



**UNIVERSITAS
PRASETIYA MULYA**

TUGAS AKHIR

**DESIGN AND IMPLEMENTATION OF IOT-
BASED AGILITY TESTER FOR SPORTS
PERFORMANCE EVALUATION**

**NENGAH SWARDANA
23401910002**

**PROGRAM STUDI S1 REKAYASA SISTEM KOMPUTER
UNIVERSITAS PRASETIYA MULYA
JAKARTA, 2023**



**UNIVERSITAS
PRASETIYA MULYA**

**DESIGN AND IMPLEMENTATION OF IOT-
BASED AGILITY TESTER FOR SPORTS
PERFORMANCE EVALUATION**

Oleh

NENGAH SWARDANA
23401910002

**TUGAS AKHIR INI SEBAGAI SALAH SATU SYARAT UNTUK
MEMPEROLEH GELAR SARJANA KOMPUTER**

**PROGRAM STUDI S1 REKAYASA SISTEM KOMPUTER
UNIVERSITAS PRASETIYA MULYA
JAKARTA, 2023**

PERNYATAAN KEASLIAN TUGAS AKHIR

Saya menyatakan dengan sesungguhnya bahwa tugas akhir dengan judul:

DESIGN AND IMPLEMENTATION OF IOT-BASED AGILITY TESTER FOR SPORTS PERFORMANCE EVALUATION

adalah hasil karya sendiri, dan semua sumber baik yang dikutip atau dirujuk sudah saya nyatakan dengan benar.

Jakarta, 02 Februari 2023

Yang menyatakan,



Nengah Swardana
23401910002

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR

Yang bertanda tangan di bawah ini, saya mahasiswa Sekolah STEM Terapan Universitas Prasetiya Mulya:

Nama : Nengah Swardana
NIM : 23401910002
Program Studi : Rekayasa Sistem Komputer

memberikan kepada Universitas Prasetiya Mulya Hak Bebas Royalti Non Eksklusif (*Non-exclusive Royalty Free Right*) atas karya ilmiah saya baik dalam bentuk teks lengkap maupun ringkasan yang berjudul:

DESIGN AND IMPLEMENTATION OF IOT-BASED AGILITY TESTER FOR SPORTS PERFORMANCE EVALUATION

beserta perangkat yang diperlukan (bila ada). Dengan demikian saya memberikan hak kepada Universitas Prasetiya Mulya untuk menyimpan, mengalih media/formatkan, mengelola dalam bentuk pangkalan data, merawat, dan mempublikasikan tugas akhir saya untuk kepentingan akademis tanpa perlu meminta izin dari saya maupun memberikan royalti kepada saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Jakarta, 02 Februari 2023

Yang menyatakan,



Nengah Swardana
23401910002

PENGESAHAN TUGAS AKHIR

Tugas akhir dengan judul:

DESIGN AND IMPLEMENTATION OF IOT-BASED AGILITY TESTER FOR SPORTS PERFORMANCE EVALUATION

Disusun oleh:

Nama : Nengah Swardana

NIM : 2340191002

Program Studi : Rekayasa Sistem Komputer

untuk melengkapi sebagian persyaratan menjadi Sarjana Sarjana Komputer pada pendidikan Sarjana Universitas Prasetya Mulya, telah diterima dan disahkan.

Jakarta, 02 Februari 2023



Djukarna, M.T.

Pembimbing 1

Menyetujui,



Edwin Yudayana, M.T.

Pembimbing 2

Mengesahkan,



Agung Alfiansyah, Ph.D.

Ketua Program Studi S1 Rekayasa Sistem Komputer

UCAPAN TERIMA KASIH

Segala puji dan syukur ke hadirat Tuhan Yang Maha Esa peneliti haturkan karena berkat rahmat dan anugerah-Nya, akhirnya peneliti telah berhasil menyelesaikan laporan tugas akhir yang berjudul:

“DESIGN AND IMPLEMENTATION OF IOT-BASED AGILITY TESTER FOR SPORTS PERFORMANCE EVALUATION”

Laporan tugas akhir ini disusun untuk memenuhi persyaratan guna memperoleh gelar Sarjana Komputer pada Sekolah Terapan STEM Universitas Prasetiya Mulya.

Peneliti menyadari bahwa tanpa adanya dukungan dari berbagai pihak, penyusunan skripsi ini tidak akan pernah terwujud. Maka pada kesempatan ini, peneliti ingin menyampaikan banyak terima kasih kepada berbagai pihak, yaitu:

1. Bapak Agung Alfiansyah, Ph.D. selaku Ketua Program Studi S1 Rekayasa Sistem Komputer Universitas Prasetiya Mulya.
2. Bapak Djukarna, M.T selaku pembimbing satu dan Bapak Edwin Yudayana, M.T. selaku pembimbing dua yang telah membimbing dan memberikan kritik, saran, dan masukan untuk menyempurnakan tugas akhir ini.
3. Seluruh *Faculty Member* baik dari Program Studi S1 Rekayasa Sistem Komputer maupun *Faculty Member* program studi lainnya atas ilmu yang telah diajarkan.
4. Untuk Orang Tua serta keluarga besar yang ada di rumah atas semangat dan motivasi bagi penulis untuk menyelesaikan tugas akhir ini.
5. Mahasiswa S1 Rekayasa Sistem Komputer angkatan 2019 yang menjadi teman seperjuangan saat perkuliahan berlangsung.

Dalam penyusunan laporan tugas akhir ini penulis menyadari bahwa laporan ini masih jauh dari kata sempurna, karena didalamnya masih terdapat kekurangan-kekurangan. Hal ini dikarenakan keterbatasan yang dimiliki oleh penulis baik dalam segi kemampuan, pengetahuan serta pengalaman penulis. Oleh sebab itu penulis

mengharapkan kritik dan saran yang sifatnya membangun agar dalam penyusunan karya tulis selanjutnya dapat menjadi lebih baik.

Akhir kata, penulis mengucapkan terima kasih sekali lagi dan berharap agar tugas akhir yang dikerjakan dapat berguna bagi semua pihak.

Jakarta, 02 Februari 2023

A handwritten signature in black ink, appearing to read 'Nengah Swardana', with a stylized flourish at the end.

Nengah Swardana

ABSTRAK

Kelincahan merupakan salah satu komponen penting bagi atlet. Atlet dapat melatih tingkat kelincahannya dengan metode latihan konvensional seperti *shuttle run*, *squat thrust* dan *zig zag run*, namun metode latihan ini hanya mencakup komponen CODS, sehingga untuk melatih kelincahan pada aspek pengambilan keputusan berdasarkan sebuah rangsangan diperlukan sebuah metode latihan yang lebih modern. Selain itu, hingga saat ini, sebagian besar tes kelincahan masih dievaluasi secara manual dengan menggunakan *stopwatch* sehingga berpotensi menimbulkan kesalahan pencatatan waktu karena adanya *human error*. Berdasarkan permasalahan tersebut peneliti merancang sebuah alat *agility tester* berbasis IoT, alat ini mampu melatih kelincahan pada aspek *perceptual and decision making* dengan *output* berupa NeoPixel LED, selain itu untuk melakukan pencatatan waktu secara otomatis dan presisi, alat ini menggunakan modul RTC DS1302. Alat ini terdiri atas node, *control box*, *database*, dan *web application*, adapun metode komunikasi yang digunakan antara node dan *control box* adalah komunikasi radio dengan memanfaatkan modul nRF24L01. Berdasarkan pengujian yang telah dilakukan, alat ini telah mampu melakukan latihan dan tes kelincahan dengan jangkauan maksimum 5 meter dan latensi rata-rata 150,2 ms. Alat ini juga telah mampu menyimpan data hasil latihan dan tes kelincahan kedalam *database* dan menampilkan datanya pada *web dashboard*.

Kata kunci: Kelincahan, *Agility Tester*, *IoT*, RTC DS1302, nRF24L01

ABSTRACT

Agility is one of the essential components for athletes. Athletes can train their agility levels using conventional training methods such as shuttle runs, squat thrusts, and zigzag runs. However, these training methods only cover the Change of Direction Speed (CODS) component. Therefore, to train agility in terms of decision-making based on stimuli, a more modern training method is needed. Furthermore, up until now, most agility tests have been manually evaluated using stopwatches, which can potentially lead to timing errors due to human error. Based on this issue, researchers have designed an IoT-based agility tester. This device is capable of training agility in the perceptual and decision-making aspects, with NeoPixel LED outputs. Additionally, for automatic and precise time recording, the device employs the RTC DS1302 module. The device consists of nodes, control box, a database, and a web application. The communication method utilized between nodes and the control box involves radio communication using the nRF24L01 module. Based on conducted tests, the device has successfully conducted agility training and tests within a maximum range of 5 meters and an average latency of 150.2 ms. Furthermore, the device can store training and agility test data in the database and display this information on a web dashboard.

Keywords: Agility, Agility Tester, IoT, RTC DS1302, nRF24L01.

DAFTAR ISI

PERNYATAAN KEASLIAN TUGAS AKHIR.....	ii
PENGESAHAN TUGAS AKHIR	iv
UCAPAN TERIMA KASIH.....	v
ABSTRAK	vii
ABSTRACT.....	viii
DAFTAR ISI.....	ix
DAFTAR TABEL	xii
DAFTAR GAMBAR	xiii
DAFTAR LAMPIRAN	xv
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan Penelitian.....	3
1.4 Batasan Penelitian	4
BAB 2 TINJAUAN PUSTAKA	5
2.1 Kelincahan	5
2.1.1 <i>Perceptual and Decision Making</i>	5
2.1.2 <i>Change of Direction Speed</i>	5
2.1.3 Latihan Kelincahan	6
2.1.4 Tes Kelincahan	6
2.2 Produk Terkait.....	8
2.2.1 FITLIGHT®	8
2.2.2 SmarTracks	9
2.2.3 SmartSpeed.....	10
2.3 Perangkat Pembangun Sistem	12
2.3.1 Node	12
2.3.1.1 Sensor VL53L0X	12
2.3.1.2 NodeMCU ESP8266.....	13
2.3.1.3 nRF24L01	14
2.3.1.4 <i>WS2812 Bultin Driver Neopixel</i>	16
2.3.1.5 <i>Buzzer SFM-20B</i>	17
2.3.2 CB.....	18

2.3.2.1 RTC DS1302	19
2.3.2.2 OLED 0.96"	20
2.3.3 <i>Cloud Database</i>	20
2.3.4 <i>Web Application</i>	21
BAB 3 METODE PENELITIAN	23
3.1 Waktu dan Tempat Pengerjaan	23
3.2 Metodologi	23
3.3 Perancangan Prototipe	24
3.3.1 Arsitektur Sistem Versi 1	24
3.3.1.1 Node dan CB	25
3.3.1.2 MIT <i>App Inventor</i>	26
3.3.2 Arsitektur Sistem: Versi 2	26
3.3.3 Desain Node	28
3.3.3.1 Arsitektur Node	28
3.3.3.2 Diagram Alir Node	29
3.3.3.3 <i>Wiring Chart</i>	30
3.3.4 Desain CB	32
3.3.4.1 Arsitektur CB	32
3.3.4.2 Diagram Alir CB	33
3.3.4.3 <i>Wiring Chart</i>	34
3.3.5 Komunikasi antara Node dan CB	36
3.3.6 <i>Database Schema</i>	37
3.3.7 Desain <i>Web Application</i>	38
3.3.8 API	39
3.4 Metode Pengujian	44
BAB 4 HASIL DAN PEMBAHASAN	46
4.1 Prototipe <i>Agility Tester</i>	46
4.1.1 Node	46
4.1.2 CB	47
4.1.3 <i>Web Application</i>	47
4.2 Analisis Performa Sistem	50
4.2.1 Jarak Jangkauan	50
4.2.2 Latensi	51
4.3 Implementasi Sistem	53
4.3.1 Latihan Kelincahan	53

4.3.2 Tes Kelincahan	55
BAB 5 KESIMPULAN DAN SARAN.....	57
5.1 Kesimpulan	57
5.2 Saran	57
DAFTAR PUSTAKA	59
LAMPIRAN	61

DAFTAR TABEL

Tabel 2.1 Klasifikasi Kelincahan berdasarkan Skor T-Test.....	8
Tabel 2.2 Detail dari spesifikasi dan fitur yang dimiliki sensor VL53L0X.....	13
Tabel 2.3 Detail dari spesifikasi dan fitur yang dimiliki NodeMCU ESP8266.	14
Tabel 2.4 Detail dari spesifikasi dan fitur yang dimiliki nRF24L01	15
Tabel 2.5 Detail dari WS2812 Builtin Driver Neopixel.....	17
Tabel 2.6 Detail dari spesifikasi dan fitur yang dimiliki buzzer SFM-28-B.....	18
Tabel 2.7 Detail dari spesifikasi dan fitur yang dimiliki RTC DS1302.....	19
Tabel 2.8 Detail dari spesifikasi dan fitur yang dimiliki OLED 0.96”	20
Tabel 3.1 Waktu dan Tempat Pelaksanaan Tugas Akhir	23
Tabel 3.2 Spesifikasi API add-new.php	40
Tabel 3.3 Spesifikasi API edit.php	41
Tabel 3.4 Spesifikasi API delete.php	41
Tabel 3.5 Spesifikasi API getConfig.php	42
Tabel 3.6 Spesifikasi API uploadTrainingData.php.....	43
Tabel 3.7 Metode dan Prosedur Pengujian Performa Sistem.....	44
Tabel 3.8 Metode dan Prosedur Pengujian Fungsionalitas Sistem	45
Tabel 4.1 Hasil Uji Jarak Jangkauan antara Node dan CB	50
Tabel 4.2 Hasil Uji Latensi	52
Tabel 4.3 Hasil Uji Latihan Kelincahan.....	54
Tabel 4.4 Hasil Uji T-Test.....	55

DAFTAR GAMBAR

Gambar 1.1 Komponen Kelincahan	2
Gambar 2.1 FITLIGHT®	8
Gambar 2.2 SmarTracks.....	9
Gambar 2.3 SmartSpeed.....	11
Gambar 2.4 Sensor VL53L0X	12
Gambar 2.5 NodeMCU ESP8266	13
Gambar 2.6 nRF24L01	15
Gambar 2.7 WS2812 Builtin Driver Neopixel.....	16
Gambar 2.8 Buzzer SFM-20B.....	18
Gambar 2.9 RTC DS1302	19
Gambar 2.10 OLED 0.96”.....	20
Gambar 3.1 Diagram Alir Penelitian.....	23
Gambar 3.2 Arsitektur Sistem Agility Tester berbasis IoT: Versi 1	24
Gambar 3.3 Rangkaian Node Versi 1.....	25
Gambar 3.4 Rangkaian CB Versi 1	25
Gambar 3.5 Arsitektur Sistem Agility Tester berbasis IoT: Versi 2.....	27
Gambar 3.6 Arsitektur Node	28
Gambar 3.7 Diagram Alir Node	29
Gambar 3.8 Desain PCB dari Node	31
Gambar 3.9 Arsitektur CB	32
Gambar 3.10 Diagram Alir CB	33
Gambar 3.11 Fungsi Logika Penentuan nodeID Pada CB.....	34
Gambar 3.12 Tabel Wiring Chart CB	35
Gambar 3.13 Wiring Chart CB	35
Gambar 3.14 Desain PCB dari CB.....	35
Gambar 3.15 Komunikasi saat CB mengirim Pesan ke Node.....	36
Gambar 3.16 Kode Konfigurasi Radio Pada Node (kiri) dan CB (kanan).....	36
Gambar 3.17 Komunikasi saat Node memberikan Feedback kepada CB.....	37
Gambar 3.18 Database Schema dari IoT Based Agility Tester.....	37
Gambar 3.19 Diagram Alir Web Application	38
Gambar 3.20 API yang digunakan pada Arsitektur Sistem: Versi 2.....	39
Gambar 4.1 Prototipe Node.....	46
Gambar 4.2 Prototipe CB	47
Gambar 4.3 Login Page.....	47
Gambar 4.4 Laman Dashboard: Configuration	48
Gambar 4.5 Pop Up Add Configuration.....	49
Gambar 4.6 Pop up Edit Configuration.....	49
Gambar 4.7 Laman Dashboard: Logs Recap	50
Gambar 4.8 Skema Pengujian Latensi	51

Gambar 4.9 Grafik Jarak node dan CB terhadap Latensi.....	53
Gambar 4.10 Setup Latihan Kelincahan dengan 5 Node	53
Gambar 4.11 Barchart Hasil Latihan pada Menu Logs Recap.....	54
Gambar 4.12 Setup T-Test	55
Gambar 4.13 Barchart Hasil Uji T-Test	56

DAFTAR LAMPIRAN

Lampiran 1. Kartu Bimbingan Tugas Akhir	61
Lampiran 2. <i>Log</i> Bimbingan Tugas Akhir	62
Lampiran 3. Surat Persetujuan Sidang Tugas Akhir	65
Lampiran 4. Biaya Penelitian	66
Lampiran 5. <i>Code Control Box</i>	67
Lampiran 6. <i>Code Node</i>	80

DAFTAR SINGKATAN

Singkatan	Kepanjangan
IoT	<i>Internet of Things</i>
ESST	<i>Edgren Side Step Test</i>
IAT	<i>Illinois Agility Test</i>
CODS	<i>Change of Direction Speed</i>
RAT	<i>Reactive Agility Test</i>
ECP	<i>Error Correction Processing</i>
LED	<i>Light Emitting Diode</i>
OLED	<i>Organic Light Emitting Diode</i>
RFID	<i>Radio Frequency Identification</i>
RTC	<i>Real-Time Clock</i>
Wi-Fi	<i>Wireless Fidelity</i>
HTTP	<i>Hypertext Transfer Protocol</i>
CB	<i>Control Box</i>
API	<i>Application Programming Interface</i>

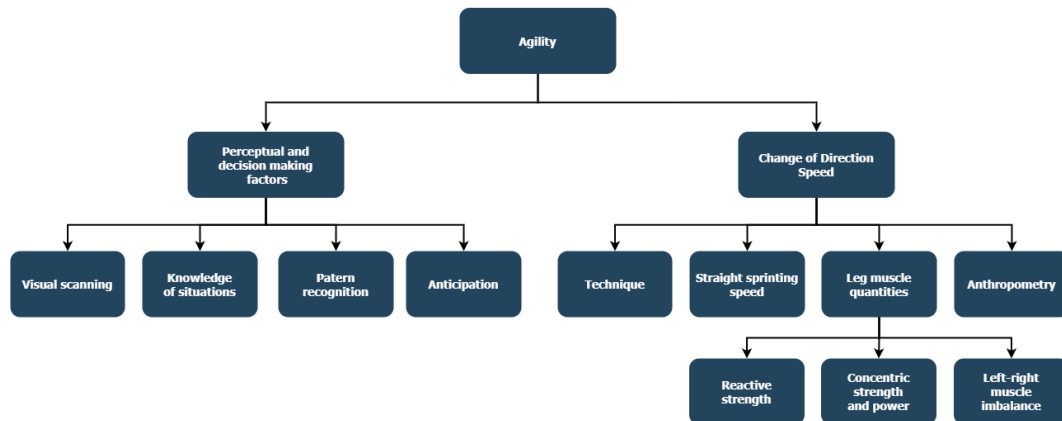
BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam olahraga tim, kemampuan kelincahan memegang peran penting dalam menghindari lawan saat menyerang dan memberikan tekanan pada lawan saat bertahan. Kemampuan kelincahan secara konvensional didefinisikan sebagai kemampuan untuk berubah arah dengan cepat, namun penelitian terbaru menunjukkan bahwa komponen utama dalam kelincahan dapat dibagi menjadi dua komponen, yaitu perubahan arah dan pengambilan keputusan berdasarkan sebuah rangsangan. Ini didasarkan pada fenomena perubahan arah dan kecepatan yang sering terjadi dalam olahraga tim sebagai tanggapan terhadap tindakan lawan. Kelincahan didefinisikan ulang oleh Sheppard dan Young (2006) sebagai pergerakan tubuh yang cepat dengan perubahan kecepatan atau arah sebagai respon terhadap sebuah rangsangan [1,2].

Tingkat kelincahan seseorang dapat ditingkatkan dengan melakukan latihan kelincahan secara rutin. Secara umum latihan kelincahan dapat meningkatkan koordinasi, stabilitas, keseimbangan, dan fleksibilitas. Bagi atlet, latihan kelincahan dapat membantu mempercepat respon terhadap situasi permainan dan memperbaiki tingkat performa [3]. Sementara bagi individu non-atlet, latihan kelincahan dapat membantu meningkatkan kualitas hidup dan mencegah terjadinya cedera akibat aktivitas sehari-hari. Secara konvensional terdapat berbagai macam latihan kelincahan yang dapat dilakukan secara mandiri oleh atlet maupun non-atlet, beberapa diantaranya adalah *shuttle run*, *squat thrust*, lari zig-zag, latihan dengan lompat tangga, dan latihan dengan menggunakan *agility balls*. Namun latihan kelincahan yang dilakukan secara konvensional hanya mencakup komponen CODS pada kelincahan, sehingga untuk melatih kelincahan pada aspek pengambilan keputusan berdasarkan sebuah rangsangan diperlukan sebuah metode latihan yang lebih modern [11].



