TUGAS AKHIR – KI141502

SISTEM PENGAWASAN DAN PENGENDALIAN LAMPU PENERANGAN JALAN UMUM BERBASIS WEB

ANDARU KHARISMANDA
NRP 5113100138

Dosen Pembimbing I
Ir. Muchammad Husni, M. Kom.

Dosen Pembimbing II
Ridho Rahman Hariadi, S. Kom., M. Sc.

Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[halaman ini sengaja dikosongkan]

TUGAS AKHIR – KI141502

SISTEM PENGAWASAN DAN PENGENDALIAN LAMPU PENERANGAN JALAN UMUM BERBASIS WEB

ANDARU KHARISMANDA
NRP 5113100138

Dosen Pembimbing I
Ir. Muchammad Husni, M. Kom.

Dosen Pembimbing II
Ridho Rahman Hariadi, S. Kom., M. Sc.

Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[halaman ini sengaja dikosongkan]

TUGAS AKHIR – KI141502

WEB BASE ROAD LIGHTING CONTROLING AND MONITORING SYSTEM

ANDARU KHARISMANDA
NRP 5113100138

First Advisor
Ir. Muchammad Husni, M. Kom.

Second Advisor
Ridho Rahman Hariadi, S. Kom., M. Sc.

Department of Informatics
Faculty of Information Technology and Communication
Sepuluh Nopember Institute of Technology
Surabaya 2019

[halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

SISTEM PENGAWASAN DAN PENGENDALIAN LAMPU PENERANGAN JALAN UMUM BERBASIS WEB

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Analisis Jaringan Komputer
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
Andaru Kharismanda
NRP:5113100138

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Ir. Muchammad Husni, M. Kom. .....
NIP: 195701011983031004 (pembimbing 1)

Ridho Rahman Hariadi, S. Kom., M. Sc. .....
NIP: 198705112012121003 (pembimbing 2)

SURABAYA
JANUARI 2019

[halaman ini sengaja dikosongkan]

SISTEM PENGAWASAN DAN PENGENDALIAN LAMPU PENERANGAN JALAN UMUM BERBASIS WEB

Nama : Andaru Kharismanda
NRP : 5113100138
Jurusan : Informatika FTIK-ITS
Dosen Pembimbing 1 : Ir. Muchammad Husni, M. Kom.
Dosen Pembimbing 2 : Ridho Rahman Hariadi, S. Kom., M. Sc.

ABSTRAK

Lampu PJU merupakan sarana penting bagi pengguna jalan saat kondisi gelap atau malam. Saat siang Lampu PJU dimatikan supaya terjadi efisiensi. Saat ini teknologi PJU pintar tidak menggunakan intervensi tangan manusia dari balik layar kontrol apabila terjadi kerusakan sensor. Jika hanya menggunakan tangan manusia maka dibutuhkan banyak staff pengawasan dan pengendalian mengontrol lampu PJU dan tidak terjadi efisiensi. Diharapkan dalam pengembangan sistem PJU pintar menurunkan biaya tetapi akurasi terjamin.

Pada Tugas Akhir ini akan dilakukan implementasi Sistem Pengawasan dan Pengendalian lampu Penerangan Jalan Umum. Sistem pengawasan dilakukan dengan menampilkan status lampu, status saklar dan status pemakaian listrik di halaman web. Pengecekan status pemakaian listrik dan status lampu dilakukan menggunakan sensor ACS712. Sistem pengendalian dilakukan dengan intervensi mikrokontroler dari sensor cahaya dan timer atau staff monitoring dan conctrolling dengan menyalakan atau mematikan lampu.

Hasil dari implementasi ini adalah sistem pengendalian berjalan sangat baik.

Kata kunci: Lampu PJU, Sensor Cahaya, Web, ACS712, Efisiensi.

[halaman ini sengaja dikosongkan]

WEB BASED ROAD LIGHTING MONITORING AND CONCTROLLING SYSTEM

Student Name : Andaru Kharismanda
Student : 5113100138
Major : Informatika FTIK-ITS
Advisor 1 : Ir. Muchammad Husni, M. Kom.
Advisor 2 : Ridho Rahman Hariadi, S. Kom., M. Sc.

ABSTRACT

Public Road Lighting are vital instruments for road user when dark or night condition. At noon the Public Road Lighting light are turned off for efficiency. Today, smart public road lighting technology does not use human intervention from control room when sensor is damaged. if using only human intevene, then requires many monitoring and controlling staff to control the road lighting and there are no efficiency. The development of smart public road lighting system goal is to reduce costs but accuracy is assured.

In this the Final Project will be implementating the monitoring and controlling system of Public Road Lighting System. The monitoring system, which displays lights status, electricity usage and switch chosen status on the web page, all from ACS712 except switch status. The control system turn on or turn off the lights by intervene by microcontroller from the light sensor and timer or by monitoring and controlling staff.

The result of this implementation is only a control system that works very well.

Keywords: Public Road Lightings, Light Sensors, Web, ACS712, Efficiency.

[halaman ini sengaja dikosongkan]

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Segala puji bagi Allah SWT atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul : "**Sistem Pengawasan dan Pengendalian Lampu PJU berbasis web**" Harapan dari penulis semoga apa yang tertulis di dalam buku Tugas Akhir ini dapat bermanfaat bagi pengembangan ilmu pengetahuan saat ini, serta dapat memberikan kontribusi yang nyata. Dalam pelaksanaan dan pembuatan Tugas Akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak, tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih kepada:

1. Allah SWT
2. Orang Tua si penulis yang telah mendukung penyelesaian tugas akhir dan memberikan dukungan baik dengan semangat maupun finansial
3. Selaku dosen pembimbing yang telah membantu, menyemangati, dan membimbing penulis dengan ilmu-ilmu dalam pengerjaan tugas akhir
4. Dosen, Staff, dan Karyawan Jurusan Informatika yang telah memberikan ilmu bermanfaat, bimbingan yang tidak ternilai jasanya
5. Keluarga Muslim Informatika 2013-2014

Penulis telah berusaha dalam menyusun tugas akhir ini, namun penulis menyampaikan permohonan maaf atas kesalahan baik sengaja maupun tidak sengaja dan kekurangan selama proses tugas akhir ini. Kritik dan Saran yang membangun penulis dijadikan bahan evaluasi ke depan.

Surabaya, Januari 2019

Penulis

Andaru Kharismanda

DAFTAR ISI

DAFTAR GAMBAR, GRAFIK, DIAGRAM

DAFTAR TABEL

DAFTAR PSEUDOCODE

[halaman ini sengaja dikosongkan

BAB I
PENDAHULUAN

1.1. Latar Belakang

PJU (Penerangan Jalan Umum) merupakan sarana pendukung yang sangat penting bagi pengguna jalan terutama di malam hari atau wilayah dengan visibilitas rendah. Sebagai fasilitas publik, PJU dapat dinikmati oleh siapapun tanpa terkecuali. Berkaitan dengan hal tersebut, sudah menjadi kewajiban pelayanan bagi masyarakat maka pengelolaan PJU dalam hal ini pemeliharaan yang dilakukan oleh Pemerintah Daerah haruslah optimal sehingga mampu memberikan fasilitas bagi masyarakat secara memadai.

Pada saat ini muncul teknologi paling mutakhir untuk urusan PJU, namanya *smart system.* Teknologi ini berkembang tahun 2014. Di Indonesia lebih dikenal dengan nama PJU Pintar. Kelebihannya adalah tenaga dan dana untuk kunjungan peninjauan operasional PJU lebih sedikit. Saat ini Teknologi PJU Pintar hanya sebatas memegang kendali untuk menghemat pemakaian daya listrik yaitu dengan meredupkan cahaya lampu saat kondisi jalan sedang sepi (pukul 23.00 – 04.30 WIB).

Dalam Tugas Akhir ini diusulkan penambahan sensor cuaca dan timer pada sistem pengendalian dan pengawasan pju terpusat untuk lampu PJU. Pada alat timer memiliki kelebihan pada morfologis bumi dimana pada kondisi alam sekitar. kelemahan sensor ini adalah apabila terjadi perubahan cuaca menuju visibilitas rendah mengakibatkan kondisi yang berbahaya pada pengendara yang melintas pada wilayah itu. Pada teknologi sensor cahaya memiliki kelebihan pada sisi pengubahan cuaca yang mengakibatkan visibilitas rendah.. Menggunakan kedua sensor diharapkan meningkatkan efisiensi pemakaian PJU. akurat dan presisi sesuai dengan kondisi alam lingkungan sekitar.

1.2. Rumusan Masalah

Berdasar uraian diatas dapat dirumuskan hal-hal sebagai berikut:

1. Bagaimana cara mengendalikan penggunaan lampu pju sesuai dengan kondisi alam saat itu?
2. Bagaimana cara mengetahui lama lampu pju menyala dan tenaga listrik yang dibutuhkan?

1.3. Batasan Masalah

Batasan masalah pada kasus ini dapat dijelaskan berikut:

1. Lampu pju berada pada lingkungan suhu yang tidak ekstrim
2. jarak antar lampu PJU sesuai panduan SNI

1.4. Tujuan Pembuatan Tugas Akhir

Tujuan pembuatan tugas akhir ini adalah sebagai berikut:

1. Mengimplementasikan teknologi sensor cahaya dan alat timer dalam sistem pengawasan dan pengendalian pju
2. Membandingkan kecepatan respons yang dibutuhkan antara yang otomatis berdasar masukan sensor dengan yang hanya dikendalikan manusia
3. Melakukan uji coba sistem yang sudah diimplementasikan

1.5. Manfaat Tugas Akhir

Manfaat Tugas Akhir ini adalah:

1. Dapat diterapkan pada berbagai macam kondisi alam lingkungan sekitar dengan akurat, presisi dan efisien.
2. Mencegah terjadinya kekurangan atau kelebihan penggunaan anggaran listrik sehingga hasilnya dapat dipertanggungjawaban, tidak ada pihak yang

mengalami kerugian atau meraup keuntungan yang berlebih.

3. Dapat mengendalikan lebih banyak lampu pju dalam waktu yang lebih singkat sehingga lebih sedikit Penggunaan Sumber Daya Manusia di sistem pengendalian.

1.6. Metodologi

1.6.1. Penyusunan proposal tugas akhir

Tahap pertama dalam proses pengerjaan tugas akhir ini adalah menyusun proposal tugas akhir. Pada proposal tugas akhir ini diajukan Sistem Pengawasan dan Pengendalian Lampu Penerangan Jalan Umum dengan Sensor Cuaca dan Timer.

1.6.2. Studi literatur

Pada tahap ini, akan dicari studi literatur yang relevan untuk dijadikan referensi dalam pengerjaan tugas akhir. Studi literatur dapat diambil dari buku, internet, ataupun materi dalam suatu mata kuliah yang berhubungan dengan metode yang akan digunakan.

1.6.3. Analisis dan desain perangkat lunak

Pengembangan program kali ini akan dilakukan dengan menggunakan arduino IDE. Hasil dari pengembangan untuk mengimplementasikan sensor cahaya dan timer pada pju pintar . Akan dipasang sensor acs712 dan LDR pada setiap lampu pju.

1.6.4. Implementasi perangkat lunak

Perangkat yang akan digunakan kali ini diperuntukkan untuk membuat Sistem Pengawasan dan Pengendalian Lampu Penerangan Jalan Umum dengan sensor cuaca dan alat timer. Oleh karena itu, diperlukan perangkat yang dapat digunakan untuk

mensimulasikan pju pintar yang mengimplementasikan teknologi sensor cahaya dan alat timer. Perangkat yang digunakan adalah arduino, esp8322, acs712, sensor cahaya, lampu led.

1.6.5. Pengujian dan evaluasi

Proses pengujian dilakukan dengan mencoba perangkat tersebut secara langsung. Pada tugas akhir ini, permasalahan dititikberatkan pada pengujian. Hal ini untuk mengetahui seberapa efisien, akurat dan presisi lampu PJU pintar itu menyala. Beberapa parameter yang akan digunakan dalam melakukan pengujian adalah total lama lampu PJU menyala dan kecepatan respons dalam menghadapi perubahan kondisi visibilitas.

1.6.6. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi pju pintar yang mengimplementasikan teknologi sensor cahaya dan alat timer. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

Pendahuluan

a. Latar Belakang
b. Rumusan Masalah
c. Batasan Tugas Akhir
d. Tujuan
e. Metodologi
f. Sistematika Penulisan

Tinjauan Pustaka
Desain dan Implementasi
Pengujian dan Evaluasi
Kesimpulan dan Saran
Daftar Pustaka

1.7. Sistematika Penulisan

Buku Tugas akhir ini bertujuan untuk memberikan gambaran dari pengerjaan tugas akhir ini. Selain itu dimana dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir ini terdiri dari beberapa bagian seperti berikut:

1. Bab I Pendahuluan

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

2. Bab II Tinjauan Pustaka

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

3. Bab III Perancangan

Bab ini membahas tentang deskripsi umum sistem yang digunakan dan rancangan dari perangkat platform yang akan dibangun sesuai dengan yang telah teori teori yang telah dijelaskan pada bab tinjauan pustaka. Rancangan meliputi rancangan secara logik dan juga rancangan fisik.

4. Bab IV Implementasi

Bab ini membahas tentang implementasi dari perancangan pada bab sebelumnya. Bagaimana mengimplementasikan Sensor ACS712 dan sensor cahaya pada perangkat Arduino, NTP dan Web yang aakan digunakan.

5. Bab V Uji Coba dan Evaluasi

Bab ini membahas tentang pengujian dari sistem pengawasan dan pengendalian lampu PJU yang dibuat dengan metode yang dikembangkan. Melakukan evaluasi apakah telah memenuhi kebutuhan serta apakah telah memenuhi harapan.

6. Bab VI Kesimpulan dan Saran

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi selanjutnya.

7. Daftar Pustaka

Merupakan daftar referensi yang digunakan dalam pembuatan tugas akhir.

8. Lampiran

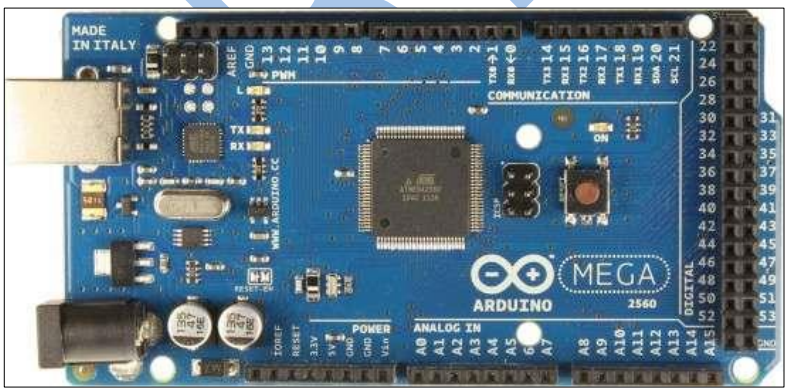
Merupakan bab tambahan yang berisi hasil uji coba yang telah dilakukan.

BAB II
TINJAUAN PUSTAKA

Bab ini menjelaskan tentang tinjauan pustaka yang menjadi dasar pembuatan Tugas Akhir. Beberapa teori, pustaka, dan teknologi yang mendasari pengerjaan Tugas Akhir ini. Penjelasan secara khusus masing-masing tinjauan pustaka dapat dilihat pada masing-masing subbab berikut ini.

2.1. Arduino Mega 2560

Arduino Mega 2560 adalah mikrokontroler Arduino yang menggunakan chip ATmega2560[1]. Arduino ini memiliki banyak pin 5v dan ground. Pin 5v dapat menghantarkan arus hingga 200 Ma yang digunakan untuk menyuplai sensor Light Dependent Resistor, ACS712 dan modul wifi ESP826 disertai voltage divider [2]. Berikut adalah gambar Arduino mega 2560 yang ditampilkan gambar 2.1. dan datasheet chip ATmega 2560.yang ditampilkan tabel 2.1.



Gambar 2.1 Arduino Mega 2560

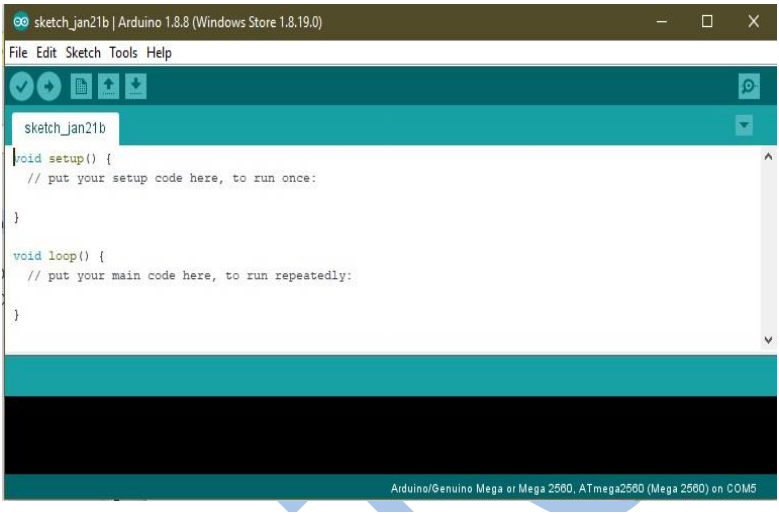
Table 2.1 Datasheet Atmega 2560

Microkontroller	ATmega2560
Tegangan Operasi	5V

Tegangan Input (yang Direkomendasikan)	7-12V
Tegangan Input (Batas)	6-20V
Pin Digital	54 (15 diantaranya PWM output)
Pin Analog Input	16
Arus DC per pin	20 Ma
Arus DC pin 3.3 v	50 Ma
Flash Memory	256 KB 8 KB digunakan bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

2.2. Arduino Integrated Development Environment (Arduino IDE)

Pemrograman Arduino dilakukan di dalam software Arduino IDE. Di dalam software ini terdapat fungsi yang diatur melalui sintaks pemrograman. Arduino menggunakan bahasa pemrograman seperti bahasa C dan dilengkapi library C/C++ Wiring. Program yang diisi di dalam software Arduino IDE disebut *sketch*. *Sketch* ditulis dalam editor teks dan disimpan dalam ekstensi .ino. Setelah sketch dikompile maka dilakukan proses uploading ke microkontroler Arduino. Pada Arduino IDE terdapat library manager untuk menginstal Library pihak ketiga [2]. Tambahan library digunakan untuk menunjang kebutuhan programmer agar bekerja maksimal. Arduino IDE yang digunakan adalah versi 1.8.8 dan tampilan utamanya dapat dilihat pada gambar 2.2.



Gambar 2.2 Tampilan Antarmuka Arduino IDE 1.8.8

2.3. Analog Digital Converter (ADC)

Analog Digital Converter adalah perangkat converter yang mengubah sinyal analog menjadi format yang dapat dibaca mesin atau biner atau digital [3].

2.3.1. ADC Control and Status Register A (ADCSRA)

Register ini bertanggung jawab untuk mengaktifkan ADC, memulai konversi ADC, *prescaler selection* dan kontrol interupsi [4]. Bit Register ADCSRA dapat dilihat pada tabel 2.2 dan tabel 2.1.

Tabel 2.1 Bit Register ACSRA

Bit	7	6	5	4	3	2	1	0

ADCSRA	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
Read	✓	✓	✓	✓	✓	✓	✓	✓
Write	✓	✓	✓	✓	✓	✓	✓	✓
Nilai Awal	0	0	0	0	0	0	0	0

2.3.1.1. ADC Enable (ADEN):

Bit ini digunakan untuk menyalakan ADC. Apabila bit bernilai 1 berarti ADC menyala.

2.3.1.2. ADC Start Conversion (ADSC):

Bit ini digunakan sebagai penanda konversi dari data analog ke data digital pada analog pin. Apabila data bernilai 1 maka proses konversi data sedang berjalan. Apabila data bernilai 0 maka proses konversi data berhenti. Secara default analog read menjalankan ADSC.

2.3.1.3. ADC Prescaler Select Bits (ADPS):

bit ini digunakan untuk mengatur frekuensi clock sirkuit. Terdiri dari ADPS2, ADPS1, ADPS0. Secara default faktor prescaler arduino bernilai 128.

