartu micro - SD , 20 digital input / output pin (yang 7 Pin dapat digunakan sebagai output PWM dan 12 Pin sebagai input analog),kristal 16 MHz osilator , koneksi micro USB , header ICSP , dan 3 tombol reset [6].

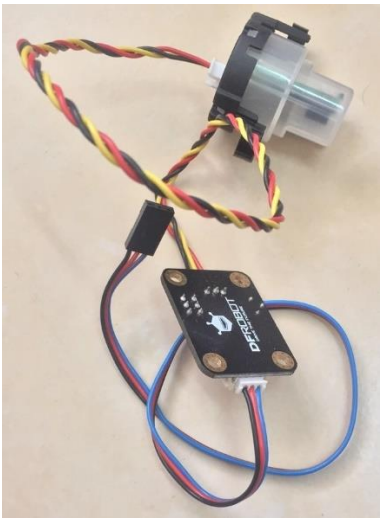
2.7 Modul Sensor PH

Gambar 2.2 Modul Sensor SEN0161

Modul sensor SEN0161 ini merupakan modul sensor PH yang sederhana. Modul ini memungkinkan output data berupa nilai dari PH air yang diteliti. **Kesalahan! Sumber referensi tidak ditemukan.** bentuk fisik dari sensor PH.

Prinsip kerja utama pH meter adalah terletak pada sensor probe berupa elektrode kaca (glass electrode) dengan jalan mengukur jumlah ion H3O+ di dalam larutan. Ujung elektrode kaca adalah lapisan kaca setebal 0,1 mm yang berbentuk bulat (bulb). Bulb ini dipasangkan dengan silinder kaca non-konduktor atau plastik memanjang, yang selanjutnya diisi dengan larutan HCl (0,1 mol/dm3). Di dalam larutan HCl, terendam sebuah kawat elektrode panjang berbahan perak yang pada permukaannya terbentuk senyawa setimbang AgCl. Konstannya jumlah larutan HCl pada sistem ini membuat elektrode Ag/AgCl memiliki nilai potensial stabil [9].

2.8 Modul Sensor Kekeruhan Air



Gambar 2.3 Modul Sensor SEN0189

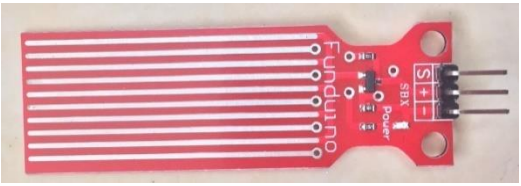
Modul sensor SEN0189 ini merupakan modul sensor kekeruhan air yang sederhana. Modul ini memungkinkan output data berupa data analog maupun data digital. Gambar 2.3 bentuk fisik dari sensor *turbidity*.

Kekeruhan merupakan keadaan mendung atau kekaburan dari cairan yang disebabkan oleh individu partikel (suspended solids) yang umumnya tidak terlihat oleh mata telanjang, mirip dengan asap di udara. Pengukuran kekeruhan adalah tes kunci dari kualitas air. Kekeruhan mengacu pada konsentrasi ketidaklarutan, Keberadaan partikel dalam cairan yang diukur dalam Nephelometric Turbidity Units (NTU). Penting untuk diketahui bahwa kekeruhan adalah ukuran kejernihan sampel, bukan warna [10].

Air dengan penampilan keruh atau tidak tembus pandang akan memiliki kekeruhan tinggi, sementara air yang jernih atau tembus pandang akan memiliki kekeruhan rendah. Nilai kekeruhan yang tinggi disebabkan oleh partikel seperti lumpur, tanah liat,

mikroorganisme, dan material organik. Berdasarkan definisi, kekeruhan bukan merupakan ukuran langsung dari partikel-partikel melainkan suatu ukuran bagaimana partikel menghamburkan cahaya [10].

2.9 Modul Sensor Ketinggian Air



Gambar 2.4 Modul Sensor EK1195

Modul sensor EK1195 ini merupakan modul sensor ketinggian air yang sederhana. Modul sensor ini memungkinkan output data berupa nilai *water level*. Gambar 2.4 bentuk fisik dari sensor *turbidity*.

Kerja dari sensor tersebut adalah membaca resistasi yang dihasilkan oleh air yang mengenai lempengan yang bergiris garis pada sensor tersebut, semakin banyak air yang mengenai permukaan bergaris garis tersebut maka hambatannya semakin kecil dan ketika tidak ada air yang mengenai lempengan sensor tersebut maka hambatanya sangat besar atau bisa dikatakan tidak terhingga [11].

2.10 Modul Sensor Suhu

Gambar 2.5 Modul Sensor DS18B20

Modul sensor DS18B20 ini merupakan modul sensor suhu yang sederhana. Modul sensor ini memungkinkan output data berupa nilai suhu. Gambar 2.5 bentuk fisik dari sensor *tempertaure*.

DS18B20 adalah sensor suhu digital seri terbaru dari Maxim IC (dulu yang buat adalah Dallas Semiconductor, lalu dicaplok oleh Maxim Integrated Products). Sensor ini mampu membaca suhu dengan ketelitian 9 hingga 12-bit, rentang -55°C hingga 125°C dengan ketelitian (+/-0.5°C). Setiap sensor yang diproduksi memiliki kode unik sebesar 64-Bit yang disematkan pada masing-masing chip, sehingga memungkinkan penggunaan sensor dalam jumlah besar hanya melalui satu kabel saja (single wire data bus/1-wire protocol). Ini merupakan komponen yang luar biasa, dan merupakan batu patokan dari banyak proyek-proyek data logging dan kontrol berbasis temperatur di luar sana [12].

2.11 Breadboard Arduino

Gambar 2.6 Breadboard Arduino

Breadboard adalah board yang digunakan untuk membuat rangkaian elektronik sementara dengan tujuan uji coba atau prototipe tanpa harus menyolder. Dengan memanfaatkan breadboard, komponen-komponen elektronik yang dipakai tidak akan rusak dan dapat digunakan kembali untuk membuat rangkaian yang lain. Breadboard umumnya terbuat dari plastik dengan banyak lubang-lubang diatasnya. Lubang-lubang pada breadboard diatur sedemikian rupa membentuk pola sesuai dengan pola jaringan koneksi di dalamnya [13].

Breadboard yang tersedia di pasaran umumnya terbagi atas 3 ukuran: mini breadboard, medium breadboard atau large breadboard. Mini breadboard memiliki 170 titik koneksi (bisa juga lebih). Kemudian medium breaboard memiliki 400 titik koneksi. Dan large breadboard memiliki 830 titik koneksi [13].

Gambar 2.6 diatas adalah medium breadboard dengan 400 titik koneksi. Medium breadboard ini biasa juga disebut half (setengah) breadboard. Karena ukurannya kurang lebih setengah dari ukuran large / full breadboard dengan 830 titik koneksi [13] .

[Halaman ini sengaja dikosongkan]

BAB III
PERANCANGAN PERANGKAT LUNAK

Bab ini membahas khusus mengenai analisis dan perancangan perangkat lunak yang akan dikembangkan. Secara teknis, aktivitas perancangan merupakan salah satu aktivitas yang sangat penting dilakukan dalam rangka pengembangan perangkat lunak. Beberapa hal yang secara umum dibahas dalam bab ini adalah deskripsi umum sistem, arsitektur umum sistem, diagram kasus penggunaan, perancangan basis data, diagram alur dan desain antarmuka perangkat lunak.

3.1 Deskripsi Umum Sistem

Pada Tugas Akhir ini akan dibangun sebuah sistem yang mampu memonitoring kualitas air dari sebuah kolam. Pengambilan data-data kualitas dari air diambil dari modul-modul sensor. Modul-modul sensor yang dipakai adalah sensor pH, turbidity, ketinggian air, temperatur.

Data yang diambil akan dikirim ke server melalui komunikasi socket. Server akan menerima data memilih apakah data-data tersebut berasal dari cluster yang ada dalam database. Jika cluster belum ada, server akan membuat container docker baru dan memasukkan data tersebut ke dalam container.

Pengguna dapat menggunakan aplikasi monitoring yang dibuat berbasis website. Di website tersebut terdapat tampilan grafik dan tabel dari cluster air kolam untuk memonitoring kualitas air tersebut.

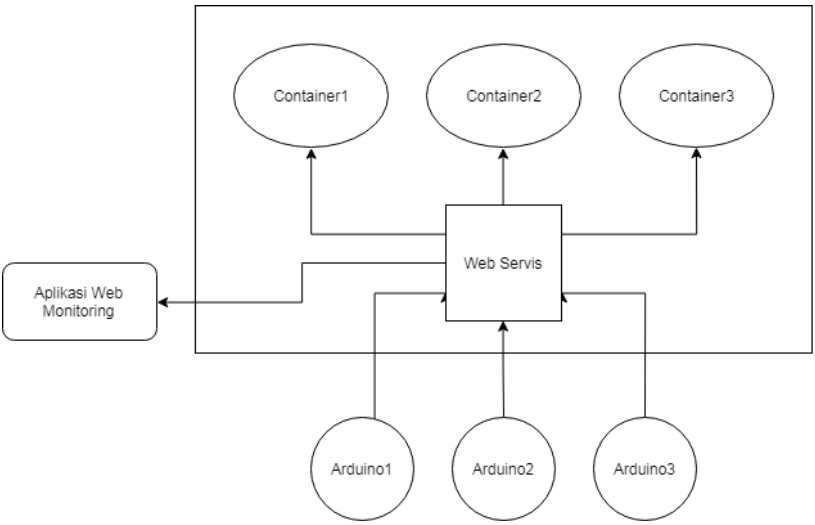
3.2 Arsitektur Jaringan Sistem

Di bagian bawah dari arsitektur umum sistem terdapat arduino sebagai mikrokontroller yang terpasang modul-modul sensor yang digunakan untuk mendeteksi kualitas air.

Setelah itu data dikirim melalui komunikasi socket. Terdapat web servis untuk mengatur data-data tersebut pada server. Container baru akan terbuat, jika arduino/cluster adalah

baru dalam sistem monitoring ini. Jika cluster sudah tersimpan dalam database, maka data akan langsung dimasukkan ke dalam container cluster tersebut.

Terdapat aplikasi web bagi pengguna untuk memonitor kualitas air tersebut.