Gambar 1.1 Komponen Kelincahan

Metode modern untuk melatih kelincahan dalam aspek pengambilan keputusan berdasarkan sebuah rangsangan dikembangkan oleh brand FITLIGHT®. brand ini menggunakan metode “*flash reflex training system*”. Sistem ini menggunakan rangsangan seperti cahaya dan suara untuk melatih seseorang dalam merespons secara cepat dan akurat. Meskipun demikian, produk ini dijual dengan harga yang cukup tinggi sehingga hanya dapat dijangkau oleh orang-orang dengan ekonomi menengah ke atas [7,8].

Tingkat kelincahan seseorang dapat diukur dengan melakukan tes kelincahan, beberapa tes kelincahan yang paling umum digunakan adalah T-Test, ESST, dan IAT [4]. Ketiga tes tersebut dirancang untuk mengukur kecepatan, keseimbangan, dan koordinasi dan membantu dalam memberikan gambaran yang akurat tentang tingkat kelincahan seseorang. Namun, hingga saat ini, sebagian besar tes kelincahan masih dievaluasi secara manual dengan menggunakan *stopwatch*. Hasil dari penelitian sebelumnya menyimpulkan bahwa penggunaan *stopwatch* dalam pencatatan waktu masih memiliki kekurangan, karena ada resiko terjadinya kesalahan pencatatan waktu akibat adanya *human error* [5]. Alternatif pencatatan waktu yang lebih presisi ditawarkan oleh alat *timing gates* dengan brand SmarTracks dan SMARTSPEED, namun kedua produk ini tidak dapat dijangkau oleh masyarakat secara luas karena harga produknya yang cukup tinggi [9,10].

Berdasarkan permasalahan di atas peneliti ingin membuat sebuah alat *agility tester* berbasis IoT yang mampu melakukan pengukuran waktu secara presisi pada saat melakukan latihan dan tes kelincahan, menyimpan data tersebut kedalam *cloud*

database dan menampilkan datanya kedalam *web dashboard* sehingga mempermudah pengguna dalam memantau tingkat kelincahannya secara historis. Dengan adanya alat ini juga diharapkan pengguna dapat melakukan latihan dan tes kelincahan secara mandiri, tanpa perlu didampingi oleh seorang pelatih.

1.2 Rumusan Masalah

Dalam pelaksanaan tugas akhir ini terdapat beberapa rumusan masalah sebagai arah penelitian. Adapun rincian dari rumusan masalah tersebut adalah sebagai berikut:

1. Bagaimana arsitektur dari alat *agility tester* berbasis IoT?
2. Bagaimana mekanisme kerja dari alat *agility tester* berbasis IoT?
3. Bagaimana spesifikasi dari alat *agility tester* berbasis IoT?

1.3 Tujuan Penelitian

Tujuan dari tugas akhir ini adalah untuk merancang sebuah alat *agility tester* berbasis IoT yang mampu melakukan pengukuran waktu secara presisi pada saat melakukan latihan maupun tes kelincahan, menyimpan data tersebut kedalam *cloud database* dan menampilkan datanya kedalam sebuah *web dashboard* sehingga mempermudah pengguna dalam memantau tingkat kelincahannya secara historis. Adapun tahapan yang akan dilakukan untuk mencapai tujuan tersebut sebagai berikut:

1. Membuat desain arsitektur dari alat *agility tester* berbasis IoT.
2. Mendeskripsikan mekanisme kerja dari alat *agility tester* berbasis IoT dalam melakukan pengukuran waktu secara presisi pada saat melakukan latihan dan tes kelincahan, menyimpan data tersebut ke dalam *cloud database* dan menampilkan datanya kedalam *web dashboard*.
3. Menguji spesifikasi dari alat *agility tester* berbasis IoT dalam melakukan pengukuran waktu secara presisi pada saat melakukan latihan dan tes kelincahan, menyimpan data tersebut kedalam *cloud database* dan menampilkan datanya kedalam *web dashboard*.

1.4 Batasan Penelitian

Dalam melaksanakan Tugas Akhir penulis membuat batasan atau cakupan proses pembahasan yang akan dijabarkan sebagai berikut:

1. Arsitektur dari alat *agility tester* berbasis IoT terdiri dari 4 komponen utama yaitu *node*, *control box*, *cloud database* dan *web application server*.
2. Pengujian prototipe dilakukan pada area terbuka yang memiliki koneksi internet stabil dan dengan intensitas cahaya yang tidak terlalu tinggi. Hal ini penting dikarenakan *node* dan *control box* memerlukan koneksi internet untuk mendapatkan konfigurasi yang digunakan, selain itu intensitas cahaya yang tinggi akan mengganggu pembacaan sensor VL53L0X.
3. Pengujian prototipe hanya dilakukan pada kondisi jarak antara *node* dan *control box* maksimum 5 meter.
4. Pengujian yang dilakukan meliputi pengujian jarak jangkauan sistem, latensi dan fungsionalitas sistem pada saat melakukan latihan dan tes kelincahan. Pengujian ini akan dilakukan pada lapangan basket perumahan Verdant Ville, BSD.

BAB 2

TINJAUAN PUSTAKA

2.1 Kelincahan

Kelincahan didefinisikan sebagai pergerakan tubuh yang cepat dengan perubahan kecepatan atau arah sebagai respon terhadap sebuah rangsangan [1,2]. Secara umum komponen kelincahan dapat dibagi menjadi dua bagian yaitu pengambilan keputusan berdasarkan sebuah rangsangan dan kecepatan perubahan arah (CODS). Pada sub-bab ini akan dijelaskan tentang pengertian dari masing-masing komponen tersebut.

2.1.1 Perceptual and Decision Making

Perceptual and Decision Making adalah komponen kelincahan yang mengukur kemampuan seseorang dalam memproses informasi dan membuat keputusan secara cepat dan akurat. Komponen ini memiliki peran yang sangat penting dalam berbagai aktivitas atletik karena atlet harus bisa memproses informasi baik visual maupun auditif dan membuat keputusan yang tepat dengan waktu yang sangat singkat.

2.1.2 Change of Direction Speed

Change of Direction Speed (CODS) adalah elemen kelincahan yang mengukur kemampuan seseorang untuk berubah arah dengan cepat tanpa menurunkan kecepatan mereka. Dalam hal kelincahan, CODS dianggap sebagai elemen yang sangat penting dan esensial untuk sukses dalam berbagai olahraga yang membutuhkan perubahan arah tiba-tiba, seperti sepak bola dan basket. CODS ditentukan oleh beberapa faktor, termasuk kekuatan otot, kekuatan, koordinasi, keseimbangan, dan waktu reaksi. Atlet dengan tingkat CODS yang tinggi harus memiliki kontrol yang baik terhadap gerakan tubuh mereka, kemampuan untuk menghasilkan gaya secara cepat, dan kemampuan untuk memproses informasi dan membuat keputusan dengan cepat.

2.1.3 Latihan Kelincahan

Latihan kelincahan adalah sebuah kegiatan fisik yang bertujuan untuk meningkatkan keterampilan dan fleksibilitas seseorang. Latihan kelincahan memiliki banyak manfaat bagi kesehatan, seperti meningkatkan keseimbangan, daya tahan, dan kekuatan otot. Berdasarkan studi pustaka yang telah dilakukan, terdapat beberapa metode umum yang bisa dilakukan untuk melatih kelincahan, yaitu latihan pliometrik dan latihan CODS [7].

Latihan pliometrik adalah metode latihan fisik yang memanfaatkan prinsip elastisitas otot untuk meningkatkan kekuatan dan daya ledak. Ini melibatkan melakukan serangkaian gerakan yang cepat dan intens, seperti melompat, yang menyebabkan kontraksi dan relaksasi otot secara bersamaan. Proses ini memicu mekanisme otot untuk memproduksi lebih banyak tenaga dalam waktu singkat, yang pada gilirannya memperkuat otot dan membantu meningkatkan performa atletik. Studi telah menunjukkan bahwa latihan pliometrik efektif dalam meningkatkan daya ledak, kecepatan, dan agilitas, serta membantu mencegah cedera pada atlet. Beberapa contoh latihan pliometrik yang sering digunakan adalah *box jump*, *squat jump*, *tuck jump*, *depth jump*, *hurdle hop*, *skater hops* dan *single-leg hops*.

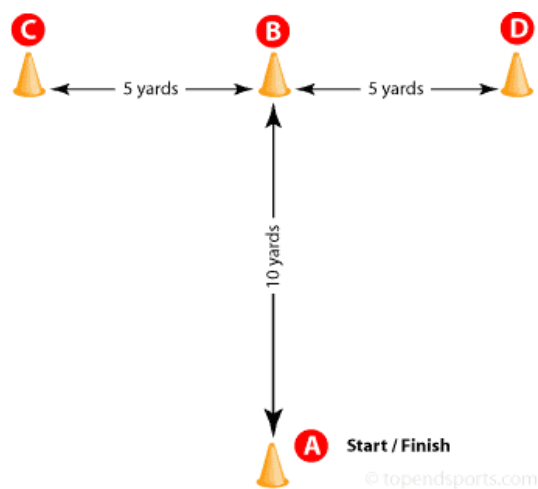
Sedangkan latihan CODS adalah jenis latihan fisik yang bertujuan untuk meningkatkan kemampuan seseorang dalam berubah arah secara cepat dan efisien. Latihan ini menekankan pada gerakan berlari dan berpindah arah, seperti lari zig zag, melakukan serpentine, dan berlari melingkar. Latihan CODS membantu meningkatkan kekuatan otot, koordinasi, dan stabilitas, serta membantu mencegah cedera. Latihan ini bermanfaat untuk atlet yang berlomba dalam olahraga seperti sepak bola, bola basket, atletik, dan lainnya yang membutuhkan kemampuan berubah arah dengan cepat dan efisien.

2.1.4 Tes Kelincahan

Tes kelincahan bertujuan untuk menilai kemampuan seseorang untuk berpindah posisi dengan cepat dan efisien, serta untuk mempertahankan keseimbangan dan berubah arah. Dalam olahraga seperti sepak bola dan basket, kecepatan dan kemampuan untuk berubah arah secara cepat sangat

penting untuk menentukan kemenangan. Oleh karena itu, tes ini sering dilakukan sebagai bagian dari seleksi atlet.

Tes kelincahan melibatkan lintasan yang ditandai dengan pagar, *pylons*, atau tanda lain dan atlet harus menjalani lintasan sesuai instruksi. Kecepatan, kualitas perpindahan posisi, dan kemampuan untuk berubah arah akan diukur menggunakan *stopwatch* atau peralatan pengukur lain. Beberapa tes kelincahan yang umum dilakukan antara lain *Illinois Agility Test*, *T-Test*, *505 Agility Test*, *Shuttle Run test*, *Side-step Test*, dan *Hexagon agility test*.



Gambar 1.2 T-Test.

T-Test merupakan salah satu jenis tes kelincahan yang umum digunakan oleh para atlet, perlengkapan yang dibutuhkan untuk melakukan tes ini adalah meteran, kerucut dan *stopwatch*, namun apabila memungkinkan *stopwatch* dapat digantikan dengan sebuah *timing gates* untuk mendapatkan pengukuran waktu yang lebih presisi. Tes ini dilakukan dengan menempatkan 4 kerucut yang diilustrasikan pada Gambar 1.2 (5 yards = 4.57 m, 10 yards = 9.14 m).

Pada awal tes, atlet harus berdiri pada kerucut A, atas aba-aba dari *timer/coach*, atlet harus berlari ke kerucut B dan menyentuh dasar kerucut dengan tangan kanannya. Lalu atlet harus bergerak menyamping menuju kerucut C dan menyentuh dasar kerucut dengan tangan kirinya, lalu bergerak menyamping ke arah kerucut D dan menyentuh alasnya dengan

tangan kanan, lalu atlet harus berbalik arah dan bergerak menyamping menuju kerucut B dan menyentuh alasnya dengan tangan kiri. Terakhir atlet harus bergerak mundur menuju tempat awal (kerucut A).

Uji coba tidak akan dihitung jika atlet menyilangkan satu kaki di depan kaki lainnya saat bergerak menyamping, gagal menyentuh dasar kerucut, atau gagal menghadap ke depan selama ujian. Ambil waktu terbaik dari tiga percobaan sukses hingga 0,1 detik terdekat. Tabel di bawah menunjukkan beberapa skor kelincahan untuk atlet olahraga beregu dewasa.

Tabel 2.1 Klasifikasi Kelincahan berdasarkan Skor T-Test

	<i>Males (seconds)</i>	<i>Females (seconds)</i>
<i>Excellent</i>	< 9.5	< 10.5
<i>Good</i>	9.5 to 10.5	10.5 to 11.5
<i>Average</i>	10.5 to 11.5	11.5 to 12.5
<i>Poor</i>	> 11.5	> 12.5

2.2 Produk Terkait

Berdasarkan studi literatur yang telah dilakukan, terdapat beberapa perusahaan yang telah memproduksi alat *agility tester* yang digunakan untuk melakukan latihan dan tes kelincahan. Dalam sub-bab ini akan dibahas beberapa merek *agility tester* yang sudah tersedia di pasaran.

2.2.1 FITLIGHT®



Gambar 2.1 FITLIGHT®

FITLIGHT® [8], merupakan sistem pemantau kebugaran yang menggunakan teknologi LED dan sensor infrared untuk membantu seseorang dalam meningkatkan kelincahan, kecepatan dan reaksi terhadap rangsangan. FITLIGHT® menggunakan sensor inframerah untuk

memantau reaksi dan aktivitas dari penggunanya, sensor ini bertugas untuk mendeteksi apakah pengguna telah menyentuh bola dengan LED yang telah diprogram dengan warna tertentu. Setelah sensor mendeteksi sentuhan, ia akan mengirimkan informasi tersebut ke kontroler, yang kemudian akan mengaktifkan LED berikutnya.

Produk ini juga dilengkapi dengan sebuah aplikasi yang bisa diakses langsung oleh penggunanya. Di dalam aplikasi ini pengguna dapat memilih lebih dari 30 jenis latihan kelincahan, melihat hasil latihan yang telah dilakukannya, serta mengkustomisasi pola dari masing-masing LED untuk membuat latihan kelincahan khusus.

Kelebihan dari produk ini adalah bentuknya yang simpel sehingga pengguna bisa melakukan latihan dimanapun dan kapanpun serta latihan kelincahan yang dapat dikostumisasi. Namun kekurangan dari produk ini adalah produk ini hanya bisa digunakan untuk latihan kelincahan saja, selain itu harga produk yang cukup tinggi, sehingga produk ini tidak dapat dijangkau masyarakat dengan ekonomi menengah keatas.

2.2.2 SmarTracks

Dilansir dari laman resminya, SmarTracks [9] merupakan merek yang menawarkan beberapa produk yang dapat digunakan untuk memantau dan menilai prestasi di bidang olahraga.



Gambar 2.2 SmarTracks

Produk pertama yang ditawarkan oleh SmarTracks adalah *Timing Gates*, produk ini merupakan sebuah gerbang magnetik yang memungkinkan pengukuran waktu yang akurat dibandingkan dengan sistem pengukuran waktu lain seperti sensor inframerah. *Timing Gates* yang ditawarkan oleh SmarTracks dilengkapi dengan produk yang kedua yaitu sensor DX5.0. Sensor DX5.0 akan merekam gerakan atlet ketika melewati

medan magnet yang terdapat pada Timing Gates SmarTracks, sensor ini akan dipakai oleh atlet pada pinggang bagian belakang dengan bantuan sebuah sabuk elastis. Produk yang ketiga yang ditawarkan berupa sebuah aplikasi *Smart Run*. Aplikasi ini mampu menampilkan data berupa pencatatan waktu latihan atlet selama latihan secara presisi, pencatatan waktu dari sensor DX5.0 akan langsung ditampilkan secara *realtime* pada *smartphone*. Kemudian produk yang terakhir adalah sebuah *diagnostics systems*. SmarTracks Diagnostics akan menganalisis data yang telah dikirimkan oleh sensor DX5.0 kemudian akan menampilkan visualisasinya kepada pelatih maupun atlet.

Kelebihan dari produk ini adalah telah menggunakan teknologi pengukuran waktu yang menggunakan metode pemantauan magnetik terkini dan mutakhir yaitu *cutting-edge magnetic timing technology*. Teknologi ini memungkinkan pengukuran waktu yang tepat dan akurat tanpa terpengaruh oleh faktor eksternal seperti cuaca dan kondisi lingkungan. Sedangkan kekurangannya adalah produk ini hanya menyediakan latihan kelincahan dalam bentuk perubahan kecepatan (CODS) tanpa diimbangi dengan latihan kelincahan dengan perubahan arah berdasarkan rangsangan (*perceptual and decision-making*) [9]. Selain itu produk dari SmarTracks dijual dengan harga yang cukup tinggi, sehingga produk ini tidak dapat dijangkau oleh segmen pasar dengan ekonomi menengah kebawah.

2.2.3 SmartSpeed

SmartSpeed merupakan merek dari *timing gates systems* yang dikembangkan oleh perusahaan VALD yang berasal dari negara Australia [10].

Produk dari perusahaan ini adalah sebuah *timing gates* yang menggunakan teknologi *single beam* yang dilengkapi dengan ECP. Teknologi *single beam* adalah sebuah teknologi pengukuran kecepatan yang menggunakan satu buah laser sebagai sumber cahaya.



Gambar 2.3 SmartSpeed

Perbedaan waktu antara saat laser ditembakkan dan saat cahaya dipantulkan kembali digunakan untuk menentukan kecepatan objek. Teknologi *single beam* biasanya lebih sederhana dan lebih ekonomis dibandingkan dengan teknologi *dual beam*, dan karena menggunakan satu sumber cahaya saja, risiko terjadinya pemicu palsu lebih rendah. Sedangkan ECP adalah sebuah proses yang digunakan untuk memperbaiki atau mengoreksi kesalahan pengukuran. Dalam hal pengukuran kecepatan, ECP dapat digunakan untuk memperbaiki hasil pengukuran yang tidak akurat atau tidak valid. Proses ini biasanya melibatkan analisis data dan perhitungan matematis untuk menentukan koreksi yang dibutuhkan dan menghasilkan hasil pengukuran yang lebih akurat dan dapat dipercaya.

Kelebihan dari produk SmartSpeed adalah produk ini sudah dilengkapi dengan RFID yang digunakan untuk mengidentifikasi siapa orang yang sedang melakukan latihan maupun tes kelincahan, selain itu *timing gates* pada produk ini sudah dilengkapi dengan dua latihan kelincahan dasar yaitu latihan CODS dan latihan kelincahan dengan perubahan arah berdasarkan rangsangan (*perceptual and decision-making*). Sedangkan kekurangan dari produk ini adalah harganya yang relatif tinggi dan mungkin tidak dapat dijangkau oleh segmen pasar dengan ekonomi menengah kebawah.

2.3 Perangkat Pembangun Sistem

Alat *agility tester* berbasis IoT ini terdiri dari 4 komponen utama yang saling berkaitan yaitu node, CB, *cloud database*, dan *web server*. Dalam sub-bab ini akan dibahas mengenai masing-masing komponen tersebut:

2.3.1 Node

Dalam rangkaian alat *agility tester* ini, terdapat lima buah node yang dilengkapi dengan sensor VL53L0X yang berfungsi untuk memantau reaksi dan aktivitas pengguna. Sensor ini berfungsi untuk mendeteksi apakah pengguna telah menyentuh node dengan LED yang telah diprogram untuk menyala atau belum. Setelah sensor mendeteksi interaksi dari pengguna, Node akan mengirimkan data berupa *timestamp* ke CB melalui perangkat nRF24L01.

2.3.1.1 Sensor VL53L0X

Berdasarkan data yang diperoleh dari *datasheet* [12], Sensor VL53L0X adalah modul pengukuran jarak berbasis *Time-of-Flight* (ToF) dengan ukuran kecil (4.4 x 2.4 x 1.0mm). Sensor ini dapat mengukur jarak absolut hingga 2 meter tanpa dipengaruhi oleh reflektansi target, beroperasi dalam kondisi cahaya inframerah tinggi, dan memiliki kompensasi optik silang terintegrasi. Sensor ini menggunakan laser Kelas 1 yang aman untuk mata manusia. Sensor ini mudah diintegrasikan dengan antarmuka I2C, tidak memerlukan optik tambahan, dan memiliki fitur Xshutdown (Reset) dan interrupt GPIO. Sensor VL53L0X menawarkan pengukuran jarak yang cepat, akurat, dan dapat diandalkan, serta menjadi solusi menarik untuk berbagai aplikasi di berbagai industri.



