p-ISSN : 2301-8364; e-ISSN : 2685-6131, available at : https://ejournal.unsrat.ac.id/index.php/informatika

Internet of Things Prototype House Light Remote Control using Contrained Application Protocol

Prototipe Aplikasi *Internet of Things* Kendali Jarak Jauh Lampu Rumah menggunakan *Contrained Application Protocol*

Rangga D. P. Firmansyah, Xaverius B. N. Najoan, Sherwin R. U. A. Sompie
Jurusan Teknik Elektro, Universitas Sam Ratulangi Manado, Jl. Kampus Bahu, 95115, Indonesia
e-mail : 15021106008@student.unsrat.ac.id , xnajoan@unsrat.ac.id, aldo@unsrat.ac.id
Received : 13 October 2020 ; revised : 10 December 2020 ; accepted: 15 January 2021

***Abstrak* — Dalam perkembangan teknologi perangkat cerdas pada saat ini, kecerdasan buatan sangatlah dibutuhkan untuk membantu kegiatan sehari-hari. Teknologi perangkat cerdas saat ini sudah sangatlah berkembang dimana - mana, dan sudah banyak istilah-istilah seperti *smart-city , smart-home,* dan lain-lainnya. Penelitian ini bertujuan untuk membangun Prototipe Aplikasi IoT Kendali Jarak Jauh Lampu Rumah menggunakan Protokol CoAP (*Contrained Application Protocol)*. Metodologi yang digunakan dalam mengembangkan aplikasi adalah metodologi prototipe, yang prosesnya dimulai dari tinjauan pustaka, pengumpulan kebutuhan, membangun prototipe, mengkodekan sistem, pengujian sistem, evaluasi sistem, dan menggunakan sistem. Penelitian telah berhasil dibuat dengan menggunakan *libraries* microcoap dan dapat dijalankan menggunakan ekstensi COPPER CU yang di *install* manual di dalam browser *Chromium*.**

***Kata kunci* — *smart-city*; *smart-home*; Protokol CoAP; COPPER CU; *chromium*; Arduino Mega 2560; *Ethernet Shield*; *Internet of Things*.**

Abstract - In the current development of smart device technology, artificial intelligence is needed to help with daily activities. Smart device technology is currently very developed everywhere, and there are many terms such as smart-city, smart-home, and others. This study aims to build a Home Light Remote Control IoT Application Prototype using the CoAP Protocol. The methodology used in developing the application is the prototype methodology, whose process starts from literature review, needs gathering, building prototypes, coding the system, testing the system, evaluating the system, and using the system. The research has been successfully made using microcoap libraries and can be run using the COPPER CU extension which is manually installed in the Chromium browser.

***Keywords* — *smart-city*; *smart-home*; CoAP Protocol; COPPER CU; *chromium*. Arduino Mega 2560; *Ethernet Shield*; *Internet of Things*.**

I. PENDAHULUAN

Dalam perkembangan teknologi perangkat cerdas pada saat ini, kecerdasan buatan sangatlah dibutuhkan untuk membantu kegiatan sehari-hari. Teknologi perangkat cerdas saat ini sudah sangatlah berkembang dimana - mana, dan sudah banyak istilah-istilah seperti *smart-city , smart-home,* dan lain-lainnya.Teknologi perangkat cerdas sendiri memiliki pengertian sebagai suatu ilmu keteknikan dimana ilmu yang mempelajari untuk membuat suatu alat yang sudah di automasi atau dapat dikendalikan melalui aplikasi.

Penerangan lampu di rumah biasanya diatur secara manual (tidak otomatis). Terkadang, disaat kita akan berpergian keluar kota, atau kemana pun, kita lupa untuk mematikan lampu rumah kita atau disaat malam kita sulit untuk menyalakan penerangan di rumah kita.

CoAP adalah protokol layer aplikasi yang dikembangkan oleh *International Engineering Task Force*(IETF) .Termasuk dalam standar RFC 7252, CoAP sendiri merupakan singkatan dari *Contrained Application Protocol*. Protokol ini biasanya dibangun untuk aplikasi M2M (*Machine-to-Machine*) seperti halnya *smart energy* dan automasi-automasi bangunan lainnya.

Dengan adanya penelitian dan aplikasi ini penulis berharap dapatmembantu pengguna dalam pemakaian listrik terlebihnya dalam hal penereangan rumah secara maksimal mungkin dan tidak akan ada ragu lagi disaat berpergian ke luar kota soal penerangan lampu di rumah dan pemakaian listrik rumah pengguna.[1]–[3]

II. METODE

Metode yang digunakan dalam pembuatan alat adalah metode Prototipe, dimana penelitian ini dilakukan terbatas prototipe yang dimulai dalam, pengumpulan kebutuhan, membangun prototipe, mengkodekan sistem, pengujian sistem, evaluasi sistem, dan menggunakan sistem.[4], [5]