Gambar 3.1 Arsitektur Jaringan

3.3 Perancangan Monitoring Kualitas Air (Server)

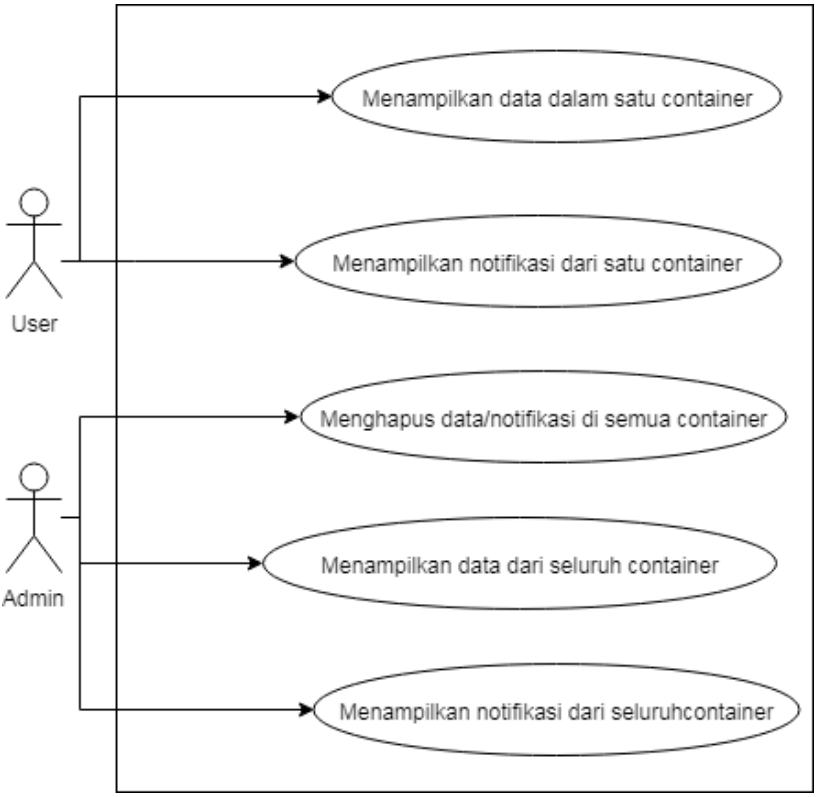
Monitoring Kualitas Air (MONAIR) yang akan dibangun merupakan sebuah website untuk memonitor atau memantau keadaan air melalui data – data yang dikirimkan oleh node sensor. Node – node sensor yang telah didaftarkan akan ditampilkan pada website.

Kegunaan lain yang dimiliki website ada beberapa hal. Beberapa hal tersebut meliputi, menampilkan data dalam grafik dan tabel, menampilkan notifikasi.

3.3.1 Perancangan Docker

Pada Sistem Monitoring Kualitas Air (MONAIR) akan dibuat container-container. Masing-masing container akan dipasang image mysql versi 5.7. Container-container tersebut nantinya berperan sebagai database dari masing-masing cluster/kelompok air yang diteliti. Di dalam database tersebut akan dibuat 2 tabel untuk menyimpan data-data dari node sensor di lapangan.

3.3.2 Perancangan Diagram Kasus Penggunaan



Gambar 3.2 Kasus Penggunaan Sistem

Pada bagian ini akan dijelaskan rincian kasus penggunaan (*use case*) yang akan dibangun dalam website. Kasus penggunaan yang akan dibuat dalam website terdiri dari beberapa hal yaitu menampilkan data dalam satu container, menampilkan notifikasi dalam satu container, mengapus data/notifikasi di seluruh container, menampilkan data dari seluruh container, menampilkan notifikasi dari seluruh container. Gambar 3.2 merupakan gambar diagram kasus penggunaan (*use case*). Berdasarkan diagram kasus penggunaan yang telah dibuat, berikut ini rincian dari setiap kasus penggunaan yang ada:

3.3.2.1 Menampikan Data dalam Satu Container

Kasus penggunaan ini merupakan halaman utama (*dashboard*) dan halaman tabel dari Pengguna. Pengguna dapat menampilkan seluruh data pH, kekeruhan air, *water kevel*, temperatur dalam grafik maupun tabel dari database container yang ia miliki. Pengguna juga dapat menampilkan data gambar-gambar yaang diperoleh dari Arduino di lapangan secara *real-time* dari database container yang ia miliki.

3.3.2.2 Menampilkan Notifikasi dalam Satu Container

Kasus penggunaan ini merupakan halaman notifikasi dari Pengguna. Pengguna dapat menampilkan seluruh data pH, kekeruhan air, *water kevel*, temperatur yang datanya tidak berada pada batas *treshold* yang ditentukan.

3.3.2.3 Menghapus Data/Notifikasi di Seluruh Container

Pada kasus penggunaan yang satu ini, admin mempuyai wewenang untuk menghapus record dari container docker manapun. Kasus penggunaan ini terdapat pada halaman tabel dari Admin.

3.3.2.4 Menampikan Data dari Seluruh Container

Kasus penggunaan ini merupakan halaman utama (*dashboard*) dan halaman tabel dari Admin. Admin dapat menampilkan seluruh data pH, kekeruhan air, *water kevel*,

temperatur dalam grafik maupun tabel dari container docker manapun. Pengguna juga dapat menampilkan data gambar-gambar yaang diperoleh dari Arduino di lapangan secara *real-time* dari container docker manapun.

3.3.2.5 Menampilkan Notifikasi dari Seluruh Container

Kasus penggunaan ini merupakan halaman notifikasi dari Admin. Admin dapat menampilkan seluruh data pH, kekeruhan air, *water kevel*, temperatur yang datanya tidak berada pada batas *treshold* yang ditentukan dari container docker manapun.

3.3.3 Perancangan Basis Data (Monitoring Kualitas Air)

Pada perancangan basis data merupakan tahapan untuk merancang basis data yang akan digunakan pada Tugas Akhir ini. Basis data ini berfungsi untuk menyimpan user, data-data sensor, dan notifikasi

3.3.3.1 Basis Data pada Komputer Host

Pada komputer *host*, terdapa 2 basis data, *users* dan *monair*. Di dalam *database users*, terdapat 1 tabel yaitu tabel *user*. Lalu di dalam *database monair* terdapat 2 tabel yaitu tabel *data* dan tabel *notification*. Tabel - tabel tersebut digunakan untuk menyimpan data *user-user* yang ada pada sistem monitoring ini dan data - data kualitas air yang diteliti di semua *cluster*.

3.3.3.1.1 Tabel *user*

Tabel 3.1 Perancangan Tabel *user*

No	Nama Atribut	Tipe Data	Keterangan
1	*Id*	*int*	Sebagai *primary key* untuk tabel *users*
2	*macaddress*	*varchar*	Sebagai *username* untuk *log in* ke MONAIR
3	*password*	*varchar*	Sebagai *password* untuk *log in* ke MONAIR

3.3.3.1.2 Tabel *data*

Tabel 3.2 Perancangan Tabel *data*

No	Nama Atribut	Tipe Data	Keterangan
1	id_data	int	Sebagai *primary key* untuk tabel *data*
2	ph	varchar	Nilai pH dari air
3	temperature	varchar	Nilai suhu dari air
4	turbidity	varchar	Nilai kekeruhan dari air
5	waterlevel	varchar	Nilai ketinggian dari air
6	photo	varchar	Gambar dari objek yang diteliti
7	docker	varchar	Menyimpan nama container

3.3.3.1.3 Tabel *notification*

Tabel 3.3 Perancangan Tabel *notification*

No	Nama Atribut	Tipe Data	Keterangan
1	*id_notif*	*int*	Sebagai *primary key* untuk tabel *notifikasi*
2	*id_data*	*int*	Sebagai *foreign key* dari tabel *data* untuk tabel *notifikasi*
3	*keterangan*	*varchar*	Sebagai keterangan dari data *notifikasi*
4	*docker*	*varchar*	Menyimpan nama container

3.3.3.2 Basis Data pada Container

Container memiliki 1 database dan 2 tabel, yaitu tabel *data* dan tabel *notifikasi*. Tabel data digunakan untuk menyimpan data-data dari node sensor dan tabel notifikasi digunakan untuk menyimpan data-data di luar batas rata-rata yang ditentukan dari node sensor.

3.3.3.2.1 Tabel *data*

Tabel 3.4 Perancangan Tabel *data*

No	Nama Atribut	Tipe Data	Keterangan
1	id_data	int	Sebagai *primary key* untuk tabel *data*
2	ph	varchar	Nilai pH dari air
3	temperature	varchar	Nilai suhu dari air
4	turbidity	varchar	Nilai kekeruhan dari air
5	waterlevel	varchar	Nilai ketinggian dari air
6	photo	varchar	Gambar dari objek yang diteliti

3.3.3.2.2 Tabel *notification*

Tabel 3.5 Perancangan Tabel *notification*

No	Nama Atribut	Tipe Data	Keterangan
1	*id_notif*	*int*	Sebagai *primary key* untuk tabel *notifikasi*
2	*id_data*	*int*	Sebagai *foreign key* dari tabel *data* untuk tabel *notifikasi*
3	*keterangan*	*varchar*	Sebagai keterangan dari data *notifikasi*

3.3.4 Perancangan Antarmuka Pengguna (Monitoring Kualitas Air)

Perancangan antarmuka pengguna merupakan satu tahap yang dilakukan untuk persiapan dalam pembuatan antarmuka website agar lebih mudah dalam pelaksanaan proses implementasi. Antarmuka website akan dibuat dengan memanfaatkan Bootstrap.

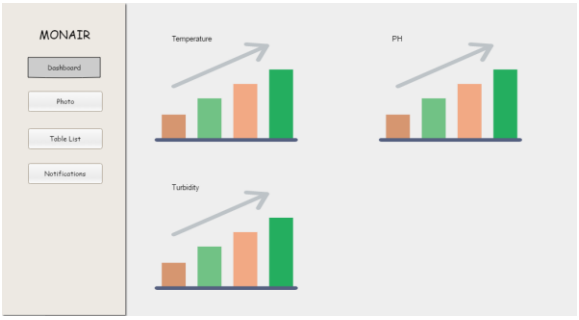
Website yang akan dibangun memiliki beberapa halaman penting. Antarmuka halaman yang akan dibuat meliputi halaman login, halaman utama (dashboard), halaman *photo*, halaman *tabel*, dan halaman *notifikasi*.

3.3.4.1 Halaman *log in*

Gambar 3.3 Rancangan Antarmuka Halaman Login

Gambar 3.3 merupakan rancangan antarmuka halaman login ketika pengguna membuka *web* dari aplikasi ini. Untuk menggunakan sistem, pengguna harus terdaftar terlebih dahulu, kemudian melakukan login untuk bisa masuk ke halaman *dashboard* yang bisa dilihat rancangannya pada Gambar 3.3

3.3.4.2 Halaman *dashboard*



Gambar 3.4 Rancangan Antarmuka Halaman *Dashboard*

Gambar 3.4 merupakan rancangan antarmuka halaman *Dashboard*. Halaman ini merupakan halaman utama dari sistem ketika pengguan sudah melakukan *login*. Pada halaman ini terdapat menu *Dashboard, Photo, Table List, Notifications*. Di

dalam halaman *Dashboard* terdapat grafik yang menunjukkan data-data dari kualitas air yang diteliti, yaitu data *turbidity*, *ph*, dan suhu.

