KONTROL DAN *MONITORING* FITING LAMPU BERBASIS ANDROID

PROYEK AKHIR

Laporan akhir ini dibuat dan diajukan untuk memenuhi salah satu syarat kelulusan Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Disusun Oleh :

Yogie Saputra NIRM : 0031627

Habi Alqadri NIRM : 0031610

POLITEKNIK MANUFAKTUR NEGERI BANGKA BELITUNG

TAHUN 2019

LEMBAR PENGESAHAN

JUDUL PROYEK AKHIR

KONTROL DAN *MONITORING* FITING LAMPU BERBASIS ANDROID

Oleh :

Yogie Saputra NIRM : 0031627

Habi Alqadri NIRM : 0031610

Laporan akhir ini telah disetujui dan disahkan sebagai salah satu syarat kelulusan Program Diploma III Politeknik Manufaktur Negeri Bangka Belitung

Menyetujui,

Pembimbing 1	Pembimbing 2
Indra Dwisaputra, M.T.	Ocsirendi, M.T.

Penguji 1	Penguji 2	Penguji 3
Aan Febriansyah, M.T.	Eko Sulistyo, M.T.	Charlotha, M.T.

PERNYATAAN BUKAN PLAGIAT

Yang bertanda tangan di bawah ini :

Nama Mahasiswa 1 : Yogie Saputra NIRM : 0031627

Nama Mahasiswa 2 : Habi Alqadri NIRM : 0031610

Dengan Judul : Kontrol dan *Monitoring* Fiting Lampu Berbasis Android

Menyatakan bahwa laporan akhir ini adalah hasil kerja kami sendiri dan bukan merupakan plagiat. Pernyataan ini kami buat dengan sebenarnya dan bila ternyata dikemudian hari ternyata melanggar pernyataan ini, kami bersedia menerima sanksi yang berlaku.

Sungailiat, 6 September 2019

Nama Mahasiswa	Tanda Tangan
1. Yogie Saputra	.....
2. Habi Alqadri	.....

ABSTRAK

Pengontrolan lampu biasanya dilakukan secara konvensional yaitu dengan saklar. Pada proyek akhir ini memanfaatkan Android yang digunakan oleh kebanyakan masyarakat dapat digunakan sebagai media pengontrolan fiting lampu yang dapat menghidupkan ataupun mematikan lampu baik secara langsung ataupun berdasarkan waktu yang diinginkan. Proyek akhir ini juga mampu monitoring pemakaian tegangan, arus, daya, total biaya dan waktu pemakaian lampu pada hari ini dan satu hari sebelumnya. Alat ini menggunakan sensor PZEM-004T untuk mengukur tegangan, arus dan daya secara realtime. Alat ini juga menggunakan RTC DS3213 untuk menjalankan fungsi waktu baik saat melakukan monitoring ataupun untuk waktu hidup atau matinya lampu. Kemudian waktu sekarang, waktu hidup dan waktu matinya lampu ditampilkan pada LCD OLED 0.91. Alat ini juga memanfaatkan Solid State Relay sebagai saklar fiting lampu serta SD Card sebagai penyimpan data monitoring. Semua data yang didapat diolah di NodeMCU, dan NodeMCU juga berfungsi sebagai media komunikasi antara Android dan fiting lampu dengan memanfaatkan jaringan WiFi. Pengontrolan ataupun monitoring dapat dilakukan melalui aplikasi Android yang dibuat menggunakan MIT App Inventor. Jarak efektif untuk komunikasi antara fiting lampu dengan Android yaitu kurang dari 30 meter tanpa halangan dan 15 meter dengan halangan, sedangkan komunikasi antar fiting lampu kurang dari 15 meter tanpa halangan dan 10 meter dengan halangan. Pada fungsi monitoring mengalami error maksimal 0,67% untuk pengukuran tegangan, error maksimal 4,34% untuk pengukuran arus dan error maksimal 2,6% untuk pengukuran daya.

Kata kunci: Android, *WiFi*, Topologi, NodeMCU, Lampu.

ABSTRACT

Controling lamps is usually done conventionally by a switch. In this final project utilizing a smartphone used by most people, it can be used as media for controlling lamp fitings that can turn on or turn off the lamps, either directly or based on the desired time. This final project is also able to monitoring voltage, current, power, total cost and time usage of lamps today and the previous day. This tool uses a PZEM-004T sensot to measure voltage, current, and power in realtime. This tool also uses RTC DS3231 to perform the function of the time when monitoring and the time of on or off the lamp. Then the realtime, off on time and off time the lamp displayed on the OLED LCD 0.91. This tool also utilizes Solid State Relay as saklar of the lamp and SD Card as a data storage monitoring. All data obtained is processed at NodeMCU, and NodeMCU also functions as a communication media between Android and lamp fitings by utilizing Wifi network. Controlling or monitoring can be done through an Android application created using MIT App Inventor. Effective distance for communication between lamp fitings with Android is less than 30 meters without obstacles and 15 meters with obstacles, while communication between lamp fitings is less than 15 meters without obstacles and 10 meters with obstacles. On monitoring function experiences a maximum error 0.67% for voltage measurements, maximum error of 4.34% for current measurements and maximum error 2.6% for power measurements.

Keywords: Android, WiFi, Topology, NodeMCU, Lamp.

KATA PENGANTAR

بِسْـــــــــــــمِ اللهِ الرَّحْمَنِ الرَّحِيْمِ

Assalamu'alaikum wr. wb

Puji syukur penulis panjatkan kehadirat Allah SWT atas berkat rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan laporan serta proyek akhir ini dengan baik dan tepat pada waktunya

Karya tulis proyek akhir ini disusun sebagai salah satu persyaratan dan kewajiban mahasiswa untuk menyelesaikan kurikulum program Diploma III di Politeknik Manufaktur Negeri Bangka Belitung.

Pada kesempatan ini penulis menyampaikan rasa terima kasih yang sebesar-besarnya kepada orang-orang yang telah berperan sehingga dapat terselesaikannya proyek akhir, sebagai berikut :

1. Keluarga besar (Ayah, Ibu, Adik-adik penulis, Nenek, Kakek, dll) yang selalu senantiasa memberikan kasih sayang, doa, dukungan moril maupun materi dan semangat.
2. Bapak Sugeng Ariyono, M.Eng., Ph.D. selaku Direktur Politeknik Manufaktur Negeri Bangka Belitung.
3. Bapak Indra Dwisaputra, M.T., selaku pembimbing 1 dan banyak memberikan *support* dalam proyek akhir ini.
4. Bapak Ocsirendi, M.T., selaku pembimbing 2 yang banyak memberikan masukan dalam proyek akhir ini.
5. Bapak Rino Christianto S.Tr.T., yang sangat banyak membantu, memberi masukan dan saran sehingga proyek akhir ini dapat diselesaikan.
6. Seluruh staf pengajar dan karyawan Politeknik Manufaktur Negeri Bangka Belitung.
7. Rekan-rekan mahasiswa Politeknik Manufaktur Negeri Bangka Belitung yang telah banyak membantu selama menyelesaikan proyek akhir.

8. Pihak-pihak lain yang telah memberikan bantuan secara langsung maupun tidak langssung dalam pembuatan proyek akhir ini yang tidak dapat disebutkan satu per satu.

Penulis menyadari bahwa penulisan proyek akhir ini masih jauh dari sempurna dikarenakan penulis adalah manusia biasa yang tidak luput dari kesalahan. Karena yang benar hanya datang dari Allah SWT dan yang salah datang dari penulis sendiri. Oleh karena itu, sangat diharapkan segala petunjuk, kritik dan saran yang membangun dari pembaca agar dapat menunjang pengembangan dan perbaikan penulisan selanjutnya.

Besar harapan penulis semoga makalah tugas akhir dan alat yang dibuat dapat memberikan manfaat bagi pihak yang berkepentingan pada khususnya dan baik bagi perkembangan ilmu teknologi pada umumnya.

Sungailiat, 6 September 2019

Penulis

DAFTAR ISI

HALAMAN JUDUL i
LEMBAR PENGESAHAN .....**Error! Bookmark not defined.**
PERNYATAAN BUKAN PLAGIAT .....**Error! Bookmark not defined.**
ABSTRAK iv
ABSTRACT v
KATA PENGANTAR vi
DAFTAR ISI viii
DAFTAR TABEL xi
DAFTAR GAMBAR xii
DAFTAR LAMPIRAN xiv
BAB I PENDAHULUAN 1
1.1. Latar Belakang Masalah 1
1.2. Rumusan dan Batasan Masalah 2
1.2.1. Rumusan Masalah 2
1.2.2. Batasan Masalah 2
1.3. Tujuan Proyek Akhir 3
BAB II TEORI DASAR 4
2.1. Topologi Jaringan 4
2.1.1. Topologi *Star* 4
2.1.2. Topologi *Tree* 5
2.2. Daya Listrik 5
2.3. Android 7

2.4. NodeMCU ESP8266 9
2.5. Sensor PZEM-004T 10
2.6. Modul Solid State Relay (SSR) 11
2.7. Modul RTC DS3231 12
2.8. OLED 0.91 14
2.9. Power Supply 15
2.10. Modul SD Card 16
2.11. MIT App Inventor 16
BAB III METODE PENELITIAN 18
3.1. Metode Pelaksanaan Pembuatan Proyek Akhir 18
3.2. Rancangan Algoritma Kerja *Hardware* 20
3.3. Diagram Blok 22
3.3.1. Diagram Blok *hardware* 22
3.3.2. Diagram Blok jaringan 24
BAB IV PEMBAHASAN 27
4.1. Rancangan Alat 27
4.1.1. Modifikasi *Hardware* Mekanik Fiting Lampu Berbasis Android 27
4.1.2. *Hardware* Elektrik Fiting Lampu Berbasis Android 28
4.2. Rangkaian Catu Daya 29
4.3. Pengujian NodeMCU ESP8266 30
4.4. Pengujian Sensor PZEM-004T 32
4.4.1. Prosedur Percobaan 33
4.4.2. Pengujian Sensor Dibanding Dengan Alat Ukur 35
4.5. Real Time Clock DS3231 47
4.6. Pengujian OLED 0.91 50

4.7. *Software* Fiting Lampu Berbasis Android 52
4.7.1. Aplikasi “Smart Fitting” 52
4.7.2. Pembuatan Aplikasi “Smart Fitting” 53
4.7.3. Pengujian Aplikasi “Smart Fitting” 54
4.8. Program NodeMCU ESP 8266 58
4.9. Pengujian Data *Logger* 59
4.10. Pengujian Fiting Lampu Secara Keseluruhan 62
4.10.1. Pengujian *ON/OFF* Lampu 62
4.10.2. Pengujian *Monitoring* Tegangan, Arus, Daya, Waktu dan Biaya 63
4.10.3. Pengujian *Timer* pada Lampu 65
4.10.4. Pengujian Topologi *Tree* 67
BAB V KESIMPULAN DAN SARAN 70
5.1. Kesimpulan 70
5.2. Saran 71
DAFTAR PUSTAKA 72

DAFTAR TABEL

Tabel 2.1 Spesifikasi dari NodeMCU ESP8266 9
Tabel 2.2 Spesifikasi Sensor PZEM-004T 10
Tabel 2.3 Konfigurasi pin Sensor PZEM-004T 11
Tabel 2.4 Spesifikasi *Solid State Relay* 12
Tabel 2.5 Konfigurasi pin *Solid State Relay* 12
Tabel 2.6 Spesifikasi dari Modul RTC DS3231 13
Tabel 2.7 Konfigurasi pin Modul RTC 13
Tabel 2.8 Spesifikasi dari OLED 0.91 14
Tabel 2.9 Konfigurasi pin OLED 0.91 15
Tabel 2.10 Konfigurasi pin Sensor PZEM-004T 16
Tabel 4.1 Kriteria Catu Daya yang Digunakan 29
Tabel 4.2 Kriteria Kontroler yang Digunakan 30
Tabel 4.3 Hasil Pengujian Jarak Efektif Komunikasi Antara NodeMCU dan Android 32
Tabel 4.4 Data Pengujian Lampu 12 Watt 36
Tabel 4.5 Perbandingan Alat Ukur dan Sensor pada Lampu 12 Watt 37
Tabel 4.6 Data Pengujian Lampu 18 Watt 39
Tabel 4.7 Perbandingan Alat Ukur dan Sensor pada Lampu 18 Watt 39
Tabel 4.8 Data Pengujian Lampu 24 Watt 41
Tabel 4.9 Perbandingan Alat Ukur dan Sensor pada Lampu 24 Watt 41
Tabel 4.10 Data Pengujian Lampu 27 Watt 43
Tabel 4.11 Perbandingan Alat Ukur dan Sensor pada Lampu 27 Watt 44
Tabel 4.12 Data Pengujian Lampu 30 Watt 45
Tabel 4.13 Perbandingan Alat Ukur dan Sensor pada Lampu 30 Watt 46
Tabel 4.14 Hasil Pengujian Jarak Efektif Komunikasi Antar Fiting Lampu 67

DAFTAR GAMBAR

Gambar 2.1 Ilustrasi Topologi *Star* 4
Gambar 2.2 Ilustrasi Topologi *Tree* 5
Gambar 2.3 Logo Versi Android 8
Gambar 2.4 NodeMCU ESP8266 9
Gambar 2.5 Sensor PZEM-004T 10
Gambar 2.6 *Solid State Relay* 1 *Chanel* 11
Gambar 2.7 Modul RTC DS3231 12
Gambar 2.8 OLED 0.91 14
Gambar 2.9 *Power Supply* 15
Gambar 2.10 Modul SD Card 16
Gambar 2.11 Logo MIT App Inventor 17
Gambar 3.1 *Flowchart* Pembuatan Proyek Akhir 18
Gambar 3.2 *Flowchart* Rancangan Algoritma Kerja Alat 20
Gambar 3.3 Diagram Blok *Hardware* Elektrik 22
Gambar 3.4 Skematik Setiap Fiting Lampu 23
Gambar 3.5 Sistem Jaringan Topologi *Star* 24
Gambar 3.6 Sistem Jaringan Topologi *Tree* 25
Gambar 4.1 *Hardware* Mekanik yang Telah Dimodifikasi 28
Gambar 4.2 *Hardware* Elektrik yang Telah Dirakit 28
Gambar 4.3 Rangkaian Catu Daya 30
Gambar 4.4 Hasil pengujian NodeMCU ESP8266 31
Gambar 4.5 *Connector Probe Kew Kyoritsu Power Quality Analyzer* 34
Gambar 4.6 Klem Meter 5A 34
Gambar 4.7 Menu *Setup* Alat Ukur 35
Gambar 4.8 Hasil Pengujian Lampu 12 Watt 35
Gambar 4.9 Hasil Pengujian Lampu 18 Watt 38
Gambar 4.10 Hasil Pengujian Lampu 24 Watt 40
Gambar 4.11 Hasil Pengujian Lampu 27 Watt 43

Gambar 4.12 Hasil Pengujian Lampu 30 Watt 45
Gambar 4.13 Skematik Pemasangan Pin pada *Real Time Clock* DS3231 48
Gambar 4.14 Hasil Pengujian *Real Time Clock* DS3231..... 49
Gambar 4.15 Skematik Mengakses OLED 0.91 50
Gambar 4.16 Karakter yang Ditampilkan pada OLED 0.91..... 51
Gambar 4.17 Tampilan Utama Aplikasi “Smart Fitting”..... 52
Gambar 4.18 Menu Aplikasi “Smart Fitting” 53
Gambar 4.19 Tampilan Menu *Setting*..... 55
Gambar 4.20 Tampilan Menu *ON/OFF*..... 55
Gambar 4.21 Menghidupkan Lampu *Mode* Normal..... 56
Gambar 4.22 Mematikan Lampu *Mode* Normal..... 56
Gambar 4.23 Pengujian Menu *Monitoring* 57
Gambar 4.24 Tampilan Menu *Timer*..... 58
Gambar 4.25 Skematik Pemasangan Pin pada Modul SD Card 59
Gambar 4.26 Pengujian Untuk Menyimpan Data *Logger* 60
Gambar 4.27 Pengujian Merekam Waktu dan Biaya Pemakaian 61
Gambar 4.28 Hasil Penyimpanan Data *Logger*..... 62
Gambar 4.29 Hasil Pengujian Kontrol Lampu 1..... 63
Gambar 4.30 Hasil Pengujian Kontrol Lampu 2..... 63
Gambar 4.31 Hasil Pengujian *Monitoring* Lampu 1..... 64
Gambar 4.32 Hasil Pengujian *Monitoring* Lampu 2..... 64
Gambar 4.33 Hasil Pengujian Monitotring Lampu Secara Bersamaan 65
Gambar 4.34 Ujicoba Pengaturan *timer*..... 66
Gambar 4.35 Ujicoba *Timer* Menghidupkan dan Mematikan Lampu 66
Gambar 4.36 Topologi *Tree* Lampu 1..... 68
Gambar 4.37 Topologi *Tree* Lampu 2..... 68

DAFTAR LAMPIRAN

LAMPIRAN 1: DAFTAR RIWAYAT HIDUP

LAMPIRAN 2: PROGRAM NODEMCU ESP8266

LAMPIRAN 3: PROGRAM APLIKASI

LAMPIRAN 4: HASIL PENYIMPANAN DATA *LOGGER*

BAB I
PENDAHULUAN

1.1. Latar Belakang Masalah

Penghematan energi listrik telah menjadi salah satu topik yang sering dibicarakan di kalangan masyarakat pada umumnya. Hal ini dikarenakan meningkatnya jumlah penggunaan peralatan elektronik yang semakin canggih dan beragam. Dengan meningkatnya penggunaan ini juga akan berdampak pada besarnya tagihan listrik yang harus dibayar. Pengguanan piranti elektronika yang dikontrol secara sistematis dan terstruktur sehingga diaplikasikan pada peralatan rumah tangga akan berdampak pada pengurangan konsumsi energi listrik yang berlebih [1].

Pada saat ini kondisi pengontrolan instalasi listrik di sebagian besar rumah masyarakat umumnya masih bersifat konvensional yakni pengontrolan yang masih bersifat manual dan harus mendatangi tempat pemutus atau saklar perangkat elektronika yang digunakan [2], sehingga dalam pengendalian penghematan daya yang dipakai masih sangat terbatas. Penggunaan daya yang menyebabkan pemborosan sehingga biaya yang harus dikeluarkan meningkat.

Oleh karena itu perlu dilakukan inovasi dan kreasi terhadap peralatan elektronik yang banyak digunakan pada peralatan rumah tangga agar dapat dikontrol pemakaiannya sehingga berdampak pada penghamatan energi serta biaya yang dikeluarkan. Hal ini juga dapat mendukung pengembangan terhadap konsep rumah cerdas (*Smart Home*) yang mempunyai banyak kelebihan terutama untuk meningkatkan efesiensi dan kenyamanan penghuninya.

Pada proyek akhir sebelumnya yang berjudul “Stop Kontak Pintar Berbasis Android” yang mampu me*monitoring* penggunaan daya listrik, biaya penggunaan dan mengontrol stop kontak melalui Android. kekurangan proyek akhir ini hanya bisa digunakan dengan jarak maksimal 10 meter serta masing-masing stop kontak tidak bisa berbagi informasi satu sama lain sehingga harus melakukan pengecekan dengan mengoneksikan ke masing-masing stop kontak[3].

Dalam memaksimalkan pengaplikasian perangkat elektronik bersistem *smart home*, penulis memiliki ide dengan membuat proyek akhir berjudul "Kontrol dan *Monitoring* Fiting Lampu Berbasis Android" yang mampu *monitoring* daya, biaya pemakaian dan mampu beroperasi dengan kontrol *timer* serta terintegrasi dengan Android melalui jaringan *wireless*. Alat ini juga dapat mengontrol fiting lampu yang berada diluar jangkauan Android melalui fiting lampu yang terkoneksi pada Android.

1.2. Rumusan dan Batasan Masalah

1.2.1. Rumusan Masalah

Adapun rumusan masalah yang diangkat berdasarkan latar belakang proyek akhir ini antara lain :

1. Bagaimana menampilkan tegangan, arus dan daya nyata setiap fiting lampu pada Android?
2. Bagaimana mengontrol *ON/OFF* lampu menggunakan *timer*?
3. Bagaimana cara mengetahui waktu pemakaian lampu?
4. Bagaimana cara menampilkan biaya yang digunakan setiap fiting lampu dan dikirimkan ke Android?
5. Bagaimana cara mengontrol fiting lampu yang berada diluar jangkauan Android melalui fiting lampu yang terkoneksi pada Android?

1.2.2. Batasan Masalah

Adapun batasan masalah pada proyek akhir ini adalah sebagai berikut:

1. Jarak efektif komunikasi antara fiting lampu dan Android kurang dari 30 meter tanpa halangan dan 15 meter dengan halangan.
2. Jarak efektif komunikasi antar fiting lampu 1 dan fiting lampu 2 kurang dari 15 meter tanpa halangan dan 10 meter dengan halangan.
3. Pengujian menggunakan lampu dengan daya maksimal 30 watt.
4. Data total biaya dan waktu pemakaian pada hari ini dan satu hari sebelumnya disimpan pada SD Card dan ditampilkan pada Android.
5. Fiting lampu yang digunakan sebanyak dua buah.

1.3. Tujuan Proyek Akhir

Adapun tujuan penulisan dalam penyusunan proyek akhir ini diantaranya adalah :

1. Menampilkan daya nyata setiap fiting lampu pada Android
2. Mengontrol *ON/OFF* lampu dengan *timer*.
3. Mengetahui waktu pemakaian lampu.
4. Menampilkan nilai tegangan, arus, daya dan biaya setiap fiting lampu pada Android.
5. Mengontrol fiting lampu yang berada diluar jangkauan Android melalui fiting lampu yang terkoneksi pada Android.

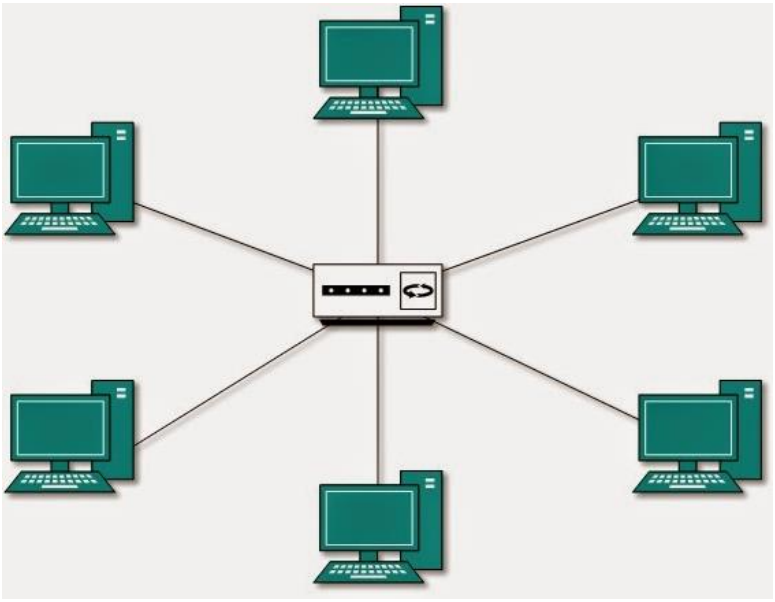
BAB II
TEORI DASAR

2.1. Topologi Jaringan

Topologi jaringan merupakan bentuk atau struktur jaringan yang menghubungkan perangkat jaringan satu dengan yang lain melului media kabel ataupun nirkabel (tanpa kabel) sehingga membentuk sebuah jaringan. Ada beberapa jenis topologi jaringan diantaranya adalah topologi *peer to peer*, *linier*, *bus*, *star*, *ring*, *mesh*, *tree* dan *hybrid*. Untuk topologi jaringan yang digunakan pada proyek akhir ini adalah topologi *star* dan *tree* [4].

2.1.1. Topologi *Star*

Sesuai dengan namanya topologi *star* atau topologi bintang membentuk jaringan yang menyerupai bentuk bintang. Topologi *star* merupakan suatu bentuk hubungan antar perangkat dimana setiap perangkat memiliki jalur komunikasi tersendiri yang terhubung ke perangkat pusat [5]. Berikut adalah ilustrasi topologi *star* ditunjukan pada Gambar 2.1 [4].

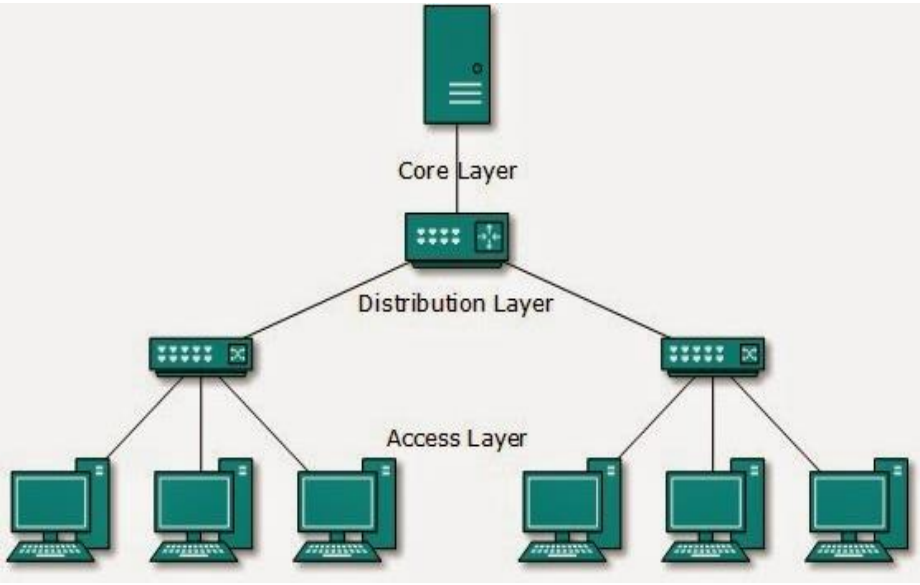


Gambar 2.1 Ilustrasi Topologi *Star*

2.1.2. Topologi *Tree*

Sesuai dengan namanya topologi *tree* atau topologi pohon membentuk jaringan yang menyerupai bentuk pohon dengan cabang dan ranting. Topologi *tree* merupakan gabungan antara topologi *bus* dan topologi *star*, yang dimana topologi *star* sebagai cabang yang dihubungkan dengan topologi *bus* yang seperti ranting [6].

Topologi *tree* juga terdapat hierarki atau tingkat jaringan, yang dimana cabang memiliki hierarki yang lebih tinggi dari ranting sehingga hierarki yang lebih tinggi dapat mempengaruhi dan mengontrol jaringan yang terdapat dibawahnya [6]. Berikut adalah ilustrasi topologi *tree* ditunjukan pada Gambar 2.2 [4].



Gambar 2.2 Ilustrasi Topologi *Tree*

2.2. Daya Listrik

Temy Nusa dalam jurnal Teknik Elektro dam Komputer Juli 2018, Universitas Sam Ratulangi yang berjudul Sistem *Monitoring* Konsumsi Energi Listrik Secara *Real Time* Berbabasis Mikrokontroler mengatakan bahwa daya listrik merupakan laju hantaran energi listrik dalam rangkaian listrik. Daya listrik terbagi menjadi tiga jenis, yaitu daya aktif, daya reaktif dan daya semu [7].