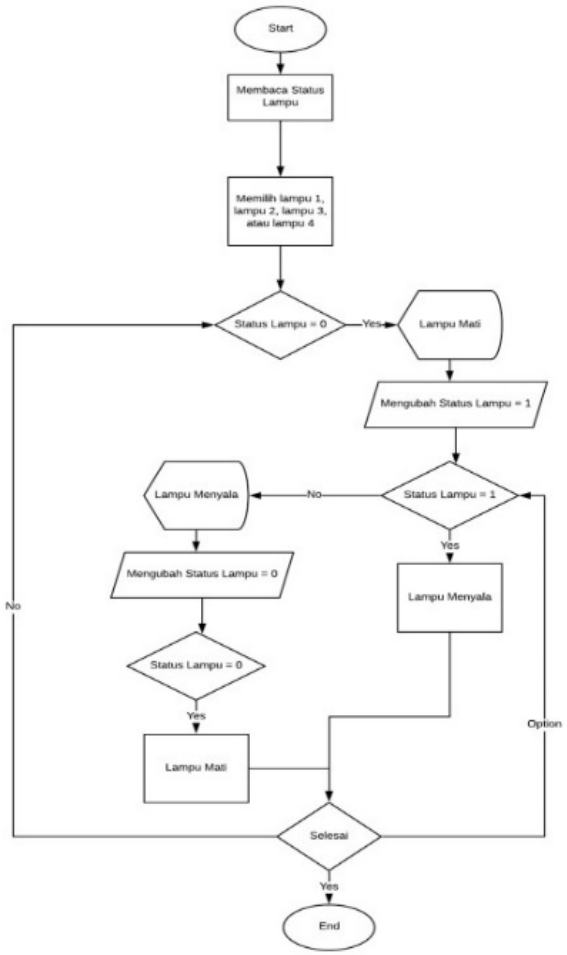
1) Tinjauan Pustaka

Pada Tahap ini peneliti mencari berbagai literatur untuk mendukung ide dalam pembuatan sistem yang diambil dari berbagai sumber, seperti *e-Book*, *Paper*, dan Jurnal terkait sistem yang akan dibuat serta mempelajari jenis *Software* dan *Hardware* yang di perlukan dalam pembuatan sistem ini dan bagaimana membuat *Algoritma* untuk pengendalian perangkat listrik melalui Jaringan *Internet*.[5], [6]

2) Pengumpulan Kebutuhan

Pada tahap pengumpulan kebutuhan penelitian ini, peneliti melakukan pengumpulan kebutuhan – kebutuhan berupa alat dan bahan yang akan di gunakan untuk membangun prototipe dan sistem.

Pada gambar 1 adalah *flowchart sistem* yang akan digunakan pada penelitian ini. Dimana disaat memulai sistem maka akan dibaca dahulu status atau kondisi lampu awal, lalu memilih



Gambar 1. *Flowchart*

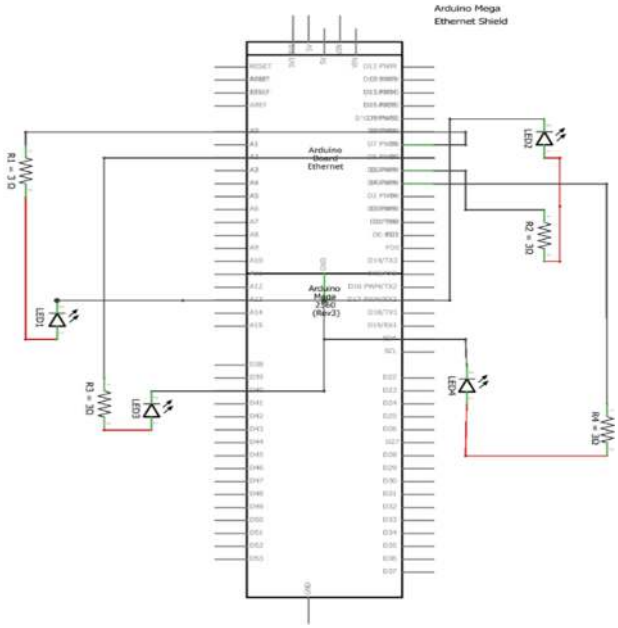
lampu berapa yang akan di kendalikan, jikalau sudah memilih lampu, sistem akan membaca kembali status lampu dengan kondisi 0 jika iya maka lampu dalam kondisi mati, jika *user* ingin mengganti status lampu maka akan memasukan nilai satu untuk mengubah kondisi lampu menjadi menyala, dan jikalau masih mau dilakukan sistem akan berulang, jikalau tidak makan sistem akan selesai. [7]

3) Membangun Prototipe

Pada tahap selanjutnya, pada penelitian ini adalah dengan membangun prototipe dengan membuat model maket rumah dan merakit *wiring* untuk digunakan dan diterapkan pada sistem yang akan dibangun.

4) Mengkodekan Sistem

Pada Tahap selanjutnya, pada penelitian ini peneliti melakukan proses *coding* (mengkodekan sistem) dengan bahasa pemograman untuk menjalankan sistem, yang dimana dalam penelitian ini menggunakan Bahasa pemograman C, menggunakan Arduino IDE dan mengkodekan *endpoints* dalam Bahasa C menggunakan *Visual Studio Code* untuk menjalankan Arduino Mega 2560 + Ethernet Shield dan CoAP.



Gambar 3. Perencanaan Wiring Prototipe

Pada gambar 3 adalah perencanaan *wiring* yang akan digunakan pada prototipe, dimana menggunakan Arduino Mega sebagai mikrokontroller dan Ethernet Shield untuk koneksi internet yang digunakan, lampu atau *led* terhubung dengan Arduino Mega melalui pin D4, D5, D6, dan D7 dengan resistor 3 Ω untuk masing masing *led*, dan juga *Ground*.[10]

5) Pengujian Sistem

Tahap selanjutnya, pada penelitian ini adalah melakukan pengujian sistem yang telah di bangun sebelumnya, apakah masih ada kesalahan atau tidak, dalam pengujian sistem ini menggunakan browser *Chromium* dan menggunakan *Extensions* COPPER CU.

6) Evaluasi Sistem

Tahap selanjutnya, pada penelitian ini adalah melakukan evaluasi sistem, dimana dalam proses ini dilihat dan dievaluasi kembali prototipe dan sistem yang sudah dirancang dan dibuat oleh peneliti jikalau masih memiliki kekurangan atau sudah selesai.

7) Menggunakan Sistem

Tahap selanjutnya, pada penelitian ini adalah menggunakan sistem yang telah dibuat dan telah dievalua

Pada gambar 4 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.

Pada gambar 5 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.

