RANCANG BANGUN PENGENDALI BEBAN KELISTRIKAN PADA RUANG KERJA BERBASIS *INTERNET OF THINGS* (IOT)

LAPORAN TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan
pendidikan diploma tiga (D-3) Program Studi Teknik Konversi Energi
Jurusan Teknik Mesin
Politeknik Negeri Ujung Pandang

MUH. FADLI ADRIYAWAN	342 18 038
RAHMAWATI	342 18 046

PROGRAM STUDI D-3 TEKNIK KONVERSI ENERGI
JURUSAN TEKNIK MESIN
POLITEKNIK NEGERI UJUNG PANDANG
MAKASSAR

2021

HALAMAN PENGESAHAN

Laporan tugas akhir dengan judul "Rancang Bangun Pengendali Beban Kelistrikan pada Ruang Kerja Berbasis *Internet of Things* (IoT)" oleh Muh. Fadli Adriyawan NIM 34218038 dan Rahmawati NIM 34218046 dinyatakan layak untuk diujikan.

Makassar, 4 Oktober 2021

Pembimbing I,	Pembimbing II,
Sonong, S.T., M.T.	Sukma Abadi, S.T., M.T.
NIP. 19621202 199203 1 002	NIP. 19751024 200312 1 001

Mengetahui
a.n. Direktur Politeknik Negeri Ujung Pandang
Ketua Jurusan Teknik Mesin,

Rusdi Nur, S.ST., M.T., Ph.D.
NIP. 19741106 200212 1 002

HALAMAN PENERIMAAN

Pada hari ini, Senin tanggal 4 Oktober 2021, tim penguji ujian sidang laporan tugas akhir telah menerima hasil ujian sidang laporan tugas akhir oleh mahasiswa Muh. Fadli Adriyawan NIM 34218038 dan Rahmawati NIM 34218046 dengan judul "Rancang Bangun Pengendali Beban Kelistrikan pada Ruang Kerja Berbasis *Internet of Things* (IoT)".

Makassar, 4 Oktober 2021

Tim Penguji Ujian Sidang Laporan Tugas Akhir:

1.	Marhatang, S.ST., M.T.	Ketua	(....................)
2.	Apollo, S.T., M.Eng.	Sekretaris	(....................)
3.	Ir. Remigius Tandioga, M.Eng.Sc.	Anggota	(....................)
4.	Abdul Rahman, S.T., M.T.	Anggota	(....................)
5.	Sukma Abadi, S.T., M.T.	Anggota	(....................)
6.	Sonong, S.T., M.T.	Anggota	(....................)

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah Subhanahu Wa Ta'ala karena berkat rahmat dan karunia-Nya, penulisan laporan tugas akhir ini yang berjudul "Rancang Bangun Pengendali Beban Kelistrikan pada Ruang Kerja Berbasis *Internet of Things* (IoT)" dapat diselesaikan dengan baik. Laporan tugas akhir ini disusun sebagai salah satu persyaratan meraih gelar ahli madya pada Program Studi D-3 Teknik Konversi Energi, Jurusan Teknik Mesin, Politeknik Negeri Ujung Pandang.

Dalam penulisan laporan tugas akhir ini tidak sedikit hambatan yang penulis alami. Namun, berkat bantuan berbagai pihak terutama pembimbing, hambatan tersebut dapat teratasi. Sehubungan dengan itu, pada kesempatan dan melalui lembaran ini penulis menyampaikan terima kasih dan penghargaan kepada:

1. Keluarga yang selalu memberikan dukungan dari segi materi maupun moril, mendoakan dan juga memberikan semangat motivasi sehingga penulis bisa menyelesaikan tugas akhir ini tepat waktu.
2. Bapak Prof. Ir. Muhammad Anshar, M.Si., Ph.D. selaku Direktur Politeknik Negeri Ujung Pandang.
3. Bapak Rusdi Nur, S.ST., M.T., Ph.D. selaku Ketua Jurusan Teknik Mesin Politeknik Negeri Ujung Pandang.
4. Ibu Sri Suwasti, S.ST., M.T. selaku Koordinator Program Studi D-3 Teknik Konversi Energi Politeknik Negeri Ujung Pandang.

5. Bapak Sonong, S.T., M.T. selaku Dosen Pembimbing I yang telah memberikan bimbingan, perhatian, dan kesempatannya untuk mengarahkan penulis dalam menyelesaikan tugas akhir ini.
6. Bapak Sukma Abadi, S.T., M.T. selaku Dosen Pembimbing II yang telah memberikan bimbingan, perhatian, dan kesempatannya untuk mengarahkan penulis dalam menyelesaikan tugas akhir ini.
7. Bapak A.M. Shiddiq Yunus, S.T., M.Eng.Sc., Ph.D. selaku Wali Kelas 3B D-3 Teknik Konversi Energi.
8. Seluruh dosen dan staf Program Studi D-3 Teknik Konversi Energi yang telah memberikan ilmunya kepada penulis selama melaksanakan perkuliahan dan telah membantu dalam menyediakan fasilitas dan sarana dalam mengerjakan tugas akhir ini.
9. Teman-teman kelas 3 D-3 Teknik Konversi Energi angkatan 2018 yang telah memberikan dukungan kepada penulis selama proses pembuatan tugas akhir ini.
10. Serta semua pihak yang telah banyak membantu dan memberikan masukan dalam penyelesaian tugas akhir ini yang tidak dapat disebutkan satu pesatu.

Semoga Allah Subhanahu Wa Ta'ala membalas kebaikan siapapun yang terlibat dalam penyusunan laporan tugas akhir ini dengan nikmat dan berkah yang melimpah. Aamiin.

Sebagai manusia biasa penulis menyadari bahwa laporan tugas akhir ini masih jauh dari kesempurnaan. Untuk itu penulis mengharapkan kritik dan saran yang membangun demi kesempurnaan laporan tugas akhir ini dan demi perbaikan pada masa mendatang. Semoga laporan tugas akhir ini dapat bermanfaat bagi siapapun yang membacanya.

Makassar,   September 2021

Penulis

DAFTAR ISI

DAFTAR TABEL

hlm.

DAFTAR GAMBAR

DAFTAR SINGKATAN

IoT	: *Internet of Things*
RFID	: *Radio Frequency Identification*
IDE	: *Integrated Development Environment*
IP	: *Internet Protocol*
IC	: *Integrated Circuit*
PWM	: *Pulse Width Modulation*
USB	: *Universal Serial Bus*
ICSP	: *In-Circuit Serial Programming*
NO	: *Normally Open*
NC	: *Normally Close*
AC	: *Alternating Current*
DC	: *Direct Current*
GPIO	: *General Purpose Input Output*
ADC	: *Analog to Digital Converter*

DAFTAR LAMPIRAN

hlm.

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Muh. Fadli Adriyawan

NIM : 34218038

menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam laporan tugas akhir ini, yang berjudul "Rancang Bangun Pengendali Beban Kelistrikan pada Ruang Kerja Berbasis *Internet of Things* (IoT)" merupakan gagasan, hasil karya saya sendiri dengan arahan pembimbing, dan belum pernah diajukan dalam bentuk apapun pada perguruan tinggi dan instansi manapun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan oleh penulis lain telah disebutkan dalam naskah dan dicantumkan dalam daftar pustaka laporan tugas akhir ini.

Jika pernyataan saya tersebut di atas tidak benar, saya siap menanggung risiko yang ditetapkan oleh Politeknik Negeri Ujung Pandang.

Makassar, 4 Oktober 2021

Muh. Fadli Adriyawan
NIM. 34218038

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Rahmawati

NIM : 34218046

menyatakan dengan sebenar-benarnya bahwa segala pernyataan dalam laporan tugas akhir ini, yang berjudul "Rancang Bangun Pengendali Beban Kelistrikan pada Ruang Kerja Berbasis *Internet of Things* (IoT)" merupakan gagasan, hasil karya saya sendiri dengan arahan pembimbing, dan belum pernah diajukan dalam bentuk apapun pada perguruan tinggi dan instansi manapun.

Semua data dan informasi yang digunakan telah dinyatakan secara jelas dan dapat diperiksa kebenarannya. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan oleh penulis lain telah disebutkan dalam naskah dan dicantumkan dalam daftar pustaka laporan tugas akhir ini.

Jika pernyataan saya tersebut di atas tidak benar, saya siap menanggung risiko yang ditetapkan oleh Politeknik Negeri Ujung Pandang.

Makassar, 4 Oktober 2021

Rahmawati
NIM. 34218046

RANCANG BANGUN PENGENDALI BEBAN KELISTRIKAN PADA RUANG KERJA BERBASIS *INTERNET OF THINGS* (IOT)

RINGKASAN

Pengendalian beban kelistrikan menggunakan saklar biasa mengharuskan seseorang untuk mendekat dan menjangkau saklar tersebut untuk mengendalikannya. Permasalahan lain muncul apabila suatu ruangan dalam keadaan kosong dan pemilik ruangan sedang berada di luar. Pemilik ruangan akan kesulitan untuk mengetahui kondisi beban kelistrikan apakah dalam keadaan menyala atau mati. Dengan adanya sistem ini, yang menggunakan konsep *Internet of Things* (IoT) diharapkan dapat mengatasi permasalahan dalam mengetahui kondisi atau pengendalian beban kelistrikan seperti lampu dan AC dari jarak jauh melalui aplikasi IoT pada *smartphone*.

Tujuan dari kegiatan ini adalah untuk membangun sistem pengendali beban kelistrikan menggunakan *smartphone* yang dapat digunakan dari jarak jauh. Keseluruhan sistem ini dapat mengetahui kondisi beban kelistrikan dan dapat mengendalikan beban kelistrikan dimana saja posisi *user* berada. Dalam penerapannya, sistem ini menggunakan modul wifi NodeMCU ESP8266 sebagai pengendali *output* dan aplikasi Telegram sebagai pengendali *input*.

Rancang bangun pengendali beban kelistrikan pada ruang kerja berbasis *internet of things* (IoT) ini telah dilaksanakan. Sistem pengendali beban kelistrikan tersebut dapat digunakan di mana saja dan semua komponen bekerja dengan baik sehingga sistem pengendali tersebut dapat membantu dalam usaha penghematan penggunaan energi listrik sebagaimana yang diharapkan.

BAB I PENDAHULUAN

1.1 Latar Belakang

Salah satu aspek perkembangan teknologi jaringan internet adalah *Internet of Things* (IoT). *Internet of Things* (IoT) adalah sebuah gagasan di mana semua benda di dunia nyata dapat berkomunikasi satu dengan yang lain sebagai bagian dari satu kesatuan sistem terpadu menggunakan jaringan internet sebagai penghubung (Anwar dan Hermanto, 2019). *Internet of Things* (IoT) dikenalkan pertama kali oleh visioner Inggris bernama Kevin Ashton, pada tahun 1999. *Internet of Things* (IoT) merupakan teknologi yang diharapkan mampu menawarkan perangkat sistem canggih dengan kemampuan konektivitas, sehingga mampu melakukan komunikasi mesin ke mesin (M2M) dan mencakup berbagai protokol, domain, dan aplikasi (Mahali, 2016).

Smart room merupakan teknologi terapan dari *Internet of Things* (IoT) yang dapat menjadi salah satu bentuk solusi alat pengendali jarak jauh. *Smart room* merupakan alat kendali ruang yang mengintegrasikan teknologi perangkat komputasi, sensor, aktuator, teknologi komunikasi (umumnya nirkabel) dan didesain untuk melayani pengguna melalui operasi otomatis dan usaha minimal dari pengguna. Dengan kata lain, *smart room* dapat mengontrol kendali ruangan dengan menggunakan sensor atau jaringan internet yang saling terintegrasi dengan perangkat yang dapat diperintah langsung oleh *user* (Muhamad, 2019).

Dengan adanya fitur pengendali dan pemantauan jarak jauh, maka penggunaan beban kelistrikan seperti lampu, AC dan beban kelistrikan lainnya dapat dikendalikan dan dipantau guna untuk mengefisienkan penggunanya. Kemudahan dalam penggunaan fitur pengendali dan pemantauan jarak jauh ini dapat digunakan ketika di area lingkungan rumah, sekolah, kantor, dan lain-lain dengan menggunakan jaringan internet yang dapat menghidupkan atau mematikan peralatan (Nurfaif, 2017).

Penerapan konsep *smart room* pada lingkungan kampus Politeknik Negeri Ujung Pandang sudah pernah dibahas dalam laporan tugas akhir (Evendi dan Erny Febryanty, 2019) dengan judul "Rancang Bangun Alat Kontrol *Switching* Kelistrikan Ruangan dengan Sensor RFID Berbasis Mikrokontroler" tersebut membahas suatu sistem pengendalian ruangan otomatis dengan menggunakan sensor RFID. Ketika sensor RFID memperoleh sinyal dari tag RFID maka sinyal tersebut akan diteruskan ke mikrokontroler Arduino. Setelah mendeteksi *output* dari sensor RFID yang terpasang pada pintu masuk, maka tanggapan mikrokontroler terhadap *output* sensor berupa kendali terhadap beban kelistrikan di ruangan tersebut.

Dalam rangka mengembangkan penerapan konsep *smart room* yang pernah dibuat oleh Rustan Evendi dan Erny Febryanty (2019) tersebut, penulis bermaksud untuk membuat tugas akhir dengan judul "Rancang Bangun Pengendali Beban Kelistrikan pada Ruang Kerja Berbasis *Internet of Things* (IoT)".

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka penulis dapat mengambil rumusan masalah sebagai berikut:

1. Bagaimana memonitor terpakai/tidaknya beban listrik di sebuah ruangan dari jarak jauh?
2. Bagaimana meng-OFF-kan beban listrik yang sedang terpakai di sebuah ruangan dari jarak jauh?
3. Dapatkah sistem pengendali jarak jauh beban kelistrikan menghasilkan penghematan energi?

1.3 Ruang Lingkup Kegiatan

Dalam pembuatan tugas akhir ini ada beberapa hal yang menjadi ruang lingkup dari kegiatan yaitu:

1. Modul wifi yang digunakan adalah NodeMCU ESP8266.
2. *Software* yang digunakan adalah Arduino IDE.
3. Pengontrolan sistem hanya bisa digunakan di tempat yang terhubung dengan jaringan internet.

1.4 Tujuan Kegiatan

Tujuan yang ingin dicapai dalam kegiatan ini adalah:

1. Untuk memonitor terpakai/tidaknya beban listrik di sebuah ruangan dari jarak jauh.
2. Untuk meng-*off*-kan beban listrik yang sedang terpakai di sebuah ruangan dari jarak jauh.

3. Untuk meneliti kemungkinan adanya penghematan energi akibat diterapkannya sistem pengendali beban kelistrikan dari jarak jauh.

1.5 Manfaat Kegiatan

Dengan kegiatan ini, dapat memberikan kemudahan dalam mengontrol beban kelistrikan yang berada di sebuah ruangan dengan menggunakan aplikasi Telegram pada *smartphone*. Sehingga sistem ini lebih praktis digunakan karena dapat mengontrol penggunaan beban kelistrikan di mana pun posisi *user* berada.

BAB II TINJAUAN PUSTAKA

2.1 Internet of Things (IoT)

Internet of Things (IoT) pertama kali diperkenalkan oleh seorang visioner Inggris bernama Kevin Ashton pada tahun 1999. *Internet of Things* (IoT) merupakan teknologi yang diharapkan mampu menawarkan perangkat sistem canggih dengan kemampuan konektivitas, sehingga mampu melakukan komunikasi mesin ke mesin (M2M) dan mencakup berbagai protokol, domain, dan aplikasi. Interkoneksi pada perangkat ini tertanam (*embedded*) sehingga diharapkan mampu untuk mengantarkan otomasi dalam hampir semua bidang (Mahali, 2016).