Gambar 2.4 Sensor VL53L0X

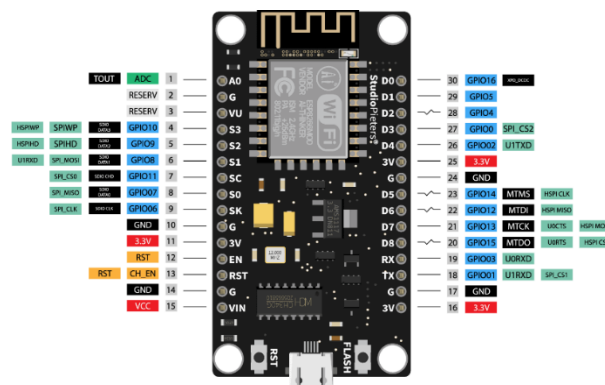
Berdasarkan data dari *datasheet* [12], berikut ini merupakan spesifikasi dari Sensor VL53L0X:

Tabel 2.2 Detail dari spesifikasi dan fitur yang dimiliki sensor VL53L0X

Model	GY-VL53L0X
Prinsip Pengukuran	<i>Time-of-Flight (ToF)</i>
Tegangan Kerja	2.8 – 5V
Suhu Kerja	-20°C – 70°C
Rentang Pengukuran	30 mm – 2000 mm
Akurasi	±3%
Tipe Interface	I2C, UART

2.3.1.2 NodeMCU ESP8266

Berdasarkan data yang diperoleh dari *datasheet* [13], NodeMCU adalah suatu *firmware* dan papan pengembangan yang menggunakan bahasa pemrograman Lua dan bersifat *open-source*. Papan pengembangan NodeMCU ESP8266 dilengkapi dengan modul ESP-12E yang berisi chip ESP8266, yang memiliki mikroprosesor Tensilica Xtensa LX106 RISC 32-bit. Mikroprosesor ini mendukung RTOS dan dapat beroperasi pada frekuensi *clock* yang dapat disesuaikan, antara 80 MHz hingga 160 MHz. NodeMCU memiliki kapasitas RAM sebesar 128 KB dan memori *flash* sebesar 4 MB untuk menyimpan data dan program. Modul ini juga dilengkapi dengan fitur Wi-Fi, yang memungkinkan perangkat untuk melakukan pengiriman dan penerimaan data secara nirkabel melalui spektrum frekuensi 2,4 GHz. Selain itu, modul ini memiliki fungsi *deep sleep operating internal* yang membuatnya sangat cocok untuk digunakan dalam proyek-proyek berbasis IoT.



Gambar 2.5 NodeMCU ESP8266

Berdasarkan data dari *datasheet* [13], berikut ini merupakan spesifikasi dari modul NodeMCU ESP8266:

Tabel 2.3 Detail dari spesifikasi dan fitur yang dimiliki NodeMCU ESP8266.

Mikroprosesor	Tensilica 32-bit RISC CPU Xtensa LX106
Tegangan Operasi	3.3V
Tegangan Masukan	7 – 12V
Digital I/O Pins (DIO)	16
Pin Input Analog (ADC)	1
UARTs	1
SPIs	1
I2Cs	1
Memori Flash	4 MB
SRAM	64 KB
<i>Clock Speed</i>	80 MHz

2.3.1.3 nRF24L01

nRF24L01 merupakan chip transmisi nirkabel yang dikembangkan oleh Nordic Semiconductor. Chip ini memiliki beberapa fitur menonjol seperti daya rendah, jarak jangkauan yang baik, kecepatan transmisi tinggi, dan dukungan untuk beberapa protokol transmisi nirkabel seperti Zigbee, *Bluetooth Low Energy* (BLE), dan 6LoWPAN. nRF24 memiliki beberapa varian yang berbeda dalam hal kinerja dan spesifikasi, seperti nRF24L01, nRF24L01+, nRF24L01P, nRF24LE1, dan nRF24LU1+. Masing-masing varian memiliki spesifikasi unik dan fitur yang berbeda. Chip ini dapat bekerja pada frekuensi 2.4 GHz dan memiliki enam kanal transmisi yang berbeda. Ini juga memiliki fitur *auto-acknowledgment* dan pengendalian energi yang efisien.

nRF24L01 memiliki aplikasi yang luas dalam berbagai bidang seperti *Internet of Things* (IoT), kendaraan tanpa pengemudi, sistem pemantauan, dan banyak lagi. Chip ini juga dapat digunakan untuk aplikasi tanpa kabel seperti pengendalian jarak jauh, komunikasi antar modul, dan banyak lagi. Secara umum, nRF24L01 adalah chip transmisi

nirkabel yang sangat efisien dan fleksibel yang memiliki aplikasi luas dalam berbagai bidang. Ini juga sangat populer di kalangan pengembang perangkat lunak dan *hardware* karena mudah digunakan dan didukung oleh banyak bibliotek dan perangkat lunak.



Gambar 2.6 nRF24L01

Berdasarkan data dari *datasheet* [14], berikut ini merupakan spesifikasi dari modul nRF24L01:

Tabel 2.4 Detail dari spesifikasi dan fitur yang dimiliki nRF24L01

Frekuensi Operasi	2.4GHz ISM band
Kecepatan Data	1 Mbps atau 2 Mbps (Dikonfigurasi oleh Mikrokontroler (MCU)).
Jumlah Kanal	127 Kanal (25 MHz <i>spacing</i>)
Daya Transmisi	-18 dBm to +18 dBm
Jarak Transmisi	Hingga 100 Meter (Tergantung pada Kondisi Lingkungan)
Modulasi	GFSK (<i>Gaussian Frequency Shift Keying</i>)
Konsumsi Daya	45 mA (TX), 40 mA (RX), dan 1.9 mA (<i>sleep mode</i>)
<i>Baud rate</i>	<i>Programmable Baud rate</i>
Tingkat keamanan	AES-128 <i>Software Encryption</i>
Antena	Internal atau Eksternal
Ukuran	18 x 45 mm

Tegangan Kerja	2.7V to 3.6V
Tipe Interface	SPI, UART (<i>optional</i>)

2.3.1.4 WS2812 *Builitn Driver Neopixel*

WS2812 merupakan sebuah modul LED yang mengintegrasikan *control circuit* dan RGB kedalam LED chip yang berukuran 5.0 mm x 5.0 mm, didalam chip ini terdapat port data pintar digital, penahan data (*latch*), dan rangkaian penguat pengubahan sinyal. Selain itu, juga terdapat osilator internal yang presisi dan bagian pengendali arus konstan yang dapat diprogram dengan tegangan 12V, yang secara efektif memastikan konsistensi warna cahaya titik piksel.

Protokol transfer data pada chip ini menggunakan mode komunikasi *single NZR*. Setelah daya dihidupkan, *port* DIN akan menerima data dari *controller*. Piksel pertama mengumpulkan data awal sebesar 24 bit dan kemudian mengirimkannya ke pengunci data internal. Data lainnya yang telah diubah bentuk oleh rangkaian penguat sinyal internal dikirimkan ke piksel kaskade berikutnya melalui port DO. Setelah transmisi untuk setiap piksel, sinyal 24 bit dikurangi untuk mengadopsi teknologi transmisi ulang secara otomatis, sehingga jumlah piksel kaskade tidak terbatas pada transmisi sinyal, hanya bergantung pada kecepatan transmisi sinyal. LED ini memiliki tegangan penggerak rendah, ramah lingkungan, hemat energi, kecerahan tinggi, sudut penyebaran yang besar, konsistensi yang baik, daya rendah, dan umur panjang. Chip kontrol yang terintegrasi di atas LED membuat sirkuit lebih sederhana, volume lebih kecil, dan pemasangan lebih mudah.



Gambar 2.7 WS2812 *Builitn Driver Neopixel*

Berdasarkan data dari *datasheet* [15], berikut ini merupakan spesifikasi dari WS2812 *Built-in Driver Neopixel*:

Tabel 2.5 Detail dari WS2812 *Built-in Driver Neopixel*

LED Chip	WS1228B
Jumlah Led Pixel	12
Tegangan Kerja	4 – 7V (DC)
<i>Communication Interface</i>	<i>Single-wire Communication</i>
Kecepatan Pengiriman Data	800Kbps
Pixel	Setiap piksel terdiri dari tiga warna primer, dapat mencapai 256 tingkat tampilan kecerahan, menampilkan 16.777.216 warna yang penuh warna, dan frekuensi pemindaian tidak kurang dari 400 Hz/s.
Diameter Ring	50 mm

2.3.1.5 *Buzzer* SFM-20B

Buzzer SFM-20B merupakan salah satu jenis *buzzer* piezoelektrik yang sering digunakan dalam berbagai aplikasi elektronik. *Buzzer* ini terbuat dari bahan keramik piezoelektrik yang dapat menghasilkan getaran mekanis saat dikenai tegangan listrik. Getaran ini kemudian dikonversi menjadi gelombang suara oleh elemen piezoelektrik, menciptakan suara yang didengar sebagai bunyi *buzzer*. Secara teknis, *Buzzer* SFM-20B biasanya memiliki dimensi yang kecil, umumnya dalam bentuk silinder atau cakram datar. Dimensi kecil ini memungkinkan pemasangan yang mudah pada papan sirkuit cetak (PCB) atau perangkat lainnya. *Buzzer* ini juga ringan, sehingga cocok untuk digunakan pada perangkat portabel atau berdaya rendah.

Buzzer SFM-20B memiliki berbagai spesifikasi dan karakteristik yang relevan. Salah satunya adalah frekuensi resonansi, yang merupakan frekuensi alami dari elemen piezoelektrik dan menentukan frekuensi suara yang dihasilkan oleh *buzzer* ketika diberi tegangan. Intensitas suara yang

dihasilkan juga merupakan parameter penting dan diukur dalam desibel (dB). Daya yang dikonsumsi oleh *buzzer* ini biasanya rendah, sehingga efisien dalam penggunaan daya. Penggunaan Buzzer SFM-20B sangat beragam, mulai dari perangkat keamanan, alarm, perangkat medis, perangkat rumah tangga, perangkat telekomunikasi, hingga berbagai sistem peringatan dan kendali dalam industri.



Gambar 2.8 Buzzer SFM-20B

Berdasarkan data dari datasheet [16], berikut ini merupakan spesifikasi dari *Buzzer SFM-20-B*:

Tabel 2.6 Detail dari spesifikasi dan fitur yang dimiliki *buzzer SFM-28-B*

Tipe	SFM-20-B
Tipe Output	<i>Continuous</i>
Tegangan Operasi	3~24 V
Keluaran Suara	≥ 95 dB
Arus Tertinggi	≤ 12 mA
Frekuensi resonansi	3900 ± 500 Hz
Suhu Operasional	(-20~+45) °C
Suhu Penyimpanan	(-20~+60) °C

2.3.2 CB

Dalam rangkaian alat *agility tester* ini, terdapat sebuah CB yang dilengkapi dengan RTC yang berfungsi untuk mendapatkan *timestamp* yang presisi ketika node mengirim data, RF24L01 yang berfungsi untuk mengirim dan menerima data dari node, NodeMCU yang berfungsi untuk menghubungkan CB dengan *web server* dan *cloud database*, OLED yang berfungsi untuk menampilkan informasi tambahan kepada pengguna, serta sebuah *Buzzer*.

2.3.2.1 RTC DS1302

RTC DS1302 merupakan sebuah chip *timekeeping* dengan *trickle charge* yang berisi *real-time clock/calendar* serta memiliki 31 byte RAM statis. DS1302 berkomunikasi dengan mikroprosesor melalui antarmuka serial sederhana. DS1302 menyediakan informasi tentang detik, menit, jam, hari, tanggal, bulan, dan tahun. Informasi tanggal akhir bulan secara otomatis disesuaikan untuk bulan-bulan dengan kurang dari 31 hari, termasuk koreksi untuk tahun kabisat. Jam ini dapat beroperasi dalam format 24 jam atau 12 jam dengan indikator AM/PM.

DS1302 memiliki dua versi, yaitu 8-Pin DIP (*Dual In-line Package*) dan 8-Pin SOIC (*Small Outline Integrated Circuit*). Pengoperasian RTC DS1302 dengan mikroprosesor dipermudah dengan menggunakan komunikasi serial sinkron. Hanya tiga kabel yang diperlukan untuk berkomunikasi dengan *clock*/RAM ini, yaitu (1) RST (*Reset*), (2) I/O (*Data line*), dan (3) SCLK (*Serial clock*). Data dapat ditransfer ke dan dari *clock*/RAM satu *byte* pada satu waktu atau dalam sekali transfer sebanyak 31 *byte*. DS1302 dirancang untuk beroperasi dengan konsumsi daya yang sangat rendah dan dapat mempertahankan data dan informasi jam dengan daya kurang dari 1 mikrowatt.



Gambar 2.9 RTC DS1302

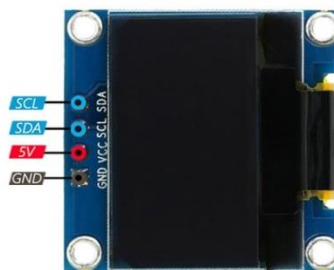
Berdasarkan data dari datasheet [16], berikut ini merupakan spesifikasi dari RTC DS1302:

Tabel 2.7 Detail dari spesifikasi dan fitur yang dimiliki RTC DS1302

Tipe	DS1302
Tegangan Kerja	3.3 – 5V (DC)
Suhu Kerja	0 – 70°C
Komunikasi	3 atau 4 kabel <i>Synchronous Serial (SPI) Interface</i>
Dimensi	31.2 x 15.5 mm

2.3.2.2 OLED 0.96"

OLED merupakan jenis layar mikro berbasis *Organic Light Emitting Diode* dengan ukuran diagonal 0,96 inch (2,4 cm). Teknologi OLED menggunakan lapisan organik yang mampu mengemisikan cahaya saat dialiri arus listrik, tanpa memerlukan sumber penerangan latar belakang, seperti pada layar LCD konvensional. Struktur OLED terdiri dari beberapa lapisan esensial, termasuk lapisan anoda dan katoda, serta lapisan organik antara keduanya. Ketika arus mengalir melalui lapisan organik, elektron bertemu dengan lubang di dalamnya, menghasilkan pasangan elektron-hole yang berada dalam keadaan ter *excite*. Saat elektron ini kembali ke keadaan dasarnya, energi yang dilepaskan dalam bentuk cahaya tampak.



Gambar 2.10 OLED 0.96"

Berdasarkan data dari *datasheet* [17], berikut ini merupakan spesifikasi dari OLED 0.96":

Tabel 2.8 Detail dari spesifikasi dan fitur yang dimiliki OLED 0.96"

Ukuran Layar	0.96 inch (24 mm)
Resolusi Layar	128 x 32 pixel
Ukuran Board	28 x 28 mm
Sudut Pandang	>160°
Tegangan Input	3.3 ~ 6V
Suhu Kerja	-30°C – 80°C

2.3.3 Cloud Database

Cloud database merupakan sebuah sistem *database* yang berjalan pada infrastruktur *cloud computing*. Hal ini memungkinkan penyimpanan

data dan akses data dari jarak jauh melalui internet. Keuntungan utama dari *cloud database* adalah fleksibilitas, skalabilitas, dan ketersediaan tinggi, sehingga memungkinkan organisasi untuk menyesuaikan jumlah penyimpanan dan pemrosesan yang mereka butuhkan tanpa harus membeli dan mengelola perangkat keras atau perangkat lunak sendiri. Ada dua jenis utama *cloud database*, yaitu *database* yang dikelola oleh pihak ketiga (DBaaS) dan *database* yang dikelola oleh pengguna (IaaS). Keamanan *cloud database* sangat penting bagi banyak organisasi, sehingga beberapa vendor *cloud database* menawarkan enkripsi data dan kerangka kerja keamanan yang kuat. Performa *cloud database* dapat ditingkatkan melalui pemuatan *balancer*, penyesuaian jumlah sumber daya, dan penyesuaian konfigurasi sistem. Studi kasus juga telah menunjukkan bahwa organisasi dapat menghemat biaya dan meningkatkan efisiensi dengan menggunakan *cloud database*. Oleh karena itu, *cloud database* memberikan solusi efisien dan fleksibel bagi organisasi yang membutuhkan penyimpanan dan pemrosesan data yang handal, namun harus mempertimbangkan keamanan dan performa *cloud database* saat membuat keputusan untuk migrasi ke *cloud*.

2.3.4 Web Application

Web application adalah program perangkat lunak yang dirancang untuk diakses dan digunakan oleh pengguna melalui browser web. Berbeda dengan halaman web statis yang hanya menyajikan konten tetap, *web application* memiliki interaktivitas yang lebih tinggi, memungkinkan pengguna untuk berinteraksi dengan data, mengirimkan formulir, berpartisipasi dalam proses, dan mengakses berbagai fungsionalitas yang ditawarkan oleh aplikasi tersebut. Berikut adalah beberapa poin penting tentang *web application*:

1. **Interaktivitas Tinggi**, *web application* dirancang untuk berinteraksi dengan pengguna. Mereka dapat merespons tindakan pengguna secara dinamis, seperti mengambil data dari basis data, menghitung hasil, dan menyajikan informasi yang disesuaikan dengan preferensi atau tindakan pengguna.

2. **Kompleksitas Lebih Tinggi**, *web application* umumnya lebih kompleks daripada halaman web statis. Mereka mungkin melibatkan logika bisnis, pemrosesan data, otentikasi pengguna, dan bahkan integrasi dengan layanan pihak ketiga.
3. **Arsitektur Umum**, *web application* umumnya beroperasi menggunakan model arsitektur *client-server*. Browser pengguna bertindak sebagai klien yang mengirim permintaan HTTP ke server. Server web menerima permintaan tersebut dan menyediakan *respons*, yang bisa berupa data mentah atau HTML yang dihasilkan
4. **Bahasa Pemrograman**, *web application* bisa ditulis dalam berbagai bahasa pemrograman, seperti JavaScript, Python, Ruby, Java, dan lainnya. Bahasa pemrograman yang dipilih tergantung pada kebutuhan aplikasi, keahlian pengembang, dan teknologi yang digunakan.
5. **Framework dan Library**, untuk mempermudah pengembangan *web application*, ada berbagai *framework* dan *library* yang tersedia. *Framework* seperti Ruby on Rails, Django, Laravel, dan Angular menyediakan struktur dan alat untuk membangun aplikasi dengan lebih efisien.
6. **Penggunaan Basis Data**, banyak *web application* bergantung pada basis data untuk menyimpan dan mengelola informasi. Basis data dapat digunakan untuk menyimpan profil pengguna, konten dinamis, data transaksi, dan lainnya.
7. **Hosting dan Deployment**, setelah *web application* selesai dikembangkan, mereka harus di-*host* di *server* web agar dapat diakses oleh pengguna. Pengguna dapat mengakses aplikasi ini melalui URL yang sesuai.

Web application memberikan pengalaman berinteraksi yang dinamis dan personal bagi pengguna. Mereka telah mengubah cara kita berinteraksi dengan layanan dan informasi di web, memungkinkan akses ke fungsionalitas yang lebih kompleks dan interaktif melalui browser web.

BAB 3 METODE PENELITIAN

3.1 Waktu dan Tempat Pengerjaan

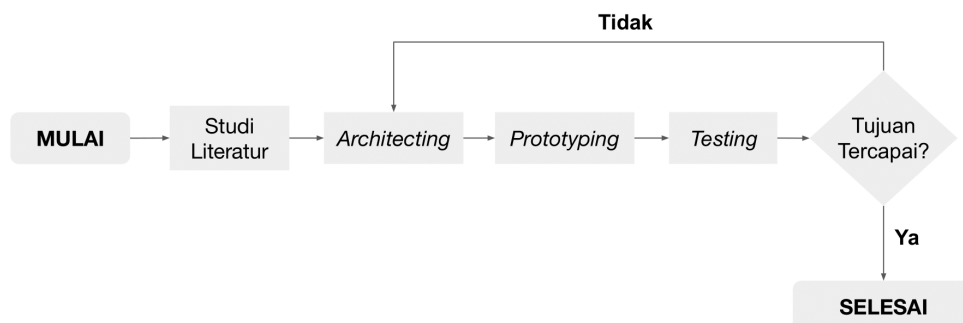
Adapun waktu dan tempat untuk pengerjaan tugas akhir ini dirincikan pada tabel berikut:

Tabel 3.1 Waktu dan Tempat Pelaksanaan Tugas Akhir

Waktu Pelaksanaan	Tempat Pelaksanaan	Keterangan
Maret 2023 – Juli 2023	Gedung Collaborative STEM Laboratory (CSL) Universitas Prasetiya Mulya.	Pembuatan Prototipe
Juli 2023 – Agustus 2023	Perumahan The Icon, BSD	Pengujian

3.2 Metodologi

Pada pelaksanaan tugas akhir ini dilakukan perancangan arsitektur dari sistem *agility tester* berbasis IoT yang dapat melakukan pengukuran waktu secara presisi pada saat melakukan latihan dan tes kelincahan, menyimpan data tersebut kedalam *cloud database* dan menampilkannya ke dalam sebuah *web application*. *Use-case* dari produk ini adalah untuk melatih serta mengevaluasi tingkat kelincahan seseorang individu, baik atlet maupun non-atlet.



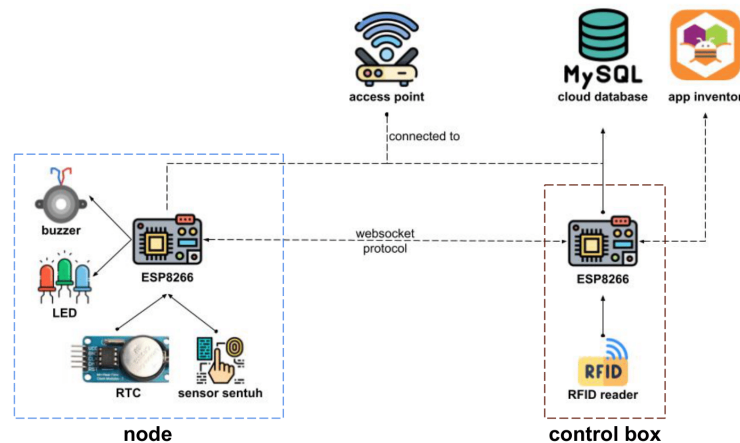
Gambar 3.1 Diagram Alir Penelitian

Adapun diagram alir dari metodologi penelitian ini dimulai dari proses studi literatur mengenai konsep kerja dari produk sebelumnya untuk mengetahui potensi pengembangan lanjutan dari produk tersebut. Setelah peneliti memahami tentang teori dan konsep kerja dari sistem *agility tester* dari penelitian terdahulu, maka peneliti akan melakukan perancangan arsitektur, desain dan prototipe dari sistem *agility tester* berbasis IoT berdasarkan kebutuhan *user*. Hingga pada akhirnya, peneliti akan melakukan proses evaluasi untuk menilai kinerja sistem *agility tester* berbasis IoT yang telah diusulkan.