2.3.2. ADC Multiplexer Selection Register (ADMUX)

Register ini digunakan untuk memilih sumber tegangan referensi, bagaimana hasil seharusnya disimpan (diadjust ke kiri atau diadjust ke kanan), saluran port analog yang akan digunakan untuk konversi [5]. Bit Register ADMUX dapat dilihat pada tabel 2.2.

Tabel 2.2 Bit Register ADMUX

Bit	7	6	5	4	3	2	1	0

ADMUX	REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0
Read	✓	✓	✓	✓	✓	✓	✓	✓
Write	✓	✓	✓	×	✓	✓	✓	✓
Nilai Awal	0	0	0	0	0	0	0	0

2.3.2.1. Reference Selection Bits (REFS):

Sumber tegangan referensi digunakan untuk mengubah bit Analog ke Digital oleh ADC. Dalam rangka memilih sumber tegangan referensi maka digunakanlah Bit REFS1 dan REFS0. Pada tugas akhir ini digunakan bit REFS sebagai pengukur voltase *Supply* Arduino dengan akurat. Tabel Reference Selection Bits dapat dilihat pada tabel 2.3.

Tabel 2.3 Reference Selection Bits

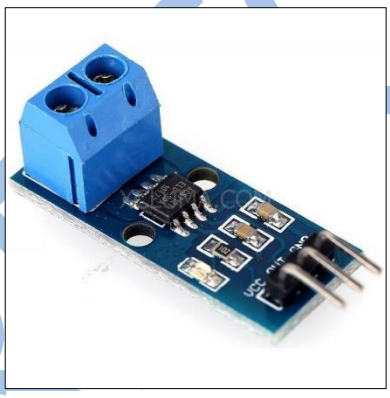
REFS1	REFS0	Vref Selection
0	0	AREF digunakan sebagai VRef dan VRef internal dimatikan
0	1	AVCC dengan kapasitor eksternal pada pin AREF digunakan sebagai VRef
1	0	Reserved bit
1	1	Tegangan referensi internal 2.56V digunakan dengan kapasitor eksternal pada pin AREF untuk VRef

2.3.2.2. Bits 3:0 – MUX 3:0: Bit Pilihan Saluran Analog

Pemilihan pin input analog menggunakan bit ini.

2.4. ACS712

ACS712 adalah sensor yang membaca arus AC atau arus DC. Sensor ACS712 menggunakan prinsip Efek Hall. Efek Hall ditemukan oleh Dr. Edwin Hall pada tahun 1879. Prinsipnya dimulai ketika arus listrik (I) mengalir melewati lempengan yang disebut elemen Hall. Penempatan elemen Hall tegak lurus dengan medan magnet (B) sehingga menimbulkan gaya Lorentz. Dikarenakan gaya Lorentz mengakibatkan pergerakan elektron berbelok menuju kedua sisi elemen Hall dan menjadi beda potensial. Beda potensial inilah yang disebut Tegangan Hall [7]. Modul ACS712 dapat dilihat pada gambar 2.3.



Gambar 2.3 ACS712

Rumus tegangan pada pin Out = 2,5 ± (0,185 x I) Volt, dimana I = arus yang terdeteksi dalam satuan Ampere [8].

2.5. Light Dependent Resistor (LDR)

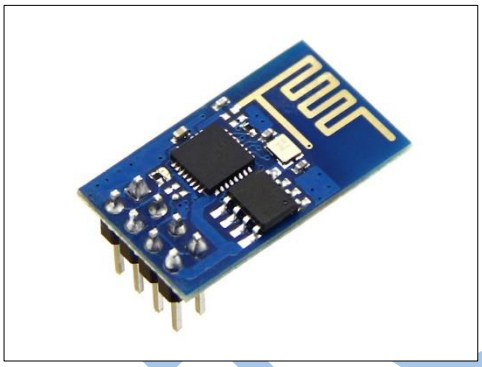
LDR adalah sensor cahaya yang resistansinya berbanding terbalik dengan intensitas cahaya. Jika intensitas cahaya pada sensor LDR berkurang maka resistansi LDR menjadi sangat tinggi. Begitu juga sebaliknya. Jika intensitas cahaya pada sensor LDR meningkat maka resistansi LDR menjadi sangat rendah. Sensor LDR ini tangguh di lingkungan keras dan memiliki sifat pasif dan tidak menghasilkan energi listrik apa pun [9]. LDR yang digunakan dalam Sistem Pengawasan dan Pengendalian PJU adalah LDR GL5539. LDR dapat dilihat pada gambar 2.4.

Gambar 2.4 Light Dependent Resistor

2.6. ESP8266

ESP8266 adalah sebuah modul WiFi pengembangan "Espressif". Modul WiFi ini sudah bersifat SoC (*System on Chip*), sehingga pemrograman dapat dilakukan secara langsung dan dapat menjalankan mode *access point adhoc* dan mode *client* secara simultan. ESP8266 dapat diprogram menggunakan AT command atau Arduino IDE yang telah tersedia library ESP8266 melalui

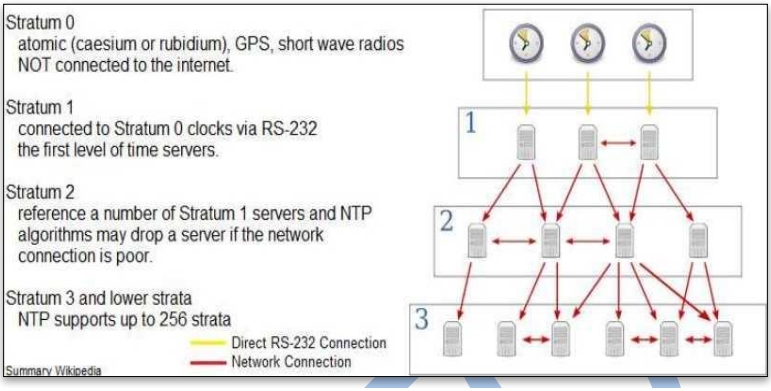
metode *download* dan *install* [10]. ESP8266 yang digunakan dalam Sistem Pengawasan dan Pengendalian PJU adalah Modul ESP8266-01. Modul ESP8266-01 dapat dilihat pada gambar 2.5.



Gambar 2.5 ESP8266-01

2.7. NTP Server

NTP adalah singkatan dari Network Time Protocol, Protokol transmisi jaringan perangkat yang terhubung dengan server Coordinated Universal Time (UTC) dalam rangka sinkronisasi waktu. Pengiriman paket NTP menggunakan port lokal UDP dan melalui klien jaringan lain [11]. Dalam tugas akhir ini NTP digunakan sebagai alat timer siang malam dan update data penggunaan listrik berdasar waktu pemakaian listrik. Cara kerja NTP dapat dilihat pada gambar 2.6.



Gambar 2.6 NTP

2.8. Relay

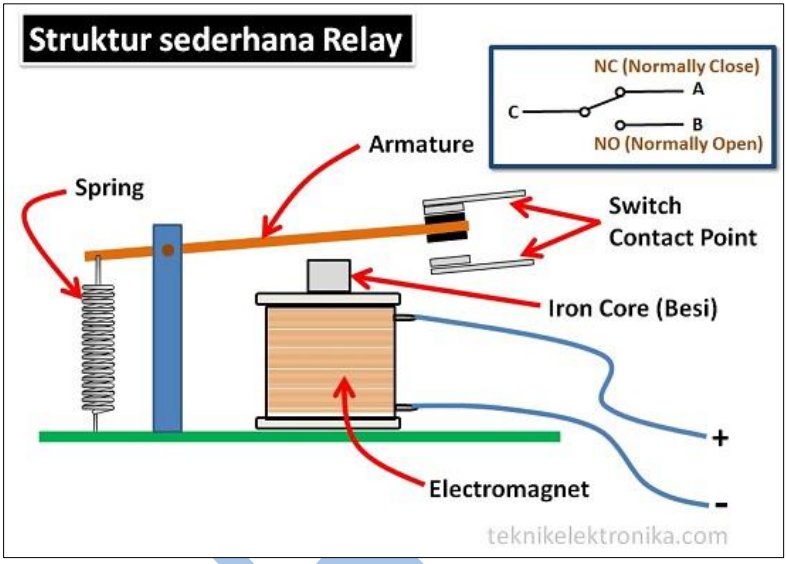
2.8.1. Pengertian Relay

Relay adalah Saklar (*Switch*) yang dipicu oleh listrik dan merupakan komponen elektromekanik yang terdiri dari 2 bagian utama yakni Elektromagnet (Coil) dan Mekanikal (seperangkat Kontak Saklar/Switch) [12].

2.8.2. Prinsip Kerja Relay

Besi (Iron Core) dikendalikan oleh kumparan yang melilit besi tersebut. Arus listrik memicu kumparan mengaktifkan gaya Elektromagnet. Kemudian posisi Armature berpindah dari posisi Normally Close (NC) menuju ke posisi Normally Open (NO) sehingga posisi NO dilalui arus listrik dan posisi NC menjadi OPEN atau tidak terhubung. Pada saat tidak teraliri arus listrik maka posisi armature kembali ke posisi semula [12].

Struktur relay dapat dijelaskan pada gambar 2.7.



Gambar 2.7 Struktur Relay

BAB III
DESAIN DAN PERANCANGAN

Pada bab ini dijelaskan mengenai rancangan sistem penerangan jalan umum. Perancangan yang dijelaskan meliputi spesifikasi jalan, kasus penggunaan arduino dan kasus penggunaan ESP8266 yang digunakan sistem yang akan dibangun.

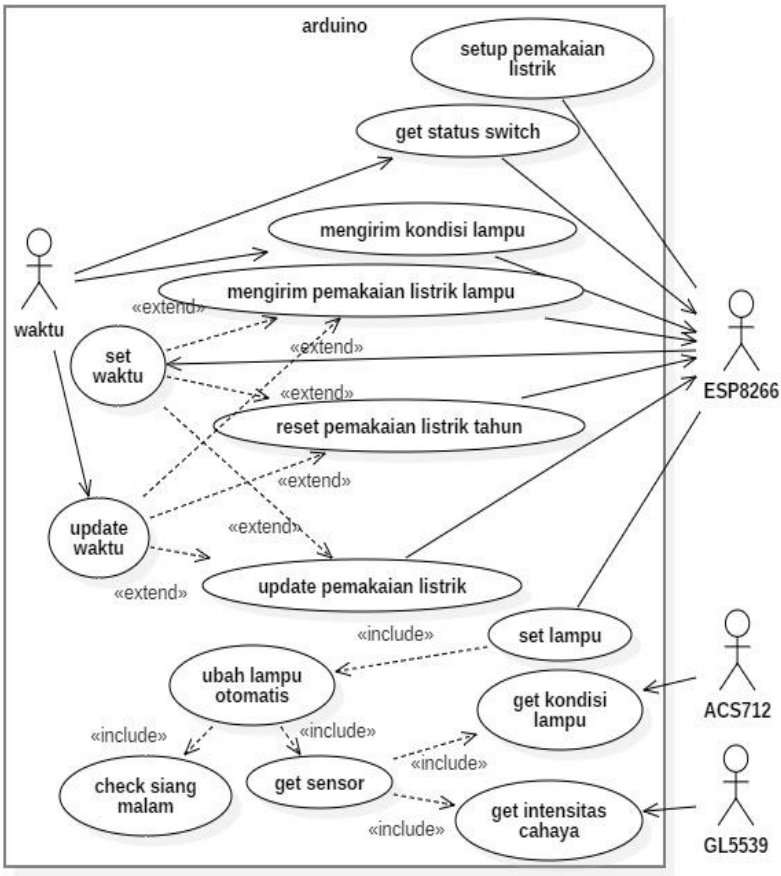
3.1. Deskripsi Umum

Miniatur jalan yang digunakan dalam tugas akhir ini adalah miniatur jalan arteri primer. Miniatur ini memiliki panjang 200 cm dan dilengkapi dengan pagar pembatas antar jalur dan 4 lampu Penerangan Jalan Umum. Ilustrasi jalan dapat dilihat pada gambar 3.1.

Gambar 3.1 Ilustrasi Jalan Raya

3.1.1. Diagram Kasus Penggunaan Arduino

Terdapat 4 aktor dalam diagram kasus penggunaan Arduino yang terdiri dari 2 sensor, 1 sistem ESP8266, dan 1 waktu yang digunakan proses otomatisasi. Diagram kasus penggunaan arduino dapat dilihat pada gambar 3.2.



Gambar 3.2 Diagram Kasus Penggunaan Arduino

3.1.2. Deskripsi Kasus Penggunaan Arduino

Deskripsi kasus penggunaan yang dibutuhkan pada Sistem Pengawasan dan Pengendalian PJU disesuaikan dengan diagram kasus penggunaan Arduino . Deskripsi kasus penggunaan arduino dapat dilihat pada tabel 3.1.

Tabel 3.1 Tabel Deskripsi Kasus Penggunaan Arduino

Kode Kasus Penggunaan	Nama Kasus Penggunaan
UC-001	Mengeset waktu
UC-002	Mengupdate waktu
UC-003	Mnegirim pemakaian listrik lampu
UC-004	Mengupdate pemakaian listrik
UC-005	Mereset pemakaian listrik tahun
UC-006	Mengirim setup pemakaian listrik
UC-007	Mengirim status switch
UC-008	Mengirim status lampu
UC-009	Mengeset lampu
UC-010	Mengubah lampu otomatis
UC-011	Mengecek siang malam
UC-012	Mendapatkan nilai sensor
UC-013	Mendapatkan intensitas cahaya
UC-014	Mendapatkan kondisi lampu

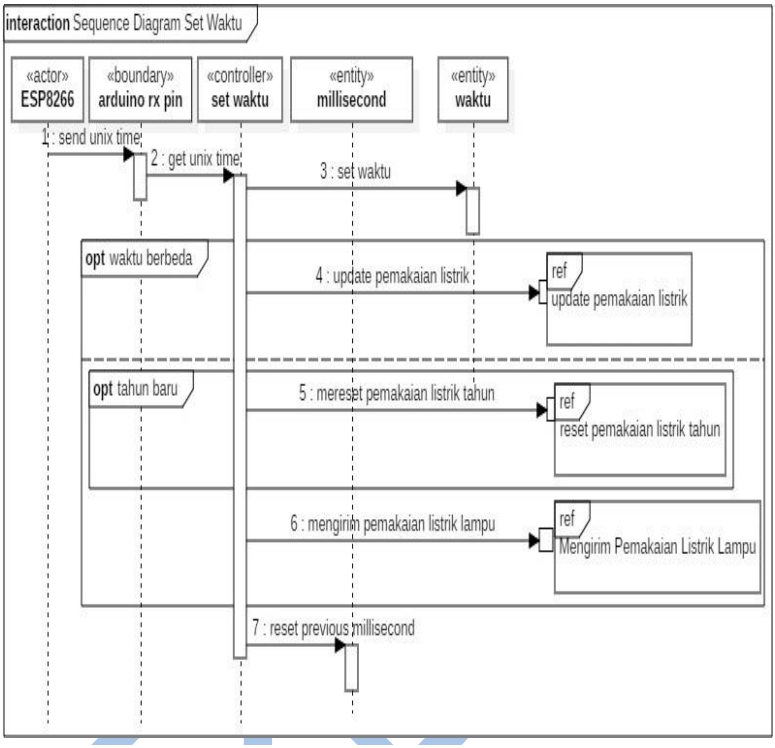
3.1.3. Use Case Mengeset Waktu

Pada kasus penggunaan ini ESP8266 dapat mengirim unix time ke Arduino dan menyimpan waktu. Tujuannya memastikan waktu, dan update data penggunaan listrik akurat. Skenario kasus penggunaan dapat dilihat pada tabel 3.2 dan diagram sequence pada gambar 3.3.

Tabel 3.2 Use Case Scenario Mengeset Waktu

Kode	UC-001
Nama	Mengeset waktu
Aktor	ESP8266

Kondisi Awal	ESP8266 mengirim Unix Time
Skenario Utama	
1. aktor mengirim unix time 2. Sistem mendapatkan unix time 3. Sistem mengeset waktu 4. Sistem mereset milidetik program	
Skenario Alternatif	
3a Jika detik berbeda 3a1 Extend use case mengirim pemakaian listrik lampu 3b Jika waktu selain detik berbeda 3b1 Jika tahun baru 3b1a Extend use case reset pemakaian listrik tahun 3b2 Extend use case update pemakaian listrik	
Kondisi Akhir	Arduino menyimpan data waktu



Gambar 3.3 Sequence Diagram Set Waktu

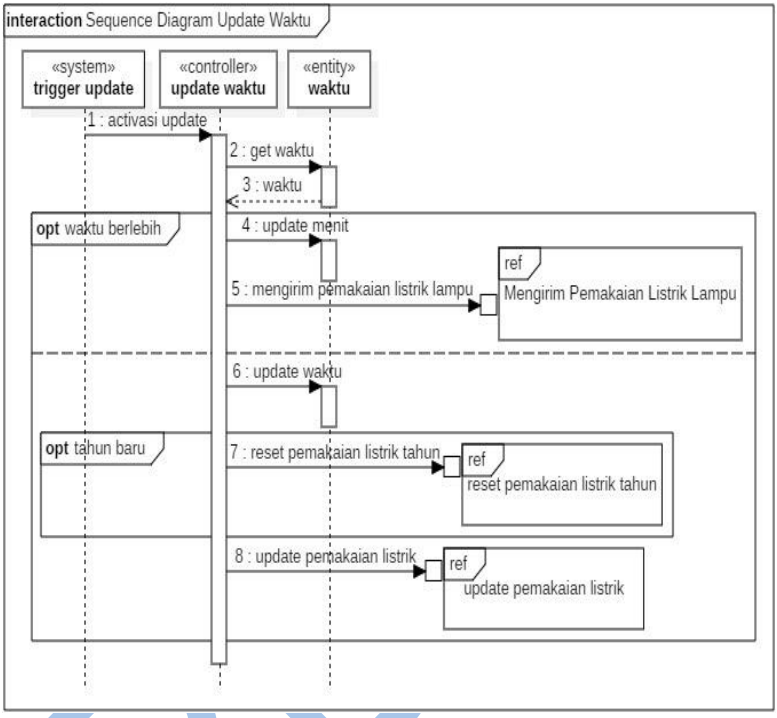
3.1.4. Use Case Update Waktu

Pada kasus penggunaan ini Arduino dapat mengupdate data waktu secara otomatis apabila tidak mendapat kiriman waktu NTP dalam interval waktu program Arduino dan waktu update program terakhir. Skenario kasus penggunaan dapat dilihat pada tabel 3.3 dan diagram sequence pada gambar 3.4.

Tabel 3.3 Use Case Scenario Mengupdate Waktu

Kode	UC-002
Tujuan	Mengupdate waktu

Aktor	Trigger Waktu
Kondisi Awal	Waktu belum diupdate
Skenario Utama	
1. Aktor melakukan trigger 2. Sistem mencari data waktu 3. Sistem mendapatkan data waktu 4. Sistem mengecek kelebihan waktu	
Skenario Alternatif	
4a Jika kelebihan detik 4a1 Sistem mengupdate menit 4a2 Extend use case mengirim pemakaian listrik lampu 4b Jika kelebihan selain detik 4b1 Sistem mengupdate waktu 4b2 Jika tahun baru 4b2a Extend use case mereset pemakaian listrik tahun 4b3 Extend use case mengupdate pemakaian listrik	
Kondisi Akhir	Waktu sudah diupdate



Gambar 3.4 Sequence Diagram Update Waktu

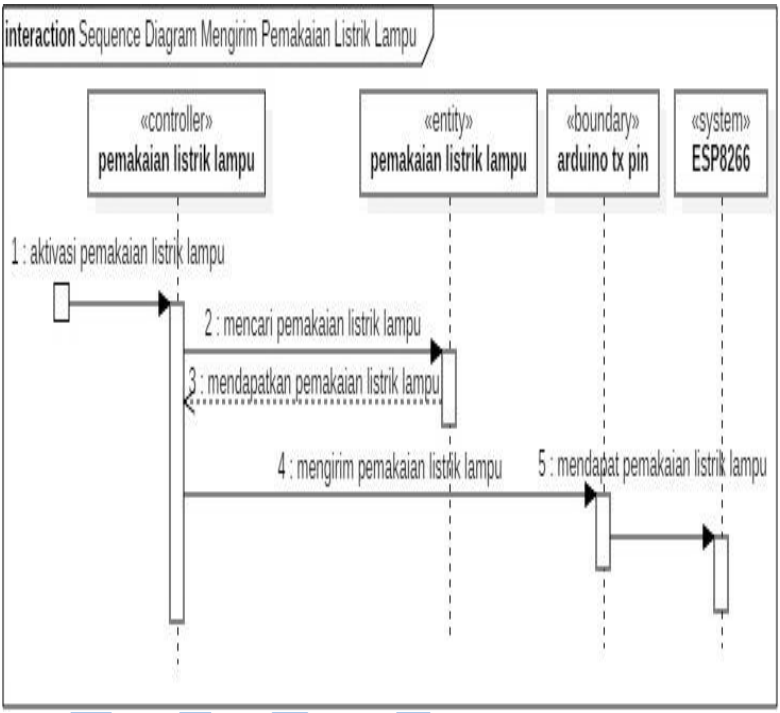
3.1.5. Use Case Mengirim Pemakaian Listrik Lampu

Pada kasus penggunaan ini Arduino dapat mengirim data pemakaian listrik lampu ke sistem ESP8266. Tujuan dari use case menampilkan hasil data pemakaian listrik lampu di halaman web. Skenario kasus penggunaan dapat dilihat pada tabel 3.4 dan diagram sequence pada gambar 3.5.