Daya aktif atau daya nyata (P) adalah daya sebenarnya yang dihamburkan atau dipakai oleh beban. Satuan daya aktif adalah Joule/detik atau watt. Daya aktif dapat dihitung dengan persamaan 2.1 [7].

$$P = V \times I \times \cos \varphi \quad \text{.....} \quad (2.1)$$

Keterangan : V = Tegangan, *volt*
I = Arus, *ampere*
φ = Sudut phi, derajat

Daya reaktif (Q) merupakan daya yang diperlukan untuk pembentukan medan magnet. Daya reaktif juga dipahami daya yang tidak dihamburkan oleh beban atau dengan kata lain merupakan daya yang diserap namun dikembalikan ke sumbernya. Satuan daya reaktif adalah VAR (*Voltampere – reactive*). Daya reaktif dapat dihitung dengan persamaan 2.2 [7].

$$Q = V \times I \times \sin \varphi \quad \text{.....} \quad (2.2)$$

Keterangan : V = Tegangan, *volt*
I = Arus, *ampere*
φ = Sudut phi, derajat

Daya tampak atau daya semu (S) merupakan hasil dari penjumlahan trigonometri daya aktif dan reaktif. Satuan daya nyata adalah VA (Voltampere). Daya nyata dapat dihitung dengan persamaan 2.3 [7].

$$S = V \times I \quad \text{.....} \quad (2.3)$$

Keterangan : V = Tegangan, *volt*
I = Arus, *ampere*

Daya adalah kecepatan energi dikeluarkan, sehingga energi yang digunakan oleh alat listrik merupakan laju penggunaan daya dikalikan dengan

waktu selama alat tersebut digunakan. Jika diukur satu jam pemakaian, maka dapat dihitung dengan menggunakan persamaan 2.4

$$\text{Daya (per-jam)} = \frac{V \, x \, I}{1000} \qquad \text{..... (2.4)}$$

Setelah melakukan pengumpulan data secara langsung dengan karyawan Perusahaan Listrik Negara, penulis mendapat info mengenai biaya pemakaian listrik per-KWh peruntukan rumah tangga yaitu Rp. 1.467,28 untuk konsumsi daya dari 901 VA sampai 53000 VA. Untuk perhitungan biaya listrik pada setiap beban dengan satuan jam dapat dihitung dengan menggunakan persamaan 2.5, sedangkan untuk perhitungan biaya listrik pada setiap beban dengan satuan detik dapat dihitung dengan menggunakan persamaan 2.6 atau persamaan 2.7.

$$\text{Biaya per-jam} = \text{Daya (per-jam)} \; x \; 1.467{,}28 \qquad \text{..... (2.5)}$$

$$\text{Biaya per-detik} = \frac{V \, x \, I}{36000} \text{ x } 1{,}46728 \qquad \text{..... (2.6)}$$

$$\text{Biaya per-detik} = \frac{Biaya \; per \; jam}{3600} \qquad \text{..... (2.7)}$$

Keterangan : V = Tegangan, *volt*

I = Arus, *ampere*

2.3. Android

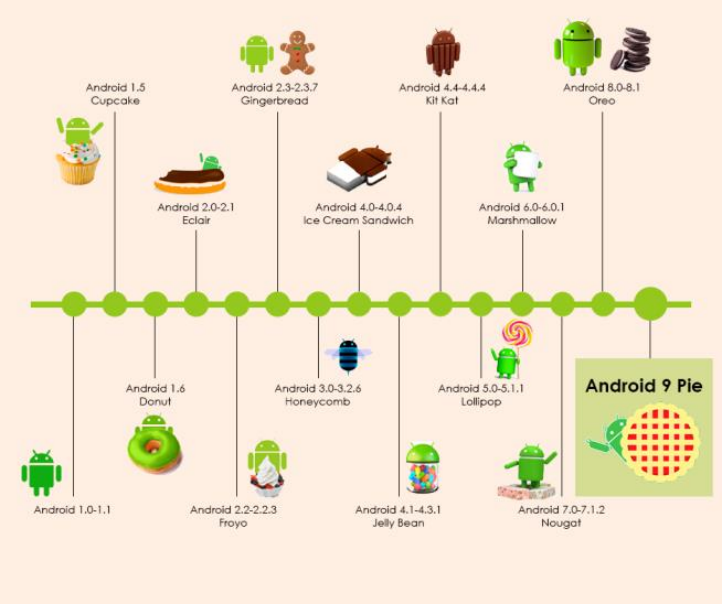
Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat seluler layar sentuh seperti telepon pintar dan komputer tablet. Android bersifat *open source* yang artinya pihak Google memperbolehkan dan membebaskan semua pihak untuk mengembangkan sistem Android [8].

Android, Inc. didirikan di Palo Alto, California, pada bulan oktober 2003 oleh Andy Rubin, Rich Miner, Nick Sears dan Chris White untuk mengembangkan perangkat seluler pintar yang lebih sadar akan lokasi dan preferensi penggunanya. Kemudian dilakukanlah pengembangan Android yang bertujuan untuk mengembangkan sebuah sistem operasi canggih yang

diperuntukkan bagi kamera digital, tetapi disadari bahwa pasar untuk perangkat tersebut tidak cukup besar dan pengembangan Android lalu dialihkan bagi pasar telepon pintar untuk menyaingi Symbian dan Windows Mobile [9].

Pada tanggal 17 Agustus 2005 Google mengakuisisi Android Inc. sehingga menjadikannya sebagai anak perusahaan yang sepenuhnya dimiliki oleh Google. Pendiri Android Inc. seperti Andy Rubin, Rich Miner dan Chris White tetap bekerja di perusahaan setelah diakuisisi oleh Google. Google mulai mengembangkan *platform* perangkat selular dengan menggunakan *kernel* Linux yang akan dipasarkan kepada produsen perangkat seluler dan operator nirkabel dengan janji bahwa mereka menyediakan sistem yang fleksibel dan bisa diperbarui [9].

Sejak tahun 2008 hingga sekarang Android dikembangkan secara bertahap oleh Google dan Open Handset Alliance (OHA) telah melakukan sejumlah pembaruan untuk meningkatkan kinerja sistem operasi, menambahkan fitur baru dan memperbaiki *bug* yang terdapat pada versi sebelumnya. Setiap versi utama yang dirilis dinamakan secara alfabetis berdasarkan nama-nama makanan pencuci mulut, kecuali Android 1.0 dan Android 1.1 [9]. Berikut logo versi Android ditunjukan pada Gambar 2.3 [10].



Gambar 2.3 Logo Versi Android

2.4. NodeMCU ESP8266

NodeMCU adalah platform *Internet of Thing* (IoT) yang bersifat *open source*. NodeMCU terdiri dari perangkat keras berupa *System On Chip Wi-Fi* ESP8266 dari ESP8266 buatan ESPressif System. NodeMCU ESP8266 menggunakan ESP8266 dengan seri ESP-12E (ESP8266MOD) dengan dilengkapi *driver* CH340G. Pengemabangan penggunaan NodeMCU ESP8266 dapat menggunakan bahasa pemrograman Lua atau dengan menggunakan *sktech* atau program dengan *software* Arduino IDE [11]. Berikut NodeMCU ESP8266 ditunjukan pada Gambar 2.4 [12].

Gambar 2.4 NodeMCU ESP8266

Spesifikasi dari NodeMCU ESP8266 dapat dilihat pada Tabel 2.1 [12] berikut.

Tabel 2.1 Spesifikasi dari NodeMCU ESP8266

No.	**Spesifikasi**	**Keterangan**
1.	Tegangan Operasi	3,3V
2.	Konsumsi Arus	10uA~170Ma
3.	*Wi-fi Direct (Peer to Peer)*	Soft-AP
4.	*GPIO*	17 (multiplexing dengan fungsi lain)
5.	*Processor*	Tensilica L106 32-bit
6.	*Processor Speed*	80~160MHz
7.	802.11 *Support*	b/g/n
8.	*Flash Memory Attachable*	16MB max (512K normal)
9.	*RAM*	32K+80K

2.5. Sensor PZEM-004T

Sensor PZEM-004T merupakan sensor yang dapat mengukur arus, tegangan dan daya pada listrik AC (*Alternatif Current*) yang keluarannya akan dikeluarkan melalui komunikasi serial. Pada dasarnya Sensor PZEM-004T tidak mampu membaca Arus dengan ketelitian mili ampere [13]. Berikut Sensor PZEM-004T ditunjukan pada Gambar 2.5 [14].

Gambar 2.5 Sensor PZEM-004T

Spesifikasi dari Sensor PZEM-004T dapat dilihat pada Tabel 2.2 [14] berikut.

Tabel 2.2 Spesifikasi Sensor PZEM-004T

No.	Spesifikasi	Keterangan
1.	Tegangan Kerja	80~260VAC
2.	Pengukuran Arus	0-100A
3.	Nilai Daya	22000W
4.	Frekuensi Operasi	45-65Hz
5.	Akurasi Pengukuran	1.0 *grade*

Untuk mengakses Sensor PZEM-004T harus menggunakan komunikasi serial dengan konfigurasi pin yang ditunjukan pada Tabel 2.3 [14] berikut.

Tabel 2.3 Konfigurasi pin Sensor PZEM-004T

No.	Pin	Keterangan
1.	5V	5V
2.	RX	Dihubungkan dengan pin serial *receiver*
3.	TX	Dihubungkan dengan pin serial *transmitter*
4.	GND	GND

2.6. Modul Solid State Relay (SSR)

Modul *solid state relay* adalah sebuah modul yang terdiri komponen semikonduktor yang bekerja menggunakan arus dan semikonduktor *solid state* untuk menangani input terhadap *output* serta memerankan fungsinya sebagai saklar. Saat kondisi *open solid state relay* memiliki fungsi pemisahan *input* dan *output* seperti halnya sebuah saklar dengan hambatan yang sangat tinggi, namun pada saat kondisi *close solid state relay* mampu mengalirkan arus dengan arus yang sangat besar dengan hambatan yang sangat kecil. *Solid state relay* bersifat aktif *low* [15]. Sehingga saat diberikan input *high* pada pin CH1 pada *Solid State Relay* maka kondisi relay akan *OFF*. Berikut Modul *Solid State Relay* ditunjukan pada Gambar 2.6.

Gambar 2.6 *Solid State Relay* 1 *Chanel*

Pada proyek akhir ini digunakan *solid state relay chanel* 1. Berikut spesifikasi dari *solid state relay* dapat dilihat pada Tabel 2.4 [16].

Tabel 2.4 Spesifikasi *Solid State Relay*

No.	Spesifikasi	Keterangan
1.	Tegangan *Input*	5V
2.	Beban Keluaran	100~240V dengan arus 2A

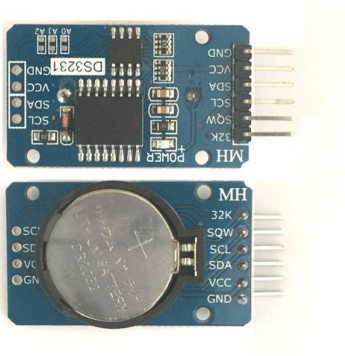
Untuk mengkases *solid state relay* dapat dilakukan dengan memberikan *input* berdasarkan konfigurasi pin *solid state relay* pada Tabel 2.5 [16] berikut.

Tabel 2.5 Konfigurasi pin *Solid State Relay*

No.	Pin	Keterangan
1.	DC+	5V
2.	DC-	GND
3.	CH1	Dihubungkan dengan pin *Digital Output*

2.7. Modul RTC DS3231

Pada proyek akhir ini penulis menggunakan RTC DS3231. Berikut RTC DS3231 dapat dilihat pada Gambar 2.7 [17].



Gambar 2.7 Modul RTC DS3231

Modul RTC DS3231 adalah modul *Real Time Clock* yang berfungsi untuk mencacah waktu seperti detik, menit, jam, tanggal, bulan, tahun serta hari

dalam seminggu dengan akurasi yang tinggi dan *valid* sampai tahun 2100. RTC DS3231 dilengkapi dengan EEPROM *external* AT24C23 sebesar 32kB, baterai yang berguna sebagai *back up* catu daya sehingga waktu akan tetap *update* walaupun catu daya utama mati, dan sensor suhu untuk membaca temperature disekitar [18].

Spesifikasi dari Modul RTC DS3231 dapat dilihat pada Tabel 2.6 [19] berikut.

Tabel 2.6 Spesifikasi dari Modul RTC DS3231

No.	Spesifikasi	Keterangan
1.	Tegangan Operasi	3,3 – 5,55V
2.	*Chip* Jam	Chip *clock* presisi tinggi DS3231
3.	Akurasi	± 3.5ppm dari -40 ° C sampai +85 ° C
4.	Ketepatan Jam	Kisaran 0 – 40, akurasi 2ppm, kesalahan sekitar 1 menit
5.	*Chip* Memori	AT24C32 (kapasitas penyimpanan 32K)
6.	Baterai *Backup*	CR2032 3V
7.	Panjang	33mm
8.	Lebar	22mm
9.	Berat	8 gram

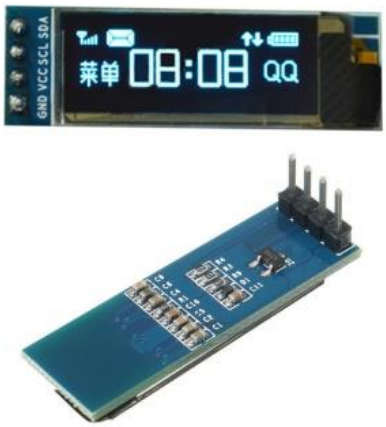
Konfigurasi pin Modul RTC DS3231 dapat dilihat pada Tabel 2.7 [19] berikut.

Tabel 2.7 Konfigurasi pin Modul RTC

No.	Pin	Keterangan
1.	VCC	3,3 – 5V
2.	GND	GND
3.	SDA	Dihubungkan dengan pin *serial data* (SDA)
4.	SCL	Dihubungkan dengan pin *serial clock* (SCL)
5.	SQW	Tidak dihubungkan
6.	32K	Tidak dihubungkan

2.8. OLED 0.91

OLED adalah kepanjangan dari *Organic Lamp Emitting Diode*. OLED memiliki kontras layar yang tinggi sehingga tidak memerlukan cahaya latar. OLED 0.91 merupakan modul tampilan grafis monokrom dengan *display* grapik dengan ukuran 0.91 inci dan resolusi 128x32 pixel dengan resolusi tinggi. OLED 0.91 inci bukan hanya dapat menampilkan karakter, tetapi juga dapat menampilkan gambar dan animasi. OLED 0.91 menggunakan komunikasi I2C [20]. Berikut Modul OLED 0.91 128x32 ditunjukan pada Gambar 2.8 [20].



Gambar 2.8 OLED 0.91

Untuk mengetahui spesifikasi OLED 0.91 dapat dilihat pada Tabel 2.8 [20] berikut.

Tabel 2.8 Spesifikasi dari OLED 0.91

No.	Spesifikasi	Keterangan
1.	Tegangan Operasi	3,3 – 5V
2.	*Driver IC*	SSD1306
3.	Komunikasi	I2C
4.	Resolusi	128x32 pixel
5.	*Display Colour*	Putih/Biru

Untuk mengakses OLED 0.91 dapat dilakukan dengan melihat konfigurasi pin yang dapat dilihat pada Tabel 2.9 [20] berikut.

Tabel 2.9 Konfigurasi pin OLED 0.91

No.	Pin	Keterangan
1.	GND	GND
2.	VCC	3,3 – 5V
3.	SCL	Dihubungkan dengan pin *serial clock* (SCL)
4.	SDA	Dihubungkan dengan pin *serial data* (SDA)

2.9. Power Supply

Power Supply atau catu daya merupakan rangkaian elektronika yang digunakan sebagai pengubah listrik tegangan AC menjadi listrik tegangan DC sehingga listrik tegangan DC dapat digunakan sebagai sumber energi listrik *hardware* elektronika [21].

Power supply terdiri dari beberapa komponen utama yaitu *transformator* atau trafo, diode, kapasitor jenis elco dan transistor. *Transformator* atau trafo berfungsi sebagai penurun tegangan misalnya tegangan AC yang awalnya sebesar 220V menjadi keluaran tegangan AC sebesar 12 V, kemudian keluaran dari trafo masuk ke rangkaian *diode bridge* yang berguna sebagai penyearah tegangan AC menjadi tegangan DC. Kapasitor jenis elco bertugas sebagai penyaring tegangan *ripple* yang masih bocor, sedangkan transistor berfungsi sebagai penstabil tegangan. *Output* dari rangkaian *power supply* akan menjadi sumber *hardware* elektronika [21]. Berikut *power supply* ditunjukan pada Gambar 2.9 [22].

Gambar 2.9 *Power Supply*

2.10. Modul SD Card

Modul SD Card merupakan modul untuk mengakses SD Card baik untuk pembacaan ataupun penulisan data. Modul SD Card cocok digunakan pada aplikasi yang membutuhkan data *logging* seperti *monitoring* daya listrik, absensi dan lain-lain. Modul SD Card menggunakan sistem antarmuka SPI (*Serial Parallel Interface*) [23]. Berikut Modul SD Card ditunjukan pada Gambar 2.10 [24].

Gambar 2.10 Modul SD Card

Untuk mengakses modul SD Card harus menggunakan komunikasi SPI dengan konfigurasi pin yang ditunjukan pada Tabel 2.10 [24] berikut.

Tabel 2.10 Konfigurasi pin Sensor PZEM-004T

No.	Pin	Keterangan
1.	GND	GND
2.	VCC	3,3 – 5V
3.	MISO	Dihubungkan dengan pin MISO
4.	MOSI	Dihubungkan dengan pin MOSI
5.	SCK	Dihubungkan dengan pin SCK
6.	CS	Dihubungkan dengan pin CS

2.11. MIT App Inventor

MIT App Inventor adalah alat pengembangan yang fungsinya untuk membuat aplikasi yang dapat di-*install* pada Android. MIT App Inventor ini diciptakan oleh MIT (Massachusetts Institute of Technology) dengan tujuan untuk

memudahkan pembuatan aplikasi Android. MIT menciptakan alat pengembangan yang mudah digunakan oleh siapa saja dengan menggunakan metode *click* dan *drag* untuk desain tampilannya dan metode penyusunan blok untuk algoritma programnya [25]. MIT App Inventor dapat diakses melalui *browser* internet. Berikut logo MIT App Inventor ditunjukan pada Gambar 2.11 [26].

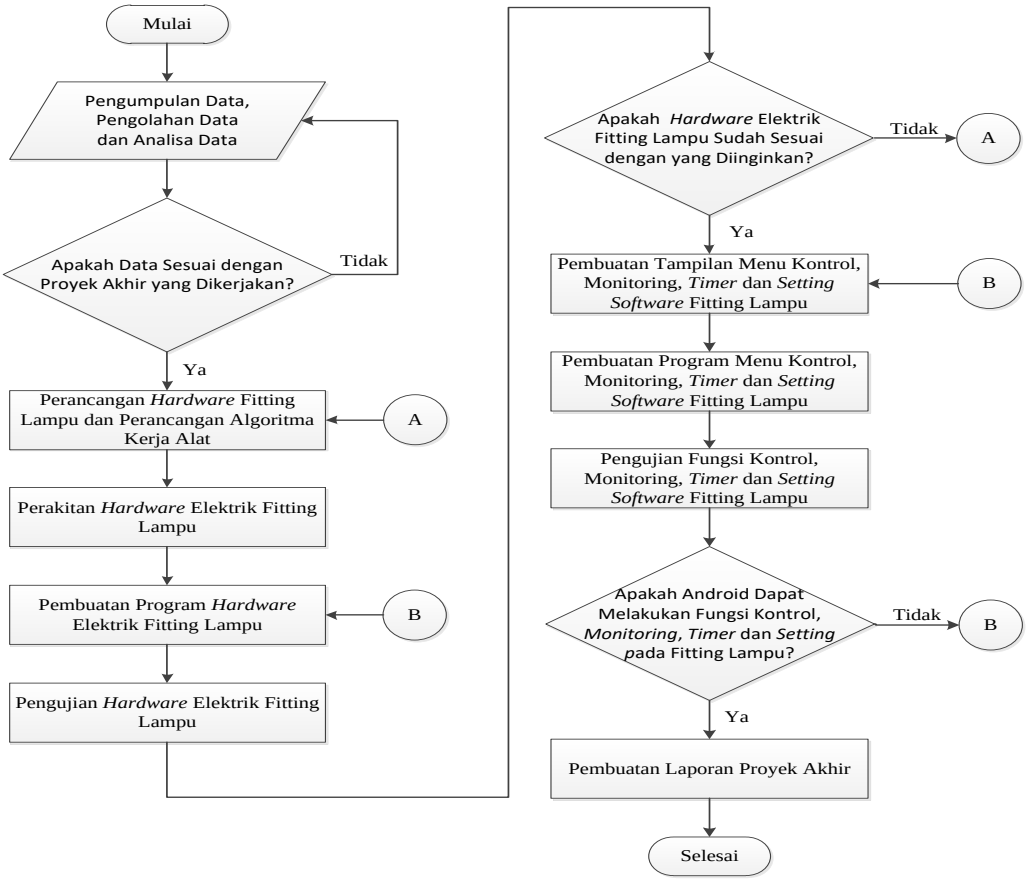
Gambar 2.11 Logo MIT App Inventor

BAB III
METODE PENELITIAN

Dalam pembuatan proyek akhir dengan judul "Kontrol dan *Monitoring* Fiting Lampu Berbasis Android" ini dilakukan beberapa metode penelitian yang bertujuan untuk mempermudah dalam penyelesaian proyek akhir. Adapun metode penelitian yang diterapkan dalam pembuatan proyek akhir seperti berikut:

3.1. Metode Pelaksanaan Pembuatan Proyek Akhir

Proses pembuatan proyek akhir dari mulai sampai selesainya proyek akhir dijelaskan pada *flowchart* pembuatan proyek akhir. *Flowchart* pembuatan proyek ditunjukan pada Gambar 3.1 berikut.



Gambar 3.1 *Flowchart* Pembuatan Proyek Akhir

Gambar 3.1 Merupakan *flowchart* pembuatan proyek akhir yang menjelaskan alur dari proses pengerjaan proyek akhir. Hal yang pertama dilakukan proses pembuatan proses akhir yaitu pengumpulan data yang diperlukan dalam menyelesaikan proyek akhir. Proses pengumpulan data dapat dilakukan secara langsung (primer) dan secara tidak langsung (sekunder). Maksud dari pengumpulan data secara langsung merupakan pengumpulan data yang diperoleh dari konsultasi dengan dosen pembimbing sedangkan untuk pengumpulan data secara tidak langsung (sekunder) merupakan data yang diperoleh dari membaca makalah proyek akhir tahun sebelumnya yang berjudul "Stop Kontak Pintar Berbasis Android" dan dari referensi – referensi dari penelitian yang masih berhubungan dengan proyek akhir yang ingin dibuat. Data - data yang didapatkan dari proses pengumpulan data selanjutnya diolah dan dianalisa. Data yang telah diolah dan dianalisa dikumpulkan lalu dipilih untuk menjadi referensi dan acuan dalam pembuatan proyek akhir.

Perancangan *hardware* fiting lampu dan perancangan algoritma kerja alat dibuat agar *hardware* dan prinsip kerja alat sesuai dengan spesifikasi proyek akhir. Proses perakitan *hardware* elektrik fiting lampu harus sesuai spesifikasi dan koneksi pin dari dari *hardware* elektrik yang kita gunakan. Pembuatan program *hardware* elektrik fiting lampu dibuat pada *software* Arduino IDE. Pembuatan program *hardware* elektrik fiting lampu harus sesuai dengan koneksi pin dan harus memenuhi spesifikasi proyek akhir yang ingin dibuat. Setelah pembuatan program *hardware* elektrik fiting lampu selesai, maka dilakukan pengujian, yang dimana tujuannya untuk mengetahui apakah *hardware* elektrik fiting lampu sudah sesuai dengan spesifikasi yang diinginkan.

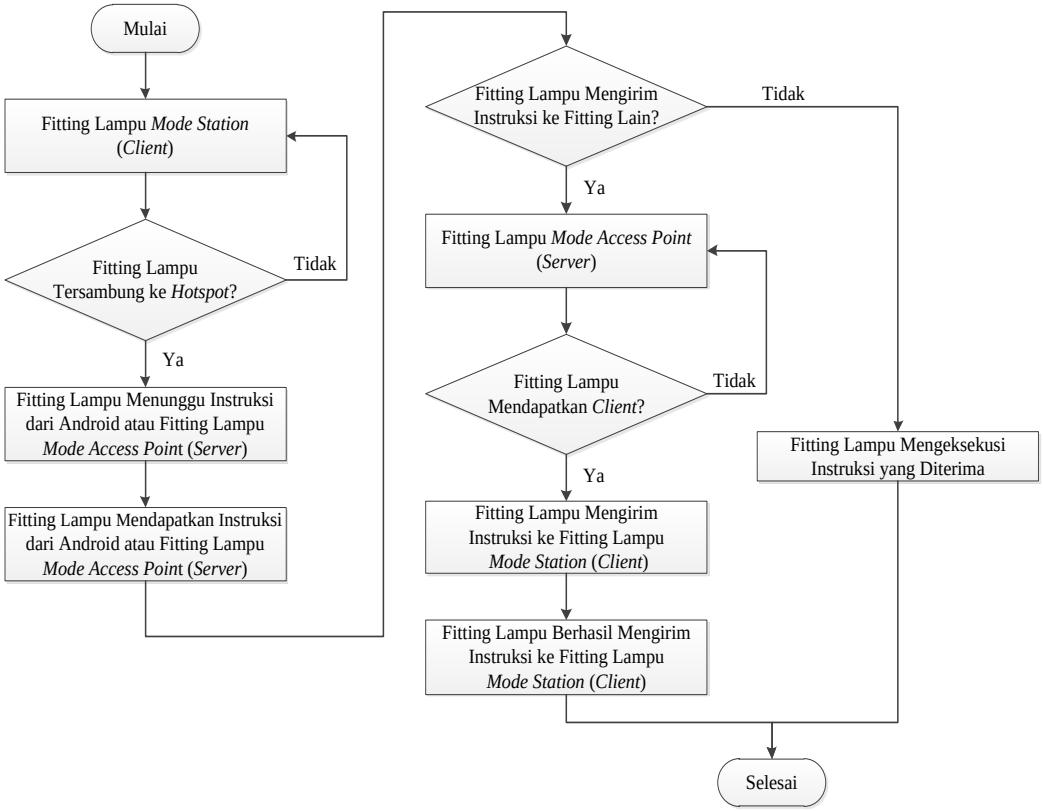
Pembuatan *software* untuk memenuhi spesifikasi dari proyek akhir dibuat menggunakan MIT App Inventor 2 yang diakses menggunakan *browser* internet. Pembuatan tampilan menu dilakukan pada bagian *designer* MIT App Inventor 2. Pembuatan tampilan menu kontrol, *monitoring*, *timer* dan *setting* harus mudah dipahami serta digunakan dan tentunya tetap memenuhi nilai fungsi dari *software* yang akan dibuat. Pembuatan program menu kontrol, *monitoring*, *timer* dan *setting software* fiting lampu dilakukan pada bagian *bloks* MIT App Inventor 2.

Pengujian fungsi kontrol, *monitoring*, *timer* dan *setting software* fiting lampu dilakukan dengan cara menginstall *software* yang telah dibuat ke Android. Pengujian fungsi dilakukan secara keseluruhan, baik dari segi kontrol, *monitoring*, *timer* dan *setting software* harus dapat bekerja sesuai dengan fungsinya.

Proses pembuatan laporan merupakan proses terakhir dalam pembuatan proyek akhir. Pembuatan laporan bertujuan merangkum keseluruhan keseluruhan yang berhubungan dengan proyek akhir, seperti latar belakang, tujuan, rumusan masalah, batasan masalah, teori dasar, metode penilitian, pembahasan serta kesimpulan dan saran.

3.2. Rancangan Algoritma Kerja *Hardware*

Rancangan algoritma kerja *hardware* merupakan rancangan dari proses cara kerja *hardware* untuk memenuhi spesifikasi yang diinginkan. Berikut adalah *flowchart* rancangan algoritma kerja *hardware* ditunjukan pada Gambar 3.2 berikut.



Gambar 3.2 *Flowchart* Rancangan Algoritma Kerja Alat

Pada Gambar 3.2 merupakan *flowchart* rancangan algoritma kerja alat yang digunakan dalam mencapai target penyelesaian sesuai dengan spesifikasi yang diinginkan. Dalam keadaan normal, maka fiting lampu berfungsi sebagai *station* (*client)*. Saat dalam *mode client* fiting lampu bertugas mencari dan menghubungkan ke *hotspot* yang sesuai dengan SSID (*Service Set Identifier*) dan kata sandi yang telah diatur sesuai dengan program yang telah dibuat. Proses pencarian akan berhenti saat fiting lampu telah terhubung ke *hotspot* yang SSID dan kata sandinya sesuai dengan program, namun jika hubungan antara fiting lampu dan *hotspot* terputus, maka fiting lampu akan melakukan pencarian *hotspot* sampai berhasil terhubung kembali ke *hotspot* yang sesuai. Ketika fiting lampu terhubung ke *hotspot*, baik itu *hotspot* Android ataupun *hotspot* fiting lampu *mode access point* (*server*), maka fiting lampu akan menunggu instruksi yang akan dikirim dari *hotspot* yang telah terhubung.

Saat fiting lampu mendapatkan instruksi maka fiting lampu akan melakukan pengerjaan sesuai dengan instruksi yang didapat. Jika fiting lampu menerima instruksi untuk mengirim instruksi ke fiting lampu lain, maka fiting lampu yang mengirim instruksi ke fiting lampu lain akan berganti *mode* yang awalnya *mode client* menjadi *mode access point* (*server*) dan koneksi ke *hotspot* Android menjadi terputus. Saat fiting lampu *mode server* telah mendapatkan *client* yang dimana *client* tersebut adalah fiting lampu *mode client*, maka fiting lampu *server* bertugas mengirimkan instruksi ke fiting lampu *client* yang tersambung sesuai dengan instruksi dari Android. Proses pengiriman ini akan terus berlangsung sampai data yang dikirimkan berhasil diterima oleh fiting lampu *mode client*. Apabila data yang dikirimkan telah diterima oleh fiting lampu *mode station*, maka fiting lampu *mode server* akan kembali berubah ke *mode client* dan kembali mencari *hotspot*, sedangkan untuk fiting lampu *mode client* yang menerima instruksi dari fiting lampu *mode server* akan menjalankan tugas sesuai dengan instruksi yang diterimanya dan kembali mencari *hotspot* yang sesuai dengan program.