Gambar 2. 1 Konsep *Internet of Things* (IoT)

(Sumber: Wibowo, 2021)

Cara kerja IoT yaitu setiap benda harus memiliki sebuah alamat Internet Protocol (IP). Alamat IP adalah sebuah identitas dalam jaringan yang membuat benda tersebut bisa diperintahkan dari benda lain dalam jaringan yang sama. Alamat IP dalam benda-benda tersebut akan dikoneksikan ke jaringan internet

(Wilianto dan Ade Kurniawan, 2018). Berdasarkan Gambar 2.1, *Internet of Things* (IoT) dapat menghubungkan berbagai perangkat agar dapat dikendalikan dari mana saja dengan bantuan koneksi internet. Misalkan di sebuah rumah atau gedung yang memiliki konsep IoT, di ruangan tersebut terdapat lampu yang berfungsi memberi cahaya di dalam ruangan, dan pada saat ruangan tersebut tidak ada yang menggunakan lagi maka secara otomatis lampu tersebut mati. Hal ini dapat terjadi karena pada ruangan tersebut diberi *input*-an ataupun alat sensor yang dapat mendeteksi keberadaan manusia (misalkan untuk kasus tersebut) dan sensor akan mengirimkan sinyal ke mikrokontroler kemudian sinyal tersebut dikirimkan ke server untuk memberitahu apa yang harus dikerjakan perangkat tersebut, misalkan dari kasus ini server mengirimkan sinyal 0 yang diterima relay, dan relay kemudian memutuskan arus listrik tersebut sehingga menyebabkan arus listrik mati dan menyebabkan lampu di dalam ruang tersebut padam (Endra, 2019).

2.2 Smart Home/Smart Room

Smart home system atau sistem rumah pintar secara sederhana dapat diartikan sebagai bangunan rumah yang dilengkapi teknologi canggih, sehingga seluruh perangkat dan sistem tersebut dapat saling terhubung. Artinya, sebagai pemilik rumah Anda dapat mengendalikan perangkat (perlengkapan maupun peralatan) didalam rumah secara *remote* atau jarak jauh (Santana, 2020).

Smart home atau rumah pintar adalah suatu rumah yang memiliki sistem otomatis yang sangat canggih untuk mengontrol peralatan rumah seperti pencahayaan dan suhu, peralatan multi-media, memantau dan mengaktifkan alarm serta membuka dan menutup jendela atau pintu dan masih banyak fungsi lainnya (Prasetio, 2017).

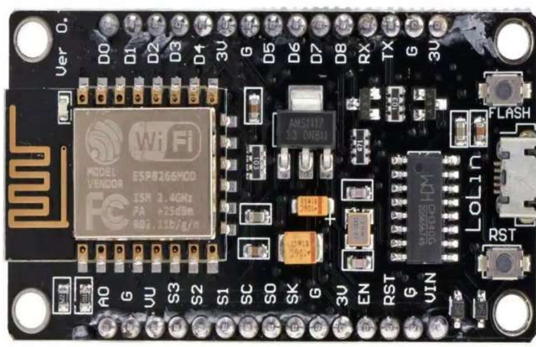
Gambar 2. 2 Ilustrasi *smart room*

(Sumber: Erawan, 2015)

Konsep rumah pintar merupakan hasil teknologi terapan yang menggabungkan antara rekayasa elektronika, informatika, dan arsitektur. Penghuni rumah dapat mengatur semua bagian rumahnya secara otomatis atau dengan menggunakan sistem yang terintegrasi ke smartphone sehingga rumah pintar pun kini menjadi trend dan solusi atas kebutuhan manusia terhadap tempat huni. Sistem ini biasanya dibekali dengan sistem keamanan berupa sistem autentifikasi *username* dan *password* (Prasetio, 2017).

2.3 NodeMCU ESP8266

Pada umumnya NodeMCU ESP8266 merupakan *platform* IoT dari pengembangan ESP8266 dengan *firmware* yang berbasis e-Lua. NodeMCU ESP8266 sendiri dilengkapi dengan *micro usb port* yang berfungsi untuk melakukan pemrograman. Pada modul NodeMCU ESP8266 juga dilengkapi dengan tombol *push button*, yaitu tombol *reset* dan *flash.* Bahasa pemograman yang digunakan pada modul NodeMCU ESP8266 adalah bahasa Lua yang juga merupakan *package* dari ESP8266. Pada bahasa Lua memiliki sistematis logika dan susunan pemrograman yang sama dengan bahasa C.



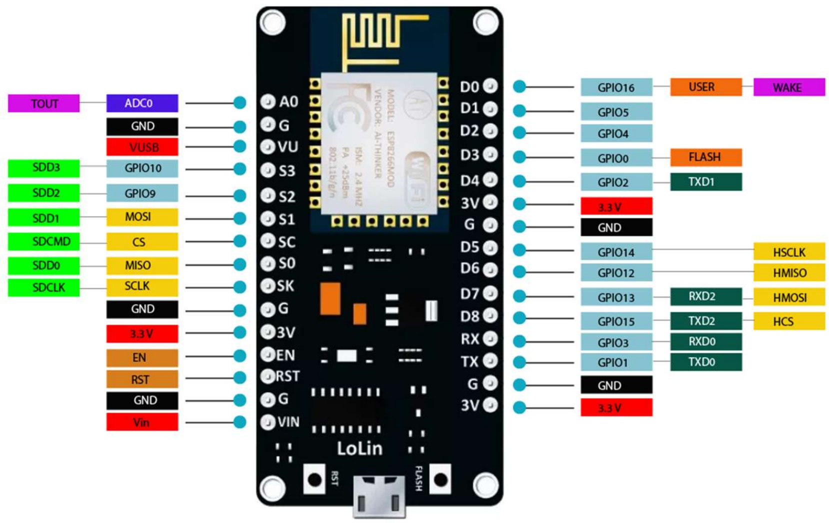
Gambar 2. 3 NodeMCU ESP8266

(Sumber: Mahardika, 2020)

NodeMCU ESP8266 dapat dioperasikan jika terhubung Wifi sehingga dapat mengirimkan informasi-informasi yang akan dikirimkan ke server. Adapun karakteristik dari NodeMCU ESP8266 dapat dilihat dari Tabel 2.1.

Tabel 2. 1 Karakteristik NodeMCU

Spesifikasi NodeMCU	
Type	ESP8266 ESP-12E
USB port	Micro USB
GPIO Pin	13
ADC	1 pin (10 bit)
USB to serial converter	CH340G
Power input	5 Vdc
Ukuran modul	57 × 30 mm

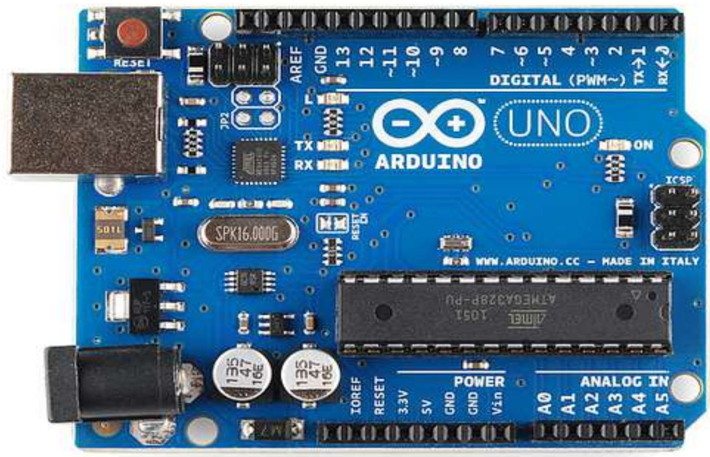


Gambar 2. 4 GPIO NodeMCU ESP8266

(Sumber: The Engineering Projects, 2018)

2.4 Arduino Uno

Arduino Uno adalah papan mikrokontroler berbasis ATmega328 yang memiliki 14 pin digital input/output (dimana 6 pin dapat digunakan sebagai output PWM) , 6 input analog, clock speed 16 MHz, koneksi USB, *jack* listrik, *header* ICSP, dan tombol *reset*. Board ini menggunakan daya yang terhubung ke komputer dengan kabel USB atau daya eksternal dengan adaptor AC-DC atau baterai.



Gambar 2. 5 *Board* Arduino uno

(Sumber: Febrianto, 2014)

Spesifikasi Arduino Uno :

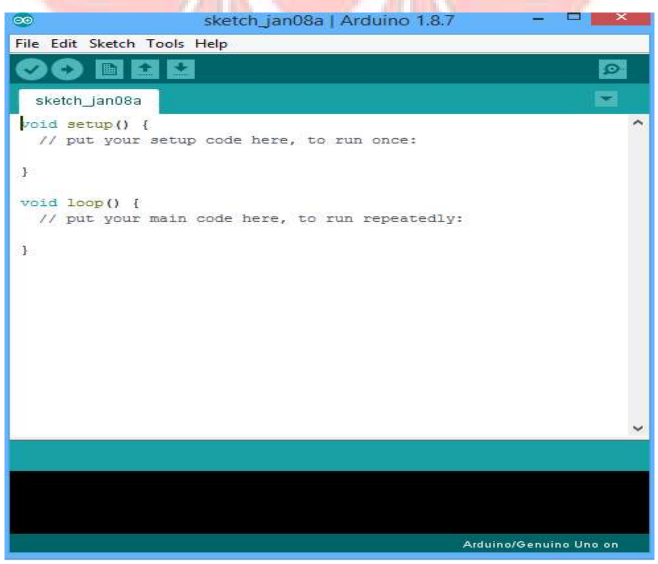
• Mikrokontroler	ATmega328
• Tegangan operasi	5V
• Tegangan input	7-12V
• Tegangan input (*limit*)	6-12 V
• Jumlah pin I/O digital	14 pin (6 diantaranya pin PWM)
• Jumlah pin input analog	6 pin
• Arus DC per pin I/O	40 mA
• Arus DC untuk pin 3,3 V	50 mA
• *Memori Flash*	32 KB (ATmega328) dimana 0,5 KB digunakan oleh *bootloader*
• SRAM	2 KB (ATmega328)
• EEPROM	1 KB (ATmega328)
• *Clock Speed*	16 MHz

2.5 Arduino IDE

Arduino IDE (*Integrated Development Environment*) adalah program khusus yang digunakan untuk menulis sketch untuk Arduino dengan bahasa yang sederhana, dalam penulisan sketch menggunakan bahasa pemrograman C.

Arduino IDE (*Integrated Development Environment*) merupakan editor teks untuk menulis kode, sebuah pesan, konsol teks, Arduino IDE menghubungkan Arduino dan hardware untuk meng-*upload* program dan berkomunikasi dengan alat elektronik.

Arduino IDE ini berguna sebagai *text editor* untuk membuat, mengedit, dan juga mevalidasi kode program. bisa juga digunakan untuk meng-*upload* ke *board* Arduino. Kode program yang digunakan pada Arduino disebut dengan istilah Arduino "*sketch*" atau disebut juga *source code* arduino, dengan ekstensi file *source code* .ino.



Gambar 2. 6 Tampilan utama *software* Arduino IDE

1. Toolbar

Tabel 2. 2 Simbol dan fungsi dari *toolbar software* Arduino IDE

No.	Icon	Nama *Icon*	Fungsi
1		Verify	Untuk mengecek kesalahan dari program yang dibuat
2		Upload	Untuk mengecek dan memasukkan program ke IC Arduino
3		New	Untuk membuat sketch yang baru
4		Open	Untuk membuka file sketch yang tersimpan
5		Save	Untuk menyimpan sketch
6		Serial Monitor	untuk menampilkan komunikasi serial antara Arduino dan computer

2. *Sketch*

Sketch merupakan kumpulan instruksi atau perintah yang akan ditanamkan pada Arduino dan bahasa pemrograman yang digunakan untuk menanamkan perintah pada Arduino yaitu bahasa C, *Syntax* yang digunakan pada Arduino bertipe *case sensitive* yaitu penggunaan huruf kapital dan non-kapital sangat berpengaruh terhadap berhasil atau tidak perintah tersebut ditanamkan.

Sketch pada Gambar 2.6 merupakan *sketch* dasar yang ada pada *software* Arduino IDE, terdapat dua buah metode yaitu “void setup()” dan "void loop()", dimana setup() berguna untuk mendefinisikan pin yang digunakan atau untuk membuka jalur komunikasi serial dan berlaku hanya satu kali saat program pertama kali dijalankan dan selanjutnya perintah yang diberikan akan tertanam pada Arduino, sedangkan loop() berguna untuk mengulang setiap perintah yang ditanamkan pada Arduino selamanya.

2.6 Aplikasi Telegram

Telegram adalah sebuah aplikasi layanan pengirim pesan instan multiplatform berbasis *cluod* yang bersifat gratis. Telegram tersedia untuk perangkat telepon seluler (Android, iOS, *Windows Phone*, Ubuntu *Touch*) dan sistem perangkat komputer (Windows, OS X, Linux). Para pengguna dapat mengirim pesan dan bertukar foto, video, stiker, audio, dan tipe berkas lainnya.

Berbagai kelebihan yang ditawarkan sangat berguna pada penelitian ini seperti adanya *cloud* pada server *Telegram Messenger* yang memungkinkan untuk menyimpan data-data seperti percakapan, foto, dan video. Salah satu kelebihan Telegram adalah terdapat fitur *channel* dan *bot*. Fitur *bot* ini memiliki kecerdasan artifisial yang merupakan fitur yang dapat terintegrasi dengan berbagai layanan melalui internet.

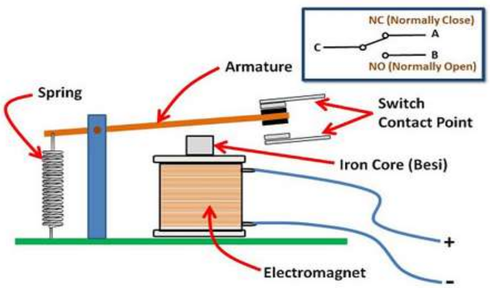
Gambar 2. 7 Telegram bot

2.7 Relay

Gambar 2. 8 Bentuk relay dan simbol relay

(Sumber: Kho, 2020)

Relay merupakan saklar elektromagnetik, yang di operasikan secara listrik dan termasuk komponen elektromekanikal, relay menggunakan prinsip elektromagnetik untuk menggerakkan kontak saklar sehingga memungkinkan sirkuit daya rendah untuk beralih ke tegangan yang relatif tinggi atau arus *on*/*off*. Fungsi sirkuit relay sebagai pengalih energi dimana ketika berlogika “1” maka relay akan *on* dan relay akan *off* ketika berlogika “0”.



Gambar 2. 9 Struktur relay

(Sumber: Kho, 2020)

Relay berkerja dengan prinsip elektromagnetik, ketika ada arus listrik lemah mengalir melalui kumparan, inti besi akan menjadi magnet dan menarik jangkar besi lunak sehingga kontak akan tersambung dan arus listrik kuat dapat mengalir keperangkat yang terhubung.

Beberapa fungsi relay yang telah umum diaplikasikan kedalam peralatan elektronika diantaranya adalah :

1) Relay digunakan untuk menjalankan fungsi logika (*logic function*);
2) Relay digunakan untuk memberikan fungsi penundaan waktu (*time delay function*);
3) Relay digunakan untuk mengendalikan sirkuit tegangan tinggi dengan bantuan dari signal tegangan rendah.

Pada dasarnya relay adalah sebuah kumparan yang dialiri arus listrik sehingga kumparan mempunyai sifat sebagai magnet. Magnet sementara tersebut digunakan untuk menggerakkan suatu sistem saklar yang terbuat dari logam sehingga pada saat Relay dialiri listrik maka kumparan akan terjadi kemagnetan dan menarik logam tersebut, saat arus listrik diputus maka logam akan kembali pada posisi semula.

2.8 Sensor Suhu DS18B20

Sensor suhu DS18B20 berfungsi untuk merubah besaran panas yang di tangkap menjadi besaran tegangan. Jenis sensor suhu yang digunakan dalam sistem ini adalah IC DS18B20, sensor ini memiliki presisi tinggi. Sensor ini sangat sederhana dengan hanya memiliki buah 3 kaki. Kaki pertama IC DS18B20 dihubung ke sumber daya, kaki kedua sebagai output dan kaki ketika di hubungkan ke ground.

Gambar 2. 10 Sensor suhu DS18B20

(Sumber: Prastyo, 2020)

Karakteristik dari IC DS18B20 adalah sebagai berikut:

1. Dapat dikalibrasikan langsung ke dalam besaran Celsius.
2. Faktor skala linear +10mV/°C .
3. Tingkat akurasi 0,5°C . Saat suhu kamar (25°C).
4. Jangkauan suhu antara -55°C . Sampai 150°C.
5. Bekerja pada tegangan 4 volt hingga 30 volt.
6. Arus kerja kurang dari 60 µA.
7. Impedensi keluaran rendah 0,1Ω untuk beban 1 mA.