Tabel 3.4 Scenario Mengirim Pemakaian Listrik

Kode	UC-003

Tujuan	Mengirim pemakaian listrik
Aktor	Arduino
Kondisi Awal	Pemakaian listrik belum dikirim
Skenario Utama	
1. Sistem mendapatkan trigger 2. Sistem mencari pemakaian listrik lampu 3. Sistem mendapatkan pemakaian listrik lampu 4. Sistem mengirim pemakaian listrik lampu 5. ESP8266 menerima pamakaian listrik lampu	
Skenario Alternatif	
-	
Kondisi Akhir	Pemakaian listrik sudah dikirim



Gambar 3.5 Sequence Diagram Mengirim Pemakaian Listrik Lampu

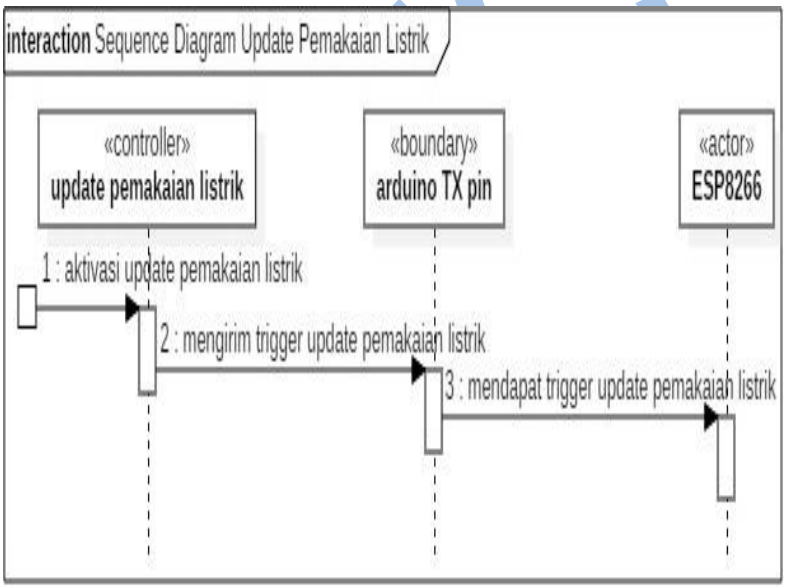
3.1.6. Use Case Mengupdate Pemakaian Listrik

Pada kasus penggunaan ini Arduino dapat mengupdate data pemakaian listrik dan mengirim data ke sistem ESP8266. Tujuan dari use case menampilkan hasil update data pemakaian listrik di halaman web. Skenario kasus penggunaan dapat dilihat pada tabel 3.5 dan diagram sequence pada gambar 3.6.

Tabel 3.5 Scenario Update Pemakaian Listrik

Kode	UC-004

Tujuan	Mengupdate pemakaian listrik
Aktor	Arduino
Kondisi Awal	Pemakaian listrik belum diupdate
Skenario Utama	
1. Sistem mendapatkan trigger update pemakaian listrik 2. Sistem mengirimkan trigger update pemakaian listrik 3. ESP8266 mendapatkan trigger update pemakaian listrik	
Skenario Alternatif	
-	
Kondisi Akhir	Pemakaian listrik sudah diupdate



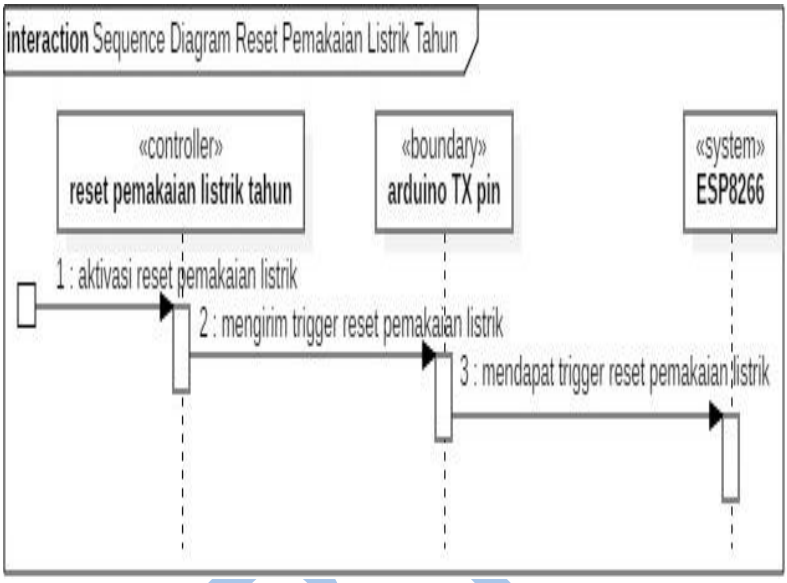
Gambar 3.6 Sequence Diagram Update Pemakaian Listrik

3.1.7. Use Case Reset Pemakaian Listrik Tahun

Pada kasus penggunaan ini Arduino dapat mereset data pemakaian listrik tahun dan mengirim data ke sistem ESP8266. Tujuan dari use case menampilkan hasil reset data pemakaian listrik tahun di halaman web. Skenario kasus penggunaan dapat dilihat pada tabel 3.6 dan diagram sequence pada gambar 3.7.

Tabel 3.6 Use Case Scenario Reset Pemakaian Listrik Tahun

Kode	UC-005
Tujuan	Mereset Data Pemakaian Listrik Tahun
Aktor	ESP8266
Kondisi Awal	Data Pemakaian Listrik Tahun belum direset
Skenario Utama	
1. Sistem mendapatkan trigger reset pemakaian listrik tahun 2. Sistem mengirimkan trigger reset pemakaian listrik tahun 3. Aktor mendapatkan trigger pemakaian listrik tahun	
Skenario Alternatif	
-	
Kondisi Akhir	Data Pemakaian Listrik Tahun sudah direset



Gambar 3.7 Sequence Diagram Reset Pemakaian Listrik Tahun

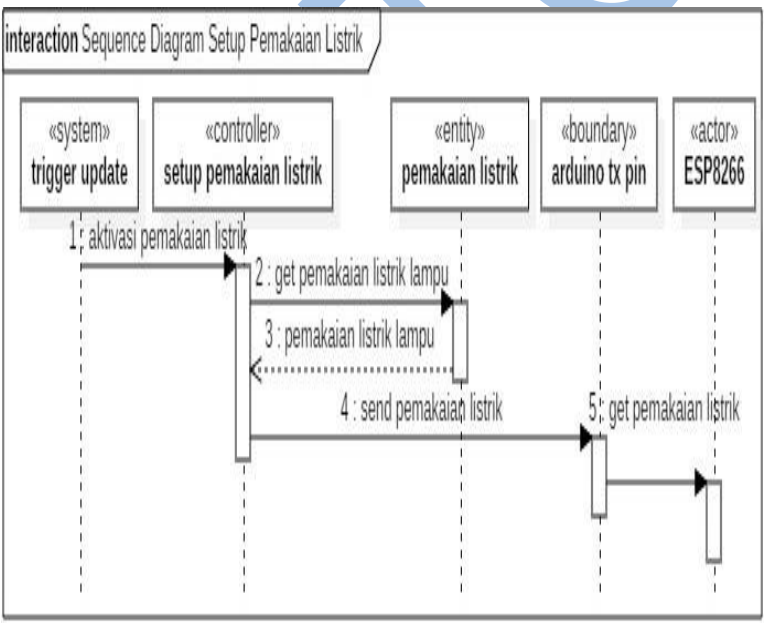
3.1.8. Use Case Get Setup Pemakaian Listrik Lampu

Pada kasus penggunaan ini ESP8266 dapat menerima data pemakaian listrik lampu setelah meminta data pemakaian listrik lampu k ke Arduino pada saat inisialisasi. Tujuan dari use case menampilkan data pemakaian listrik lampu saat server mulai berjalan. Skenario kasus penggunaan dapat dilihat pada tabel 3.7 dan diagram sequence pada gambar 3.8.

Tabel 3.7 Use Case Scenario Get Setup Pemakaian Listrik

Kode	UC-006
Tujuan	Mendapatkan setup pemakaian listrik lampu
Aktor	ESP8266

Kondisi Awal	ESP8266 belum mendapat setup pemakaian listrik lampu
Skenario Utama	
1. Aktor meminta setup pemakaian listrik 2. Sistem mencari pemakaian listrik lampu 3. Sistem mendapakan pemakaian listrik lampu 4. Sistem mengirimkan pemakaian listrik lampu 5. Aktor mendapat pemakaian listrik lampu	
Skenario Alternatif	
-	
Kondisi Akhir	ESP8266 sudah mendapat setup pemakaian listrik lampu



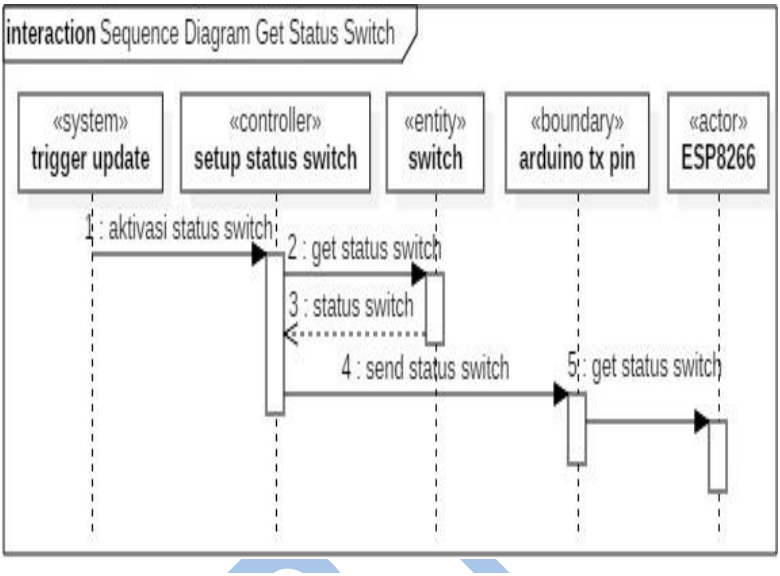
Gambar 3.8 Sequence Diagram Setup Pemakaian Listrik

3.1.9. Use Case Mengirim Status Switch

Pada kasus penggunaan ini ESP8266 dapat menerima status tombol setelah aktor mengirm status switch. Tujuan dari use case ini staff monitoring dan pengendalian dapat melihat status tombol melalui web. Skenario kasus penggunaan dapat dilihat pada tabel 3.8 dan diagram sequence pada gambar 3.9.

Tabel 3.8 Use Case Scenario Mengirim Status Switch

Kode	UC-007
Tujuan	Mendapatkan status saklar
Aktor	Waktu
Kondisi Awal	ESP8266 belum mendapat status saklar
Skenario Utama	
1. Aktor mentrigger sistem 2. Sistem mencari status saklar 3. Sistem mendapatkan status saklar 4. Sistem mengirimkan status saklar 5. Aktor menerima status saklar	
Skenario Alternatif	
-	
Kondisi Akhir	ESP8266 sudah mendapat status saklar



Gambar 3.9 Sequence Diagram Get Status Switch

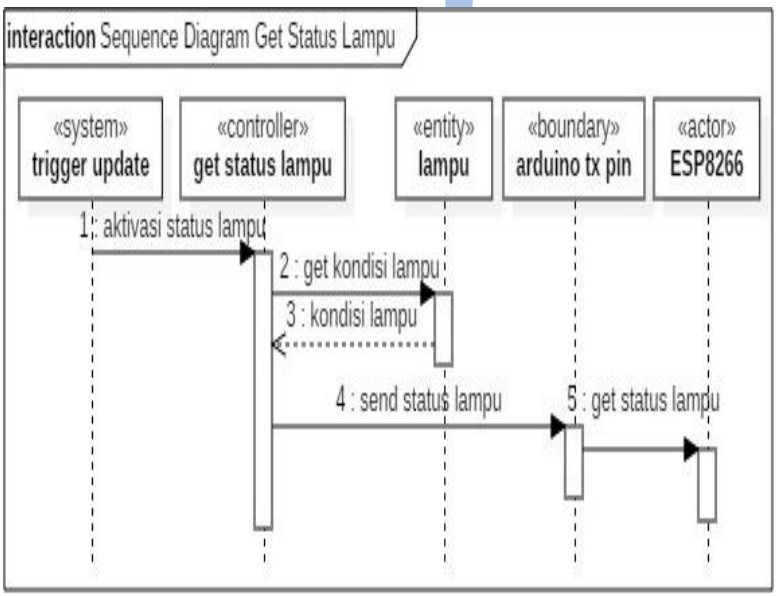
3.1.10. Use Case Mengirim Kondisi Lampu

Pada kasus penggunaan ini ESP8266 dapat menerima status lampu setelah aktor mengirim kondisi lampu . Tujuan dari use case ini staff monitoring dan pengendalian dapat melihat kondisi lampu melalui web. Skenario kasus penggunaan dapat dilihat pada tabel 3.9 dan diagram sequence pada gambar 3.10.

Tabel 3.9 Use Case Scenario Mengirim Kondisi Lampu

Kode	UC-008
Tujuan	Mendapatkan kondisi lampu
Aktor	Waktu
Kondisi Awal	Esp8266 belum mendapat kondisi lampu
Skenario Utama	

1. Aktor mentrigger sistem 2. Sistem mencari kondisi lampu 3. Sistem mendapatkan kondisi lampu 4. Sistem mengirimkan kondisi lampu 5. Aktor menerima kondisi lampu	
Skenario Alternatif	
-	
Kondisi Akhir	ESP8266 sudah mendapat kondisi lampu



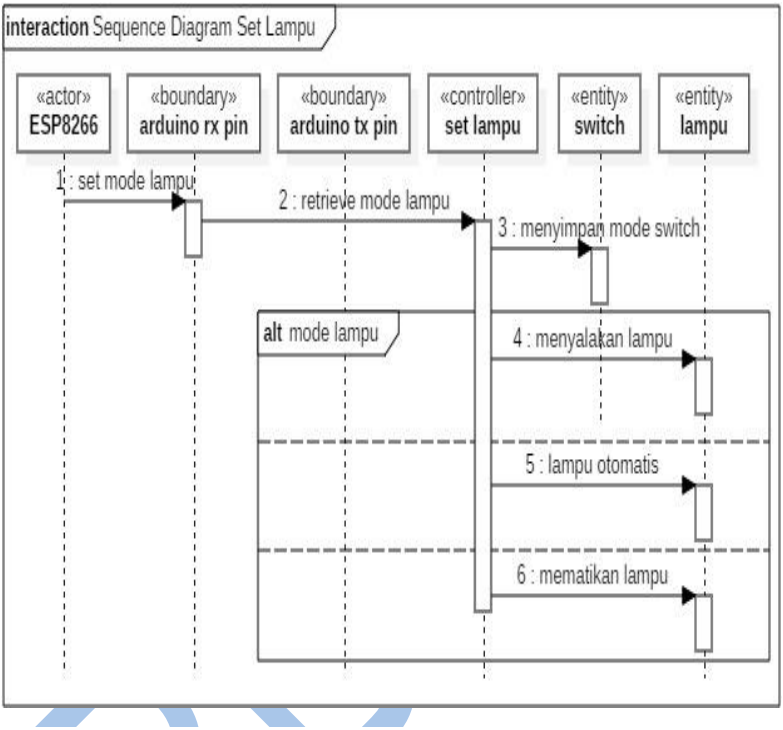
Gambar 3.10 Sequence Diagram Set Lampu

3.1.11. Use Case Set Lampu

Pada kasus penggunaan ini Arduino dapat mengubah mode lampu sesuai input yang diterima dari ESP8266. Tujuan dari use case adalah lampu dapat diubah sesuai instruksi yang diberikan. Skenario kasus penggunaan dapat dilihat pada tabel 3.10 dan diagram sequence pada gambar 3.11.

Tabel 3.10 Use Case Scenario Set Lampu

Kode	UC-009
Tujuan	Mendapatkan setup kondisi lampu
Aktor	ESP8266
Kondisi Awal	ESP8266 belum mendapat setup kondisi lampu
Skenario Utama	
1. Aktor mengeset lampu 2. Sistem mendapatkan mode lampu 3. Sistem menyimpan mode saklar 4. Sistem mengubah mode lampu	
Skenario Alternatif	
4 Jika menerima mode lampu mati 4a1 Sistem mematikan lampu 4b Jika menerima mode lampu otomatis 4b1 Sistem mengotomatisasi lampu 4c Jika menerima mode lampu menyala 4c1 Sistem menyalakan lampu	
Kondisi Akhir	ESP8266 sudah mendapat setup kondisi lampu



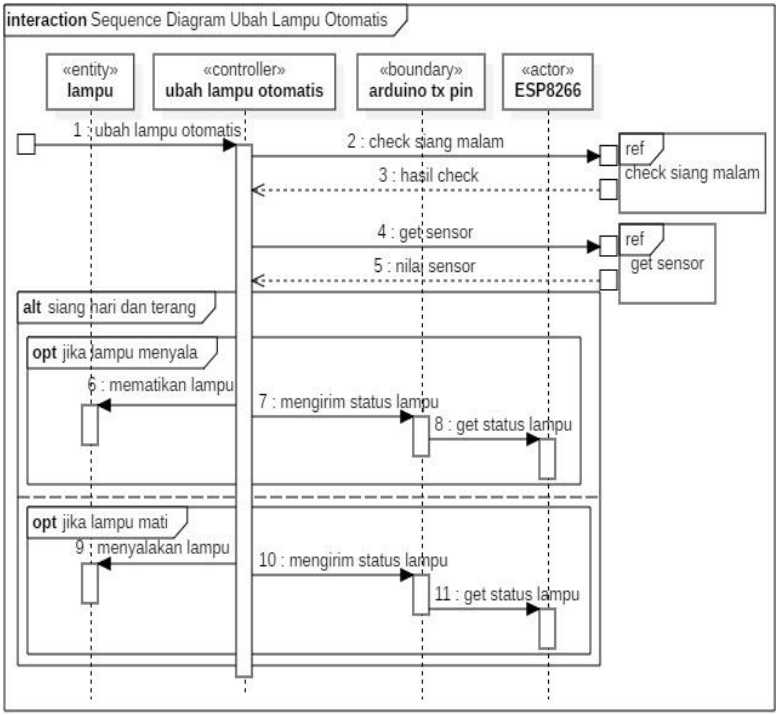
Gambar 3.11 Sequence Diagram Set Lampu

3.1.12. Use Case Ubah Lampu Otomatis

Pada kasus penggunaan ini Arduino dapat mengintervensi mode lampu secara otomatis tergantung gelap terang lingkungan sekitar dan kondisi siang malam. Tujuan dari Use Case adalah lampu PJU berubah secara otomatis sesuai kondisi sekitar. Skenario kasus penggunaan dapat dilihat pada tabel 3.11 dan diagram sequence pada gambar 3.12.