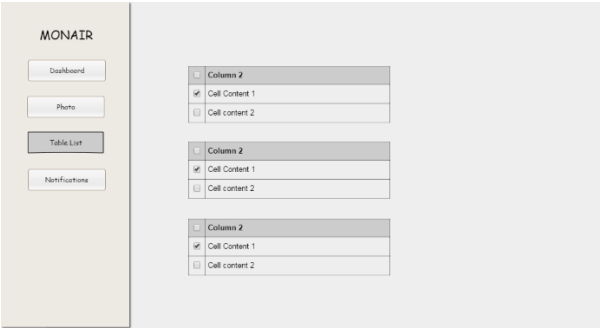
3.3.4.3 Halaman *photo*

Gambar 3.5 Rancangan Antarmuka Halaman *Photo*

Gambar 3.5 merupakan rancangan antarmuka halaman *Photo*. Halaman ini merupakan halaman *Photo* yang akan muncul ketika menu *Photo ditekan*. Pada halaman ini terdapat menu *Dashboard, Photo, Table List, Notifications*. Di dalam halaman *Photo,* terdapat gambar dari kolam terkait yang sedang diteliti kualitas airnya.

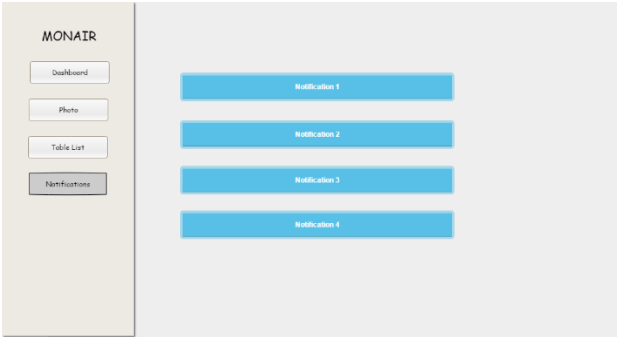
3.3.4.4 Halaman *tabel*

Gambar 3.6 merupakan rancangan antarmuka halaman *Table List*. Halaman ini merupakan halaman *Table List* yang akan muncul ketika menu *Table List ditekan*. Pada halaman ini terdapat menu *Dashboard, Photo, Table List, Notifications*. Di dalam halaman *Photo,* terdapat tabel – tabel yang berisi semua data – data dari kualitas air yang diteliti di kolam.



Gambar 3.6 Rancangan Antarmuka Halaman *Table List*

3.3.4.5 Halaman *notifikasi*



Gambar 3.7 Rancangan Antarmuka Halaman *Notifications*

Gambar 3.7 merupakan rancangan antarmuka halaman *Notifications*. Halaman ini merupakan halaman *Notifications* yang akan muncul ketika menu *Notifications*. Pada halaman ini terdapat menu *Dashboard, Photo, Table List, Notifications*. Di dalam halaman *Notifications*, terdapat data yang berisi nilai kualitas air yang bermasalah.

3.4 Perancangan Web Service

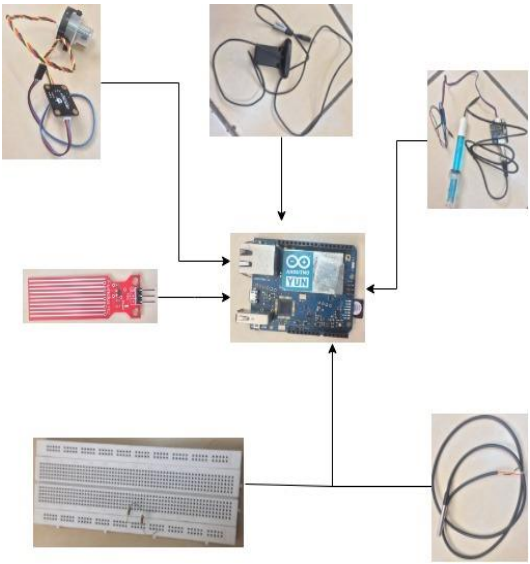
Web service pada Tugas Akhir ini digunakan untuk menerima data dari *node* sensor fisik yang terdaftar. *Web service*

yang dirancang merupakan *web service* sederhanda dengan memanfaatkan komunikasi *socket* untuk mengirimkan data dari node sensor ke *cloud* sever.

3.5 Perancangan Arduino

Arduino yang dipakai adalah Arduino tipe Yun. Arduino Yun akan dipasangi beberapa modul sensor. Modul sensor yang dipasang ke Arduino Yun adalah:

- Modul sensor *pH*
- Modul sensor *turbidity*
- Modul sensor *water level*
- Modul sensor *temperature*
- Modul *webcam*



Gambar 3.8 Rancangan Sensor Arduino

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Pada bab sebelumnya telah dijelaskan mengenai rancangan dari keseluruhan sistem yang akan dibangun. Setelah dilakukan proses perancangan, proses selanjutnya yang harus ditempuh adalah proses implementasi. Oleh karena itu, pada bab ini akan berisi penjelasan mengenai implementasi dari perancangan yang sudah dilakukan pada bab sebelumnya. Implementasi berupa secara umum meliputi *pseudocode*, antarmuka aplikasi, rangkaian utama dan sebagainya

4.1 Lingkungan Implementasi

Lingkungan implementasi merupakan suatu lingkungan dimana sistem akan dibangun. Pada bagian lingkungan implementasi ini, untuk mempermudahnya akan dibagi menjadi dua. Pembahasan yang pertama yaitu Lingkungan Implementasi Perangkat Keras dan Lingkungan Implementasi Perangkat Lunak.

4.1.1. Perangkat Keras

Pada bagian ini akan dibahas mengenai perangkat keras apa saja yang dibutuhkan untuk membangun sistem. Lingkungan implementasi perangkat keras dari sistem yang akan dibangun secara lebih lengkap dijelaskan pada Tabel 4-1 di bawah ini.

Tabel 4.1 Lingkungan Implementasi Perangkat Keras

Perangkat	Detail Perangkat
Perangkat Komputer	**Manufaktur:** • DigitalOcean. **Processor:** • Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz **Memori:** • 2048 MB **Sistem Operasi:**

Perangkat	Detail Perangkat
	• Ubuntu 16.04.4 LTS
Perangkat Mikrokontroler	**Mikrokontroler:** • Atmega32u4 **Model:** • Arduino Yun **Tegangan:** • 5 V **Memori *Flash*:** • 32 KB **SRAM:** • 2 KB **Sensor:** • Modul DS18B20 (*temperature*) • Modul SEN0189 (*turbidity*) • Modul SEN0161 (*pH*) • Modul *water level* • Modul *webcam*

4.1.2. Perangkat Lunak

Setelah mengetahui perangkat keras apa saja yang dibutuhkan, sekarang saatnya untuk menjabarkan perangkat lunak yang dibutuhkan. Pada bagian ini akan dibahas mengenai perangkat lunak apa saja yang dibutuhkan untuk membangun sistem. Lingkungan implementasi perangkat lunak dari sistem yang akan dibangun secara lebih lengkap dijelaskan pada **Kesalahan! Sumber referensi tidak ditemukan.** di bawah ini.

Tabel 4.2 Lingkungan Implementasi Perangkat Lunak

Perangkat	Detail Perangkat
Perangkat Komputer	**Sistem Operasi:** • Ubuntu 16.04.4 LTS ***Software* Arduino:**

Perangkat	Detail Perangkat
	• Arduino IDE 1.6.12 ***Software* Web Service:** • Python 2.7.10 **Webserve:** • XAMPP 1.8.2 • Apache 2.4.17 • PHP 5.6.14 **Basis Data:** • MySQL 5.7 **Text Editor:** • Sublime Text 3 Build 3103

4.2 Implementasi Monitoring Kualitas Air

Pada sub-bab ini akan dibahas mengenai bagaimana implementasi yang dilakukan untuk membangun *Monitoring Kualitas Aur* yang mengacu pada bab **Kesalahan! Sumber referensi tidak ditemukan.**. Sistem berupa website ini dibangun dengan menggunakan *framework* Bootstrap. Basis data yang digunakan adalah MySQL.

4.2.1 Implementasi Docker

Container akan dibuat di secara otomatis, jika *mac address* dari Arduino tidak erdapat dalam database host. Berikut ini ada *bash command* cara membuat container docker dengan base image mysql 5.7.

```
1.  docker run -d --name container1 -e
    MYSQL_ROOT_PASSWORD=a mysql:5.7
```

Kode Sumber 4.1 Bash Command Pembuatan Container

Container akan terbuat dengan nama *container1* dan terdapat mysql dengan password ‘a’. Setiap container memiliki *IP address* masing-masing. *IP address* dibutuhkan untuk terhubung masing-masing container, agar data bisa masuk ke dalam database

masing-masing container. Berikut ini adalah *bash command* cara mendapatkan *IP address* dari salah satu container.

```
1.  docker inspect container1 | grep 'IPAddress' | tail -1 | awk
    '{print $2}' | tr -d ',' | tr -d '\"'
```

Kode Sumber 4.2 Bash Command IP Container

Dengan *docker inspect, IP address* dari salah satu container bisa didapat. Selanjutnya IP tersebut akan dipakai untuk menghubungkan ke dalam database docker. Kedua bash command di atas terdapat dalam *web service code* yang nantinya bisa otomatis dijalankan dengan adanya deteksi dari Arduino

4.2.2 Implementasi Kasus Penggunaan

Berdasarkan kasus kegunaan yang dirancang pada bab **Kesalahan! Sumber referensi tidak ditemukan.**, ada lima kasus kegunaan yang akan diimplementasikan. Kasus kegunaan tersebut meliputi kasus kegunaan menampilkan data dalam satu container, menampilkan notifikasi dalam satu container, mengapus data/notifikasi di seluruh container, menampilkan data dari seluruh container, menampilkan notifikasi dari seluruh container. Berikut ini penjelasan secara lebih menyeluruh.

4.2.2.1 Menampilkan Data dalam Satu Container

Kasus kegunaan ini akan menyajikan halaman utama Pengguna dan merupakan implementasi dari bab **Kesalahan! Sumber referensi tidak ditemukan.**.