Sensor DS18B20 bekerja dengan mengubah besaran suhu menjadi besaran tegangan. Tegangan ideal yang keluar dari DS18B20 mempunyai perbandingan 100°C setara dengan 1 volt. Sensor ini mempunyai pemanasan diri (*self heating*) kurang dari 0,1°C yang dapat dioperasikan dengan menggunakan *power supplay* tunggal dan dapat dihubungkan antar muka (*interface*) rangkaian kontrol yang sangat mudah.

Mikrokontroler IC DS18B20 dapat langsung dihubungkan dengan PIN A. Dimana PIN A merupakan PIN mikrokontroler yang dapat mengkonversi tegangan menjadi bilangan digital (*analog digial conversation*) atau lebih dikenal dengan ADC.

2.9 Adaptor

Adaptor adalah sebuah perangkat berupa rangkaian elektronika untuk mengubah tegangan listrik yang besar menjadi tegangan listrik lebih kecil, atau rangkaian untuk mengubah arus bolak-balik (arus AC) menjadi arus searah (arus DC). Adaptor/*power supplay* merupakan komponen inti dari peralatan elektronik. Adaptor digunakan untuk menurunkan tegangan AC 220 Volt menjadi kecil antara 3 volt sampai 12 volt sesuai kebutuhan alat elektronika.

Gambar 2. 11 Bentuk fisik adaptor

2.10 Soket USB

Soket USB adalah soket kabel USB yang sambungkan ke komputer atau laptop yang berfungsi untuk mengirimkan program ke Arduino dan juga sebagai *port* komunikasi serial.

Gambar 2. 12 Soket USB

(Sumber: Mbahseno, 2015)

BAB III METODE KEGIATAN

3.1 Tempat dan Waktu

Lokasi pembuatan Tugas akhir ini adalah di ruang dosen dan halaman samping bengkel listrik, Jurusan Teknik Mesin, Politeknik Negeri Ujung Pandang. Pengujian dilaksanakan di ruang kerja dosen, bengkel mekanik, serta di area Pintu-1 Unhas. Waktu pembuatan dan pengerjaan dimulai dari bulan Maret 2021 sampai bulan September 2021.

3.2 Alat dan Bahan

3.2.1 Alat

Alat yang digunakan pada proses pembuatan dan pengerjaan ini yaitu :

1. Laptop
2. *Smartphone*
3. Multimeter
4. Tang
5. Obeng
6. *Cutter*
7. Gerinda Tangan
8. Terminal kabel
9. Gergaji besi
10. Solder

3.2.2 Bahan

Bahan yang digunakan pada proses pembuatan dan pengerjaan ini yaitu :

1. Modul Wifi NodeMCU ESP8266
2. Arduino Uno
3. Modul relay
4. Sensor suhu DS18B20
5. *Projectboard*
6. Adaptor
7. kWh meter
8. MCB
9. Rel panel aluminium
10. Kotak kontak
11. T-dos
12. Saklar
13. Kabel jumper Arduino
14. Kabel
15. Isolasi kabel
16. Pipa
17. Sekrup
18. Klem pipa
19. Timah

3.3 Prosedur/Langkah Kerja

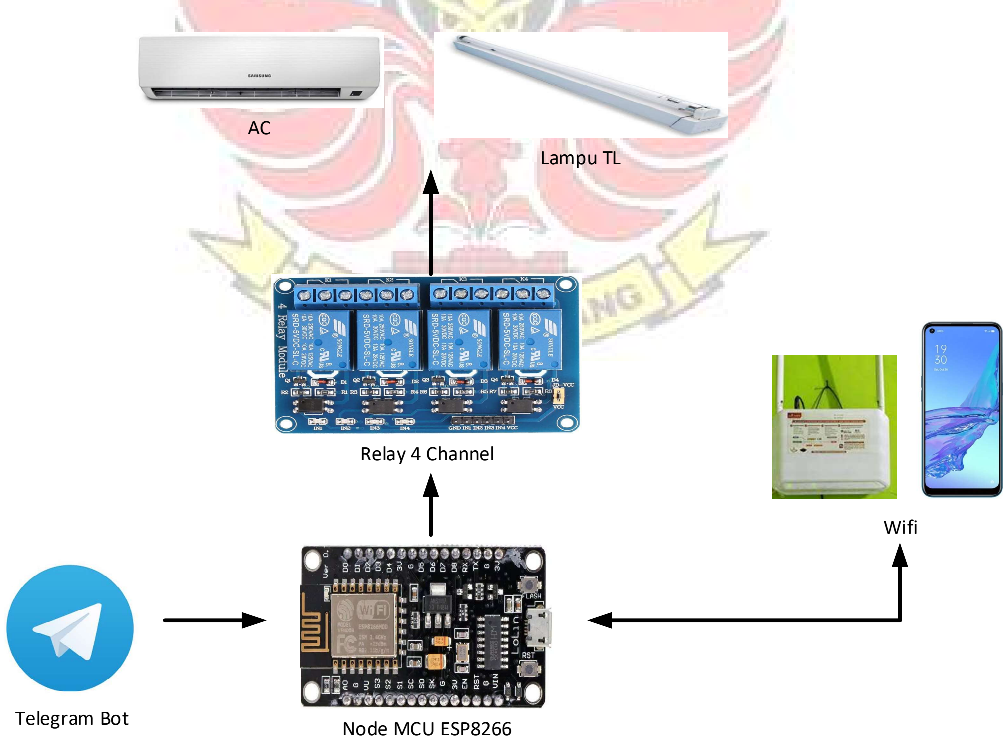
3.3.1 Studi Literatur

Pada tahap studi literatur ini, penulis mengumpulkan informasi melalui media cetak maupun elektronik yang berhubungan dengan judul yang akan diangkat agar memudahkan pengerjaan pada tahap selanjutnya yaitu tahap perancangan.

3.3.2 Tahap Perancangan

3.3.2.1 Gambaran Umum Sistem

Berikut ini adalah gambaran umum sistem pengendali beban kelistrikan berbasis *Internet of Things* (IoT).



Gambar 3. 1 Gambaran umum sistem

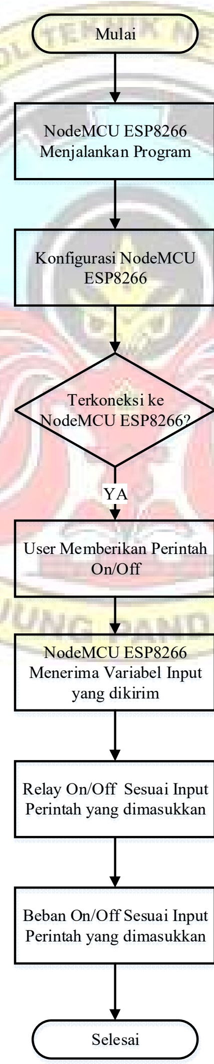
Pada perancangan sistem secara umum ini terdiri dari beberapa komponen yaitu modul wifi NodeMCU ESP8266 dan modul relay. Aplikasi Telegram difungsikan sebagai pengendali dan monitoring beban kelistrikan. Sistem ini nantinya dapat meng-*on*-kan dan meng-*off*-kan beban kelistrikan menggunakan aplikasi Telegram dan dapat memberikan informasi apakah beban kelistrikan sedang terpakai atau tidak.

3.3.2.2 Model Diagram Sistem

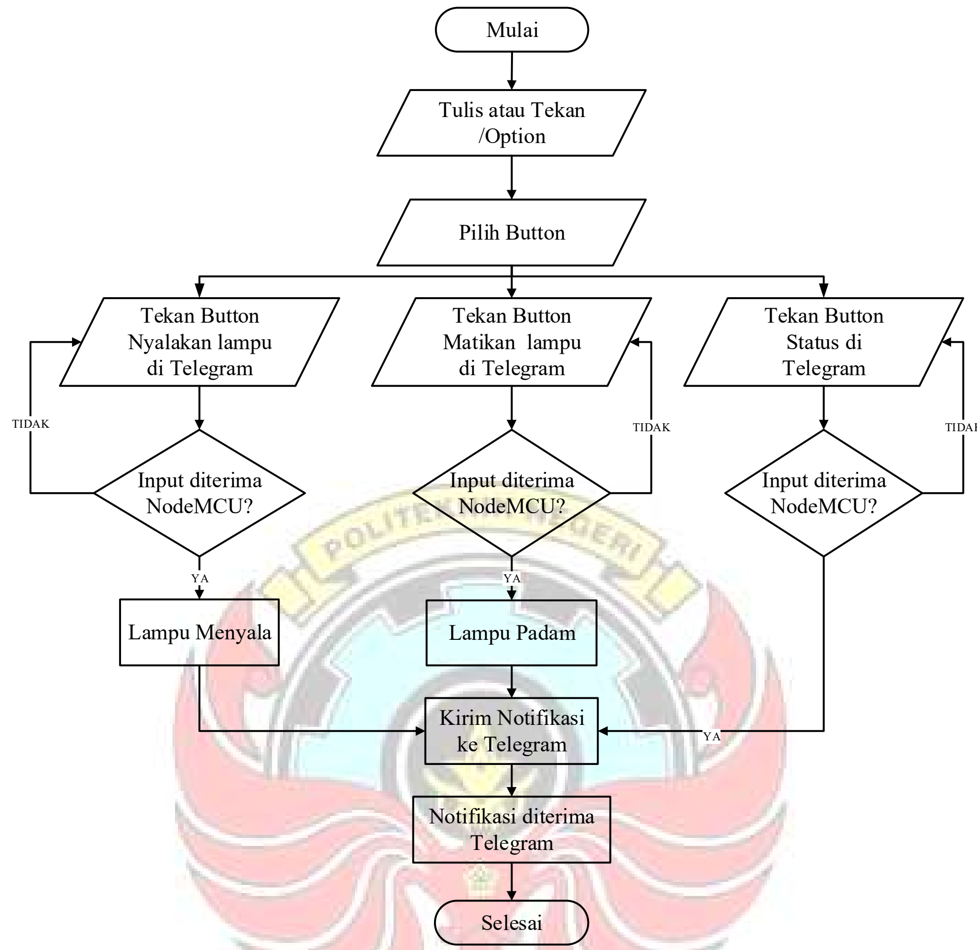
Adapun rancangan model sistem yang digunakan pada perancangan ini dapat dilihat pada Gambar 3.2.

Gambar 3. 2 Model diagram sistem

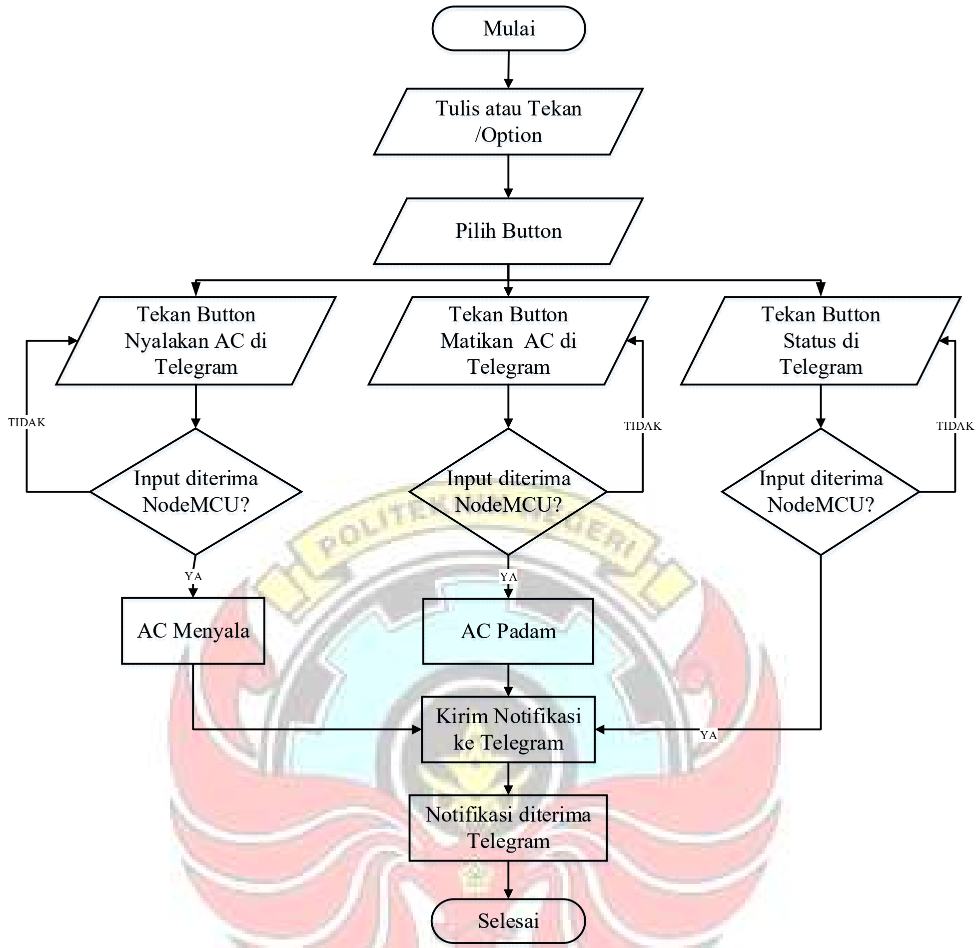
Pada Gambar 3.2, secara keseluruhan instalasi listrik terhubung dengan internet melalui beberapa perangkat seperti modul relay dan modul wifi NodeMCU ESP8266. NodeMCU ESP8266 sebagai pusat pengendalian beban dihubungkan ke modul relay. Adapun sistem kerja dari diagram blok pada Gambar 3.2 yaitu: setelah *smartphone* terhubung dengan internet, *user* akan memberikan suatu perintah (*on/off*) melalui Telegram Bot (Tugas Akhir Pengendalian Beban) yang ada pada *smartphone*. Kemudian NodeMCU ESP8266 akan menerima/memproses variabel *input* yang dikirimkan oleh *user* dan beban kelistrikan akan melakukan proses untuk *on*/*off* melalui relay yang terhubung dengan beban, kemudian status terpakai/tidak beban kelistrikan akan ditampilkan pada Telegram Bot (Tugas Akhir Pengendalian Beban) yang ada pada *smartphone*.



Gambar 3. 3 *Flowchart* sistem pengontrolan beban kelistrikan



Gambar 3. 4 *Flowchart* pengontrolan lampu pada Telegram



Gambar 3. 5 *Flowchart* pengontrolan AC pada Telegram

3.3.3 Pembuatan dan Perakitan

Setelah proses perancangan selesai, maka dilanjutkan dengan proses pembuatan dan perakitan alat. Langkah-langkah yang dikerjakan adalah sebagai berikut :

1. Membuat program pada *software* Arduino IDE.
2. Membuat bot telegram pada aplikasi telegram.
3. Merakit rangkaian sistem pengendali beban kelistrikan.
4. Mengupload program ke dalam mikrokontroler.

5. Mengaplikasikan pada instalasi kelistrikan.

3.4 Tahap Pengujian

Setelah tahap perancangan selesai, maka dilanjutkan dengan tahap pengujian alat. Adapun pengujian yang akan dilakukan yaitu:

1. Melakukan pengujian koneksi aplikasi Telegram Bot dengan nodeMCU ESP8266.
2. Melakukan pengujian sistem kontrol *on*/*off* lampu.
3. Melakukan pengujian sistem kontrol *on*/*off* AC.
4. Mengukur pemakaian energi listrik.

3.5 Teknik Analisis Data

Adapun langkah-langkah analisis data yang akan dilakukan yaitu sebagai berikut:

1. Melakukan analisa pengujian koneksi aplikasi Telegram Bot dengan NodeMCU ESP8266.
2. Melakukan analisa waktu rata-rata pada sistem kontrol *on*/*off* lampu.
3. Melakukan analisa waktu rata-rata pada sistem kontrol *on*/*off* AC.
4. Melakukan analisa penggunaan energi listrik.