Tabel 3.11 Use Case Scenario Ubah Lampu Otomatis

Kode	UC-010
Tujuan	Mengeset lampu sesuai lingkungan sekitar
Aktor	Arduino
Kondisi Awal	Lampu belum sesuai kondisi lingkungan sekitar
Skenario Utama	
1. Sistem mendapatkan trigger 2. Include Cek Siang Malam 3. Sistem mendapatkan status siang malam 4. Include Get Sensor 5. Sistem mendapatkan nilai sensor 6. Sistem menganalisa kondisi sekitar	
Skenario Alternatif	
6a Jika lampu menyala dan kondisi sekitar siang dan terang 6a1 Sistem mematikan lampu 6a2 Sistem mengirim status lampu mati 6a3 Aktor mendapat status lampu mati 6b Jika lampu mati dan kondisi sekitar gelap atau malam 6b1 Sistem menyalakan lampu 6b2 Sistem mengirim status lampu menyala 6b3 Aktor mendapat status lampu menyala	
Kondisi Akhir	Lampu sudah sesuai kondisi lingkungan sekitar



Gambar 3.12 Sequence Diagram Ubah Lampu Otomatis

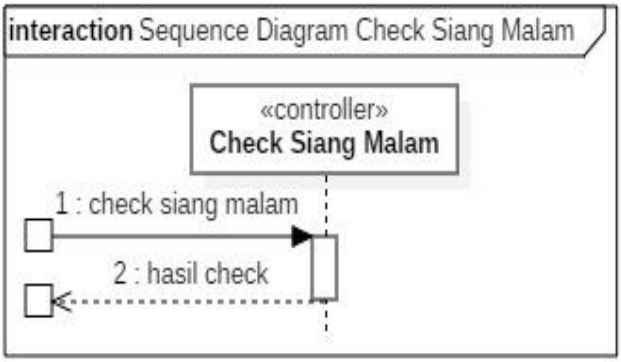
3.1.13. Use Case Cek Siang Malam

Pada kasus penggunaan ini Arduino dapat mengecek siang atau malam berdasarkan waktu yang diberikan oleh NTP. Tujuan dari use case adalah sistem PJU pintar dapat mengubah lampu sesuai siang dan malam saat disetel otomatis. Skenario kasus penggunaan dapat dilihat pada tabel 3.12 dan diagram sequence pada gambar 3.13.

Tabel 3.12 Use Case Scenario Cek Siang Malam

Kode	UC-011
Tujuan	Memastikan lampu sesuai siang malam
Aktor	Waktu

Kondisi Awal	Sistem belum mengetahui siang atau malam
Skenario Utama	
1. Sistem memeriksa situasi sedang siang atau malam 2. Sistem mengembalikan hasil siang atau malam	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mengetahui siang atau malam



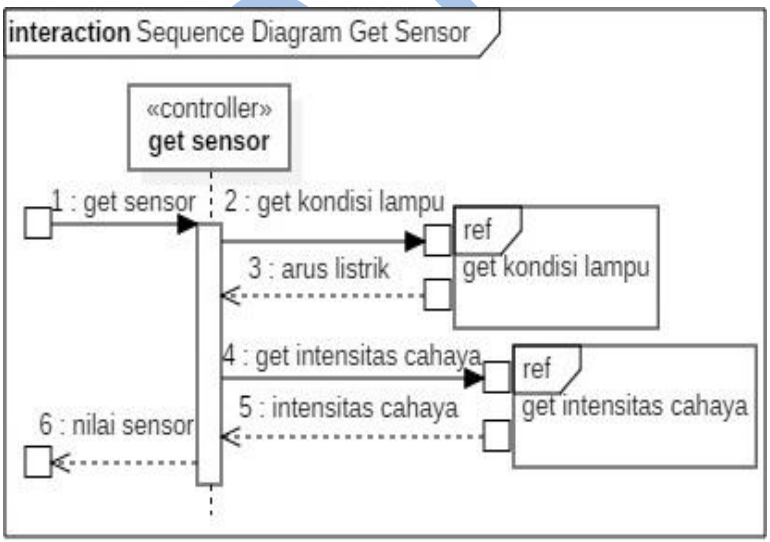
Gambar 3.13 Sequence Diagram Check Siang Malam

3.1.14. Use Case Get Nilai Sensor

Pada kasus penggunaan ini Arduino dapat mengecek lampu sedang menyala atau mati dan kondisi sekitar gelap atau terang. Tujuan dari use case adalah sistem PJU pintar dapat mengubah lampu sesuai gelap dan terang dan nyala mati lampu saat disetel otomatis. Skenario kasus penggunaan dapat dilihat pada tabel 3.13 dan diagram sequence pada gambar 3.14.

Tabel 3.13 Use Case Scenario Get Nilai Sensor

Kode	UC-012
Tujuan	Mendapatkan nilai sensor
Aktor	Waktu
Kondisi Awal	Sistem belum mendapat nilai sensor
Skenario Utama	
1. Sistem mendapatkan trigger 2. Include Use Case Get Kondisi Lampu 3. Sistem mendapatkan arus listrik 4. Include Use Case Get Intensitas Cahaya 5. Sistem mendapatkan intensitas cahaya 6. Sistem mengirim nilai sensor	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mendapat nilai sensor



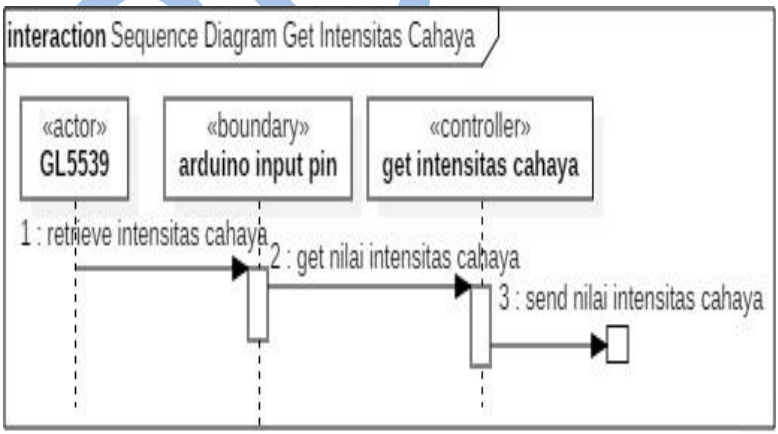
Gambar 3.14 Sequence Diagram Get Sensor

3.1.15. Use Case Get Intensitas Cahaya

Pada kasus penggunaan ini sensor cahaya LM5539 dapat mengecek kondisi sekitar gelap atau terang. Tujuan dari use case adalah sistem otomatis lampu PJU bekerja sesuai kondisi sekitar. Skenario kasus penggunaan dapat dilihat pada tabel 3.14 dan diagram sequence pada gambar 3.15.

Tabel 3.14 Use Case Scenario Get Intensitas Cahaya

Kode	UC-013
Tujuan	Mendapatkan besaran intensitas cahaya
Aktor	LM5539
Kondisi Awal	Arduino belum mendapat nilai intensitas cahaya
Skenario Utama	
1. Aktor mengirimkan besaran intensitas cahaya 2. Sistem mendapatkan besaran intensitas cahaya 3. Sistem mengirimkan nilai intensitas cahaya	
Skenario Alternatif	
-	
Kondisi Akhir	Arduino sudah mendapat nilai intensitas cahaya



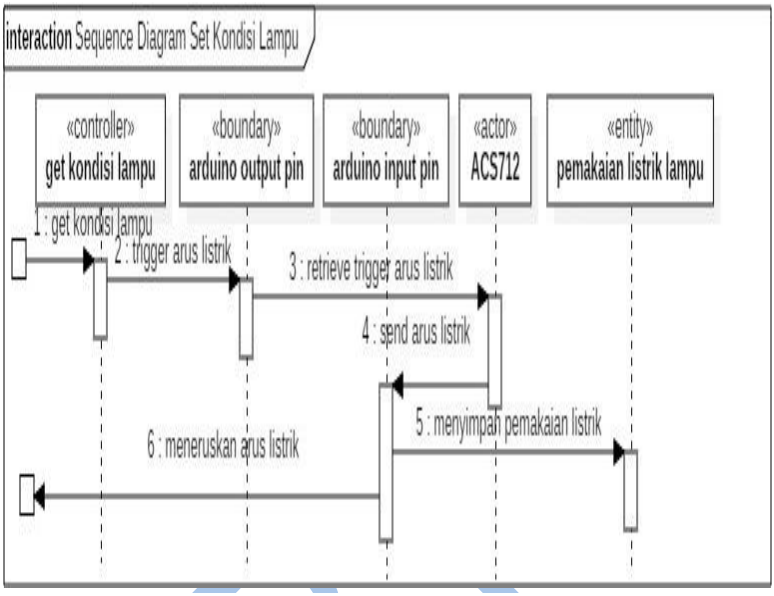
Gambar 3.15 Sequence Diagram Get Intensitas Cahaya

3.1.16. Use Case Get Kondisi Lampu

Pada kasus penggunaan ini sensor arus ACS712 dapat mengecek mengecek lampu sedang menyala atau mati. Tujuan dari use case adalah menampilkan lampu menyala atau mati di halaman web. Skenario kasus penggunaan dapat dilihat pada tabel 3.15 dan diagram sequence pada gambar 3.16.

Tabel 3.15 Use Case Scenario Get Kondisi Lampu

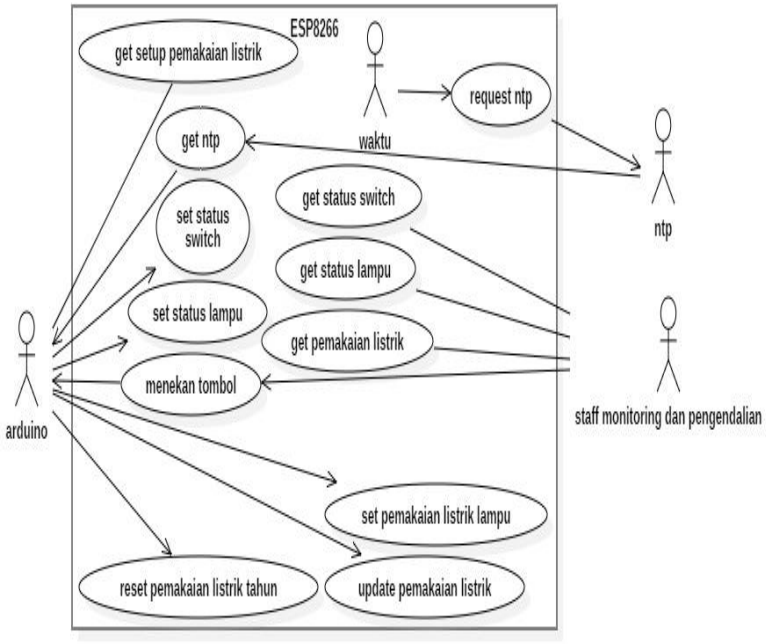
Kode	UC-014
Tujuan	Mendapatkan besaran arus listrik
Aktor	ACS712
Kondisi Awal	Sistem belum mendapatkan besaran arus listrik
Skenario Utama	
1. Sistem meminta kondisi lampu 2. Sistem mengirim trgger arus listrik 3. Aktor menerima trigger arus listrik 4. Aktor mengirim nilai arus listrik 5. Sistem mendapat nilai arus listrik 6. Sistem menyimpan pemakaian listrik lampu 7. Sistem meneruskan nilai arus listrik	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mendapatkan besaran arus listrik



Gambar 3.16 Sequence Diagram Get Kondisi Lampu

3.1.17. Diagram Kasus Penggunaan ESP8266'

Terdapat 3 aktor dalam kasus penggunaan ESP8266 yang terdiri dari server NTP, sistem Arduino dan staff monitoring dan pengendalian. Diagram kasus penggunaan ESP8266 dapat dilihat pada gambar 3.2.



Gambar 3.17 Diagram Kasus Penggunaan ESP8266

3.1.18. Deskripsi Kasus Penggunaan ESP8266

Deskripsi kasus penggunaan ESP8266 yang dibutuhkan pada sistem pengawasan dan pengendalian PJU disesuaikan dengan diagram kasus penggunaan ESP8266. Deskripsi kasus penggunaan ESP8266 dapat dilihat pada Tabel 3.16.

Tabel 3.16 Tabel Deskripsi Kasus Penggunaan ESP8266

Kode Kasus Penggunaan	Nama Kasus Penggunaan
UC-111	Mendapat setup pemakaian listrik

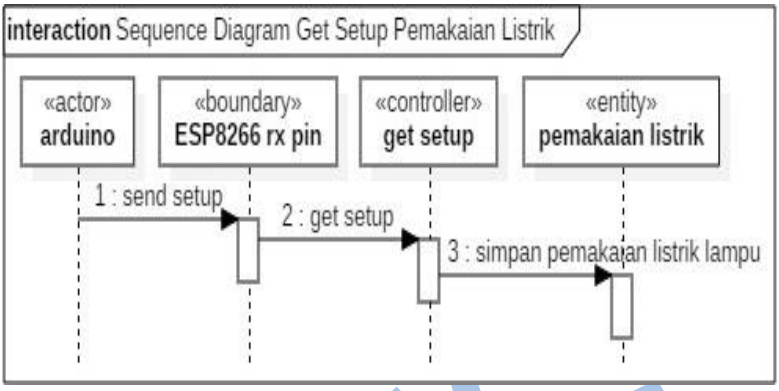
UC-112	Meminta NTP
UC-113	Mendapatkan NTP
UC-114	Menekan tombol
UC-115	Mengeset status tombol
UC-116	Mendapatkan status tombol
UC-117	Mendapatkan status lampu
UC-118	Mendapatkan pemakaian listrik
UC-119	Mengeset status lampu
UC-120	Mengupdate pemakaian listrik
UC-121	Mereset pemakaian listrik tahun
UC-122	Mengeset pemakaian listrik lampu

3.1.19. Use Case Get Setup Pemakaian Listrik

Pada kasus penggunaan ini ESP8266 mendapatkan data pemakaian lsitrik dari Arduino pada saat melakukan proses setup. Tujuan dari use case ini pemakaian listrik dapat ditampilkan saat program ESP8266 dimulai. Skenario kasus penggunaan dapat dilihat pada tabel 3.17 dan diagram sequence pada gambar 3.18.

Tabel 3.17 Use Case Skenario Get Setup Pemakaian Listrik

Kode	UC-111
Tujuan	Mendapatkan nilai sensor
Aktor	Arduino
Kondisi Awal	Sistem belum mendapat setup
Skenario Utama	
1. Actor mengirim setup 2. Sistem menerima setup 3. Sistem menyimpan data pemakaian listrik lampu	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mendapat setup



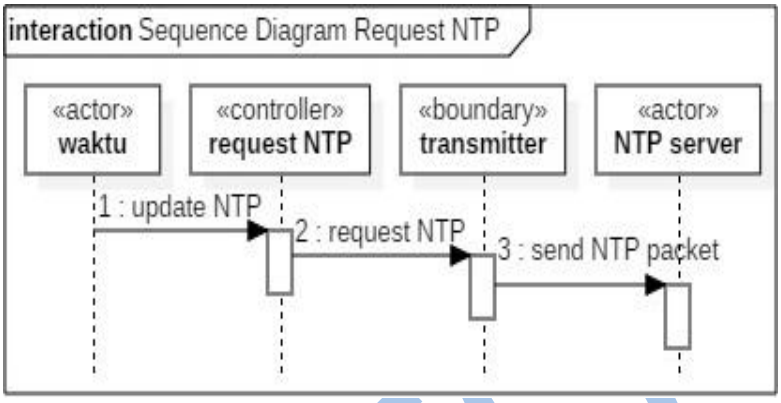
Gambar 3.18 Sequence Diagram Get Setup Pemakaian Listrik

3.1.20. Use Case Request NTP

Pada kasus penggunaan ini ESP8266 meminta NTP kepada server UTC dengan cara mengirim paket NTP. Tujuan dari kasus penggunaan ini adalah sinkronisasi waktu pada ESP8266 dan server NTP. Skenario kasus penggunaan dapat dilihat pada tabel 3.18 dan diagram sequence pada gambar 3.19.

Tabel 3.18 Use Case Scenario Request NTP

Kode	UC-112
Tujuan	Meminta NTP
Aktor	Waktu
Kondisi Awal	Server NTP belum menerima request
Skenario Utama	
1. Aktor memberikan trigger 2. Sistem meminta NTP 3. Sistem mengirim paket NTP ke server	
Skenario Alternatif	
-	
Kondisi Akhir	Server NTP sudah menerima request



3.1.21. Use Case Get NTP

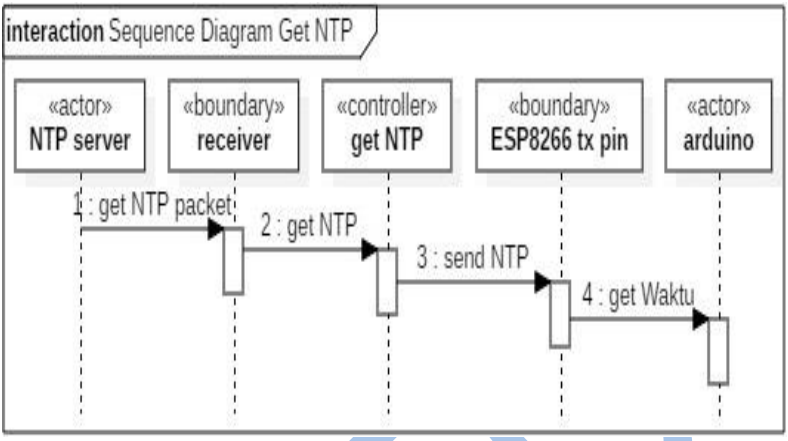
Pada kasus penggunaan ini Server UTC mengirim paket waktu NTP ke ESP8266. Kemudian NTP dikonversi ke waktu local dan dikirim ke Arduino. Tujuan dari use case adalah sistem PJU pintar mendapat waktu yang telah disinkronisasi. Skenario kasus penggunaan dapat dilihat pada tabel 3.19 dan diagram sequence pada gambar 3.20.

Tabel 3.19 Use Case Scenario Get NTP

Kode	UC-113
Tujuan	Mendapat NTP
Aktor	Server NTP
Kondisi Awal	Arduino belum menerima NTP

Gambar 3.19 Sequence Diagram Request NTP

1. Aktor mengirim paket NTP 2. Sistem menerima paket NTP 3. Sistem mengirim NTP 4. Arduino menerima NTP
Skenario Alternatif
-
Kondisi Akhir | Arduino sudah menerima NTP



Gambar 3.20 Sequence Diagram Get NTP

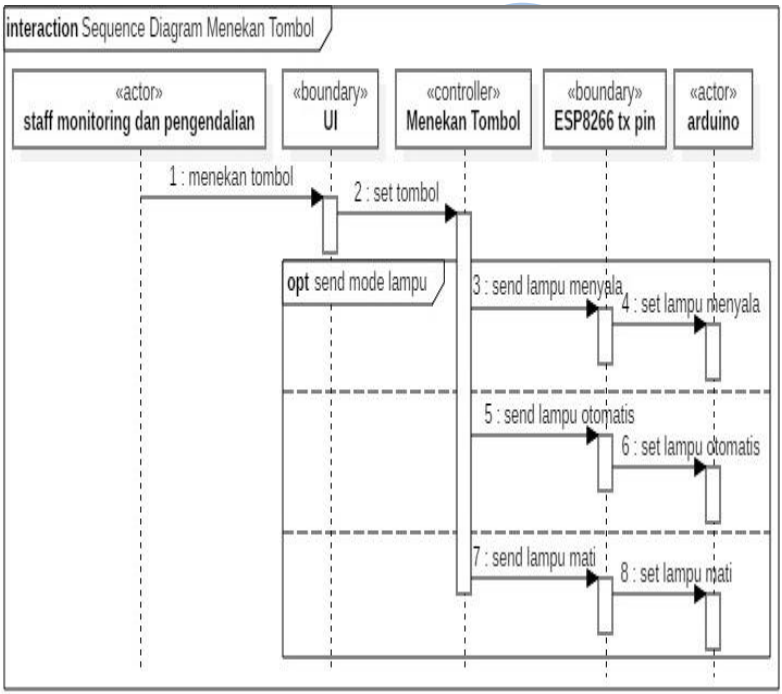
3.1.22. Use Case Menekan Tombol

Pada kasus penggunaan ini staff menekan tombol untuk menginstruksikan ke Arduino untuk mengubah mode lampu. Skenario kasus penggunaan dapat dilihat pada tabel 3.20 dan diagram sequence pada gambar 3.21.