Namun jika fiting lampu menerima instruksi dengan tidak melakukan pengiriman instruksi ke fiting lampu lain maka fiting lampu akan melakukan

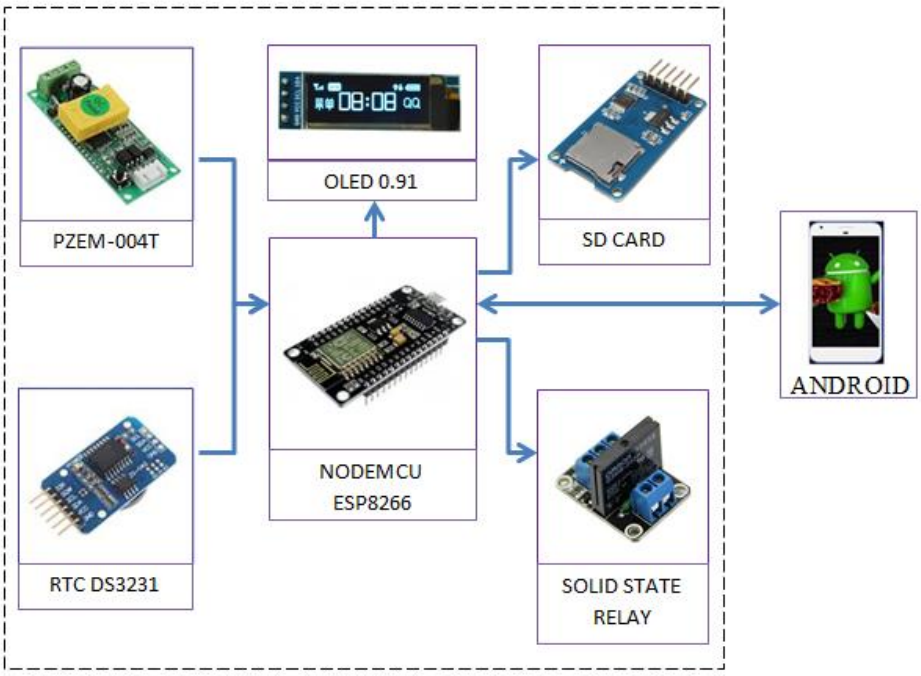
tugas sesuai dengan instruksi yang diterima dan setelah selesai melakukan tugasnya maka fiting lampu kembali menunggu instruksi. Proses akan tetap terus berlanjut dan kembali ke proses mulai.

3.3. Diagram Blok

Pada proyek akhir yang berjudul "Kontrol dan *Monitoring* Fiting Lampu Berbasis Android" mempunyai digram blok *hardware* dan diagram blok jaringan.

3.3.1. Diagram Blok *hardware*

Setiap fiting lampu yang dibuat memiliki *hardware* elektrik yang sama, sehingga setiap fiting lampu mempunyai diagram blok *hardware* setiap yang sama. Berikut diagram blok *hardware* ditunjukan pada Gambar 3.3 berikut.

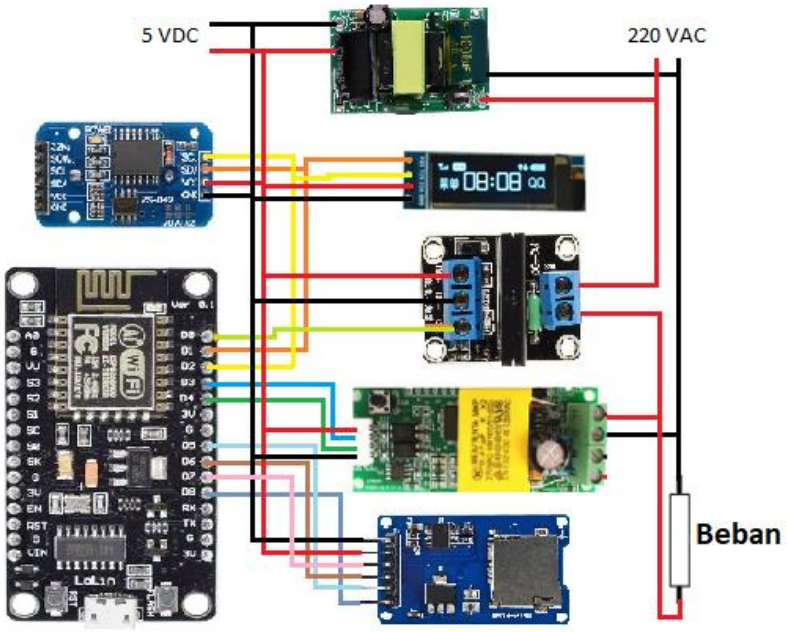


Gambar 3.3 Diagram Blok *Hardware* Elektrik

Proses pengambilan data arus, tegangan dan daya dilakukan dengan menggunakan sensor PZEM-004T. Kemudian dilakukan pengaturan waktu dari *real time clock* DS3231. Data yang didapatkan pada kedua sensor tersebut lalu

diolah oleh NodeMCU ESP8266 menjadi sebuah data *logger* sehingga arus, tegangan dan daya tersebut dapat diakses sesuai dengan waktu yang digunakan.

Dari hasil pengolahan data yang didapat selanjutnya ditampilkan pada Android dan dikirimkan pada aplikasi pada Android yang telah dibuat dengan *software* App Inventor melalui jaringan *WiFi* dengan menggunakan kontroler NodeMCU ESP8266. Android dapat menerima data serta mengirimkan perintah pada NodeMCU ESP8266 untuk men-*trigger solid state relay* sehingga dapat mematikan dan menghidupkan lampu yang digunakan. Berikut skematik setiap fiting lampu dapat ditunjukan pada Gambar 3.4 berikut.



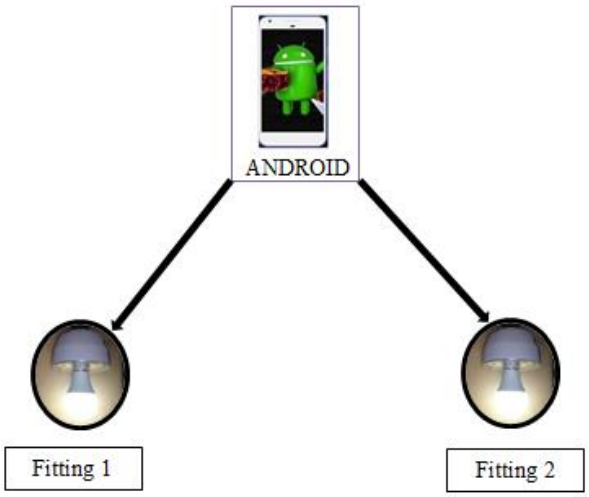
Gambar 3.4 Skematik Setiap Fiting Lampu

Pada dasarnya prinsip kerja Gambar 3.4 sama halnya dengan diagram blok pada Gambar 3.3 Untuk mengetahui pemakaian arus, tegangan dan daya pada listrik AC maka digunakan sensor PZEM-004T yang keluarannya melalui komunikasi serial RX TX akan terhubung pada pin D3 dan D4 di NodeMCU ESP8266. Selanjutnya untuk mengetahui fungsi waktu seperti hari, tanggal dan jam digunakan *real time clock* DS3231 memiliki keluaran dengan komuikasi SDA SCL yang terhubung pada pin D1 dan D2. Komunikasi SDA SCL ini juga

digunakan untuk menampilkan waktu pada OLED 0.91. Dari data waktu dan sensor yang didapat dilakukan proses pengolah data yang kemudian data tersebut disimpan pada SD Card. Modul SD Card ini terhubung dengan NodeMCU ESP8266 menggunakan komunikasi SPI yaitu MISO, MOSI, CS dan SCK yang masing-masing pin terhubung pada pin D6, D7, D8 dan D5. Kemudian untuk melakukan *switching (ON/OFF)* pada lampu maka digunakan *solid state relay.*

3.3.2. Diagram Blok jaringan

Untuk melakukan fungsi komunikasi antara fiting dengan Android dibuatlah diagram blok jaringan. Berikut diagram blok jaringan ditunjukan pada Gambar 3.5 berikut.



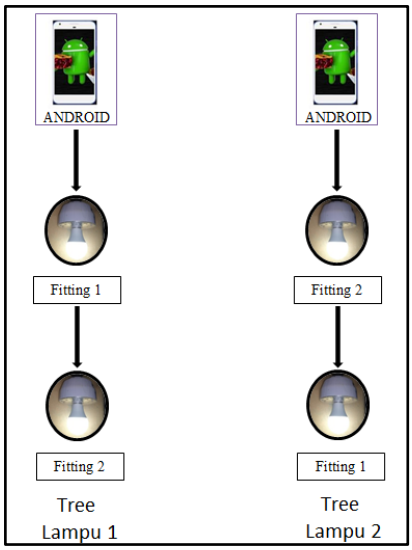
Gambar 3.5 Sistem Jaringan Topologi *Star*

Proyek akhir ini dibuat dengan dua fiting lampu yang mempunyai komponen dan diagram blok *hardware* yang sama untuk setiap fiting lampu. Sebab alat yang dibuat akan mempunyai fungsi yang sama yaitu dapat berkomunikasi satu sama lain dengan memanfaatkan kontroler NodeMCU ESP8266 menggunakan jaringan *WiFi*. Sehingga dapat membentuk sebuah topologi jaringan yang berbentuk topologi *star*. Dengan memanfaatkan kontroler NodeMCU ESP8266 maka fiting lampu yang dibuat dapat digunakan sebagai

client dan *server* sekaligus serta mendukung jaringan *peer to peer*. Android dapat menerima data dari fiting lampu serta mengirimkan data ke fiting lampu selama masih terjangkau jaringan *WiFi* yang telah dibuat dan masih termasuk jarak komunikasi yang efektik untuk komunikasi data.

Dalam keadaan normal alat yang dibuat akan membentuk sistem jaringan seperti Gambar 3.5 Android akan dapat mengontrol dan *monitoring* tegangan, arus dan daya pada setiap fiting lampu. Selain keadaan normal Android juga harus dapat mematikan semua lampu walaupun hanya salah satu fiting lampu yang terhubung dengan Android. Untuk melakukan ini penulis harus menggunakan metode lain selain topologi *star* tanpa harus mengganggu sistem topologi *star* yang telah penulis buat. Dalam hal ini penulis memutuskan untuk menggunakan topologi *tree*.

Pada dasarnya topologi *tree* mempunyai kemiripan dengan topologi *star* namun topologi *tree* mempunyai kelebihan yakni dapat mengontrol fiting lampu yang jaraknya tidak terjangkau Android, dengan syarat fiting lampu tersebut tetap terhubung dengan fiting lampu lain yang terkoneksi dengan Android. Sehingga secara tidak langsung penulis menghubungkan dua topologi jaringan yang berbeda. Berikut adalah penerapan topologi *tree* ditunjukan pada gambar 3.6 berikut.



Gambar 3.6 Sistem Jaringan Topologi *Tree*

Penerapan topologi *tree* dapat dilihat pada Gambar 3.6 yang dimana Android tetap dapat mengontrol hidup atau matinya lampu pada fiting lampu yang jauh. Pada topologi *tree* 1 jarak fiting lampu 2 tidak terjangkau oleh Android akan tetapi dengan memanfaatkan fiting lampu 1 maka Android dapat mengontrol hidup atau matinya fiting lampu 2 asalkan masih dalam cangkupan jaringan fiting lampu 1. Begitupun pada topologi *tree* 2 jarak fiting lampu 1 tidak terjangkau oleh Android maka Android dapat memanfaatkan fiting lampu 2 untuk mengontrol hidup atau matinya fiting lampu 1 asalkan masih dalam cangkupan jaringan fiting lampu 2. Sehingga secara fungsi alat yang dibuat menerapkan topologi *tree* dan tetap dapat mengontrol fiting lampu dengan topologi *star*.

BAB IV
PEMBAHASAN

Pada bab ini akan dibahas mengenai proses pembuatan dan pengujian alat serta metode yang dipakai dalam pembuatan proyek akhir yang berjudul "Kontrol dan *Monitoring* Fiting Lampu Berbasis Android", yaitu sebagai berikut:

4.1. Rancangan Alat

Untuk memastikan alat yang dibuat sesuai dengan rumusan masalah dilakukan proses perancangan alat pada *hardware* mekanik maupun *hardware* elektrik. Hal ini sangat dibutuhkan karena selain memaksimalkan fungsi dari alat ini juga agar terdapat kesesuaian antara *hardware* mekanik dan *hardware* elektrik.

4.1.1. Modifikasi *Hardware* Mekanik Fiting Lampu Berbasis Android

Dalam proses pembuatan *hardware* mekanik fiting lampu berbasis Android dilakukan dengan pembelian fiting lampu *emergency* yang dijual dipasaran. Terdapat banyak jenis fiting lampu yang dijual dipasaran. Salah satunya berbentuk tabung yang mempunyai dimensi ±467.173cm^3. Dalam penentuan *hardware* mekanik ini terdapat kriteria yang diperhatikan yakni memiliki dimensi yang cukup besar. Salah satu yang terjual dipasaran mempunyai dimensi yang lebih besar yaitu dengan dimensi ±674.805cm^3.

Dari fiting lampu yang didapat kemudian dimodifikasi dengan membuang rangkaian yang terdapat pada fiting lampu serta membersihkan bagian dalam fiting lampu sehingga bagian dalam fiting lampu tampak lebih lebar. Sehingga *hardware* elektrik yang digunakan dapat dimasukkan pada *hardware* mekanik yang telah dimodifikasi.

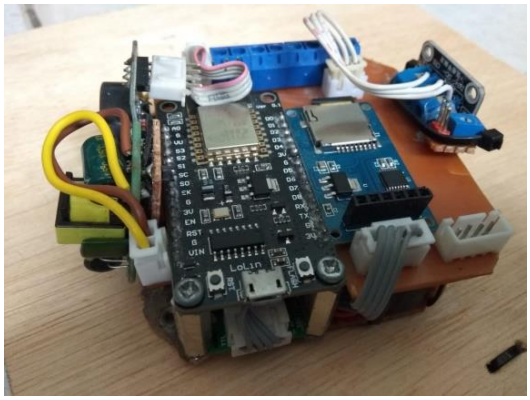
Berikut gambar *hardware* mekanik versi yang lebih besar yang dijual dipasaran dan akan digunakan pada proyek akhir ini ditunjukan pada Gambar 4.1.

Gambar 4.1 *Hardware* Mekanik yang Telah Dimodifikasi

Pada Gambar 4.1 merupakan fiting lampu yang telah dimodifikasi bagian dalamnya dan merupakan *hardware* mekanik yang akan digunakan pada proyek akhir ini.

4.1.2. *Hardware* Elektrik Fiting Lampu Berbasis Android

Proses lain yang penting pada proyek akhir ini adalah perancangan, perakitan dan percobaan terhadap *hardware* elektrik. Pada sub bab ini akan dibahas *hardware* elektrik yang akan digunakan. Berikut gambar *hardware* elektrik yang telah dirakit ditunjukan pada Gambar 4.2.



Gambar 4.2 *Hardware* Elektrik yang Telah Dirakit

Dalam merakit *hardware* elektrik penulis harus mempertimbangkan dimensi pada *hardware* mekanik. Setelah selesai *hardware* elektrik yang telah penulis rakit, lalu dimasukkan kedalam *hardware* mekanik yang telah dimodifikasi.

4.2. Rangkaian Catu Daya

Rangkaian catu daya yang digunakan dalam pembuatan proyek akhir ini mempertimbangkan pemakaian arus total pada rangakaian kontrol yang digunakan. Rangkaian catu daya ini juga mempertimbangkan masalah dimensi karena harus menyesuaikan dengan besarnya dimensi fiting lampu yang digunakan.

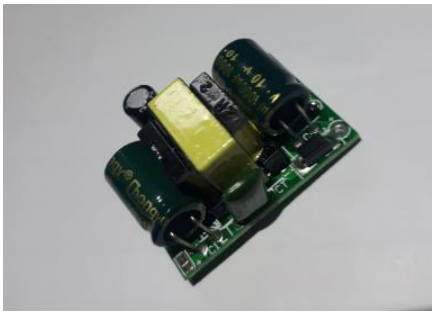
Dalam pembuatan rangkaian catu daya pada proyek akhir ini diputuskan untuk membeli rangkaian catu daya yang sudah jadi dengan memperhitungkan dengan segala kebutuhan rangkaian kontrol yang digunakan. Setelah dilakukan pengujian didapat kriteria yang diinginkan dan ditunjukan pada Tabel 4.1 berikut.

Tabel 4.1 Kriteria Catu Daya yang Digunakan

No.	**Kriteria**	**Alasan**
1.	Input AC~220V	Karena mendapatkan dari summber PLN AC~220V
2.	*Output* ±5V	Agar sesuai dengan catu daya yang diizinkan pada rangkaian kontrol.
3.	Arus minimum 500 mA	Agar rangkaian kontrol tidak mereset
4.	Ukuran maksimum 30 mm x 30 mm x 20 mm	Agar sesuai dengan dimensi yang disediakan
5.	Harga terjangkau	Agar alat maksimum tanpa biaya banyak.

Setelah melakukan pengamatan Tabel 4.1 maka didapatkan kriteria catu daya yang akan digunakan dengan input 220 VAC dan *output* dengan tegangan maksimal ±5V dan arus *output* minimal 500 mA. Serta dimensi rangkaian 18 cm^3.

Rangkaian catu daya yang digunakan pada proyek akhir ini dapat ditunjukan pada Gambar 4.3.



Gambar 4.3 Rangkaian Catu Daya

Berdasarkan Gambar 4.3 rangkaian catu daya ini memiliki spesifikasi tegangan input 220 VAC, tegangan *output* 5 VDC dan arus 700 mA.

4.3. Pengujian NodeMCU ESP8266

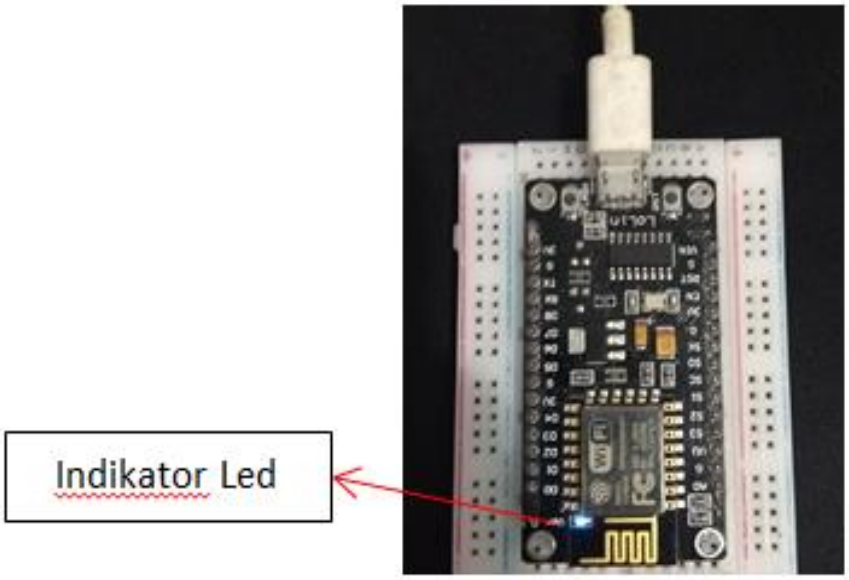
Untuk menentukan kontroler yang digunakan maka dibutuhkan kemampuan kontroler seperti Tabel 4.2

Tabel 4.2 Kriteria Kontroler yang Digunakan

No.	Kriteria	Alasan
1.	Mempunyai *Digital Input/Digital Output* (DI/DO) minimal 10	Sesuai dengan banyaknya pin komponen yang digunakan
2.	*Board* dengan ukuran kecil	Agar sesuai dengan rancangan yang dibuat .
3.	Diutamakan kontroler dengan *wireless*	Agar mempunyai kemampuan yang maksimal dengan piranti yang kecil.
4.	Mudah dalam penggunaan.	Agar tidak susah saat pembuatan proyek akhir
5	Harga terjangkau	Agar alat maksimum tanpa biaya banyak.

Dalam memilih kontroler yang akan digunakan penulis membandingkan dengan beberapa kontroler dari seperti arduino Nano, NodeMCU, X-bee pro dan ESP8266. Dari hasil yang didapat penulis memilih NodeMCU ESP8266. Hal ini karena selain mempunyai kemampuan jaringan *WiFi* NodeMCU ini dapat mendukung perangkat lain yang ingin penulis gunakan.

Untuk memastikan alat dapat berfungsi dengan baik dan dapat bekerja sesuai dengan kemampuan yang diinginkan maka dilakukan pengujian untuk menghidupkan LED dengan menggunakan *output* pada NodeMCU ESP8266. Berikut hasil pengujian yang dilakukan terdapat pada gambar 4.4.



Gambar 4.4 Hasil pengujian NodeMCU ESP8266

LED yang digunakan merupakan piranti yang sudah terdapat pada pin GPIO2 di *board* NodeMCU ESP8266. Apabila LED yang berkedip maka *board* NodeMCU ESP8266 telah di-*install* pada *software* Arduino IDE

Selain pengujian dengan menggunakan pin *output* NodeMCU ESP8266 dilakukan pengujian pada jarak yang dapat dijangkau oleh NodeMCU. Pada pengujian jarak ini dilakukan pada ruangan terbuka (tanpa penghalang) dan dalam ruangan tertutup (dengan penghalang). Untuk melihat kemampuan komunikasi

data antara NodeMCU dan Android maka dilakukan pengambilan data jarak efektif komunikasi data. Berikut hasil percobaan yang dilakukan pada Tabel 4.3.

Tabel 4.3 Hasil Pengujian Jarak Efektif Komunikasi
Antara NodeMCU dan Android

No	Jarak	Tanpa Penghalang	Dengan Penghalang
1.	5 Meter	Ya	Ya
2.	10 Meter	Ya	Ya
3.	15 Meter	Ya	Ya
4.	20 Meter	Ya	Tidak
5.	25 Meter	Ya	Tidak
6.	30 Meter	Ya	Tidak

Percobaan dilakukan ditempat terbuka dan ruangan tertutup. Sangat jelas bahwa apabila pada tempat terbuka akan dapat melakukan komunikasi lebih mudah dibandingkan dengan menggunakan penghalang. Perlu diperhatikan bahwa posisi fiting lampu juga sangat mempengaruhi proses pengiriman data. Apabila pada satu garis lurus maka untuk mengontrol fiting lampu akan lebih jauh daripada fiting lampu membentuk cabang.

4.4. Pengujian Sensor PZEM-004T

Sensor ini digunakan untuk mengukur besarnya nilai tegangan, arus dan daya pada beban yang digunakan. Pada pengujian sensor ini yang dipakai adalah tipe sensor PZEM-004T V2.0 dengan *range* 0-100 A. Untuk mengolah data dari hasil pengukuran menggunakan sensor dikirimkan dan diolah oleh NodeMCU ESP8266 yang menggunakan komunikasi serial RX TX.

Modul PZEM-004T dibundel dengan kumparan trafo arus yang mempunyai diameter 33mm. Bagian utama dari modul PZEM-004T adalah chip SD3004 dari SDIC Microelectronics Co., Ltd. Modul ini dilengkapi dengan antarmuka komunikasi data serial TTL melalui *port serial* dapat dibaca dan mengatur parameter yang relevan [3].

Pada pengujian sensor dengan cara melakukan pengukuran tegangan, arus dan daya pada lampu AC. Sensor ini kemudian dihubungkan dan diprogram menggunakan *board* NodeMCU ESP8266. Berikut adalah *list* program pengujian pada sensor PZEM-004T.

```
float v,i,p,

void loop(){

    v = PZEM.voltage(ip);

    if (v < 0.0) v = 0.0;

    Serial.print(v);Serial.print("V;");

    i = PZEM.current(ip);

    if (i < 0.0) i = 0.0;

    Serial.print(v);Serial.print("V;");

    p = PZEM.power(ip);

    if (p < 0.0) p = 0.0;

    Serial.print(v);Serial.print("V;");}
```

Pembacaan Nilai Tegangan (V)

Pembacaan Nilai Arus (I)

Pembacaan Nilai Daya (P)

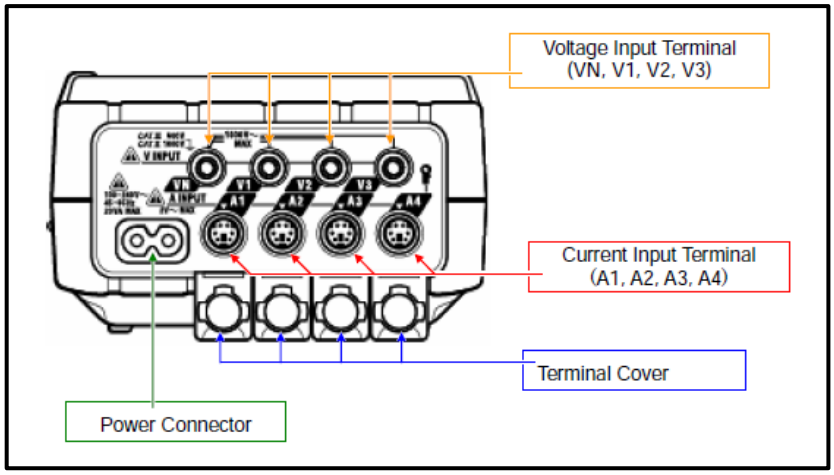
Dengan menggunakan *coding* program yang telah dibuat didapatkan nilai tegangan, arus dan daya. Data yang telah didapat dibandigkan dengan alat ukur *Kew Kyoritsu Power Quality Analizer.*

4.4.1. Prosedur Percobaan

Untuk menguji kebenaran nilai arus, tegangan dan daya pada alat yang dibuat maka dilakukan perbandingan antara alat yang dibuat dengan alat ukur yang digunakan. Berikut langkah-langkah yang perlu diperhatikan dalam melakukan pengukuran dengan alat ukur.

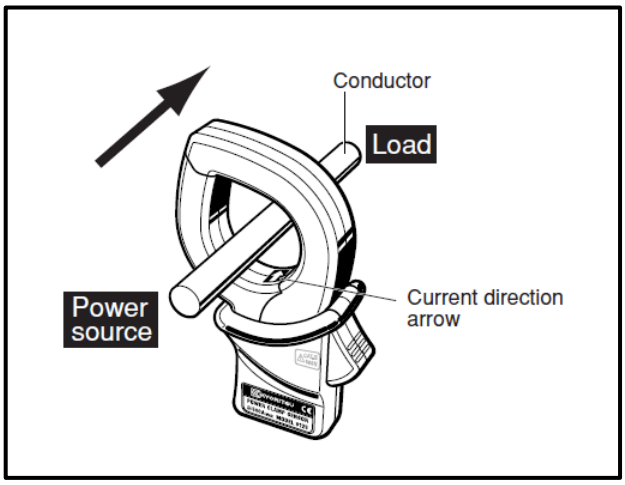
1. Pastikan alat yang dibuat telah siap untuk diuji.
2. Siapkan alat ukur *Kew Kyoritsu Power Quality Analyzer.*

3. hubungkan kabel netral pada VN di alat ukur dan kabel fasa pada V1. Kemudian hubungkan pada beban yang akan diukur. *Connector Probe Kew Kyoritsu Power Quality Analyzer* ditunjukan pada Gambar 4.5.



Gambar 4.5 *Connector Probe Kew Kyoritsu Power Quality Analyzer*

4. Pasang klem meter pada beban. Cara memasang klem meter yang benar ditunjukkan pada Gambar 4.6.



Gambar 4.6 Klem Meter 5A

Klem meter yang digunakan adalah klem meter yang mempunyai range arus dari 0-5A. Pada klem meter terdapat tanda panah yang digunakan sebagai acuan posisi klem meter tersebut. Klem meter dipasang pada kabel fasa dengan arah panah menuju beban.

5. Setting alat ukur *Kew Kyoritsu Power Quality Analyzer*.

Gambar 4.7 Menu *Setup* Alat Ukur

Setelah selesai melakukan pengaturan pada alat ukur, tekan tombol *start* pada alat ukur. Maka tampilan pada alat ukur langsung dapat menampilkan arus, tegangan dan daya.