BAB IV HASIL DAN DESKRIPSI KEGIATAN

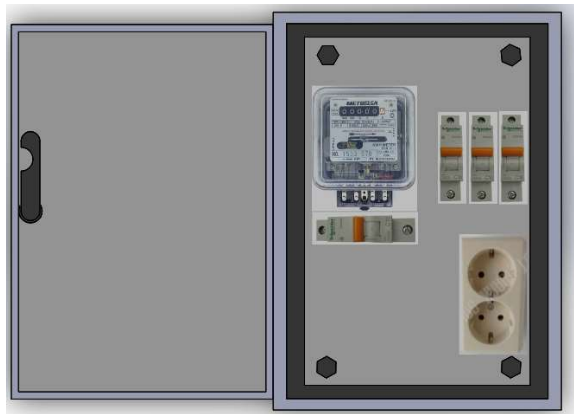
4.1 Hasil Perancangan

Hasil perancangan pengendali beban kelistrikan berbasis *Internet of Things* (IoT) dipasang pada ruang kerja dengan ukuran 3×9 m dengan beberapa peralatan listrik antara lain: 2 AC dan 2 lampu. Gambar ruang kerja dapat dilihat pada Gambar 4.1.

Gambar 4. 1 Ruang kerja yang dikendalikan dengan IoT

4.1.1 Hasil Perancangan Sistem Pengontrolan Beban Kelistrikan

4.1.1.1 Sistem Daya



(a)

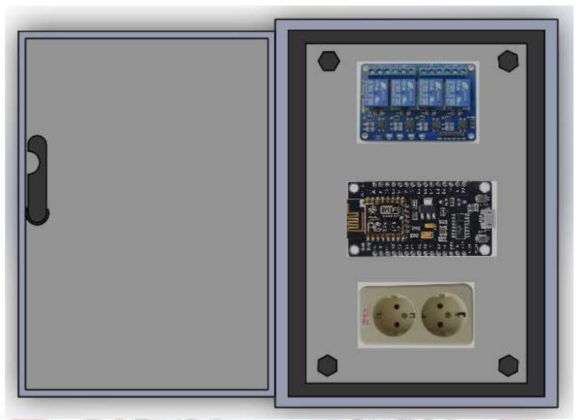
(b)

Gambar 4. 2 Hasil perancangan sistem daya. (a) Desain komponen, (b) *Layout* penyaklaran dan alat ukur

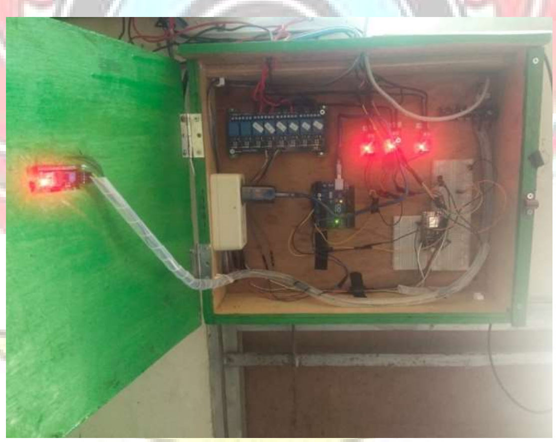
Gambar 4.2 merupakan hasil perancangan sistem daya, dimana pada *layout* penyaklaran dan alat ukur terdapat kWh meter, MCB (*Miniature Circuit*

Breaker) dan kotak kontak. Prinsip kerja dari sistem daya ini adalah memutus dan menyambung arus listrik.

4.1.1.2 Sistem Kontrol



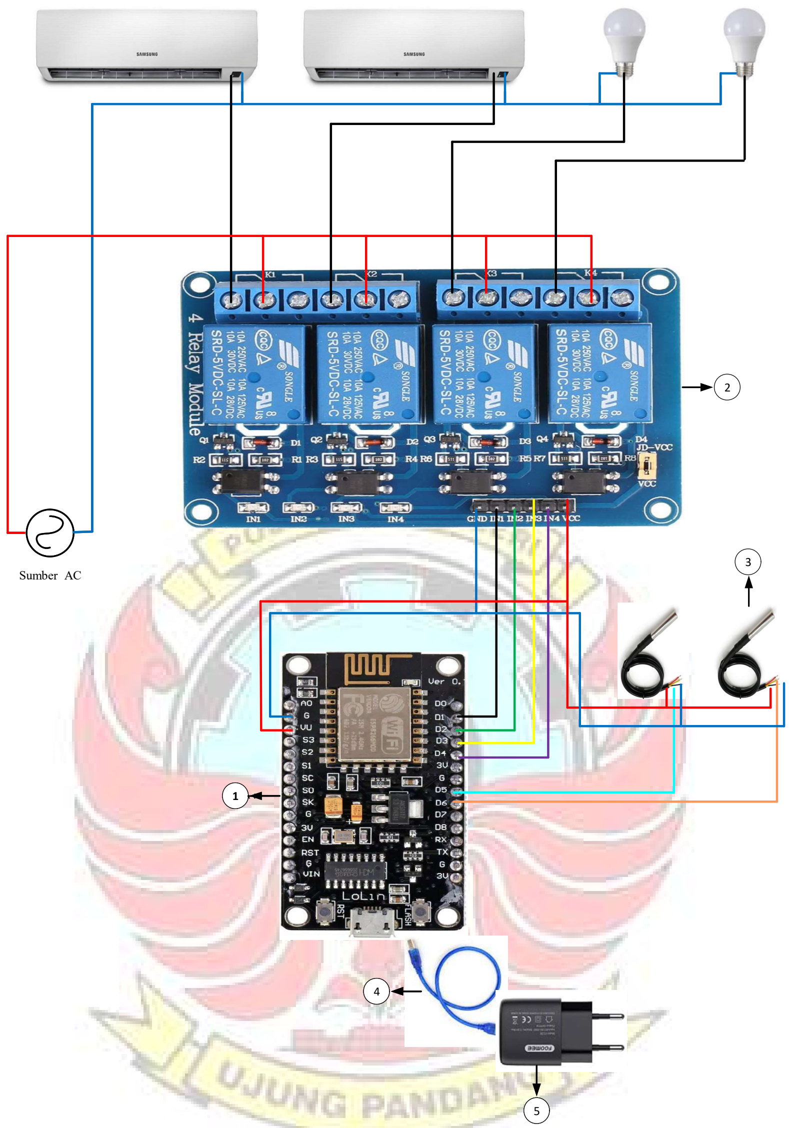
(a)



(b)

Gambar 4. 3 Hasil perancangan sistem kontrol. (a) Desain komponen, (b) *Layout* rangkaian kontrol

Gambar 4.3b merupakan *layout* rangkaian kontrol, dimana pada rangkaian ini terdapat modul wifi NodeMCU ESP8266, *power supply* 12V, relay 8 *channel* dan *board* PCB yang disimpan pada sebuah panel kontrol. NodeMCU ESP8266 digunakan untuk pengontrolan beban kelistrikan menggunakan sistem IoT.

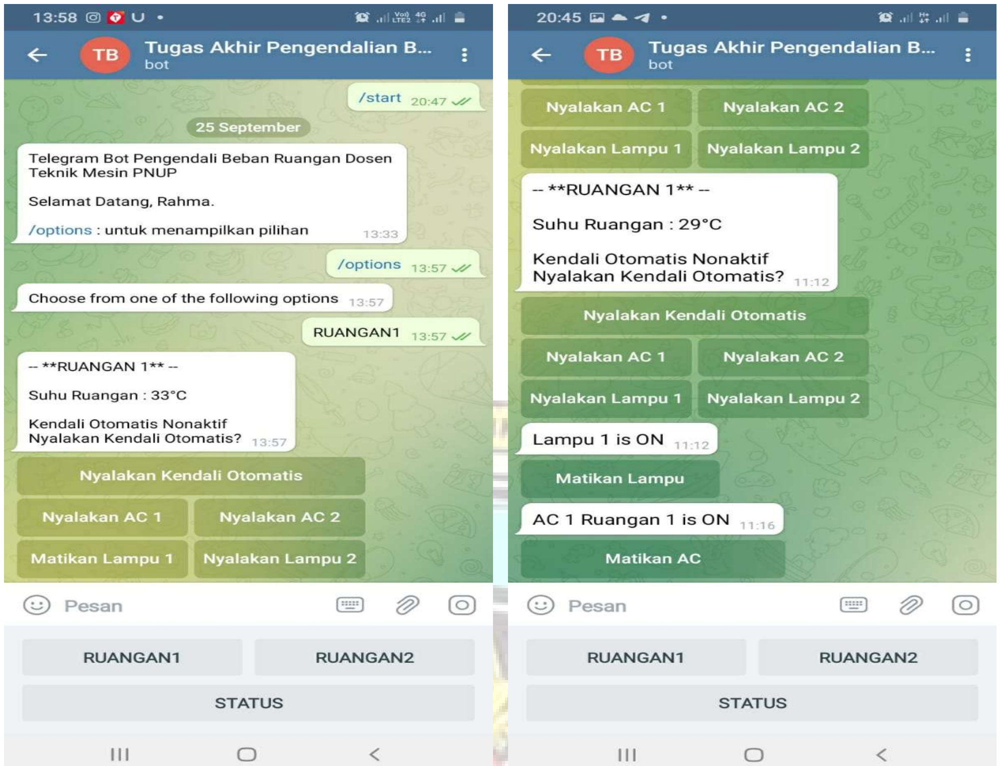


Gambar 4. 4 Skema rangkaian sistem pengontrolan beban kelistrikan

Keterangan:

1. Modul wifi NodeMCU ESP8266
2. Relay
3. Sensor suhu
4. Soket USB
5. Adaptor

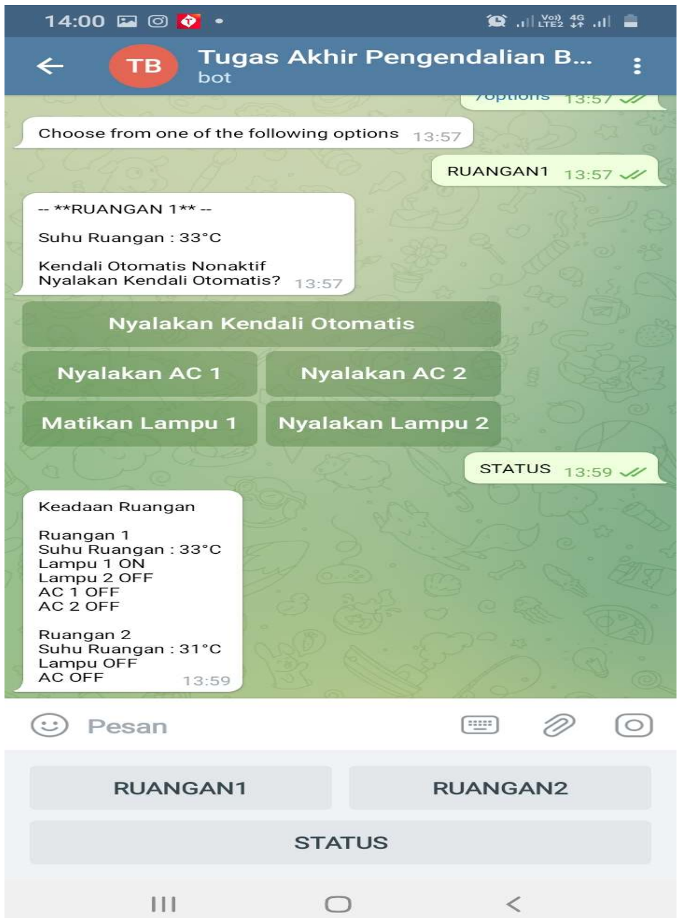
4.1.2 Hasil Perancangan Telegram Bot



Gambar 4. 5 Tampilan menu kontrol

Gambar 4.5 merupakan hasil perancangan menu-menu pengontrolan pada aplikasi Telegram Bot "Tugas Akhir Pengendalian Beban", di mana terdapat tampilan menu kontrol (RUANGAN1 dan RUANGAN2) dan menu monitor (STATUS). Gambar 4.5 merupakan tampilan ketika *user* telah masuk pada aplikasi Telegram Bot "Tugas Akhir Pengendalian Beban". Di mana untuk melakukan pengontrolan, *user* terlebih dahulu mengklik *start*, kemudian mengklik */options* dan akan muncul menu kontrol (RUANGAN1 dan RUANGAN2). Setelah itu *user* mengklik RUANGAN1 dan akan muncul beberapa perintah untuk meng-*on*-kan lampu dan AC. Apabila perintah *user* sukses maka aplikasi

Telegram akan menerima respon balik sesuai dengan perintah yang sudah dijalankan.



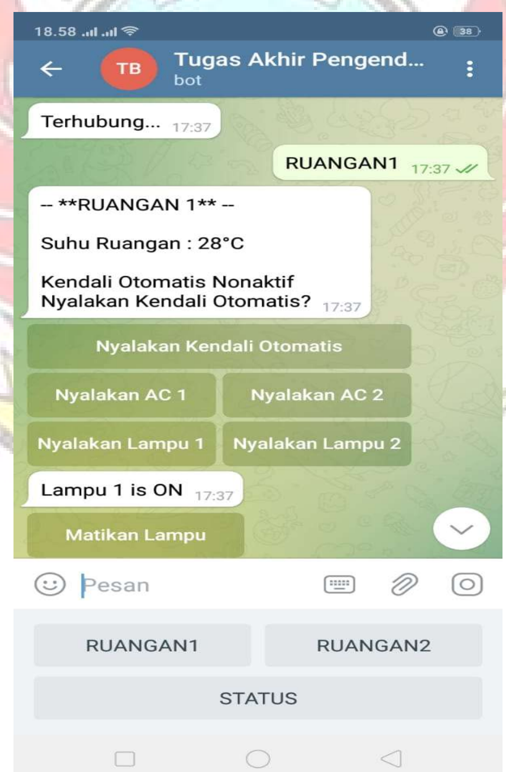
Gambar 4. 6 Tampilan status

Gambar 4.6 merupakan tampilan status keseluruhan beban kelistrikan yang telah dikendalikan melalui aplikasi Telegram. Apabila pengendalian sukses, maka aplikasi Telegram akan menerima *feedback* yang memberitahukan beban kelistrikan mana yang *on* dan yang *off*.

4.2 Pengujian

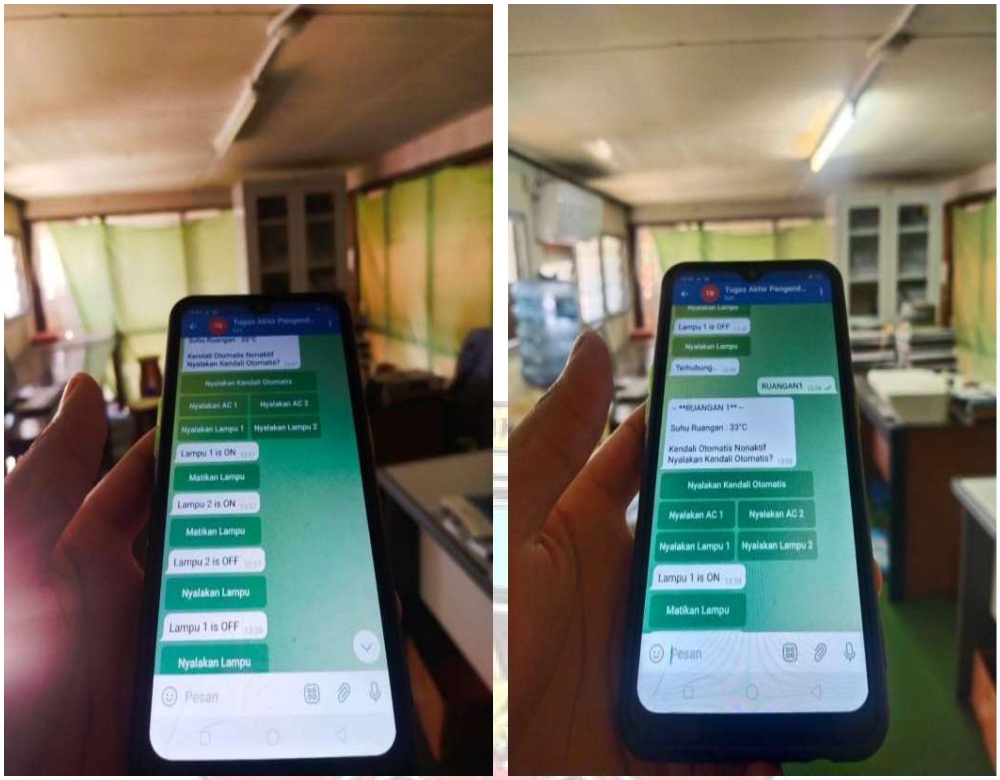
4.2.1 Pengujian Koneksi Aplikasi Telegram Bot dan NodeMCU ESP8266

Pengujian koneksi dilakukan untuk mengetahui apakah aplikasi Telegram Bot berhasil mengirim perintah ke NodeMCU ESP8266. Untuk pengujian pengiriman perintah dilakukan dengan cara menekan tombol perintah (*on*/*off*) pada aplikasi Telegram Bot dan NodeMCU ESP8266 akan menjalankan *output* sesuai perintah yang diberikan, seperti pada Gambar 4.7 diperlihatkan hasil pengiriman variabel perintah (*on*/*off*) telah terbaca dan adanya *feedback* yang menampilkan status beban kelistrikan telah *on*/*off*.



