



# Internet of Things (IoT)

## Ujian Akhir Semester (UAS)

[unsia.ac.id](http://unsia.ac.id)



### YAYASAN MEMAJUKAN ILMU DAN KEBUDAYAAN UNIVERSITAS SIBER ASIA

Kampus Menara, Jl. RM. Harsono, Ragunan - Jakarta Selatan. Daerah Khusus Ibukota Jakarta 12550.  
Telp. (+6221) 27806189. [asiacyberuni@acu.ac.id](mailto:asiacyberuni@acu.ac.id). [www.unsia.ac.id](http://www.unsia.ac.id)

Mata Kuliah	: Internet of Things
Kelas	: IT-602
Prodi	: PJJ Informatika
Nama Mahasiswa	: Hendro Gunawan
NIM	: 200401072103
Dosen	: Anjrah Mintana, M.Kom

[unsia.ac.id](http://unsia.ac.id)



# Kata Pengantar

Saat ini komputer dan manusia, sepenuhnya sangat tergantung pada internet untuk mendapatkan informasi. *Internet of Things* (IoT) mempunyai kemampuan untuk mengubah dunia termasuk dunia pendidikan Indonesia. Sebelum pandemi Covid-19, *Internet of Things* (IoT) sudah bergema seantero nusantara, walaupun nyaris tak terasa. Namun, pandemi *Covid-19* mengubah segalanya menjadi mungkin. *Internet of Things* (IoT) menggema lebih nyaring dari biasanya. Dampak *Covid-19* pada IoT dapat melampaui batas, terutama jika virus terus menyebar ke seluruh dunia. Pandemi telah memisahkan dunia pendidikan menjadi dua jenis, yaitu pendidikan yang siap menggunakan teknologi untuk peningkatan kualitas dan kemajuan peserta didiknya, serta pendidikan yang menolak beradaptasi dengan cara kerja baru. Memang, kesenjangan digital ini sudah ada sebelumnya, tetapi tidak pernah setajam atau begitu menentukan seperti saat pandemi. *Internet of Things* (IoT) telah memantapkan keberadaannya sebagai solusi untuk menangani masalah pendidikan bagi masyarakat di masa depan.

Terima kasih saya ucapkan kepada Bapak Anjrah Mintana, M.Kom selaku dosen mata kuliah Internet of Things di Universitas Siber Asia (UNSA) yang telah mengajarkan kepada kami ilmu yang bermanfaat melalui pertemuan sinkron maupun asinkron dengan beberapa materi yang telah diajarkan yaitu: pertemuan 1 Pendahuluan, pertemuan 2 membahas Board dan ESPloreIDE, pertemuan 3 Membuat Program Sederhana dan membahas tentang Dasar-Dasar Bahasa Lua, pertemuan 4 membahas tentang GPIO (General Purpose Input Output), pada pertemuan 5 membahas tentang Timer dan Wifi, pada pertemuan 6 membahas tentang RTC (Real Time Clock) dan External Interrupts, pada pertemuan 7 membahas tentang UART (Universal Asynchronous Receiver Transmitter) dan ADC (Analog To Digital Converter), pertemuan ke 8 UTS, pada pertemuan 9 membahas tentang PWM (Pulse Width Modulation), pada pertemuan 10 membahas tentang Web Server dan Kontrol melalui Jaringan, pada pertemuan 11 membahas tentang Kontrol Melalui Internet, pada pertemuan 12 membahas Program WEB dan Android Untuk kontrol WEB, kemudian pada pertemuan 13 membahas tentang Meracik Binari Sendiri Untuk Di Flash di ESP, pada pertemuan 14 membahas Hacking Wi-Fi (Wi-Fi Jammer), dan yang terakhir pertemuan 15 membahas tentang Stack protocol Jaringan IoT. Semoga bermanfaat.

Gresik, 13 Agustus 2023

Penulis

(Hendro Gunawan)



# Daftar Isi

Kata Pengantar.....	i
Daftar Isi.....	ii
<b>BAB I (PWM) PULSE WITH MODULATION.....</b>	<b>2</b>
1.1 Pengertian PWM.....	2
1.2 Penggunaan PWM.....	2
1.3 Frekuensi dan Resolusi.....	3
1.4 Rangkaian Hardware.....	3
1.5 Program PWM.....	3
1.6 Implementasi PWM Untuk Mengatur Kecerahan LED.....	4
<b>BAB II KONTROL MELALUI JARINGAN.....</b>	<b>5</b>
2.1 Pengertian.....	5
2.2 Gambaran Konsep IoT.....	5
2.3 Contoh Aplikasi.....	6
2.4 Implementasi Program.....	8
<b>BAB III PROGRAM WEB DAN ANDROID UNTUK KONTROL LED...11</b>	<b>11</b>
3.1 Pengertian.....	11
3.2 Kontrol LED pada Internet of Things.....	12
3.2.1 Kontrol LED Melalui Program WEB.....	12
3.2.2 Kontrol LED Melalui Aplikasi Android.....	12
3.3 Program Berbasis WEB.....	13
3.4 Aplikasi Mobile Berbasis Android.....	14
3.5 Menambahkan Tombol Cek Status.....	21