3.3 Perancangan Prototipe

Berdasarkan hasil studi literatur yang telah dilakukan dalam Bab II, peneliti menyadari bahwa beberapa komponen diperlukan untuk membangun sistem *Agility Tester* berbasis IoT. Komponen-komponen ini mencakup node, CB, *database*, dan sebuah aplikasi pengendalian. Setelah menentukan komponen-komponen yang diperlukan, peneliti selanjutnya merancang arsitektur yang sesuai dengan komponen-komponen tersebut. Dalam hal ini, peneliti merancang dua skema arsitektur yang dijelaskan dalam sub-bab 3.3.1 dan 3.3.2.

3.3.1 Arsitektur Sistem Versi 1

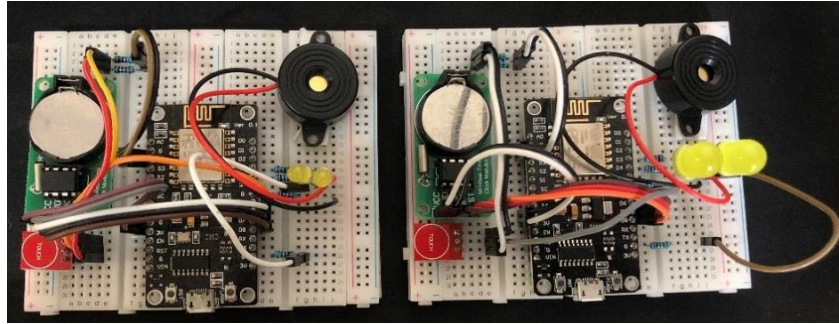


Gambar 3.2 Arsitektur Sistem *Agility Tester* berbasis IoT: Versi 1

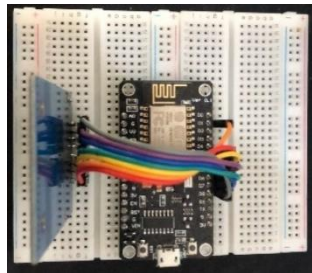
Pada arsitektur sistem pertama, sistem *agility tester* terdiri dari node, CB, *cloud database*, dan MIT *App Inventor*. Namun, setelah menjalani serangkaian pengujian sistem, ditemukan bahwa arsitektur ini kurang efisien

untuk digunakan. Oleh karena itu, pada bab ini akan dijelaskan tentang kekurangan-kekurangan yang terdapat dalam arsitektur sistem versi 1.

3.3.1.1 Node dan CB



Gambar 3.3 Rangkaian Node Versi 1



Gambar 3.4 Rangkaian CB Versi 1

Pada tahap awal perancangan sistem, sensor yang digunakan untuk mendeteksi pergerakan atlet adalah sensor sentuh kapasitif TTP223. Sensor ini beroperasi dengan mendeteksi perubahan kapasitansi antara sensor dan objek yang mendekatinya. Namun, setelah melalui pengujian pengalaman pengguna dalam penggunaan node, terungkap bahwa sensor ini dianggap terlalu kecil oleh para pengguna. Ukurannya membuat sensor sulit dijangkau selama proses latihan dan uji kelincahan. Untuk mengatasi hal tersebut, sensor sentuh TTP223 diganti menggunakan sensor VL53L0X pada perancangan arsitektur yang kedua.

Masing-masing node dilengkapi dengan RTC untuk melakukan pengukuran waktu yang presisi, setiap RTC dikalibrasi dengan menggunakan waktu terkini dari API yang disediakan oleh situs “pool.ntp.org”. Namun, meskipun sudah melakukan tahap kalibrasi, nilai pencatatan waktu pada masing-masing node memiliki selisih antara 100

hingga 500 ms. Untuk mengatasi permasalahan tersebut, dalam perancangan arsitektur yang kedua digunakan pencatatan waktu secara terpusat pada CB.

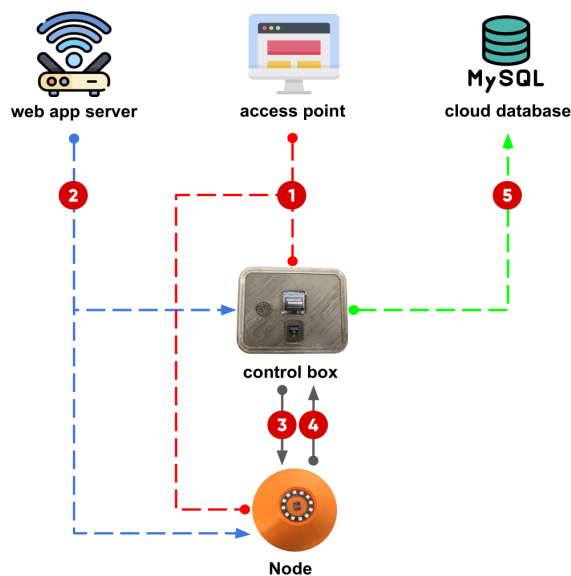
Metode komunikasi yang digunakan antara node dan CB melibatkan komunikasi *client-server* dengan menggunakan protokol WebSocket. Dalam metode ini, terdapat keterlambatan sekitar 500 hingga 1000 ms saat mengirim perintah dari CB ke node. Keterlambatan dalam rentang tersebut tidak dapat ditoleransi oleh pengguna. Oleh karena itu, pada arsitektur yang kedua, komunikasi antara CB dan node diganti dengan menggunakan modul nRF24L01.

3.3.1.2 MIT App Inventor

Setelah melaksanakan telaah pustaka mengenai *platform* MIT App Inventor, diketahui bahwa *platform* ini memiliki sejumlah keterbatasan ketika diaplikasikan dalam kerangka kerja *agility tester*. Lebih tepatnya, *platform* ini hanya mampu mengirimkan satu perintah melalui satu tombol, sementara sistem *agility tester* membutuhkan sebuah konfigurasi yang terdiri dari beberapa variabel yang saling terhubung dan tersimpan di dalam sebuah *database*. Untuk merespons kebutuhan tersebut, peneliti memutuskan untuk mengembangkan sebuah *web application* sebagai pengganti dari *platform* MIT App Inventor.

3.3.2 Arsitektur Sistem: Versi 2

Setelah studi literatur mengenai konsep kerja produk terkait dan pengujian pada arsitektur pertama, akhirnya peneliti mulai merancang sebuah sistem arsitektur untuk mendukung aspek fungsionalitas dari produk *agility tester* berbasis IoT yang ditawarkan dan memperbaiki arsitektur sebelumnya. Berikut ini adalah gambaran umum dari arsitektur dari sistem yang telah dirancang.



Gambar 3.5 Arsitektur Sistem *Agility Tester* berbasis IoT: Versi 2

Arsitektur ini terdiri atas CB, node, *web application server* dan *cloud database*. Adapun mekanisme kerja dari arsitektur ini adalah sebagai berikut: CB dan node terhubung pada sebuah *access point* khusus yang disediakan oleh pengguna (Wifi atau MiFi). Setelah terhubung ke koneksi internet, CB akan *merequest* data *real time clock* melalui API yang disediakan oleh situs "pool.ntp.org" (1).

Setelah terhubung ke koneksi internet, CB akan mengirimkan permintaan data konfigurasi ke sebuah *web application server*. Tujuan dari permintaan ini adalah untuk memperoleh data konfigurasi yang diperlukan guna mengatur dan mengoptimalkan fungsi-fungsi pada CB sesuai dengan kebutuhan pengguna. *Web application server* berfungsi sebagai sumber informasi yang menyediakan data konfigurasi dan mengirimkannya ke CB, memastikan bahwa CB dapat beroperasi dengan tepat dan sesuai dengan parameter yang telah ditetapkan (2). Setelah mendapatkan konfigurasi dari *server*, CB akan mengirimkan perintah kepada node menggunakan protokol radio (3).

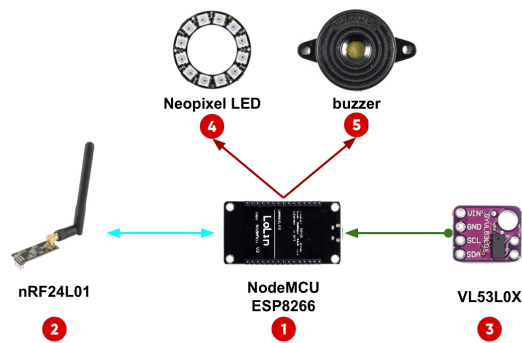
Node yang menerima perintah dari *server* akan menyalakan modul NeoPixel. Setelah itu, apabila *user/atlet* telah melambaikan tangan di atas node yang menyala, maka node akan mengirimkan *feedback* kepada CB. CB akan mencatat *timestamp* saat node memberikan *feedback*. Data tersebut

akan dicatat ke dalam sebuah *array struct*. Proses komunikasi antara node dan CB akan terus berlangsung hingga seluruh repetisi selesai dijalankan (4). Setelah repetisi selesai, CB akan mengunggah data tersebut ke dalam *cloud database* (5).

3.3.3 Desain Node

3.3.3.1 Arsitektur Node

Dalam sistem ini, node memiliki beberapa aspek fungsional yang harus terpenuhi, yaitu terhubung dengan internet, kemampuan untuk berkomunikasi dengan *web application server* dan CB, serta kemampuan untuk berinteraksi dengan pengguna. Untuk mencapai aspek fungsionalitas yang disebutkan, penulis merancang sebuah desain arsitektur node sebagai berikut.



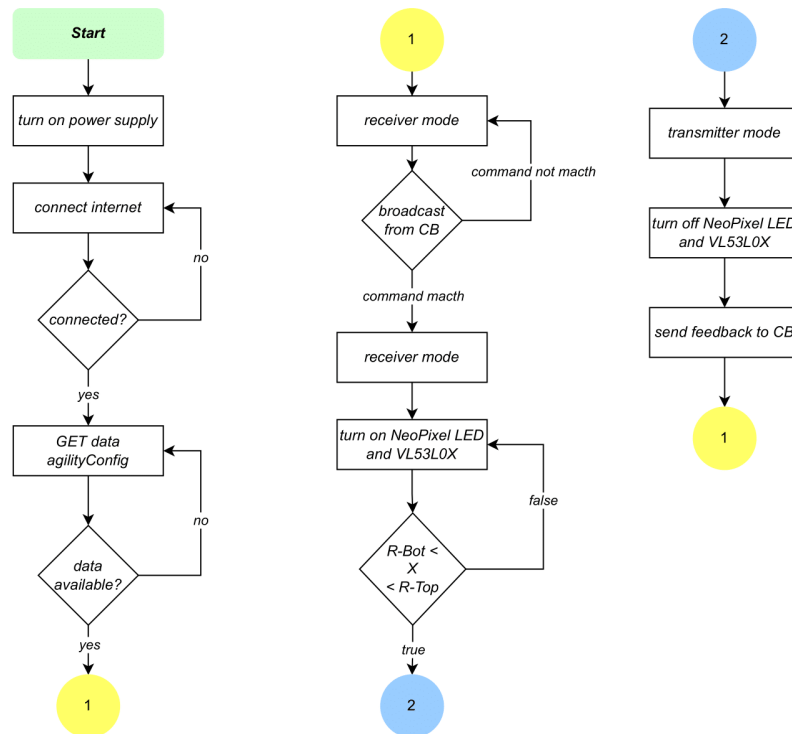
Gambar 3.6 Arsitektur Node

Pada arsitektur ini, node menggunakan mikroprosesor NodeMCU (1) yang telah dilengkapi dengan fitur Wi-Fi. Hal ini memungkinkan perangkat untuk melakukan pengiriman dan penerimaan data secara nirkabel melalui spektrum frekuensi 2,4 GHz. Dengan demikian, node dapat berkomunikasi dengan *web server* untuk mendapatkan data konfigurasi untuk sensor VL53L0X (3). Selain itu, node juga dilengkapi dengan modul nRF24L01 (2). Modul ini berfungsi untuk mengirim dan menerima data dari CB. Sehingga, node bisa mengetahui kapan waktu node aktif dan memberikan *feedback* kepada CB ketika node sudah mendeteksi lambaian tangan dari pengguna.

Ketika CB mengirimkan perintah sesuai dengan ID pada node, maka node akan menghidupkan sensor VL53L0X (3), NeoPixel LED (4) dan *buzzer* (5), sensor VL53L0X digunakan untuk mendeteksi apakah pengguna sudah melambaikan tangan diatas node yang sedang aktif atau belum.

3.3.3.2 Diagram Alir Node

Untuk mendukung arsitektur node yang telah dibangun, penulis mulai merancang diagram alir untuk menggambarkan cara kerja node secara keseluruhan. Berikut ini adalah diagram alir dari node yang telah dirancang.



Gambar 3.7 Diagram Alir Node

Pada saat *switch* berubah *state* menjadi on, aliran listrik dari *power supply* akan menghidupkan NodeMCU yang berada di dalam node. Selanjutnya, node akan terhubung ke koneksi internet. Setelah berhasil terhubung dengan koneksi internet, node akan meminta data *agilityConfig* ke *database* melalui API yang disediakan oleh *web server*. Jika data yang diminta tidak tersedia di *database*, node akan terus-menerus meminta data

tersebut ke server hingga data tersedia. Data *agilityConfig* yang diminta oleh node berupa variabel batas atas dan batas bawah pembacaan node, yang pada *database* diberi label **range_node_bottom_mm** dan **range_node_top_mm**.

Setelah node mendapatkan data konfigurasi dari *server*, nRF24L01 akan diatur ke mode *receiver*. Dalam mode ini, node hanya akan mendengarkan *broadcast* yang dikirim oleh *transmitter*. Node akan terus-menerus mendengarkan *broadcast* dari CB (*transmitter mode*) sampai *broadcast* mengirim perintah sesuai dengan ID *node* pada *node* tersebut. Setelah mendapatkan perintah yang sesuai, nRF24L01 akan diubah ke mode *transmitter*. Dalam mode ini, *node* hanya dapat mengirim pesan (*broadcast*) ke CB (*receiver mode*). Setelah itu, NodeMCU akan menghidupkan NeoPixel LED dan sensor VL53L0X. NeoPixel yang menyala menandakan bahwa node sedang aktif dan pengguna harus segera berlari ke arah *node* untuk menonaktifkannya dengan cara melambaikan tangan di atas sensor VL53L0X.

Sensor VL53L0X akan mendeteksi lambaian tangan pengguna dan memberikan output berupa jarak antara node ke tangan pengguna. Jika output dari sensor VL53L0X berada di dalam rentang yang ditentukan ($\text{range_node_bottom_mm} < \text{outputVL53L0X} < \text{range_node_top_mm}$), maka NodeMCU akan mematikan NeoPixel LED dan sensor VL53L0X. Selanjutnya, node akan mengirimkan *feedback* kepada CB dalam mode penerimaan (*receiver mode*) yang berisi informasi *nodeID*. Setelah itu, node akan melakukan *looping* proses dari blok 1 dan 2 hingga repetisi dari CB selesai.

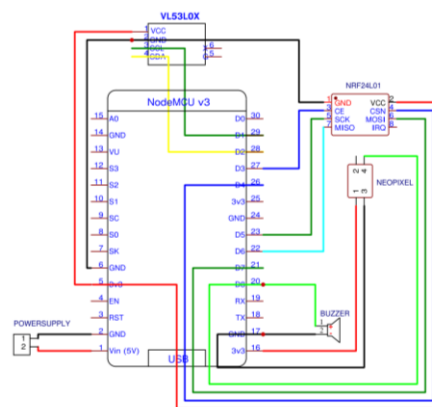
3.3.3.3 Wiring Chart

Setelah selesai membangun arsitektur dan diagram alir dari node, peneliti mulai membuat *wiring chart* dan desain PCB dari node. *Wiring chart* biasanya dibuat menggunakan *platform* fritzing, namun dikarenakan *platform* ini sekarang sudah mulai berbayar, maka penulis akhirnya membuat *wiring chart* dan desain PCB menggunakan satu *platform* yang sama yaitu EasyEDA.

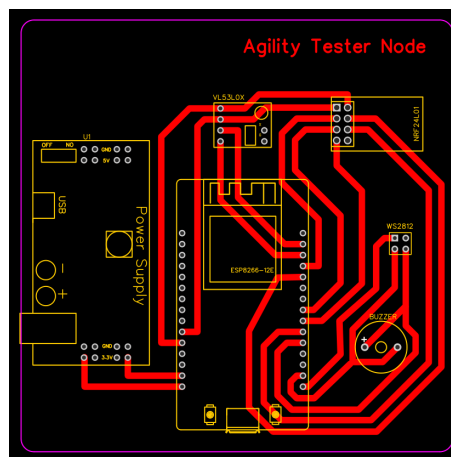
nRF24L01	NodeMCU	VL53L0X	NodeMCU	NeoPixel	NodeMCU
VCC	VCC	VCC	VCC	VCC	VCC
GND	GND	GND	GND	GND	GND
CE	D3 (GPIO00)	SCL	D1 (GPIO05)	DI	D8 (GPIO15)
CSN	D4 (GPIO02)	SDA	D2 (GPIO04)	DO	-
SCK	D5 (GPIO14)				
MOSI	D7 (GPIO13)				
MISO	D6 (GPIO12)				
IRQ	-				

Buzzer	NodeMCU	Power Supply	NodeMCU
VCC	D8 (GPIO15)	VCC	Vin (5V)
GND	GND	GND	GND

Gambar 3.8 Tabel Wiring Chart Node



Gambar 3.9 Wiring Chart Node

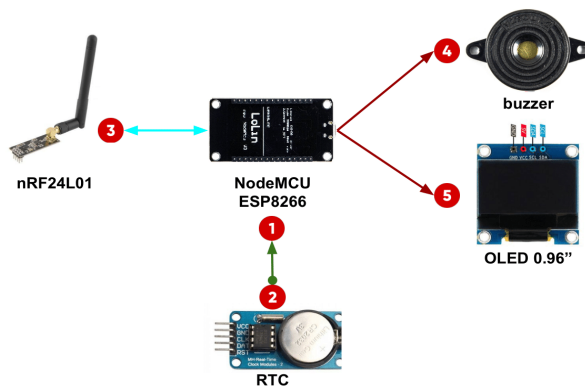


Gambar 3.8 Desain PCB dari Node

3.3.4 Desain CB

3.3.4.1 Arsitektur CB

Dalam sistem ini, CB memiliki beberapa aspek fungsional yang harus terpenuhi, yaitu terhubung dengan internet, kemampuan untuk berkomunikasi dengan node, *web server* dan *database*, serta kemampuan untuk berinteraksi dengan pengguna. Untuk mencapai aspek fungsionalitas yang disebutkan, penulis merancang sebuah desain arsitektur CB sebagai berikut.

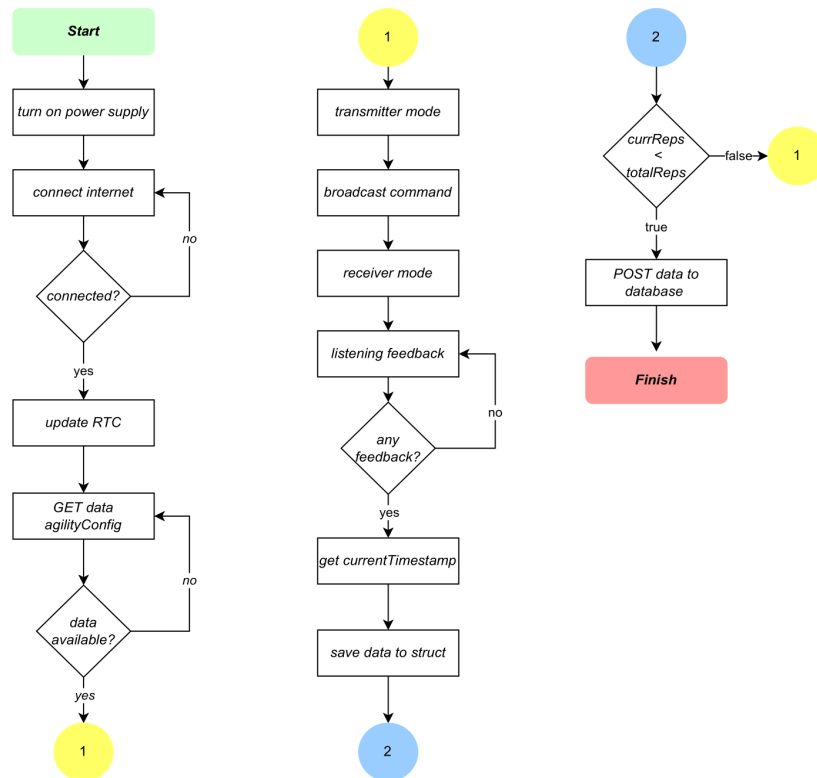


Gambar 3.9 Arsitektur CB

Pada arsitektur ini, CB menggunakan mikroprosesor NodeMCU ESP8266 (1) yang memungkinkan perangkat untuk melakukan pengiriman dan penerimaan data secara nirkabel. Sehingga CB dapat berkomunikasi dengan *web server* untuk mendapatkan data konfigurasi latihan maupun tes kelincahan. Selain itu, CB juga dilengkapi dengan modul nRF24L01 (3) yang berfungsi untuk mengirim dan menerima data dari node. Saat node memberikan *feedback* berupa nodeID sesuai dengan instruksi sebelumnya, CB akan mencatat *timestamp* dari *feedback* tersebut. CB menggunakan modul RTC DS1302 (2) untuk mendapatkan komponen timestamp dengan tingkat presisi hingga *milliseconds*, CB menggunakan modul RTC DS1302 (2). Setelah repetisi program selesai, NodeMCU akan melakukan POST data ke *database* melalui API yang terdapat pada *web server*. Sedangkan komponen *buzzer* (4) dan OLED (5) berfungsi untuk memberikan visualisasi kepada pengguna berupa audio dan visual.