Pada kasus kegunaan ini yang akan dilakukan adalah mengolah data waktu, pH, *turbidity*, suhu, *water level*, dan *camera* dari database docker. Lalu menampilkan data – data tersebut pada halaman *dashboard* (dalam bentuk grafik), halaman *photo*, dan halaman *tabel*.

```
1  FUNCTION getTime()
2    load model
3    initialize data
```

```
4   set data → group_id
5   set db → query ← 'SELECT time FROM data;'
6   set query ← db → query
7   return query → result
8   ENDFUNCTION
```

Kode Sumber 4.3 *Pseudocode* Fungsi Tampilkan Waktu

Pada Kode Sumber 4.3 diperlihatkan cara menampilkan waktu dari tabel *data* dalam database container. Variabel *$con* yang dipakai adalah untuk menghubungkan ke database container dengan IP dari container tersebut.

```
1   FUNCTION getPH()
2   load model
3   initialize data
4   set data → group_id
5   set db → query ← 'SELECT ph FROM data;'
6   set query ← db → query
7   return query → result
8   ENDFUNCTION
```

Kode Sumber 4.4 *Pseudocode* Fungsi Tampilkan Nilai pH

Pada Kode Sumber 4.4 diperlihatkan cara menampilkan ph dari tabel *data* dalam database container. Variabel *$con* yang dipakai adalah untuk menghubungkan ke database container dengan IP dari container tersebut. Data pH diubah ke dalam tipe *float* dari tipe *varchar* agar dapat ditampilkan dalam grafik.

```
1   FUNCTION getTurbid()
2   load model
3   initialize data
4   set data → group_id
5   set db → query ← 'SELECT turbidity FROM data;'
6   set query ← db → query
7   return query → result
8   ENDFUNCTION
```

Kode Sumber 4.5 *Pseudocode* Fungsi Tampilkan Nilai *Turbidity*

Pada Kode Sumber 4.5 diperlihatkan cara menampilkan *turbidity* dari tabel *data* dalam database container. Variabel *$con* yang dipakai adalah untuk menghubungkan ke database container dengan IP dari container tersebut. Data *turbidity* diubah ke dalam tipe *float* dari tipe *varchar* agar dapat ditampilkan dalam grafik.

```
1  FUNCTION getTemp()
2    load model
3    initialize data
4    set data → group_id
5    set db → query ← 'SELECT temperature FROM data;'
6    set query ← db → query
7    return query → result
8  ENDFUNCTION
```

Kode Sumber 4.6 *Pseudocode* Fungsi Tampilkan Nilai Suhu

Pada Kode Sumber 4.6 diperlihatkan cara menampilkan suhu dari tabel *data* dalam database container. Variabel *$con* yang dipakai adalah untuk menghubungkan ke database container dengan IP dari container tersebut. Data *temperature* diubah ke dalam tipe *float* dari tipe *varchar* agar dapat ditampilkan dalam grafik.

```
1  FUNCTION getWL()
2    load model
3    initialize data
4    set data → group_id
5    set db → query ← 'SELECT waterlevel FROM data;'
6    set query ← db → query
7    return query → result
8  ENDFUNCTION
```

Kode Sumber 4.7 *Pseudocode* Fungsi Tampilkan Nilai *Water Level*

Pada Kode Sumber 4.7 diperlihatkan cara menampilkan ph dari tabel *data* dalam database container. Variabel *$con* yang dipakai

adalah untuk menghubungkan ke database container dengan IP dari container tersebut. Data pH diubah ke dalam *float* dari tipe *varchar* agar dapat ditampilkan dalam grafik.

```
1  FUNCTION getPhoto()
2    load model
3    initialize data
4    set data → group_id
5    set db → query ← 'SELECT photo FROM data;'
6    set query ← db → query
7    return query → result
8  ENDFUNCTION
```

Kode Sumber 4.8 *Pseudocode* Fungsi Tampilkan *Photo*

Pada Kode Sumber 4.8 diperlihatkan cara menampilkan *photo* dari tabel *data* dalam database container. Variabel *$con* yang dipakai adalah untuk menghubungkan ke database container dengan IP dari container tersebut. Data *photo* ditampilkan ke dalam menu *photo.*

4.2.2.2 Menampilkan Notifikasi dalam Satu Container

Kasus kegunaan ini akan menyajikan halaman notifikasi dan merupakan implementasi dari bab **Kesalahan! Sumber referensi tidak ditemukan.**.

Pada kasus kegunaan ini yang akan dilakukan adalah mengolah data notifikasi dari database docker. Lalu menampilkan data – data tersebut pada halaman notifikasi.

```
1  FUNCTION getNotif()
2    load model
3    initialize data
4    set data → group_id
5    set db → query ← 'SELECT data.time, notification,
       keterangan FROM data, notification WHERE
       notification.id_data = data.id_data;'
6    set query ← db → query
```

```
7   return query → result
8   ENDFUNCTION
```

Kode Sumber 4.9 *Pseudocode* Fungsi Tampilkan Notifikasi

Pada Kode Sumber 4.9 diperlihatkan cara menampilkan data notifikasi dari tabel *notification* dalam database container. Variabel *$con* yang dipakai adalah untuk menghubungkan ke database container dengan IP dari container tersebut.

4.2.2.3 Menampilkan Data dari Seluruh Container

Kasus kegunaan ini akan menyajikan halaman utama Pengguna dan merupakan implementasi dari bab **Kesalahan! Sumber referensi tidak ditemukan.**.

Pada kasus kegunaan ini yang akan dilakukan adalah mengolah data waktu, pH, *turbidity*, suhu, *water level*, dan *camera* dari database docker. Lalu menampilkan data – data tersebut pada halaman *dashboard* (dalam bentuk grafik), halaman *photo*, dan halaman *tabel*.

4.2.2.4 Mengapus Data/Notifikasi di Seluruh Container

Kasus kegunaan ini akan menyajikan halaman notifikasi dan merupakan implementasi dari bab **Kesalahan! Sumber referensi tidak ditemukan.**.

Pada kasus kegunaan ini yang akan dilakukan adalah mengolah data notifikasi dari database docker. Lalu menampilkan data – data tersebut pada halaman notifikasi.

4.2.2.5 Menampilkan Notifikasi dari Seluruh Ccontainer

Kasus kegunaan ini akan menyajikan halaman notifikasi dan merupakan implementasi dari bab **Kesalahan! Sumber referensi tidak ditemukan.**.

Pada kasus kegunaan ini yang akan dilakukan adalah mengolah data notifikasi dari database docker. Lalu menampilkan data – data tersebut pada halaman notifikasi.

4.2.3 Implementasi Web Service

Web Service dibuat dalam bahasa pemrograman *python.* Di dalam *web service* Monitoring Kualitas Air (server.py) terdapat beberapa tahap. Web Service menggunakan library MySQLdb yang dijalankan pada *bash command*. Berikut adalah tahap-tahap dalam web service *server.py*.

```
1   FUNCTION socket()
2     mac_address ← recv(1024)
3     send ACK
4     print mac_address
5
6     time ← c.recv(1024)
7     send ACK
8     print time
9
10    waterlevel ← recv(1024)
11    send ACK
12    print waterlevel
13
14    ph ← recv(1024)
15    send ACK
16    print ph
17
18    turbidity ← recv(1024)
19    send ACK
20    print turbidity
21
22    temperature ← recv(1024)
23    send ACK
24    print temperature
25  ENDFUNCTION
```

Kode Sumber 4.10 Penerimaan Data dengan Komunikasi *Socket*

Pertama-tama menerima data dari Arduino dengan komunikasi *socket*. Seperti pada Kode Sumber 4.10. Data yang dikirim kan berupa tipe *varchar* atau *string* yaitu *mac_address, time, waterlevel, ph, turbidity, temperature*.

```
1   FUNCTION socketphoto()
2     image_name ← recv(1024)
3
4     send ACK
5     image_size ← recv(1024)
6
7     send ACK
8
9
10    while image_size is greater than 0:
11      if image_size is greater than 1024:
12        image_data ← recv(1024)
13      else
14        image_data ← recv(image_size)
15      image_size -= len(image_data)
16      image_string += image_data
17
18    send ACK
19    image_decode ← base64.decodestring(image_string)
20    write image_result(image_decode)
21    write dummy (image_string)
22  ENDFUNCTION
```

Kode Sumber 4.11 Penerimaan Data Image *Base64*

Kode Sumber 4.11 menunjukan bagaimana cara menerima dan men-*decode* data *photo* dari format base64. Perulangan digunakan untuk mengirimkan *image_data*. Jika *image_size* lebih dari 1024, data akan dipotong menjadi beberapa bagian untuk dikirim (sebesar 1024). Setelah itu memasuki tahap pengecekkan mac address dari Arduino dan menentukan langkah selajutnya.

```
1  FUNCTION getMac()
2    load model
3    initialize data
4    set data → group_id
5    set db → query ← 'SELECT macaddress FROM user
       WHERE macaddress = id;'
6    set query ← db → query
7    return query → result
8  ENDFUNCTION
```

Kode Sumber 4.12 *Pseudocode* Fungsi Tampilkan *macaddress*

Kode Sumber 4.12 menunjukan bagaimana pengecekan mac address. Apakah *mac address* sudah ada dalam *database* komputer *host* atau belum. Jika belum, maka langkah selanjutnya adalah pembuatan container lewat web service.

```
1  command = 'docker run -d --name {} -e
   MYSQL_ROOT_PASSWORD=a
   mysql:5.7'.format(mac_address)
2  process = subprocess.Popen(command,
   stdout=subprocess.PIPE, shell=True)
3  output, error = process.communicate()
4  process.wait()
5  sleep(16)
6  command = "docker inspect {} | grep 'IPAddress' | tail -1 | awk
   '{{print $2}}' | tr -d ',' | tr -d '\"'".format(mac_address)
7  ipaddress = subprocess.check_output(command,
   shell=True).strip()
```

Kode Sumber 4.13 *Bash Command* Pembuatan dan Pencarian IP Container

Bash command untuk membuat container sama seperti Kode Sumber 4.13, hanya saja pada Kode Sumber x.x nantinya dijalankan secara otomatis di pyton. Fungsi sleep digunakan, karena diperlukan waktu dari komputer host untuk membuat container docker.

```
1  FUNCTION insert_macaddress()
2    load model
3    select database
4    insert newData ← database
5  ENDFUNCTION
```

Kode Sumber 4.14 *Pseudocode* Fungsi *inset_macaddress*

Bash command pada baris ke-6 di Kode Sumber x.x untuk mengetahui IP dari container yang baru dibuat, agar IP tersebut bisa dimasukkan ke dalam database komputer host seperti pada Kode Sumber 4.14. Setelah itu akan dbuat database dengan nama *"monair"*. Lalu connect ke database docker menggunakan IP container tersebut.

```
1  cursor_docker.execute("""CREATE TABLE data(id_data int
   NOT NULL AUTO_INCREMENT,time varchar(255),ph
   varchar(255),
2  temperature varchar(255),waterlevel varchar(255),
3  turbidity varchar(255),photo varchar (255),PRIMARY KEY
   (id_data))""")
4
5  cursor_docker.execute("""CREATE TABLE
   notification(id_notif int NOT NULL
   AUTO_INCREMENT,id_data int NOT NULL,
6  keterangan varchar(255),PRIMARY KEY (id_notif),FOREIGN
   KEY (id_data) REFERENCES data(id_data))""")
```

Kode Sumber 4.15 *Bash Command* Pembuatan *table*

Kode Sumber 4.15 menunjukkan bagaimana pembuatan tabel *data* dan tabel *notifkasi*. Langkah terakhir ada pemasukkan data ke dalam database container docker di tabel *data* maupun tabel *notifikasi* seperti di atas.