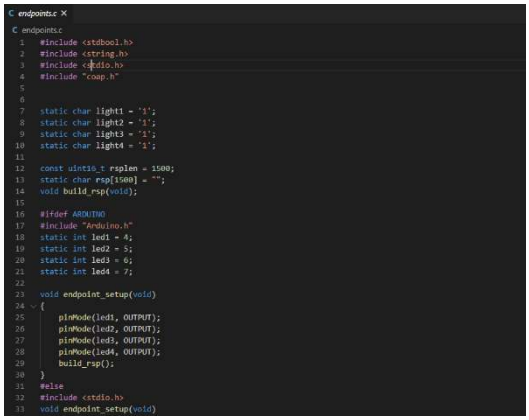
Pada gambar 6 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.

Pada gambar 7 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.

Pada gambar 8 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.

Pada gambar 9 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.

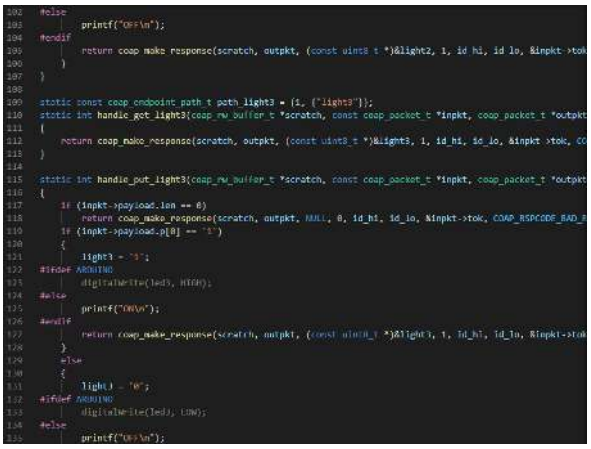
Pada gambar 10 adalah kode program *endpoints*.c yang digunakan untuk fungsi utama dari fitur utama.



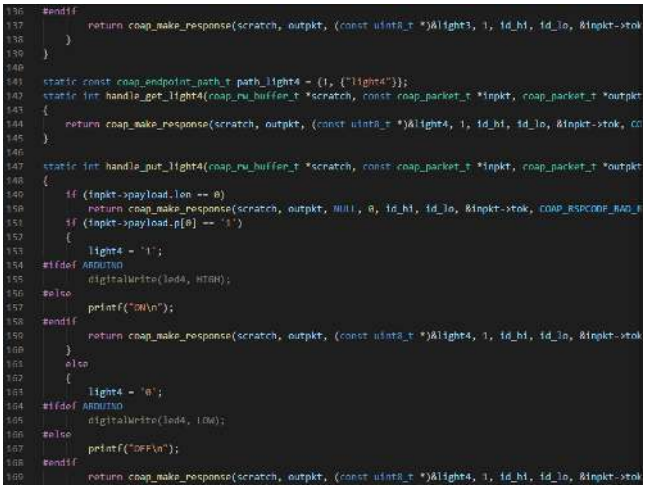
Gambar 4. Program *endpoints* (a)

Gambar 5. Program *endpoints* (b)

Gambar 6. Program *endpoints* (c)

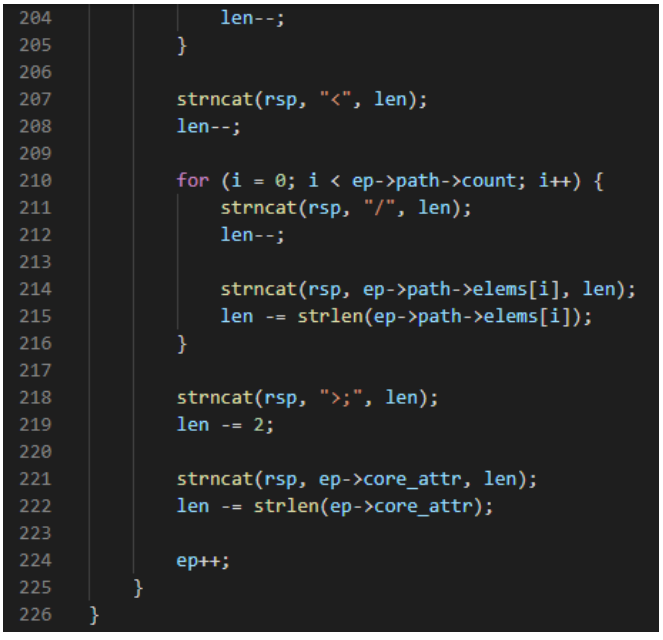


Gambar 7. Program *endpoints* (d)



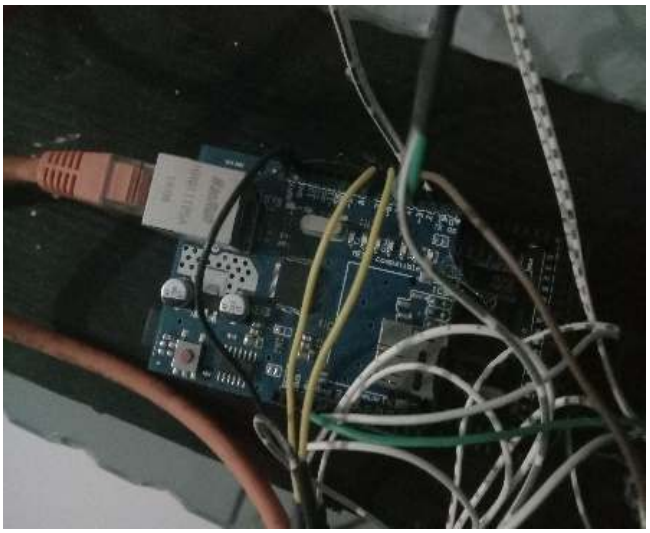
Gambar 8. Program *endpoints* (e)

Gambar 9. Program *endpoints* (f)



Gambar 10. Program *endpoints* (g)

Gambar 11. Hasil *Wiring* dengan menggunakan Arduino Mega 2560 + Ethernet Shield pada Prototipe yang dibuat (1)