Tabel 3.20 Use Case Scenario Menekan Tombol

Kode	UC-114
Tujuan	Menekan tombol
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Arduino belum mendapat input tombol
Skenario Utama	
1. Aktor menekan tombol 2. Sistem mendapat input tombol	
Skenario Alternatif	
2a Input tombol menyala 2a1 Mengirim tombol lampu menyala 2a2 Arduino menerima lampu menyala	

2b Input tombol otomatis 2b1 Mengirim tombol lampu otomatis 2b2 Arduino menerima lampu otomatis 2c Input tombol mati 2c1 Mengirim tombol lampu mati 2c2 Arduino menerima lampu mati	
Kondisi Akhir	Arduino sudah mendapat input tombol



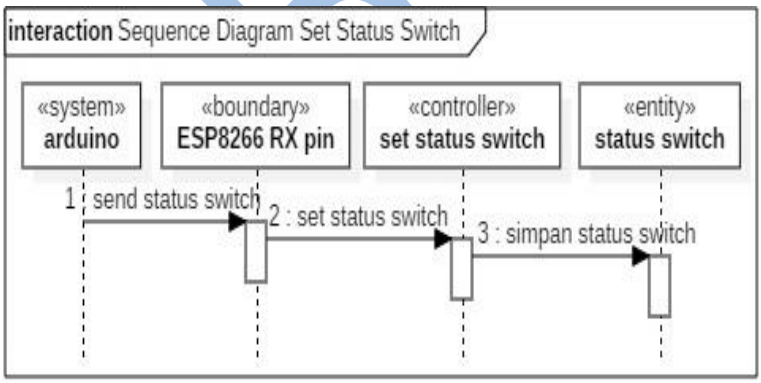
Gambar 3.21 Sequence Diagram Menekan Tombol

3.1.23. Use Case Set Status Tombol

Pada kasus penggunaan ini ESP8266 mendapatkan status tombol yang ditekan dari Arduino kemudian status tombol disimpan. Skenario kasus penggunaan dapat dilihat pada tabel 3.21 dan diagram sequence pada gambar 3.22.

Tabel 3.21 Use Case Scenario Set Status Tombol

Kode	UC-115
Tujuan	Mengeset status tombol
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Status tombol belum diupdate
Skenario Utama	
1. Aktor mengirim status tombol 2. Sistem mengeset status tombol 3. Sistem menyimpan status tombol	
Skenario Alternatif	
-	
Kondisi Akhir	Status tombol sudah diupdate



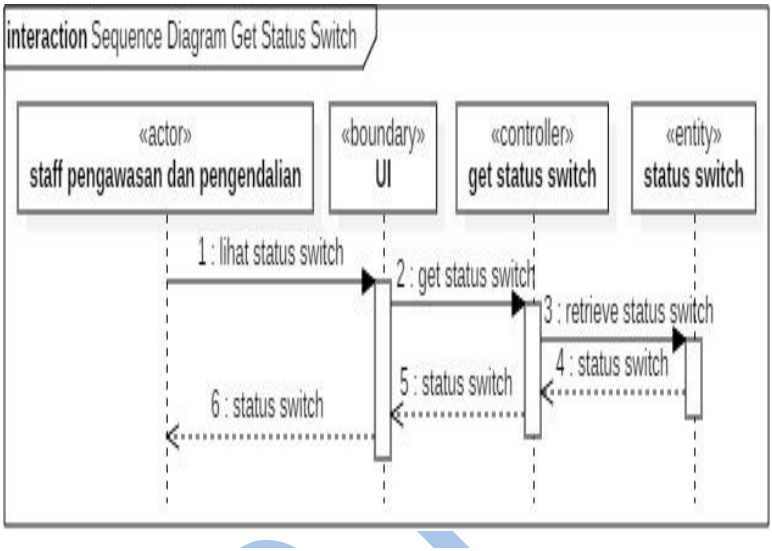
Gambar 3.22 Sequence Diagram Set Status Tombol

3.1.24. Use Case Get Status Tombol

Pada kasus penggunaan ini staff pengawasan dan pengendalian meminta status tombol ke ESP8266 dan mendapatkan respons status tombol yang diminta di halaman website. Tujuan dari kasus penggunaan ini adalah staff pengawasan dan pengendalian mengetahui kondisi status tombol saat itu. Skenario kasus penggunaan dapat dilihat pada tabel 3.22 dan diagram sequence pada gambar 3.23.

Tabel 3.22 Use Case Scenario Get Status Tombol

Kode	UC-116
Tujuan	Mendapat status tombol
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Aktor belum mengetahui status tombol
Skenario Utama	
1. Aktor meminta status tombol 2. Sistem menerima permintaan status tombol 3. Sistem mencari status tombol 4. Sistem menerima status tombol 5. Sistem mengirim status tombol 6. Actor menerima status tombol	
Skenario Alternatif	
-	
Kondisi Akhir	Aktor sudah mengetahui status tombol



Gambar 3.23 Sequence Diagram Get Status Tombol

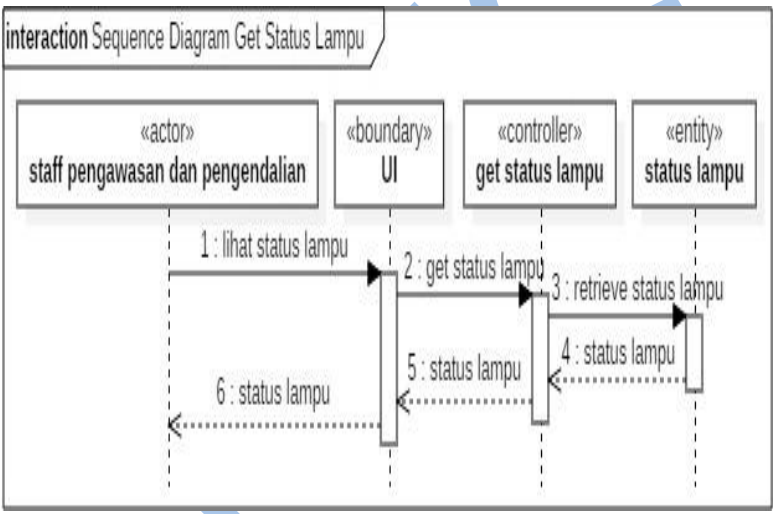
3.1.25. Use Case Get Status Lampu

Pada kasus penggunaan ini staff pengawasan dan pengendalian meminta status lampu ke ESP8266 dan mendapatkan respons status lampu yang diminta di halaman website. Tujuan dari kasus penggunaan ini adalah staff pengawasan dan penendalian mengetahui kondisi status lampu saat itu. Skenario kasus penggunaan dapat dilihat pada tabel 3.23 dan diagram sequence pada gambar 3.24.

Tabel 3.23 Use Case Scenario Get Status Lampu

Kode	UC-117
Tujuan	Mendapat status lampu
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Aktor belum mengetahui status lampu

Skenario Utama	
1. Aktor meminta status lampu 2. Sistem menerima permintaan status lampu 3. Sistem mencari status lampu 4. Sistem menerima status lampu 5. Sistem mengirim status lampu 6. Actor menerima status lampu	
Skenario Alternatif	
-	
Kondisi Akhir	Aktor sudah mengetahui status lampu



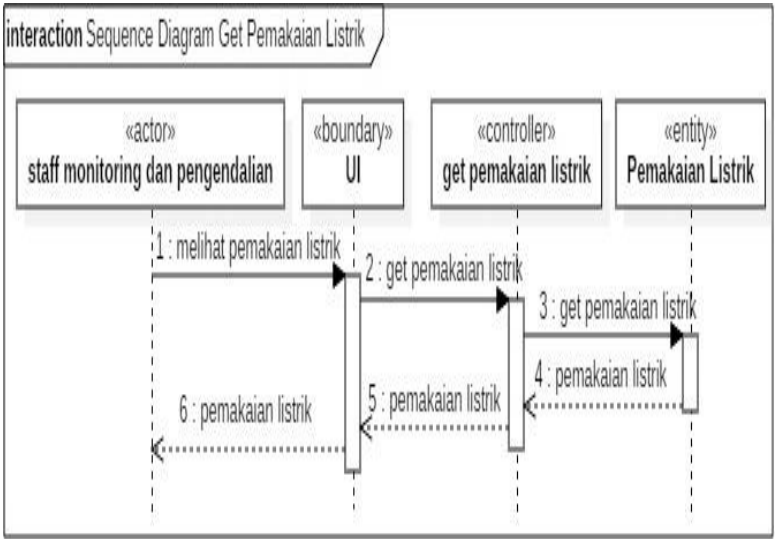
Gambar 3.24 Sequence Diagram Get Status Lampu

3.1.26. Use Case Get Pemakaian Listrik

Pada kasus penggunaan ini staff pengawasan dan pengendalian meminta status pemakaian listrik ke ESP8266 dan mendapatkan respons status pemakaian listrik yang diminta di halaman website. Tujuan dari kasus penggunaan ini adalah staff pengawasan dan penendalian mengetahui kondisi pemakaian listrik saat itu. Skenario kasus penggunaan dapat dilihat pada tabel 3.24 dan diagram sequence pada gambar 3.25.

Tabel 3.24 Use Case Scenario Get Status Pemakaian Listrik

Kode	UC-118
Tujuan	Mendapatkan pemakaian listrik
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Sistem belum mendapat pemakaian listrik
Skenario Utama	
1. Aktor meminta pemakaian listrik 2. Sistem meminta pemakaian listrik 3. Sistem mencari pemakaian listrik 4. Sistem mendapat pemakaian listrik 5. Sistem menampilkan pemakaian listrik 6. Aktor mendapat pemakaian listrik	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mendapat pemakaian listrik



Gambar 3.25 Sequence Diagram Get Pemakaian Listrik

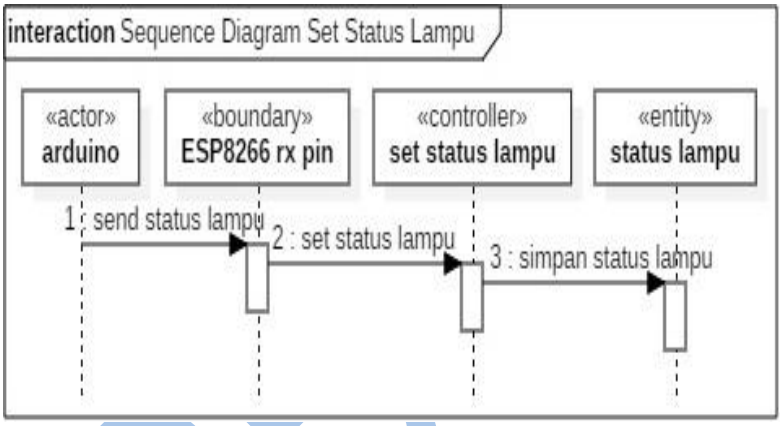
3.1.27. Use Case Set Status Lampu

Pada kasus penggunaan ini ESP8266 mendapatkan status lampu saat itu dari Arduino kemudian disimpan. Skenario kasus penggunaan dapat dilihat pada tabel 3.25 dan diagram sequence pada gambar 3.26.

Tabel 3.25 Use Case Scenario Set Status Lampu

Kode	UC-119
Tujuan	Mengupdate status lampu
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Sistem belum mengupdate status lampu
Skenario Utama	
1. Aktor mengirim status lampu 2. Sistem mengeset status lampu	

3. Sistem menyimpan status lampu	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mengupdate status lampu



Gambar 3.26 Sequence Diagram Set Status Lampu

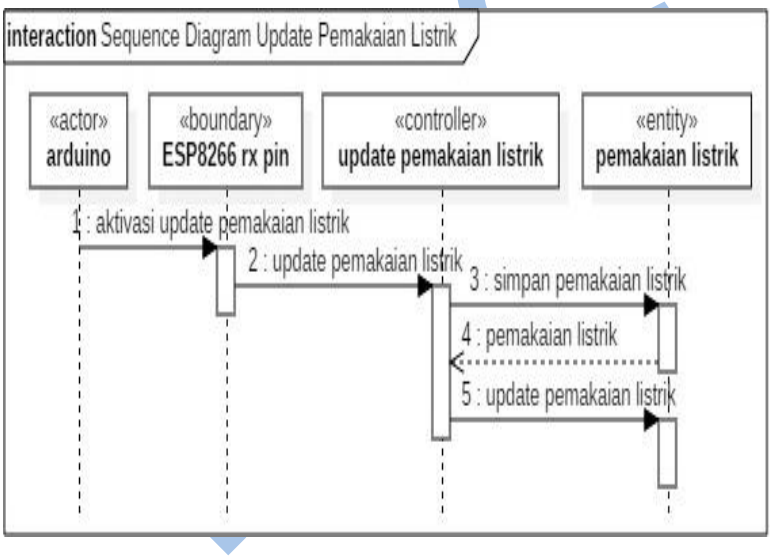
3.1.28. Use Case Update Pemakaian Listrik

Pada kasus penggunaan ini ESP8266 mendapatkan trigger update data pemakaian listrik dari Arduino kemudian melakukan update pemakaian listrik dan disimpan. Tujuan dari kasus penggunaan ini adalah data pemakaian listrik diperbarui. Skenario kasus penggunaan dapat dilihat pada tabel 3.26 dan diagram sequence pada gambar 3.27.

Tabel 3.26 Use Case Update Pemakaian Listrik

Kode	UC-120
Tujuan	Mengupdate pemakaian listrik

Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Sistem belum mengupdate pemakaian listrik
Skenario Utama	
1. Aktor mengirim trigger update pemakaian listrik 2. Sistem mencari data pemakaian listrik 3. Sistem mendapatkan data pemakaian listrik 4. Sistem mengupdate data pemakaian listrik 5. Sistem menyimpan data pemakaian listrik	
Skenario Alternatif	
Kondisi Akhir	Sistem sudah mengupdate pemakaian listrik



Gambar 3.27 Sequence Diagram Update Pemakaian Listrik

3.1.29. Use Case Reset Pemakaian Listrik Tahun

Pada kasus penggunaan ini ESP8266 mendapatkan trigger reset data pemakaian listrik tahun kemudian melakukan reset pemakaian listrik tahun dan disimpan. Skenario kasus penggunaan dapat dilihat pada tabel 3.27 dan diagram sequence pada gambar 3.28.

Tabel 3.27 Use Case Reset Pemakaian Listrik Tahun

Kode	UC-121
Tujuan	Mereset pemakaian listrik tahun
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Sistem belum mereset pemakaian listrik tahun
Skenario Utama	
1. Aktor mengirim trigger reset pemakaian listrik tahun 2. Sistem mereset data pemakaian listrik tahun 3. Sistem menyimpan data pemakaian listrik tahun	
Skenario Alternatif	
Kondisi Akhir	Sistem sudah mereset pemakaian listrik tahun

Gambar 3.28 Sequence Diagram Reset Pemakaian Listrik Tahun

3.1.30. Use Case Set Pemakaian Listrik Lampu

Pada kasus penggunaan ini ESP8266 mendapatkan data pemakaian listrik lampu dari Arduino kemudian disimpan. Tujuan dari use case adalah menampilkan data pemakaian listrik di halaman web. Skenario kasus penggunaan dapat dilihat pada tabel 3.28 dan diagram sequence pada gambar 3.29.

Tabel 3.28 Use Case Scenario Set Pemakaian Listrik

Kode	UC-113
Tujuan	Mengupdate pemakaian listrik
Aktor	Staff monitoring dan pengendalian
Kondisi Awal	Sistem belum mengeset pemakaian listrik
Skenario Utama	

1. Aktor mengirim data pemakaian listrik 2. Sistem mengeset data pemakaian listrik 3. Sistem menyimpan data pemakaian listrik	
Skenario Alternatif	
-	
Kondisi Akhir	Sistem sudah mengeset pemakaian listrik

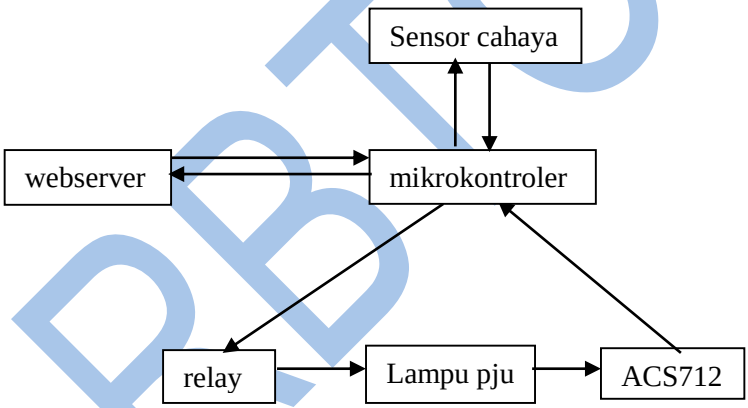
Gambar 3.29 Sequence Diagram Set Pemakaian Listrik Lampu

BAB IV
IMPLEMENTASI

Pada bab implementasi terdiri dari diagram blok beserta modul arduino dan modul ESP8266 dari sistem pengawasan dan pengendalian lampu PJU.

4.1. Diagram Blok Sistem Pengawasan dan Pengendalian Lampu PJU

Diagram blok sistem pengawasan dan pengendalian lampu PJU terdiri dari blok web server, sensor cahaya, mikrokontroller, relay, lampu pju dan ACS712 yang terhubung.



4.2. Implementasi Arduino

Sub-bab implementasi arduino terdiri dari konfigurasi arduino, kode ascii perintah arduino dan modul-modul arduino berdasarkan rincian kode arduino pada tabel 8.2.

4.2.1. Konfigurasi Arduino

Tabel 4.1Konfigurasi Arduino

Pin	Kegunaan
6	Switch lampu
5V	supply relay, supply ESP8266, supply light dependent resistor, supply ACS712

4.2.2. Kode ASCII Perintah Arduino

Kode Perintah Arduino digunakan saat mengirim pesan dari Arduino ke ESP8266 melalui serial dan kemudian ditampilkan di halaman web. Kode ASCII perintah Arduino dapat dilihat pada tabel 4.2.