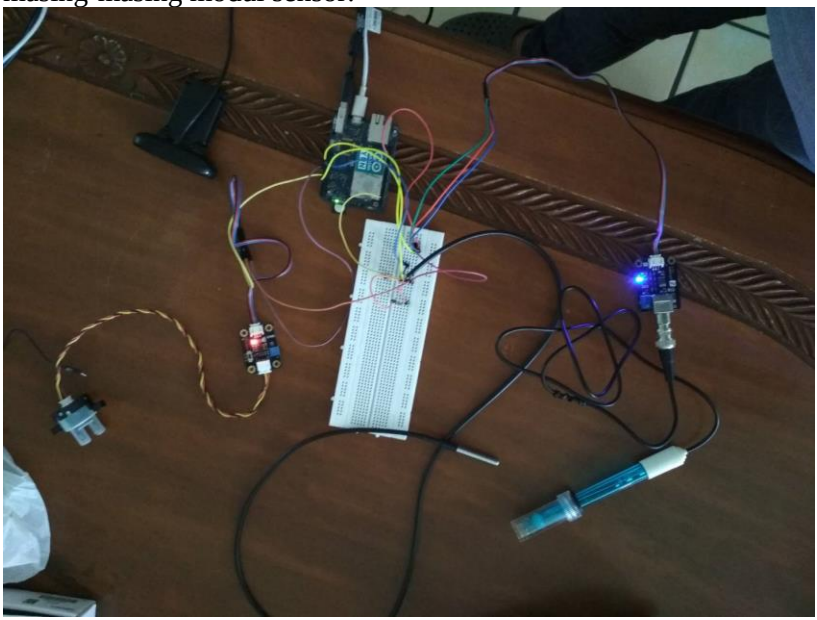
4.3 Implementasi Arduino

Arduino YUN yang diapakai akan dipasangi beberapa modul Sensor. Sensor yang dipakai meliputi sensor *ph*, *temperature*, *water level*, *turbidity*, dan modul webcam. Implementasi Arduino terbagi

mejadi 2 bagian, yaitu pengambilan data dan pengirim data ke server.

4.3.1 Implementasi Pengambilan Data Sensor

Pada sub bab ini, akan dijelaskan bagaimana data diambil dari masing-masing modul sensor.



Gambar 4.1 Rangkain Modul Sensor dan Arduino

```
1   FUNCTION getph()
2     set analogInPin ← A0
3     set sensorValue ← 0
4     set avgValue
5     initialize counter
6     initialize buf[10] and temp
8
9     while i is less than ten
10      buf[i] ← analogRead(analogInPin)
```

```
11    delay 10
12
13   while i is less than nine
14
15    while j is less than ten and j = i
16
17         If buff[i] greater than buff[j]
18
19           temp ← buf[i]
20           buf[i] ← buf[j]
21           buf[j] ← temp
22
24
25      avgValue ← 0
26      while i equals to two and less than eight
27        set avgValue += buf[i]
28        set pHVol ← avgValue*5.0/1024/6;
29        set phValue ← -5.70 * pHVol + 21.34;
30      return phValue;
31  ENDFUNCTION
```

Kode Sumber 4.16 *Pseudocode* Fungsi *getph*

Kode Sumber 4.16. menjelaskan bagaimana mendapatkan nilai ph dari sensor. Perulangan pertama diguakan untuk membaca nilai dari analog. Perualangan kedua dilakukan perhitungan agar mendapatkan nilai final

```
1  FUNCTION getTurbidity()
2   set val ← analogRead(A1)
3   set tegangan ← val*(5.0/1024)
4   set kekeruhan ← 100.00 - (tegangan/4.16)*100.00
5
6   return kekeruhan;
7  ENDFUNCTION
```

Kode Sumber 4.17 *Pseudocode* Fungsi *getTurbidity*

```
1  FUNCTION getWaterLevel()
2    set sensorValue ← 0
3    set sensorValue ← analogRead(A0);
4    return sensorValue;
5  ENDFUNCTION
```

Kode Sumber 4.18 *Pseudocode* Fungsi *getWaterLevel*

```
1  FUNCTION capture(filename)
2    process process
3    process.runShellCommand("fswebcam  -i 0 -d
     v4l2:/dev/video0  --jpeg 100  --save "+filename+".jpg -S 20 -r
     640x480");
4    while process is running()
5  ENDFUNCTION
```

Kode Sumber 4.19 *Pseudocode* Fungsi *capture*

```
1  FUNCTION getMAC()
2    process process;
3    process.runShellCommand("/usr/bin/pretty-wifi-info.lua | grep
     'MAC' | awk '{print $3}' ");
4    initialize result
5    while process.available is greater than zero
6      set c ← process.read();
7      set result += c
8
9    return result
10 ENDFUNCTION
```

Kode Sumber 4.20 *Pseudocode* Fungsi *getMAC*

```
1  FUNCTION getTime()
2    Process process;
3    process.runShellCommand("echo GMT-7 > /etc/TZ && date |
     awk '{print $1,$2,$3,$4,$6}'");
4    initialize result;
5    while process.available is greater than zero
6      set c ← process.read();
```

```
7     set result += c;
8
9    return result;
10   ENDFUNCTION
```

Kode Sumber 4.21 *Pseudocode* Fungsi *getTime*

4.3.2 Implementasi Pengiriman Data Sensor ke Server

```
1   Process process;
2    process.runShellCommand("python /root/send_server.py --
    mac="+String(mac)+" --waktu="+String(waktu)+" --
    waterlevel="+String(waterlevel)+" --
    phvalue="+String(phvalue)+" --turbidity="+String(turbidity)+"
    --image="+waktu+".jpg");
3    while(process.running());
```

Kode Sumber 4.22 Proses Pengiriman Data dengan Komunikasi Socket

```
1   def send_image():
2   print '[INFO] Sending {} to server'.format(args['image'])
3   set image ← open('{}'.format(args['image']), 'rb')
4   set dummy ← open('log', 'wb')
5   set image_read ← image.read()
6   set image_encode ← base64.encodestring(image_read)
7   set image_size ← len(image_encode)
8   write dummy (image_encode)
9
10  FUNCTION send()
11  send (args['image'])
12  recv(1024)
14
15  send image_size
16  print image_size
17  recv(1024)
18
20  send image_encode
21  recv(1024)
```

```
22 ENDFUNCTION
```

Kode Sumber 4.23 Proses Pengiriman Gambar dengan Komunikasi Socket

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas hasil dan pembahasan pada aplikasi yang dikembangkan. Pada bab ini akan dijelaskan tentang data yang digunakan, hasil yang didapatkan dari penggunaan perangkat lunak dan uji coba yang dilakukan pada perangkat lunak yang telah dikerjakan untuk menguji apakah fungsionalitas aplikasi telah diimplementasikan dengan benar dan berjalan sebagaimana mestinya.

5.1 Lingkungan Uji Coba

Lingkungan uji coba mencakup perangkat dan peralatan apa saja yang digunakan dalam uji coba. Uji coba dilakukan di beberapa lokasi yang telah ditentukan sesuai dengan kebutuhan pada setiap skenario. Lingkungan uji coba seperti di bawah ini:

- Laptop ASUS ROG GL552JX
  - Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz
  - RAM 2 GB
  - Ubuntu 16.04.4 LTS

- Tiga node sensor dengan setiap node memiliki komponen sebagai berikut :
  - Mikrokontroler Arduino Yun
  - Modul sensor temperature
  - Modul sensor pH
  - Modul sensor turbidity
  - Modul sensor waterlevel
  - Modul webcam
  - Papan kerja *breadboard*
  - Resistor 4.8K Ω
  - Beberapa kabel *jumper*
  - Powerbank 9V 220mAh
  - Panel Surya

- Tempat Uji Coba sebagai berikut:
  - Kolam di belakang Research Center ITS

Gambar 5.1 Lingkungan Uji Coba Pertama

Gambar 5.2 Uji Coba di Lokasi Pertama

Gambar 5.3 Rangkaian Arduino

- Kolam Blok J

Gambar 5.4 Lingkungan Uji Coba Kedua

Gambar 5.5 Uji Coba di Lokasi Kedua

Gambar 5.6 Rangkaian Arduino

5.2 Skenario Uji Coba Fungsionalitas

Uji coba fungsionalitas merupakan sebuah pengujian yang dilakukan untuk menguji bekerja atau tidaknya fungsi – fungsi utama yang ada pada sistem. Fungsionalitas yang diuji dalam bagian ini yaitu fungsionalitas dari setiap node. Node yang dimaksud meliputi website, sms *receiver*, node sensor dan Android gateway.

5.2.1 Skenario Uji Coba (UJ-F01) - Menampilkan Data Halaman *Dashboard*

Uji coba ini dijalankan dengan tujuan untuk menguji fungsionalitas dari website pada halaman utama. Pada halaman utama ditampilkan data-data sensor dalam bentuk tabel. Pengujian fungsionalitas ini diawali dengan *login* pada website terlebih dahulu. **Kesalahan! Sumber referensi tidak ditemukan.** adalah tabel mengenai skenario uji coba menampilkan data dari satu container.

Tabel 5.1 Skenario Uji Coba Menampilkan Data di Halaman *Dashboard*

ID	**UJ-F01**
Nama	Uji Coba Menampilkan Data di Halaman Dashboard
Tujuan Uji Coba	Menguji fungsionalitas website untuk menampilkan data di halaman dashboard.
Kondisi Awal	Membuka halaman website
Skenario	1. Membuka halaman website 2. *Login* sebagai Pengguna 3. Pengguna diarahkan ke halaman dashboard
Masukan	-
Keluaran	Data sensor dalam grafik
Hasil yang Diharapkan	Tampilan halaman utama yang berisi data – data sensor dalam grafik

5.2.2 Skenario Uji Coba (UJ-F02) - Menampilkan Gambar di Halaman *Photo*

Uji coba ini dijalankan dengan tujuan untuk menguji fungsionalitas dari website pada halaman *photo*. Pada halaman *photo* ditampilkan data-data gambar dalam bentuk tabel. Tabel 5.2 adalah

tabel mengenai skenario uji coba menampilkan data gambar dari satu container.