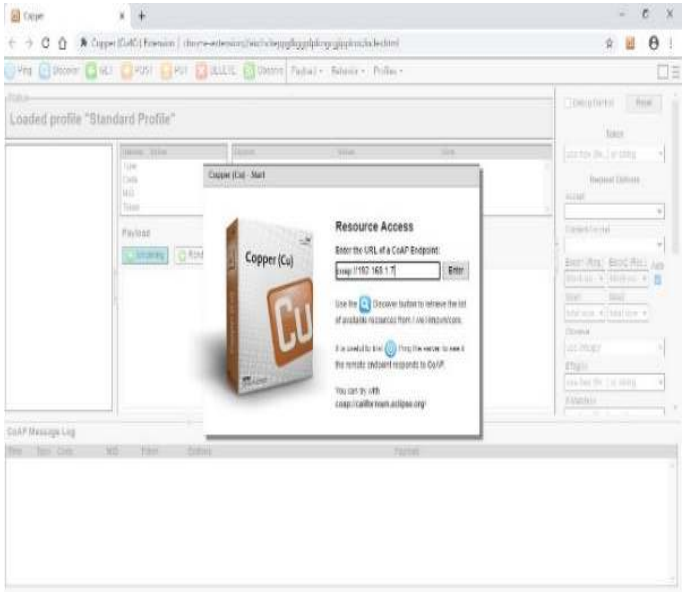
Gambar 12. Hasil *Wiring* dengan menggunakan Arduino Mega 2560 + Ethernet Shield pada Prototipe yang dibuat (2)

III. HASIL DAN PEMBAHASAN

1) Hardware

Hasil dari penelitian dalam bentuk *Hardaware* adalah sebuah Prototipe Aplikasi IoT Kendali Jarak Jauh Lampu Rumah menggunakan Protokol CoAP, dengan menggunakan *Arduino Mega 2560* sebagai pusat kontrol perangkat listrik yang menerima dan meneruskan perintah ke *LED* untuk merubah kondisi menyala atau mati.[7], [9]

Pada gambar 11 dan gambar 12 adalah gambar rangkaian dari *Arduino Mega 2560 + Ethernet Shield* yang dipasang pada prototipe, dimana terdapat kabel yang terhubung dengan 4 buah *LED* yang sudah dipasang pada maket untuk dikendalikan langsung dari *Arduino Mega 2560* melalui sistem yang dipasang.



Gambar 13. Tampilan COPPER CU Halaman Awal untuk mengisi *IP ADDRESS*

2) Software

Dalam pembuatan *interface* sistem peniliti menggunakan *browser Chromium* dengan menggunakan *extensions* COPPER CU ini terdapat halaman utama yang berisi tampilan *realtime* dan sebuah tombol untuk masing masing led.

a. Halaman Awal COPPER CU

Pada halaman awal *COPPER CU* adalah halaman yang oertama kali di lihat oleh *user* saat mengakses sistem ini. Halaman ini adalah halaman berfungsi untuk dimana user mengisi *IP Address* yang tertera pada mikrokontroller. Pada gambar13 dapat dilihat tampilan dan halaman awal dari *COPPER CU*.

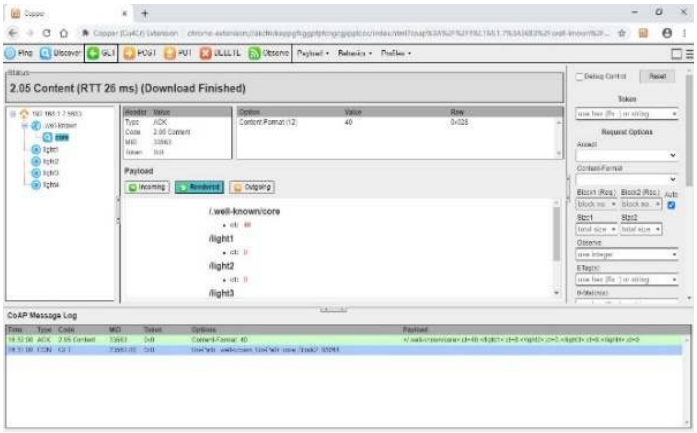
b. Halaman Awal COPPER CU setelah memasukkan IP Address

Pada halaman ini adalah halaman yang akan tampil setelah user mengisikan *IP Address* yang sudah tertera pada mikrokontroller. Pada halaman ini tertera beberapa data lampu yang sudah di sediakan peneliti untuk dapat di kendalikan menggunakan halaman ini. Pada Gambar 14 dapat dilihat tampilan *COPPER CU* yang sudah dipanggil fitur menyalakan dan mematikan lampu.

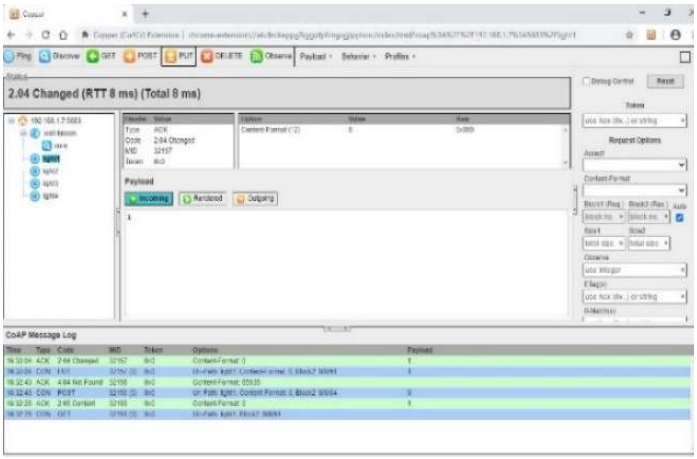
c. Halaman lampu yang tersedia

Pada halaman ini adalah halaman yang akan tampil setelah user mengisikan memilih lampu mana yang di kendalikan, dimana tersedianya 4 lampu yang dapat dikendalikan, dan cara mengendalikannya adalah user memilih lampu berapa yang akan dikendalikan