Tabel 4.2 Kode Ascii Perintah Arduino

Kode	Penjelasan Kode
0	Lampu off
1	Lampu auto
2	Lampu on
3	Lampu mati
4	Lampu Menyala
5	Kirim pemakaian listrik lampu
6	Update pemakaian listrik menit
7	Update pemakaian listrik jam
8	Update pemakaian listrik hari
9	Update pemakaian listrik bulan
10	New Line
11	Update pemakaian listrik tahun
12	Reset pemakaian listrik tahun
13	Carriage Return
126	~

4.2.3. Modul Setup Arduino

Modul ini merupakan inisialisasi sebelum sistem pengawasan dan pengendalian lampu PJU berjalan. Modul setup Arduino dapat dilihat pada pseudocode 4.1.

```
Declare an Current Input Variable InCurrent;
Declare a Light Intensity Input Variable InLight;
Declare a Last Update Time Variable LUT;
Declare a Last Update Lamp Status Time Variable LULST;
Declare iterable variable I, J;
Declare an ampere value A;
Declare a light intensity value LI;
Declare a light dependent resistor switch variable LDR;
Declare a lamp switch variable LAMP;
Declare an electric usage variable EU;
Declare an electric usage lamp variable EUL;
Declare a time stamp serial variable TSS;

Set LDR to output;
Set serial 2 port to 9600 baud rate; // serial 2 is for server
Set LAMP to output;
```

Pseudocode 4.1 Modul Setup Arduino

4.2.4. Modul Main Program Arduino

Modul ini merupakan modul utama yang bertujuan untuk menjalankan Arduino dan memanggil modul lainnya. Berisi tentang looping program selama Arduino berjalan. Tujuan dari modul ini adalah program Sistem Pengawasan dan Pengendalian PJU berjalan. Modul main program arduino dapat dilihat pada pseudocode 4.2.

```
WHILE Arduino is available:
```

```
  Get message from serial event 2;
  IF LUT-program time >= 1 second then:
    Call AutoUpdateTime(LUT-program time in 1 second);
  END IF;
  IF LULST-program time >= 0.1 second then:
    Call LampMonitor();
    LULST=program time;
  END IF;
  Convert ReadVCC() to VRef;
  Collect inCurrent average;
  Get VRef and Convert inCurrent to Ampere;
  Update Ampere Value;
  Update Calibration;
  Update Sample Number;
  IF switch mode == auto, THEN:
    Convert  InLight to LI;
    call CheckNight() and return NIGHT;
    IF LI==dark or NIGHT==true, THEN
      Turn On Lamp
    ELSE IF  Night = false and LI != dark, THEN
      Turn Off Lamp
    END IF;
END WHILE;
```

Pseudocode 4.2 Modul Main Program Arduino

4.2.5. Modul Mengeset Tanggal

Modul ini berisi fungsi mengubah waktu NTP ke tahun, bulan dan hari. Tujuan dari modul ini adalah data pemakaian listrik akurat sesuai waktu. Modul mengeset tanggal dapat dilihat pada pseudocode 4.3.

```
FUNCTION SetTanggal(NTP){
  Convert NTP to day;
  Convert day to year;
  IF a leap year, THEN set LeapYear=true;
  ELSE set LeapYear=false;
  END IF;
  IF PreviousYear < Year, THEN:
    Call UpdateElectricUsage(Year);
    Update year to previous year;
  END IF;
  Get Leapyear and convert Day to month;
  IF PreviousMonth < Month, THEN:
    Call UpdateElectricUsage(Month);
    Update month to previous month;
  END IF;
  set Day value;
  IF PreviousDay < Day, THEN:
    IF AfterNewYear, THEN:
      Call  ResetPemakaianListrikTahun();
    ENDIF;
  END IF;
  Call UpdateElectricUsage(Day);
  Update day to previous day;
}
```

Pseudocode 4.3 Modul Mengeset Tanggal

4.2.6. Modul Pengukuran Tegangan Input Arduino

Modul ini berisi tentang pengukuran tegangan input arduino. Tujuan dari modul ini adalah pengukuran arus dan intensitas cahaya akurat. Modul pengukuran tegangan input arduino dapat dilihat pada pseudocode 4.4.

```
FUNCTION ReadVcc() {
  Read 1.1V reference against AVcc
  Settle input;
  Start Conversion;
  End Conversion;
  read ADC value;
  return ADC value;
}
```

4.2.7. Modul Mengeset Waktu

Modul ini berisi tentang algoritma mengubah waktu NTP ke jam, menit, detik. Tujuan dari modul ini adalah mengeset jam, menit dan detik. Modul mengeset waktu dapat dilihat pada pseudocode 4.4.

```
FUNCTION SetTime(NTP){
  Convert NTP to Hour;
  IF PreviousHour < Hour, THEN:
    Call UpdateElectricUsage(Hour);
    Update month to previous hour;
  END IF;
  Convert NTP to Minute;
  IF PreviousMinute < Minute, THEN:
    Call UpdateElectricUsage(Minute);
    Update minute to previous minute;
  END IF;
  Convert NTP to Second;
  IF PreviousSecond < Second, THEN:
    Call UpdateElectricUsage(Second);
    Update second to previous second;
  END IF;
  Reset last update time;
}
```

```

```

Pseudocode 4.4 Modul Mengeset Waktu

4.2.8. Modul Mengeset Time Stamp

Modul ini memanggil modul mengeset tanggal dan waktu. Tujuan dari modul ini adalah waktu telah diperbarui dan akurat. Modul mengeset time stamp dapat dilihat pada pseudocode 4.5.

```
FUNCTION SetTimeStamp(NTP){
  call SetTanggal(NTP);
  call SetTime(NTP);
}
```

Pseudocode 4.5 Modul Mengeset Time Stamp

4.2.9. Modul Mengecek Malam

Modul ini berisi fungsi pengecekan siang malam. Hasil dari fungsi ini memastikan lampu PJU dalam kondisi menyala pada malam hari dan mati pada siang hari dan disetel otomatis. Modul ini dapat dilihat pada pseudocode 4.6.

```
FUNCTION CekNight(){
  IF malam, THEN return true;
  ELSE return false;
  END IF;
}
```

Pseudocode 4.6 Modul Mengecek Malam

4.2.10. Modul Update Pemakaian Listrik

Modul ini berisi pseudocode mengupdate data pemakaian listrik listrik . Tujuan dari modul ini adalah mengirimkan trigger

update data pemakaian listrik ke ESP8266. Modul Update Pemakaian listrik dapat dilihat pada pseudocode 4.7.

```
FUNCTION UpdatePemakaianListrik(TimeStamp){
  Send Update Pemakaian Listrik Commanad to ESP8266;
}
```

Pseudocode 4.7 Modul Update Pemakaian Listrik

4.2.11. Modul Reset Pemakaian Listrik Tahun

Modul ini berisi pseudocode mereset data pemakaian listrik tahun. Tujuan dari modul ini adalah mengirimkan trigger reset data pemakaian listrik tahun ke ESP8266. Modul Reset Pemakaian listrik tahun dapat dilihat pada pseudocode 4.8.

```
FUNCTION ResetPemakaianListrikTahun(){
  Send Reset Pemakaian Listrik Tahun Command to ESP8266;
}
```

Pseudocode 4.8 Modul Reset Pemakaian Listrik Tahun

4.2.12. Modul Kirim Pemakaian Listrik Lampu

Modul ini berisi fungsi kirim pemakaian listrik lampu dan mengirim pemakaian listrik lampu ke server. Tujuan dari modul ini adalah menyampaikan informasi data pemakaian listrik ke ESP8266. Modul kirim pemakaian listrik dapat dilihat pada pseudocode 4.9.

```
FUNCTION SendPemakaianListrikLampu(){
  Add TimeStamp command to TSS;
  Convert EU to ASCII base;
  Add ASCII base to TSS;
  Send TSS;
}
```

Pseudocode 4.9 Modul Kirim Pemakaian Listrik Lampu

4.2.13. Update Status Lampu dan Saklar

Modul ini berisi mengirimkan status lampu dan saklar secara berkala ke ESP8266. Tujuan dari modul menampilkan status lampu dan saklar yang telah diperbarui ke halaman web. Modul update status lampu dan saklar dapat dilihat pada pseudocode 4.10.

```
LampMonitor(){
  Send all lamp status;
  Send all switch status;
}
```

Pseudocode 4.10 Update Status Lampu dan Saklar

4.2.14. Modul Update Otomatis Waktu

Modul ini berisi fungsi update otomatis lampu apabila tidak ada sinkronisasi waktu dalam 1 detik. Tujuan dari modul adalah menjamin waktu akurat. Modul Update otomatis waktu dapat dilihat pada pseudocode 4.11 dan pseudocode 4.12.

```
AutoUpdateLamp(){
  Reset last update time;
  Update Second;
  Call UpdateElectricUsage(Second);
  IF Second >= Minute, THEN:
    Update Minute;
    Set Second mod 60;
    Call UpdateElectricUsage(Minute);
  END IF;
  IF Minute >= Hour, THEN:
    Update Hour;
    Reset Minute;
    Call UpdateElectricUsage(Hour);
  END IF;
  IF Hour >= Day, THEN:
    IF New Year, THEN: Call ResetPemakaianListrikTahun();
```

```
    END IF;
    Update Day;
    Reset Hour;
    Call UpdateElectricUsage(Day);
  END IF;
  IF Day >= Month, THEN:
    Update Month;
    Reset Day;
    Call UpdateElectricUsage(Month);
  END IF;
  IF Month >= Year, THEN:
    Update Year;
    Reset Month;
    Call UpdateElectricUsage(Year);
  END IF;
}
```

Pseudocode 4.11 Modul Update Otomatis Waktu

4.2.15. Modul Menerjemahkan Perintah ESP8266

Modul ini berisi fungsi penerjemahan perintah ESP8266. Tujuan dari modul membuat sistem PJU pintar berjalan sesuai perintah yang didapat. Modul menerjemahkan perintah ESP8266 dapat dilihat pada pseudocode 4.13.

```
Serial2Command(command){
  IF 0 <= command < 3, THEN:
    Convert command to LAMP mode;
  ELSE IF command==3, THEN: NTPstatus=true;
  ELSE IF command==4, THEN:
    Call SendPemakaianListrikLampu();
  END IF;
  Set SetCommand=true;
}
```

Pseudocode 4.12 Modul Menerjemahkan Perintah ESP8266

4.2.16. Modul Menerima Input Serial ESP8266

Modul ini berisi fungsi menerima input dari ESP8266. Tujuan dari modul membuat sistem PJU pintar berjalan dengan baik. Modul Menerima Input Serial ESP8266 dapat dilihat pada pseudocode 4.14.

```
SerialEvent(){
  WHILE serial 2 available:
    Read byte from serial 2 into C;
    IF C==CarriageReturn OR C==NewLine, THEN:
      IF NTPstatus==true, THEN: call SetTimeStamp(string);
      END IF;
    Reset NTPstatus;
    ELSE IF 0<=C<=5, THEN: call Serial1Command(C);
    ELSE IF NTPstatus==true, THEN:
      Remove all command character and merge up;
      Convert C from ASCII base to decimal base NTP;
    END IF;
  END WHILE;
}
```

Pseudocode 4.13 Modul Menerima Input Serial ESP8266

4.3. Implementasi ESP8266

Sub-bab implementasi ESP8266 terdiri dari konfigurasi ESP8266, kode ascii perintah ESP8266 dan modul-modul ESP8266 berdasarkan rincian kode ESP8266 pada tabel 7.2.

4.3.1. Kode ASCII ESP8266

Kode Perintah ESP8266 digunakan saat mengirim pesan dari ESP8266 KE Arduino melalui serial dan kemudian diolah di arduino. Kode ASCII perintah ESP8266 dapat dilihat pada tabel 4.2.

Tabel 4.3 Kode ASCII ESP8266

Kode	Penjelasan Kode
0	Lampu 1 off
1	Lampu 1 auto
2	Lampu 1 on
3	NTP
4	Setup pemakaian listrik
10	New Line
13	Carriage return
126	~

4.3.2. Modul Setup ESP8266

Modul ini merupakan inisialisasi sebelum ESP8266 berjalan, mengeset ntp, baud rate, ip, gateway, subnet, dns dan hyperlink. Modul setup Arduino dapat dilihat pada pseudocode 4.15.

```
Declare a lamp status LS;
Declare a lamp switch SW;
Declare a Send Interval SI;
Call SetNTPUDP();
Set serial baud rate;
Set wifi ip, gateway, subnet, DNS;
Delay program until wifi connected;
Call LampOff() for TurnOff url link;
Call LampAuto() for TurnAuto url link;
Call LampOn() for TurnOn url link;
Call HandlenotFound() for PagenotFound url link;
Starting server;
Call BeginNTP();
```

Pseudocode 4.14 Modul Setup ESP8266

4.3.3. Modul Main program ESP8266

Modul ini modul utama yang bertujuan untuk menjalankan ESP8266 dan memanggil modul lainnya. Berisi fungsi update waktu NTP. Tujuan dari modul ini adalah staff pengawasan dan pengendalian dapat melihat data terbaru dan akurat. Modul main program arduino dapat dilihat pada pseudocode 4.16.

```
WHILE ESP8266 available DO:
  Call updateNTP();
  Get message from serial event;
  Call LocalTime();
  IF SI>program time, THEN:
    Add NTP command to NTPS;
    Convert NTP LocalTime() to ascii base;
    Add asci base NTP to NTPS;
    Send NTPS time to serial;
    SI=Program time;
  END IF;
  Listen client HTTP request;
END WHILE;
```

Pseudocode 4.15 Modul Main Program ESP8266

4.3.4. Modul Inisialisasi NTP PORT

Modul ini berfungsi mengupdate udp wifi udp. Modul inisialisasi NTP port dapat dilihat pada pseudocode 4.17.

```
SetNTPUDP(){
  Set UDP to WIFI UDP;
}
```

Pseudocode 4.16 Modul Inisialisasi NTP PORT

4.3.5. Modul Begin NTP

Modul ini berisi fungsi mengupdate port dafault NTP. Modul Begin NTP dapat dilihat pada pseudocode 4.18.

```
BeginNTP(){
  Set NTP default port;
  Begin NTP default port;
}
```

Pseudocode 4.17 Modul Begin NTP

4.3.6. Modul Request NTP

Modul ini berisi fungsi meminta waktu server NTP.Tujuan dari modul ini adalah mendapat waktu NTP. Modul Request NTP dapat dilihat pada pseudocode 4.19.

```
Send NTP(){
  Set NTP protocol;
  Begin NTP packet to NTP server;
  Write NTP protocol to NTP packet;
  End NTP packet;
}
```

Pseudocode 4.18 Modul Requeset NTP

4.3.7. Modul Update NTP

Modul ini berisi fungsi mengupdate waktu NTP sesuai server NTP. Tujuan dari modul ini adalah sinkronisasi waktu data pemakaian listrik dan kalibrasi arus. Modul update NTP dapat dilihat pada pseudocode 4.20.

```
UpdateNTP(){
  IF last update-program time > interval update NTP, THEN:
  Call SendNTP();
  DO{
```

```
    Parse NTP packet;
    IF timeout, THEN: return false;
    END IF;
  }WHILE cannot parse NTP packet;
  LastUpdate = ProgramTime;
  Read NTP packet from NTP server;
  Convert NTP packet to unix time;
 END IF;
}
```

Pseudocode 4.19 Modul Update NTP

4.3.8. Modul Mendapatkan Waktu Lokal

Modul ini berisi fungsi mengubah waktu NTP ke waktu lokal. Tujuan dari modul ini adalah mengubah waktu sesuai zona waktu berlaku. Modul mendapatkan waktu lokal dapat dilihat pada pseudocode 4.21.

```
LocalTime(){
  Convert NTP server time to NTP local time;
  Return local time;
}
```

Pseudocode 4.20 Modul Mendapatkan Waktu Lokal

4.3.9. Menerjemahkan Perintah Arduino

Modul ini berisi fungsi penerjemahan perintah Arduino. Tujuan dari modul adalah memperbarui status lampu dan status saklar, mengirim status pemakaian listrik lampu, mengupdate pemakaian listrik, dan mereset pemakaian listrik tahun. Modul menerjemahkan perintah ESP8266 dapat dilihat pada pseudocode 4.22.

```
SerialCommand(command){
  IF 0 <= command < 3, THEN:
    Convert command to SWITCH index mode;
  ELSE IF 3<=command<5, THEN:
    Convert Command to LAMP mode;
  ELSE IF command==5, THEN:
    Set SetCommand=true;
  ELSE IF 6<=command<12, THEN:
    Update Time electic usage;
  ELSE IF command==12, THEN:
    Update Time electic usage;
  END IF;
  }
```

Pseudocode 4.21 Modul Menerjemahkan Perintah Arduino

4.3.10. Modul Menerima Input Serial

Modul ini berisi fungsi menerima input dari Arduino. Tujuan dari modul membuat data pemakaian listrik, status saklar dan status lampu diperbarui. Modul Menerima Input Serial Arduino dapat dilihat pada pseudocode 4.23.

```
SerialEvent(){
WHILE serial available:
    Read character from serial into C;
    IF C==CarriageReturn OR C==NewLine, THEN:
      IF TimeStampStatus==true, THEN:
        Calculate WattHour;
        Calculate Cost;
      END IF;
    END IF;
    Reset TimeStampStatus;
    IF 0<=C<13, THEN: call SerialCommand(C);
    ELSE IF TimeStampStatus==true, THEN:
      Remove all comand char and then merge up;
      Convert C from ASCII base to decimal base NTP;
```

```
    END IF;
END WHILE;
}
```

Pseudocode 4.22Modul Menerima Input Serial

4.3.11. Modul Handle Lamp Off

Modul ini berisi fungsi Mematikan saklar sesuai instruksi yang diberikan. Tujuan dari modul ini adalah mematikan lampu PJU sesuai lokasi yang dituju. Modul handle lamp off dapat dilihat pada pseudocode 4.24.

```
LampOff(){
  Send command turn off lamp to serial;
  Redirect page;
}
```

Pseudocode 4.23 Modul Handle Lamp Off

4.3.12. Modul Handle Lamp Auto

Modul ini berisi fungsi mengubah saklar ke posisi sesuai instruksi yang diberikan. Tujuan dari modul ini adalah lampu PJU berubah otomatis berdasar nilai pemberian sensor cahaya dan waktu. Modul handle lamp auto dapat dilihat pada pseudocode 4.25.

```
LampAuto(){
  Send command turn lamp auto to serial;
  Redirect page;
}
```

Pseudocode 4.24 Modul Handle Lamp Auto

4.3.13. Modul Handle Lamp On

Modul ini berisi fungsi Menyalakan saklar sesuai instruksi yang diberikan. Tujuan dari modul ini adalah menyalakan

lampu PJU sesuai lokasi yang dituju. Modul handle lamp off dapat dilihat pada pseudocode 4.26.

```
LampOn(){
  Send command turn on lamp to serial;
  Redirect page;
}
```

Pseudocode 4.25 Modul Handle Lamp On

4.3.14. Modul Menampilkan Informasi Lampu, Tombol, dan Pemakaian Listrik

Modul ini berisi fungsi menampilkan halaman webserver. Tujuan dari modul ini memberikan informasi ke staff pengawasan dan pengendalian mengenai status lampu, tombol dan data pemakaian listrik. Modul menampilkan informasi lampu, tombol, dan data pemakaiana listrik dapat dilihat pada pseudocode 4.27.

```
Website(){
  Show LAMP status
  IF LAMP==ON, THEN background-color=green;
  ELSE background-color=red;
  ENDIF
  IF SWITCH=BUTTON, THEN background-color=blue;
  ELSE background-color=red;
  END IF;
FOR i=0 to size of TimeStamp;
  Show TimeStamp;
  Show WattHour;
  Show Cost;
END FOR;
}
```

Pseudocode 4.26 Modul Menampilkan Informasi Lampu, Tombol, dan Pemakaian Listrik

4.3.15. Modul Handle Not Found

Modul ini berisi fungsi menampilkan halaman web tidak tersedia. Tujuan dari modul ini memberikan informasi ke staff pengawasan dan pengendalian bahwa url tidak tersedia. Modul Handle not found dapat dilihat pada pseudocode 4.28.

```
HandlenotFound(){
  Show page not found;
}
```

Pseudocode 4.27 Modul Handle Not Found

[halaman ini sengaja dikosongkan]

BAB V
UJI COBA DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada sistem pengawasan dan pengendalian PJU yang dibangun. Pengujian yang dilakukan adalah pengujian funsionalitas scenario dan pengujian performa. Pengujian fungsionalitas mengacu pada kasus penggunaan pada bab tiga. Pengujian performa dilakukan dengan menguji sistem. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

5.1. Lingkungan Implementasi

Lingkungan uji coba meliputi perangkat keras dan perangkat lunak yang digunakan dalam sistem ini. Spesifikasi perangkat keras dan perangkat lunak dalam rangka uji coba dapat dilihat pada tabel

Tabel 5.1 Lingkungan Pengujian

Perangkat Keras	
Sistem Pengawasan dan Pengendalian PJU	Terdiri dari :
	Arduino Mega 2560
	ACS712
	Light Dependent Resistor
	Relay
Modul Wifi	Terdiri dari :
	ESP8266
	Voltage Divider
Lampu PJU	Terdiri dari:
	LM1117
	Lampu LED
Power Suply Relay	Arduino Uno
Perangkat Lunak	
Sistem Operasi	Windows 10 64 bit
Software Arduino	Arduino IDE 1.8.8.
Peramban	Google Chrome 71.0.3578.98 64 bit

5.2. Uji Performa Pengukuran Analog Read

Pengukuran Performa Analog Read dilakukan dengan mengubah nilai prescale ADC pada Arduino.Berikut adalah hasil pengukuran analogread pada faktor *prescale* 128 yamg ditampilkan pada tabel 5.2 dan faktor *prescale* 16 yang ditampilkan pada tabel 5.3.

Tabel 5.2 Analog Read dengan Prescale Factor 128

Faktor *Prescale* 128		
Baudrate (Bps)	Waktu (µs)	Frekuensi (Hz)
9600	112,0016	8.928,374
115200	112,0033	8.928,463

Tabel 5.3 Analog Read dengan Prescale Factor 16

Faktor *Prescale* 16		
Baudrate (Bps)	Waktu (µs)	Frekuensi (Hz)
9600	15,05994	66.400,35
115200	15,05883	66.407,38

Dari hasil pengujian didapat kesimpulan perubahan ADPS dapat memengaruhi kecepatan membaca analog read. Namun dengan jumlah lapu PJU yang melimpah maka solusi ini tidak efektif dilakukan.