Tabel 5.2 Skenario Uji Coba Menampilkan Gambar di Halaman *Photo*

ID	**UJ-F02**
Nama	Uji Coba Menampilkan Gambar di Halaman *Photo*
Tujuan Uji Coba	Menguji fungsionalitas website untuk menampilkan gambar di halaman *photo*.
Kondisi Awal	Membuka halaman website
Skenario	1. Website menampilkan halaman dashboard 2. Penggunan memilih menu Photo 3. Website menampilkan gambar dari kondisi di lapangan
Masukan	-
Keluaran	Gambar-gambar dari database container
Hasil yang Diharapkan	Tampilan halaman photo yang berisi data – data gambar

5.2.3 Skenario Uji Coba (UJ-F03) - Menampilkan Data di Halaman *Table List*

Uji coba ini dijalankan dengan tujuan untuk menguji fungsionalitas dari website pada halaman tabel. Pada halaman tabel ditampilkan data-data sensor dalam bentuk tabel. Tabel 5.3 adalah tabel mengenai skenario uji coba menampilkan data dari satu container.

Tabel 5.3 Skenario Uji Coba Menampilkan Data di Halaman *Table List*

ID	**UJ-F03**
Nama	Uji Coba Menampilkan Data di Halaman Tabel
Tujuan Uji Coba	Menguji fungsionalitas website untuk menampilkan gambar di halaman *photo*.
Kondisi Awal	Membuka halaman website
Skenario	1. Website menampilkan halaman dashboard 2. Penggunan memilih menu Table 3. Website menampilkan data sensor
Masukan	-
Keluaran	Data-data sensor dalam bentuk tabel

Hasil yang Diharapkan	Tampilan halaman photo yang berisi data – data sensor dalam bentuk tabel

5.2.4 Skenario Uji Coba (UJ-F04) - Menampilkan Notifikasi di Halaman Notification

Uji coba ini dijalankan dengan tujuan untuk menguji fungsionalitas dari website pada halaman notifikasi. Pada halaman notifikasi ditampilkan data-data sensor yang masuk ke dalam tabel notifikasi. Tabel 5.3 adalah tabel mengenai skenario uji coba menampilkan notifikasi di halaman *notification.*

Tabel 5.4 Skenario Uji Coba Menampilkan Notifikasi di Halaman *Notifications*

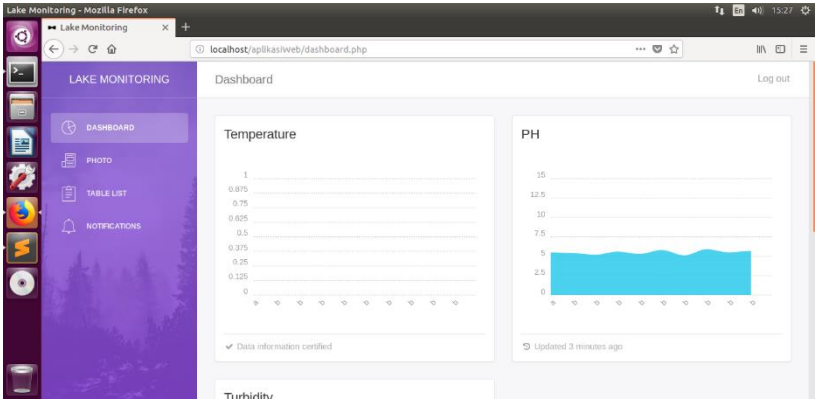
ID	**UJ-F03**
Nama	Uji Coba Menampilkan Notifikasi di Halaman Notification
Tujuan Uji Coba	Menguji fungsionalitas website untuk menampilkan notifikasi di halaman *notification*.
Kondisi Awal	Website menampilkan halaman dashboard
Skenario	1. Website menampilkan halaman dashboard 2. Penggunan memilih menu *Notification* 3. Website menampilkan data sensor
Masukan	-
Keluaran	Data-data sensor di tabel *notification* dalam database
Hasil yang Diharapkan	Tampilan halaman *notification* yang berisi data – data sensor di tabel *notification* dalam database

5.3 Hasil Uji Coba Fungsionalitas

Telah dijabarkan pada bagian sebelumnya mengenai skenario dari keseluruhan uji coba fungsionalitas. Skenario uji coba yang telah dijabarkan terdiri dari beberapa hal.

5.3.1 Hasil Uji Coba (UJ-F01) - Menampilkan Data di Halaman *Dashboard*

Menurut skenario pada bab 0, Pengguna pada awalnya akan *login* ke dalam website dengan menggunakan *username* dan *password* yang telah dimiliki. Setelah berhasil *login*, maka akan ditampilkan sebuah halaman utama yang menyajikan sensor – sensor yang telah disesuaikan dengan kelompok masing – masing. Pada uji coba yang telah dilakukan, didapatkan tampilan seperti pada **Kesalahan! Sumber referensi tidak ditemukan.**. Hal ini menunjukkan bahwa fungsionalitas menampilkan sensor berdasarkan kelompok bekerja dengan baik.

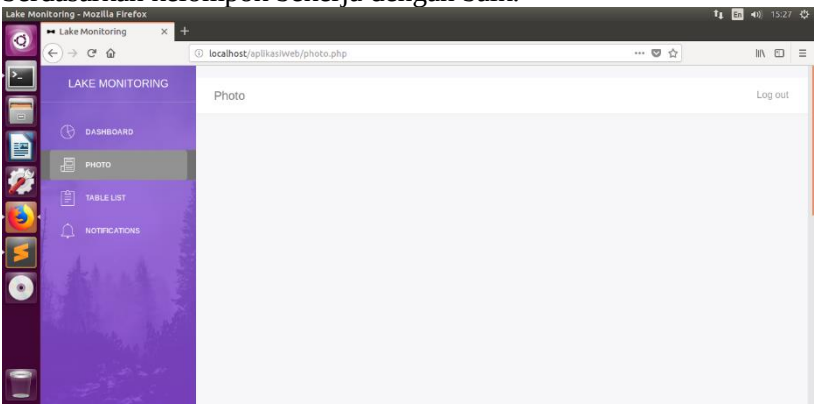


Gambar 5.7 Halaman *Dashboard*

5.3.2 Hasil Uji Coba (UJ-F02) - Menampilkan Gambar di Halaman *Photo*

Menurut skenario pada bab 0, Pengguna pada awalnya akan *login* ke dalam website dengan menggunakan *username* dan *password* yang telah dimiliki. Setelah berhasil *login*, maka akan ditampilkan sebuah halaman utama yang menyajikan sensor – sensor yang telah disesuaikan dengan kelompok masing – masing. Pada uji coba yang telah dilakukan, didapatkan tampilan seperti pada **Kesalahan! Sumber referensi tidak ditemukan.**. Hal ini

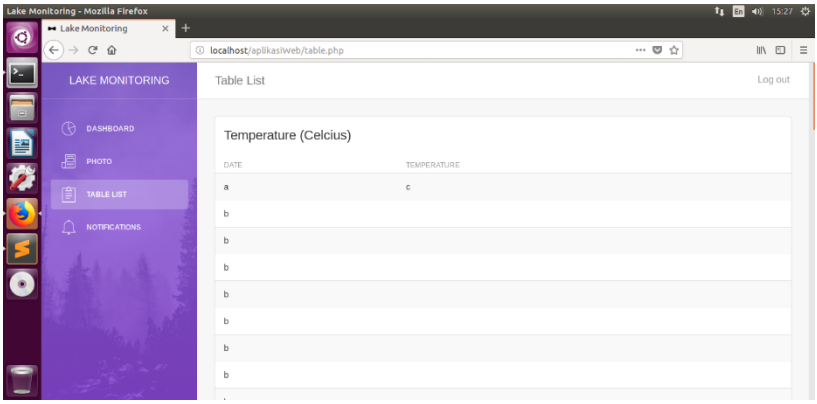
menunjukkan bahwa fungsionalitas menampilkan sensor berdasarkan kelompok bekerja dengan baik.



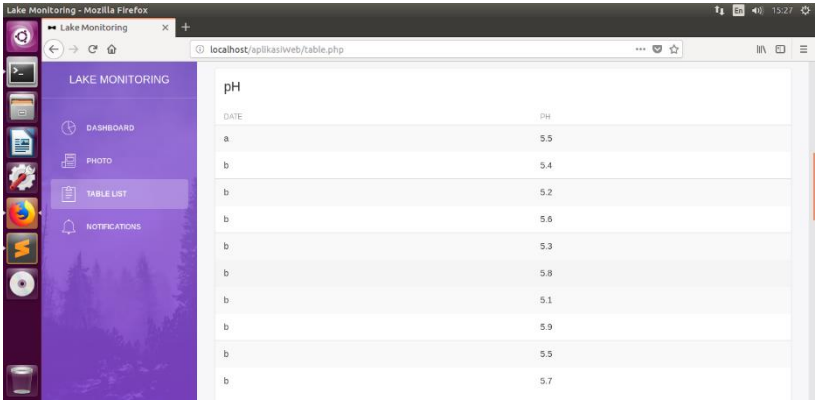
Gambar 5.8 Halaman *Photo*

5.3.3 Hasil Uji Coba (UJ-F03) – Menampilkan Data di Halaman *Table*

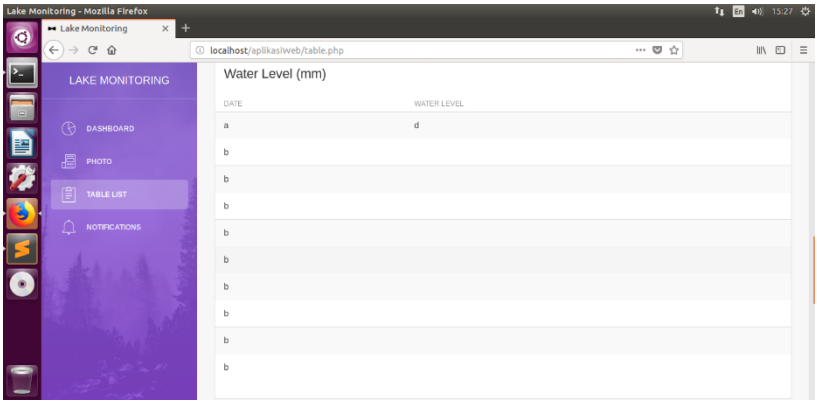
Menurut skenario pada bab 0, Pengguna pada awalnya akan *login* ke dalam website dengan menggunakan *username* dan *password* yang telah dimiliki. Setelah berhasil *login*, maka akan ditampilkan sebuah halaman utama yang menyajikan sensor – sensor yang telah disesuaikan dengan kelompok masing – masing. Pada uji coba yang telah dilakukan, didapatkan tampilan seperti pada **Kesalahan! Sumber referensi tidak ditemukan.**. Hal ini menunjukkan bahwa fungsionalitas menampilkan sensor berdasarkan kelompok bekerja dengan baik.