3.3.4.2 Diagram Alir CB

Untuk mendukung arsitektur yang telah dibangun pada CB, penulis mulai merancang diagram alir untuk menggambarkan cara kerja CB secara keseluruhan. Berikut ini adalah diagram alir dari CB.



Gambar 3.10 Diagram Alir CB

Pada saat switch berubah *state* menjadi on, aliran listrik dari *power supply* akan menghidupkan NodeMCU yang berada di dalam CB. Selanjutnya, CB akan terhubung ke koneksi internet. Setelah berhasil terhubung ke internet, CB akan mengupdate RTC dengan waktu *realtime* yang didapatkan dari API yang disediakan oleh situs “pool.ntp.org”. Kemudian CB akan meminta data *agilityConfig* ke *database* melalui API yang disediakan oleh web server pada URL [serverConfigUrl](#). Jika data yang diminta tidak tersedia di *database*, node akan terus-menerus meminta data tersebut melalui API hingga data tersedia. Data *agilityConfig* yang diminta oleh CB berupa *username*, *mode*, *pattern*, *patern type*, *set_repetition*, *status* dan *details*.

Setelah CB memperoleh data konfigurasi dari *server*, nRF24L01 akan diatur ke mode pengirim (*transmitter*). Pada mode ini, CB hanya mampu melakukan *broadcast* pesan ke node. Jika nilai variabel *pattern_type* pada konfigurasi adalah "*random*", maka CB akan memilih *array* secara acak dari variabel *pattern*. Namun, jika nilai variabel *pattern_type* adalah "*repetitive*", CB akan memilih *array* secara berurutan dari variabel *pattern*. Berikut adalah fungsi logika yang digunakan untuk menentukan node yang akan menerima instruksi dari CB:

```
if (config.patternType == random) {  
    nodeID = patternArray[random(0, patternArray.size())];  
} else {  
    nodeID = patternArray[repsCount % patternArray.size()];  
}
```

Gambar 3.11 Fungsi Logika Penentuan nodeID Pada CB

Setelah mendapatkan variabel *nodeID*, CB akan mem-*broadcast* *nodeID* tersebut melalui *pipe* yang tersedia. Setelah *nodeID* berhasil di *broadcast*, nRF24L01 akan diatur ke mode *receiver* untuk menerima *feedback* dari node. Jika node memberikan *feedback* yang sesuai maka CB akan meminta data *timestamp* ke modul RTC. Selanjutnya, data tersebut akan disimpan dalam sebuah *struct array*.

Blok proses 2 akan terus berulang selama nilai dari variabel **currReps** kurang dari **totalReps**. Setelah semua repetisi selesai dijalankan, CB akan mengunggah data ke dalam *database* menggunakan metode POST melalui API yang disediakan oleh web server. API tersebut dapat diakses melalui URL berikut: [serverUploadDataurl](#).

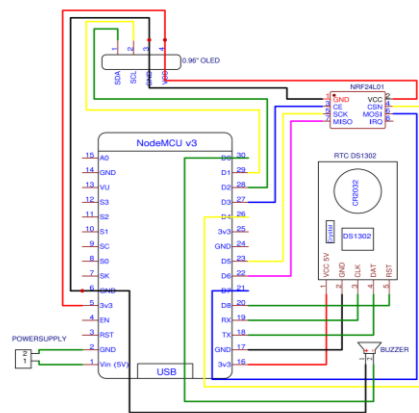
3.3.4.3 Wiring Chart

Setelah selesai membangun arsitektur dan diagram alir dari CB, peneliti mulai membuat *wiring chart* dan desain PCB dari CB menggunakan EasyEDA.

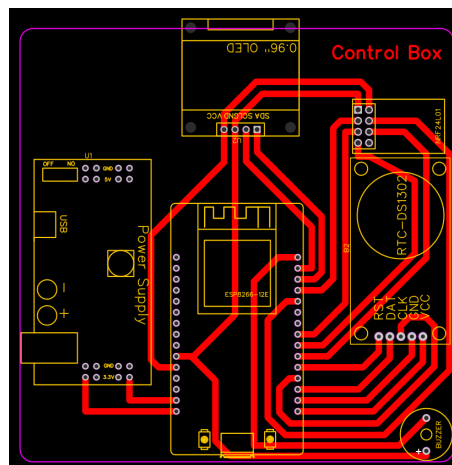
nRF24L01	NodeMCU	RTC 1302	NodeMCU	OLED 0.96"	NodeMCU
VCC	VCC	VCC	VCC	VCC	VCC
GND	GND	GND	GND	GND	GND
CE	D3 (GPIO00)	CLK	RX (GPIO03)	SCL	D1
CSN	D4 (GPIO02)	DAT	TX (GPIO01)	SDA	D2
SCK	D5 (GPIO14)	RST	D0 (GPIO16)		
MOSI	D7 (GPIO13)				
MISO	D6 (GPIO12)				
IRQ	-				

Buzzer	NodeMCU	Power Supply	NodeMCU
VCC	D8 (GPIO15)	VCC	Vin (5V)
GND	GND	GND	GND

Gambar 3.12 Tabel *Wiring Chart CB*



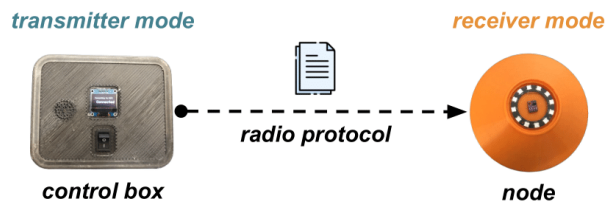
Gambar 3.13 *Wiring Chart CB*



Gambar 3.14 Desain PCB dari CB

3.3.5 Komunikasi antara Node dan CB

CB dan node saling berkomunikasi menggunakan protokol radio yang terdapat pada modul nRF24L01. Dalam protokol ini, kedua perangkat hanya dapat berkomunikasi secara *half duplex*. Oleh karena itu, saat CB mengirim pesan ke node, CB akan berperan sebagai pengirim (*transmitter*), sementara node akan berperan sebagai penerima (*receiver*).



Gambar 3.15 Komunikasi saat CB mengirim Pesan ke Node

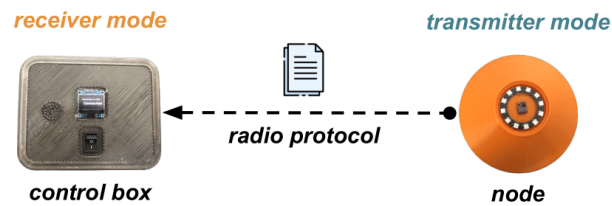
. Data yang dikirimkan oleh CB berbentuk *string* dengan ukuran 1 *byte*. Data yang terdapat di dalamnya adalah ID dari node yang akan menerima instruksi untuk menyala. Konfigurasi yang digunakan saat CB mengirim pesan ke node ditampilkan pada kode berikut:

```
1 #include <RF24.h>
2
3 const int CE      = 0;
4 const int CSN     = 2;
5
6 const byte address1[6] = "00001";
7 const byte address2[6] = "00002";
8
9 RF24 radio(CE, CSN);
10
11 void nRF24Config() {
12     radio.begin();
13     radio.setDataRate(RF24_2MBPS);
14     radio.setChannel(124);
15     radio.openWritingPipe(address1);
16     radio.openReadingPipe(1, address2);
17     radio.setPALevel(RF24_PA_MIN);
18 }
19
```

```
1 #include <RF24.h>
2
3 const int CE      = 0;
4 const int CSN     = 2;
5
6 const byte address1[6] = "00001";
7 const byte address2[6] = "00002";
8
9 RF24 radio(CE, CSN);
10
11 void nRF24Config() {
12     radio.begin();
13     radio.setDataRate(RF24_2MBPS);
14     radio.setChannel(124);
15     radio.openWritingPipe(address2);
16     radio.openReadingPipe(1, address1);
17     radio.setPALevel(RF24_PA_MIN);
18 }
19
```

Gambar 3.16 Kode Konfigurasi Radio Pada Node (kiri) dan CB (kanan)

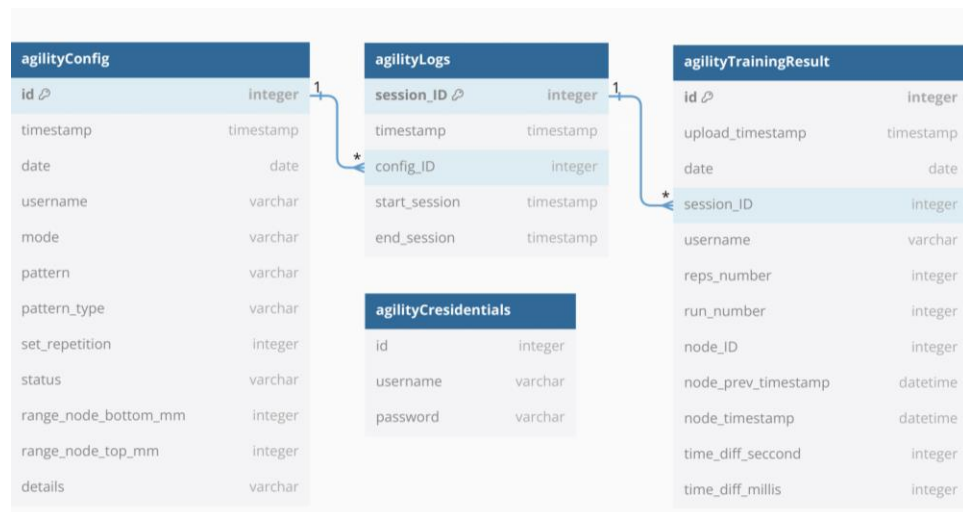
Berdasarkan kode tersebut, modul nRF24L01 menggunakan saluran frekuensi radio 124. Selain itu, alamat *pipe address2* digunakan sebagai media oleh CB untuk mengirim pesan ke node dengan kecepatan pengiriman data sebesar 2Mbps.



Gambar 3.17 Komunikasi saat Node memberikan Feedback kepada CB

Sedangkan pada saat node memberikan *feedback* kepada CB, node akan berperan sebagai pengirim (*transmitter*), sementara CB akan berperan sebagai penerima (*receiver*). Data yang dikirimkan oleh node berbentuk *string* dengan ukuran 1 *byte*. Data yang terdapat di dalamnya adalah ID dari node tersebut. Data tersebut akan dikirimkan melalui *pipe address1*. Sedangkan saluran frekuensi radio dan kecepatan pengiriman data yang digunakan tetap sama seperti konfigurasi sebelumnya.

3.3.6 Database Schema



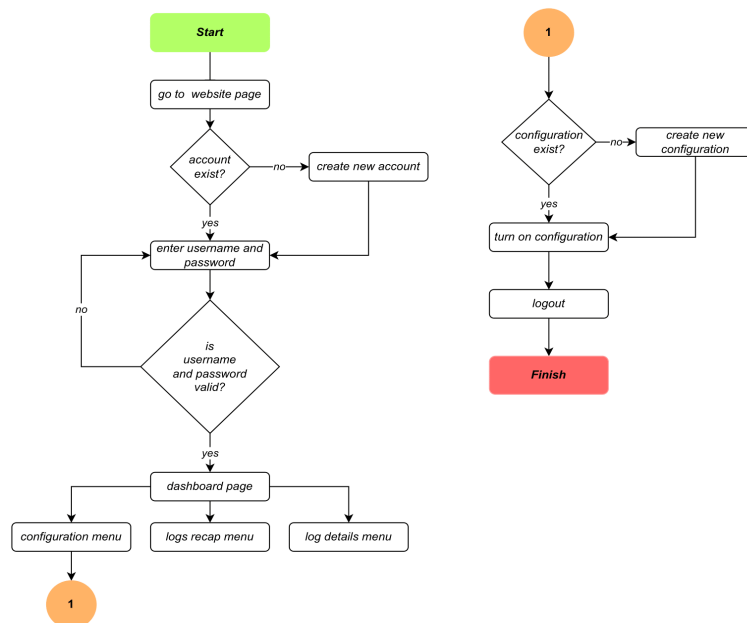
Gambar 3.18 Database Schema dari IoT Based Agility Tester

Database dari sistem *agility tester* terdiri dari empat *table*, yaitu: *agilityCresidentials*, *agilityConfig*, *agilityLogs*, dan *agilityTrainingResult*. Table *agilityCresidentials* berfungsi untuk menyimpan data kredensial pengguna, yang nantinya akan digunakan untuk verifikasi saat pengguna melakukan proses *sign in* dan *sign up*. Tabel *agilityConfig* digunakan untuk menyimpan konfigurasi yang diperlukan saat menjalankan sistem ini. Tabel

agilityLogs berperan dalam menyimpan data pelatihan dan pengujian yang dilakukan oleh pengguna. Table ini hanya menyimpan informasi waktu mulai dan waktu selesai dari pelatihan atau pengujian. Terakhir, table agilityTrainingResult berfungsi untuk menyimpan detail dari data-data dalam agilityLogs. Tabel ini juga bertindak sebagai sumber data yang akan ditampilkan pada *dashboard*.

3.3.7 Desain Web Application

Untuk mempermudah akses node dan CB secara remote, maka peneliti membuat desain dari sebuah *Web Application*. *Web application* ini berfungsi untuk menyediakan data konfigurasi untuk CB dan node serta menampilkan data hasil latihan dan tes kelincihan secara online. Berikut ini merupakan diagram alir dari *web application* dalam menyediakan konfigurasi untuk node dan CB.

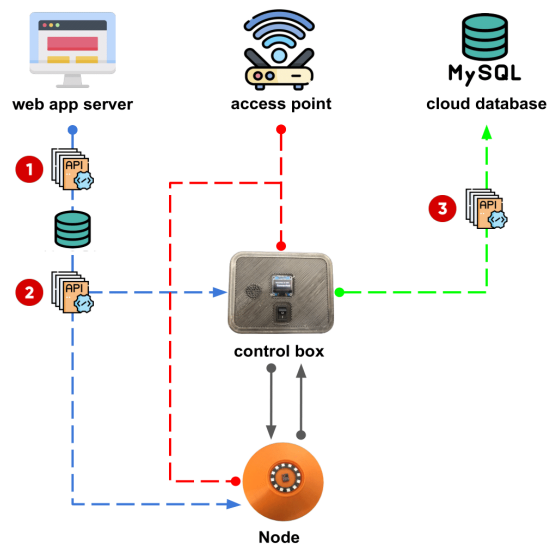


Gambar 3.19 Diagram Alir Web Application

Pertama-tama pengguna harus mengunjungi halaman *login* terlebih dahulu. Jika belum memiliki akun, pengguna dapat membuat akun terlebih dahulu di halaman *sign up* dan memasukkan *username* yang belum digunakan oleh pengguna lain. Setelah itu pengguna dapat *login* dengan menggunakan kredensial yang sudah didaftarkan.

Pada halaman *dashboard* terdapat 3 menu yaitu *configuration*, *logs recap*, dan *log details*. Pada *use-case* ini, pengguna bisa mengakses menu *configuration* terlebih dahulu, apabila konfigurasi untuk latihan maupun tes belum tersedia atau belum sesuai, maka pengguna dapat membuat konfigurasi baru, Setelah selesai membuat konfigurasi baru, maka pengguna bisa langsung mengaktifkan konfigurasi tersebut. Setelah aktivasi konfigurasi, konfigurasi tersebut akan disimpan dalam *database* dan dapat diakses secara tidak langsung oleh CB melalui sebuah API. Setelah pengguna selesai, mereka dapat melakukan *logout* dari situs web.

3.3.8 API



Gambar 3.20 API yang digunakan pada Arsitektur Sistem: Versi 2

Pada arsitektur ini, terdapat 3 skema API yang digunakan untuk menghubungkan *web apps server* dengan *database* serta *database* dengan *node* dan CB. Pada skema pertama (1), API digunakan untuk menghubungkan *web app server* dengan *database*. API yang disediakan mendukung fungsi CRUD (*create, read, update and delete*) yang terdapat pada *web apps server*. Terdapat 3 API yang digunakan pada skema pertama yaitu *add-new.php*, *edit.php* dan *delete.php*. *add-new.php* berfungsi untuk menambahkan data konfigurasi pada *table* *agilityConfig*. *edit.php* berfungsi

untuk memperbaharui data konfigurasi yang telah ada pada *table* agilityConfig. delete.php berfungsi untuk menghapus konfigurasi yang terdapat pada *table* agilityConfig. Adapun spesifikasi dari masing-masing API yang digunakan dijelaskan pada tabel berikut.

a. add-new.php

Tabel 3.2 Spesifikasi API add-new.php

Fungsi	Menambahkan data konfigurasi pada <i>table</i> agilityConfig.
Endpoint	https://base-url/agility-tester/dashboard/crudTable/add-new.php.
Metode	POST.
Parameter	username, mode, pattern, pattern_type, set_repetition, status, range_node_top_mm, range_node_bottom_mm, details dan addNewConfig.
Cara Pemakaian	<ol style="list-style-type: none"> Buka Postman. Pilih metode HTTP "POST". Masukkan URL <i>Endpoint</i>: https://base-url/agility-tester/dashboard/crudTable/add-new.php. Pilih tab "Body". Pilih opsi "form-data". Tambahkan parameter dan nilai pada kolom Key dan Value. Contohnya seperti berikut. <ul style="list-style-type: none"> • username: Dana • mode: training • pattern: 1,2,3,4,5 • pattern_type: random • repetition: 2 • status: on • range_node_top_mm: 200 • range_node_bottom_mm: 40 • details: Training using 5 node with 2 reps • addNewConfig: true
Respon	<ul style="list-style-type: none"> • <i>New record created successfully</i> (sukses). • <i>Failed: Column 'mode' cannot be null</i> (parameter <i>mode</i> bernilai <i>null</i>).

b. edit.php

Tabel 3.3 Spesifikasi API edit.php

Fungsi	Memperbaharui data konfigurasi pada <i>table</i> agilityConfig sesuai ID yang ditentukan.
Endpoint	https://base-url/agility-tester/dashboard/crudTable/edit.php.
Metode	POST.
Parameter	username, mode, pattern, pattern_type, set_repetition, status, range_node_top_mm, range_node_bottom_mm, details dan editConfig.
Cara Pemakaian	<ol style="list-style-type: none">Buka Postman.Pilih metode HTTP "POST".Masukkan URL <i>Endpoint</i>: https://base-url/agility-tester/handler/edit.php.Pilih tab "Body".Pilih opsi "form-data".Tambahkan parameter dan nilai pada kolom Key dan Value. Contohnya seperti berikut.<ul style="list-style-type: none">• id: 11• username: Dana• mode: training• pattern: 1,2,3,4,5• pattern_type: random• repetition: 2• status: on• range_node_top_mm: 200• range_node_bottom_mm: 40• details: Training using 5 node with 2 reps• editConfig: true
Respon	<ul style="list-style-type: none">• <i>Data updated successfully</i> (sukses).• <i>ID Not Found</i> (id yang dimasukan tidak terdapat pada <i>table</i> agilityConfig).

c. delete.php

Tabel 3.4 Spesifikasi API delete.php

Fungsi	Menghapus konfigurasi yang terdapat pada <i>table</i> agilityConfig sesuai id yang ditentukan.
--------	--

<i>Endpoint</i>	https://base-url/agility-tester/dashboard/crudTable/delete.php.
Metode	POST.
Parameter	delete_id dan deleteConfig.
Cara Pemakaian	a. Buka Postman. b. Pilih metode HTTP "POST". c. Masukkan URL <i>Endpoint</i> : https://base-url/agility-tester/handler/delete.php. d. Pilih tab "Body". e. Pilih opsi "form-data". f. Tambahkan parameter dan nilai pada kolom Key dan Value. Contohnya seperti berikut. <ul style="list-style-type: none"> • delete_id: 11 • deleteConfig: true
Respon	<ul style="list-style-type: none"> • <i>Data deleted successfully</i> (sukses). • <i>ID Not Found</i> (id yang dimasukan tidak terdapat pada <i>table</i> agilityConfig).

Pada skema kedua (2), API digunakan untuk memberikan data konfigurasi dari *database* ke node dan CB. Pada skema ini terdapat API getConfig.php, API ini menggunakan metode GET. API ini akan meminta data konfigurasi yang tersedia pada *table* agilityConfig dengan nilai variabel status = "ON". Data konfigurasi tersebut kemudian akan diproses oleh masing-masing perangkat, sehingga CB akan menerima data konfigurasi dengan parameter *mode*, *pattern*, *pattern_type*, *set_repetition*, *status* dan *details*. Sedangkan node akan menerima data konfigurasi dengan parameter *range_node_top_mm* dan *range_node_bottom_mm*. Adapun spesifikasi dari API getConfig.php yang digunakan dijelaskan pada tabel berikut.