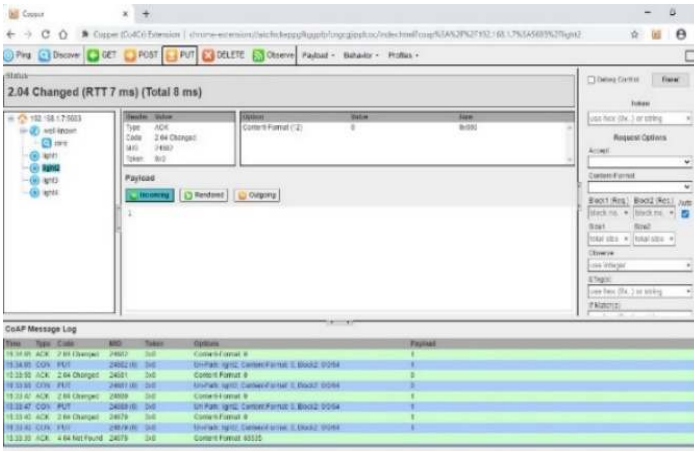
Pada gambar 15 adalah Tampilan untuk halaman lampu 1, dimana pada tampilan ini *user* dapat mengubah kondisi lampu yang dari awalnya mati menjadi menyala.



Gambar 14. Tampilan COPPER CU Halaman Awal terlihat 4 pilihan lampu



Gambar 15. Tampilan COPPER CU Halaman Lampu 1

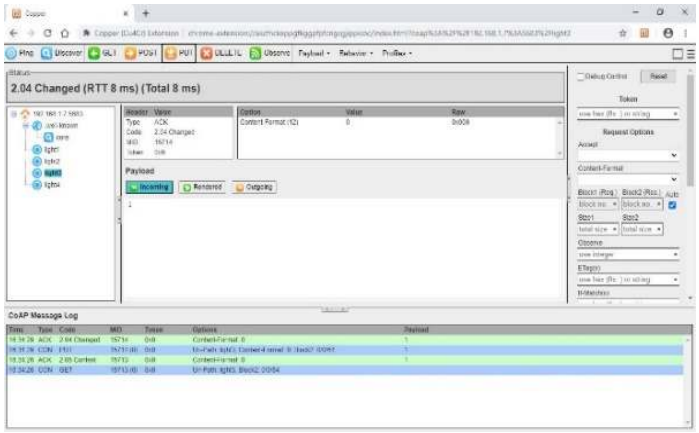


Gambar 16. Tampilan COPPER CU Halaman Lampu 2

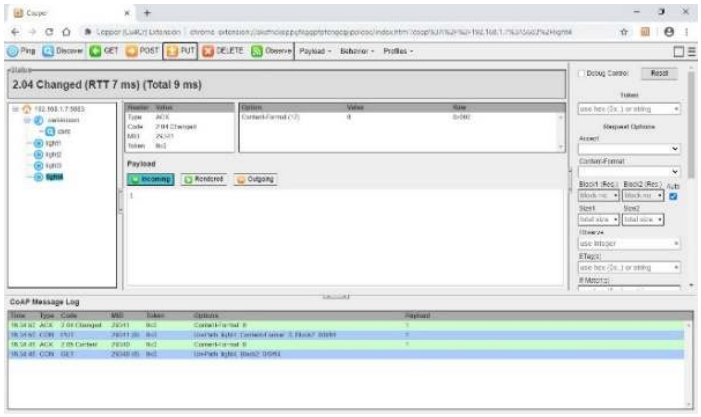
Pada gambar 16 adalah Tampilan untuk halaman lampu 2, dimana pada tampilan ini *user* dapat mengubah kondisi lampu yang dari awalnya mati menjadi menyala.

Pada gambar 17 adalah Tampilan untuk halaman lampu 3, dimana pada tampilan ini *user* dapat mengubah kondisi lampu yang dari awalnya mati menjadi menyala.

Pada gambar 18 adalah Tampilan untuk halaman lampu 4, dimana pada tampilan ini *user* dapat mengubah kondisi lampu yang dari awalnya mati menjadi menyala.



Gambar 17. Tampilan COPPER CU Halaman Lampu 3



Gambar 18. Tampilan COPPER CU Halaman Lampu 4

3) Konfigurasi Perangkat Keras

Pada konfigurasi perangkat keras ini akan di jelaskan semua perangkat keras yang digunakan serta cara konfigurasi dari setiap alat tersebut sampai bisa di gunakan untuk membuat sistem.

a. Arduino Mega 2560 ESP8266

Arduino Mega 2560 ESP8266 pada sistem ini berfungsi sebagai pusat kendali dari semua aktifitas yang dilakukan oleh sistem, pada konfigurasi *Arduino Mega 2560 ESP8266* ada beberapa hal yang harus dilakukan yaitu mengatur saklar untuk mematikan *ESP8266* dan hanya mengaktifkan *Arduino Mega 2560* untuk disambungkan dengan *Ethernet Shield*.[9], [10]

b. Ethernet Shield

Ethernet Shield pada sistem ini berfungsi sebagai pengganti dari *ESP8266* pada *Arduino Mega 2560* yang akan dihubungkan dengan kabel *RJ45* sebagai penghubung antara jaringan internet dengan *Arduino Mega 2560* yang digunakan sebagai pusat kendali sistem yang telah dibuat.[9]

4) Kode Program Sistem

Kode program mikrokontroller menggunakan Arduino IDE ini berfungsi untuk menghubungkan mikrokontroler dengan *library microcoap* dan *client* menggunakan bahasa pemograman C. Pada sistem ini program akan mengirimkan kode *library microcoap* untuk di jalankan pada mikrokontroler.

```
#include <SPI.h>
#include <Ethernet.h>
#include <stdint.h>
#include <EthernetUdp.h>
#include "coap.h"

#define PORT 5683
static uint8_t mac[] = {0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x02};
```