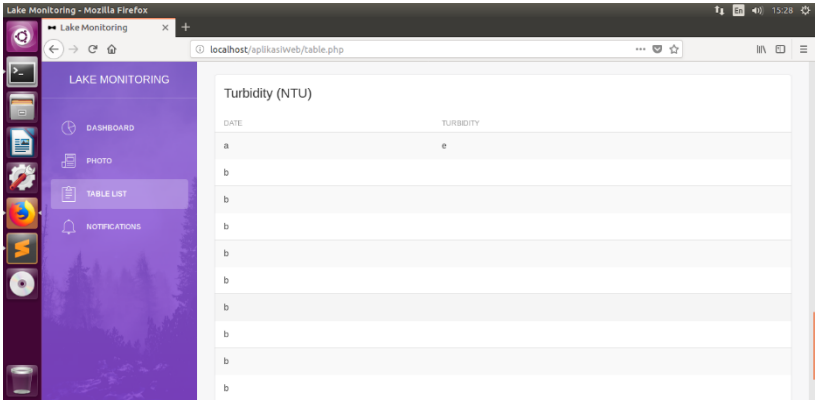
Gambar 5.9 Halaman *Table List 1*



Gambar 5.10 Halaman *Table List 2*



Gambar 5.11 Halaman *Table List 3*

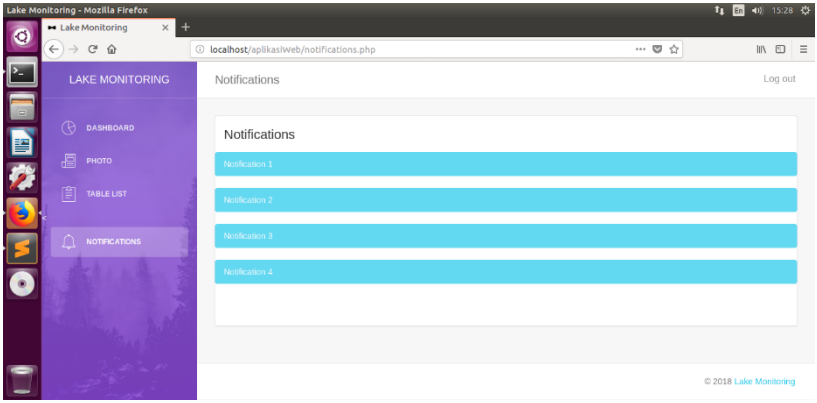


Gambar 5.12 Halaman *Table List 4*

5.3.4 Hasil Uji Coba (UJ-F04) - Menampilkan Notifikasi di Halaman *Notifications*

Menurut skenario pada bab 0, Pengguna pada awalnya akan *login* ke dalam website dengan menggunakan *username* dan *password* yang telah dimiliki. Setelah berhasil *login*, maka akan ditampilkan sebuah halaman utama yang menyajikan sensor – sensor yang telah disesuaikan dengan kelompok masing – masing. Pada uji coba yang telah dilakukan, didapatkan tampilan seperti pada

Kesalahan! Sumber referensi tidak ditemukan.. Hal ini menunjukkan bahwa fungsionalitas menampilkan sensor berdasarkan kelompok bekerja dengan baik.



Gambar 5.13 Halaman *Notifications*

5.4 Skenario Uji Coba Performa

Uji coba performa merupakan sebuah pengujian yang dilakukan untuk menguji performa kerja dari sistem yang telah dibuat. Skenario Uji Coba Peforma yaitu *strest test website* dan performa ketersediaan dari setiap container.

5.4.1 Skenario Uji Coba (UJ-P01) – *Stress Test Website*\

Skenario ini akan menggunakan aplikasi JMeter. Jumlah user yang akan digunakan untuk skenario ini berjumlah 500 user. 500 user nantinya akan mengakses website *Monitoring Kualitas Air*.

Tabel 5.5 Skenario Uji Coba (UJ-P01) – *Stress Test Website*

Nama	Uji Coba *Stress Test Website*
Tujuan Uji Coba	Menguji performa website, jika banyak user yang mengakses website tersebut
Kondisi Awal	Buka Aplikasi Jmeter
Skenario	1. Membuat *Thread Group* pada *Test Plan* 2. Masukkan jumlah user sebanyak 500 user

	3. Membuat HTTP Request Sampler 4. Masukkan IP, *Path*, dan *method website* 5. Tambahkan Listener untuk mengetahui hasilnya 6. Jalankan pekerjaan 7. Hasil keluar
Masukan	-
Keluaran	Hasil dari percobaan
Hasil yang Diharapkan	Mendapatkan semua user berhasil mengakses website

5.4.2 Skenario Uji Coba (UJ-P02) – Ketersediaan Container

Skenario dimulai dengan membuat dua container dengan image *mysql* yang aktif beserta database masing-masing. Lalu salah satu container akan dihentikan. *Availability* atau ketersediaan dari docker yang nantinya akan diketahui hasilnya

Tabel 5.6 Skenario Uji Coba Ketersediaan Container

Nama	Uji Coba Ketersediaan Container
Tujuan Uji Coba	Menguji performa yang berupa ketersediaan container, jika container lain diberhentikan
Kondisi Awal	Membuat 2 container yang berisi database dari masing-masing kolam
Skenario	1. Menjalankan 2 container docker 2. Memberhentikan pekerjaan salah satu container 3. Mengecek ketersediaan container yang masih akif
Masukan	-
Keluaran	Container yang ada
Hasil yang Diharapkan	Mendapatkan container yang masih aktif

5.4.3 Skenario Uji Coba (UJ-P03) – Pemakaian Energi dari Arduino dan Sensor

Skenario dimulai dengan menghitung arus tegangan dari baterai yang digunakan untuk uji coba sebelum melakukan

percobaan. Arduino dan sensor - sensor akan dinyalakan. Ada dua percobaan yang akan dilakukan, yang pertama adalah pengiriman data ke server dengan delay 1 menit selama setengah jam, yang kedua adalah pengiriman data ke server dengan delay 1 jam selama setengah jam. Lalu baterai akan dihitung arus tegangan nya kembali menggunakan multimeter dan dihitung selisih awal dan akhir pemakaian agar didapatkan berapa energi yang dipakai untuk melakukan pekerjaan masing - masing kasus.

Tabel 5.7 Skenario Uji Coba Pemakaian Energi dari Arduino dan Sensor

Nama	Uji Coba Pemakaian Energi dari Arduino dan Sensor
Tujuan Uji Coba	Menghitung berapa energi yang dipakai untuk melakukan pekerjaan yang dibutuhkan
Kondisi Awal	Perhitungan baterai yang dipakai untuk melakukan uji coba
Skenario	1. Arduino mengirimkan data per-satu menit selama setengah jam 2. Baterai diukur berapa tegangan arusnya 3. Arduino mengirimkan data per-satu jam selama setengah jam 4. Baterai diukur berapa tegangan arusnya
Masukan	-
Keluaran	Tegangan arus dari baterai yang diapakai untuk uji coba
Hasil yang Diharapkan	Mendapatkan nilai energi yang dipakai untuk melakukan pekerjaan masing- masing

5.5 Hasil Uji Coba Peforma

Pada bagian sebelumnya telah dijelaskan mengenai skenario dari uji coba performa. Berikut ini akan dijabarkan mengenai hasil dari uji coba performa yang telah dijalankan.

5.5.1 Hasil Uji Coba (UJ-P01) – Stress Test Website

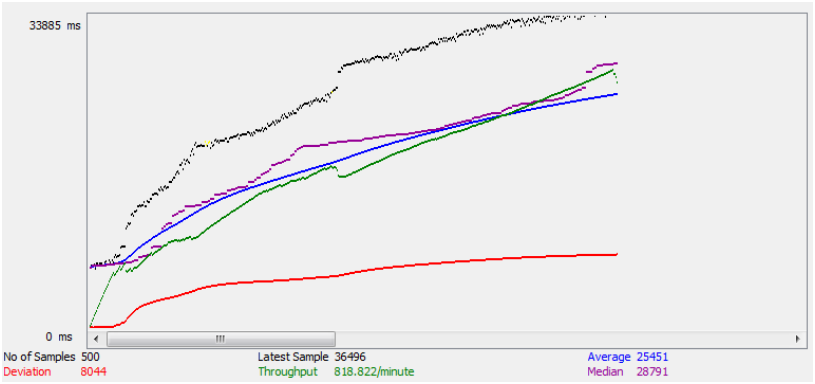
Hasil yang didapat dari Skenario Uji Coba (UJ-P01) terletak di Tabel 5-7, Tabel 5-8, Gambar 5.14 , dan Gambar 5.15. Terdapat 5 user dari 500 user yang tidak berhasil mengakses website *Monitoring Kualitas Air* seperti di Gambar .

Tabel 5.8 *Summary Report 1*

	User	Response Time (ms)			Std. Dev.	Error %
		Average	Min	Max		
Total	500	2541	6484	36496	8044.23	1.00

Tabel 5.9 *Summary Report 2*

	User	Troughput (/sec)	Received KB/sec	Sent KB/sec	Avg Byte
Total	500	13.6	825.2	3.4	6919.9



Gambar 5.14 Grafik Hasil dari Uji Coba

Sample #	Start Time	Thread Name	Label	Sample Ti...	Status	Bytes	Sent Bytes	Latency	Connect Ti...
481	03:44:49.553	Group 1 1-476	HTTP Request	34054	✓	62540	257	11744	435
482	03:44:49.060	Group 1 1-146	HTTP Request	34547	✓	62540	257	12197	916
483	03:44:49.165	Group 1 1-248	HTTP Request	34443	✓	62534	257	12351	1059
484	03:44:49.060	Group 1 1-180	HTTP Request	34586	✓	62540	257	12111	912
485	03:44:49.289	Group 1 1-348	HTTP Request	34373	✓	62540	257	11977	687
486	03:44:49.284	Group 1 1-217	HTTP Request	34379	✓	62540	257	11894	688
487	03:44:49.587	Group 1 1-411	HTTP Request	34080	✓	62540	257	11757	406
488	03:44:49.366	Group 1 1-327	HTTP Request	34305	✓	62544	257	17193	860
489	03:44:49.836	Group 1 1-495	HTTP Request	33843	✓	62540	257	11493	156
490	03:44:49.085	Group 1 1-116	HTTP Request	34596	✓	62540	257	12163	892
491	03:44:49.227	Group 1 1-277	HTTP Request	34467	✓	62540	257	12066	759
492	03:44:49.336	Group 1 1-361	HTTP Request	34357	✓	62534	257	5284	623
493	03:44:49.244	Group 1 1-285	HTTP Request	34474	✓	62540	257	11949	732
494	03:44:49.060	Group 1 1-149	HTTP Request	34659	✓	62540	257	12184	913
495	03:44:49.271	Group 1 1-53	HTTP Request	34454	✓	62536	257	23040	774
496	03:44:49.066	Group 1 1-124	HTTP Request	34656	✓	62546	257	2035	852
497	03:44:49.207	Group 1 1-8	HTTP Request	35111	✓	62544	257	17379	1029
498	03:44:49.205	Group 1 1-21	HTTP Request	35205	✓	62544	257	17380	1030
499	03:44:49.176	Group 1 1-252	HTTP Request	35674	✓	62544	257	17343	1049
500	03:44:49.200	Group 1 1-239	HTTP Request	36496	✓	62544	257	17354	1026
110	03:44:49.136	Group 1 1-207	HTTP Request	20221	✗	918	257	17344	1087
111	03:44:49.375	Group 1 1-331	HTTP Request	19986	✗	918	257	17109	848
113	03:44:49.233	Group 1 1-255	HTTP Request	20129	✗	918	257	17249	990
114	03:44:49.144	Group 1 1-211	HTTP Request	20219	✗	918	257	17339	1079
230	03:44:49.058	Group 1 1-157	HTTP Request	25719	✗	918	257	23210	999