Tabel 3.5 Spesifikasi API getConfig.php

Fungsi	Memberikan data konfigurasi dari <i>database</i> ke node dan CB.
<i>Endpoint</i>	https://base-url/agility-tester/handler/getConfig.php.
Metode	GET.
Parameter	id, timestamp, username, mode, pattern, pattern_type, set_repetition, status, range_node_top_mm,

	range_node_bottom_mm, session_ID, node_delay_mm dan controlbox_delay_ms.
Cara Pemakaian	GET https://base-url/agility-tester/handler/getConfig.php.
Respon	<ul style="list-style-type: none"> [<pre> { "id": "7", "timestamp": "2023-08-26 20:39:48", "username": "Dana", "mode": "training", "pattern": "1,2,3,4,5", "pattern_type": "random", "set_repetition": "2", "status": "on", "range_node_bottom_mm": "50", "range_node_top_mm": "150", "session_ID": "28", "node_delay_ms": "17", "controlbox_delay_ms": "19" } </pre>]. • 0 results (tidak ada konfigurasi yang aktif).

Pada skema ketiga (3), API digunakan untuk mengunggah data hasil latihan dan tes kelincahan kedalam *database*. Pada skema ini terdapat API `uploadDataTraining.php`. API ini berfungsi untuk menambahkan data hasil latihan dan tes kelincahan pada *table* `agilityTrainingResult`. Adapun spesifikasi dari API `uploadDataTraining.php` yang digunakan dijelaskan pada tabel berikut.

Tabel 3.6 Spesifikasi API `uploadTrainingData.php`

Fungsi	Mengunggah data hasil latihan atau tes kelincahan kedalam <i>database</i> .
Endpoint	https://base-url/agility-tester/handler/uploadDataTraining.php.
Metode	POST.
Parameter	session_ID, username, run_number, reps_number, node_ID, node_prev_timestamp, node_timestamp, time_diff_second dan time_diff_millis.

Cara Pemakaian	a. Buka Postman. b. Pilih metode HTTP "POST". c. Masukkan URL <i>Endpoint</i> : https://base-url/agility-tester/handler/uploadDataTraining.php . d. Pilih tab "Body". e. Pilih opsi "form-data". f. Tambahkan parameter dan nilai pada kolom Key dan Value. Contohnya seperti berikut. <ul style="list-style-type: none"> • session_ID: 30 • username: Dana • run_number: 1 • reps_number: 1 • node_ID: 1 • node_prev_timestamp: 2023-08-12 17:57:51.398 • node_timestamp: 2023-08-12 17:57:52.158 • time_diff_second: 0 • time_diff_millis: 760
Respon	<ul style="list-style-type: none"> • <i>Data inserted successfully</i> (sukses). • <i>Error: Column 'username' cannot be null</i> (parameter username bernilai <i>null</i>).

Saat menulis laporan tugas akhir, domain yang digunakan sebagai base URL dari API adalah <https://weather-database.site/>. Namun, penting untuk diingat bahwa domain tersebut hanya dapat digunakan dalam jangka waktu tertentu. Jika hosting pada rumahweb.com tidak diperpanjang, maka base URL tersebut tidak akan dapat diakses lagi.

3.4 Metode Pengujian

Pengujian tugas akhir ini terbagi dalam beberapa kategori. Masing-masing kategori pengujian beserta rincian, metode pengujian, dan prosedur pengujian terurai mulai dari **Tabel 3.7** hingga **Tabel 3.8**.

Tabel 3.7 Metode dan Prosedur Pengujian Performa Sistem

No.	Metode	Prosedur
1	Uji Coba Sistem	Peneliti akan melakukan pengujian terhadap jarak komunikasi maksimal yang bisa dicapai oleh node dan

		CB dengan menggunakan 2 jenis <i>power supply</i> yang berbeda, yaitu baterai 18650 dan <i>socket</i> listrik.
2	Uji Coba Sistem	Peneliti akan melakukan pengujian terhadap latensi pengiriman data antar node, dengan variasi jarak antara node dan CB 1 hingga 5 meter.

Tabel 3.8 Metode dan Prosedur Pengujian Fungsionalitas Sistem

No.	Metode	Prosedur
1	Uji Coba Sistem	Prototipe akan diuji untuk melatih kelincahan, dengan skema pengujian sebagai berikut: jarak antara node dan CB adalah 2 meter. Variasi jumlah node yang digunakan adalah 2 hingga 5 node, dan setiap skema pengujian dilakukan sebanyak 2 repetisi.
2	Uji Coba Sistem	Prototipe akan diuji untuk melakukan uji kelincahan dengan menggunakan uji T-Test.

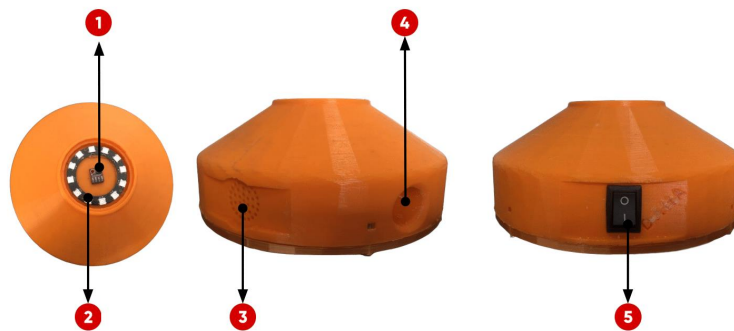
BAB 4

HASIL DAN PEMBAHASAN

4.1 *Prototipe Agility Tester*

Hasil dari penelitian ini adalah sebuah sistem pengujian kelincahan berbasis IoT yang dapat membantu individu dalam menjalankan latihan dan uji kelincahan secara mandiri. Sistem ini terdiri dari perangkat keras berupa node dan CB, serta perangkat lunak berupa aplikasi web. Perangkat keras dan perangkat lunak dalam sistem ini terintegrasi secara menyeluruh melalui koneksi internet dengan menggunakan protokol HTTP.

4.1.1 Node

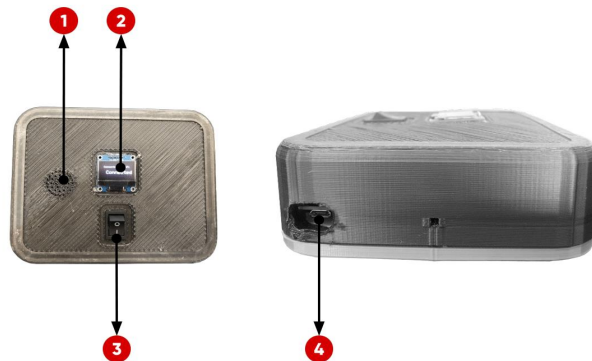


Gambar 4.1 Prototipe Node

Pada penelitian ini telah dirakit 5 buah node untuk mendukung fungsionalitas dari sistem ini, Gambar 4.1 memperlihatkan wujud dari prototipe node yang digunakan dari beberapa sisi dengan keterangan sebagai berikut:

1. Sensor VL53L0X
2. NeoPixel LED
3. *Buzzer*
4. *Port USB Type B*
5. *Switch*

4.1.2 CB

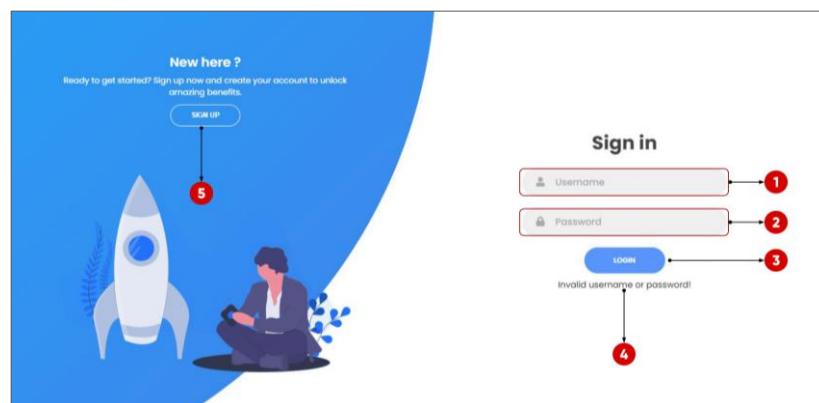


Gambar 4.2 Prototipe CB

Pada penelitian ini, peneliti juga berhasil membuat prototipe dari CB. digunakan dari beberapa sisi dengan keterangan sebagai berikut: Gambar 4.2 memperlihatkan wujud prototipe dari CB yang

1. *Buzzer*
2. *OLED 0.96"*
3. *Switch*
4. *Port USB Type B*

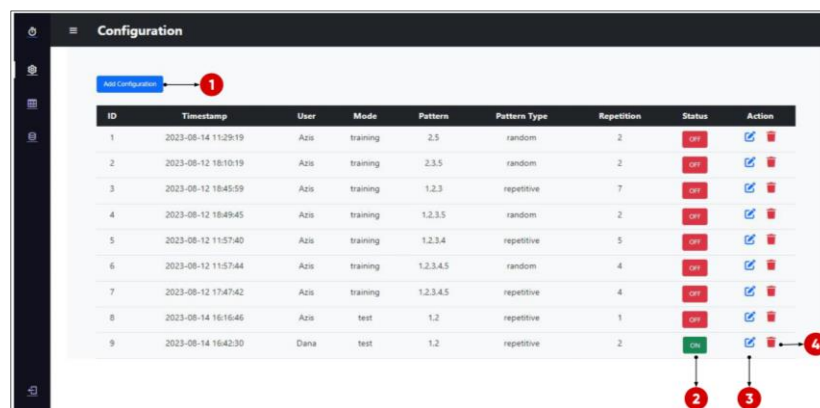
4.1.3 Web Application



Gambar 4.3 Login Page

Pada penelitian ini, peneliti juga telah berhasil membuat sebuah *web application* yang telah di-*hosting* melalui rumahweb.com, sehingga pengguna bisa mengakses web tersebut secara online.

Pengguna dapat mengakses *web application* pada URL: [loginPage](#). Pengguna bisa membuat akun terlebih dahulu dengan mengklik tombol SIGN UP (5). Namun, jika pengguna sudah memiliki akun, mereka hanya perlu mengisi kredensial berupa nama pengguna dan kata sandi pada kolom 1 dan 2, lalu klik tombol LOGIN (3). Jika nama pengguna dan kata sandi yang dimasukkan telah terverifikasi oleh sistem, pengguna akan diarahkan menuju halaman dasbor. Namun, jika nama pengguna atau kata sandi yang dimasukkan salah, situs web akan menampilkan pesan kesalahan “*Invalid username or password*” (4).



Gambar 4.4 Laman Dashboard: Configuration

Halaman *dashboard* memiliki 3 menu utama yaitu *configuration*, *logs recap* dan *logs details*. Pada menu *configuration*, pengguna dapat mengatur skema latihan dan uji kelincahan yang ingin digunakan. Pengguna dapat mengaktifkan konfigurasi dengan cara menekan tombol Status (2) pada baris konfigurasi yang ingin diaktifkan. Konfigurasi yang aktif ditandai dengan status ON.

Menu ini juga dilengkapi dengan fungsi CRUD (*Create, Read, Update, Delete*), sehingga pengguna bisa membuat skema konfigurasi secara *custom* dengan mengklik tombol *Add Configuration* (), Ketika tombol tersebut di klik maka akan muncul *pop-up* seperti berikut:

Add New Configuration

Username
Enter Username

Mode
Select

Pattern
Enter Node Patern Example 1,2,3,4,5

Pattern Type
Select

Repetition
Enter the Number of Repetitions

Status
Select

Range Node (Bottom)
Enter Values in mm

Range Node (Top)
Enter Values in mm

Details
Enter Descriptions of this Training/Test

Close Save Data

Gambar 4.5 Pop Up Add Configuration

Pada menu ini pengguna juga dapat mengubah skema konfigurasi yang sudah ada dengan mengklik ikon edit (3), Ketika ikon edit ditekan, maka akan muncul *pop-up* seperti berikut:

Edit Configuration

Username
Azis

Mode
Training

Pattern
2,5

Pattern Type
Random

Repetition
2

Status
Off

Range Node (Bottom)
40

Range Node (Top)
200

Details
distance between node and CB 2 meters

Close Update Data

Gambar 4.6 Pop up Edit Configuration

Selain itu, pengguna juga dapat menghapus konfigurasi yang sudah tidak digunakan dengan cara mengklik ikon hapus (4).

Session ID	Date	Configuration ID	Username	Session Start	Session End	Details	Recap	Chart
1	2023-08-12	1	Azis	17:13:00.000	18:00:57.644	distance between node and CB	Details	Chart
2	2023-08-12	1	Azis	18:01:04.801	18:02:25.436	distance between node and CB 2 meters	Details	Chart
3	2023-08-12	2	Azis	18:05:52.177	18:07:54.950	distance between node and CB 2 meters	Details	Chart
4	2023-08-12	4	Azis	18:34:49.000	19:45:15.078	distance between node and CB 2 meters	Details	Chart
5	2023-08-12	4	Azis	18:36:08.000	19:45:20.812	distance between node and CB 2 meters	Details	Chart
6	2023-08-12	4	Azis	18:38:13.000	19:45:25.717	distance between node and CB 2 meters	Details	Chart
7	2023-08-12	4	Azis	18:45:59.000	19:45:28.895	distance between node and CB 2 meters	Details	Chart
8	2023-08-14	9	Diana	12:09:45.000	14:46:40.991	Latency Test	Details	Chart
9	2023-08-14	9	Diana	14:53:05.000	14:53:41.954	Latency Test 1 Meter	Details	Chart
10	2023-08-14	9	Diana	14:54:45.000	15:38:42.706	Latency Test 2 Meter	Details	Chart
11	2023-08-14	9	Diana	15:41:34.000	15:48:52.549	Latency Test 3 Meter	Details	Chart
12	2023-08-14	9	Diana	15:55:52.549	15:48:59.549	Latency Test 4 Meter	Details	Chart
13	2023-08-14	9	Diana	16:00:00.000	16:10:52.549	Latency Test 5 Meter	Details	Chart

Gambar 4.7 Laman Dashboard: Logs Recap

Pada menu *Logs Recap*, Pengguna dapat melihat *log* hasil latihan secara *historical*, untuk melihat rekap dari *log* tersebut, pengguna dapat mengklik menu *Details* (1), sedangkan untuk melihat data tersebut dalam bentuk *barchart*, pengguna dapat mengklik tombol *Chart* (2).

4.2 Analisis Performa Sistem

4.2.1 Jarak Jangkauan

Berdasarkan metode pengujian pada Tabel 3.2, peneliti melakukan pengujian terhadap jarak komunikasi maksimum yang mampu dilakukan oleh node dan CB dengan menggunakan dua jenis *power supply* yang berbeda, adapun hasil dari pengujian tersebut dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil Uji Jarak Jangkauan antara Node dan CB

<i>Power Supply</i>	Tegangan	Arus	Jarak Jangkauan
Baterai 18650	5V	2A	5 Meter
Soket Listrik	5V	3A	7 Meter

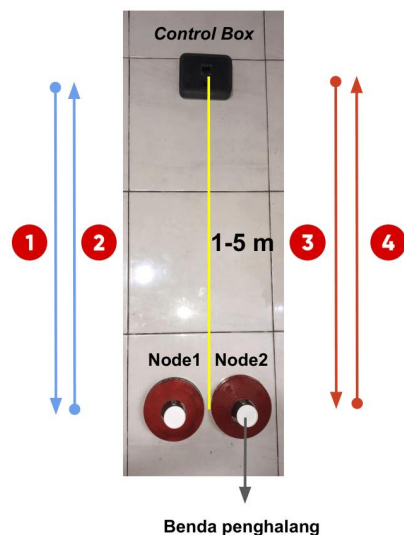
Pada pengujian pertama *power supply* yang digunakan berasal dari baterai 18650. Baterai ini menghasilkan tegangan 3.7V (DC), *output* dari baterai ini kemudian dihubungkan dengan sebuah modul *battery* 18650 *shield power supply* yang didalamnya terdapat DC-DC *step-up converter*, sehingga modul ini mampu menghasilkan tegangan *output* 5V dan arus *output* 2A. Pada pengujian menggunakan *power supply* yang berasal dari baterai 18650, node dan CB hanya mampu berkomunikasi hingga jarak 5 meter.

Sedangkan pada pengujian kedua *power supply* yang digunakan berasal dari soket listrik, *output* dari socket tersebut kemudian dihubungkan dengan *charger smartphone* sehingga menghasilkan tegangan *output* 5V dan arus *output* 3A. Pada pengujian kedua, node dan CB mampu berkomunikasi hingga jarak 7 meter.

Berdasarkan pengujian tersebut peneliti menyimpulkan bahwa jarak komunikasi antara node dan CB dipengaruhi oleh *power supply* yang digunakan, semakin tinggi arus *output* dari *power supply* tersebut maka akan semakin jauh jarak komunikasi yang bisa dilakukan oleh sistem ini, namun dikarenakan *requirements* dari sistem ini harus *portable*, maka penggunaan baterai 18650 merupakan solusi yang ditawarkan oleh penulis.

4.2.2 Latensi

Berdasarkan metode pengujian pada Tabel 3.2, telah dilakukan pengujian terhadap latensi yang diperlukan agar dua buah node dapat saling berkomunikasi melalui perantara CB. Adapun skema dari pengujian yang dilakukan adalah sebagai berikut.



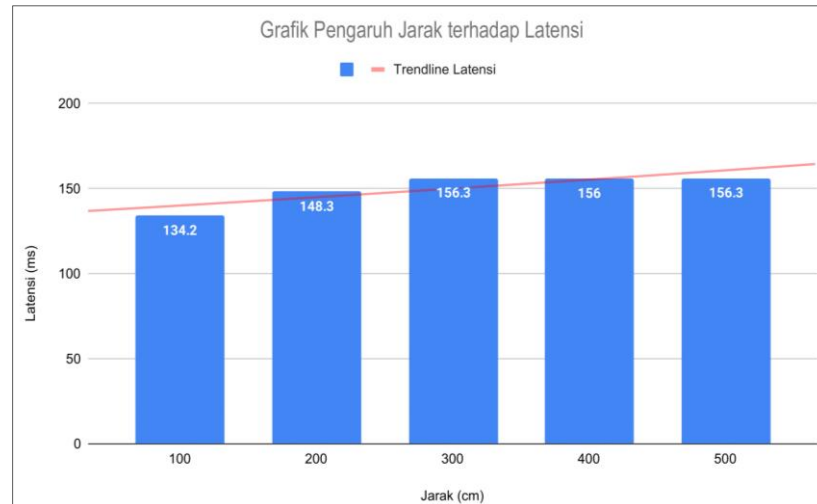
Gambar 4.8 Skema Pengujian Latensi

Pada pengujian ini peneliti menempatkan sebuah benda penghalang diatas node, sehingga ketika CB mengirim perintah kepada node, node akan langsung mengirimkan *feedback* kepada CB. Adapun definisi latensi pada pengujian ini adalah selisih waktu antara penerimaan *feedback* dari node1 (2) dan penerimaan *feedback* dari node2 (4). Berdasarkan definisi tersebut latensi yang dimaksud adalah waktu yang dibutuhkan oleh sistem untuk mengirim perintah ke node2 (3) dan menerima *feedback* dari node2 (4). Adapun hasil dari pengujian tersebut adalah sebagai berikut.

Tabel 4.2 Hasil Uji Latensi

Jarak (cm)	Latensi Minimum (ms)	Latensi Maksimum (ms)	Latensi Rata-rata (ms)
100	119	156	134,2
200	133	156	148,3
300	156	157	156,3
400	156	156	156
500	156	157	156,3
Rata-rata			150,2

Data tersebut merupakan hasil dari 15 kali pengujian yang telah dilakukan. Dari ke-15 pengujian tersebut didapatkan bahwa latensi minimum dicapai pada pengujian dengan jarak 1 meter, sedangkan latensi maksimum dicapai pada pengujian dengan jarak 3 dan 5 meter. Pada pengujian ini sistem dapat mencapai latensi minimum sebesar 119 ms, dan latensi maksimum sebesar 157 ms dengan rata-rata latensi 150,2 ms. Berdasarkan data tersebut peneliti mencoba membuat grafik jarak terhadap latensi yang ditampilkan pada gambar berikut.

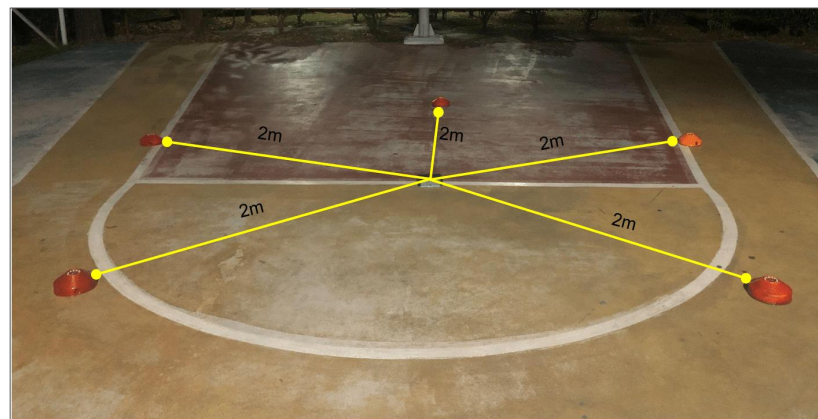


Gambar 4.9 Grafik Jarak node dan CB terhadap Latensi

Gambar 4.9 menunjukkan bahwa semakin jauh jarak antara node dengan CB, maka latensi yang ditimbulkan untuk mengirim data antar node akan cenderung lebih tinggi.