Gambar 20. Kode Program pada Mikrokontroler (a)

```
EthernetClient client;
EthernetUDP udp;
uint8_t packetbuf[256];
static uint8_t scratch_raw[32];
static coap_rw_buffer_t scratch_buf = {scratch_raw,
sizeof(scratch_raw)};

void setup()
{
    int i;
    Serial.begin(9600);
    while (!Serial)
    {
        ;
    }

    if (Ethernet.begin(mac) == 0)
    {
        Serial.println("Failed to configure Ethernet using DHCP");
        while(1);
    }
    Serial.print("My IP address: ");
    for (i=0;i<4;i++)
    {
        Serial.print(Ethernet.localIP()[i], DEC);
        Serial.print(".");
    }
    Serial.println();
    udp.begin(PORT);

    coap_setup();
    endpoint_setup();
}

void udp_send(const uint8_t *buf, int buflen)
{
    udp.beginPacket(udp.remoteIP(), udp.remotePort());
    while(buflen--)
        udp.write(*buf++);
    udp.endPacket();
}
```

Gambar 21. Kode Program pada Mikrokontroler (b)

```
void loop()
{
    int sz;
    int rc;
    coap_packet_t pkt;
    int i;

    if ((sz = udp.parsePacket()) > 0)
    {
        udp.read(packetbuf, sizeof(packetbuf));

        for (i=0;i<sz;i++)
        {
            Serial.print(packetbuf[i], HEX);
            Serial.print(" ");
        }
        Serial.println("");

        if (0 != (rc = coap_parse(&pkt, packetbuf, sz)))
        {
            Serial.print("Bad packet rc=");
            Serial.println(rc, DEC);
        }
        else
        {
            size_t rsplen = sizeof(packetbuf);
            coap_packet_t rsppkt;
            coap_handle_req(&scratch_buf, &pkt, &rsppkt);

            memset(packetbuf, 0, UDP_TX_PACKET_MAX_SIZE);
            if (0 != (rc = coap_build(packetbuf, &rsplen, &rsppkt)))
            {
                Serial.print("coap_build failed rc=");
                Serial.println(rc, DEC);
            }
            else
            {
                udp_send(packetbuf, rsplen);
            }
        }
    }
}
```

Gambar 22. Kode Program pada Mikrokontroler (c)

```
#include <stdbool.h>
#include <string.h>
#include <stdio.h>
#include "coap.h"
static char light1 = '1';
static char light2 = '1';
static char light3 = '1';
static char light4 = '1';
const uint16_t rsplen = 1500;
static char rsp[1500] = "";
void build_rsp(void);

#ifdef ARDUINO
#include "Arduino.h"
static int led1 = 4;
static int led2 = 5;
static int led3 = 6;
static int led4 = 7;

void endpoint_setup(void)
{
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
    pinMode(led4, OUTPUT);
    build_rsp();
}
#else
#include <stdio.h>
void endpoint_setup(void)
{
    build_rsp();
}
#endif
static const coap_endpoint_path_t path_well_known_core = {2, {".wel
l-known", "core"}};
static int handle_get_well_known_core(coap_rw_buffer_t *scratch, co
nst coap_packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uin
t8_t id_lo)
```

Gambar 23. Kode Program *library* microcoap (a)

```
{
    return coap_make_response(scratch, outpkt, (const uint8_t *)rsp
, strlen(rsp), id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CONTENT, COAP_CONTENTTYPE_APPLICATION_LINKFORMAT
);
}

static const coap_endpoint_path_t path_light1 = {1, {"light1"}};
static int handle_get_light1(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    return coap_make_response(scratch, outpkt, (const uint8_t *)&li
ght1, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CONTENT, COAP_CONTENTTYPE_TEXT_PLAIN);
}

static int handle_put_light1(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    if (inpkt->payload.len == 0)
        return coap_make_response(scratch, outpkt, NULL, 0, id_hi,
id_lo, &inpkt-
>tok, COAP_RSPCODE_BAD_REQUEST, COAP_CONTENTTYPE_TEXT_PLAIN);
    if (inpkt->payload.p[0] == '1')
    {
        light1 = '1';
#ifdef ARDUINO
        digitalWrite(led1, HIGH);
#else
        printf("ON\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light1, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
    else
    {
        light1 = '0';
#ifdef ARDUINO
        digitalWrite(led1, LOW);
#else
        printf("OFF\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light1, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
}
```

Gambar 24. Kode Program *library* microcoap (b)

```
static int handle_put_light2(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    if (inpkt->payload.len == 0)
        return coap_make_response(scratch, outpkt, NULL, 0, id_hi,
id_lo, &inpkt-
>tok, COAP_RSPCODE_BAD_REQUEST, COAP_CONTENTTYPE_TEXT_PLAIN);
    if (inpkt->payload.p[0] == '1')
    {
        light2 = '1';
#ifdef ARDUINO
        digitalWrite(led2, HIGH);
#else
        printf("ON\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light2, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
    else
    {
        light2 = '0';
#ifdef ARDUINO
        digitalWrite(led2, LOW);
#else
        printf("OFF\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light2, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
}

static const coap_endpoint_path_t path_light3 = {1, {"light3"}};
static int handle_get_light3(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    return coap_make_response(scratch, outpkt, (const uint8_t *)&li
ght3, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CONTENT, COAP_CONTENTTYPE_TEXT_PLAIN);
}

static int handle_put_light3(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    if (inpkt->payload.len == 0)
```