## Pertanyaan 1

1. Anda diberikan tugas untuk merancang dan mengimplementasikan sistem pengendalian kecerahan lampu LED menggunakan metode PWM (Pulse Width Modulation) pada platform Arduino dalam proyek Internet of Things (IoT). Sistem ini akan memungkinkan pengguna untuk mengatur kecerahan lampu LED secara nirkabel melalui aplikasi smartphone. Gambarkan desain rangkaian untuk mengendalikan lampu LED menggunakan PWM pada Arduino, sertakan koneksi modul nirkabel dan lampu LED ke Arduino.

## Jawaban

### BAB I

#### PWM (PULSE WITH MODULATION)

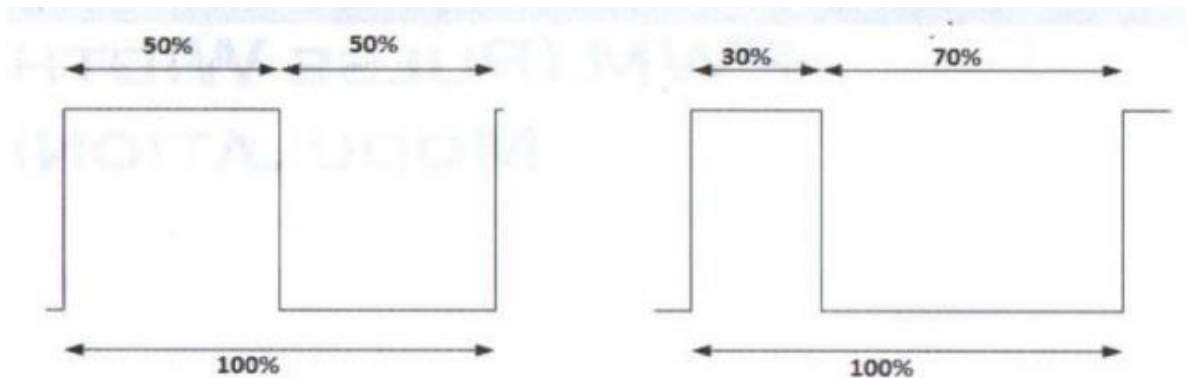
### BAB 1

#### PWM (Pulse With Modulation)

##### 1.1 Pengertian PWM

PWM (*Pulse With Modulation*) adalah teknik modulasi dengan cara mengubah lebar denyut (*duty cycle*) sinyal dalam satu siklus gelombang penuh dengan amplitudo dan frekuensi yang sama.

Sedangkan yang dimaksud dengan *duty cycle* adalah prosentase dari lebar pulsa high dibanding dengan lebar total satu siklus gelombang penuh. Lihat gambar 1.1.



Gambar 1.1 Duty cycle

Gambar kiri adalah contoh gelombang dengan *duty cycle* 50%, karena pulsa high lebarnya 50% dibandingkan dengan total gelombang. Sedangkan gambar kanan adalah contoh gelombang dengan *duty cycle* 30%, karena pulsa high lebarnya 30% dibandingkan dengan total gelombang.

*Duty cycle* bernilai antara 0% - 100%, dimana 0% berarti full low signal, sedangkan 100% berarti full high signal.

##### 1.2 Penggunaan PWM



Contoh pemanfaatan dari PWM adalah untuk mengendalikan kecepatan motor DC, dengan cara semakin besar *duty cycle* maka semakin cepat putaran motor begitu pula sebaliknya semakin kecil *duty cycle* maka semakin lambat putaran motor. Contoh lainnya adalah mengatur kecerahan LED, mengendalikan sudut motor servo. Pengendalian kecerahan layar laptop juga adalah salah satu contoh pemanfaatan dari PWM ini.