4.3 Implementasi Sistem

4.3.1 Latihan Kelincahan



Gambar 4.10 Setup Latihan Kelincahan dengan 5 Node

Pengujian fungsionalitas sistem dilakukan pada lapangan basket perumahan Verdant Ville, BSD. Pengujian pertama adalah uji fungsionalitas sistem pada saat latihan kelincahan, pada pengujian ini CB diletakkan di tengah-tengah lapangan, kemudian node lainnya kan mengelilingi CB dengan jarak 2 meter seperti yang terlihat pada Gambar 4.10. Adapun hasil dari pengujian ini dapat dilihat pada tabel berikut:

Tabel 4.3 Hasil Uji Latihan Kelincahan

No	Nama	Jumlah Node	Jumlah Repetisi	Waktu Total (ms)	Waktu Rata-rata (ms)
1	Azis	2	2	3549	1183
2	Azis	3	2	9837	1967,4
3	Azis	3	2	9941	1988,2
4	Dicky	4	2	12984	1854,9
5	Dicky	4	2	10821	1545,9
6	Novan	4	2	12100	1728,6
7	Azis	5	2	17703	1967
8	Dicky	5	2	19354	2150,5
9	Novan	5	2	16384	1820,5
10	Cakra	5	2	17508	1945,3

Berdasarkan hasil dari pengujian tersebut, waktu rata-rata terendah diperoleh pada saat pengujian pertama, dengan rata-rata 1183 ms, sedangkan waktu rata-rata tertinggi diperoleh saat pengujian ke-8, dengan rata-rata 2150,5 ms.

Data tersebut merupakan hasil pengolahan manual dari data yang terdapat pada *database* menggunakan Microsoft Excel, dan untuk menguji apakah *web application* telah menampilkan data dengan benar, data olahan sebelumnya dibandingkan dengan data yang ditampilkan pada web. Hasilnya, kedua data menampilkan nilai yang sama. Selain itu, *web application* juga menampilkan barchart dari data latihan yang telah tersimpan. Berikut adalah *barchart* yang ditampilkan oleh web:



Gambar 4.11 Barchart Hasil Latihan pada Menu Logs Recap

Gambar diatas merupakan *barchart* yang ditampilkan oleh web, masing-masing nomor pada grafik tersebut merepresentasikan data dari hasil pengujian pada Tabel 4.3, dengan keterangan sebagai berikut:

1. Uji ke-7
2. Uji ke-8
3. Uji ke-9
4. Uji ke-10

4.3.2 Tes Kelincahan



Gambar 4.12 Setup T-Test

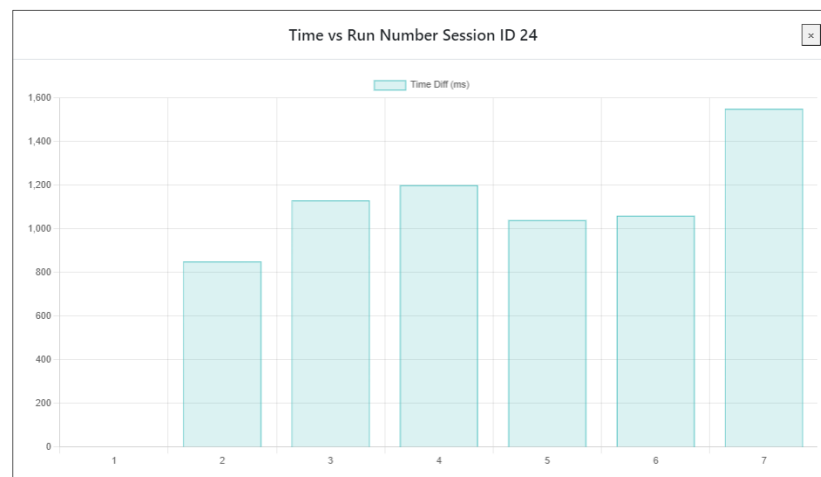
Pengujian fungsionalitas yang kedua adalah uji fungsionalitas sistem saat melakukan tes kelincahan, pengujian ini menggunakan metode T-Test yang dimodifikasi dengan skala 1:2. Pengujian ini dilakukan pada lapangan basket perumahan Verdant Ville, BSD dengan menggunakan sebuah CB dan 4 buah node yang disusun sedemikian rupa sehingga membentuk huruf T seperti yang terlihat pada Gambar 4.12. Adapun hasil dari pengujian ini dapat dilihat pada tabel berikut:

Tabel 4.4 Hasil Uji T-Test

No	Nama	Node ID	Waktu (ms)
1	Dicky	4	850
2	Dicky	5	1130
3	Dicky	4	1200

4	Dicky	3	1040
5	Dicky	4	1060
6	Dicky	2	1550
Waktu Total			6830

Berdasarkan hasil dari pengujian tersebut, waktu terendah diperoleh pada saat lari ke-1, sedangkan tertinggi diperoleh saat lari ke-6. Data tersebut telah tersimpan pada *database*, Sistem juga menampilkan grafik yang mempresentasikan hasil dari pengujian tersebut, adapun grafik yang ditampilkan oleh sistem adalah sebagai berikut:



Gambar 4.13 Barchart Hasil Uji T-Test

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dalam menyimpulkan tugas akhir ini, terdapat beberapa hal yang diperoleh berdasarkan rumusan masalah yang telah diajukan, implementasi sistem, serta pengujian langsung di lapangan.

1. Jarak jangkauan maksimum yang dapat dicapai oleh prototipe ini adalah 5 meter, hal ini dipengaruhi beberapa hal yaitu, nRF24L01 yang digunakan pada penelitian ini tertutup oleh *cover* prototipe sehingga komunikasinya terganggu, kemudian prototipe ini juga ditenagai oleh baterai 18650, dimana kapasitasnya hanya 1000 mAh dan outputnya kurang stabil.
2. Metode komunikasi menggunakan radio memiliki latensi yang lebih kecil dibandingkan dengan menggunakan websocket. Metode komunikasi websocket pada sistem ini memiliki latensi pada rentang 500-1000 ms sedangkan metode komunikasi radio pada sistem ini bisa mencapai latensi minimum sebesar 119 ms dengan rata-rata latensinya adalah 150.2 ms.
3. Sistem *agility tester* berbasis IoT ini telah berhasil melakukan latihan dan tes kelincahan, mencatat waktu secara presisi dan menampilkan datanya secara historis pada *web dashboard*.

5.2 Saran

Berdasarkan temuan dari penelitian yang dilakukan, terdapat beberapa saran yang dapat diusulkan dalam rangka meningkatkan efektivitas sistem *agility tester* berbasis IoT, antara lain:

1. Untuk penelitian selanjutnya, agar *web application* lebih sistematis dan terstruktur, disarankan untuk mengembangkan web ini menggunakan framework yang telah tersedia seperti React, Vue, Laravel, Django.

2. Untuk penelitian selanjutnya, agar jarak jangkauan pada sistem bisa melebihi 5 meter, disarankan untuk membuat desain cover yang tidak menutupi modul radio, selain itu disarankan juga mengganti baterai 18650 dengan baterai dengan kapasitas lebih tinggi dan output lebih stabil seperti baterai 21700.

DAFTAR PUSTAKA

- [1] Sheppard, J., & Young, W. (2006). Agility literature review: Classifications, training and testing. In *Journal of Sports Sciences* (Vol. 24, Issue 9, pp. 919–932). <https://doi.org/10.1080/02640410500457109>
- [2] Young, W. B., & Willey, B. (2010). Analysis of a reactive agility field test. *Journal of Science and Medicine in Sport*, 13(3), 376–378. <https://doi.org/10.1016/j.jsams.2009.05.006>
- [3] Kovacikova, Z., & Zemková, E. (2021). The Effect of Agility Training Performed in the Form of Competitive Exercising on Agility Performance. *Research Quarterly for Exercise and Sport*, 92(3), 271–278. <https://doi.org/10.1080/02701367.2020.1724862>
- [4] Raya, M. A., Gailey, R. S., Gaunaud, I. A., Jayne, D. M., Campbell, S. M., Gagne, E., Manrique, P. G., Muller, D. G., & Tucker, C. (2013). Comparison of three agility tests with male servicemembers: Edgren Side Step Test, T-Test, and Illinois Agility Test. *Journal of Rehabilitation Research and Development*, 50(7), 951–960. <https://doi.org/10.1682/JRRD.2012.05.0096>
- [5] Germa', G., Vicente-Rodri'guez, G., Rodri'guez, R., Rey-Lo'pez, J. P., Lo'pez, L., Rui'z, J. R., Rui'z, R., Jimé Nez-Pavo'n, D., Pavo'n, P., Bergman, P., Ciarapica, D., Heredia, J. M., Molnar, D., Gutierrez, A., Moreno, L. A., & Ortega, F. B. (2011). Interrater reliability and time measurement validity of speed–agility field tests in adolescents. *J Strength Cond Res* 25(7): 2059–2063.
- [6] Paul, D. J., Gabbett, T. J., & Nassis, G. P. (2016). Agility in Team Sports: Testing, Training and Factors Affecting Performance. In *Sports Medicine* (Vol. 46, Issue 3, pp. 421–442). Springer International Publishing. <https://doi.org/10.1007/s40279-015-0428-2>
- [7] Forster, J. W. D., Uthoff, A. M., Rumpf, M. C.; Cronin, J. B. (2022). Training to Improve Pro-Agility Performance: A Systematic Review. *Journal of Human Kinetics*, 85 (1), 35–51. <https://doi.org/10.2478/hukin-2022-0108>.
- [8] FITLIGHT®, [Online]. Available: <https://www.fitlighttraining.com/>. [Accessed 02 March 2023].

- [9] SmarTracks, [Online]. Available: <https://smartracks.run/>. [Accessed 03 March 2023].
- [10] SmartSpeed, [Online]. Available: [https://valdperformance.com/smart speed/](https://valdperformance.com/smart-speed/). [Accessed 03 March 2023].
- [11] Young, W., & Farrow, D. (n.d.). The Importance of a Sport-Specific Stimulus for Training Agility. *National Strength and Conditioning Association*. DOI: 10.1519/SSC.0b013e31828b6654.
- [12] STMicroelectronics. “VL53L0X - Time-of-Flight (ToF) ranging sensor. [Online]. Available: <https://www.st.com/en/imaging-and-photonics-solutions/vl53l0x.html>. [Accessed 14 August 2023].
- [13] E. Systems, "ESP8266EX Datasheet" [Online]. Available: https://cdn-shop.adafruit.com/product-files/2471/0A-ESP8266__Datasheet__EN_v4.3.pdf. [Accessed 14 August 2023].
- [14] NORDIC SEMICONDUCTOR®. “NRF24L01 Datasheet, Equivalent, 2.4GHz Transmitter ”. [Online]. Available: <https://datasheetspdf.com/pdf-file/829184/Nordic/NRF24L01/1>. [Accessed 14 Aug 2023].
- [15] Worldsemi. “WS2812 Intelligent control LED integrated light source”. [Online]. Available: <https://www.digikey.com/htmldatasheets/production/1960329/0/0/1/ws2812.pdf>. [Accessed 14 August 2023].
- [16] KEYWORD®, “SFM27-buzzer-Datasheet”. [Online]. Available: <https://www.kwtss.com/wp-content/uploads/2020/05/SFM27-buzzer-Datasheet.pdf>. [Accessed 10 March 2023].
- [17] Maxim Integrated, “DS1302 Trickle-Charge Timekeeping Chip”. [Online]. Available: <https://pdf.utmel.com/r/datasheets/maximintegrated-ds1302ztr-datasheets-8160.pdf>. [Accessed 14 August 2023].
- [18] VISHAY®, “128 x 64 Graphic OLED”. [Online]. Available: <https://www.vishay.com/docs/37902/oled128o064dbpp3n00000.pdf>. [Accessed 14 August 2023].

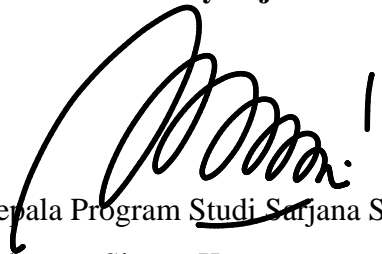
LAMPIRAN

Lampiran 1. Kartu Bimbingan Tugas Akhir









KARTU BIMBINGAN TUGAS AKHIR
S1 SEKOLAH STEM TERAPAN
PRASETIYA MULYA
JAKARTA





Nama Mahasiswa : NENGAH SWARDANA
NIM : 23401910002
Program Studi : S1 REKAYASA SISTEM KOMPUTER
Judul TA : DESIGN AND IMPLEMENTATION OF IOT-BASED
AGILITY TESTER FOR SPORTS PERFORMANCE
EVALUATION
Pembimbing : 1) Djukarna, M.T.
2) Edwin Yudayana, M.T.
Tanggal Mulai : 2 Februari 2023
Selesai Riset : 21 Agustus 2023









Menyetujui:


Kepala Program Studi Sarjana S1
Rekayasa Sistem Komputer

Lampiran 2. Log Bimbingan Tugas Akhir

Hari, Tanggal	Jam		Topik Bahasan & Saran	Paraf Inisial Pembimbing 1	Paraf Inisial Pembimbing 2
	Mulai	Selesai			
Selasa, 7/02/2023	13.01	14.06	Diskusi bersama pak Djukarna. Beliau memberikan gambaran dan saran mengenai arsitektur dari sistem <i>agility tester</i> yang diajukan.		
Sabtu, 11/03/2023	20.36	21.38	Diskusi bersama pak Edwin. Beberapa hal yang perlu diperhatikan: <ol style="list-style-type: none"> 1. Coba bandingkan delay antara komunikasi Http dengan websocket. 2. Coba tanyakan terkait <i>budgeting</i> TA dengan mba Windi. 3. Coba riset sensor yang lebih <i>reliable</i> dibanding sensor sentuh. 4. Pakai RTC untuk kalibrasi <i>timestamp</i>. 		
Selasa, 14/03/ 2023	18.28	20.21	Bimbingan dengan pak Edwin. Belajar konsep dasar websocket, sintaks websocket pada bahasa c++ serta beberapa <i>tools</i> yang bisa digunakan untuk debugging koneksi websocket.		
Senin, 3/04/2023	21.01	22.15	Bimbingan dengan pak Edwin. Beliau memberikan beberapa poin yang harus diselesaikan untuk <i>next meeting</i> : <ol style="list-style-type: none"> 1. Penggunaan <i>access point</i> (<i>public/private</i>). 		

			<ol style="list-style-type: none"> 2. Selesaikan 1 skenario dulu, 1 server 2 node. 3. Bagaimana metode pencatatan waktu yang digunakan (apakah disimpan pada EEPROM atau langsung ke <i>cloud</i>). 4. Buat sketsa rangkaian dari produk. 		
Jum'at, 16/06/2023	18.21	19.35	<p>Bingbingan dengan pak Edwin. Beliau memberikan beberapa poin yang harus diselesaikan untuk <i>next meeting</i>:</p> <ol style="list-style-type: none"> 1. Menggabungkan kode kalibrasi RTC dengan code node. 2. Tambah <i>field</i> informasi yang dikirim node ke server (node_name, timestamp, node status[onpress, onrelease]). 3. Selesaikan skenario test 2 node (dapatkan selisih <i>timestamp</i> yang lebih presisi). 4. Cari metode kalibrasi RTC yang lebih efisien. 		
Rabu, 21/06/2023	13.01	15.05	<p>Diskusi bersama pak Djukarna, Beberapa hal yang perlu diperhatikan:</p> <ol style="list-style-type: none"> 1. Database pada rumahweb.com sudah bisa diakses. 2. Tambahkan foto dan video progress pada PPT progress tugas akhir. 3. Buat <i>timeline</i> untuk <i>planning</i> kedepannya. 		

Selasa, 27/06/2023	15.55	16.30	Bingbingan dengan pak Edwin mengenai pergantian metode komunikasi yang digunakan yaitu dari websocket ke radio, beliau memberikan beberapa contoh proyek yang menggunakan komunikasi ini.		
Jumat, 14/07/2023	21.55	22.57	Diskusi bersama pak Edwin mengenai bagaimana arsitektur komunikasi antara <i>control box</i> dan <i>web server</i> . Beliau memberikan saran untuk menggunakan salah satu dari 2 metode yang telah ada yaitu menggunakan API dan menggunakan MQTT.		
Senin, 24/07/2023	17.55	18.55	Diskusi bersama pak Edwin mengenai tampilan <i>web application</i> yang telah dibuat. Beliau memberikan masukan untuk menambahkan menu khusus untuk menampilkan <i>dashboard</i> hasil latihan.		
Selasa, 08/08/2023	17:35	18.25	Diskusi Bersama pak Edwin mengenai papir yang telah dibuat.		

Lampiran 3. Surat Persetujuan Sidang Tugas Akhir



UNIVERSITAS
PRASETIYA MULYA

PERSETUJUAN MAJU SIDANG TUGAS AKHIR

Nama Mahasiswa : NENGAH SWARDANA
NIM : 23401910002
Program Studi : Computer Systems Engineering
Judul Tugas Akhir : DESIGN AND IMPLEMENTATION OF IOT-BASED AGILITY
TESTER FOR SPORTS PERFORMANCE EVALUATION

Kelengkapan Laporan Tugas Akhir

Bubuhkan tanda (✓) pada boks dibawah ini. Semua boks harus terisi untuk dapat melanjutkan ke Sidang Tugas Akhir. Jika belum lengkap maka mahasiswa wajib untuk melengkapinya terlebih dahulu sebelum mengikuti ujian

<input checked="" type="checkbox"/>	Sampul Depan	<input checked="" type="checkbox"/>	Pendahuluan *
<input checked="" type="checkbox"/>	Sampul Dalam	<input checked="" type="checkbox"/>	Tinjauan Pustaka *
<input checked="" type="checkbox"/>	Pernyataan Keaslian Tugas Akhir	<input checked="" type="checkbox"/>	Metodologi *
<input checked="" type="checkbox"/>	Pernyataan Persetujuan Publikasi Tugas Akhir	<input checked="" type="checkbox"/>	Hasil dan Pembahasan *
<input checked="" type="checkbox"/>	Pengesahan Tugas Akhir (draft)	<input checked="" type="checkbox"/>	Kesimpulan dan Saran *
<input checked="" type="checkbox"/>	Kata Pengantar	<input checked="" type="checkbox"/>	Daftar Pustaka
<input checked="" type="checkbox"/>	Abstrak (ID & EN)	<input checked="" type="checkbox"/>	Lampiran (Opsional)
<input checked="" type="checkbox"/>	Daftar Isi	<input checked="" type="checkbox"/>	Nomor Halaman
<input checked="" type="checkbox"/>	Daftar Tabel (Opsional)	<input checked="" type="checkbox"/>	Plagiarism Cek *
<input checked="" type="checkbox"/>	Daftar Gambar (Opsional)	<input checked="" type="checkbox"/>	Kartu Bimbingan
<input checked="" type="checkbox"/>	Daftar Lampiran (Opsional)	<input checked="" type="checkbox"/>	Tata bahasa sudah baik dan benar
<input checked="" type="checkbox"/>	Daftar Singkatan (Opsional)		

Dinyatakan sudah lengkap dan boleh mengikuti Sidang akhir (Minimal 8x akumulasi bimbingan).

Laporan tugas akhir dinyatakan sudah lengkap dan boleh mengikuti sidang tugas akhir (apabila telah tutup teori dan syarat-syarat lain yang ditetapkan oleh Program Studi sudah terpenuhi).

Menyetujui,

Djukarna, M.T.

Pembimbing 1

Edwin Yudayana, M.T.