5.3. Uji Skenario Arduino

5.3.1. Skenario Mengeset Waktu

Pada skenario mengeset waktu dilakukan dengan cara mengirimkan data dari NTP ke Arduino kemudian disimpan. Hasil uji coba mengeset waktu dapat dilihat pada table 5.4.

Tabel 5.4 Skenario Mengeset Waktu

Nama	Mengeset waktu
Tujuan Uji Coba	Menguji fungsionalitas waktu
Kondisi Awal	Waktu belum diubah
Skenario	NTP server mengirimkan waktu kemudian mengkonversi ke waktu
Masukan	Waktu NTP
Keluaran	Waktu sudah diubah
Hasil Pengujian	Belum berhasil

5.3.2. Skenario Mengupdate Waktu

Pada skenario mengupdate waktu dilakukan dengan cara mengupdate data waktu kemudian disimpan. Hasil uji coba mengeset waktu dapat dilihat pada tabel 5.5.

Tabel 5.5 Skenario Mengupdate Waktu

Nama	Mengupdate Waktu
Tujuan Uji Coba	Menguji fungsionalitas update waktu
Kondisi Awal	Waktu belum diubah
Skenario	Interval waktu program mentrigger update waktu kemudian mengkonversi ke menit
Masukan	Interval waktu program
Keluaran	Waktu sudah diubah
Hasil Pengujian	Belum berhasil

5.3.3. Skenario Mengirim Data Pemakaian Listrik Lampu

Pada skenario mengirim data pemakaian listrik lampu dilakukan dengan cara mengirim data pemakaian listrik lampu

kemudian disimpan dan dikirim. Hasil uji coba mengupdate data pemakaian listrik dapat dilihat pada tabel 5.6.

Tabel 5.6 Skenario Mengirim Data Pemakaian Listrik Lampu

Nama	Mengirim Data Pemakaian Listrik Lampu
Tujuan Uji Coba	Menguji fungsionalitas update data pemakaian listrik
Kondisi Awal	Data pemakaian listrik belum dikirim
Skenario	Masukan mengirim data pemakaian listrik lampu dalam kurun detik
Masukan	Data pemakaian listrik dalam kurun waktu
Keluaran	Data Pemakaian Listrik lampu sudah dikirim
Hasil Pengujian	Belum berhasil

5.3.4. Skenario Mengupdate Data Pemakaian Listrik

Pada skenario mengupdate data pemakaian listrik dilakukan dengan cara mengupdate data pemakaian listrik kemudian disimpan dan dikirim. Hasil uji coba mengupdate data pemakaian listrik dapat dilihat pada tabel 5.7.

Tabel 5.7 Skenario Mengupdate Data Pemakaian Listrik

Nama	Mengupdate Data Pemakaian Listrik
Tujuan Uji Coba	Menguji fungsionalitas update data pemakaian listrik
Kondisi Awal	Trigger update data pemakaian listrik belum dikirim
Skenario	Masukan mengirim trigger update data pemakaian listrik dalam kurun waktu tertentu
Masukan	Trigger update data pemakaian listrik
Keluaran	Trigger update data Pemakaian Listrik sudah dikirim
Hasil Pengujian	Belum berhasil

5.3.5. Skenario Mereset Data Pemakaian Listrik Tahun

Pada skenario mereset data pemakaian listrik tahun dilakukan dengan cara mereset data pemakaian listrik tahun kemudian disimpan dan dikirim. Hasil uji coba mereset data pemakaian listrik tahun dapat dilihat pada tabel 5.8.

Nama	Mereset Data Pemakaian Listrik Tahun
Tujuan Uji Coba	Menguji fungsionalitas reset data pemakaian listrik tahun
Kondisi Awal	Trigger reset data pemakaian listrik tahun belum dikirim
Skenario	Masukan mentrigger reset data pemakaian listrik tahun kemudian mereset data pemakaian listrik tahun
Masukan	Trigger reset data pemakaian listrik
Keluaran	Trigger reset data Pemakaian Listrik tahun sudah dikirim
Hasil Pengujian	Belum Dilakukan

Tabel 5.8 Skenario Mereset Data Pemakaian Listrik Tahun

5.3.6. Skenario Mendapatkan Setup Data Pemakaian Listrik

Pada skenario mendapatkan setup pemakaian listrik dilakukan dengan cara mengupdate data pemakaian listrik lampu kemudian disimpan dan dikirim. Hasil uji coba mendapatkan setup data pemakaian listrik dapat dilihat pada tabel 5.9.

Tabel 5.9 Skenario Mendapatkan Setup Data Pemakaian Listrik

Nama	Mendapatkan Setup Data Pemakaian Listrik

Tujuan Uji Coba	Menguji fungsionalitas setup data pemakaian listrik
Kondisi Awal	Setup data pemakaian listrik belum dikirim
Skenario	Masukan mentrigger setup data pemakaian listrik kemudian mengirim setup data pemakaian listrik
Masukan	Mendapatkan perintah setup pemakaian listrik dari ESP8266
Keluaran	Setup data pemakaian listrik sudah dikirim
Hasil Pengujian	Belum berhasil

5.3.7. Skenario Mendapatkan Status Switch

Pada skenario mendapatkan status switch dilakukan dengan cara mengirimkan status switch ke ESP8266 kemudian ditampilkan ke halaman web. Hasil uji coba mendapatkan status switch dapat dilihat pada tabel 5.10.

Tabel 5.10 Skenario Mendapatkan Status Switch

Nama	Mendapatkan Status Switch
Tujuan Uji Coba	Menguji fungsionalitas mendapat status switch
Kondisi Awal	Status switch belum dikirim
Skenario	Status switch dikirimkan ke ESP8266 secara rutin
Masukan	Status switch
Keluaran	Status switch sudah dikirim
Hasil Pengujian	Berhasil

5.3.8. Skenario Mendapatkan Status Lampu

Pada skenario mendapatkan status lampu dilakukan dengan cara mengirimkan status lampu ke ESP8266 kemudian

ditampilkan ke halaman web. Hasil uji coba mendapatkan status lampu dapat dilihat pada tabel 5.11.

Tabel 5.11 Skenario Mendapatkan Status Lampu

Nama	Mendapatkan Status Lampu
Tujuan Uji Coba	Menguji fungsionalitas mendapat status lampu
Kondisi Awal	Status lampu belum dikirim
Skenario	Status lampu dikirimkan ke ESP8266 secara rutin
Masukan	Status lampu
Keluaran	Status lampu sudah dikirim
Hasil Pengujian	Berhasil

5.3.9. Skenario Mengeset Lampu

Pada skenario mengeset lampu dilakukan dengan menekan tombol di halaman web kemudian dikirim dan mengubah saklar. Hasil uji coba mengeset lampu dapat dilihat pada tabel 5.12.

Tabel 5.12 Skenario Mengeset Lampu

Nama	Mengeset lampu
Tujuan Uji Coba	Menguji fungsionalitas rangkaian lampu dan saklar
Kondisi Awal	Lampu belum berubah
Skenario	Staff pengawasan dan pengendalian menekan tombol di website kemudian dikirim ke Arduino untuk mengubah saklar
Masukan	Instruksi lampu
Keluaran	Lampu sudah berubah
Hasil Pengujian	Berhasil

5.3.10. Skenario Mengubah Lampu Otomatis

Pada skenario mengubah lampu otomatis dilakukan dengan melihat perubahan lampu pada malam hari dan siang hari dan gelap terang kondisi sekitar jika disetel otomatis. Hasil uji coba mengubah lampu otomatis dapat dilihat pada tabel 5.13.

Tabel 5.13 Skenario Mengubah Lampu Otomatis

Nama	Mengubah lampu otomatis
Tujuan Uji Coba	Menguji fungsionalitas saklar, arus, intensitas cahaya dan siang-malam
Kondisi Awal	Lampu belum berubah
Skenario	Mikrokontroller mendapatkan nilai dari sensor arus, intensitas cahaya dan siang-malam kemudian mengubah lampu
Masukan	Instruksi lampu berubah sesuai kondisi sekitar
Keluaran	Lampu sudah berubah
Hasil Pengujian	Berhasil

5.3.11. Skenario Mengecek Siang Malam

Pada skenario mengecek siang malam dilakukan dengan melihat perubahan lampu pada malam hari dan siang hari jika disetel otomatis. Hasil uji coba mengecek siang malam dapat dilihat pada table 5.14.

Tabel 5.14 Skenario Mengecek siang malam

Nama	Mengecek siang malam
Tujuan Uji Coba	Mengecek siang malam dari data waktu
Kondisi Awal	Sistem PJU pintar belum mengetahui siang atau malam

Skenario	Mikrokontroller mengecek siang malam kemudian apabila malam lampu tidak mati ketika diset otomatis dan apabila siang lampu berubah sesuai kondisi sekitar
Masukan	Meminta status siang malam dan apabila siang lampu berubah sesuai kondisi sekitar
Keluaran	Lampu tidak berubah ketika malam
Hasil Pengujian	Belum Berhasil

5.3.12. Skenario Mendapatkan Nilai Sensor

Pada skenario mendapatkan nilai sensor dilakukan dengan melihat nilai sensor pada serial. Hasil uji coba mendapatkan nilai sensor dapat dilihat pada tabel 5.15.

Tabel 5.15 Skenario Mengecek Nilai Sensor

Nama	Mengecek nilai sensor
Tujuan Uji Coba	Mengecek nilai intensitas dan kondisi lampu
Kondisi Awal	Sensor belum melakukan pengukuran
Skenario	Mikrokontroller meminta nilai dari sensor kemudian mengubah lampu ketika diset otomatis
Masukan	Meminta nilai sensor
Keluaran	Sensor sudah melakukan pengukuran
Hasil Pengujian	Berhasil

5.3.13. Skenario Mendapatkan Nilai Intensitas Cahaya

Pada skenario mendapatkan nilai intensitas cahaya dilakukan dengan melihat nilai intensitas cahaya pada serial. Hasil uji coba mendapatkan nilai intensitas cahaya dapat dilihat pada tabel 5.16.

Tabel 5.16 Skenario Mengecek Nilai Sensor Intensitas Cahaya

Nama	Mengecek nilai sensor intensitas cahaya
Tujuan Uji Coba	Mengecek nilai sensor intensitas cahaya
Kondisi Awal	Sensor cahaya belum melakukan pengukuran
Skenario	Mikrokontroller meminta nilai dari sensor cahaya kemudian mengubah lampu ketika diset otomatis
Masukan	Meminta nilai sensor cahaya
Keluaran	Sensor cahaya sudah melakukan pengukuran
Hasil Pengujian	Berhasil

5.3.14. Skenario Mendapatkan Arus Listrik

Pada skenario mendapatkan arus listrik dilakukan dengan melihat nilai intensitas cahaya pada serial. Hasil uji coba mendapatkan arus listrik dapat dilihat pada tabel 5.17.

Tabel 5.17 Skenario Mendapatkan Arus Listrik

Nama	Mendapatkan Arus Listrik
Tujuan Uji Coba	Mendapatkan Kondisi Lampu
Kondisi Awal	Sensor arus belum melakukan pengukuran
Skenario	Mikrokontroller meminta nilai dari arus kemudian mengubah lampu ketika diset otomatis
Masukan	Meminta nilai sensor arus
Keluaran	Sensor arus sudah melakukan pengukuran
Hasil Pengujian	Berhasil

5.4. Uji Skenario ESP8266

5.4.1. Skenario Mendapatkan Setup Pemakaian listrik

Pada skenario mendapatkan setup pemakaian listrik dilakukan dengan mengirim permintaan pemakaian listrik saat setup. Hasil uji coba mendapatkan kondisi lampu dapat dilihat pada tabel 5.18.

Tabel 5.18 Skenario Mendapatkan Setup Pemakaian Listrik

Nama	Mendapat setup pemakaian listrik
Tujuan Uji Coba	ESP8266 Mendapat pemakaian listrik saat inisialisasi
Kondisi Awal	ESP8266 belum mendapat data pemakaian listrik
Skenario	ESP8266 meminta setup pemakaian listrik kemudian mendapat data pemakaian listrik
Masukan	Meminta setup pemakaian listrik
Keluaran	ESP8266 sudah mendapat data pemakaian listrik
Hasil Pengujian	Berhasil

5.4.2. Skenario Meminta NTP

Pada skenario meminta NTP dilakukan dengan mengirim paket ke server NTP. Hasil uji coba meminta NTP dapat dilihat pada tabel 5.19.

Tabel 5.19 Skenario Meminta NTP

Nama	Meminta NTP
Tujuan Uji Coba	ESP8266 Mendapatkan waktu NTP
Kondisi Awal	ESP8266 belum mengirim paket NTP
Skenario	ESP8266 mengirim paket ke server NTP

Masukan	Paket NTP
Keluaran	ESP8266 belum mengirim paket NTP
Hasil Pengujian	Berhasil

5.4.3. Skenario Mendapatkan NTP

Pada skenario mendapatkan NTP dilakukan dengan mengirim waktu NTP ke arduino. Hasil uji coba mendapatkan NTP dapat dilihat pada tabel 5.20.

Tabel 5.20 Skenario Mendapatkan NTP

Nama	Mendapatkan NTP
Tujuan Uji Coba	Arduino Mendapatkan waktu NTP
Kondisi Awal	Arduino belum mendapatkan waktu NTP
Skenario	server NTP mengirim paket ke ESP8266 kemudian diteruskan ke arduino
Masukan	Waktu NTP
Keluaran	Arduino sudah mendapatkan waktu NTP
Hasil Pengujian	Berhasil

5.4.4. Skenario Menekan Tombol

Pada skenario menekan tombol dilakukan dengan mengirim tombol ke arduino dan disimpan. Hasil uji coba menekan tombol dapat dilihat pada tabel 5.21.

Tabel 5.21 Skenario Menekan Tombol

Nama	Menekan tombol
Tujuan Uji Coba	Menguji fungsionalitas menyimpan status tombol
Kondisi Awal	Tombol belum ditekan

Skenario	Staff pengawasan dan pengendalian menekan tombol kemudian mengirim tombol ke arduino
Masukan	tombol
Keluaran	Tombol sudah ditekan
Hasil Pengujian	Berhasil

5.4.5. Skenario Mengeset Status Tombol

Pada skenario mengeset status tombol dilakukan dengan mengirim status tombol ke ESP8266 dan disimpan. Hasil uji coba mengeset status tombol dapat dilihat pada tabel 5.22.

Tabel 5.22 Skenario Mengeset status tombol

Nama	Menegeset status tombol
Tujuan Uji Coba	Menguji fungsionalitas menyimpan status tombol
Kondisi Awal	Data tombol belum diperbarui
Skenario	Arduino mengirim perintah status tombol dan disimpan
Masukan	Instruksi tombol
Keluaran	Data tombol sudah diperbarui
Hasil Pengujian	Berhasil

5.4.6. Skenario Mendapatkan Status Tombol

Pada skenario mendapatkan status tombol dilakukan dengan menampilkan status tombol ke halaman web. Hasil uji coba mendapatkan status tombol dapat dilihat pada tabel 5.23.

Tabel 5.23 Skenario Mendapatkan Status Tombol

Nama	Mendapatkan status tombol

Tujuan Uji Coba	Menguji fungsionalitas melihat status tombol
Kondisi Awal	Data tombol belum ditampilkan
Skenario	ESP8266 menampilkan data tombol ke staff pengawasan dan pengendalian
Masukan	Data tombol
Keluaran	Data tombol sudah ditampilkan
Hasil Pengujian	Berhasil

5.4.7. Skenario Mengeset Status Lampu

Pada skenario mengeset status lampu dilakukan dengan mengirim status lampu ke ESP8266 dan disimpan. Hasil uji coba mengeset status lampu dapat dilihat pada tabel 5.24.

Tabel 5.24 Skenario Mengeset Status Lampu

Nama	Menegeset status lampu
Tujuan Uji Coba	Menguji fungsionalitas menyimpan status lampu
Kondisi Awal	Data lampu belum diperbarui
Skenario	Arduino mengirim perintah status lampu dan disimpan
Masukan	Instruksi tombol
Keluaran	Data lampu sudah diperbarui
Hasil Pengujian	Berhasil

5.4.8. Skenario Mendapatkan Status Lampu

Pada skenario mendapatkan status lampu dilakukan dengan menampilkan status lampu ke halaman web. Hasil uji coba mendapatkan status lampu dapat dilihat pada tabel 5.25.

Tabel 5.25 Skenario Mendapatkan Status Lampu

Nama	Mendapatkan status lampu
Tujuan Uji Coba	Menguji fungsionalitas melihat status lampu
Kondisi Awal	Data lampu belum ditampilkan
Skenario	ESP8266 menampilkan data lampu ke staff pengawasan dan pengendalian
Masukan	Data lampu
Keluaran	Data lampu sudah ditampilkan
Hasil Pengujian	Berhasil

5.4.9. Skenario Mengeset Data Pemakaian Listrik Lampu

Pada skenario mengeset data pemakaian listrik lampu dilakukan dengan mengirim status pemakaian listrik lampu ke ESP8266 dan disimpan. Hasil uji coba mengeset data pemakaian listrik lampu dapat dilihat pada table 5.26.

Tabel 5.26 Skenario Mengeset Data Pemakaian Listrik Lampu

Nama	Mengeset data pemakaian listrik lampu
Tujuan Uji Coba	Menguji fungsionalitas menyimpan data pemakaian listrik lampu
Kondisi Awal	Data pemakaian listrik lampu belum diset
Skenario	Arduino mengirim perintah data pemakaian listrik dan disimpan
Masukan	Instruksi dan data pemakaian listrik lampu
Keluaran	data pemakaian listrik lampu sudah diset
Hasil Pengujian	Belum berhasil

5.4.10. Skenario Mengupdate Data Pemakaian Listrik

Pada skenario mengupdate data pemakaian listrik dilakukan dengan mengirim trigger update data pemakaian listrik

lampu ke ESP8266 dan mengupdate data pemakaian listrik. Hasil uji coba mengupdate data pemakaian listrik dapat dilihat pada table 5.27.

Tabel 5.27 Skenario Mengupdate Data Pemakaian Listrik

Nama	Mengupdate data pemakaian listrik
Tujuan Uji Coba	Menguji fungsionalitas mengupdate data pemakaian listrik
Kondisi Awal	Data pemakaian listrik belum diupdate
Skenario	Arduino mengirim perintah update data pemakaian listrik dan melakukan update
Masukan	Instruksi update pemakaian listrik lampu
Keluaran	data pemakaian listrik sudah diupdate
Hasil Pengujian	berhasil

5.4.11. Skenario Mereset Data Pemakaian Listrik Tahun

Pada skenario mereset data pemakaian listrik tahun dilakukan dengan mengirim trigger reset pemakaian listrik tahun ke ESP8266 dan mereset pemakaian listrik tahun. Hasil uji coba mereset data pemakaian listrik tahun dapat dilihat pada table 5.28.

Tabel 5.28 Skenario Mereset Data Pemakaian Listrik

Nama	Menegeset data pemakaian listrik lampu
Tujuan Uji Coba	Menguji fungsionalitas menyimpan data pemakaian listrik tahun
Kondisi Awal	Data pemakaian listrik tahun belum direset
Skenario	Arduino mengirim perintah reset data pemakaian listrik dan mereset data pemakaian listrik
Masukan	Instruksi reset pemakaian listrik tahun
Keluaran	data pemakaian listrik tahun sudah direset
Hasil Pengujian	Belum dilakukan

5.4.12. Skenario Mendapatkan Data Pemakaian Listrik

Pada skenario mendapatkan data pemakaian listrik dilakukan dengan menampilkan data pemakaian listrik ke halaman web. Hasil uji coba mendapatkan data pemakaian listrik dapat dilihat pada tabel 5.29.