Gambar 4. 7 Koneksi aplikasi Telegram Bot dan NodeMCU ESP8266

4.2.2 Pengujian Sistem Kontrol *On*/*Off* pada Lampu



Gambar 4. 8 Tampilan aplikasi Telegram Bot pada *smartphone* dalam pengujian *on*/*off* lampu

Pengujian sistem kontrol on/*off* pada lampu dilakukan untuk mengetahui apakah perintah (*on*/*off*) dari aplikasi Telegram Bot dapat dijalankan oleh modul wifi NodeMCU ESP8266, untuk mengetahui jangkauan antara aplikasi dengan *hardware*, dan juga untuk menghitung selang waktu *input* dan *output*-nya. Pengujian dilakukan pada 3 lokasi yang berbeda dengan meng-*on*-kan dan meng-*off*-kan lampu melalui aplikasi Telegram menggunakan *provider* Telkomsel dan *provider* XL.

Tabel 4. 1 Data hasil pengujian Telegram Bot di area *Hardware* dalam pengontrolan lampu di Ruang Kerja menggunakan *Provider* Telkomsel

No	Meng-*on*-kan Lampu			Meng-*off*-kan Lampu		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	09:00:00	09:00:07.02	7,02	09:10:00	09:10:04.94	4,94
2	09:20:00	09:20:06.07	6,07	09:30:00	09:30:02.90	2,90
3	09:40:00	09:40:08.63	8,63	09:50:00	09:50:03.39	3,39
4	10:00:00	10:00:05.91	5,91	10:10:00	10:10:06.56	6,56
5	10:10:00	11:10:05.58	5,58	10:20:00	11:20:03.09	3,09

Tabel 4. 2 Data hasil pengujian Telegram Bot di area Bengkel Mekanik dalam pengontrolan lampu di Ruang Kerja menggunakan *Provider* Telkomsel

No	Meng-*on*-kan Lampu			Meng-*off*-kan Lampu		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	11:00:00	11:00:11.01	11	11:10:00	11:10:05.21	5,21
2	11:20:00	11:20:11.45	11,45	11:30:00	11:30:05.75	5,75
3	11:40:00	11:40:12.90	12,90	11:50:00	11:50:06.93	6,93
4	12:00:00	12:00:10.05	10,05	12:10:00	12:10:06.09	6,09
5	12:20:00	12:20:12.56	12,56	12:30:00	12:30:05.80	5,80

Tabel 4. 3 Data hasil pengujian Telegram Bot di area Pintu-1 Unhas dalam pengontrolan lampu di Ruang Kerja menggunakan *Provider* Telkomsel

No	Meng-*on*-kan Lampu			Meng-*off*-kan Lampu		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	16:00:00	16:00:09.33	9,33	16:10:00	16:10:07.02	7,02
2	16:20:00	16:20:10.23	10,23	16:30:00	16:30:08.73	8,73
3	16:40:00	16:40:09.88	9,88	16:50:00	16:50:07.66	7,66
4	17:00:00	17:00:09.02	9,02	17:10:00	17:10:07.89	7,89
5	17:20:00	17:20:11.98	11,98	17:30:00	17:30:08.80	8,80

Tabel 4. 4 Data hasil pengujian Telegram Bot di area *Hardware* dalam pengontrolan lampu di Ruang Kerja menggunakan *Provider* XL

No	Meng-*on*-kan Lampu			Meng-*off*-kan Lampu		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	09:10:00	09:10:07.33	7,33	09:15:00	09:15:04.96	4,96
2	09:20:00	09:20:08.18	8,18	09:25:00	09:25:03.90	3,90
3	09:30:00	09:30:08.63	8,63	09:35:00	09:35:04.39	4,39
4	09:40:00	09:40:07.75	7,75	09:45:00	09:45:05.56	5,56
5	09:50:00	09:50:07.88	7,88	09:55:00	09:55:04.09	4,09

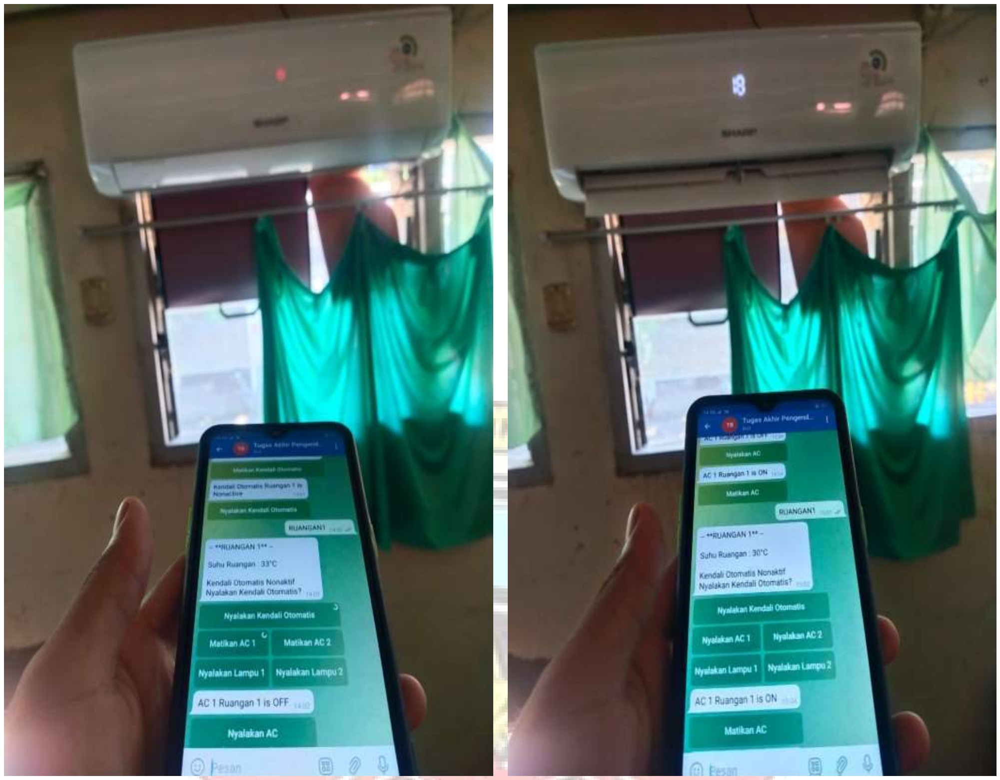
Tabel 4. 5 Data hasil pengujian Telegram Bot di area Bengkel Mekanik dalam pengontrolan lampu di Ruang Kerja menggunakan *Provider* XL

No	Meng-*on*-kan Lampu			Meng-*off*-kan Lampu		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	11:10:00	11:10:13.05	13.05	11:15:00	11:15:06.77	6,77
2	11:20:00	11:20:12.45	12,45	11:25:00	11:25:06.89	6,89
3	11:30:00	11:30:12.90	12,90	11:35:00	11:35:06.67	6,67
4	11:40:00	11:40:13.45	13,45	12:45:00	12:45:06.24	6,24
5	11:50:00	11:50:12.65	12,65	12:55:00	12:55:06.78	6,78

Tabel 4. 6 Data hasil pengujian Telegram Bot di area Pintu-1 Unhas dalam pengontrolan lampu di Ruang Kerja menggunakan *Provider* XL

No	Meng-*on*-kan Lampu			Meng-*off*-kan Lampu		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	16:10:00	16:10:10.44	10,44	16:15:00	16:15:07.34	7,02
2	16:20:00	16:20:10.34	10,34	16:25:00	16:25:08.76	8,73
3	16:30:00	16:30:11.77	11,77	16:35:00	16:35:08.66	8,66
4	16:40:00	16:40:11.02	11,02	17:45:00	17:45:08.89	8,89
5	16:50:00	16:50:11.93	11,93	17:55:00	17:55:08.80	8,80

4.2.3 Pengujian Sistem Kontrol *On*/*Off* pada AC



Gambar 4. 9 Tampilan aplikasi Telegram Bot pada *smartphone* dalam pengujian *on*/*off* AC

Pengujian sistem kontrol on/*off* pada AC dilakukan untuk mengetahui apakah perintah (*on*/*off*) dari aplikasi Telegram Bot dapat dijalankan oleh modul wifi NodeMCU ESP8266, untuk mengetahui jangkauan antara aplikasi dengan *hardware* dan juga untuk menghitung selang waktu *input* dan *output*-nya. Pengujian dilakukan pada 3 lokasi yang berbeda yang berbeda dengan meng-*on*-kan dan meng-*off*-kan AC melalui aplikasi telegram menggunakan *provider* Telkomsel dan *provider* XL.

Tabel 4. 7 Data hasil pengujian Telegram Bot di area *Hardware* dalam pengontrolan AC di Ruang Kerja menggunakan *Provider* Telkomsel

No	Meng-*on*-kan AC			Meng-*off*-kan AC		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	09:05:00	09:05:07.28	7,28	09:15:00	09:15:06.05	6,05
2	09:25:00	09:25:05.91	5,91	09:35:00	09:35:07.09	7,09
3	09:45:00	09:45:08.06	8,06	09:55:00	09:55:08.29	8,29
4	10:05:00	10:05:05.86	5,86	10:15:00	10:15:04.95	4,95
5	10:25:00	11:25:05.76	5,76	10:35:00	11:35:02.95	2,95

Tabel 4. 8 Data hasil pengujian Telegram Bot di area Bengkel Mekanik dalam pengontrolan AC di Ruang Kerja menggunakan *Provider* Telkomsel

No	Meng-*on*-kan AC			Meng-*off*-kan AC		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	13:00:00	13:00:08.13	8,13	13:10:00	13:10:05.18	5,18
2	13:20:00	13:20:06.55	6,55	13:30:00	13:30:04.42	4,42
3	13:40:00	13:40:05.75	5,75	13:50:00	13:50:05.02	5,02
4	14:05:00	14:00:05.05	5,05	14:14:00	14:10:06.33	6,33
5	14:20:00	14:20:06.12	6,12	14:30:00	14:30:06.95	6,95

Tabel 4. 9 Data hasil pengujian Telegram Bot di area Pintu-1 Unhas dalam pengontrolan AC di Ruang Kerja menggunakan Provider Telkomsel

No	Meng-*on*-kan AC			Meng-*off*-kan AC		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	16:05:00	16:05:07.66	7,66	16:15:00	16:15:05.68	4,68
2	16:25:00	16:25:07.97	7,97	16:35:00	16:35:03.11	3,11
3	13:45:00	13:45:07.42	7,42	16:55:00	16:55:03.35	3,35
4	17:05:00	17:05:06.28	6,28	17:15:00	17:15:05.40	5,40
5	17:25:00	17:25:08.38	8,38	17:35:00	17:35:05.83	5,83

Tabel 4. 10 Data hasil pengujian Telegram Bot di area *Hardware* dalam pengontrolan AC di Ruang Kerja menggunakan *Provider* XL

No	Meng-*on*-kan AC			Meng-*off*-kan AC		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	09:10:00	09:10:08.38	8,38	09:15:00	09:15:06.47	6,47
2	09:30:00	09:30:07.91	7,91	09:35:00	09:35:05.79	5,79
3	09:50:00	09:50:08.06	8,06	09:55:00	09:55:04.29	4,29
4	10:10:00	10:10:09.86	9,86	10:15:00	10:15:04.95	4,95
5	10:30:00	10:30:09.76	9,76	10:35:00	11:35:05.95	5,95

Tabel 4. 11 Data hasil pengujian Telegram Bot di area Bengkel Mekanik dalam pengontrolan AC di Ruang Kerja menggunakan *Provider* XL

No	Meng-*on*-kan AC			Meng-*off*-kan AC		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	13:05:00	13:05:08.13	8,13	13:05:00	13:05:05.28	5,28
2	13:15:00	13:15:08.55	8,55	13:15:00	13:15:04.42	4,42
3	13:25:00	13:25:06.75	6,75	13:25:00	13:25:05.02	5,02
4	13:35:00	13:35:07.05	7,05	13:35:00	13:35:06.33	6,33
5	13:45:00	13:45:07.12	7,12	13:45:00	13:45:07.95	7,95

Tabel 4. 12 Data hasil pengujian Telegram Bot di area Pintu-1 Unhas dalam pengontrolan AC di Ruang Kerja menggunakan Provider XL

No	Meng-*on*-kan AC			Meng-*off*-kan AC		
	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]	Waktu Pengontrolan	Waktu Respon Beban	Delay [s]
1	16:05:00	16:05:08.60	8,60	16:15:00	16:15:05.55	5,55
2	16:25:00	16:25:08.95	8,85	16:35:00	16:35:04.89	4,89
3	16:45:00	16:45:08.42	8,42	16:55:00	16:55:04.68	4,68
4	17:05:00	17:05:07.28	7,28	17:15:00	17:15:05.40	5,40
5	17:25:00	17:25:08.38	8,38	17:35:00	17:35:05.83	5,83

4.2.4 Pengukuran Pemakaian Energi Listrik

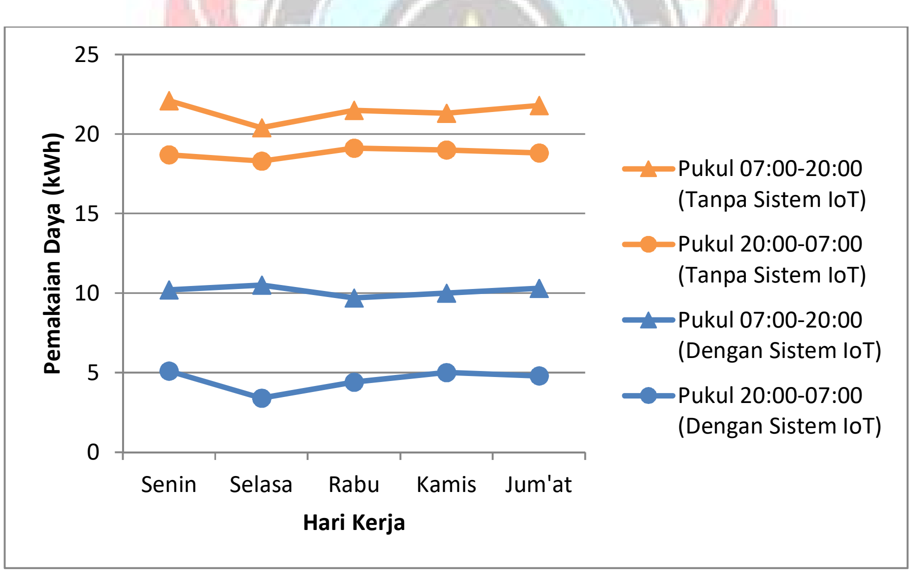
Pengukuran pemakaian energi listrik dilakukan untuk mengetahui jumlah pemakaian energi listrik yang digunakan pada ruang kerja. Pengukuran pemakaian energi listrik ini dilakukan dengan simulasi dua kondisi. Kondisi pertama yaitu mengukur jumlah pemakaian energi listrik tanpa sistem IoT, di mana seluruh beban kelistrikan di-*on*-kan maupun di-*off*-kan dengan saklar. Kondisi kedua yaitu mengukur jumlah pemakaian energi listrik dengan sistem IoT, di mana beban kelistrikan di-*on*-kan maupun di-*off*-kan melalui aplikasi Telegram sehingga beban kelistrikan bisa dikendalikan dan dimonitor kapan pun dan di mana pun selama sistem terhubung dengan jaringan internet.