Pembimbing 2

Lampiran 4. Biaya Penelitian

Hasil akhir dari penelitian ini berupa produk purwarupa. Oleh karena itu, perancangan produk membutuhkan beberapa alat dan bahan. Tabel berikut merupakan uraian komponen beserta biayanya:

`	Barang	Qty	Satuan	Harga	Subtotal
1	NodeMCU ESP8266	6	Unit	31,300	187,800
2	RTC DS1302	1	Unit	8,500	8,500
3	nRF24L01	6	Unit	14,500	87,000
4	GY-530 VL53L0X	5	Unit	43,288	216,440
5	OLED 0.96"	1	Unit	42,500	42,500
6	LED NeoPixel	5	Unit	25,000	125,000
7	<i>Buzzer</i>	6	Unit	4,800	28,800
8	<i>Batery 18650 Shield Power Supply</i>	6	Unit	38,000	228,000
9	<i>Switch 2 Pin (On/Off)</i>	6	Unit	1,200	7,200
10	Kabel JST SM 4 Pin	30	Unit	2,700	81,000
11	Kabel JST SM 4 Pin	20	Unit	1,400	28,000
TOTAL					1,040,240

Lampiran 5. Code Control Box

a. Library

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <ArduinoJson.h>
#include <TimeLib.h>
#include <virtuabotixRTC.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <stdio.h>
#include <math.h>
```

b. Global Constants and Variables

Bagian ini mendefinisikan nilai-nilai konstan dan variabel yang digunakan dalam program.

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

const int buzzerPin = 16;
const int CE        = 0;
const int CSN       = 2;
const int CLK       = 3;
const int DAT       = 1;
const int RST       = 15;

int randNum;
int nodeNum;
std::vector<int> patternArray;

const char* ssid      = "iPhone";
const char* password  = "omswastiastu";

const char* serverConfigUrl = "http://weather-
database.site/agility-tester/handler/getConfig.php";

const char* serverUploadDataurl = "http://weather-
database.site/agility-tester/handler/uploadDataTraining.php";
```

```
const byte address1[6]      = "00001";
const byte address2[6]      = "00002";
```

c. *Struct Definitions*

Di sini, terdapat definisi untuk berbagai struktur data (Timestamp, Config, dan Data).

```
struct Timestamp {
    time_t currentTime;
    int currentSeconds;
    int currentMillis;
    int currentMinutes;
    int currentHours;
    int currentDay;
    int currentMonth;
    int currentYear;
    int prevHours;
    int prevMinutes;
    int prevSeconds;
    int prevMillis;
    String timestamp;
    String prevTimestamp;
    int timeDiffSec;
    int timeDiffMillis;
};
Timestamp times;

struct Config {
    String session_ID;
    String responseJSON;
    String username;
    String mode;
    std::vector<int> patternArray;
    String patternJSON;
    String patternType;
    int repetition;
    String status;
    int controlBoxDelay;
};
Config config;

struct Data {
    String session_ID;
    String username;
    int runNum;
```

```

int repsNum;
String nodeID;
String prevTimestamp;
String timestamp;
int timeDiffSec;
int timeDiffMillis;
};
Data data[500];

```

d. *Setup Configuration*

Bagian ini berfungsi untuk mengatur dan menginisialisasi berbagai komponen yang diperlukan saat menjalankan program.

```

WiFiUDP ntpUDP;
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -
1);
NTPClient timeClient(ntpUDP, "pool.ntp.org", 25200);
virtuabotixRTC rtcds1302(CLK, DAT, RST);
RF24 radio(CE, CSN);

```

e. *Functions for Displaying on OLED*

Bagian ini mendefinisikan sebuah fungsi untuk menampilkan teks di layar OLED.

```

void displayOLED(float textSize, int textColor, int cursorX,
int cursorY, const char* textOutput) {
    display.setTextSize(textSize);
    display.setTextColor(textColor);
    display.setCursor(cursorX, cursorY);
    display.println(textOutput);
    display.display();
}

```

f. *WiFi Connection Function*

Fungsi connectWiFi didefinisikan disini, yang mengelola proses koneksi WiFi.

```

void connectWiFi() {
    Wire.begin();
    display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
    pinMode(buzzerPin, OUTPUT);
    display.clearDisplay();
    displayOLED(1, SSD1306_WHITE, 10, 10, "Connecting to WiFi");
    WiFi.begin(ssid, password);
}

```

```

int i = 0;
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi...");
    displayOLED(1, SSD1306_WHITE, 10 + i, 20, ".");
    i++;
    delay(500);
}
display.clearDisplay();
displayOLED(1, SSD1306_WHITE, 10, 10, "Connecting to WiFi");
displayOLED(2, SSD1306_WHITE, 20, 30, "Connected");
display.clearDisplay();

for (int j = 0; j < 2; j++) {
    digitalWrite(buzzerPin, HIGH);
    delay(500);
    digitalWrite(buzzerPin, LOW);
    delay(500);
}
}

```

g. *Time and RTC Functions*

Bagian ini berisi fungsi terkait pembaruan waktu pada RTC menggunakan API yang tersedia pada “pool.ntp.org” .

```

void updateRTC() {
    timeClient.begin();
    timeClient.update();

    times.currentTime      = timeClient.getEpochTime();
    times.currentSeconds   = second(times.currentTime);
    times.currentMinutes   = minute(times.currentTime);
    times.currentHours     = hour(times.currentTime);
    times.currentDay       = day(times.currentTime);
    times.currentMonth     = month(times.currentTime);
    times.currentYear      = year(times.currentTime);

    rtcds1302.setDS1302Time(times.currentSeconds,
times.currentMinutes, times.currentHours, 2, times.currentDay,
times.currentMonth, times.currentYear);
}

```

h. *Configuration Retrieval Function*

Fungsi getConfig mengambil data konfigurasi dari API serverConfigUrl menggunakan permintaan HTTP.

```

void getConfig(){
    display.clearDisplay();
    displayOLED(1, SSD1306_WHITE, 10, 10, "Request config from");
    displayOLED(2, SSD1306_WHITE, 10, 30, "Webserver");

    WiFiClient client;
    HTTPClient http;

    int getConfig = 0;
    int failsReqCount = 0;

    while (getConfig == 0) {
        failsReqCount++;
        http.begin(client, serverConfigUrl);
        int httpResponseCode = http.GET();

        if (httpResponseCode > 0) {
            String response = http.getString();
            Serial.println(response);

            if (response == "0 results") {
                Serial.println(getConfig);
                displayOLED(1, SSD1306_WHITE, 10 + failsReqCount, 50,
                    ".");

                delay(500);

            } else {
                getConfig = 1;
                Serial.println("result available");

                const size_t capacity = JSON_ARRAY_SIZE(6) +
                    6*JSON_OBJECT_SIZE(1) + 270;

                DynamicJsonDocument doc(capacity);
                DeserializationError error = deserializeJson(doc,
                    response);

                if (error) {
                    Serial.print("deserializeJson() failed: ");
                    Serial.println(error.c_str());
                    return;
                }

                config.session_ID = doc[0]["session_ID"].as<const
                    char*>();
            }
        }
    }
}

```

```

    config.username      = doc[0]["username"].as<const
                           char*>();
    config.mode          = doc[0]["mode"].as<const
                           char*>();
    config.patternType   = doc[0]["pattern_type"].as<
                           const char*>();
    config.repetition    = doc[0]["set_repetition"];
    config.status        = doc[0]["status"].as<const
                           char*>();
    config.patternJSON   = doc[0]["pattern"].as<const
                           char*>();
    config.controlBoxDelay = doc[0]["
                           controlbox_delay_ms"];

    const char* patternJSON = doc[0]["pattern"].as<const
                              char*>();

    char* patternParse      = strtok((char*)patternJSON,
                                     ",");

    while (patternParse!= NULL) {
        config.patternArray.push_back(atoi(patternParse));
        patternParse = strtok(NULL, ",");
    }
}
else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
}
http.end();

display.clearDisplay();
displayOLED(2, SSD1306_WHITE, 10, 30, "Config");

for (int j = 0; j < 2; j++) {
    digitalWrite(buzzerPin, HIGH);
    delay(500);
    digitalWrite(buzzerPin, LOW);
    delay(500);
}

char repetitionChar[10];

display.clearDisplay();

char combinedText[50];

```



```

    sprintf(combinedText, "session : %s",
              config.session_ID.c_str());
    displayOLED(1, SSD1306_WHITE, 10, 0, combinedText);

    sprintf(combinedText, "user      : %s",
              config.username.c_str());
    displayOLED(1, SSD1306_WHITE, 10, 10, combinedText);

    sprintf(combinedText, "mode      : %s",
              config.mode.c_str());
    displayOLED(1, SSD1306_WHITE, 10, 20, combinedText);

    sprintf(combinedText, "pattern: %s",
              config.patternJSON.c_str());
    displayOLED(1, SSD1306_WHITE, 10, 30, combinedText);

    sprintf(combinedText, "type    : %s",
              config.patternType.c_str());
    displayOLED(1, SSD1306_WHITE, 10, 40, combinedText);

    sprintf(combinedText, "reps    : %s",
              itoa(config.repetition, repetitionChar, 10));
    displayOLED(1, SSD1306_WHITE, 10, 50, combinedText);
    delay(5000);

    if(config.patternType == "random"){
        randNum = random(0, config.patternArray.size());
        nodeNum = config.patternArray[randNum];
        Serial.println("end getConfig");
    } else {
        nodeNum = config.patternArray[0];
    }
}
}

```

i. *nRF24L01 Configuration Function*

Fungsi nRF24Config mengkonfigurasi modul radio nRF24L01.

```

void nRF24Config() {
    radio.begin();
    radio.setDataRate(RF24_2MBPS);
    radio.setChannel(124);
    radio.openWritingPipe(address2);
    radio.openReadingPipe(1, address1);
    radio.setPALevel(RF24_PA_MIN);
    randomSeed(analogRead(A0));
}

```

```

display.clearDisplay();
displayOLED(2, SSD1306_WHITE, 10, 30, "Prepare");

for (int j = 0; j < 5; j++) {
    digitalWrite(buzzerPin, HIGH);
    delay(500);
    digitalWrite(buzzerPin, LOW);
    delay(500);
}
display.clearDisplay();
}

```

j. *Setup Function*

Fungsi setup dieksekusi sekali saat program dimulai untuk menginisialisasi berbagai komponen. Langkah pertama, fungsi ini akan menghubungkan perangkat ke *access point* menggunakan SSID dan *password* yang telah disediakan. Selanjutnya, waktu pada modul RTC akan di sinkronisasi dengan variabel waktu dari API pool.ntp.org. Setelah itu, perangkat akan meminta konfigurasi dari *web server*. Akhirnya, konfigurasi radio akan dijalankan

```

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

    connectWiFi();

    updateRTC();

    getConfig();

    nRF24Config();
}

```

k. *Main Loop*

Fungsi *loop*, yang berjalan berulang setelah setup, berisi logika utama program. Ini mencakup pengiriman dan penerimaan data menggunakan modul nRF24L01, penyimpanan data ke dalam *array*, dan pengunggahan data ke *server*.

```

void loop() {
    int repsCount;
    for(repsCount = 0; repsCount <
        config.repetition*config.patternArray.size(); repsCount++)
    {
        char nodeNumChar[10];
        sprintf(nodeNumChar, "%d", nodeNum);

        radio.stopListening();

        if(radio.write(nodeNumChar, strlen(nodeNumChar) + 1)){
            displayOLED(1, SSD1306_WHITE, 10, 25, "Active Node");
            displayOLED(2, SSD1306_WHITE, 10, 40, "Node");
            displayOLED(2, SSD1306_WHITE, 70, 40, nodeNumChar);
            display.println(nodeNum);
        }
        else{
            Serial.print("failed send response to node");
            Serial.println(nodeNum);
        }

        while(true){
            radio.startListening();
            if(radio.available()){
                char message[32] = "";
                radio.read(&message, sizeof(message));
                Serial.print("receive message from node");
                Serial.println(message);

                saveData(message, repsCount);

                if(strcmp(nodeNumChar, message) == 0){
                    Serial.println("correct response");
                    while (nodeNum == atoi(message)){
                        if(config.patternType == "random"){
                            randNum = random(0, config.patternArray.size());
                            nodeNum = config.patternArray[randNum];
                        } else {
                            nodeNum = config.patternArray[(repsCount + 1)
                                % config.patternArray.size()];
                            char nodeNumChar[10];
                        }
                    }

                    Serial.print("next node: ");
                    Serial.println(nodeNum);
                }
                break;
            }
        }
    }
}

```

```

    }
    }
    Serial.println("waiting for node");
    delay(config.controlBoxDelay);
  }
}

for (int j = 0; j < 3; j++) {
  digitalWrite(buzzerPin, HIGH);
  delay(500);
  digitalWrite(buzzerPin, LOW);
  delay(500);
}

for (int arrayNum = 0; arrayNum < repsCount; arrayNum++) {
  sendDataToServer(data[arrayNum]);

  display.clearDisplay();
  displayOLED(2, SSD1306_WHITE, 10, 10, "Upload");
  displayOLED(2, SSD1306_WHITE, 10, 30, "Data");

  char currentReps[10];
  char currentRepsString[10];
  sprintf(currentReps, "%d", repsCount);
  sprintf(currentRepsString, "%d", (arrayNum + 1));
  displayOLED(2, SSD1306_WHITE, 65, 30, currentRepsString);
  displayOLED(2, SSD1306_WHITE, 90, 30, "/");
  displayOLED(2, SSD1306_WHITE, 100, 30, currentReps);
  delay(10);

  display.clearDisplay();
}

for (int j = 0; j < 4; j++) {
  digitalWrite(buzzerPin, HIGH);
  delay(500);
  digitalWrite(buzzerPin, LOW);
  delay(500);
}
display.clearDisplay();
}

```

1. *Data Handling Functions*

Fungsi `saveData` menyimpan data yang diterima ke dalam *struct array* data. Fungsi `sendDataToServer` mengirimkan data ke *server* melalui API `serverUploadDataurl` dengan menggunakan metode POST.

```
void saveData(char* message, int repsCount) {

    rtcds1302.updateTime();

    unsigned long currentMillis = millis();
    times.currentMillis = currentMillis % 1000;
    times.timestamp      = String(String(rtcds1302.year)
                                   + "-" + String(rtcds1302.month)
                                   + "-" + rtcds1302.dayofmonth)
                                   + " " + String(rtcds1302.hours)
                                   + ":" + String(rtcds1302.minutes)
                                   + ":" + String(rtcds1302.seconds)
                                   + "." + String(times.currentMillis);

    if(repsCount == 0){
        times.prevTimestamp = times.timestamp;
        times.prevHours     = rtcds1302.hours;
        times.prevMinutes   = rtcds1302.minutes;
        times.prevSeconds   = rtcds1302.seconds;
        times.prevMillis    = times.currentMillis;
    }

    times.timeDiffSec       = floor((((rtcds1302.hours-
        times.prevHours)*3600 +
        (rtcds1302.minutes-
        times.prevMinutes)*60 +
        (rtcds1302.seconds-
        times.prevSeconds))*1000 +
        (times.currentMillis-
        times.prevMillis))/1000);

    if(times.prevMillis <= times.currentMillis){
        times.timeDiffMillis = times.currentMillis -
                                times.prevMillis;
    } else {
        times.timeDiffMillis = 1000 - (times.prevMillis -
                                        times.currentMillis);
    };

    data[repsCount].session_ID = config.session_ID;
    data[repsCount].username   = config.username;
    data[repsCount].runNum     = repsCount+1;
```

```

data[repsCount].repsNum          = ceil(static_cast<double>
                                         (repsCount + 1) /
                                         config.patternArray.size());

data[repsCount].nodeID           = message;
data[repsCount].prevTimestamp    = times.prevTimestamp;
data[repsCount].timestamp        = times.timestamp;
data[repsCount].timeDiffSec      = times.timeDiffSec;
data[repsCount].timeDiffMillis   = times.timeDiffMillis;

times.prevTimestamp              = times.timestamp;
times.prevHours                  = rtcds1302.hours;
times.prevMinutes                = rtcds1302.minutes;
times.prevSeconds                = rtcds1302.seconds;
times.prevMillis                 = times.currentMillis;

display.clearDisplay();
display.setTextSize(1);
display.setTextColor(SSD1306_WHITE);
display.setCursor(10, 10);
display.println(times.timeDiffSec);
display.setCursor(15, 10);
display.println(", ");
display.setCursor(20, 10);
display.println(times.timeDiffMillis);
display.display();
}

void sendDataToServer(Data data) {
    if (WiFi.status() == WL_CONNECTED) {
        HTTPClient http;
        WiFiClient client;

        http.begin(client, serverUploadDataurl);
        http.addHeader("Content-Type", "application/x-www-form-
            urlencoded");

        String httpRequestData = "session_ID=" + data.session_ID +
                                   "&username=" + data.username +
                                   "&run_number=" +
                                   String(data.runNum) +
                                   "&reps_number=" +
                                   String(data.repsNum) +
                                   "&node_ID=" + data.nodeID +
                                   "&node_prev_timestamp=" +
                                   data.prevTimestamp +
                                   "&node_timestamp=" +
                                   data.timestamp +

```

```

        "&time_diff_second=" +
        String(data.timeDiffSec) +
        "&time_diff_millis=" +
        String(data.timeDiffMillis);

    int httpResponseCode = http.POST(httpRequestData);

    if (httpResponseCode > 0) {
        Serial.println("HTTP Response code: ");
        Serial.println(httpResponseCode);
        String response = http.getString();
        Serial.println(response);
    } else {
        Serial.println("Error code: " +
        String(httpResponseCode));
    }

    http.end();
}
}

```

Lampiran 6. Code Node

a. Library

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <ArduinoJson.h>
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include "Adafruit_VL53L0X.h"
#include <Adafruit_NeoPixel.h>
```

b. Global Constants and Variables

Bagian ini mendefinisikan nilai-nilai konstan dan variabel yang digunakan dalam program.

```
#define buzzerandLEDPin      15
#define numLED              12

const int CE                = 0;
const int CSN               = 2;

const char* ssid            = "iPhone";
const char* password        = "omswastiastu";
const char* serverConfigUrl = "http://weather-
                             database.site/agility-
                             tester/handler/getConfig.php";

const byte address1[6]     = "00001";
const byte address2[6]     = "00002";
```

c. Struct Definitions

Di sini, terdapat definisi untuk struktur data Config yang digunakan oleh node.

```
struct Config {
    int nodeRangeBot;
    int nodeRangeTop;
    int nodeDelay;
};
Config config;
```


d. Setup Configuration

Bagian ini berfungsi untuk mengatur dan menginisialisasi berbagai komponen yang diperlukan saat menjalankan program.

```
Adafruit_VL53L0X lox = Adafruit_VL53L0X();
Adafruit_NeoPixel strip(numLED, buzzerandLEDPin, NEO_GRB +
NEO_KHZ800);
RF24 radio(CE, CSN);
```

e. WiFi Connection Function

Fungsi connectWiFi didefinisikan di sini, yang mengelola proses koneksi WiFi.

```
void connectWiFi() {
    pinMode(buzzerandLEDPin, OUTPUT);
    Wire.begin();
    WiFi.begin(ssid, password);

    int i = 0;
    while (WiFi.status() != WL_CONNECTED) {
        i++;
        delay(500);
    }
    digitalWrite(buzzerandLEDPin, HIGH);
    delay(500);
    digitalWrite(buzzerandLEDPin, LOW);
    delay(500);
}
```

f. Configuration Retrieval Function

Fungsi getConfig mengambil data konfigurasi berupa **nodeRangeBot** dan **nodeRangeTop** dari API serverConfigUrl menggunakan permintaan HTTP.

```
void getConfig(){
    WiFiClient client;
    HTTPClient http;

    int getConfig = 0;
    int failsReqCount = 0;

    while (getConfig == 0) {
        failsReqCount++;
        http.begin(client, serverConfigUrl);
        int httpResponseCode = http.GET();
```

```

if (httpResponseCode > 0) {
    String response = http.getString();
    Serial.println(response);

    if (response == "0 results") {
        Serial.println(getConfig);
        delay(500);
    } else {
        getConfig = 1;
        Serial.println("result available");

        const size_t capacity = JSON_ARRAY_SIZE(6) +
                                6*JSON_OBJECT_SIZE(1) +
                                270;

        DynamicJsonDocument doc(capacity);
        DeserializationError error = deserializeJson(doc,
                                                    response);

        if (error) {
            Serial.print("deserializeJson() failed: ");
            Serial.println(error.c_str());
            return;
        }

        config.nodeRangeBot= doc[0]["range_node_bottom_mm"];
        config.nodeRangeTop= doc[0]["range_node_top_mm"];
        config.nodeDelay    = doc[0]["node_delay_ms"];
    }
} else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
}
Serial.println("end getConfig");
}
for (int j = 0; j < 2; j++) {
    digitalWrite(buzzerandLEDPin, HIGH);
    delay(500);
    digitalWrite(buzzerandLEDPin, LOW);
    delay(500);
}
}

```

g. nRF24L01 Configuration Function

Fungsi nRF24Config mengkonfigurasi modul radio nRF24L01, NeoPixel dan VL53L0X.

```

void nRF24Config(){
    lox.begin();
    strip.begin();
    strip.show();
    strip.setBrightness(100);
    radio.begin();
    radio.setDataRate(RF24_2MBPS);
    radio.setChannel(124);
    radio.openWritingPipe(address1);
    radio.openReadingPipe(1, address2);
    radio.setPALevel(RF24_PA_MIN);
    digitalWrite(buzzerandLEDPin, LOW);
    VL53L0X_RangingMeasurementData_t measure;
}

```

h. *Setup Function*

Fungsi setup dieksekusi sekali saat program dimulai untuk menginisialisasi berbagai komponen. Langkah pertama, fungsi ini akan menghubungkan perangkat ke *access point* menggunakan SSID dan *password* yang telah disediakan. Selanjutnya, perangkat akan meminta konfigurasi dari *web server*. Akhirnya, konfigurasi radio akan dijalankan.

```

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

    connectWiFi();

    getConfig();

    nRF24Config();
}

```

i. *Main Loop*

Fungsi *loop*, yang berjalan berulang setelah setup, berisi logika utama program. Ini mencakup pengiriman dan penerimaan data menggunakan modul nRF24L01.

```

void loop() {
    radio.startListening();
    char message[32] = "";
    if(radio.available()){
        radio.read(&message, sizeof(message));
    }
}

```

```

Serial.println(message);
String message_str = String(message);
if(message_str == "1") {
  for (int i = 0; i < strip.numPixels(); i++) {
    strip.setPixelColor(i, 0, 0, 255);
  }
  strip.show();
  while(true){
    VL53L0X_RangingMeasurementData_t measure;
    lox.rangingTest(&measure, false);
    int distance = measure.RangeMilliMeter;
    while(distance > config.nodeRangeTop | distance <
config.nodeRangeBot){
      VL53L0X_RangingMeasurementData_t measure;
      lox.rangingTest(&measure, false);
      distance = measure.RangeMilliMeter;
      Serial.println(measure.RangeMilliMeter);
      radio.stopListening();
    }
    for (int i = 0; i < strip.numPixels(); i++) {
      strip.setPixelColor(i, 0, 0, 0);
    }
    strip.show();
    radio.stopListening();

    const char data[] = "1";
    if(radio.write(&data, sizeof(data))){
    }
    else{}
    break;
  }
}
}
Serial.println("waiting for server");
delay(config.nodeDelay);
}

```