Gambar 25. Kode Program *library* microcoap (b)

```
return coap_make_response(scratch, outpkt, NULL, 0, id_hi, id_lo, &
inpkt->tok, COAP_RSPCODE_BAD_REQUEST, COAP_CONTENTTYPE_TEXT_PLAIN);
    if (inpkt->payload.p[0] == '1')
    {
        light3 = '1';
#ifdef ARDUINO
        digitalWrite(led3, HIGH);
#else
        printf("ON\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light3, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
    else
    {
        light3 = '0';
#ifdef ARDUINO
        digitalWrite(led3, LOW);
#else
        printf("OFF\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light3, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
}

static const coap_endpoint_path_t path_light4 = {1, {"light4"}};
static int handle_get_light4(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    return coap_make_response(scratch, outpkt, (const uint8_t *)&li
ght4, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CONTENT, COAP_CONTENTTYPE_TEXT_PLAIN);
}

static int handle_put_light4(coap_rw_buffer_t *scratch, const coap_
packet_t *inpkt, coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_l
o)
{
    if (inpkt->payload.len == 0)
        return coap_make_response(scratch, outpkt, NULL, 0, id_hi,
id_lo, &inpkt-
>tok, COAP_RSPCODE_BAD_REQUEST, COAP_CONTENTTYPE_TEXT_PLAIN);
    if (inpkt->payload.p[0] == '1')
    {
        light4 = '1';
#ifdef ARDUINO
```

Gambar 26. Kode Program *library* microcoap (d)

```
digitalWrite(led4, HIGH);
#else
        printf("ON\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light4, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
    else
    {
        light4 = '0';
#ifdef ARDUINO
        digitalWrite(led4, LOW);
#else
        printf("OFF\n");
#endif
        return coap_make_response(scratch, outpkt, (const uint8_t *
)&light4, 1, id_hi, id_lo, &inpkt-
>tok, COAP_RSPCODE_CHANGED, COAP_CONTENTTYPE_TEXT_PLAIN);
    }
}

const coap_endpoint_t endpoints[] =
{
    {COAP_METHOD_GET, handle_get_well_known_core, &path_well_known_
core, "ct=40"},
    {COAP_METHOD_GET, handle_get_light1, &path_light1, "ct=0"},
    {COAP_METHOD_PUT, handle_put_light1, &path_light1, NULL},
    {COAP_METHOD_GET, handle_get_light2, &path_light2, "ct=0"},
    {COAP_METHOD_PUT, handle_put_light2, &path_light2, NULL},
    {COAP_METHOD_GET, handle_get_light3, &path_light3, "ct=0"},
    {COAP_METHOD_PUT, handle_put_light3, &path_light3, NULL},
    {COAP_METHOD_GET, handle_get_light4, &path_light4, "ct=0"},
    {COAP_METHOD_PUT, handle_put_light4, &path_light4, NULL},
    {(coap_method_t)0, NULL, NULL, NULL}
};

void build_rsp(void)
{
    uint16_t len = rsplen;
    const coap_endpoint_t *ep = endpoints;
    int i;

    len--; // Null-terminated string

    while(NULL != ep->handler)
    {
        if (NULL == ep->core_attr) {
            ep++;
            continue;
```

Gambar 27. Kode Program *library* microcoap (e)

```
}

        if (0 < strlen(rsp)) {
            strncat(rsp, ",", len);
            len--;
        }

        strncat(rsp, "<", len);
        len--;

        for (i = 0; i < ep->path->count; i++) {
            strncat(rsp, "/", len);
            len--;

            strncat(rsp, ep->path->elems[i], len);
            len -= strlen(ep->path->elems[i]);
        }

        strncat(rsp, ">;", len);
        len -= 2;

        strncat(rsp, ep->core_attr, len);
        len -= strlen(ep->core_attr);

        ep++;
    }
}
```

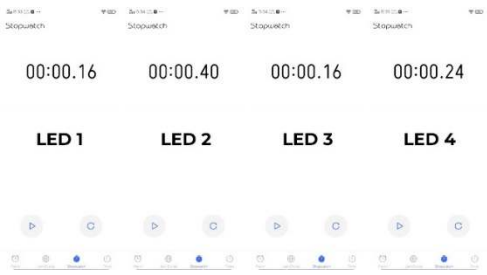
Gambar 28. Kode Program *library* microcoap (f)

a. *Kode Program Mikrokontroler menggunakan Arduino IDE*

Kode program mikrokontroller menggunakan Arduino IDE ini berfungsi untuk menghubungkan mikrokontroler dengan *library microcoap* dan *client* menggunakan bahasa pemograman C. Pada sistem ini program akan mengirimkan kode *library microcoap* untuk di jalankan pada mikrokontroler. Kode program dapat dilihat pada gambar 20, gambar 21, dan gambar 22.

b. *Kode Program library microcoap menggunakan Visual Studio Code*

Kode program *library microcoap* menggunakan *visual studio code* ini berfungsi untuk menjalankan sistem dalam proses kendali atau kontrol untuk menyalakan dan mematikan lampu melalui *browser chromium* dan menggunakan *extension* COPPER CU. Kode program dapat dilihat pada gambar 23 sampai dengan gambar 28.



Gambar 29. Hasil uji coba sistem.

5) Uji Coba Sistem

Pada tahap ini akan dilakukan pengujian terhadap sistem yang dibuat. Pengujian ini bertujuan untuk mengidentifikasi kekurangan dari sistem yang telah dibuat, serta kelayakan dari sistem tersebut untuk dapat diterapkan. Dalam pengujian ini adalah pengujian tentang kecepatan respon dari *server* untuk memproses perintah dari *user* dalam menyalakan sebuah lampu, dan pada percobaan ini peneliti mencoba 4 buah *LED* yang di uji menggunakan jaringan internet. Hasil uji coba respon server dapat dilihat pada gambar 29.