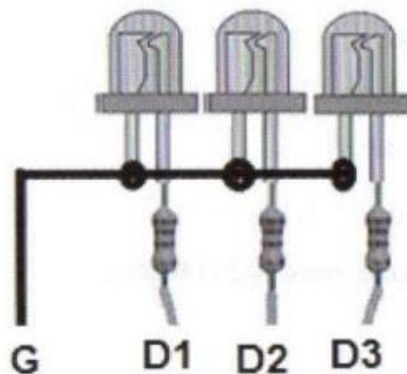
### 1.3 Frekuensi dan Resolusi

Frekuensi menyatakan banyaknya sinyal yang terbentuk dalam setiap detik. Bisa saja bernilai 500Hz, 1000Hz (1 KHz) dan lain-lain tergantung alat yang dikontrol.

Resolusi adalah berapa banyak macam *duty cycle* yang diperlukan. Misalnya kita ingin mengatur kecerahan LED dalam 16 tingkat, berarti kita membutuhkan 16 tingkat resolusi dan seterusnya.

### 1.4 Rangkaian Hardware

Dalam contoh ini kita memakai 3 channel di ESP, yaitu D1, D2, D3. Sediakan 3 buah LED dan 3 buah resistor pembatas arus 220 Ohm seperti gambar. Colokkan bagian anoda masing-masing ke B6, B7, dan B8. Sedangkan bagian Katoda ke GND. Lihat gambar 1.2.



*Gambar 1.2 Rangkaian hardware*

### 1.5 Program PWM

Adapun programnya adalah sebagai berikut:

```
pwm.setup (1, 3, 512) - pin 1, frek 3 Hz, duty cycle separuh
pwm.setup (2, 3, 512)
pwm.setup (3, 3, 512)
pwm.stsrt (1)
pwm.start (2)
pwm.start (3)
function led (r, g, b)
    pwm.setduty (1, g)
    pwm.setduty (2, b)
    pwm.setduty (3, r)
end
led (128, 256, 512)
```

**Penjelasannya:**



Pertama lakukan setup dengan perintah:

```
pwm.setup (pin, clock, duty)
```

- **pin:** 1 – 12, nomor gpio
- **clock:** 1 – 1000, frekuensi
- **duty:** 0 – 1023, duty cycle, max 1023 (100%)

Kemudian lakukan start untuk masing-masing pwm:

```
pwm.start (pin)
```

- **pin:** 1 – 12, nomor gpio

Selanjutnya, lakukan setduty:

```
pwm.setduty (pin, duty)
```

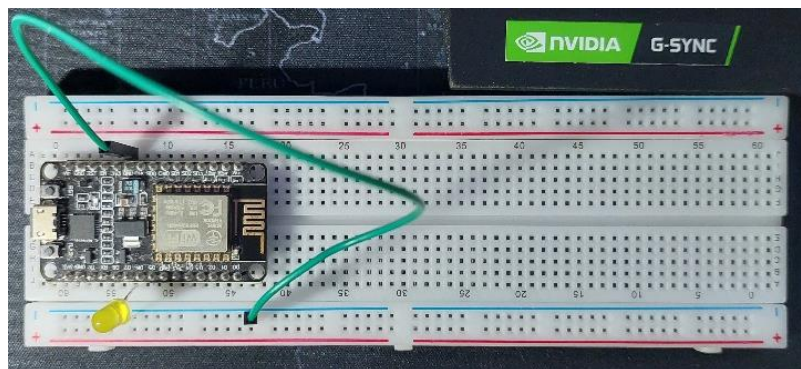
- **pin:** 1 – 12, nomor gpio
- **duty:** 0 – 1023, duty cycle, max 1023 (100%)

Ubahlah nilai:

```
led (128, 256, 512)
```

Perhatikan variasi lamanya kedip dari masing-masing LED.

## 1.6 Implementasi PWM Untuk Mengatur Kecerahan LED



*Gambar 1.3 Desain rangkaian*

Langkah yang harus kita lakukan adalah yang pertama kita harus mendaftarkan akun kita di *Blynk Cloud* di <https://blynk.io/> setelah itu kita buat nama Device name dengan nama **Kontrol LED**. Kemudian kita atur dengan pengaturan sesuai yang kita inginkan.