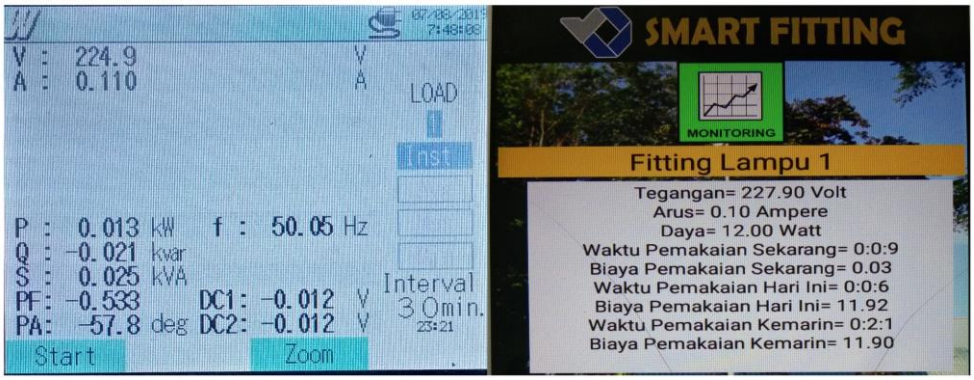
6. Bandingkan hasil pengukuran menggunakan sensor PZEM-004T dengan alat ukur *Kew Kyoritsu Power Quality Analyzer.*

4.4.2. Pengujian Sensor Dibanding Dengan Alat Ukur

Untuk menguji kepresisian alat yang telah dibuat dalam mengukur nilai tegangan, arus dan daya yang digunakan, maka dibandingkan dengan alat ukur *Kew Kyoritsu Power Quality Analyzer*. Berikut hasil pengujian yang dilakukan pada lampu yang berbeda:

a. Hasil pengujian lampu 12 watt

Pengujian sensor PZEM-004T dilakukan dengan menggunakan beban lampu sebesar 12 watt. Berikut hasil pengujian dapat dilihat pada Gambar 4.8.



Gambar 4.8 Hasil Pengujian Lampu 12 Watt

Dari pengukuran yang diperoleh terdapat *range* atau selisih pada alat ukur dan sensor. Untuk melihat besarnya selisih yang didapat pada sensor dan alat ukur dapat dilhat pada Tabel 4.4

Tabel 4.4 Data Pengujian Lampu 12 Watt

Pengujian ke-	Vuk	Va	Iuk	Ia	Puk	Pa
1	226,7	226,8	0,095	0,11	13	13
2	266,9	226,8	0,099	0,10	12	12
3	224,9	227,9	0,110	0,10	13	12
Rata-rata	226,16	227,16	0,101	0,103	12,66	12,33

Keterangan:

Vuk = nilai tegangan yang terbaca pada alat ukur

Va = nilai tegangan yang terbaca pada alat yang dibuat

Iuk = nilai arus yang terbaca pada alat ukur

Ia = nilai arus yang terbaca pada alat yang dibuat

Puk = nilai daya yang terbaca pada alat ukur

Pa = nilai daya yang terbaca pada alat yang dibuat

PFt = *power factor* secara perhitungan

Berdasarkan data pada Tabel 4.4 dapat dihitung PFt adalah:

$$PFt = \frac{\text{Daya lampu}}{\text{Tegangan rata-rata alat ukur x Arus rata-rata alat ukur}}$$

$$= \frac{12\ watt}{226{,}16\ V\ x\ \ 0{,}101\ A} = 0{,}525$$

Dari PFt yang didapat maka dapat dihitung besarnya Pt (daya secara perhitungan) yang digunakan yaitu:

$$Pt = Va\ x\ Ia\ x\ PFt$$

$$= 227{,}16\ x\ 0{,}103\ x\ 0{,}515 = \ 12{,}04\ watt$$

Data yang didapat dari alat ukur dan sensor selanjutnya dibandingkan dan dapat dilihat pada Tabel 4.5.

Tabel 4.5 Perbandingan Alat Ukur dan Sensor pada Lampu 12 Watt

PFt	**Vruk (V)**	**Vra (V)**	**Iruk (A)**	**Ira (A)**	**Pruk (Watt)**	**Pra (Watt)**	**Pt (Watt)**	**Persentase *error* (%)**		
								V	**I**	**P**
0,515	226,16	227,16	0,101	0,103	12,66	12,33	12,04	0,02	17,82	2,60

Keterangan:

Vruk = Tegangan rata-rata alat ukur

Vra = Tegangan rata-rata alat yang dibuat

Iruk =Arus rata-rata alat ukur

Ira = Arus rata-rata alat yang dibuat

Ira = Arus rata-rata alat yang dibuat

Ira = Arus rata-rata alat yang dibuat

Pruk = Daya rata-rata alat ukur

Pra = Daya rata-rat sensor

Perhitungan persentase *error* tegangan, arus dan daya yang didapat merupakan pembandingan pada alat ukur dan alat yang dibuat, sehingga rumusnya adalah:

- Persentase *error* tegangan $= \left|\frac{226,16-227,16}{226,16}\right| \times 100\%$

  $= 0,44\ \%$

- Persentase *error* arus $= \left|\frac{0,101-0,103}{0,101}\right| \times 100\%$

  $= 1,98\ \%$

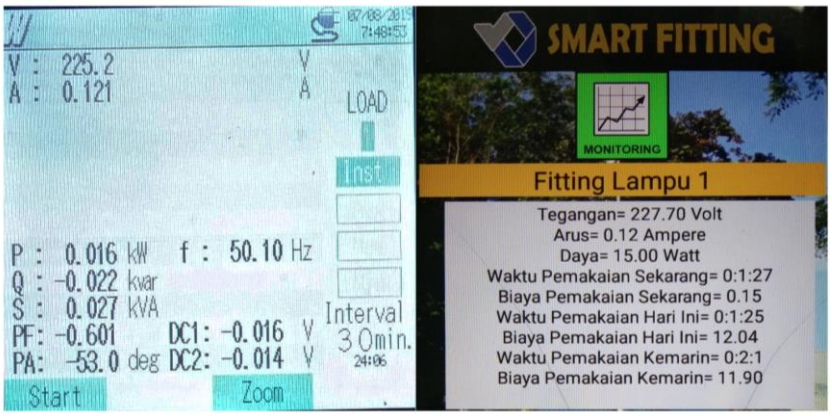
- Persentase *error* daya $= \left|\frac{12{,}66-12{,}33}{12{,}66}\right| \times 100\%$

$= 2{,}60\ \%$

Berdasarkan alat yang dibuat dan dibandingkan dengan alat ukur maka pengujian pada lampu 12 watt terdapat selisih pada pengukuran arus sehingga mempengaruhi hasil dari perhitungan daya yakni 12,04 watt. Akan tetapi pada pengukuran tegangan mendapatkan hasil yang sangat baik yakni dengan selisih rata-rata sebesar 1 V. Jika dibandingkan antara alat ukur dan alat yang dibuat maka mendapatkan persentase *error* tegangan yaitu sebesar 0,44%, persentase *error* arus sebesar 1,98% dan persentase *error* daya sebesar 2,60%. Hal ini terjadi karena terdapat perubahan nilai tegangan, arus dan daya yang dibaca sensor disetiap waktu, namun sensor ini dapat digunakan sesuai dengan semestinya.

b. Hasil pengujian lampu 18 watt

Pengujian sensor PZEM-004T dilakukan dengan menggunakan beban lampu sebesar 18 watt. Berikut hasil pengujian dapat dilihat pada Gambar 4.9.



Gambar 4.9 Hasil Pengujian Lampu 18 Watt

Dari pengukuran yang diperoleh terdapat *range* atau selisih pada alat ukur dan sensor. Untuk melihat besarnya selisih yang didapat pada sensor dan alat ukur dapat dilhat pada Tabel 4.4

Tabel 4.6 Data Pengujian Lampu 18 Watt

Pengujian	Vuk	Va	Iuk	Ia	Puk	Pa
1	227,6	227,1	0,122	0,12	15	15
2	266,8	226,8	0,122	0,13	16	16
3	225,2	227,7	0,121	0,12	16	15
Rata-rata	226,53	227,2	0,122	0,123	15,66	15,33

Berdasarkan data pada Tabel 4.6 dapat dihitung PFt adalah:

$$PFt = \frac{\text{Daya lampu}}{\text{Tegangan rata-rata alat ukur x Arus rata-rata alat ukur}}$$

$$= \frac{18\ watt}{226{,}53\ V\ x\ 0{,}122\ A} = 0{,}651$$

Dari PFt yang didapat maka dapat dihitung besarnya Pt (daya secara perhitungan) yang digunakan yaitu:

$$Pt = Va\ x\ Ia\ x\ PFt$$

$$= 227{,}2\ x\ 0{,}123\ x\ 0{,}651 = 18{,}19\ watt$$

Data yang didapat dari alat ukur dan sensor selanjutnya dibandingkan dan dapat dilihat pada Tabel 4.7.

Tabel 4.7 Perbandingan Alat Ukur dan Sensor pada Lampu 18 Watt

PFt	Vruk (V)	Vra (V)	Iruk (A)	Ira (A)	Pruk (Watt)	Pra (Watt)	Pt (Watt)	Persentase *error* (%)		
								V	I	P
0,651	226,53	227,2	0,122	0,123	15,66	15,33	18,19	0,29	0.81	4,21

Perhitungan persentase *error* tegangan, arus dan daya yang didapat merupakan pembandingan pada alat ukur dan alat yang dibuat, sehingga rumusnya adalah:

- Persentase *error* tegangan $= \left|\frac{226{,}53-227{,}2}{226{,}53}\right| \times 100\%$

$= 0{,}29\ \%$

- Persentase *error* arus $= \left|\frac{0{,}122-0{,}123}{0{,}122}\right| \times 100\%$

  $= 0.81\%$

- Persentase *error* daya $= \left|\frac{15{,}66-15{,}33}{15{,}66}\right| \times 100\%$

  $= 2{,}11\ \%$

Pengujian dilakukan pada beban lampu 18 watt. Pada pengujian sensor ini dilakukan perbandingan antara alat ukur dengan sensor yang digunakan. Berdasarkan data pada Tabel 4.7 dapat diketahui bahwa tegangan yang didapat mempunyai persentase yang kecil yaitu sebesar 0,29%. Akan tetapi arus yang diukur oleh sensor mempunyai persentase *error* sebesar 0,81%. Ini disebabkan karena sensor PZEM-004T yang digunakan mempunyai kelemahan dalam membaca arus yang kecil, sehingga mempengaruhi perhitungan daya berdasarkan teori yang kemudian didapatkan daya yang lebih besar melalui perhitungan jika dibandingkan dengan alat ukur dan sensor. Namun pada perhitungan persentase *error* dilakukan pembandingan antara alat ukur dan sensor yang digunakan didapatkan sebesar 0,81% sehingga alat ini dapat digunakan sebagaimana mestinya

c. Hasil Pengujian lampu 24 watt

Pengujian sensor PZEM-004T dilakukan dengan menggunakan beban lampu sebesar 24 watt. Berikut hasil pengujian dapat dilihat pada Gambar 4.10.

Gambar 4.10 Hasil Pengujian Lampu 24 Watt

Dari pengukuran yang diperoleh terdapat *range* atau selisih pada alat ukur dan sensor. Untuk melihat besarnya selisih yang didapat pada sensor dan alat ukur dapat dilhat pada Tabel 4.8

Tabel 4.8 Data Pengujian Lampu 24 Watt

Pengujian	**Vuk**	**Va**	**Iuk**	**Ia**	**Puk**	**Pa**
1	228,3	228,2	0,184	0,18	24	23
2	266,6	226,3	0,180	0,18	24	23
3	224,0	227,9	0,188	0,17	23	24
Rata-rata	226,3	227,46	0,184	0,176	23,66	23,33

Berdasarkan data pada Tabel 4.8 dapat dihitung PFt adalah:

$$PFt = \frac{\text{Daya lampu}}{\text{Tegangan rata−rata alat ukur x Arus rata−rata alat ukur}}$$

$$= \frac{24\ watt}{226{,}3\ V\ x\ \ 0{,}184\ A} = 0{,}576$$

Dari PFt yang didapat maka dapat dihitung besarnya Pt (daya secara perhitungan) yang digunakan yaitu:

$$Pt = Va\ x\ Ia\ x\ PFt$$

$$= 227{,}46\ x\ 0{,}176\ x\ 0{,}576 = \ 23{,}05\ watt$$

Data yang didapat dari alat ukur dan sensor selanjutnya dibandingkan dan dapat dilihat pada Tabel 4.9.

Tabel 4.9 Perbandingan Alat Ukur dan Sensor pada Lampu 24 Watt

PFt	**Vruk (V)**	**Vra (V)**	**Iruk (A)**	**Ira (A)**	**Pruk (Watt)**	**Pra (Watt)**	**Pt (Watt)**	**Persentase *error* (%)**		
								V	**I**	**P**
0,576	226,3	227,46	0,184	0,176	23,33	23,66	23,05	0,51	4,86	1,41

Perhitungan persentase *error* tegangan, arus dan daya yang didapat merupakan pembandingan pada alat ukur dan alat yang dibuat, sehingga rumusnya adalah:

- Persentase *error* tegangan $= \left|\frac{226,3-227,46}{226,3}\right| \times 100\%$

  $= 0,51\ \%$

- Persentase *error* arus $= \left|\frac{0,184-0,176}{0,184}\right| \times 100\%$

  $= 4,34\ \%$

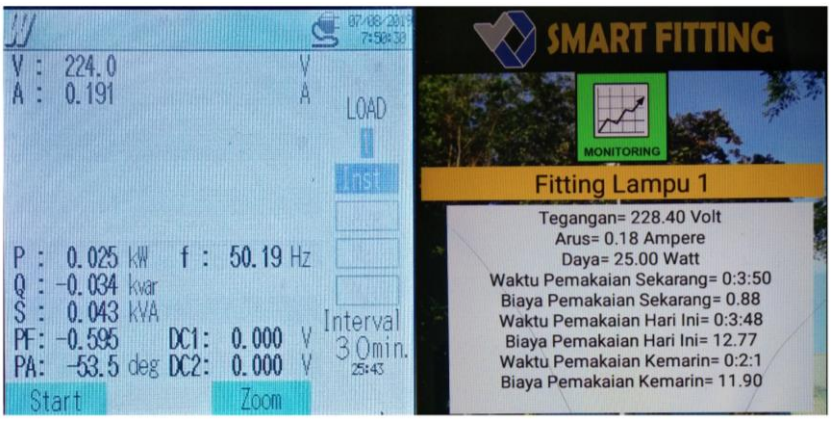
- Persentase *error* daya $= \left|\frac{23,33-23,66}{23,33}\right| \times 100\%$

  $= 1,41\ \%$

Berdasarkan Tabel 4.9 didapakan tegangan rata-rata dan arus rata-rata dari alat ukur dan sensor yang digunakan. Dari data yang didapat dilakukan perhitungan *power factor* sebesar 0,573. Perhitungan dilakukan dengan menggunakan nilai tegangan dan arus rata-rata pada alat ukur. Dari *power factor* yang didapat maka dilakukan perhitungan daya secara teori, sehingga Pt=22,94. Terdapat perbedaan daya yang diukur oleh sensor dan alat ukur. Hal ini juga karena besarnya persntase *error* arus yang diukur oleh sensor yakni 4,34%. Ketika dilakukan perhitungan persentase *error* daya didapatkan selisih yang kecil yakni 1,41%, namun dengan demikian alat ini dapat digunakan sebagaimana mestinya.

d. Hasil Pengujian lampu 27 watt

Pengujian sensor PZEM-004T dilakukan dengan menggunakan beban lampu sebesar 27 watt. Berikut hasil pengujian dapat dilihat pada Gambar 4.11.



Gambar 4.11 Hasil Pengujian Lampu 27 Watt

Dari pengukuran yang diperoleh terdapat *range* atau selisih pada alat ukur dan sensor. Untuk melihat besarnya selisih yang didapat pada sensor dan alat ukur dapat dilhat pada Tabel 4.10.

Tabel 4.10 Data Pengujian Lampu 27 Watt

Pengujian	Vuk	Va	Iuk	Ia	Puk	Pa
1	226,4	226,5	0,185	0,19	25	26
2	266,5	226,6	0,185	0,18	25	25
3	224	228,4	0,191	0,18	25	25
Rata-rata	225,63	227,16	0,187	0,183	25	25,33

Berdasarkan data pada Tabel 4.10 dapat dihitung PFt adalah:

$$PFt = \frac{\text{Daya lampu}}{\text{Tegangan rata-rata alat ukur x Arus rata-rata alat ukur}}$$

$$= \frac{27\ watt}{225{,}63\ V\ x\ 0{,}187\ A} = 0{,}639$$

Dari PFt yang didapat maka dapat dihitung besarnya Pt (daya secara perhitungan) yang digunakan yaitu:

$$Pt = Vra\ x\ Ira\ x\ PFt$$

$$= 227{,}16\ x\ 0{,}183\ x\ 0{,}639 = 26{,}56\ watt$$

Data yang didapat dari alat ukur dan sensor selanjutnya dibandingkan dan dapat dilihat pada Tabel 4.11.

Tabel 4.11 Perbandingan Alat Ukur dan Sensor pada Lampu 27 Watt

PFt	**Vruk (V)**	**Vra (V)**	**Iruk (A)**	**Ira (A)**	**Pruk (Watt)**	**Pra (Watt)**	**Pt (Watt)**	**Persentase *error* (%)**		
								V	**I**	**P**
0,639	225,63	227,16	0,187	0,183	25	25,33	26,56	0,67	2,14	1,32

Perhitungan persentase *error* tegangan, arus dan daya yang didapat merupakan pembandingan pada alat ukur dan alat yang dibuat, sehingga rumusnya adalah:

- Persentase *error* tegangan $= \left|\frac{225{,}63-227{,}16}{225{,}63}\right| \times 100\%$

  $= 0{,}67\ \%$

- Persentase *error* arus $= \left|\frac{0{,}188-0{,}183}{0{,}188}\right| \times 100\%$

  $= 2{,}14\ \%$

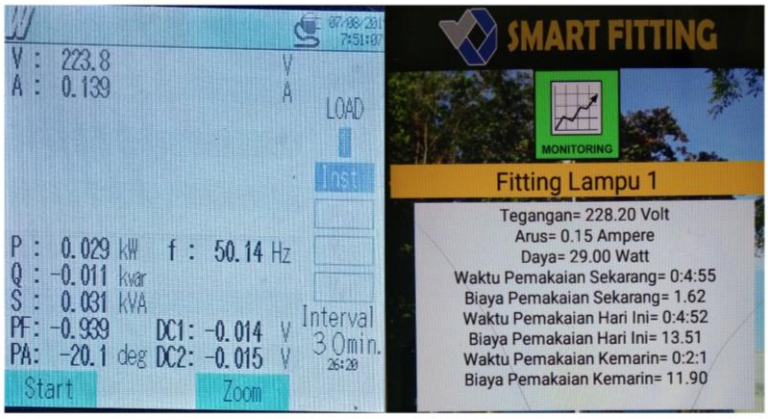
- Persentase *error* daya $= \left|\frac{25-25{,}33}{25}\right| \times 100\%$

  $= 1{,}32\ \%$

Berdasarkan data pada Tabel 4.10 maka dilakukan penghitungan nilai *power factor* dari alat ukur yang digunakan, sehingga didapatkan nilai PFt sebesar 0,639. Selanjutnya dari PFt yang didapat dilakukan perhitungan daya yang digunakan sensor secara teori, sehingga didapatkan daya secara perhitungan sebesar 26,56 watt. Pada Tabel 4.11 dilakukan penghitungan nilai persentase *error* antara daya pada alat ukur dan daya pada sensor, sehingga didapatkan *error* pada daya yang sebesar 1,32%, tegangan sebesar 0,67% dan persentase *error* arus sebesar 2,14% Namun secara keseluruhan alat yang dibuat dapat digunakan sebagaimana mestinya.

e. Hasil Pengujian lampu 30 watt

Pengujian sensor PZEM-004T dilakukan dengan menggunakan beban lampu sebesar 30 watt. Berikut hasil pengujian dapat dilihat pada Gambar 4.12



Gambar 4.12 Hasil Pengujian Lampu 30 Watt

Dari pengukuran yang diperoleh terdapat *range* atau selisih pada alat ukur dan sensor. Untuk melihat besarnya selisih yang didapat pada sensor dan alat ukur dapat dilhat pada Tabel 4.12

Tabel 4.12 Data Pengujian Lampu 30 Watt

Pengujian	Vuk	Va	Iuk	Ia	Puk	Pa
1	226,5	226,8	0,139	0,13	30	29
2	266,4	226,7	0,136	0,13	29	29
3	223,8	228,2	0,139	0,15	29	29
Rata-rata	225,56	227,23	0,138	0,136	29,3	29

Berdasarkan data pada Tabel 4.12 dapat dihitung PFt adalah:

$$PFt = \frac{\text{Daya lampu}}{\text{Tegangan rata-rata alat ukur x Arus rata-rata alat ukur}}$$

$$= \frac{30\ watt}{225{,}56\ V\ x\ 0{,}138\ A} = 0{,}963$$

Dari PFt yang didapat maka dapat dihitung besarnya Pt (daya secara perhitungan) yang digunakan yaitu:

$$Pt = Vra\ x\ Ira\ x\ PFt$$
$$= 225{,}73\ x\ 0{,}136\ x\ 0{,}963 = 29{,}53\ watt$$

Data yang didapat dari alat ukur dan sensor selanjutnya dibandingkan dan dapat dilihat pada Tabel 4.13.

Tabel 4.13 Perbandingan Alat Ukur dan Sensor pada Lampu 30 Watt

PFt	**Vruk (V)**	**Vra (V)**	**Iruk (A)**	**Ira (A)**	**Pruk (Watt)**	**Pra (Watt)**	**Pt (Watt)**	**Persentase *error* (%)**		
								V	**I**	**P**
0,963	225,63	226,73	0,138	0,136	29,3	29	29,56	0,04	1,45	1,02

Perhitungan persentase *error* tegangan, arus dan daya yang didapat merupakan pembandingan pada alat ukur dan alat yang dibuat, sehingga rumusnya adalah:

- Persentase *error* tegangan $= \left|\frac{225{,}63-225{,}73}{225{,}63}\right| \times 100\%$
  $= 0{,}04\ \%$

- Persentase *error* arus $= \left|\frac{0{,}138-0{,}136}{0{,}138}\right| \times 100\%$
  $= 1{,}45\ \%$

- Persentase *error* daya $= \left|\frac{29{,}3-29}{29{,}3}\right| \times 100\%$
  $= 1{,}02\ \%$

Pada Tabel 4.13 dapat diketahui nilai rata-rata tegangan arus dan daya pada alat ukur maupun sensor yang digunakan. Dari pengujian lampu 30 watt didapatkan *power factor* sebesar 0,963, sehingga ketika dilakukan penghitungan daya yang digunakan didapatkan sebesar 29,56 watt. Daya yang didapat

dibandingkan dengan alat ukur dan sensor yang digunakan mendapatkan selisih yang tidak terlalu jauh. Perbandingan daya menggunakan alat ukur dan sensor yang dibuat mempunyai persentase *error* sebesar 1,02%, begitu juga tegangan sebesar 0,04% dan persentase *error* pada arus sebesar 1,45% namun degan demikian alat ini dapat digunakan sebagaimana semestinya.

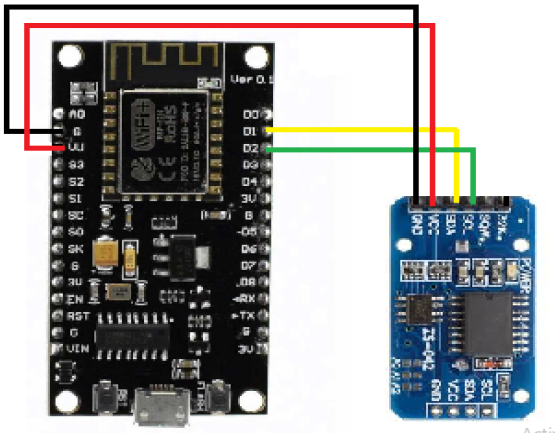
Berdasarkan hasil pengujian yang dilakukan pada lampu 12 watt, 18 watt, 24 watt, 27 watt dan 30 watt dengan membandingkan alat ukur yang digunakan yaitu *Kew Kyoritsu Power Quality Analyzer.*dan sensor PZEM-004T didapatkan persentase *error* maksimal 0,67% saat pengujian tegangan, persentase *error* maksimal 4,34% saat pengujian arus dan persentase *error* maksimal 2,6% saat pengujian daya. Pada pengujian tegangan dan daya didapatkan *error* yang cukup kecil. Akan tetapi pada pengukuran arus didapatkan persentase *error* yang cukup besar pada lampu 12 watt, hal ini disebabkan karena arus yang diukur sebesar 0,095-0,110.

Selain membandingkan dengan alat ukur pengujian ini juga dilakukan pembandingan secara perhitungan (teori). Adanya *error* yang didapat pada pengukuran arus akan sangat berdampak pada daya yang didapat melalui perhitungan. Dapat dilihat pada lima kali percobaan pada beban yang berbeda didapatkan *error* terbesar pengukuran arus terjadi pada lampu dengan beban terkecil yaitu lampu 12 watt. Akan tetapi semakin besar beban yang diukur maka persentase *error* pada arus akan semakin kecil.

4.5. Real Time Clock DS3231

Real time clock DS3231 merupakan piranti elektronika pencacah waktu yang mempunyai tingkat akurasi dan presisi yang sangat tinggi**.** Dengan menggunakan IC *real time clock* DS3231 yang merupakan salah satu jenis *real time clock* yang menggunakan komunikasi serial SDA dan SCL. *real time clock* DS3231 menggunakan *battery* 3V dan 6 pin *interface.* Modul ini dilengkapi dengan IC AT24C32 yang memberikan EEPROM tambahan sebesar 4 KB (32.768 bit) [3].

Pada proyek akhir ini *real time clock* difungsikan sebagai penghitung waktu dan tanggal serta hari dalam merekam waktu pemakaian listrik. Waktu yang didapat akan diakumulasikan menjadi biaya total yang akan dibayar lalu ditampilkan ke Android. Berikut skematik pemasangan pin pada *real time clock* DS3231 dapat ditunjukan pada Gambar 4.13.



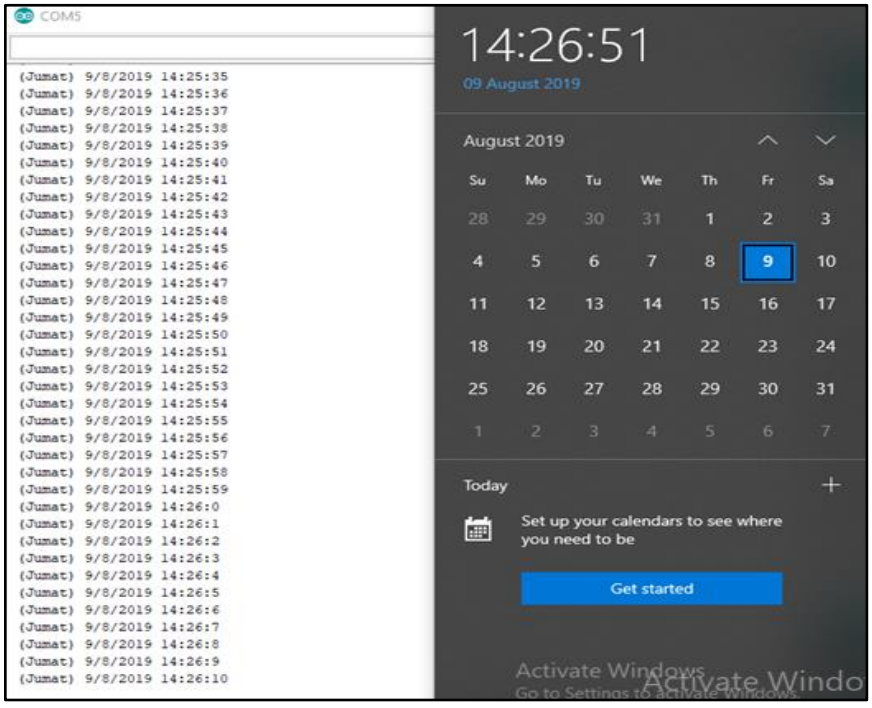
Gambar 4.13 Skematik Pemasangan Pin pada *Real Time Clock* DS3231

Untuk mengetahui fungsi dari *real time clock* ini apakah telah dapat bekerja dengan baik yaitu dengan menampilkan hari, tanggal dan waktu pada *serial monitor* di NodeMCU ESP8266. Berikut adalah *list* program hasil pengujian dari *real time clock*.

```
Void Setup(){Serial.begin(9600);}

void loop(){

    DateTime now = rtc.now();

    Serial.print(" (");

    Serial.print(daysOfTheWeek[now.dayOfTheWeek()]);

    Serial.print(") ");

    Serial.print(now.day(), DEC); Serial.print('/');

    Serial.print(now.month(), DEC); Serial.print('/');
```

```
Serial.print(now.year(), DEC);

Serial.print('\t');

Serial.print(now.hour(), DEC);

Serial.print(':');

Serial.print(now.minute(), DEC);

Serial.print(':');

Serial.println(now.second(), DEC);}
```

Setelah dilakukan *compile* dan *upload* pada *board* NodeMCU ESP8266 didapatkan hasil pada Gambar 4.14.