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Pada pembuatan prototipe aplikasi IoT kendali jarak jauh lampu rumah menggunakan protokol CoAP, maka dapat disimpulkan bahwa penulis berhasil merancang dan membangun prototipe aplikasi IoT kendali jarak jauh lampu rumah menggunakan protocol CoAP. Proses pembuatan ini dengan menggunakan mikrokontroler Arduino mega 2560 + ESP8266 dan jugan menggunakan modul tambahan yaitu adalah *Ethernet Shield*, menggunakan *library microcoap* dalam membuat sistem berikut. Berdasarkan konfigurasi Arduino mega 2560 dan *Ethernet Shield* serta proses kode program mikrokontroler menggunakan Arduino IDE dan proses kode program *library microcoap* menggunakan *Visual Studio Code* bahwa data yang telah didapatkan akan dikirim ke server menggunakan mikrokontroler Arduino Mega 2560 yang dihubungkan dengan kabel *LAN RJ-45* yang dipasang pada *Ethernet Shield*. *User* dapat menggunakan sistem ini melalui *browser Chromium* dan menggunakan *extensions COPPER CU* sebagai tampilan *client* – nya.

B. Saran

Pada penelitian ini tentunya masih memiliki kekurangan – kekurangan yang ditemukan. Untuk itu diharapkan kedepan dapat dilakukan pengembangan terhadap sistem yang memanfaatkan protocol CoAP untuk sistem kontrol berbagai peralatan listrik lainnya. Ada beberapa saran – saran yang dapat dibagikan oleh penulis yaitu, Kedepan diharapkan, sistem ini sudah dapat di operasikan dengan jaringan wireless dan dapat menggunakan *smartphone* menggunakan sistem ini,

V. KUTIPAN

[1] M. R. Nurkamiden, M. E. I. Najoan, and M. D. Putro, "Rancang Bangun Sistem Pengendalian Perangkat Listrik Berbasis Web Server Menggunakan Mini PC Raspberry Pi Studi Kasus Gedung Fakultas Teknik Universitas Sam Ratulangi," *J. Tek. Inform.*, vol. 11, no. 1, 2017, doi: 10.35793/jti.11.1.2017.15980.

[2] M. R. Putra, E. S. Pramukantoro, and F. A. Bakhtiar, "Implementasi Constrained Application Protocol (CoAP) Pada Semantic IoT Web Service," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 3, no. 5, pp. 4671–4679, 2019.

[3] Ruuhwan, R. Rizal, and I. Karyana, "Sistem Kendali dan Monitoring pada Smart Home Berbasis Internet of Things (IoT)," *Innov. Res. Informatics*, vol. 2, no. October, pp. 43–50, 2019.

[4] Y. F. Wiryawan, D. P. Kartikasari, and M. Data, "Implementasi Constrained Application Protocol (CoAP) pada Sistem Pengamatan Kelembaban Tanah," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 8, pp. 2480–2487, 2017.

[5] M. Iglesias-Urkia, A. Orive, and A. Urbieta, "Analysis of CoAP Implementations for Industrial Internet of Things: A Survey," *Procedia Comput. Sci.*, vol. 109, no. 2016, pp. 188–195, 2017, doi: 10.1016/j.procs.2017.05.323.

[6] C. Martín, M. Díaz, and B. Rubio, "Run-time deployment and management of CoAP resources for the internet of things," *Int. J. Distrib. Sens. Networks*, vol. 13, no. 3, 2017, doi: 10.1177/1550147717698969.

[7] C. B. Da Porciúncula, S. Beskow, D. S. Marcon, and J. C. Nobre, "Constrained Application Protocol (CoAP) no Arduino UNO R3: Uma Análise Prática," *Univ. do Val. do Rio dos Sinos Porto Alegre – RS – Bras.*, 2020, doi: 10.5753/wpietf.2018.3212.

[8] A. R. and P. Renold A, "Coap Based Acute Parking Lot Monitoring System Using Sensor Networks," *ICTACT J. Commun. Technol.*, vol. 05, no. 02, pp. 923–928, 2014, doi: 10.21917/ijct.2014.0132.

[9] S. Trilles, A. González-Pérez, and J. Huerta, "A comprehensive IoT node proposal using open hardware. A smart farming use case to monitor vineyards," *Electron.*, vol. 7, no. 12, 2018, doi: 10.3390/electronics7120419.

[10] P. C, S. S, and K. KV, "Remote Patient Monitoring System Based Coap in Wireless Sensor Networks," *Int. J. Sens. Networks Data Commun.*, vol. 5, no. 3, pp. 5–11, 2016, doi: 10.4172/2090-4886.1000145.

Rangga Dwi Putra Firmansyah, lahir di Jakarta 9 Juni 1997. Penulis merupakan anak ke-2 dari 2 bersaudara, dari pasangan Fremodharis dan Vinny Rikha Mentang. Penulis mulai menempuh Pendidikan di Sekolah Dasar Marsudirini Jakarta (2003-2008) dan Sekolah Dasar Kristen Kalam Kudus Jayapura (2008-2009) .Penulis lalu melanjutkan ke Sekolah Menengah Pertama Kristen Kalam Kudus Jayapura (2009-2012). Kemudian penulis melanjutkan ke Sekolah Menengah Atas Negeri 4 Jayapura (2012-2015). Pada tahun 2015 penulis melanjutkan Pendidikan ke salah satu perguruan tinggi yang ada di Manado yaitu Universitas Sam Ratulangi Manado, dengan mengambil Program Studi S-1Teknik Informatika di Fakultas Teknik. Penulis mengajukan proposal Skripsi untuk memenuhi syarat meraih gelar sarjana (S1) dengan judul Prototipe Aplikasi Iot Kendali Jarak Jauh Lampu Rumah menggunakan Protokol CoAP, skripsi ini dibimbing oleh dua dosen pembimbing, yaitu Xaverius B. N. Najoan, ST., MT. dan Sherwin R. U. A. Sompie, ST., MT. Pada tanggal 16 Oktober 2020, penulis resmi menyelesaikan skripsi dan menyandang gelar sarjana komputer .