Tabel 5.29 Skenario Mendapatkan Data Pemakaian Listrik

Nama	Mendapatkan data pemakaian listrik
Tujuan Uji Coba	Menguji fungsionalitas melihat data pemakaian listrik
Kondisi Awal	data pemakaian listrik belum ditampilkan
Skenario	ESP8266 menampilkan data pemakaian listrik ke staff pengawasan dan pengendalian
Masukan	data pemakaian listrik
Keluaran	data pemakaian listrik sudah ditampilkan
Hasil Pengujian	Berhasil

[halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan Tugas Akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari hasil perancangan, implementasi dan pengujian maka dapat diambil kesimpulan sebagai berikut :

1. Sistem pengendalian sudah berhasil diimplementasikan
2. ADSC dan ADPS mempengaruhi kecepatan membaca dan keakuratan sensor karena membaca data dari setiap analog read secara bergantian.
3. Setiap analog read memakan waktu 112 mikro second yang membuat Arduino lambat

6.2. Saran

Berikut saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Diantara saran sebagai berikut :

1. Sistem Pengawasan perlu dikembangkan lagi.
2. Apabila biaya operasional mikrokontroller lebih hemat daripada biaya akibat ketidakakuratan data karena ADSC, maka penggunaan lebih banyak mikrokontroller atau mikrokontroler dengan clock frequency lebih tinggi dapat dipertimbangkan.

DAFTAR PUSTAKA

[1] "Arduino Mega 2560 Rev 3," Arduino USA store, [Online]. Available: https://playground.arduino.cc/Main/ArduinoPinCurrentLimitations. [Accessed 6 January 2019].

[2] "Arduino Pin Current Limitation," Arduino Playground, [Online]. Available: https://playground.arduino.cc/Main/ArduinoPinCurrentLimitations. [Accessed 6 January 2019].

[3] sinauarduino, "Mengenal Arduino Software (IDE)," 16 March 2016. [Online]. Available: https://www.sinauarduino.com/artikel/mengenal-arduino-software-ide/. [Accessed 14 January 2019].

[4] "ADC Tutorial: Analog to Digital Conversion," Robot Platform, [Online]. Available: http://www.robotplatform.com/knowledge/ADC/adc_tutorial.html. [Accessed 6 January 2019].

[5] "ADC in AtmeI Microcontrollers (using Atmega8)," Robot Platform, [Online]. Available: http://www.robotplatform.com/knowledge/ADC/adc_tutorial_2.html. [Accessed 6 January 2019].

[6] "ADMUX - ADC Multiplexer Selection Register," Robot Platform, [Online]. Available: http://www.robotplatform.com/knowledge/ADC/adc_tutorial_3.html. [Accessed 6 January 2019].

[7] R-B, "a Brief Overview of Allegro ACS712 Current Sensor (part 1)," Embedded Lab, 25 January 2012. [Online]. Available: http://embedded-lab.com/blog/a-brief-overview-of-allegro-acs712-current-sensor-part-1/. [Accessed 6 January 2019].

[8] B. Y. Yoga, "Monitoring Arus Beban yang Tersalurkan pada Gardu Induk PLTU Gresik dengan Android

Menggunakan Bluetooth HC-05 Berbasis Mikrokontroller ARM," *JIPPTUMG*, 2017.

[9] A. Tarun, "Know All About Light Sensor," 2016. [Online]. Available: https://www.elprocus.com/light-sensor-circuit-with-applications/. [Accessed 6 January 2019].

[10] SinauArduino, "Modul Wifi ESP8266," 6 April 2016. [Online]. Available: http://www.sinauarduino.com/artikel/esp8266/. [Accessed 6 January 2019].

[11] M. Sandeep, FWeinb, N. Peeter, M. Pablo, pnewb and Kosh, "NTPClient," Github, 28 June 2018. [Online]. Available: https://github.com/arduino-libraries/NTPClient/blob/master/NTPClient.cpp. [Accessed 6 January 2019].

[12] K. Dickson, "Pengertian Relay dan Fungsinya," Teknik Elektronika, [Online]. Available: https://teknikelektronika.com/pengertian-relay-fungsi-relay/. [Accessed 6 January 2019].

[13] A. W. Faiz, "Pengertian HTML, Fungsi HTML serta Sejarah HTML," May 2015. [Online]. Available: https://www.burung-net.com/2015/05/pengertian-html-fungsi-html-serta-sejarah-html.html. [Accessed 22 January 2019].

[14] "Arduino Time Sync From NTP Server using ESP8266 WiFi Module," geekstips, [Online]. Available: https://www.geekstips.com/arduino-time-sync-ntp-server-esp8266-udp/. [Accessed 6 January 2019].

[15] Matt, "Making Accurate ADC Readings on the Arduino," Majenko Technology, 1 February 2016. [Online]. Available: https://majenko.co.uk/blog/making-accurate-adc-readings-arduino. [Accessed 6 January 2019].

[16] Allegro Microsystems, LCC, ACS712 Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 2.1

kVRMS Isolation and a Low-Resistance Current Conductor, Massachussets: Allegro Microsystems, LCC.

[17] R. P. Lintang, H. Hardi, Fatimah and Hendro, "Detektor Ketebalan Kabut/Asap Berbasis Arduino Uno sebagai Antisipasi Terjadi Kecelakaan di Jalan Raya," in *Prosiding Simposium Nasional Inovasi dan Pembelajaran Sains 2015 (SNIPS2015)*, Bandung, 2015.

LAMPIRAN

Tabel 7.1 Rincian Kode Arduino

```
#include<SoftwareSerial.h>
#include<math.h>
float vall,ia,il;
float aver=0.0;
int aves=0;
int kl=0,i,l=0,mo,da,y,sw,st;
float Vcc;
bool ly,HD=false;
unsigned long pMill=0,pMill2=0,k1,E1;
String E2;
int m[12]={31,28,31,30,31,30,31,31,30,31,30,31};
class NS {
  private:
    int _mo,_y,_da,_s,_m,_h;
public:
//438
//1817
  long readVcc() {
    long result;
    // Read 1.1V reference against AVcc
    ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) |
_BV(MUX2) | _BV(MUX1);
    delay(2);
    ADCSRA |= _BV(ADSC);
    while (bit_is_set(ADCSRA,ADSC));
    result = ADCL;
    result |= ADCH<<8;
    result = 1125300L / result;
    return result;
  }
  int sY(unsigned long d){
    d-=978307200L;
    d/=86400L;
```

```
    y=2001;
    while(d>=146097){
      d-=146097;
      y+=400;
    }
    while(d>=36524){
      d-=36524;
      y+=100;
    }
    while(d>=1461){
      d-=1461;
      y+=4;
    }
    while(d>=365){
      d-=365;
      y++;
    }
    if((y%4==0&&y%100!=0)||y%400==0)ly=true;
    else ly=false;
    if(this->_y<y)this->SeVal(5);
    this->_y=y;
    i=0;

while((i==1&&ly==true&&d>=29)||(d>=m[i]&&(ly==false||(i!
=1&&ly==true)))){
      if(i==1&&ly==true)d-=29;
      else d-=m[i];
      i++;
    }
    if(this->_mo<i)this->SeVal(4);
      this->_mo=i;
    if(this->_da<d){
      if(d==1&&this->_mo==0)this->SeVal(6);
      this->SeVal(3);
    }
```

```
    this->_da=d;
  }
  void sC(unsigned int d){
    int h,m,s;
    h=(d%86400L)/3600;
    if(this->_h<h)this->SeVal(2);
      this->_h=h;
    m=(d%3600)/60;
    if(this->_m<m)this->SeVal(1);
      this->_m=m;
    s=d%60;
    if(this->_s<s)this->SeVal(0);
      this->_s=s;
    pMill=millis();

  }
  void sE(unsigned long d){
    unsigned long eo=d;
    sY(d);
    sC(eo);
  }
  bool cht()const{
    int h,m;
    h=this->_h;
    m=this->_m;
    if(h<4||(h==4&&m<=38)||h>18||(h==18&&m>=17))return
false;
    else return true;
  }
  void SeVal(int d){
    if(d==0){
      E2=char(5);
      unsigned long E=vall;
     while(E>0){
      E1=E%113;
```

```
    E1+=14;
    if(E1>=126)E1++;
    E/=113;
    E2+=char(E1);
   }
    Serial2.println(E2);
      vall=0.0;
   }
   else if(d>0&&d<=6){
    d+=5;
    if(d>=10)d++;
    Serial2.println(char(d));
   }
}
void Atm(unsigned long d){
  pMill=millis();
  this->_s+=d;
  this->SeVal(0);
  if(this->_s>59){
    this->SeVal(1);
    this->_m+=(this->_s/60);
    this->_s%=60;
  }
  if(this->_m>59){
    this->SeVal(2);
    this->_h+=1;
    this->_m=0;
  }
  if(this->_h>23){
    if(this->_da==0&&this->_mo==0)this->SeVal(6);
    this->SeVal(3);
    this->_da+=1;
    this->_h=0;
  }
  mo=this->_mo;
```

```
    da=this->_da;

if((i==1&&ly==true&&da>=29)||(da>=m[i]&&(ly==false||(i!=
1&&ly==true)))){
      this->SeVal(4);
      this->_mo+=1;
      this->_da=0;
    }
    if(this->_m>=12){
      this->SeVal(5);
      this->_y+=1;
      this->_m=0;
    }
    y=this->_y;
    if((y%4==0&&y%100!=0)||y%400==0)ly=true;
    else ly==false;
  }
  void motife(char s){
    if(s>=0&&s<3){
      sw=s%3;
      if(s%3==0)digitalWrite(6,LOW);
      else digitalWrite(6,HIGH);
    }
    else if(s==3)HD=true;
    else if(s==4){
      this->SeVal(0);
    }
  }
  void LamEv(){
      kl=sw;
      Serial2.println(char(kl));
      kl=3;
      if((aver/float(aves))>=0.03)kl++;
      Serial2.println(char(kl));
      aver=0.0;
```

```
    aves=0;
  }
  void SerEv(){
   while(Serial2.available()){
    char inCh=Serial2.read();
    if(inCh==126)k1=0;
    else if(inCh==13||inCh==10){
     if(HD==true)this->sE(k1);
     HD=false;
     k1=l=0;
    }
    else if(inCh>=0&&inCh<5){
     this->motife(inCh);
    }
    else if(HD==true){
     if(inCh>120)inCh--;
      if(E1>=126)inCh--;
      if(E1>=13)inCh--;
      if(E1>=10)inCh--;
     inCh-=5;
     k1+=(unsigned long)inCh*(120^l);
     l++;
    }
   }
  }
};
NS ac;
void setup() {
  Serial2.begin(9600);
  DDRH|=B01111000;
}
void loop() {
  ac.SerEv();
  if(millis() - pMill > 1000) ac.Atm((millis()-pMill)/1000);
  if(millis() - pMill2 > 100){
```

```
    ac.LamEv();
    pMill2=millis();
  }
  Vcc = ac.readVcc()/1000.0;
  long average = 0;
  for (i = 0; i < 10; i++) {
  average = average + analogRead(A1);
  }
  average = average / 10;
  float acofs=(((Vcc*average)/1023.0)-(Vcc/2.0))/0.185;
  vall=acofs;
  aver+=average;
  aves++;
  if(sw==1){
  il = 0;
  for (i = 0; i < 10; i++) {
  il = il + analogRead(A0);
  }
  if(il<1018||ac.cht()==false)digitalWrite(6,HIGH);
  else if(il>=1018&&ac.cht()==true)digitalWrite(6,LOW);
  }
}
```

Tabel 7.2 Rincian Kode ESP8266

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <WiFiUdp.h>
#include <Udp.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266mDNS.h>
#include <ESP8266WebServer.h>
#include<SoftwareSerial.h>
#define NTP_DEF_LP 1337
ESP8266WiFiMulti wifiMulti;
```

```
ESP8266WebServer server(80);
WiFiClient espClient;
WiFiUDP ntpUDP;
String k,E2,iStr;
int i,j,l=0,p0=0;
boolean strC,prvD,HC;
String r[3]={"o","a","i"};
String sw[3]= {"OFF", "AUTO", "ON"};
String ib[2]={"OFF","ON"};
String T[6]={"1 DETIK","1 MENIT","1 JAM","1 HARI","1
BULAN","1 TAHUN"};
int io,pos;
char p;
unsigned long E,E1,pMill2=0;
float W[9],W1[9];
class NTPC {
  private:
    UDP*          _udp;
    const char*   _pSN  = "2.id.pool.ntp.org";
    int           _port = NTP_DEF_LP;
    long          _toff = 25200;
    unsigned long _ui   = 1200;  // In ms
    unsigned long _cE   = 0;     // In s
    unsigned long _lu   = 0;     // In ms
    byte          _pBuff[48];
    void          sendNTPPacket(){
      memset(this->_pBuff, 0, 48);
      this->_pBuff[0]  = 0b11100011;   // LI, Version, Mode
      this->_pBuff[1]  = 0;     // Stratum, or type of clock
      this->_pBuff[2]  = 6;     // Polling Interval
      this->_pBuff[3]  = 0xEC;
      this->_pBuff[12]  = 49;
      this->_pBuff[13]  = 0x4E;
      this->_pBuff[14]  = 49;
      this->_pBuff[15]  = 52;
```

```
    this->_udp->beginPacket(this->_pSN, 123);
    this->_udp->write(this->_pBuff, 48);
    this->_udp->endPacket();
    }
  public:
   NTPC(UDP& udp){
    this->_udp=&udp;
   }
   void begin() {
    this->_port = NTP_DEF_LP;
    this->_udp->begin(this->_port);
   }
   bool update() {
    if ((millis() - this->_lu >= this->_ui)|| this->_lu == 0){
      #ifdef DEBUG_NTPClient
        Serial.println("Update from NTP Server");
      #endif
      this->sendNTPPacket();
      byte timeout = 0;
      int cb = 0;
      do {
       delay ( 10 );
       cb = this->_udp->parsePacket();
       if (timeout > 100) return false;
       timeout++;
      } while (cb == 0);
      this->_lu = millis() - (10 * (timeout + 1)); // Account for
delay in reading the time
      this->_udp->read(this->_pBuff, 48);
      unsigned long highWord = word(this->_pBuff[40], this-
>_pBuff[41]);
      unsigned long lowWord = word(this->_pBuff[42], this-
>_pBuff[43]);
      unsigned long s1900 = highWord << 16 | lowWord;
      this->_cE = s1900 - 2208988800UL;
```

```
      return true;
    }
  }
  unsigned long gET() const {
    return this->_toff + this->_cE + ((millis() - this->_lu) /
1000);
  }
  void motife(char s){
    if(s>=0&&s<3){
      pos=s%3;
    }
    else if(s>=3&&s<5){
      io=s/4;
    }
    else if(s==5){
      prvD=true;
    }
    else if(s>=6&&s<11){
      s-=5;
      W[s]+=W[s-1];
      W1[s]= W[s-1]*1.46728;
      W[s-1]=0;
    }
    else if(s==12){
      W[5]=0;
      W1[5]=0;
    }
    HC=true;
  }
  void SerEv(){
    while(Serial.available()){
      char inChar=(char)Serial.read();
      if(inChar==13||inChar==10){
        if(prvD==true){
          W[p]=float(p0)/100.0;
```

```
          W1[p]= W[p]*1.46728;
         }
         HC=false;
         p0=l=0;
       }
       else if(inChar>=0&&inChar<13)this->motife(inChar);
       else if(HC==true){
         if(inChar>126)inChar--;
         inChar-=14;
          p0+=(unsigned long)inChar*(113^l);
          l++;
       }
     }
  }
};
void hLDo() {
  Serial.println(char(0));
  server.sendHeader("Location","/");
  server.send(303);
}
void hLDa() {
  Serial.println(char(1));
  server.sendHeader("Location","/");
  server.send(303);
}
void hLDi() {
  Serial.println(char(2));
  server.sendHeader("Location","/");
  server.send(303);
}
void handleRoot() {
k="";
k+="<!DOCTYPE html><html><head><title>Lampu
PJU</title><meta charset=\"utf-8\">";
```

```
k+="<meta name=\"viewport\" content=\"width=device-width,
initial-scale=1\">";
k+="<link rel=\"stylesheet\"
href=\"https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bo
otstrap.min.css\">";
k+="<link
href=\"https://fonts.googleapis.com/css?family=Alegreya:800|
Eczar:800|Bitter:700\" rel=\"stylesheet\">";
k+="<style>h2{font-family: 'Alegreya', serif; background-
color: #FFA631;}";
k+="h5{font-family:'Eczar',serif;color:#4B3C39;font-size:
18px;}div.aa{padding-top:30px;padding-bottom:30px;}";
k+="a.c{background-color:#317589;}a.c:hover{background-
color:#2165A9;}a.b{background-color:#351E1C;}";
k+="a.b:hover{background-
color:#550E0C;}a,a:hover{color:white}p.c{color:#DC3023;}p.
b{color:#6B9B62;}";
k+="div.a{padding-top: 40px;}thead{background-
color:#351E1C;color:white;}";
k+="body{font-family: 'Bitter', serif;background-
color:#EFEFEF;}</style></head><body>";
k+="<div class=\"a\" align=\"middle\"><h2 style=\"font-
size:1.8em\">";
k+="SISTEM PEMANTAUAN DAN PENGENDALIAN
PENERANGAN JALAN UMUM</h2><div class=\"row\">";
  k+="<div class=\"aa col-sm-6 col-lg\"><h5>STATUS
TOMBOL</h5><p class=\"";
  if(io==0)k+="c\">OFF";
  else k+="b\">ON";
  k+="</p>";
  for(i=0;i<3;i++){
    k+="<a href=\"/LD"+r[i]+"\" class=\"btn ";
    if(pos==i)k+="c";
    else k+="b";
    k+="\">"+sw[i]+"</a>";
```

```
    if(i<2)k+=" ";
  }
  k+="</div>";
k+="</div></div><table class=\"container table table-
hover\"><thead><tr><td>WAKTU</td><td>KWH</td><td>B
IAYA</td>";
k+="</tr></thead><tbody>";
for(i=0;i<6;i++){
k+="<tr><td>"+String(T[i])+"</td><td>"+String(W[i])+"</td>
<td>"+String(W1[i])+"</td></tr>";
}
k+="</tbody></table></div></body></html>";
server.send(200, "text/html", k);
}
void hNF(){
  server.send(404, "text/plain", "404: Not found");
}
NTPC tcli(ntpUDP);
void setup(void){
  Serial.begin(9600);
  IPAddress ip(10, 151, 253, 253);
  IPAddress gateway(10, 151, 252, 1);
  IPAddress subnet(255, 255, 252, 0);
  IPAddress dns(10, 151, 40, 90);

// Static IP Setup Info Here...
WiFi.config(ip, dns, gateway, subnet); //If you need Internet
Access You should Add DNS also...
WiFi.begin("Informatics_Wifi", "");
wifiMulti.addAP("Informatics_Wifi", "");
  while (wifiMulti.run() != WL_CONNECTED)delay(250);
  server.on("/",  handleRoot);
  server.on("/LDo",  hLDo);
  server.on("/LDa",  hLDa);
  server.on("/LDi",  hLDi);
```

```
  server.onNotFound(hNF);
  server.begin();
  tcli.begin();
  Serial.println(char(4));
}
void loop(void){
tcli.update();
tcli.SerEv();
E=tcli.gET();
if(millis() - pMill2 > 100){
  E2=char(3);
  while(E>0){
    E1=E%120;
    E1+=5;
    if(E1>=10)E1++;
    if(E1>=13)E1++;
    if(E1>=126)E1++;
    E/=120;
    E2+=char(E1);
  }
  Serial.println(E2);
  pMill2=millis();
}
server.handleClient();              // Listen for HTTP requests
from clients
};
```

BIODATA PENULIS

Penulis dilahirkan di Surabaya, 20 April 1995 dan hanya memiliki 1 saudara kandung.

Penulis pernah bersekolah di SD Giki 2 Surabaya (2001-2007), SMP Negeri 6 Surabaya (2007-2010), dan SMA Negeri 5 Surabaya (2010-2013). Setelah lulus SMA dilanjutkan ke jenjang perkuliahan di Jurusan Informatika, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya.

Bidang Studi yang diambil oleh penulis pada saat kuliah di Informatika ITS adalah Analisis Jaringan Komputer, Komputasi Berbasi Jaringan, Komputasi Cerdas Visual dan Rekayasa Perangkat Lunak.

Selama menempuh kuliah penulis mengikuti kegiatan pengembangan kepribadian baik dari institut, fakultas, dan jurusan. Penulis pernah menjadi bagian dari schematics 2013-2014 dan menjadi bagian dari Keluarga Muslim Informatika pada tahun yang sama. Penulis dapat dihubungi di andaru.kharisma@gmail.com.