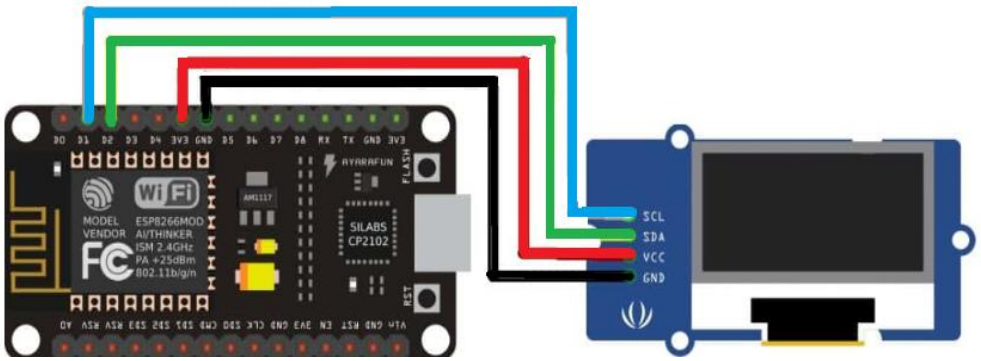


Gambar 4.14 Hasil Pengujian *Real Time Clock* DS3231

Dari hasil pengujian *real time clock* DS3231 tersebut dapat diketahui bahwa hari, tanggal dan waktu yang diinginkan berjalan dengan baik dan sesuai dengan pengaturan yang telah dilakukan.

4.6. Pengujian OLED 0.91

Salah satu piranti yang digunakan pada proyek akhir ini adalah OLED 0.91. piranti ini digunakan untuk menampilkan waktu yang diatur pada setiap fiting lampu yang dibuat. Hal ini dikarenakan untuk mencegah kesalahan dalam mengatur *timer* hidup/mati lampu. Waktu yang ditampilkan pada OLED juga digunakan untuk mengetahui kondisi RTC apakah waktu yang diatur telah sesuai atau belum. Untuk mengetahui fungsi dari OLED 0.91 maka dilakukan pengujian menampilkan karakter pada OLED 0.91. Berikut skematik untuk menampilkan karakter pada OLED 0.91 dapat ditunjukan pada Gambar 4.15.



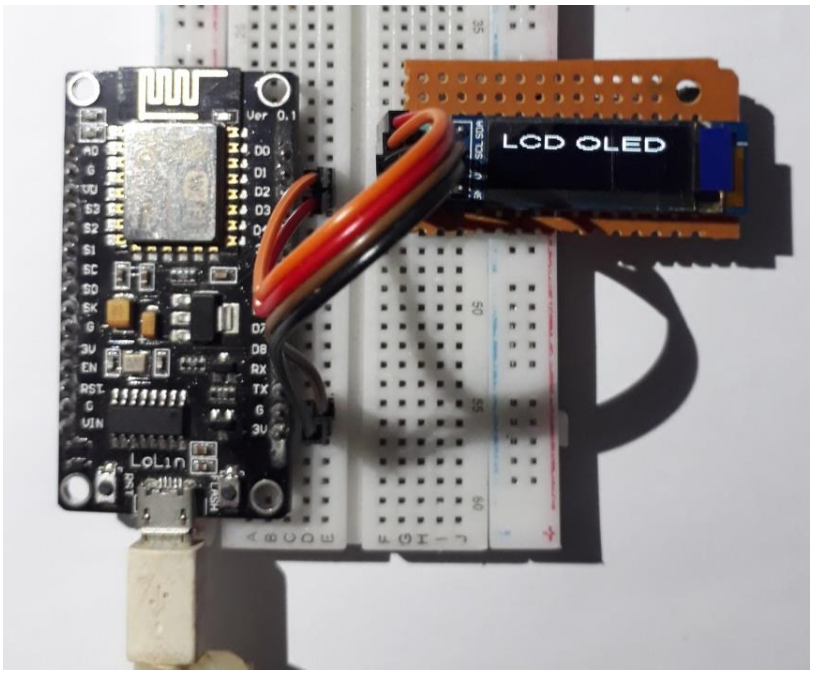
Gambar 4.15 Skematik Mengakses OLED 0.91

Dari skematik yang telah dibuat selanjutnya dilakukan pemrograman dengan *list* program berikut.

```
void setup() {
      display.init();

      display.flipScreenVertically();

      display.setFont(ArialMT_Plain_10);

      }

void loop() {

      display.clear();
```

```
display.setTextAlignment(TEXT_ALIGN_LEFT);
display.setFont(ArialMT_Plain_24);
display.drawString(0,0, "LCD OLED" );
display.display();
}
```

Setelah dilakukan pemrograman pada NodeMCU ESP8266 maka karakter yang ditampilkan pada OLED 0.91 dapat ditunjukan pada Gambar 4.16.



Gambar 4.16 Karakter yang Ditampilkan pada OLED 0.91

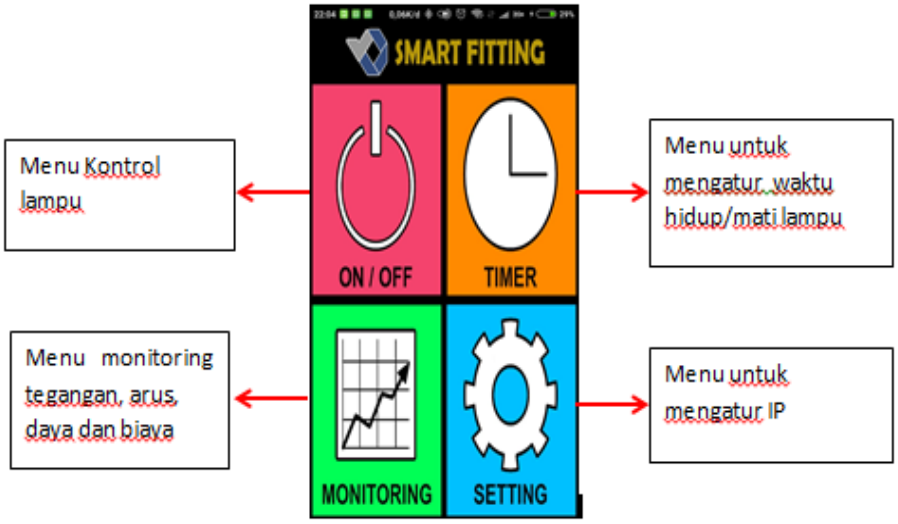
Dalam mengakses OLED 0.91 menggunakan komunikasi SDA SCL. Pin SDA terhubung pada pin D2 dan pin SCL terhubung pada pin D1 pada NodeMCU ESP8266. Pada Gambar 4.16 dapat ditampilkan hasil dari pengujian OLED 0.91 yakni menampilkan karakter sesuai dengan pemrograman yang dilakukan.

4.7. *Software* Fiting Lampu Berbasis Android

Proyek akhir "Kontrol dan *Monitoring* Fiting Lampu Berbasis Android" menggunakan aplikasi yang dapat digunakan pada Android. aplikasi ini digunakan untuk *monitoring* arus, tegangan, daya dan biaya serta dapat mengontrol perangkat yang akan dibuat. Dalam hal ini penulis memutuskan membuat aplikasi Android menggunakan *software* App Inventor. Disamping mudah digunakan aplikasi ini mempunyai fitur yang sesuai dengan aplikasi yang ingin dibuat.

4.7.1. Aplikasi "Smart Fitting"

Aplikasi "Smart Fitting" merupakan aplikasi yang dibuat untuk mengontrol dan *monitoring* fiting lampu yang dibuat pada proyek akhir ini. Aplikasi "Smart Fitting" ini dibuat dengan menggunakan *software* App Inventor. Pada pengoperasian aplikasi ini dirancang khusus dengan menggunakan *SSID* yang terdapat pada NodeMCU ESP8266 sehingga aplikasi ini hanya dapat diakses dengan mengatur SSID pada Android sama dengan SSID pada NodeMCU. Berikut tampilan utama aplikasi "Smart Fitting" pada Gambar 4.17



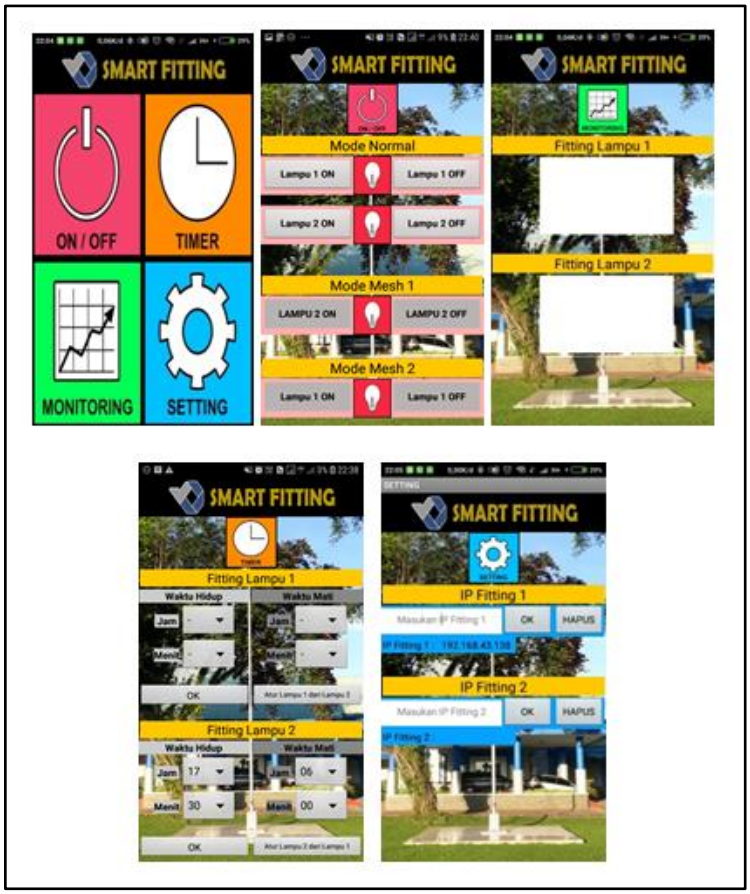
Gambar 4.17 Tampilan Utama Aplikasi "Smart Fitting"

Untuk mengoperasikan aplikasi yang telah dibuat, *user* haruslah memasukan SSID yang telah disediakan. Untuk mengetahui perangkat yang

terhubung dapat diketahui sesuai dengan kemampuan Android *user*. Setelah perangkat terhubung maka dapat dilakukan pengontrolan dan *monitoring* fiting lampu melalui jaringan *WiFi* yang telah dibuat dengan memanfaatkan kontroler NodeMCU ESP8266.

4.7.2. Pembuatan Aplikasi "Smart Fitting"

Pembuatan aplikasi "Smart Fitting" yang digunakan pada Android dilakukan dengan menggunakan *software* App Inventor secara *online* menggunakan komputer. Pembuatan aplikasi Android dengan *software* App Inventor ini dibagi menjadi tiga tahap, yakni rancangan tampilan, blok editor dan proses komunikasi. Adapun hasil tampilan yang telah dibuat dapat dilihat pada Gambar 4.18.



Gambar 4.18 Menu Aplikasi "Smart Fitting"

Untuk mengoperasikan aplikasi ini harus mengatur SSID pada Android sesuai dengan SSID pada fiting lampu yang dibuat. Aplikasi yang dibuat mempunyai tampilan berdasarkan fungsi masing-masing antara lain fungsi sebagai saklar, *monitoring*, pengaturan *timer* dan pengaturan IP.

4.7.3. Pengujian Aplikasi "Smart Fitting"

Pengujian aplikasi "Smart Fitting" dilakukan untuk mengetahui apakah alat yang dibuat sesuai dengan fungsi yang diinginkan. Pengujian dilakukan pada setiap menu yang dibuat, yaitu menghidupkan atau mematikan lampu, *monitoring* arus, tegangan, daya dan biaya yang digunakan serta pengaturan *timer* untuk mematikan atau mematikan lampu dengan waktu yang diinginkan.

Adapun untuk melakukan pengujian pada aplikasi yang dibuat dapat dilakukan dengan langkah-langkah sebagai berikut:

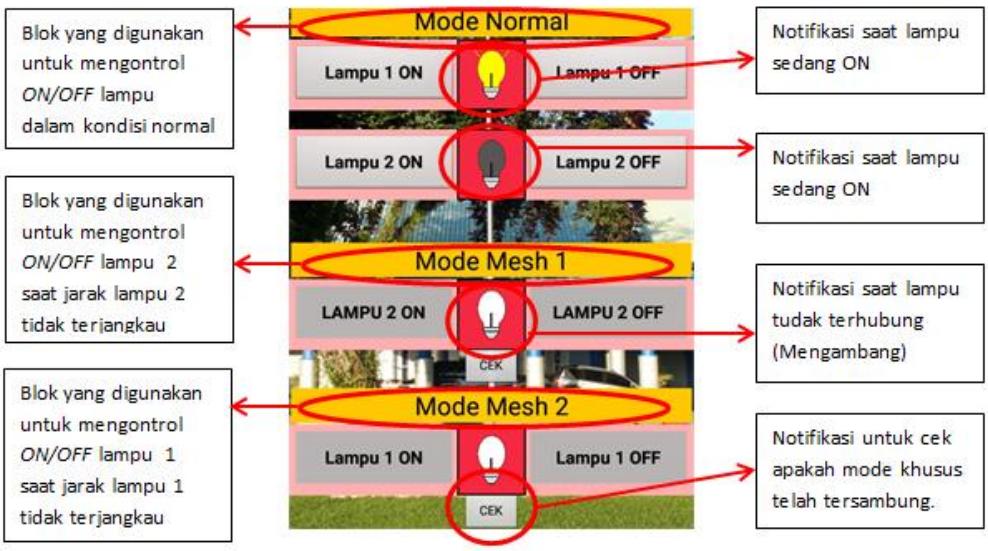
1. Hidupkan *hotspot* pada perangkat Android.
2. Tunggu hingga perangkat terhubung (*connected*) pada Android.
3. Buka aplikasi "Smart Fitting".
4. Masuk ke menu setting lalu setting IP fiting 1 dan IP Fiting 2. Tekan tombol oke lalu kembali ke menu utama.
5. Masuk ke menu *ON/OFF*. Pilih tombol Lampu *ON* untuk menghidupkan lampu dan Pilih tombol Lampu *OFF* untuk mematikan lampu.
6. Setelah menghidupkan lampu masuk ke menu *monitoring* untuk *monitoring* arus, tegangan, daya dan biaya yang digunakan pada beban yang diukur.
7. Masuk ke menu *timer* untuk mengatur jadwal hidup atau mati pada lampu.

Fiting lampu akan tetap menjalankan perintah yang telah diberikan oleh *user* melalui Android meskipun *user* keluar dari aplikasi "Smart Fitting". Pada aplikasi ini telah disediakan indikator untuk mengetahui kondisi lampu pada saat ini. Indikator ini terdapat pada menu *ON/OFF* pada aplikasi "Smart Fitting". Untuk dapat terhubung dengan jaringan pada fiting lampu yang dibuat maka hal pertama yang harus dilakukan adalah memasukkan IP setiap fiting lampu pada

"menu setting". Berikut tampilan "Menu Setting" yang telah dibuat pada Android dapat ditunjukan pada Gambar 4.19.

Gambar 4.19 Tampilan Menu *Setting*

Fungsi dari IP ini digunakan sebagai pengenal untuk setiap fiting lampu. IP yang ada bersifat tetap (tidak berubah). Sehingga untuk *user* mengetahui IP masing-masing fiting lampu hanya perlu melihat label IP yang telah penulis tempel pada setiap fiting lampu. Berikut tampilan "Menu *ON/OFF*" yang telah dibuat pada Android dapat ditunjukan pada Gambar 4.20.



Gambar 4.20 Tampilan Menu *ON/OFF*

Pada saat melakukan pengontrolan maka kondisi lampu akan sesuai dengan notifikasi yang terdapat pada Smartphone. Berikut hasil pengontrolan lampu pada setiap fiting lampu yang dibuat.

a. Hasil pengujian menghidupkan lampu "*Mode* Normal" dapat dilihat pada Gambar 4.21.

Gambar 4.21 Menghidupkan Lampu *Mode* Normal

b. Hasil pengujian mematikan lampu "*Mode* Normal" dapat dilihat pada Gambar 4.22.

Gambar 4.22 Mematikan Lampu *Mode* Normal

Pada Gambar 4.19 merupakan proses menghidupkan lampu dengan *mode* normal. Pada menu *ON/OFF* terdapat indikator kondisi lampu. Ketika lampu hidup maka indikator lampu hidup akan menyala. Ketika tekan blok fungsi "Lampu 1 ON" dan "Lampu 2 ON" maka lampu akan menyala seperti Gambar 4.21. Apabila blok fungsi "Lampu 1 ON" dan "Lampu 2 ON" ditekan maka lampu akan mati seperti Gambar 4.22

Selanjutnya melakukan pengujian menu *monitoring* yang dibuat. Pada Gambar 4.23 menu *monitoring* mempunyai label "Fiting Lampu 1" dan "Fiting Lampu 2" yang masing-masing label berisi data berupa tegangan, arus, daya, waktu pemakaian hari ini, biaya pemakaian hari ini, waktu pemakaian kemarin dan total biaya kemarin. Berikut hasil dari tampilan menu *monitoring* dapat ditunjukan pada Gambar 4.23.

Gambar 4.23 Pengujian Menu *Monitoring*

Dalam proses pengiriman data dari NodeMCU ESP8266 mempunyai waktu kurang lebih selama tiga detik, sehingga perubahan data pada Android dapat lebih lama dibandingkan data pada NodeMCU ESP8266 di serial monitor. Ini disebabkan karena pengolahan data yang cukup banyak pada sensor PZEM-004T, ditambah proses pengiriman data dari NodeMCU ESP8266 itu sendiri. Hal ini juga dilakukan pengaturan waktu yang diberikan penulis pada *list* program

yang dibuat untuk mencegah penumpukan data ketika NodeMCU ESP8266 mengirimkan data pada Android. Fitur lain yang dibuat pada aplikasi "Smart Fitting" adalah menu *timer*. Berikut tampilan menu *timer* dapat ditunjukan pada Gambar 4.24.

Gambar 4.24 Tampilan Menu *Timer*

Untuk menentukan waktu hidup dan mati lampu, *user* bisa mengatur melalui menu *timer*. *User* bisa mengatur salah satu *timer* atau keduanya dalam waktu bersamaan. Pada Gambar 4.24 terdapat *list picker* jam dan menit untuk masing-masing fiting lampu. Angka yang diatur akan dikirimkan ke NodeMCU ESP8266 dan diolah menjadi *timer* dengan membandingkan dengan waktu di *real time clock* DS3231. Apabila waktu telah terpenuhi maka lampu akan hidup atau mati sesuai *timer* yang telah diatur.

4.8. Program NodeMCU ESP 8266

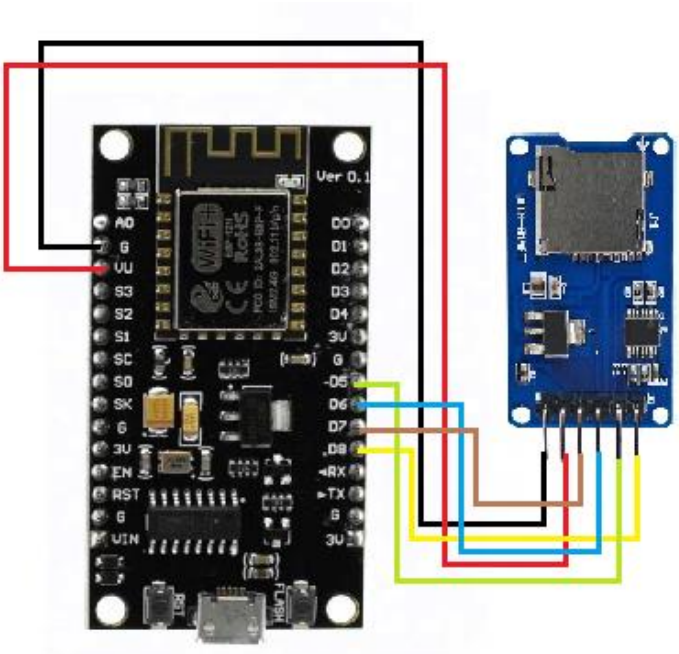
Pembuatan *list program* dilakukan menggunakan *software* Arduino IDE. Pemograman proyek akhir fiting lampu ini dibagi menjadi beberapa tahapan sesuai dengan perancangan yang telah dibuat yaitu sebagai berikut :

a. Pemograman *real time clock* DS3231 untuk mengatur fungsi waktu.
b. Pemograman Modul *Solid State Relay* untuk mengontrol *ON/OFF* pada fiting lampu.

c. Pemograman modul sensor PZEM-004T untuk mendeteksi nilai tegangan, arus dan daya yang didapatkan pada setiap lampu yang digunakan.
d. Pemograman SD Card di *software* Arduino IDE sebagai penyimpanan data biaya dan waktu pemakaian.
e. Pemrograman metode Tree dengan NodeMCU ESP8266.
f. Pemograman fiting lampu secara keseluruhan.

4.9. Pengujian Data *Logger*

Dalam pembuatan *logger* waktu komponen yang diperlukan sebagai antarmuka untuk menyimpan data adalah modul SD Card. Berikut skematik Pemasangan pin pada modul SD Card dapat ditunjukan pada Gambar 4.25.

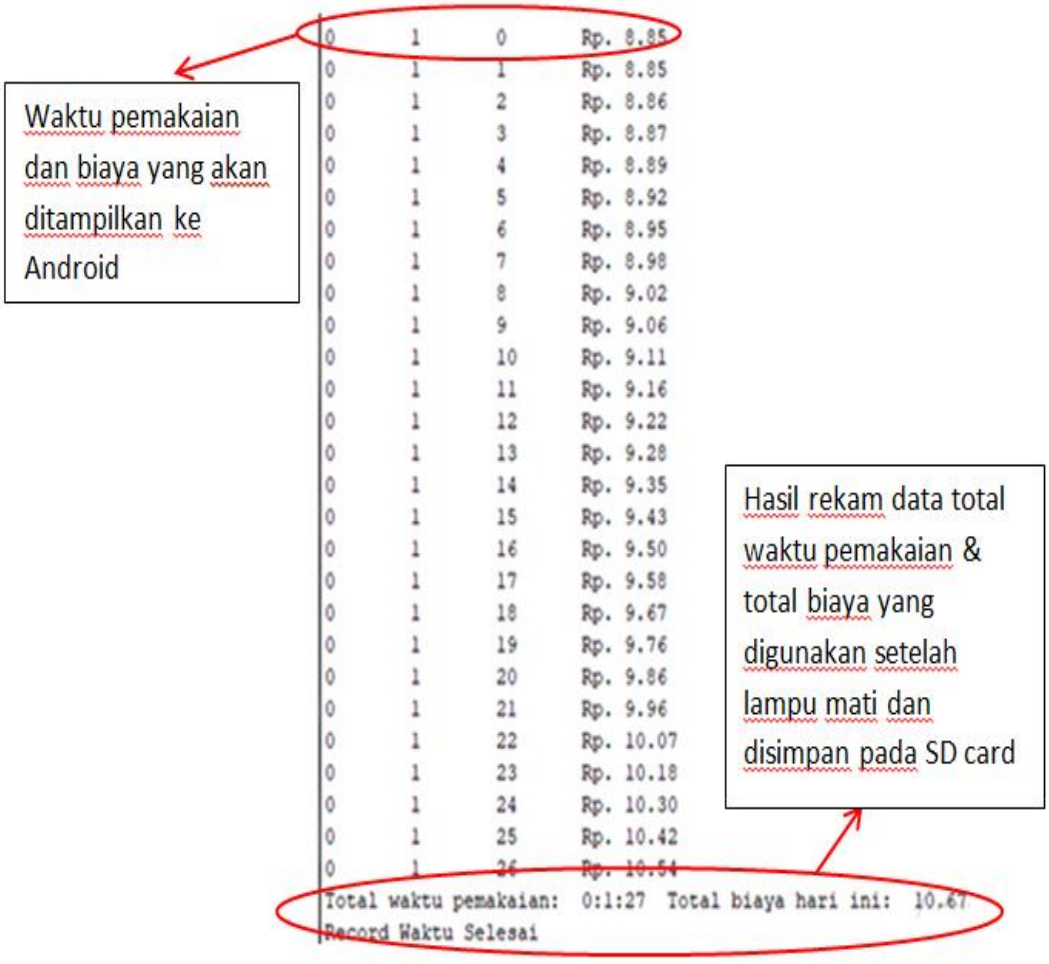


Gambar 4.25 Skematik Pemasangan Pin pada Modul SD Card

Untuk mengetahui waktu pemakaian lampu penulis membuat sistem *logger* yang dapat merekam waktu pemakaian hari ini dan daya yang terpakai saat lampu hidup. Dari waktu yang diperoleh serta daya yang dipakai dapat diketahui daya per-KWh sehingga diformulasikan kedalam bentuk biaya yang dipakai selama lampu hidup. Tujuan dibuatnya data *logger* ini agar tidak terjadi *over*

storage pada memori NodeMCU ESP8266. Selanjutnya agar data yang telah diolah akan tetap disimpan walaupun lampu yang digunakan telah dimatikan.

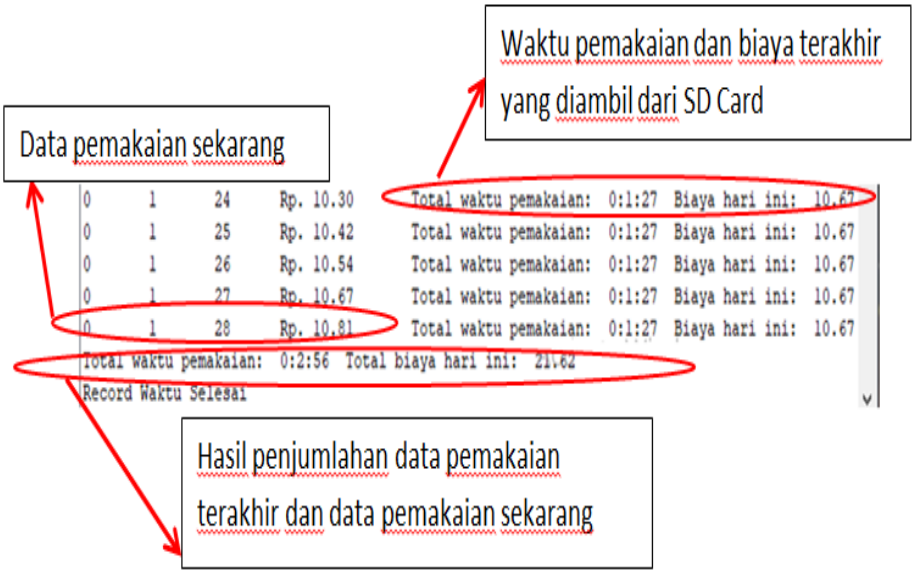
Data *logger* yang telah dibuat kemudian akan terus disimpan selama dua hari. Ketika waktu tersebut telah tercapai maka data yang terdapat pada SD Card akan terhapus dan diisi dengan data pemakaian yang baru. Untuk mengetahui apakah *logger* yang dibuat telah sesuai dengan yang diinginkan maka diakukan pengujian *logger* dengan serial monitor pada *software* Arduino IDE. Berikut gambar hasil *logger* yang dibuat ditunjukan pada Gambar 4.26.



Gambar 4.26 Pengujian Untuk Menyimpan Data *Logger*

Data yang ditampilkan merupakan *counter* yang diatur menjadi waktu pemakaian. Setelah lampu mati maka waktu pemakaian dan biaya yang digunakan akan direkam dan disimpan pada SD Card. Ini dimaksudkan untuk menghindari kehilangan data saat *power* AC fiting lampu mati. Ketika lampu dihidupkan maka NodeMCU ESP8266 mengambil data terakhir yang disimpan di SD Card kemudian dijumlahkan dengan data baru hasil pemakaian sekarang.

Data yang telah ada akan terus disimpan pada SD Card. Ketika lampu dihidupkan kembali maka *counter* waktu akan berjalan dari nol lagi. Sehingga waktu tersebut dapat digunakan sebagai waktu *realtime* lalu diolah menjadi biaya terkini dan ditampilkan pada aplikasi "Smart Fitting" yang telah dibuat. Berikut hasil operasi data dapat ditunjukan pada Gambar 4.27.