Tabel 4. 13 Data hasil pengukuran pemakaian energi listrik tanpa sistem IoT

<table>
<tr><th>No</th><th>Hari Kerja</th><th>Waktu</th><th>Pemakaian Daya [kWh]</th><th>Waktu</th><th>Pemakaian Daya [kWh]</th></tr>
<tr><td>1</td><td>Senin</td><td rowspan="5">07:00-20:00</td><td>22,1</td><td rowspan="5">20:00-07:00</td><td>18,7</td></tr>
<tr><td>2</td><td>Selasa</td><td>20,4</td><td>18,3</td></tr>
<tr><td>3</td><td>Rabu</td><td>21,5</td><td>19,1</td></tr>
<tr><td>4</td><td>Kamis</td><td>21,3</td><td>19</td></tr>
<tr><td>5</td><td>Jum'at</td><td>21,8</td><td>18,8</td></tr>
<tr><td colspan="3">Jumlah</td><td>107,1</td><td>Jumlah</td><td>93,9</td></tr>
<tr><td colspan="3">Rata-rata</td><td>21,42</td><td>Rata-rata</td><td>18,78</td></tr>
<tr><td>6</td><td>Sabtu</td><td rowspan="2">07:00-07:00</td><td>30,5</td><td colspan="2" rowspan="4"></td></tr>
<tr><td>7</td><td>Ahad</td><td>30,7</td></tr>
<tr><td colspan="3">Jumlah</td><td>61,2</td></tr>
<tr><td colspan="3">Rata-rata</td><td>30,6</td></tr>
</table>

Tabel 4. 14 Data hasil pengukuran pemakaian energi listrik dengan sistem IoT

No	Hari Kerja	Waktu	Pemakaian Daya [kWh]	Waktu	Pemakaian Daya [kWh]
1	Senin	07:00-20:00	10,2	20:00-07:00	5,1
2	Selasa		10,5		3,4
3	Rabu		9,7		4,4
4	Kamis		10		5
5	Jum'at		10,3		4,8
Jumlah			50,7	Jumlah	22,7
Rata-rata			10,14	Rata-rata	4,54
6	Sabtu	07:00-07:00	3,4		
7	Ahad		3,6		
Jumlah			7		
Rata-rata			3,5		



Gambar 4. 10 Grafik perbandingan pemakaian energi listrik sebelum dan setelah pengaplikasian sistem IoT

Berdasarkan grafik pada Gambar 4.10 dapat dilihat perbedaan pemakaian energi listrik sebelum dan setelah pengaplikasian sistem IoT, di mana pemakaian energi listrik pada instalasi kelistrikan tanpa sistem IoT berkisar antara 20,4-30,7 kWh cenderung lebih tinggi jika dibandingkan dengan pemakaian energi listrik dengan sistem IoT yang berkisar antara 3,4-10,5 kWh. Hal ini telah sesuai dengan tujuan utama pengaplikasian sistem IoT untuk penghematan energi, dimana ketika *user* lupa untuk meng-*off*-kan beban kelistrikan saat keluar ruangan, maka *user* bisa meng-*off*-kan beban kelistrikan di mana pun posisi *user* berada. Sehingga status beban terpakai/tidak dapat dimonitor dan dikendalikan melalui aplikasi Telegram pada *smartphone*.

4.3 Pembahasan

4.3.1 Analisa Hasil Pengujian Koneksi Aplikasi Telegram Bot dan NodeMCU ESP8266

Dari hasil pengujian yang telah dilakukan pada pengujian koneksi aplikasi Telegram Bot dapat dilihat bahwa NodeMCU ESP8266 mampu menjalankan perintah yang diberikan oleh *user* melalui aplikasi Telegram Bot, baik dalam meng-*on*-kan dan meng-*off*-kan lampu maupun meng-*on*-kan dan meng-*off*-kan AC.

4.3.2 Analisa Hasil Pengujian Sistem Kontrol *On*/*Off* pada Lampu

Berdasarkan data hasil pengujian sistem kontrol *on*/*off* pada lampu yang diperlihatkan pada Tabel 4.1-Tabel 4.3 diketahui bahwa waktu untuk meng-*on*-kan dan meng-*off*-kan lampu berbeda-beda. Perintah *on*/*off* yang dikirim melalui aplikasi Telegram mengalami *delay* sebelum modul wifi NodeMCU ESP8266 menjalankan perintah *on*/*off* tersebut. Ada pun untuk mengetahui rata-rata *delay* yang dibutuhkan untuk meng-*on*-kan dan meng-*off*-kan lampu melalui aplikasi Telegram Bot dapat dilihat sebagai berikut:

1) Pengujian pada area *hardware*

a) Rata-rata *delay* untuk meng-*on*-kan lampu

$$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{33,21}{5} = 6,64 \text{ s}$$

b) Rata-rata *delay* untuk meng-*off*-kan lampu

$$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{20,88}{5} = 4,18 \text{ s}$$

2) Pengujian pada area Bengkel Mekanik

a) Rata-rata *delay* untuk meng-*on*-kan lampu

$$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{57,96}{5} = 11,59 \text{ s}$$

b) Rata-rata *delay* untuk meng-*off*-kan lampu

$$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{29,78}{5} = 5,96 \text{ s}$$

3) Pengujian pada area Pintu-1 Unhas

a) Rata-rata *delay* untuk meng-*on*-kan lampu

$$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{50,44}{5} = 10,09 \text{ s}$$

b) Rata-rata *delay* untuk meng-*off*-kan lampu

$$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{40,1}{5} = 8,02 \text{ s}$$

4.3.3 Analisa Hasil Pengujian Sistem Kontrol *On*/*Off* pada AC

Berdasarkan data hasil pengujian sistem kontrol *on*/*off* pada AC yang diperlihatkan pada Tabel 4.4-Tabel 4.6 diketahui bahwa waktu untuk meng-*on*-kan dan meng-*off*-kan AC berbeda-beda. Perintah *on*/*off* yang dikirim melalui aplikasi Telegram mengalami *delay* sebelum modul wifi NodeMCU ESP8266 menjalankan perintah *on*/*off* tersebut. Ada pun untuk mengetahui rata-rata *delay* yang dibutuhkan untuk meng-*on*-kan dan meng-*off*-kan AC melalui aplikasi Telegram Bot dapat dilihat sebagai berikut:

1) Pengujian pada area *hardware*

   a) Rata-rata *delay* untuk meng-*on*-kan AC

   $$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{32,87}{5} = 6,57 \text{ s}$$

   b) Rata-rata *delay* untuk meng-*off*-kan AC

   $$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{29,33}{5} = 5,87 \text{ s}$$

2) Pengujian pada area Bengkel Mekanik

   a) Rata-rata *delay* untuk meng-*on*-kan AC

   $$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{31,6}{5} = 6,32 \text{ s}$$

   b) Rata-rata *delay* untuk meng-*off*-kan AC

   $$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{27,9}{5} = 5,58 \text{ s}$$

3) Pengujian pada area Pintu-1 Unhas

   a) Rata-rata *delay* untuk meng-*on*-kan AC

   $$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{37,71}{5} = 7,54 \text{ s}$$

   b) Rata-rata *delay* untuk meng-*off*-kan AC

   $$\frac{\sum_{n=1}^{5} t_n}{5} = \frac{22,37}{5} = 4,47 \text{ s}$$

4.3.4 Analisa Pemakaian Energi Listrik

Berdasarkan grafik pada Gambar 4.10 dapat dilihat perbedaan pemakaian energi listrik sebelum dan setelah pengaplikasian sistem IoT, di mana pemakaian energi listrik pada instalasi kelistrikan tanpa sistem IoT berkisar antara 20,4-30,7 kWh cenderung lebih tinggi jika dibandingkan dengan pemakaian energi listrik dengan sistem IoT yang berkisar antara 3,4-10,5 kWh. Hal ini telah sesuai dengan tujuan utama pengaplikasian sistem IoT untuk penghematan energi, dimana ketika *user* lupa untuk meng-*off*-kan beban kelistrikan saat keluar ruangan, maka *user* bisa meng-*off*-kan beban kelistrikan di mana pun posisi *user* berada. Sehingga status beban terpakai/tidak dapat dimonitor dan dikendalikan melalui aplikasi Telegram pada *smartphone*.

Berdasarkan Tabel 4.7 dan Tabel 4.8 diketahui bahwa rata-rata pemakaian energi listrik tanpa sistem IoT (hari Senin-Jum'at) pada pukul 07:00-20:00 dan pemakaian energi listrik dengan sistem IoT (hari Senin-Jum'at) pada pukul 07:00-20:00 masing-masing 107,1 kWh dan 50,7 kWh sehingga:

Penghematan energi = Pemakaian energi tanpa sistem IoT - Pemakaian energi dengan sistem IoT

= (107,1 − 50,7) kWh

= 56,4 kWh

Jadi persentase penghematan:

Persentase penghematan = (Penghematan energi : Pemakaian energi listrik tanpa sistem IoT) × 100%

= (56,4 : 107,1) × 100%

= 52,66%

Jika dirupiahkan:

Estimasi tagihan listrik = Jumlah penghematan energi × tarif lisrik per-kWh

= 56,4 kWh × *)Rp. 1444,70

= Rp. 81.481,08

Jadi dengan penggunaaan sistem IoT ini dapat menghemat Rp. 81.481,08 selama pengujian.

Berdasarkan Tabel 4.7 dan Tabel 4.8 diketahui bahwa rata-rata pemakaian energi listrik tanpa sistem IoT (hari Senin-Jum'at) pada pukul 20:00-07:00 dan pemakaian energi listrik dengan sistem IoT (hari Senin-Jum'at) pada pukul 20:00-07:00 masing-masing 93,9 kWh dan 22,7 kWh sehingga:

Penghematan energi = Pemakaian energi tanpa sistem IoT – Pemakaian energi dengan sistem IoT

= (93,9 – 22,7) kWh

= 71,2 kWh

Jadi persentase penghematan:

Persentase penghematan = (Penghematan energi : Pemakaian energi listrik tanpa sistem IoT) × 100%

= (71,2 : 93,9) × 100%

= 75,83%

Jika dirupiahkan:

Estimasi tagihan listrik = Jumlah penghematan energi × tarif lisrik per-kWh

= 71,2 kWh × *)Rp. 1444,70

= Rp. 102.862,64

Jadi dengan penggunaaan sistem IoT ini dapat menghemat Rp. 102.862,64 selama pengujian.

Berdasarkan Tabel 4.7 dan Tabel 4.8 diketahui bahwa rata-rata pemakaian energi listrik tanpa sistem IoT (hari Sabtu-Ahad) pada pukul 07:00-07:00 dan pemakaian energi listrik dengan sistem IoT (hari Sabtu-Ahad) pada pukul 07:00-07:00 masing-masing kWh dan 22,7 kWh sehingga:

Penghematan energi = Pemakaian energi tanpa sistem IoT - Pemakaian energi dengan sistem IoT

= (61,2 – 7) kWh

= 54,2 kWh

Jadi persentase penghematan:

Persentase penghematan = (Penghematan energi : Pemakaian energi listrik tanpa sistem IoT) × 100%

= (54,2 : 61,2) × 100%

= 88,56%

Jika dirupiahkan:

Estimasi tagihan listrik = Jumlah penghematan energi × tarif lisrik per-kWh

= 54,2 kWh × *)Rp. 1444,70

= Rp. 78.302,74

Jadi dengan penggunaaan sistem IoT ini dapat menghemat Rp. 78.302,74 selama pengujian.

Keterangan:

*) Berdasarkan penetapan penyesuaian tarif tenaga listrik (*tariff adjustment*)

Tabel 4. 15 Hasil analisis data pemakaian energi listrik

No	Hari Kerja	Jumlah Pemakaian Daya Tanpa Sistem IoT	Jumlah Pemakaian Daya Dengan Sistem IoT	Penghematan Energi	Persentase Penghematan
1	Senin-Jumat 07:00-20:00	107,1 kWh	50,7 kWh	56,4 kWh	52,66%
2	Senin-Jumat 20:00-07:00	93,9 kWh	22,7 kWh	71,2 kWh	75,83%
3	Sabtu-Ahad 07:00-07:00	61,2 kWh	7	54,2 kWh	88,56%

BAB V KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan dan pengujian sistem, maka dapat ditarik kesimpulan sebagai berikut:

1. Monitoring terpakai/tidaknya beban listrik pada Ruang Kerja dari jarak jauh menggunakan aplikasi Telegram pada *smartphone* telah dilakukan dalam Tugas Akhir ini. Hasil pengujian menunjukkan sistem bekerja dengan baik sehingga status terpakai/tidaknya beban kelistrikan pada Ruang Kerja dapat dimonitor pada aplikasi Telegram.
2. Dalam Tugas Akhir ini telah di-*off*-kan beban kelistrikan yang sedang terpakai pada Ruang Kerja dari jarak jauh menggunakan aplikasi Telegram dan sistem bekerja dengan baik. Hasil pengujian menunjukkan adanya *delay* ketika meng-*off*-kan beban kelistrikan dari jarak jauh pada saat pengujian ditiga lokasi yang berbeda. Pada lokasi pertama (area *Hardware*), rata-rata *delay* untuk meng-*off*-kan lampu yaitu 4,18 s dan rata-rata *delay* untuk meng-*off*-kan AC yaitu 5,87 s. Pada lokasi kedua (area Bengkel Mekanik), rata-rata *delay* untuk meng-*off*-kan lampu yaitu 5,96 s dan rata-rata *delay* untuk meng-*off*-kan AC yaitu 5,58 s. Sedangkan pada lokasi ketiga (area Pintu-1 Unhas), rata-rata *delay* untuk meng-*off*-kan lampu yaitu 8,02 s dan rata-rata *delay* untuk meng-*off*-kan AC yaitu 4,47 s.
3. Telah diteliti kemungkinan adanya penghematan energi listrik dari penerapan sistem pengendali beban kelistrikan berbasis *Internet of Things* (IoT) menggunakan aplikasi Telegram. Hasil penelitian menunjukkan bahwa ada

penghematan energi listrik sebesar 50,7 kWh dan jika di persentasekan maka persentase penghematan energi listrik sebesar 52,66% pada hari kerja,ada pun diluar hari kerja sebesar 71,2 kWh persentase penghematan 75,83% dan penghematan energi listrik sebesar 54,2 kWh persentase penghematan 88,56% pada hari libur akibat diterapkannya sistem pengendali beban kelistrikan dari jarak jauh ini.

5.2 Saran

Rancang bangun pengendali beban kelistrikan pada ruang kerja berbasis *Internet of Things* (IoT) ini masih banyak kekurangan, oleh sebab itu beberapa hal yang disarankan untuk pengembangan selanjutnya adalah sebagai berikut:

1. Pengontrolan beban kelistrikan melalui aplikasi telegram masih memiliki *delay* sehingga nodeMCU ESP8266 tidak dapat langsung menjalankan perintah ketika menu pengontrolan ditekan, maka dari itu untuk pengembangan lebih lanjut disarankan untuk menggunakan program yang dapat menghubungkan aplikasi Telegram dengan NodeMCU ESP8266 dengan lebih cepat dan menggunakan internet yang lebih stabil.
2. Menambahkan sensor daya untuk menghitung penggunaan energi listrik dengan menggunakan sistem IoT.

DAFTAR PUSTAKA

Endra, Robby Yuli dkk. 2019. *Smart Room* Menggunakan *Internet Of Things* Untuk Efisiensi Biaya dan Keamanan Ruangan. (*Online*), (https://osf.io/gz6mb), Diakses 15 Desember 2020.

Erawan, Anto. 2015. Gunakan *Smart Home*, Kenali Dulu Fitur-Fiturnya!. (*Online*), (https://www.rumah.com/berita-properti/2015/12/112343/gunakan-smart-home-kenali-dulu-fitur-fiturnya), Diakses 12 Oktober 2021.