Gambar 5.15 Tabel Uji Coba

5.5.2 Hasil Uji Coba (UJ-P02) – Ketersediaan Container Docker

Terdapat 2 container docker dengan image *mysql:5.7*. Masing-masing container diberi nama *cont1* dan *cont2* seperti Gambar 5.16. Container *cont1* memiliki database seperti ditunjukkan di Gambar 5.17.

```
adiwinoto@wintol:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAM
ES
2fdbc6c81865        mysql:5.7           "docker-entrypoint..."   3 months ago        Up About a minute   3306/tcp            cob
ata
aba550deb4c2        mysql:5.7           "docker-entrypoint..."   3 months ago        Up 33 seconds       3306/tcp            cob
ata2
adiwinoto@wintol:~$
```

Gambar 5.16 Jumlah Container yang Tersedia

Setelah itu container *cont1* di-*stop* menggunakan bash command *docker stop* seperti pada Gambar 5.18. *Availability* atau ketersediaan dari container *cont1* masih aktif seperti ditunjukkan di Gambar 5.19.

```
adiwinoto@wintol:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAM
ES
2fdbc6c81865        mysql:5.7           "docker-entrypoint..."   3 months ago        Up About a minute   3306/tcp            cob
ata
aba550deb4c2        mysql:5.7           "docker-entrypoint..."   3 months ago        Up 33 seconds       3306/tcp            cob
ata2
adiwinoto@wintol:~$ docker exec -it cobata2 mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.21 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.04 sec)

mysql>
```

Gambar 5.17 Database di dalam *cont1*

```
adiwinoto@wintol:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAM
ES
2fdbc6c81865        mysql:5.7           "docker-entrypoint..."   3 months ago        Up 3 minutes        3306/tcp            cob
ata
aba550deb4c2        mysql:5.7           "docker-entrypoint..."   3 months ago        Up 3 minutes        3306/tcp            cob
ata2
adiwinoto@wintol:~$ docker stop cobata2
cobata2
adiwinoto@wintol:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAM
ES
2fdbc6c81865        mysql:5.7           "docker-entrypoint..."   3 months ago        Up 3 minutes        3306/tcp            cob
ata
adiwinoto@wintol:~$
```

Gambar 5.18 Penghentian *cont2*

```
adiwinoto@wintol:~$ docker exec -it cobata mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.21 MySQL Community Server (GPL)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.11 sec)

mysql>
```

Gambar 5.19 Ketersediaan *cont1*

5.5.3 Hasil Uji Coba (UJ-P03) – Pemakaian Energi dari Arduino dan Sensor

Hasil yang didapat dari Skenario Uji Coba (UJ-P03) terletak di Tabel 5-10. Arus tegangan yang dipakai untuk melakukan *testing* yang pertama, yaitu pengiriman data per-satu menit selama setengah jam adalah 12A, sedangkan pengiriman data per-satu jam selama setengah adalah 3A. Sehingga mendapatkan hasil 6.67×10^{-3} A/s untuk percobaan pertama dan 1.67×10^{-3} A/s

Tabel 5.10 Hasil Uji Coba Peforma

Uji Coba	Kondisi Baterai Awal		Kondisi Baterai Akhir		Selisih (A)
	X (A)	Y (A)	X (A)	Y (A)	
Delay 1 menit	136	138	130	132	12
Delay 1 jam	137	136	135	135	3

5.6 Evaluasi Hasil Uji Coba

Pada bagian sebelumnya, telah dilakukan uji coba terhadap sistem yang dibuat. Uji coba yang telah dilakukan berkenaan dengan uji coba fungsionalitas dan uji coba performa. Berikut ini Tabel 5.5 merangkum evaluasi hasil uji coba fungsionalitas dan Tabel 5.6 merangkum hasil uji coba performa yang telah dilakukan.

Tabel 5.11 Evaluasi Hasil Uji Coba Fungsionalitas

No	Kode Uji Coba	Evaluasi
1	UJ-F01	Dapat menampilkan data di halaman *dashboard*
2	UJ-F02	Dapat menampilkan gambar di halaman *photo*
3	UJ-F03	Dapat menampilkan data di halaman *table list*
4	UJ-F04	Dapat menampilkan notifikasi di halaman

		notifications

Tabel 5.12 Evaluasi Hasil Uji Coba Peforma

No	Kode Uji Coba	Evaluasi
1	UJ-P01	Terdapat 5 user dari 500 user yang tidak berhasil mengakses website
2	UJ-P02	Ketersediaan atau *avaibility* dari container docker terbukti
3	UJ-P03	Pengiriman data ke server sangat mempengaruhi daya yang dikeluarkan alat sensor

BAB VI
KESIMPULAN DAN SARAN

Bagian terakhir yang dibahas dalam buku ini adalah kesimpulan dan saran. Kesimpulan yang dijabarkan di bawah ini didapatkan berdasarkan dari hasil pengamatan uji coba sistem. Selain kesimpulan, dijabarkan pula saran yang diharapkan dapat berguna untuk pengembangan sistem dan penelitian selanjutnya.

6.1 Kesimpulan

Kesimpulan yang didapatkan berdasarkan hasil uji coba fungsional dan performa adalah sebagai berikut:

1. *Monitoring Kualitas Air* dapat memberikan layanan monitoring secara detail, real-time, dan tepat.
2. Database dari masing-masing *cluster* air akan terisolir satu sama lain (keamanan terjaga)
3. Jika terjadi gangguan di salah satu *container*, *container* lain tetap tersedia (ketersediaan terjaga)
4. User bisa mengakses website cukup baik dengan error sebesar 1.00%

6.2 Saran

Saran yang diberikan untuk pengembangan penelitian kedepannya adalah sebagai berikut:

1. Container docker perlu ditambah *web service*, agar pekerjaan tidak bertumpu pada komputer host.
2. Node sensor perlu adanya trigger untuk mendeteksi sesuatu, tidak hanya mendeteksi secara berkala.

.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] W. P. A. N. (. W. Group, "IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN)," pp. 1-700, 14 June 2005.

[2] M. D. Nuryanto, Rancang Bangun Monitoring Server dan Gateaway Adaptif untuk Cluster Jaringan Sensor Nirkabel Tersebar, Surabaya: Institut Teknologi Sepuluh Nopember, 2017.

[3] R. Abidin, "Pengertian dan Istilah Pada Docker. Retrieved from TeknoJurnal," 29 July 2016. [Online]. Available: https://teknojurnal.com/pengertian-dan-istilah-pada-docker.

[4] I. Harfiansyah, "Mengenal Teknologi Docker," Blogspot, 18 Juli 2016. [Online]. Available: https://www.codepolitan.com/mengenal-teknologi-docker.

[5] "Prinsip Kerja pH Meter," Artikel Teknologi Indonesia, [Online]. Available: https://artikel-teknologi.com/prinsip-kerja-ph-meter/.

[6] "Turbidity Meter disebut juga Alat Ukur Kekeruhan Air," Digital Meter Indonesia, [Online]. Available: https://indo-digital.com/turbidity-meter-disebut-juga-alat-ukur-kekeruhan-air.html.

[7] G. W. Pambudi, "Cara Mengukur ketinggian Air menggunakan Water Level Sensor Arduino," Creative Technology Indonesia, 2017 Desember 17. [Online]. Available: https://www.cronyos.com/cara-mengukur-ketinggian-air-menggunakan-water-level-sensor-arduino/.

[8] S. P. Irawan, "Pelajari tentang Sensor Suhu DS18B20 dan bagaimana penyambungan alat tersebut sebagai input pada

perangkat Raspberry Pi sebagai sensor suhu sebuah ruangan.," KL801, 26 Februari 2017. [Online]. Available: http://kl801.ilearning.me/2017/02/26/pelajari-tentang-sensor-suhu-ds18b20-dan-bagaimana-penyambungan-alat-tersebut-sebagai-input-pada-perangkat-raspberry-pi-sebagai-sensor-suhu-sebuah-ruangan/.

9] Z. Yulias2, "Tutorial Breadboard untuk Arduino," Famosa Studio, 10 Juni 2011. [Online]. Available: http://blog.famosastudio.com/2011/06/tutorial/tutorial-breadboard-untuk-arduino/59.

LAMPIRAN

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Adiwinoto Saptorenggo merupakan anak dari pasangan Bapak Anjar Saptorenggo dan Ibu Nanik Wulandarini. Lahir di Jakarta pada tanggal 03 Mei 1996. Penulis menempuh pendidikan formal dimulai dari TK Puri Asih Jakarta (2000-2002), SD Budi Wanita Jakarta (2002-2005), SDN Menteng 01 Jakata (2005-2008), SMP Islam PB Soedirman Jakarta (2008-2011), SMAN 14 Jakarta (2011-2014) dan sesudah lulus dari SMAN 14 Jakata melanjutkan menimba ilmu di jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya (2014-2018). Bidang studi yang diambil oleh penulis pada saat berkuliah di Teknik Informatika ITS adalah Komputasi Berbasis Jaringan (KBJ). Penulis aktif dalam organisasi seperti Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) sebagai Staf Departemen Hubungan Luar HMTC 2015-2016 dan Staf Ahli Departemen Hubungan Luar HMTC 2016-2017 dan juga organisasi seperti Badan Ekskutif Mahasiswa Fakultas Teknologi Informasi (BEM FTIf) sebagai Staf Organization of Social Resposibility (OSR) BEM FTIf 2015-2016. Selain itu, juga memiliki pengalaman kepanitiaan, diantaranya sebagai Staf National Seminar of Technology Schematics 2015 dan Badan Pengurus Harian (BPH) National Seminar of Techonlogy Schematics 2016. Penulis memiliki hobi membaca buku dan menyukai hal baru. Penulis dapat dihubungi melalui email: adiwinotosaptorenggo@gmail.com.