Gambar 4.27 Pengujian Merekam Waktu dan Biaya Pemakaian

Apabila lampu yang dikontrol melalui Android telah mati maka data pemakaian sekarang dijumlahkan dengan data pemakaian terakhir. Hasil yang telah dijumlahkan akan kembali disimpan pada SD Card. Data pada SD Card akan terus disimpan selama dua hari terhitung dari tanggal terakhir penyimpanan data, kemudian data akan dihapus secara otomatis dan menyimpan data yang baru.

Data yang disimpan pada SD *Card* berupa tanggal, waktu selama pemakaian, tegangan, arus, daya serta biaya yang digunakan. Berikut hasil penyimpanan data *logger* dapat dilihat pada Gambar 4.28

01/09/2019	waktu	tegangan	Arus	Daya	Biaya_Hari_ini	Waktu Kemarin	Biaya_kemarin
01/09/2019	0:10:2	226,6	0,09	12	Rp.3.02	0:10:1	Rp.3.21
01/09/2019	0:20:2	226,6	0,08	12	Rp.5.85	0:20:1	Rp.6.14
01/09/2019	0:30:3	226,7	0,08	12	Rp.8.77	0:30:1	Rp.9.82
01/09/2019	0:40:1	226,6	0,08	12	Rp.11.70	0:40:2	Rp.12.83
01/09/2019	0:50:1	226,6	0,08	12	Rp.14.62	0:50:2	Rp.15.20
01/09/2019	1:00:1	226,8	0,08	12	Rp.17.55	1:00:2	Rp.18.13
01/09/2019	1:10:1	226,8	0,08	12	Rp.20.47	1:10:2	Rp.21.05
01/09/2019	1:20:1	226,5	0,08	13	Rp.25.35	1:20:2	Rp.25.97
01/09/2019	1:30:3	226,8	0,09	12	Rp.26.32	1:30:1	Rp.26.61
01/09/2019	1:40:2	226,8	0,08	12	Rp.29.25	1:40:1	Rp.29.53
01/09/2019	1:50:2	226,7	0,09	13	Rp.34.85	1:50:1	Rp.32.46
01/09/2019	2:00:2	226,7	0,09	12	Rp.35.10	2:00:2	Rp.35.67
01/09/2019	2:10:2	225,8	0,09	12	Rp.38.02	2:10:2	Rp.38.60

Gambar 4.28 Hasil Penyimpanan Data *Logger*

Pada perekaman data *logger* terbagi menjadi 2 waktu, yang pertama setiap 5 detik sekali, hal ini bertujuan untuk mengantisipasi terjadinya pemadaman listrik secara tiba-tiba. Data yang disimpan berupa waktu yang digunakan serta biaya yang digunakan. Selanjutnya penyimpanan dilakukan setiap 10 menit sekali, ini digunakan untuk merekam data secara jelas seperti pada Gambar 4.28.

4.10. Pengujian Fiting Lampu Secara Keseluruhan

Setelah perakitan *hardware* mekanik dan *hardware* elektrik. Dilakukan pengujian alat secara keseluruhan. Pengujian dilakukan pada setiap fiting lampu dan terhubung dengan Android.

4.10.1. Pengujian *ON/OFF* Lampu

Untuk melakukan kontrol lampu maka menggunakan menu *ON/OFF* pada Android. Berikut hasil pengujian kontrol hidup/mati lampu:

a. Hasil pengujian mematikan lampu 1 dapat dilihat pada Gambar 4.29.

Gambar 4.29 Hasil Pengujian Kontrol Lampu 1

b. Hasil pengujian mematikan lampu 2 dapat dilihat pada Gambar 4.30.

Gambar 4.30 Hasil Pengujian Kontrol Lampu 2

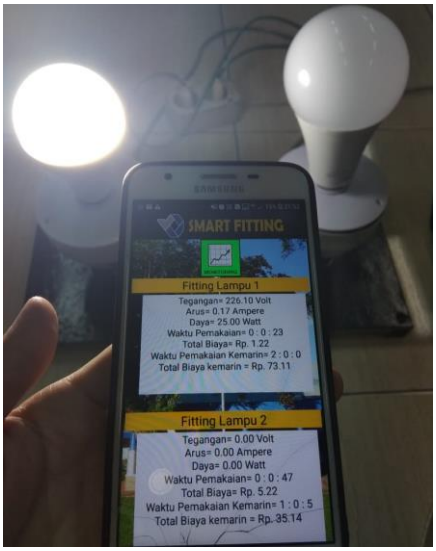
Pengujian untuk menghidupkan dan mematikan lampu dapat berjalan dengan baik. Pada pengujian ini menggunakan *mode* normal. Pada Android juga terdapat *display* yang menunjukan bahwa lampu dalam keadaan *ON* atau *OFF*.

4.10.2. Pengujian *Monitoring* Tegangan, Arus, Daya, Waktu dan Biaya

Untuk melakukan *monitoring* tegangan, arus, daya, total waktu dan biaya pemakaian lampu maka gunakanlah menu *monitoring* pada Android. *Monitoring*

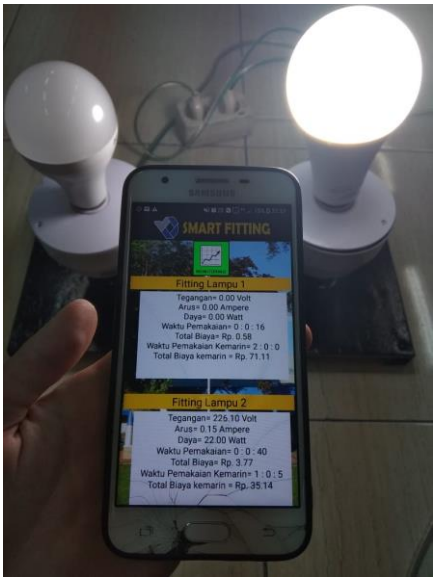
dilakukan pada setiap lampu baik secara masing-masing maupun secara keseluruhan. Berikut hasil pengujian *monitoring* pada fiting lampu.

a. Hasil pengujian monitotring lampu 1 dapat dilihat pada Gambar 4.31.



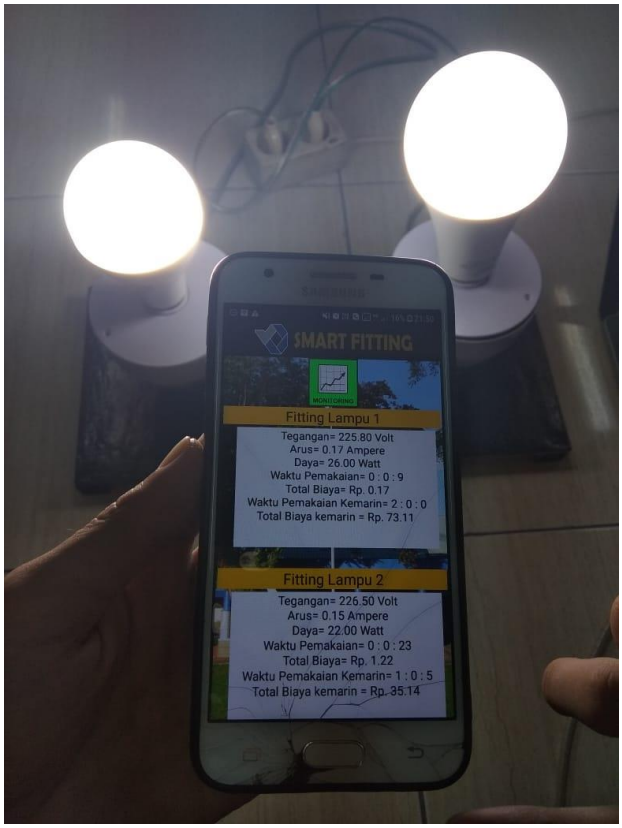
Gambar 4.31 Hasil Pengujian *Monitoring* Lampu 1

b. Hasil pengujian monitotring lampu 2 dapat dilihat pada Gambar 4.32.



Gambar 4.32 Hasil Pengujian *Monitoring* Lampu 2

c. Hasil pengujian *monitoring* semua lampu dapat dilihat pada Gambar 4.33.



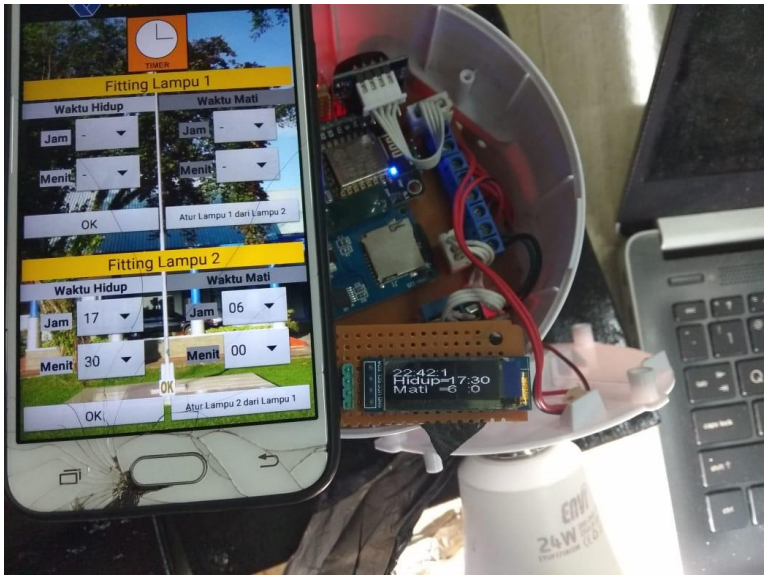
Gambar 4.33 Hasil Pengujian Monitotring Lampu Secara Bersamaan

Pada *monitoring* yang dilakukan menggunakan lampu 24 watt dan lampu 27 watt. Berdasarkan pengujian terhadap masing-masing lampu telah mendapatkan hasil yang baik. *Monitoring* dapat dilakukan pada setiap fiting lampu secara terpisah maupun secara bersamaan tergantung fiting lampu yang tersambung pada Android. Pada menu *monitoring* juga dapat ditampilkan waktu pemakaian serta biaya yang digunakan pada hari ini dan satu hari sebelumnya.

4.10.3. Pengujian *Timer* pada Lampu

Untuk melakukan kontrol *timer* hidup dan mati lampu maka menggunakan menu *timer* pada aplikasi “Smart Fitting” di Android. Berikut hasil pengujian kontrol *timer* hidup/mati lampu:

a. Hasil pengujian pengaturan *timer* lampu dapat dilihat pada Gambar 4.34.



Gambar 4.34 Ujicoba Pengaturan *timer*

b. Hasil pengujian *timer* menghidupkan dan mematikan lampu dapat dilihat pada Gambar 4.35.

Gambar 4.35 Ujicoba *Timer* Menghidupkan dan Mematikan Lampu

Dalam pengujian kontrol *timer* pada fiting lampu dilakukan pengujian untuk menghidupkan dan mematikan lampu dengan jadwal yang telah ditentukan. Pengujian dapat dilakukan pada masing-masing secara terpisah maupun secara bersamaan pada setiap fiting lampu. Dari pengujian yang dilakukan mendapatkan hasil yang telah sesuai dengan yang diinginkan. Untuk memastikan waktu yang diatur telah suesuai dengan yang diinginkan maka waktu tersebut dapat ditampilkan pada OLED 0.91.

4.10.4. Pengujian Topologi *Tree*

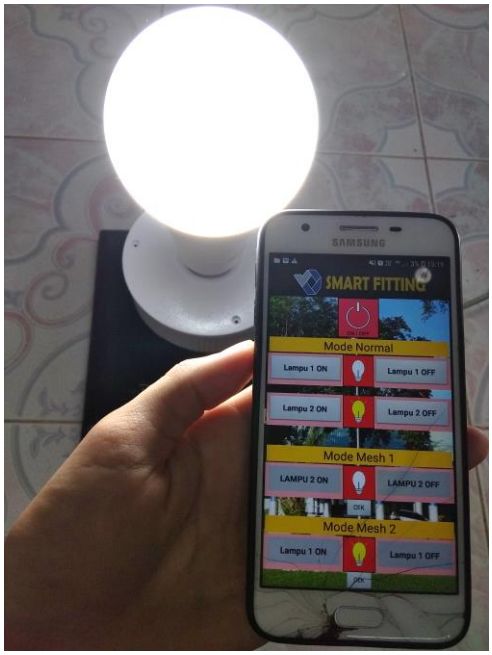
Untuk mengetahui apakah topologi *tree* yang digunakan dapat berjalan dengan baik maka dilakukan pengujian dengan "Menu Kontrol" yang dibuat. Pengujian dilakukan dengan menghidupkan "Lampu 1" dari "Lampu 2" dan sebaliknya, yakni menghidupkan "Lampu 2" dari "Lampu 1". Berikut hasil pengujian kontrol *timer* hidup/mati lampu.

Pada pengujian topologi *tree* ini juga dilakukan pengujian kemampuan komunikasi data antrar fiting lampu. Pengujian dilakukan pada rungan terbuka terbuka (tanpa penghalang) dan dalam ruangan tertutup (dengan penghalang). Untuk melihat kemampuan komunikasi data antar fiting lampu maka dilakukan pengambilan data jarak efektif komunikasi data. Berikut hasil percobaan yang dilakukan pada Tabel 4.14.

Tabel 4.14 Hasil Pengujian Jarak Efektif Komunikasi Antar Fiting Lampu

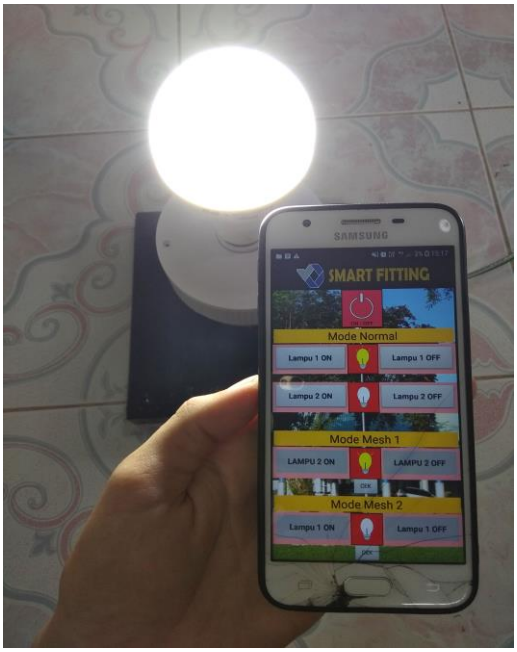
No	Jarak	Tanpa Penghalang	Dengan Penghalang
1.	5 Meter	Ya	Ya
2.	10 Meter	Ya	Ya
3.	15 Meter	Ya	Tidak
4.	20 Meter	Tidak	Tidak
5.	25 Meter	Tidak	Tidak
6.	30 Meter	Tidak	Tidak

a. Hasil pengujian topologi *tree* lampu 1 dapat dilihat pada Gambar 4.36.



Gambar 4.36 Topologi *Tree* Lampu 1

b. Hasil pengujian topologi *tree* lampu 2 dapat dilihat pada Gambar 4.37.



Gambar 4.37 Topologi *Tree* Lampu 2

Pada saat pengujian *mode tree* lampu 1 maka notifikasi lampu akan menyala sama seperti Gambar 4.36. Begitupun sebaliknya saat pengujian *mode tree* lampu 2 maka notifikasi lampu akan menyala sama seperti Gambar 4.37. Pemberian instruksi dari Android harus sesuai dengan kondisi, yang dimana topologi tree harus dilakukan jika fiting lampu yang dapat terhubung ke Android hanya satu buah fiting lampu sedangkan satu buah fiting lampu lainnya tidak dapat terhubung ke *hotspot* Android karena jaraknya melebihi jarak yang dapat dijangkau. Jikalau kedua fiting lampu terhubung ke *hotspot* Android, dan pengguna memberikan instruksi untuk menjalankan metode *tree*, maka saat salah satu fiting lampu yang diberikan instruksi untuk menjalankan instruksi topologi *tree* akan berganti *mode* dari *mode station* ke *mode access point (server)*. Fiting lampu *mode access point (server)* ini akan terus berlangsung sampai fiting lampu *access point* (*server*) terhubung ke fiting lampu *mode station* (*client*).

Lalu bagaimanakah caranya agar fiting lampu *Mode access point* (*server*) dapat terhuung ke fiting lampu *mode station* (*client*) sedangkan fiting lampu *mode station* telah terhubung ke *hotspot* Android? Solusi yang dapat dilakukan adalah menunggu sampai fiting lampu *mode access point (server)* menjadi *mode station (client)* kembali ataupun mematikan *hotspot* Android sehingga koneksi antara fiting lampu *mode station* (*client*) dengan Android terputus dan fiting lampu *mode station* mencari *hotspot* yang sesuai dengan program, sehingga fiting lampu *mode station* (*client*) dapat terhubung ke fiting lampu *mode access point* (*server*) dan fiting lampu *mode access point* (*server*) dapat melakukan pengiriman instruksi ke fiting lampu *mode station* (*client*).

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil pengujian dan analisa terhadap fungsi alat pada proyek akhir dengan judul "Kontrol dan *Monitoring* Fiting Lampu Berbasis Android" ini maka dapat ditarik kesimpulan sebagai berikut:

1. Pengujian tegangan yang diukur oleh sensor PZEM-004T dan dibandingkan dengan alat ukur mempunyai hasil yang sangat baik dengan persentase *error* maksimal 0,67%
2. Pengujian arus yang diukur oleh sensor PZEM-004T dan dibandingkan dengan alat ukur mempunyai hasil yang baik dengan persentase *error* maksimal pengujian lampu 12 watt sebesar 1,98%, pengujian lampu 18 watt sebesar 0,81%, pengujian lampu 24 watt sebesar 4,34%, pengujian lampu 27 watt sebesar 2,14% dan pengujian lampu 30 watt sebesar 1,45%.
3. Pengujian daya yang diukur oleh sensor PZEM-004T dan dibandingkan dengan alat ukur mempunyai hasil yang baik dengan persentase *error* maksimal 2,6 %.
4. Penghitungan nilai *power factor* maksimal yang didapat adalah 0,963
5. Pengiriman data dari fiting lampu ke Android mempunyai *delay* karena kemampuan pengiriman data serta mencegah penumpukan data.
6. Pengujian pada menu on/off dapat berfungsi dengan baik, sehingga dapat mematikan lampu 1 dan lampu 2.
7. Nilai tegangan, arus, daya dan biaya dapat ditampilkan pada menu *monitoring* sehingga dapat berfungsi sesuai keinginan.
8. Pengujian pada pengaturan timer lampu 1 dan lampu 2 dapat berfungsi dengan baik dan waktu yang diatur dapat ditampilkan pada OLED 0.91
9. Penyimpanan pada data *logger* dapat dilakukan setiap 10 menit sekali dengan data yang disimpan berupa total biaya dan total waktu pemakaian lampu pada hari ini dan satu hari sebelumnya.

10. Pengujian topologi *star* dapat berfungsi dengan baik yang mempunyai jarak efektif komunikasi antara fiting lampu dan Android kurang dari 30 meter tanpa halangan serta 15 meter dengan halangan.
11. Pengujian topologi *tree* dapat berfungsi dengan baik yang mempunyai jarak efektif komunikasi antar fiting lampu 1 dan fiting lampu 2 kurang dari 15 meter tanpa halangan serta 10 meter dengan halangan.

5.2. Saran

Apabila alat ini dikembangkan lebih lanjut, adapun fungsi yang perlu diperbaiki dan ditambahkan antara lain :

a. Proses pengiriman data dengan topologi jaringan yang lain.
b. Menggunakan piranti yang dapat mengontrol jarak yang lebih jauh.
c. Dapat menggunakan beban yang lebih besar.
d. Dapat membatasi energi yang digunakan dalam satu hari.

DAFTAR PUSTAKA

[1] Mahendri Saputra, "Perancangan Sistem Kendali Jarak Jauh Piranti Elektronika Berbasis Web Menggunakan Raspberry PI 3 Model B Dengan Menerapkan Konsep Internet of Things Untuk Aplikasi Rumah Pintar," *Tugas akhir*, vol. 1, p. 1, 2017.

[2] C. K. Das, M Sanaullah, H. M. G. Sarower, and M. M. Hassan, "Development of Cell Phone based Remote Control System; an Efective Switching System for Controlling Home and Office Appliances," *Electrical & Computer Sciences* , vol. 9, no. 10, 2009.

[3] F Novaldy Steven and Anggrainy K'K, "Stop Kontak Pintar Berbasis Android," *Tugas Akhir*, 2018.

[4] Zakaria. (2019, September) Topologi Jaringan Komputer – Pengertian, Jenis, Kelebihan dan Kekurangannya. [Online]. https://www.nesabamedia.com/topologi-jaringan-komputer/

[5] Hermawan. (2019, September) Pengertian Topologi Star Beserta Kelebihan dan Kekurangannya. [Online]. https://www.nesabamedia.com/topologi-star/

[6] Hermawan. (2019, Februari) Pengertian Topologi Tree Beserta Kelebihan dan Kekurangannya. [Online]. https://www.nesabamedia.com/topologi-tree/

[7] Temy Nusa, "Sistem *Monitoring* Konsumsi Energi Listrik Secara Real Time Berbasis Mikrokontroler," *Teknik Elektro dan Komputer*, vol. 4, 2015.

[8] Moch. Azam. (2017, Agustus) Pengertian Android Beserta Sejarah, Kelebihan dan Kekurangannya. [Online]. https://www.nesabamedia.com/pengertian-Android-beserta-kelebihan-dan-kekurangannya/

[9] Wikipedia. (2019, Mei) Android (sistem operasi). [Online]. https://id.wikipedia.org/wiki/Android_(sistem_operasi)

[10] Bluumi. (2018, November) Android Pie, la nouvelle actualisation Android. [Online]. https://bluumi.net/actualisation-Android-pie/portada-post-Android-pie-1/

[11] Danila Loginiv. (2018, Desember) Quick start with NodeMCU v3 (ESP8266), Arduino ecosystem and PlatformIO IDE. [Online]. https://medium.com/@loginov_rocks/quick-start-with-nodemcu-v3-esp8266-arduino-ecosystem-and-platformio-ide-b8415bf9a038

[12] Handson Technology. (2017, Maret) ESP8266 Lua NodeMCU User Guide. [Online]. https://www.handsontec.com/pdf_learn/esp8266-V10.pdf

[13] Jimmi Sitepu. (2019, April) Membaca Sensor PZEM-004t dengan nodemcu Arduino. [Online]. https://mikroavr.com/sensor-pzem-004t-arduino/

[14] InnovatorsGuru. (2018, November) PZEM-004T Module for Designing Smart Meter - InnovatorsGuru. [Online]. https://innovatorsguru.com/ac-digital-multifunction-meter-using-pzem-004t/

[15] WikiKomponen. (2018, Maret) Teori Dasar dan Pengertian Komponen Solid State Relay. [Online]. https://www.wikikomponen.com/teori-dasar-dan-pengertian-komponen-solid-state-relay/#Pengertian_Solid_State_Relay_SSR

[16] Omron Electronics LLC. (2001, Februari) OMRON Global. [Online]. https://www.openhacks.com/uploadsproductos/g3mb-ssr-datasheet.pdf

[17] Components101. (2018, Maret) DS3231 RTC Module Pinout, Configuration, Example Circuit & Datasheet. [Online]. https://components101.com/modules/ds3231-rtc-module-pinout-circuit-datasheet

[18] Jimmi Sitepu. (2018, Oktober) Cara Mudah Membuat Jam Digital dengan RTC DS3231 - mikroavr.com. [Online]. https://mikroavr.com/jam-digital-arduino-ds3231/

[19] Agus Faudin. (2017, Desember) Tutorial Arduino mengakses module RTC DS3231. [Online]. https://www.nyebarilmu.com/tutorial-arduino-mengakses-module-rtc-ds3231/

[20] mybotic. (2018, Januari) Tutorial to Interface OLED 0.91inch 128x32 With Arduino UNO. [Online]. https://www.instructables.com/id/Tutorial-to-Interface-OLED-091inch-128x32-With-Ard/

[21] Nyebarilmu.com. (2017, Juli) Rangkaian Elektronika Power Supply. [Online]. https://www.nyebarilmu.com/rangkaian-elektronika-power-supply/

[22] INT-EL ELEKTRONİK SAN. VE TİC. LTD. Sti. 5V 700mA AC-DC-Step-Down Module at Affordable Price- Direnc.net. [Online]. https://www.direnc.net/5v-700ma-ac-dc-step-down-modul

[23] Nyebarilmu.com. (2018, April) Cara mengakses module micro SD menggunakan Arduino. [Online]. https://www.nyebarilmu.com/cara-mengakses-module-micro-sd-menggunakan-arduino/https://www.nyebarilmu.com/cara-mengakses-module-micro-sd-menggunakan-arduino/

[24] mybotic. (2017, Mei) Micro SD Card Tutorial. [Online]. https://www.instructables.com/id/Micro-SD-Card-Tutorial/

[25] Abdul Kadir, *Pemrograman Arduino & Android Menggunakan App Inventor*. Jakarta: PT Elex Media Komputindo, 2017.