Evendi, Rustan dan Erny Febryanty. 2019. Rancang Bangun Alat Kontrol *Switching* Kelistrikan Ruangan dengan Sensor RFID Berbasis Mikrokontroler. Laporan Tugas Akhir. Makassar: Jurusan Teknik Mesin Politeknik Negeri Ujung Pandang.

Febrianto. 2014. Apa itu Arduino Uno?. (*Online*), (https://ndoware.com/apa-itu-arduino-uno.html/amp), Diakses 12 Oktober 2021.

Kho, Dickson. 2020. Pengertian Relay dan Fungsinya. (*Online*), (https://teknikelektronika.com/pengertian-relay-fungsi-relay/), Diakses 12 Oktober 2021.

Mahali, Muhammad Izzuddin. 2016. *Smart Door Locks Based on Internet of Things Concept with Mobile Backend as a Service*. (Online), (https://journal.uny.ac.id/index.php/elinvo/article/view/14260/9453), Diakses 14 Desember 2020.

Mahardika, Derwin. 2020. Perbedaan NodeMCU, Wemos, dan ESP8266 Wifi Module untuk Perangkat IoT Mikrokontroler. (*Online*), (https://www.teknodika.com/2020/04/perbedaan-nodemcu-wemos-dan-esp8266.html?m=1), Diakses 12 Oktober 2021.

Mbahseno. 2015. Mengenal Arduino. (*Online*), (https://www.google.com/amp/s/duniaarduino.wordpress.com/2015/08/04/mengenal-arduino/amp/), Diakses 23 Oktober 2021.

Muhamad, Rizky Naga. 2019. Perancangan *Smart Room* Berbasis Arduino. Tugas Akhir. Malang: Universitas Muhammadiyah Malang.

Nurfaif, Muhammad Bagus. 2017. Rancang Bangun Sistem Rumah Cerdas menggunakan Jaringan Internet. (*Online*), (http://digilib.unila.ac.id/29811/), Diakses 15 Desember 2020.

Prasetio, Barlian Henryranu dan Dahnial Syauqy. 2017. Desain Protokol Suara Sebagai Pengendali Dalam *Smart Home* Menggunakan FPGA. (*Online*), (https://core.ac.uk/download/pdf/193292355.pdf), Diakses 15 Desember 2020.

Prastyo, Elga Aris. 2020. Sensor Suhu DS18B20. (*Online*), (https://www.edukasielektronika.com/2020/09/sensorsuhuds18b20.html), Diakses 12 Oktober 2021.

Santana, Gardewa. 2020. Apa itu *Smart Home System*?. (*Online*), (https://www.ariston.com/id-id/the-comfort-way/trik-dan-kiat/apa-itu-smart-home-system), Diakses 12 Oktober 2021.

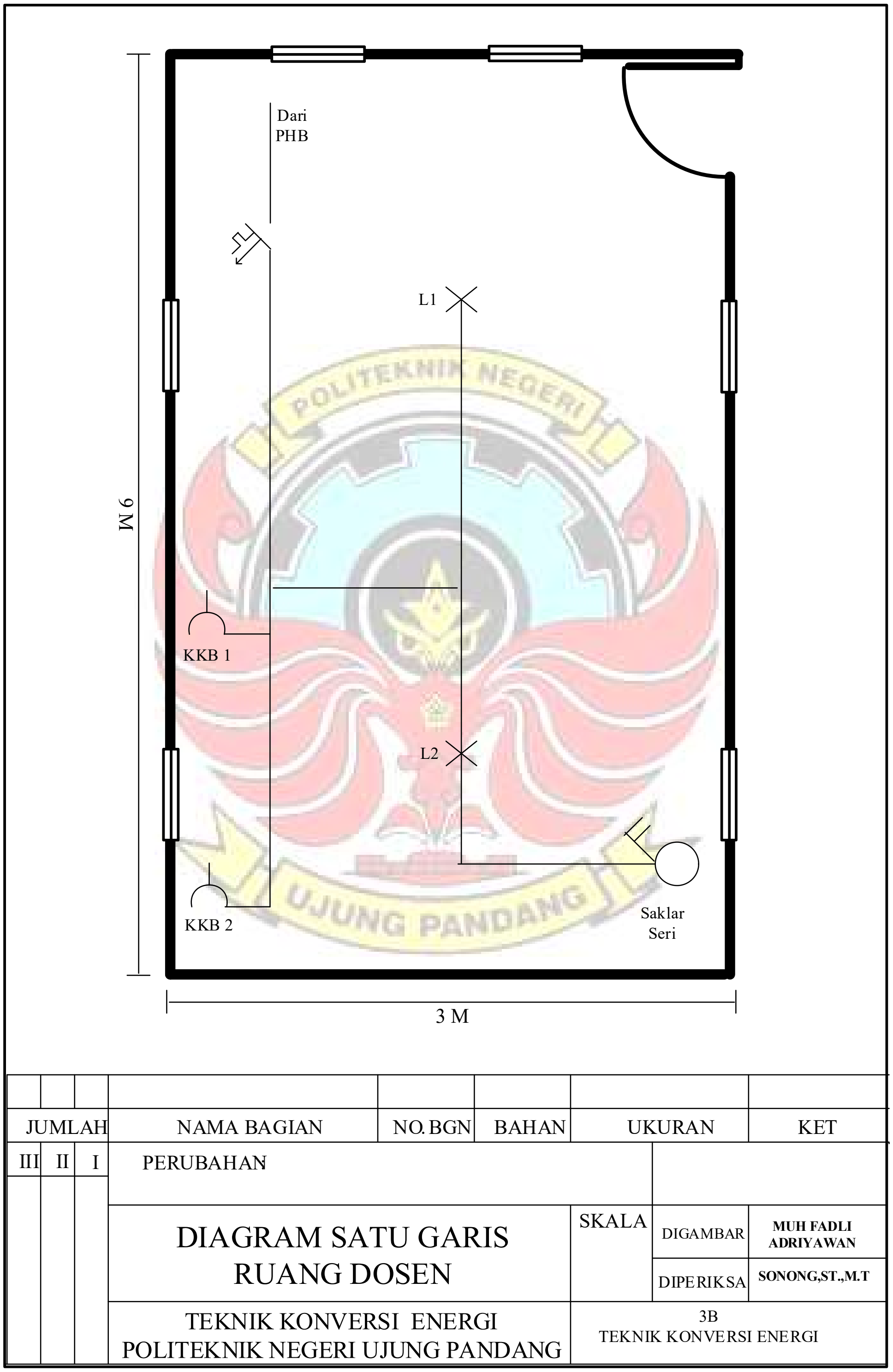
The Engineering Projects. 2018. *Introduction to* NodeMCU V3. (*Online*), (https://www.theengineeringprojects.com/2018/10/introduction-to-nodemcu-v3.html/amp), Diakses 12 Oktober 2021.

Wibowo, Danti. 2021. Definisi *Internet of Things* (IoT). (*Online*), (https://www.jojonomic.com/blog/kenali-apa-itu-internet-of-things-iot/), Diakses 7 Oktober 2021.

Wilianto dan Ade Kurniawan. 2018. Sejarah, Cara Kerja dan Manfaat *Internet of Things*. (*Online*), (https://garuda.ristekbrin.go.id/documents/detail/739466), Diakses 12 Oktober 2021.

L A M P I R A N

Lampiran 1 Diagram Satu Garis Ruang Dosen



Keterangan :