[26] Karen. (2017, Agustus) About our new MIT App Inventor. [Online]. https://appinventor.mit.edu/explore/blogs/karen/2017/08/about.html

LAMPIRAN

LAMPIRAN 1

Daftar Riwayat Hidup

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama Lengkap	: Yogie Saputra
Tempat & Tanggal Lahir	: Koba, 23 Juni 1998
Alamat	: Jalan Senang Hati Koba, Bangka Tengah, Bangka Belitung
	Telp : -
	Hp : 0853 7790 1497
	Email : ogiesa23@gmail.com
Jenis Kelamin	: Laki-laki
Agama	: Islam

2. Riwayat Pendidikan

SD Negeri 16 Pangkal Niur	Lulus 2010
SMP Negeri 3 Riau Silip	Lulus 2013
SMA Negeri 1 Kelapa	Lulus 2016

3. Pendidikan Non Formal

-

Sungailiat, 22 Agustus 2019

Yogie Saputra

DAFTAR RIWAYAT HIDUP

1. Data Pribadi

Nama Lengkap : Habi Alqadri

Tempat & Tanggal Lahir : Pangkalpinang, 1 Mei 1998

Alamat : Jalan Belinjo Bukit Merapin,
Pangkalpinang, Bangka Belitung

Telp : -

Hp : 0822 9444 4429

Email : habialq@gmail.com

Jenis Kelamin : Laki-laki

Agama : Islam

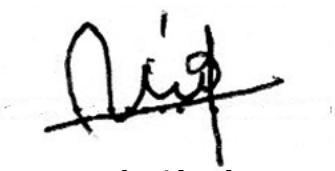
2. Riwayat Pendidikan

SD Negeri 15 Pangkalpinang	Lulus 2010
SMP Negeri 1 Pangkalpinang	Lulus 2013
SMK Negeri 2 Pangkalpinang	Lulus 2016

3. Pendidikan Non Formal

-

Sungailiat, 22 Agustus 2019



Habi Alqadri

LAMPIRAN 2

Program NodeMCU ESP8266

```
#include <ESP8266WiFi.h>

//PZEM
#include <SoftwareSerial.h> // Arduino IDE <1.6.6
#include <PZEM004T.h>

//RTC
#include <Wire.h>
#include "RTClib.h"

//EPROM
#include <EEPROM.h>

//OLED
#include <SSD1306.h>
//#include <splash.h>
SSD1306  display(0x3c, 4, 5);
//#define OLED_RESET 4
//Adafruit_SSD1306 display(OLED_RESET);
int x=0; //cek kondisi mesh apakah berhasil
int y=0; //counter waktu server
int Mode=0;
//Nama & Password Hotspot Android yang ingin
disambungkan
const char* ssid = "lampu11";
const char* password = "habi20199";
//const char* ssid1 = "lampu12";
//const char* password1 = "habi20190";
IPAddress IP(192,168,43,1);
IPAddress mask = (255, 255, 255, 0);
WiFiServer server(80);
```

```
WiFiClient client;
//PZEM
PZEM004T pzem(D4,D3);  // (RX,TX) connect to TX,RX of
PZEM
IPAddress ip(192,168,1,1);

//RTC
RTC_DS3231 rtc;

//relay
int relay = 16;

//Deklarasi Parsing Data untuk Timer Sendiri
 String dataIn;
 String dt[10];
 int pd;
 boolean parsing=false;

//Deklarasi Parsing Data untuk Timer Client
 String dataIn1;
 String dt1[10];
 int pd1;
 boolean parsing1=false;

//HARI PADA RTC
char daysOfTheWeek[7][12] = {"Sunday", "Monday",
"Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday"};

//////////////////////deklarasi timer/////////////
int JamOn;
```

```
 int MenitOn;
 int JamOff;
 int MenitOff;

//////////////////////deklarasi timerLain////////////
int JamOnLain;
 int MenitOnLain;
 int JamOffLain;
 int MenitOffLain;

//////////////////////////////DATA
SENSOR//////////////////////////////////
float v,i,p;

//////////////////////////VOID
STATION//////////////////////////
void ModeStation()
               {
              //Serial.print("Connecting to ");

                Serial.println(ssid);
                WiFi.begin(ssid, password);
              //  while (WiFi.status() != WL_CONNECTED)
{
              //    delay(500);
              //    Serial.print(".");
              //  }
              //  Serial.println("");
              //  Serial.println("WiFi connected");

                // Start the server
```

```
                server.begin();
                Serial.println(WiFi.localIP());
              }

//////////////////////////VOID ACCES
POINT//////////////////////////
void ModeAccesPoint()
              {
              IPAddress AP(192,168,43,1);
              IPAddress mask = (255, 255, 255, 0);
              WiFi.mode(WIFI_AP);
              WiFi.softAP("lampu12", "habi20190");
              WiFi.softAPConfig(AP, AP, mask);
              server.begin();
              Serial.print("IP: ");
Serial.println(WiFi.softAPIP());
              Serial.print("MAC:");
Serial.println(WiFi.softAPmacAddress());
              }

////////////////////////////////////////////VOID
SENSOR////////////////
void sensor()
             {
             v = pzem.voltage(ip);
            if (v < 0.0) v = 0.0;
//            Serial.print(v);Serial.print("V; ");


             i = pzem.current(ip);
            if (i < 0.0) i = 0.0;
```

```
//             if(i >= 0.00){
Serial.print(i);Serial.print("A; "); }


             p = pzem.power(ip);
            if (p < 0.0) p = 0.0;
//             if(p >= 0.0){
Serial.print(p);Serial.println("W; "); }
            }


//////////////////////////////VOID PHARSING untuk
Timer Sendiri//////////////
void parsingData()
            {
            int j=0;

            //kirim data yang telah diterima sebelumnya
            Serial.print("data masuk : ");
            Serial.print(dataIn);
            Serial.print("\n");

            //inisialisasi variabel, (reset isi
variabel)
            dt[j]="";
            //proses parsing data
            for(pd=1;pd<dataIn.length();pd++){
            //pengecekan tiap karakter dengan karakter
(#) dan (,)
            if ((dataIn[pd] == '#') || (dataIn[pd] ==
','))
```

```
            {
            //increment variabel j, digunakan untuk
merubah index array penampung
            j++;
            dt[j]="";       //inisialisasi variabel
array dt[j]
            }
            else
            {
            //proses tampung data saat pengecekan
karakter selesai.
            dt[j] = dt[j] + dataIn[pd];
            }
            }

            //kirim data hasil parsing
            Serial.print("data 1 : ");
            Serial.print(dt[0].toInt());
            Serial.print("\n");
            Serial.print("data 2 : ");
            Serial.print(dt[1].toInt());
            Serial.print("\n");
            Serial.print("data 3 : ");
            Serial.print(dt[2].toInt());
            Serial.print("\n");
            Serial.print("data 4 : ");
            Serial.print(dt[3].toInt());
            Serial.print("\n");
            Serial.print("data 5 : ");
            Serial.print(dt[4].toInt());
            Serial.print("\n\n");
```

```
            }

//////////////////////////////VOID PHARSING untuk
Timer Sendiri/////////////
void parsingData1()
            {
            int j1=0;

            //kirim data yang telah diterima sebelumnya
            Serial.print("data masuk : ");
            Serial.print(dataIn1);
            Serial.print("\n");

            //inisialisasi variabel, (reset isi
variabel)
            dt1[j1]="";
            //proses parsing data
            for(pd1=1;pd1<dataIn1.length();pd1++){
            //pengecekan tiap karakter dengan karakter
(#) dan (,)
            if ((dataIn1[pd1] == '#') || (dataIn1[pd1]
== ','))
            {
            //increment variabel j, digunakan untuk
merubah index array penampung
            j1++;
            dt1[j1]="";       //inisialisasi variabel
array dt[j]
            }
            else
            {
```

```
            //proses tampung data saat pengecekan
karakter selesai.
            dt1[j1] = dt1[j1] + dataIn1[pd1];
            }
            }

            //kirim data hasil parsing
            Serial.print("data 1 : ");
            Serial.print(dt1[0].toInt());
            Serial.print("\n");
            Serial.print("data 2 : ");
            Serial.print(dt1[1].toInt());
            Serial.print("\n");
            Serial.print("data 3 : ");
            Serial.print(dt1[2].toInt());
            Serial.print("\n");
            Serial.print("data 4 : ");
            Serial.print(dt1[3].toInt());
            Serial.print("\n");
            Serial.print("data 5 : ");
            Serial.print(dt1[4].toInt());
            Serial.print("\n\n");
            }


void setup() {
EEPROM.begin(512);
Serial.begin(115200);
delay(10);
pzem.setAddress(ip);
```

```
//Relay
pinMode(relay, OUTPUT);
digitalWrite(relay,LOW);

//OLED
//display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
//display.clearDisplay();
  display.init();

  display.flipScreenVertically();
  display.setFont(ArialMT_Plain_10);

//EPROM
  JamOn= EEPROM.read(0);
  MenitOn= EEPROM.read(1);
  JamOff= EEPROM.read(2);
  MenitOff= EEPROM.read(3);
  Serial.println("");
  Serial.println(JamOn);
  Serial.println(MenitOn);
  Serial.println(JamOff);
  Serial.println(MenitOff);
  delay(100);
ModeStation();
//RTC
if (! rtc.begin())
    {
    Serial.println("Couldn't find RTC");
    while (1);
    }
     rtc.adjust(DateTime(2019, 8, 20, 22, 27, 30));
```

```
if (rtc.lostPower())
    {
    Serial.println("RTC lost power, lets set the
time!");
    // following line sets the RTC to the date & time
this sketch was compiled
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
    // This line sets the RTC with an explicit date &
time, for example to set
    // January 21, 2014 at 3am you would call:
    // rtc.adjust(DateTime(2014, 1, 21, 3, 0, 0));
    }
}

///////////////////////////////////////////////////VOID
LOOP/////////////////////////////////////////////////
/
void loop() {

  Serial.println(Mode);
  sensor();
  Serial.println(WiFi.localIP());
  DateTime now = rtc.now();
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.println(now.second(), DEC);

    display.clear();
    display.setTextAlignment(TEXT_ALIGN_LEFT);
```

```
     display.setFont(ArialMT_Plain_24);
    display.drawString(0,0, String(now.hour(), DEC) );
    display.drawString(25,0, ":");
    display.drawString(30,0, String(now.minute(), DEC)
);
    display.drawString(55,0, ":");
    display.drawString(60,0, String(now.second(), DEC)
);
    display.drawString(0,20, "Hidup= " );
    display.drawString(70,20, String(JamOn) );
    display.drawString(95,20, ":" );
    display.drawString(100,20, String(MenitOn) );
    display.drawString(0,42, "Mati  = " );
    display.drawString(70,42, String(JamOff) );
    display.drawString(95,42, ":" );
    display.drawString(100,42, String(MenitOff) );
  display.display();
  if (now.hour() == JamOn && now.minute() == MenitOn &&
now.second() <=5){
    digitalWrite(relay,LOW);
    Serial.println("LAMPU MENYALA(TIMER)");
    }
  else  if(now.hour() == JamOff && now.minute() ==
MenitOff && now.second() <=5){
  digitalWrite(relay,HIGH);
  Serial.println("LAMPU MATI(TIMER)");
    }


  if (Mode==0)
  {
```

```
  Serial.print("Nilai Mode= ");
  Serial.println(Mode);

  char inChar;


////////////////////////////////////KONEKSI//////////
//////////
  if  (WiFiClient client = server.available())
     {
     if (!client) return;
     String req = client.readStringUntil('\r');
     Serial.println(req);
//     client.flush();

//////////////////////////////////////////variabel//////
////////////////
     float d=0;
     float biaya;

          ////////////////////////////////
KOMUNIKASI////////////////////////////////
          if (req.indexOf("/lampu1on") != -1)
                {
                 Serial.println("HIDUP");
                 digitalWrite(relay,LOW);
                 String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                 kondisi += "1";
                 kondisi += "</html>\n";
```

```
                client.print(kondisi);
                }
          else if (req.indexOf("/lampu1off") != -1)
                {
                Serial.println("MATI");
                digitalWrite(relay,HIGH);
                String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                kondisi += "0";
                kondisi += "</html>\n";
                client.print(kondisi);
                }
          else if (req.indexOf("/data") != -1)
                {
                //a=v;
                //b=i;
                //c=p;
                sensor();
                d=biaya;
                String data = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                data += "Tegangan=  ";
                data += (v);
                data += " Volt";
                data += "<br/>";
                data += "Arus=  ";
                data += (i);
                data += " Ampere";
                data += "<br/>";
```

```
                data += "Daya=  ";
                data += (p);
                data += " Watt";
                data += "<br/>";
                data += "Biaya Pemakaian= Rp.";
                data += (d);
                data += "</html>\n";
                client.print(data);
                delay(1);
                }

          //  Serial.print(dataIn);
            else if (req.indexOf("/*") != -1)
                {
                dataIn += req;
                parsing = true;

                    if(parsing)
                    {
                    parsingData();
                    parsing=false;
                    dataIn="";
                    JamOn = dt[1].toInt();
                    MenitOn = dt[2].toInt();
                    JamOff = dt[3].toInt();
                    MenitOff = dt[4].toInt();
                    EEPROM.put(0, JamOn);
EEPROM.commit();
                    EEPROM.put(1, MenitOn);
EEPROM.commit();
```

```
                    EEPROM.put(2, JamOff);
EEPROM.commit();
                    EEPROM.put(3, MenitOff);
EEPROM.commit();
                    String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                    kondisi += "1";
                    kondisi += "</html>\n";
                    client.print(kondisi);
                    Serial.print("Jam Hidup= ");
                    Serial.println(JamOn);
                    Serial.print("Menit Hidup= ");
                    Serial.println(MenitOn);
                    Serial.print("Jam Mati= ");
                    Serial.println(JamOff);
                    Serial.print("Menit Mati= ");
                    Serial.println(MenitOff);
                    }
               }

          else if (req.indexOf("/!") != -1)
               {
               dataIn1 += req;
               parsing1 = true;

                    if(parsing1)
                    {
                    parsingData1();
                    parsing1=false;
                    dataIn1="";
```

```
                    JamOnLain = dt1[1].toInt();
                    MenitOnLain = dt1[2].toInt();
                    JamOffLain = dt1[3].toInt();
                    MenitOffLain = dt1[4].toInt();
                    Serial.print("Jam Hidup Client= ");
                    Serial.println(JamOnLain);
                    Serial.print("Menit Hidup Client=
");
                    Serial.println(MenitOnLain);
                    Serial.print("Jam Mati Client= ");
                    Serial.println(JamOffLain);
                    Serial.print("Menit Mati Client=
");
                    Serial.println(MenitOffLain);
                    }
                    Mode=3;
                    y=0;
                }

            else if (req.indexOf("/kondisi") != -1)
                {
                if (v>0.0){
                Serial.println("Lampu ON");
                String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                kondisi += "1";
                kondisi += "</html>\n";
                client.print(kondisi);
                          }
                else if (v<=0.0){
```

```
                Serial.println("Lampu OFF");
                String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                kondisi += "0";
                kondisi += "</html>\n";
                client.print(kondisi);
                         }
                }
          else if (req.indexOf("/cek") != -1)
                {
                    if (x==1)
                        {
                        Serial.println("Mengirim 2");
                        String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                        kondisi += "2";
                        kondisi += "</html>\n";
                        client.print(kondisi);
                        x=0;
                        }
                    else if (x==2)
                        {
                        Serial.println("Mengirim 3");
                        String kondisi = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                        kondisi += "3";
                        kondisi += "</html>\n";
                        client.print(kondisi);
```

```
                        x=0;
                        }
                  }

          else if (req.indexOf("/lampulainoff") != -1)
                  {
                    Mode=1;
                    y=0;
                  }
          else if (req.indexOf("/lampulainon") != -1)
                  {
                    Mode=2;
                    y=0;
                  }
       }

 else if   (client.connect(IP, 80))
     {
     if (!client) return;
     String req = client.readStringUntil('\r');
     Serial.println(req);
//     client.flush();

//////////////////////////////////////////variabel//////
////////////////
     float d=0;
     float biaya;

          ////////////////////////////////
KOMUNIKASI////////////////////////////////
          if (req.indexOf("/lampu1on") != -1)
```

```
                {
                  Serial.println("HIDUP");
                  digitalWrite(relay,LOW);
                }
          else if (req.indexOf("/lampu1off") != -1)
                {
                Serial.println("MATI");
                digitalWrite(relay,HIGH);
                }
          else if (req.indexOf("/data") != -1)
                {
                //a=v;
                //b=i;
                //c=p;
                sensor();
                d=biaya;
                String data = "HTTP/1.1 200
OK\r\nContent-Type: text/html\r\n\r\n<!DOCTYPE
HTML>\r\n<html>\r\n ";
                data += "Tegangan=  ";
                data += (v);
                data += " Volt";
                data += "<br/>";
                data += "Arus=  ";
                data += (i);
                data += " Ampere";
                data += "<br/>";
                data += "Daya=  ";
                data += (p);
                data += " Watt";
                data += "<br/>";
```

```
                data += "Biaya Pemakaian= Rp.";
                data += (d);
                data += "</html>\n";
                client.print(data);
                }

          //  Serial.print(dataIn);
            else if (req.indexOf("/*") != -1)
                {
                dataIn += req;
                parsing = true;

                    if(parsing)
                    {
                    parsingData();
                    parsing=false;
                    dataIn="";
                    JamOn = dt[1].toInt();
                    MenitOn = dt[2].toInt();
                    JamOff = dt[3].toInt();
                    MenitOff = dt[4].toInt();
                    EEPROM.put(0, JamOn);
EEPROM.commit();
                    EEPROM.put(1, MenitOn);
EEPROM.commit();
                    EEPROM.put(2, JamOff);
EEPROM.commit();
                    EEPROM.put(3, MenitOff);
EEPROM.commit();
                    Serial.print("Jam Hidup= ");
                    Serial.println(JamOn);
```

```
                    Serial.print("Menit Hidup= ");
                    Serial.println(MenitOn);
                    Serial.print("Jam Mati= ");
                    Serial.println(JamOff);
                    Serial.print("Menit Mati= ");
                    Serial.println(MenitOff);
                    }
                }

          else if (req.indexOf("/lampu2off") != -1)
                {
                  Mode=1;
                }
          else if (req.indexOf("/lampu2on") != -1)
                {
                  Mode=2;
                }



//          client.flush();
      }
  }


//else if (Mode==1)
//  {
//    ModeAccesPoint();
////    WiFi.mode(WIFI_AP);
////  WiFi.softAP("lampu12", "habi20190");
////  WiFi.softAPConfig(AP, AP, mask);
```

```
////  server.begin();
////  Serial.print("IP: ");
Serial.println(WiFi.softAPIP());
////  ModeAccesPoint();
//  delay(3000);
//  Serial.print("Nilai Mode= ");
//  Serial.print(Mode);
//  Serial.println("---> Server Mematikan Lampu");
//  WiFiClient client = server.available();
////  if (!client) {return;}
//  delay(12000);
////  Serial.print("Mengirim Perintah ON");
////  Serial.println(client.println(request + "ca" +
"\r"));
//    Serial.println(client.println("/lampu1off \r"));
//    delay(1000);
//        Serial.println(client.println("/lampu1off
\r"));
//        delay(1000);
//            Serial.println(client.println("/lampu1off
\r"));
//            delay(1000);
//
Serial.println(client.println("/lampu1off \r"));
//                delay(1000);
//
Serial.println(client.println("/lampu1off \r"));
//    Serial.println(client.println("/lampu1off \r"));
//   WiFi.mode(WIFI_STA);
//  Mode=0;
//  }
```

```
//
//else if (Mode==2)
//  {
//       ModeAccesPoint();
////  WiFi.mode(WIFI_AP);
////  WiFi.softAP("lampu12", "habi20190");
////  WiFi.softAPConfig(AP, AP, mask);
////  server.begin();
//delay(3000);
////  Serial.print("IP: ");
Serial.println(WiFi.softAPIP());
////  ModeAccesPoint();
//  Serial.print("Nilai Mode= ");
//  Serial.print(Mode);
//  Serial.println("---> Server Menghidupkan Lampu");
//  WiFiClient client = server.available();
//  delay(12000);
////  if (!client) {return;}
//  Serial.print("Mengirim Perintah OFF");
////  Serial.println(client.println(request + "ca" +
"\r"));
////  Serial.println(client.println("/lampu1on \r"));
//    Serial.println(client.println("/lampu1on \r"));
//    delay(1000);
//        Serial.println(client.println("/lampu1on
\r"));
//        delay(1000);
//            Serial.println(client.println("/lampu1on
\r"));
//            delay(1000);
```

```
//
Serial.println(client.println("/lampu1on \r"));
//                delay(1000);
//
Serial.println(client.println("/lampu1on \r"));
//  WiFi.mode(WIFI_STA);
//  Mode=0;
//  }
 if (Mode==1){
  delay(1000);
  WiFi.mode(WIFI_AP);
  WiFi.softAP("lampu11", "habi20199");
  WiFi.softAPConfig(IP, IP, mask);
  server.begin();
  WiFiClient client = server.available();
  if (!client)
    {
      y++;
      Serial.println(y);
      if(y<=20)
          {
            return;
          }
    }
  Serial.print("Byte sent to the station: ");
  Serial.println(client.println("/lampu1off \r"));
  delay(300);
  delay(300);
  WiFi.mode(WIFI_STA);
  ModeStation();
  Mode=0;
```

```
  x=1;
  if (y>=21)
    {
      delay(300);
      WiFi.mode(WIFI_STA);
      ModeStation();
      Mode=0;
      x=0;
    }
  }



   if (Mode==2){
  delay(1000);
  WiFi.mode(WIFI_AP);
  WiFi.softAP("lampu11", "habi20199");
  WiFi.softAPConfig(IP, IP, mask);
  server.begin();
WiFiClient client = server.available();
   if (!client)
    {
      y++;
      Serial.println(y);
      if(y<=20)
          {
            return;
          }
    }
  Serial.print("Byte sent to the station: ");
  Serial.println(client.println("/lampu1on \r"));
```

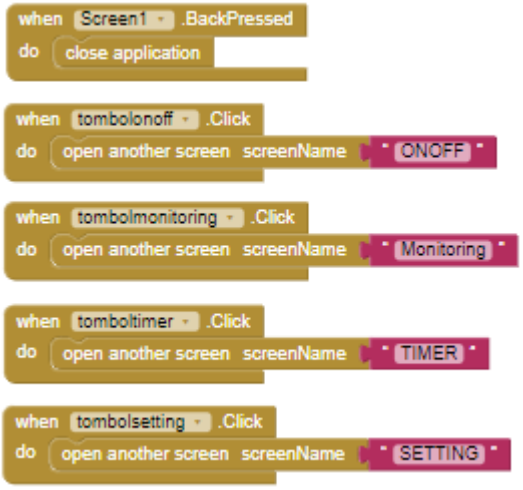
```
  delay(300);
  WiFi.mode(WIFI_STA);
  ModeStation();
  Mode=0;
  x=2;
  if (y>=21)
    {
      delay(300);
      WiFi.mode(WIFI_STA);
      ModeStation();
      Mode=0;
      x=0;
    }
  }

if (Mode==3){
  delay(1000);
  WiFi.mode(WIFI_AP);
//    server.flush();
  WiFi.softAP("lampu11", "habi20199");
  WiFi.softAPConfig(IP, IP, mask);
  server.begin();
  WiFiClient client = server.available();
  if (!client)
    {
      y++;
      Serial.println(y);
      if(y<=20)
          {
            return;
          }
```

```
    }
  Serial.print("Byte sent to the station: ");
  String TimerLain = "/*0,";
         TimerLain += (JamOnLain);
         TimerLain += ",";
         TimerLain += (MenitOnLain);
         TimerLain += ",";
         TimerLain += (JamOffLain);
         TimerLain += ",";
         TimerLain += (MenitOffLain);
         TimerLain += "#";
         Serial.println(client.print(TimerLain));
  if (y>=21)
    {
      delay(300);
      WiFi.mode(WIFI_STA);
      ModeStation();
      Mode=0;
    }
  delay(300);
  WiFi.mode(WIFI_STA);
  ModeStation();
  Mode=0;
  }
  client.flush();
}
```

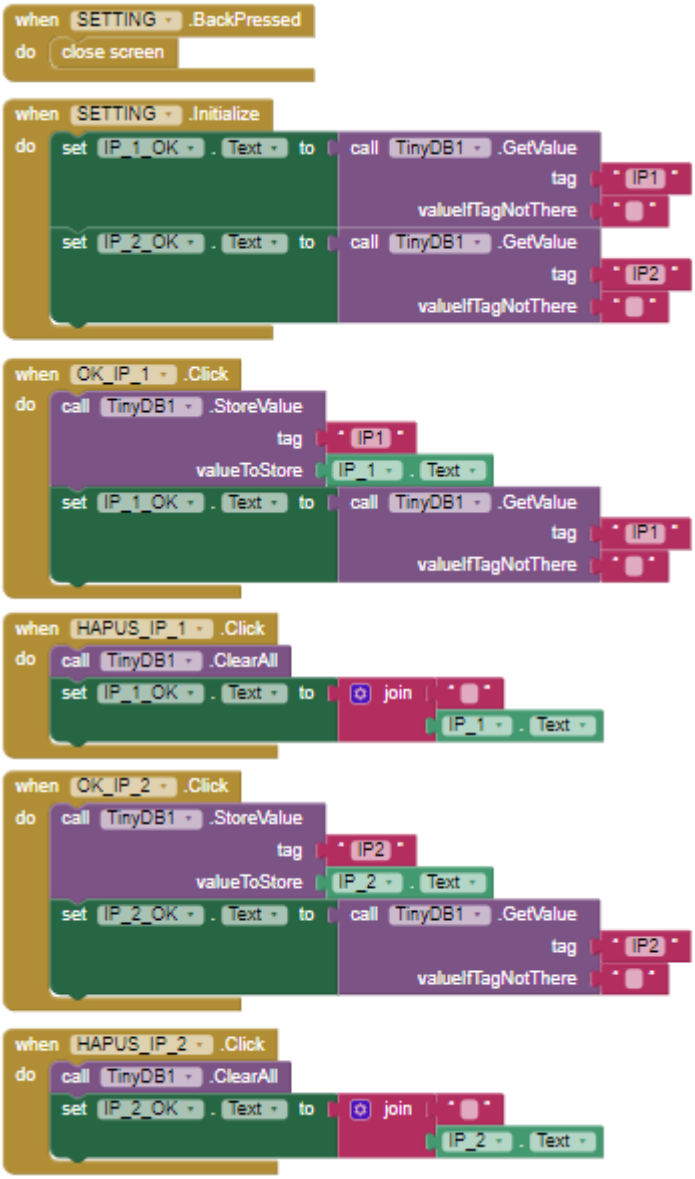
LAMPIRAN 3

Program Aplikasi

Menu Utama

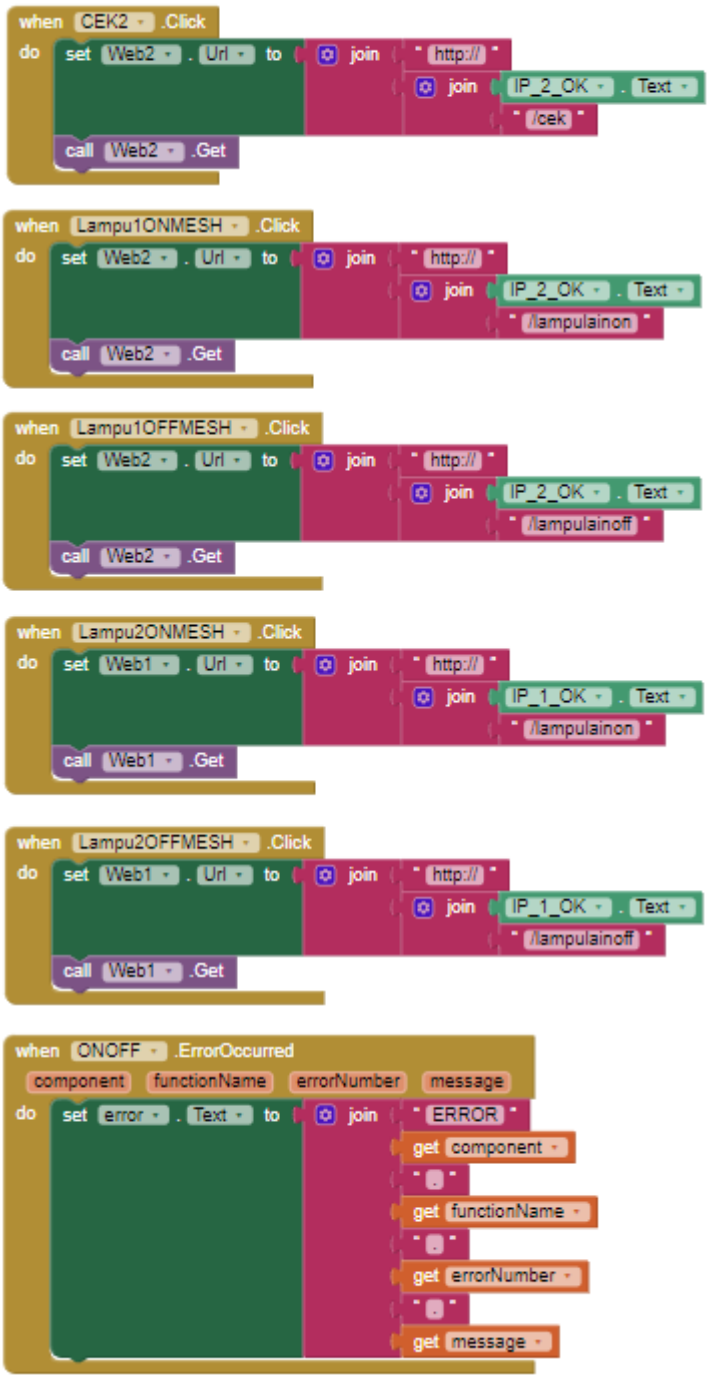

when Screen1 .BackPressed
do close application
when tombolonoff .Click
do open another screen screenName " ONOFF "
when tombolmonitoring .Click
do open another screen screenName " Monitoring "
when tomboltimer .Click
do open another screen screenName " TIMER "
when tombolsetting .Click
do open another screen screenName " SETTING "

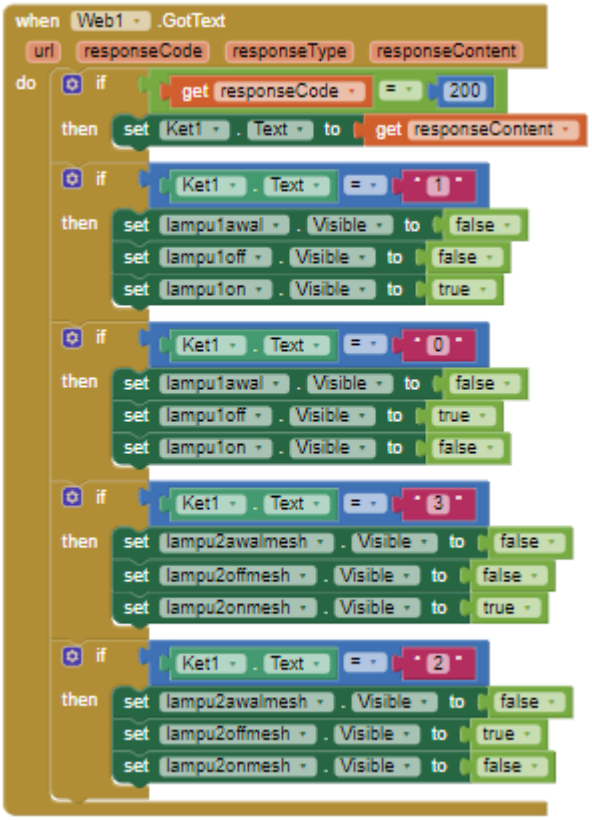
Menu Setting

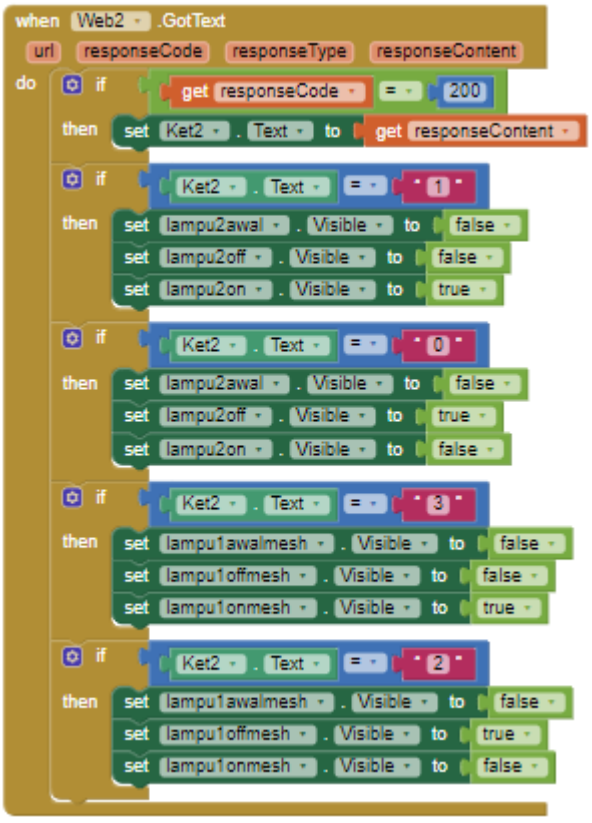


Menu ON/OFF

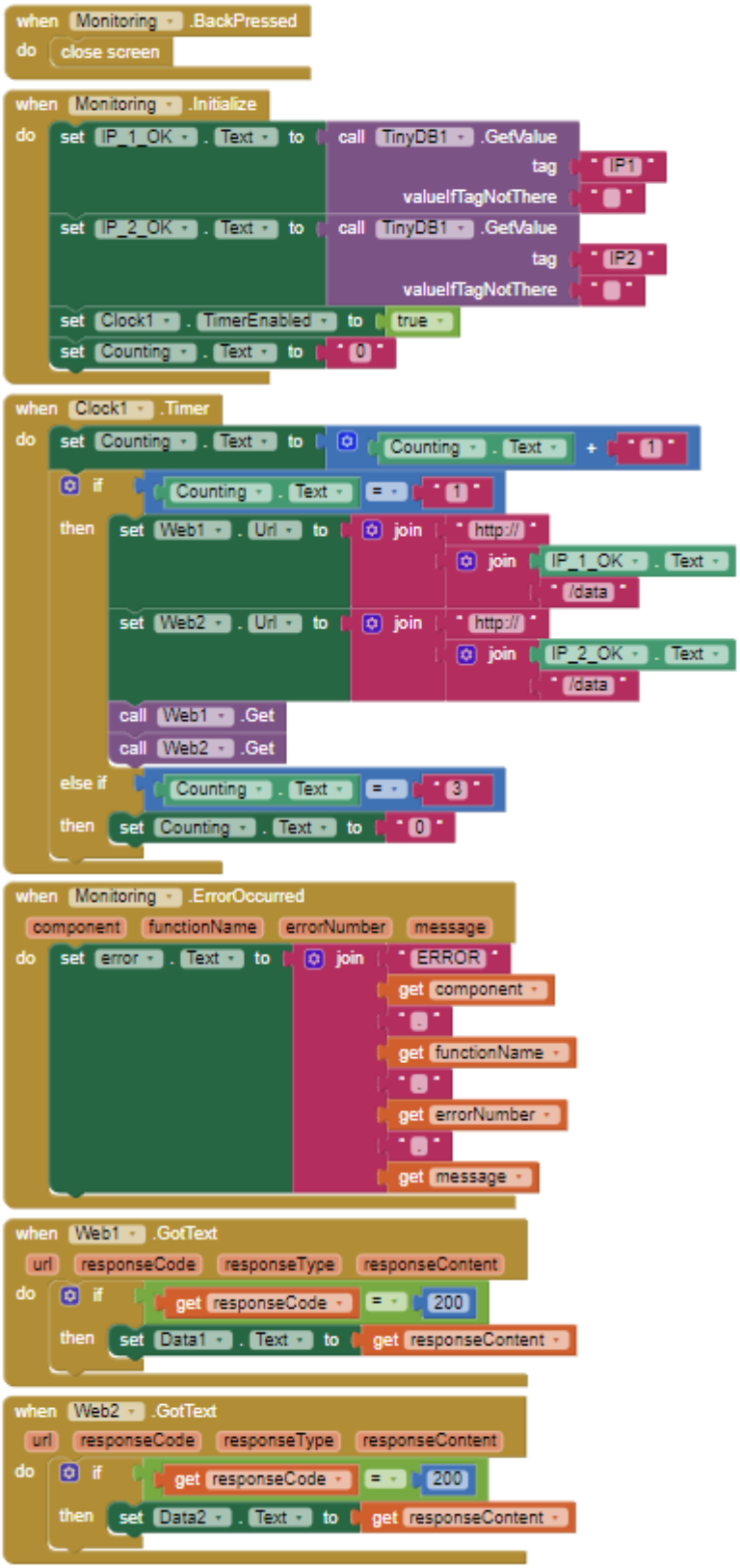

when ONOFF .BackPressed
do close screen
when ONOFF .Initialize
do set IP_1_OK . Text to call TinyDB1 .GetValue
tag "IP1"
valueIfTagNotThere ""
set IP_2_OK . Text to call TinyDB1 .GetValue
tag "IP2"
valueIfTagNotThere ""
set Web1 . Url to join "http://"
join IP_1_OK . Text
"/kondisi"
call Web1 .Get
set Web2 . Url to join "http://"
join IP_2_OK . Text
"/kondisi"
call Web2 .Get
when Lampu1ON .Click
do set Web1 . Url to join "http://"
join IP_1_OK . Text
"/lampu1on"
call Web1 .Get
when Lampu1OFF .Click
do set Web1 . Url to join "http://"
join IP_1_OK . Text
"/lampu1off"
call Web1 .Get
when Lampu2ON .Click
do set Web2 . Url to join "http://"
join IP_2_OK . Text
"/lampu1on"
call Web2 .Get
when Lampu2OFF .Click
do set Web2 . Url to join "http://"
join IP_2_OK . Text
"/lampu1off"
call Web2 .Get
when CEK1 .Click
do set Web1 . Url to join "http://"
join IP_1_OK . Text
"/cek"
call Web1 .Get


when CEK2 .Click
do set Web2 . Url to join " http:// " join IP_2_OK . Text " /cek "
call Web2 .Get
when Lampu1ONMESH .Click
do set Web2 . Url to join " http:// " join IP_2_OK . Text " /lampulainon "
call Web2 .Get
when Lampu1OFFMESH .Click
do set Web2 . Url to join " http:// " join IP_2_OK . Text " /lampulainoff "
call Web2 .Get
when Lampu2ONMESH .Click
do set Web1 . Url to join " http:// " join IP_1_OK . Text " /lampulainon "
call Web1 .Get
when Lampu2OFFMESH .Click
do set Web1 . Url to join " http:// " join IP_1_OK . Text " /lampulainoff "
call Web1 .Get
when ONOFF .ErrorOccurred
component functionName errorNumber message
do set error . Text to join " ERROR " get component " . " get functionName " . " get errorNumber " . " get message

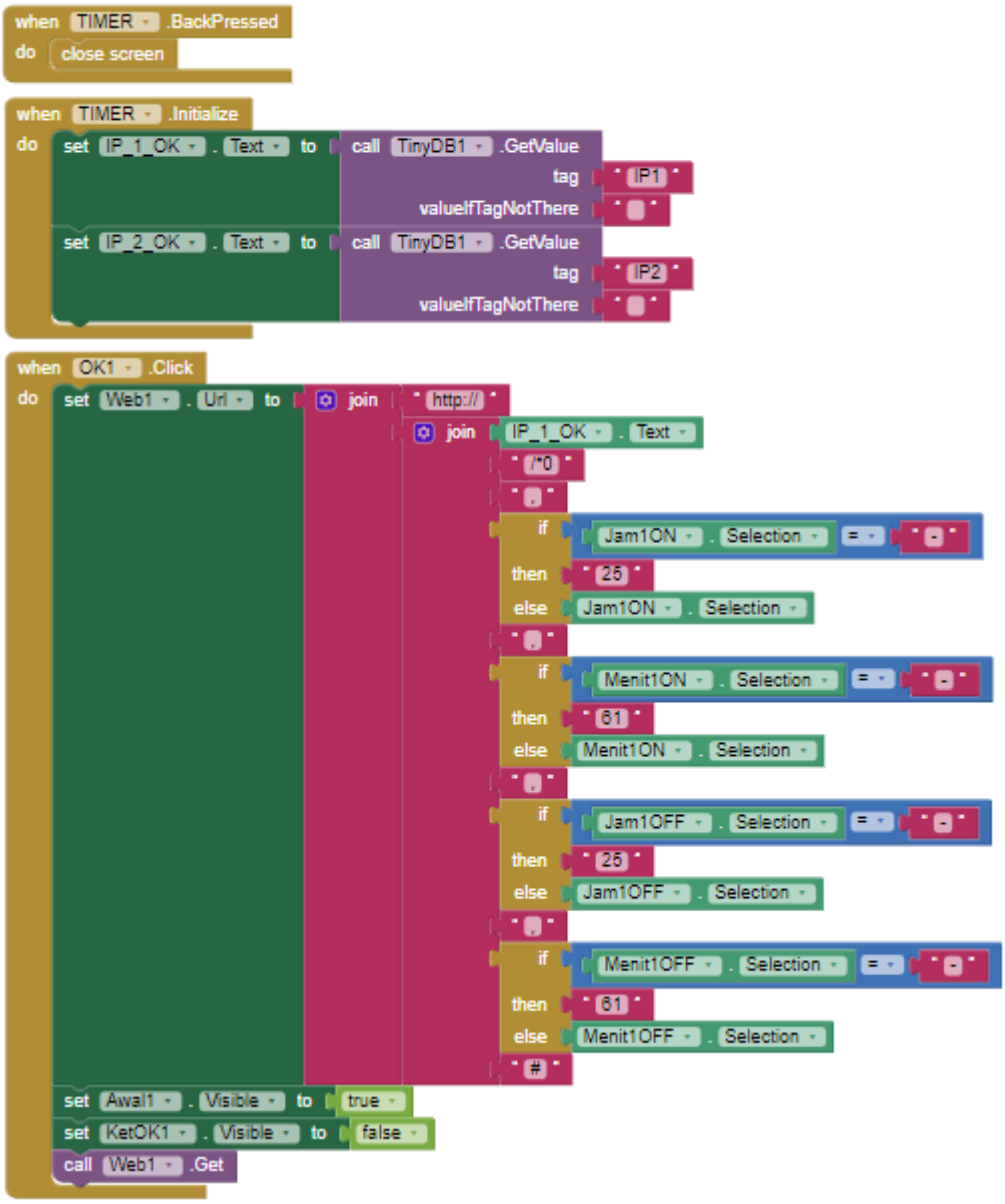

when Web1 .GotText
url responseCode responseType responseContent
do if get responseCode = 200
then set Ket1 . Text to get responseContent
if Ket1 . Text = " 1 "
then set lampu1awal . Visible to false
set lampu1off . Visible to false
set lampu1on . Visible to true
if Ket1 . Text = " 0 "
then set lampu1awal . Visible to false
set lampu1off . Visible to true
set lampu1on . Visible to false
if Ket1 . Text = " 3 "
then set lampu2awalmesh . Visible to false
set lampu2offmesh . Visible to false
set lampu2onmesh . Visible to true
if Ket1 . Text = " 2 "
then set lampu2awalmesh . Visible to false
set lampu2offmesh . Visible to true
set lampu2onmesh . Visible to false

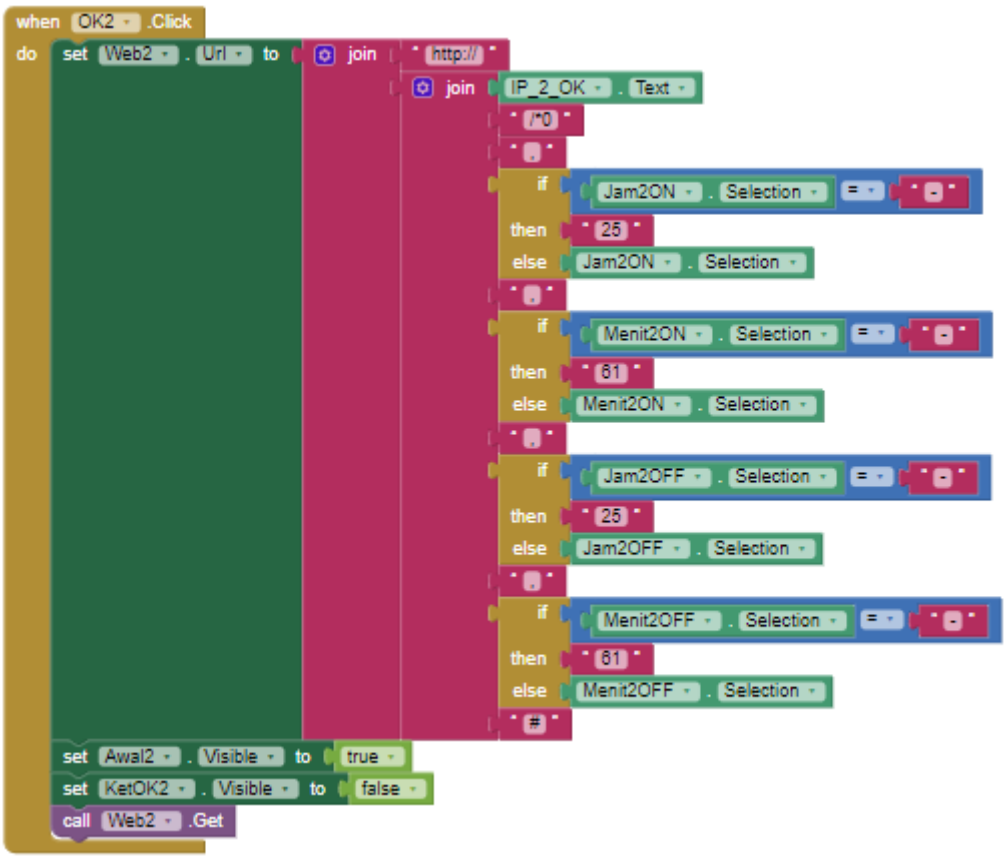

when Web2 .GotText
url responseCode responseType responseContent
do if get responseCode = 200
then set Ket2 . Text to get responseContent
if Ket2 . Text = " 1 "
then set lampu2awal . Visible to false
set lampu2off . Visible to false
set lampu2on . Visible to true
if Ket2 . Text = " 0 "
then set lampu2awal . Visible to false
set lampu2off . Visible to true
set lampu2on . Visible to false
if Ket2 . Text = " 3 "
then set lampu1awalmesh . Visible to false
set lampu1offmesh . Visible to false
set lampu1onmesh . Visible to true
if Ket2 . Text = " 2 "
then set lampu1awalmesh . Visible to false
set lampu1offmesh . Visible to true
set lampu1onmesh . Visible to false

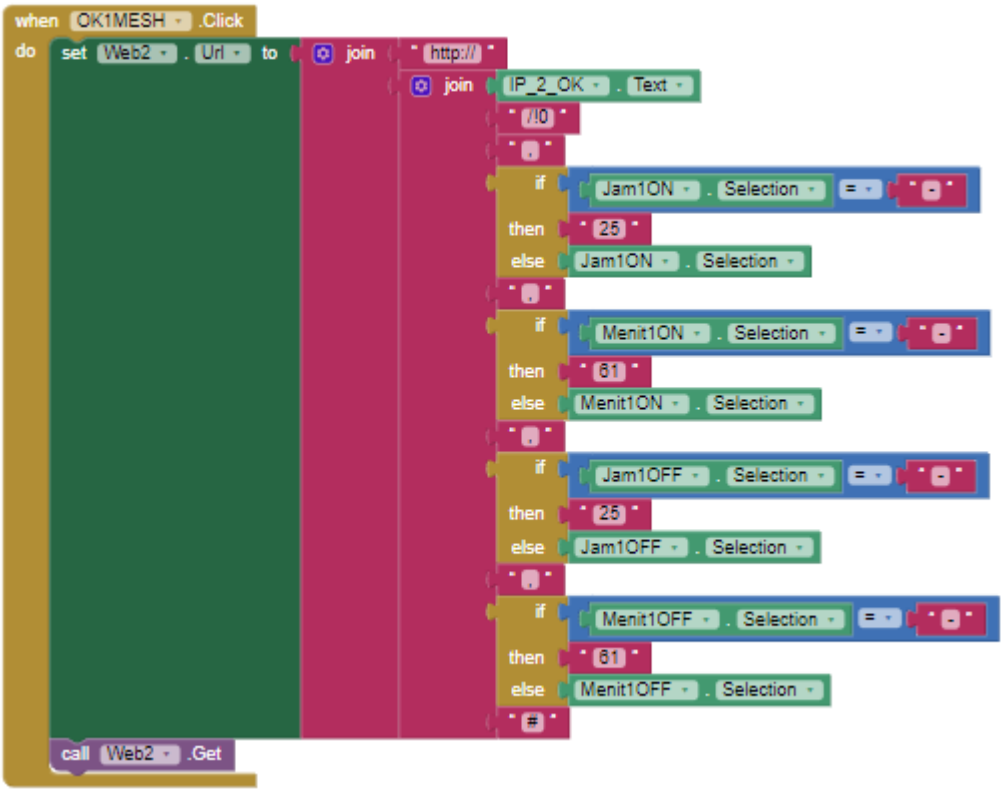
Menu Monitoring

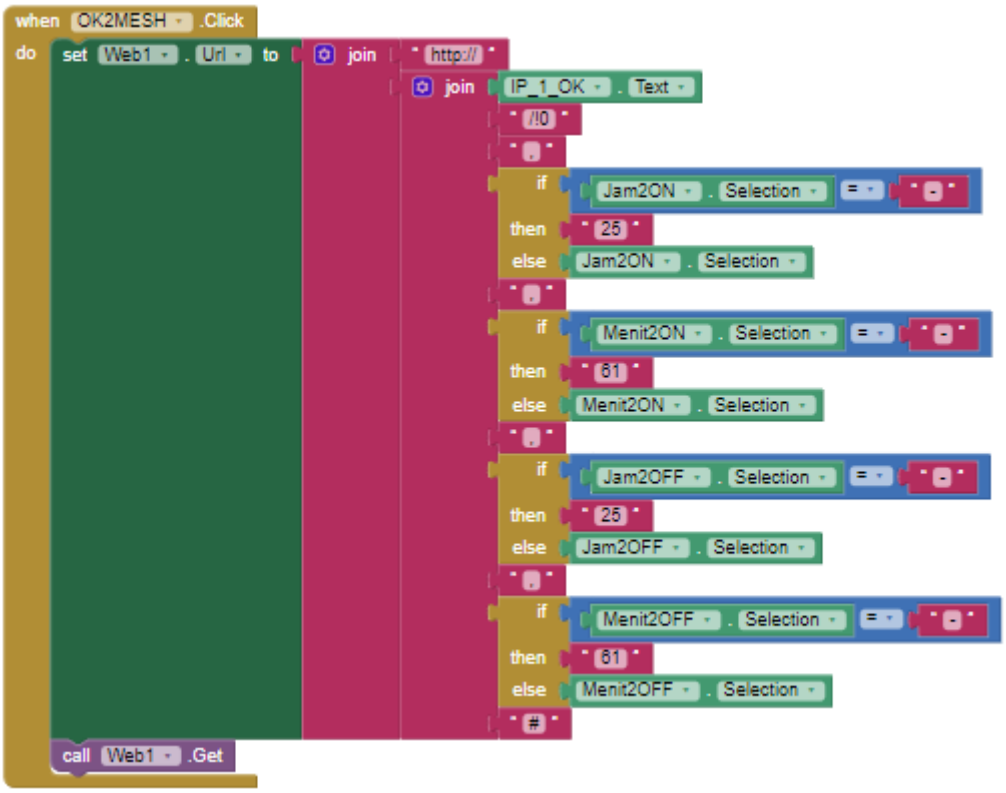


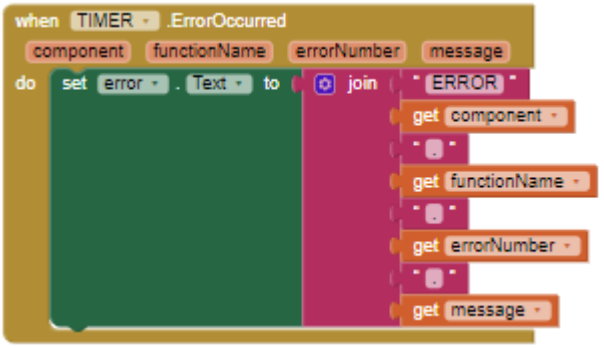
Menu Timer

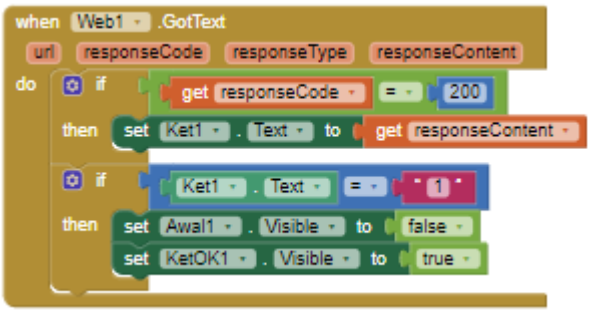

when TIMER .BackPressed
do close screen
when TIMER .Initialize
do set IP_1_OK . Text to call TinyDB1 .GetValue
tag "IP1"
valueIfTagNotThere ""
set IP_2_OK . Text to call TinyDB1 .GetValue
tag "IP2"
valueIfTagNotThere ""
when OK1 .Click
do set Web1 . Url to join "http://"
join IP_1_OK . Text
"/*0"
","
if Jam1ON . Selection = "-"
then "25"
else Jam1ON . Selection
","
if Menit1ON . Selection = "-"
then "61"
else Menit1ON . Selection
","
if Jam1OFF . Selection = "-"
then "25"
else Jam1OFF . Selection
","
if Menit1OFF . Selection = "-"
then "61"
else Menit1OFF . Selection
"#"
set Awal1 . Visible to true
set KetOK1 . Visible to false
call Web1 .Get

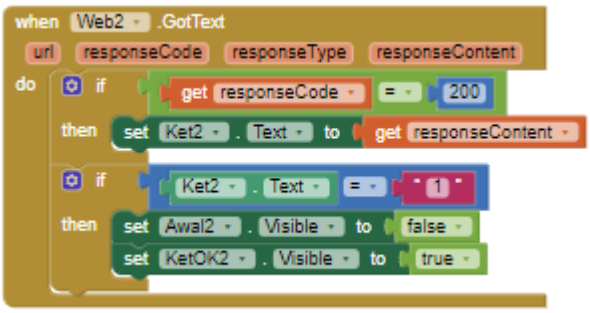

when OK2 .Click
do set Web2 . Url to join " http:// "
join IP_2_OK . Text
" /*0 "
" , "
if Jam2ON . Selection = " - "
then " 25 "
else Jam2ON . Selection
" , "
if Menit2ON . Selection = " - "
then " 61 "
else Menit2ON . Selection
" , "
if Jam2OFF . Selection = " - "
then " 25 "
else Jam2OFF . Selection
" , "
if Menit2OFF . Selection = " - "
then " 61 "
else Menit2OFF . Selection
" # "
set Awal2 . Visible to true
set KetOK2 . Visible to false
call Web2 .Get


when OK1MESH .Click
do set Web2 . Url to join " http:// "
join IP_2_OK . Text
" /!0 "
" , "
if Jam1ON . Selection = " - "
then " 25 "
else Jam1ON . Selection
" , "
if Menit1ON . Selection = " - "
then " 61 "
else Menit1ON . Selection
" , "
if Jam1OFF . Selection = " - "
then " 25 "
else Jam1OFF . Selection
" , "
if Menit1OFF . Selection = " - "
then " 61 "
else Menit1OFF . Selection
" # "
call Web2 .Get


when OK2MESH .Click
do set Web1 . Url to join "http://"
join IP_1_OK . Text
"/!0"
","
if Jam2ON . Selection = "-"
then "25"
else Jam2ON . Selection
","
if Menit2ON . Selection = "-"
then "61"
else Menit2ON . Selection
","
if Jam2OFF . Selection = "-"
then "25"
else Jam2OFF . Selection
","
if Menit2OFF . Selection = "-"
then "61"
else Menit2OFF . Selection
"#"
call Web1 .Get


when TIMER .ErrorOccurred
component functionName errorNumber message
do set error . Text to join "ERROR"
get component
"."
get functionName
"."
get errorNumber
"."
get message


when Web1 .GotText
url responseCode responseType responseContent
do if get responseCode = 200
then set Ket1 . Text to get responseContent
if Ket1 . Text = "1"
then set Awal1 . Visible to false
set KetOK1 . Visible to true


when Web2 .GotText
url responseCode responseType responseContent
do if get responseCode = 200
then set Ket2 . Text to get responseContent
if Ket2 . Text = " 1 "
then set Awal2 . Visible to false
set KetOK2 . Visible to true

LAMPIRAN 4

Hasil pengujian data *logger*

DATA LOGGER

01/09/2019	waktu	tegangan	Arus	Daya	Biaya_Hari_ini	Waktu Kemarin	Biaya_kemarin
01/09/2019	0:10:2	226,6	0,09	12	Rp.3.02	0:10:1	Rp.3.21
01/09/2019	0:20:2	226,6	0,08	12	Rp.5.85	0:20:1	Rp.6.14
01/09/2019	0:30:3	226,7	0,08	12	Rp.8.77	0:30:1	Rp.9.82
01/09/2019	0:40:1	226,6	0,08	12	Rp.11.70	0:40:2	Rp.12.83
01/09/2019	0:50:1	226,6	0,08	12	Rp.14.62	0:50:2	Rp.15.20
01/09/2019	1:00:1	226,8	0,08	12	Rp.17.55	1:00:2	Rp.18.13
01/09/2019	1:10:1	226,8	0,08	12	Rp.20.47	1:10:2	Rp.21.05
01/09/2019	1:20:1	226,5	0,08	13	Rp.25.35	1:20:2	Rp.25.97
01/09/2019	1:30:3	226,8	0,09	12	Rp.26.32	1:30:1	Rp.26.61
01/09/2019	1:40:2	226,8	0,08	12	Rp.29.25	1:40:1	Rp.29.53
01/09/2019	1:50:2	226,7	0,09	13	Rp.34.85	1:50:1	Rp.32.46
01/09/2019	2:00:2	226,7	0,09	12	Rp.35.10	2:00:2	Rp.35.67
01/09/2019	2:10:2	225,8	0,09	12	Rp.38.02	2:10:2	Rp.38.60
01/09/2019	2:20:1	225,7	0,09	12	Rp.40.95	2:20:1	Rp.41.23
01/09/2019	2:30:1	225,9	0,08	12	Rp.43.87	2:30:1	Rp.45.84
01/09/2019	2:40:2	226,9	0,08	12	Rp.46.80	2:40:2	Rp.47.37
01/09/2019	2:50:2	226,6	0,08	13	Rp.53.87	2:50:1	Rp.50.00
01/09/2019	3:00:1	226,6	0,09	13	Rp.55.04	3:00:1	Rp.57.00
01/09/2019	3:10:1	226,8	0,08	12	Rp.57.57	3:10:2	Rp.59.07
01/09/2019	3:20:1	226,7	0,09	12	Rp.58.49	3:20:2	Rp.60.83
01/09/2019	3:30:1	226,8	0,08	12	Rp.61.42	3:30:2	Rp.62.00
01/09/2019	3:40:1	226,8	0,08	12	Rp.64.34	3:40:2	Rp.64.92
01/09/2019	3:50:1	225,9	0,08	13	Rp.72.87	3:50:2	Rp.67.84
01/09/2019	4:00:1	225,9	0,08	12	Rp.75.19	4:00:2	Rp.76.67
01/09/2019	4:10:1	225,9	0,08	13	Rp.79.21		
01/09/2019	4:20:2	225,8	0,08	12	Rp.81.04		
01/09/2019	4:30:2	225,5	0,08	13	Rp.85.54		
01/09/2019	4:40:2	225,9	0,08	12	Rp.81.89		
01/09/2019	4:50:2	225,9	0,09	12	Rp.84.81		
01/09/2019	5:00:1	226,9	0,09	12	Rp.87.74		
01/09/2019	5:10:1	226,8	0,08	12	Rp.90.66		
01/09/2019	5:20:2	226,8	0,09	13	Rp.101.39		
01/09/2019	5:30:2	226,9	0,09	12	Rp.96.51		
01/09/2019	5:40:2	226,6	0,09	12	Rp.99.43		
01/09/2019	5:50:2	226,7	0,08	12	Rp.102.36		
01/09/2019	6:00:2	226.5	0.08	13	Rp.114.06		