Pintu

Jendela

Kotak Kontak

Lampu

Lampiran 2 *Listing* Program Pengendali Jarak Jauh

```
#ifdef ESP32
#include <WiFi.h>
#else
#include <ESP8266WiFi.h>
#endif
#include <OneWire.h>
#include <DallasTemperature.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>   // Universal Telegram Bot Library written by Brian
Lough: https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot
#include <ArduinoJson.h>

#define ONE_WIRE_BUS_1 2
#define ONE_WIRE_BUS_2 0

// Replace with your network credentials
const char* ssid = "anonymous";
const char* password = "namakuji";

// Initialize Telegram BOT
#define BOTtoken "1978834697:AAH97eCzFBD9YT17IyhglFa05h2b25T5wdg"  // your
Bot Token (Get from Botfather)

// Use @myidbot to find out the chat ID of an individual or a group
// Also note that you need to click "start" on a bot before it can
// message you
//#define CHAT_ID "606983490"
#define CHAT_ID "2008447725"

#ifdef ESP8266
  X509List cert(TELEGRAM_CERTIFICATE_ROOT);
#endif

WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

// Checks for new messages every 1 second.
int botRequestDelay = 1000;
unsigned long lastTimeBotRan;

OneWire oneWire_1(ONE_WIRE_BUS_1);
OneWire oneWire_2(ONE_WIRE_BUS_2);

DallasTemperature sensor_1(&oneWire_1);
```

```
DallasTemperature sensor_2(&oneWire_2);

int Lampu_1 = LOW;
int Lampu_2 = LOW;
int Lampu_3 = LOW;
int AC_1 = LOW;
int AC_2 = LOW;
int AC_3 = LOW;
int Ruangan_1 = LOW;
int Ruangan_2 = LOW;

int t1,t2;

const byte R1 = 14; //AC3
const byte R2 = 12; //AC2
const byte R3 = 13; //AC1
const byte R4 = 16; //Lampu1
const byte R5 = 5; ///Lampu2
const byte R6 = 4; ///Lampu3

// Handle what happens when you receive new messages
void handleNewMessages(int numNewMessages) {
 Serial.println("handleNewMessages");
 Serial.println(String(numNewMessages));

 for (int i=0; i<numNewMessages; i++) {
  // Chat id of the requester
  if (bot.messages[i].type ==  F("callback_query")) {
   String chat_id = String(bot.messages[i].chat_id);
   String text = bot.messages[i].text;
   Serial.print("Call back button pressed with text: ");
   Serial.println(text);

   if (text == F("L1_ON")) {
    Lampu_1 = HIGH;
    digitalWrite(R4, HIGH);
    String keyboardJson = F("[[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L1_OFF\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 1 is ON", "", keyboardJson);
   } else if (text == F("L1_OFF")) {
    Lampu_1 = LOW;
    digitalWrite(R4, LOW);
    String keyboardJson = F("[[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L1_ON\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 1 is OFF", "", keyboardJson);
   } else if (text == F("L2_ON")) {
    Lampu_2 = HIGH;
```

```
    digitalWrite(R5, HIGH);
    String keyboardJson = F("[[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L2_OFF\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 2 is ON", "", keyboardJson);
   } else if (text == F("L2_OFF")) {
    Lampu_2 = LOW;
    digitalWrite(R5, LOW);
    String keyboardJson = F("[[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L2_ON\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 2 is OFF", "", keyboardJson);
   } else if (text == F("L3_ON")) {
    Lampu_3 = HIGH;
    digitalWrite(R6, HIGH);
    String keyboardJson = F("[[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L3_OFF\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 3 is ON", "", keyboardJson);
   } else if (text == F("L3_OFF")) {
    Lampu_3 = LOW;
    digitalWrite(R6, LOW);
    String keyboardJson = F("[[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L3_ON\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 3 is OFF", "", keyboardJson);
   } else if (text == F("AC1_ON")) {
    AC_1 = HIGH;
    digitalWrite(R3, HIGH);
    String keyboardJson = F("[[{ \"text\" : \"Matikan AC\", \"callback_data\" :
\"AC1_OFF\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "AC 1 Ruangan 1 is ON", "",
keyboardJson);
   } else if (text == F("AC1_OFF")) {
    AC_1 = LOW;
    digitalWrite(R3, LOW);
    String keyboardJson = F("[[{ \"text\" : \"Nyalakan AC\", \"callback_data\" :
\"AC1_ON\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "AC 1 Ruangan 1 is OFF", "",
keyboardJson);
   } else if (text == F("AC2_ON")) {
    AC_2 = HIGH;
    digitalWrite(R2, HIGH);
    String keyboardJson = F("[[{ \"text\" : \"Matikan AC\", \"callback_data\" :
\"AC2_OFF\" }]]");
    bot.sendMessageWithInlineKeyboard(chat_id, "AC 2 Ruangan 1 is ON", "",
keyboardJson);
   } else if (text == F("AC2_OFF")) {
    AC_2 = LOW;
    digitalWrite(R2, LOW);
```

```
      String keyboardJson = F("[[{ \"text\" : \"Nyalakan AC\", \"callback_data\" :
\"AC2_ON\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "AC 1 Ruangan 1 is OFF", "",
keyboardJson);
    } else if (text == F("AC3_ON")) {
      AC_3 = HIGH;
      digitalWrite(R1, HIGH);
      String keyboardJson = F("[[{ \"text\" : \"Matikan AC\", \"callback_data\" :
\"AC3_OFF\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "AC Ruangan 2 is ON", "",
keyboardJson);
    } else if (text == F("AC3_OFF")) {
      AC_3 = LOW;
      digitalWrite(R1, LOW);
      String keyboardJson = F("[[{ \"text\" : \"Nyalakan AC\", \"callback_data\" :
\"AC3_ON\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "AC Ruangan 2 is OFF", "",
keyboardJson);
    } else if (text == F("OTOMATIS_R1_ON")) {
      Ruangan_1 = HIGH;
      String keyboardJson = F("[[{ \"text\" : \"Matikan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R1_OFF\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "Kendali Otomatis Ruangan 1 is
Active", "", keyboardJson);
    } else if (text == F("OTOMATIS_R1_OFF")) {
      Ruangan_1 = LOW;
      String keyboardJson = F("[[{ \"text\" : \"Nyalakan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R1_ON\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "Kendali Otomatis Ruangan 1 is
Nonactive", "", keyboardJson);
    } else if (text == F("OTOMATIS_R2_ON")) {
      Ruangan_2 = HIGH;
      String keyboardJson = F("[[{ \"text\" : \"Matikan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R2_OFF\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "Kendali Otomatis Ruangan 2 is
Active", "", keyboardJson);
    } else if (text == F("OTOMATIS_R2_OFF")) {
      Ruangan_2 = LOW;
      String keyboardJson = F("[[{ \"text\" : \"Nyalakan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R2_ON\" }]]");
      bot.sendMessageWithInlineKeyboard(chat_id, "Kendali Otomatis Ruangan 2 is
Nonactive", "", keyboardJson);
    }
   }else{
    String chat_id = String(bot.messages[i].chat_id);
//     if (chat_id != CHAT_ID){
//       bot.sendMessage(chat_id, "Unauthorized user", "");
```

```
//       continue;
//     }

      // Print the received message
      String text = bot.messages[i].text;
      Serial.println(text);

      String from_name = bot.messages[i].from_name;

      if (text == "/LAMPU1") {
        if(Lampu_1 == LOW){
          String keyboardJson = F("[[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L1_ON\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 1 Mati\nNyalakan
Lampu?", "", keyboardJson);
        }else{
          String keyboardJson = F("[[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L1_OFF\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 1 Nyala \nMatikan
Lampu?", "", keyboardJson);
        }
      }
      if (text == "/LAMPU2") {
        if(Lampu_2 == LOW){
          String keyboardJson = F("[[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L2_ON\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 2 Mati \nNyalakan
Lampu?", "", keyboardJson);
        }else{
          String keyboardJson = F("[[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L2_OFF\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 2 Nyala \nMatikan
Lampu?", "", keyboardJson);
        }
      }
      if (text == "/LAMPU3") {
        if(Lampu_3 == LOW){
          String keyboardJson = F("[[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L3_ON\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 3 Mati\nNyalakan
Lampu?", "", keyboardJson);
        }else{
          String keyboardJson = F("[[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L3_OFF\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "Lampu 3 Nyala \nMatikan
Lampu?", "", keyboardJson);
        }
```

```
    }

    if (text == "RUANGAN1") {
      if(Ruangan_1 == LOW){
        String ket = "-- **RUANGAN 1** --\n\nSuhu Ruangan : " + String(t1) +
"°C\n\nKendali Otomatis Nonaktif\nNyalakan Kendali Otomatis?";
        String keyboardJson = F("[[{ \"text\" : \"Nyalakan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R1_ON\" }],");
        if(AC_1 == LOW && AC_2 == LOW){
          keyboardJson += F("[{ \"text\" : \"Nyalakan AC 1\", \"callback_data\" :
\"AC1_ON\" }, { \"text\" : \"Nyalakan AC 2\", \"callback_data\" : \"AC2_ON\" }],");
        }else if(AC_1 == LOW && AC_2 == HIGH){
          keyboardJson += F("[{ \"text\" : \"Nyalakan AC 1\", \"callback_data\" :
\"AC1_ON\" }, { \"text\" : \"Matikan AC 2\", \"callback_data\" : \"AC2_OFF\" }],");
        }else if(AC_1 == HIGH && AC_2 == LOW){
          keyboardJson += F("[{ \"text\" : \"Matikan AC 1\", \"callback_data\" :
\"AC1_OFF\" }, { \"text\" : \"Nyalakan AC 2\", \"callback_data\" : \"AC2_ON\" }],");
        }else {
          keyboardJson += F("[{ \"text\" : \"Matikan AC 1\", \"callback_data\" :
\"AC1_OFF\" }, { \"text\" : \"Matikan AC 2\", \"callback_data\" : \"AC2_OFF\" }],");
        }
        if(Lampu_1 == LOW && Lampu_2 == LOW){
          keyboardJson += F("[{ \"text\" : \"Nyalakan Lampu 1\", \"callback_data\" :
\"L1_ON\" }, { \"text\" : \"Nyalakan Lampu 2\", \"callback_data\" : \"L2_ON\" }]]");
        }else if(Lampu_1 == LOW && Lampu_2 == HIGH){
          keyboardJson += F("[{ \"text\" : \"Nyalakan Lampu 1\", \"callback_data\" :
\"L1_ON\" }, { \"text\" : \"Matikan Lampu 2\", \"callback_data\" : \"L2_OFF\" }]]");
        }else if(Lampu_1 == HIGH && Lampu_2 == LOW){
          keyboardJson += F("[{ \"text\" : \"Matikan Lampu 1\", \"callback_data\" :
\"L1_OFF\" }, { \"text\" : \"Nyalakan Lampu 2\", \"callback_data\" : \"L2_ON\" }]]");
        }else {
          keyboardJson += F("[{ \"text\" : \"Matikan Lampu 1\", \"callback_data\" :
\"L1_OFF\" }, { \"text\" : \"Matikan Lampu 2\", \"callback_data\" : \"L2_OFF\" }]]");
        }
        bot.sendMessageWithInlineKeyboard(chat_id, ket, "", keyboardJson);
      }else{
        String keyboardJson = F("[[{ \"text\" : \"Matikan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R1_OFF\" }],");
        if(Lampu_1 == LOW && Lampu_2 == LOW){
          keyboardJson += F("[{ \"text\" : \"Nyalakan Lampu 1\", \"callback_data\" :
\"L1_ON\" }, { \"text\" : \"Nyalakan Lampu 2\", \"callback_data\" : \"L2_ON\" }]]");
        }else if(Lampu_1 == LOW && Lampu_2 == HIGH){
          keyboardJson += F("[{ \"text\" : \"Nyalakan Lampu 1\", \"callback_data\" :
\"L1_ON\" }, { \"text\" : \"Matikan Lampu 2\", \"callback_data\" : \"L2_OFF\" }]]");
        }else if(Lampu_1 == HIGH && Lampu_2 == LOW){
          keyboardJson += F("[{ \"text\" : \"Matikan Lampu 1\", \"callback_data\" :
\"L1_OFF\" }, { \"text\" : \"Nyalakan Lampu 2\", \"callback_data\" : \"L2_ON\" }]]");
```

```
          }else {
            keyboardJson += F("[{ \"text\" : \"Matikan Lampu 1\", \"callback_data\" :
\"L1_OFF\" }, { \"text\" : \"Matikan Lampu 2\", \"callback_data\" : \"L2_OFF\" }]]");
          }
          bot.sendMessageWithInlineKeyboard(chat_id, "-- **RUANGAN 1** --\n\nSuhu
Ruangan : " + String(t1) + "°C\n\nKendali Otomatis Aktif\nMatikan Kendali Otomatis?",
"", keyboardJson);
        }
      }

      if (text == "RUANGAN2") {
        if(Ruangan_2 == LOW){
          String ket = "-- **RUANGAN 2** --\n\nSuhu Ruangan : " + String(t2) +
"°C\n\nKendali Otomatis Nonaktif\nNyalakan Kendali Otomatis?";
          String keyboardJson = F("[[{ \"text\" : \"Nyalakan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R2_ON\" }],");
          if(AC_3 == LOW){
            keyboardJson += F("[{ \"text\" : \"Nyalakan AC\", \"callback_data\" : \"AC3_ON\"
}],");
          }else {
            keyboardJson += F("[{ \"text\" : \"Matikan AC\", \"callback_data\" : \"AC3_OFF\"
}],");
          }
          if(Lampu_3 == LOW){
            keyboardJson += F("[{ \"text\" : \"Nyalakan Lampu\", \"callback_data\" :
\"L3_ON\" }]]");
          }else {
            keyboardJson += F("[{ \"text\" : \"Matikan Lampu\", \"callback_data\" :
\"L3_OFF\" }]]");
          }
          bot.sendMessageWithInlineKeyboard(chat_id, ket, "", keyboardJson);
        }else{
          String keyboardJson = F("[[{ \"text\" : \"Matikan Kendali Otomatis\",
\"callback_data\" : \"OTOMATIS_R2_OFF\" }]]");
          bot.sendMessageWithInlineKeyboard(chat_id, "-- **RUANGAN 2** --\n\nSuhu
Ruangan : " + String(t2) + "°C\n\nKendali Otomatis Aktif\nMatikan Kendali Otomatis?",
"", keyboardJson);
        }
      }

      if (text == "/onALL") {
        digitalWrite(R1, HIGH);digitalWrite(R2, HIGH);
        digitalWrite(R3, HIGH);digitalWrite(R4, HIGH);
        digitalWrite(R5, HIGH);digitalWrite(R6, HIGH);
       bot.sendMessage(chat_id, "all relay on", "");
      }
```

```
    if (text == "/offALL") {
      digitalWrite(R1, LOW);digitalWrite(R2, LOW);
      digitalWrite(R3, LOW);digitalWrite(R4, LOW);
      digitalWrite(R5, LOW);digitalWrite(R6, LOW);
     bot.sendMessage(chat_id, "all relay on", "");
    }

    if (text == "/options") {
     String keyboardJson = "[[\"RUANGAN1\", \"RUANGAN2\"],";
//           keyboardJson += "[\"/LAMPU1\", \"/LAMPU2\",\"/LAMPU3\"],";
          keyboardJson += "[\"STATUS\"]]";
     bot.sendMessageWithReplyKeyboard(chat_id, "Choose from one of the following
options", "", keyboardJson, true);
    }

    if (text == "/start") {
     String welcome = "**Telegram Bot Pengendali Beban Ruangan Dosen Teknik Mesin
PNUP**\n\n**Selamat Datang**, " + from_name + ".\n\n";
     welcome += "/options : untuk menampilkan pilihan\n";
     bot.sendMessage(chat_id, welcome, "Markdown");
    }
    if (text == "STATUS") {
     String Status_Lampu_1;
     String Status_Lampu_2;
     String Status_Lampu_3;
     String Status_AC_1;
     String Status_AC_2;
     String Status_AC_3;

     if (Lampu_1 == HIGH) {
      Status_Lampu_1 = "Lampu 1 ON";
     } else {
      Status_Lampu_1 = "Lampu 1 OFF";
     }
     if (Lampu_2 == HIGH) {
      Status_Lampu_2 = "Lampu 2 ON";
     } else {
      Status_Lampu_2 = "Lampu 2 OFF";
     }
     if (Lampu_3 == HIGH) {
      Status_Lampu_3 = "Lampu ON";
     } else {
      Status_Lampu_3 = "Lampu OFF";
     }
     if (AC_1 == HIGH) {
      Status_AC_1 = "AC 1 ON";
     } else {
```

```
      Status_AC_1 = "AC 1 OFF";
    }
    if (AC_2 == HIGH) {
      Status_AC_2 = "AC 2 ON";
    } else {
      Status_AC_2 = "AC 2 OFF";
    }
    if (AC_3 == HIGH) {
      Status_AC_3 = "AC ON";
    } else {
      Status_AC_3 = "AC OFF";
    }
    String statuss = "**Keadaan Ruangan**\n\n__Ruangan 1__\nSuhu Ruangan : " +
String(t1) + "°C\n" + Status_Lampu_1 + "\n" + Status_Lampu_2 + "\n" + Status_AC_1 +
"\n" + Status_AC_2 + "\n\n__Ruangan 2__\nSuhu Ruangan : " + String(t2) + "°C\n" +
Status_Lampu_3 + "\n" + Status_AC_3;
    bot.sendMessage(chat_id, statuss, "Markdown");
   }
  }
 }
}

void setup() {
 Serial.begin(115200);

 #ifdef ESP8266
  configTime(0, 0, "pool.ntp.org");      // get UTC time via NTP
  client.setTrustAnchors(&cert); // Add root certificate for api.telegram.org
 #endif

 pinMode(R1, OUTPUT);
 pinMode(R2, OUTPUT);
 pinMode(R3, OUTPUT);
 pinMode(R4, OUTPUT);
 pinMode(R5, OUTPUT);
 pinMode(R6, OUTPUT);

 sensor_1.begin();
 sensor_2.begin();

 // Connect to Wi-Fi
 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);
 #ifdef ESP32
  client.setCACert(TELEGRAM_CERTIFICATE_ROOT); // Add root certificate for
api.telegram.org
 #endif
```

```
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  // Print ESP32 Local IP Address
  Serial.println(WiFi.localIP());
  bot.sendMessage(CHAT_ID, "Terhubung...", "Markdown");
}

void loop() {
  sensor_1.requestTemperatures();
  sensor_2.requestTemperatures();
  t1 = sensor_1.getTempCByIndex(0);
  t2 = sensor_2.getTempCByIndex(0);

  if ( millis() > lastTimeBotRan + botRequestDelay)  {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

    while(numNewMessages) {
      Serial.println("got response");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }
    lastTimeBotRan = millis();
  }

  if(Ruangan_1 == HIGH){
    ruanganIotomatis();
  }

  if(Ruangan_2 == HIGH){
    ruanganIIotomatis();
  }
}

void ruanganIotomatis(){
  if(t1<=22){
    digitalWrite(R3, LOW);
    AC_1 = LOW;
    digitalWrite(R2, LOW);
    AC_2 = LOW;
  }else{
    digitalWrite(R3, HIGH);
    AC_1 = HIGH;
    digitalWrite(R2, HIGH);
    AC_2 = HIGH;
  }
```

```
}
void ruanganIIotomatis(){
 if(t2<=22){
   digitalWrite(R1, LOW);
   AC_3 = LOW;
 }else{
   digitalWrite(R1, HIGH);
   AC_3 = HIGH;
 }
}
```

Lampiran 3 Proses Pembuatan Telegram Bot

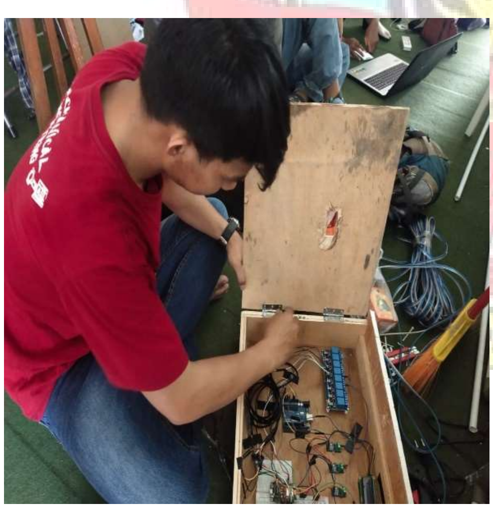
Proses Pembuatan Telegram Bot

1. Memastikan laptop atau *smartphone* sudah terinstal aplikasi telegram.
2. Login ke akun Telegram, lalu pada bagian search, ketikkan @manybot.
3. Kemudian akan tampil dialog chat dan menekan *START* untuk memulai *setup*.
4. Setelah menekan *START,* akan muncul *command* awal. Lalu menekan tulisan/addbot atau pada "*Create a New Bot*".
5. Kemudian akan muncul *tutorial* pembuatan bot secara detail
6. Pada informasi tersebut, disarankan membuat Bot menggunakan @BotFather, klik pada username @BotFather tersebut, kemudian akan masuk langsung ke dialog chat dengan BotFather tersebut. Lalu menekan *START* untuk memulai.
7. Setelah menekan *START*, nanti akan muncul tutorial penggunaan BotFather. Untuk memulai pembuatan Bot Telegram pilih pada link /*newbot*.
8. Kemudian akan ditanyakan username dari Bot yang akan digunakan dan syaratnya yaitu *username* harus berakhiran dengan kata Bot.
9. Setelah menentukan username, maka Bot Telegram telah selesai dibuat kemudian mencatat token yang tercantum untuk akses HTTP API-nya. Ini digunakan untuk menghubungkannya dengan NodeMCU ESP8266.
10. Selanjutnya memverifikasi token Bot tersebut. Salin token yang tercantum kemudian masuk kembali ke Manybot, kemudian menekan "*I've copied the API token*". Pastikan Manybot telah menjawab degan tulisan "*Good. Send the API token in the next massage".*

Lampiran 4 Foto Kegiatan

1. Proses Pengerjaan Instalasi dan Perakitan Komponen

Proses pemasangan instalasi



Proses perakitan rangkaian kontrol

Proses perakitan rangkaian daya

2. Proses Pengambilan Data

Senin, 2 Agustus 2021 pukul 09:00 WITA dan 16:00 WITA (tanpa sistem IoT)

Selasa, 3 Agustus 2021 pukul 09:00 WITA dan 16:00 WITA (tanpa sistem IoT)

Rabu, 4 Agustus 2021 pukul 09:00 WITA dan 16:00 WITA (tanpa sistem IoT)

Kamis, 5 Agustus 2021 pukul 09:00 WITA dan 16:00 WITA (tanpa sistem IoT)

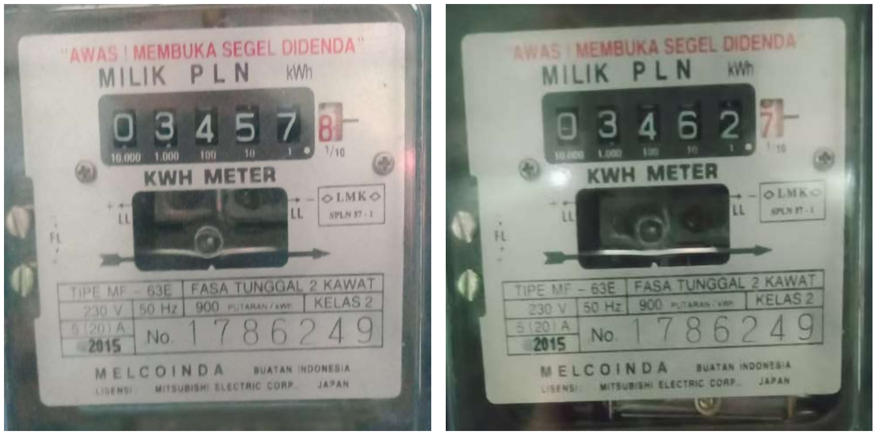
Jum'at, 6 Agustus 2021 pukul 09:00 WITA dan 16:00 WITA (tanpa sistem IoT)

Senin, 9 Agustus 2021 pukul 09:0 WITA dan 16:00 WITA (dengan sistem IoT)

Selasa, 10 Agustus 2021 pukul 09:0 WITA dan 16:00 WITA (dengan sistem IoT)

Rabu, 11 Agustus 2021 pukul 09:0 WITA dan 16:00 WITA (dengan sistem IoT)

Kamis, 12 Agustus 2021 pukul 09:0 WITA dan 16:00 WITA (dengan sistem IoT)



Jum'at, 13 Agustus 2021 pukul 09:0 WITA dan 16:00 WITA (dengan sistem IoT)

Proses pengambilan data pemakian energi listrik

Proses pengambilan data pada saat pengontrolan beban kelistrikan