Adapun kode programnya adalah sebagai berikut:

```
#define BLYNK_PRINT Serial

/* Fill in information from Blynk Device Info here */
#define BLYNK_TEMPLATE_ID      "TMPL6tCLgA2mi"
#define BLYNK_TEMPLATE_NAME    "Kontrol LED"
#define BLYNK_AUTH_TOKEN       "6eBeg1vm5ZrI4w7_z3_bjas0ze5t0wAD"
#include <ESP8266WiFi.h>
```



```
#include <BlynkSimpleEsp8266_SSL.h>

char auth[] = "6eBeg1vm5ZrI4w7_z3_bjas0ze5t0wAD";
// Your WiFi credentials.
// Set password to "hendro.gnwn@gmail.com" for open networks.
char ssid[] = "hendro.gnwn@gmail.com";
char pass[] = "Hendro0181";

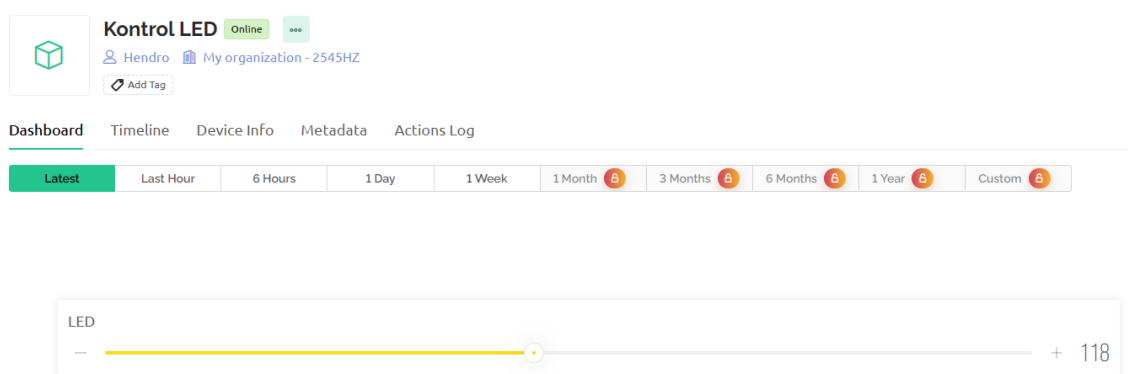
int led = 13;
int pinValue;
BLYNK_WRITE(V1)

{
  //assigning incoming value from pin V1 to a variable
  int pinValue = param.asInt();
  analogWrite(led,pinValue);
  Blynk.virtualWrite(V13,pinValue);
  Serial.print("V13 Slider value is: ");
  Serial.print(pinValue);
}

void setup()
{
  pinMode (led,OUTPUT);
  // Debug console
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
}

void loop()
{
  Blynk.run();
}
```

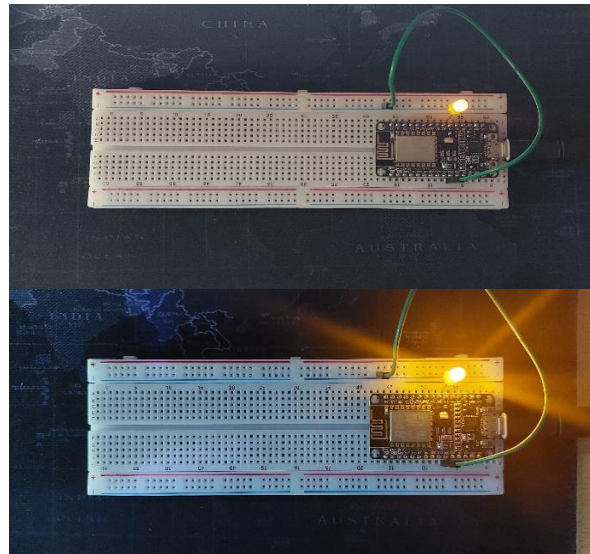
Setelah selesai, buka aplikasi Arduino-nya, kemudian program tersebut kita upload, dan kita cek pada akun Blynk yang telah kita buat, kemudian kita bisa atur nyala lampu LED menjadi redup dan terang dengan intensitas 1 hingga 255. Seperti tampak pada gambar berikut.



**Gambar 1.4 Pengaturan Nyala LED pada Blynk**



Hasilnya adalah seperti tampak pada gambar 1.4.



*Gambar 1.5 Hasil percobaan menyalakan lampu KLED*

## Pertanyaan 2

2. Jelaskan pengertian dari "kontrol melalui jaringan" dalam konteks Internet of Things (IoT). Gambarkan bagaimana konsep ini memungkinkan perangkat IoT untuk dikendalikan dari jarak jauh melalui koneksi jaringan. Berikan contoh aplikasi praktis dari kontrol melalui jaringan pada proyek IoT.

### Jawaban

## BAB II

### KONTROL MELALUI JARINGAN

## BAB 2

### Kontrol Melalui Jaringan

#### 2.1 Pengertian

**Kontrol melalui jaringan** adalah konsep dalam Internet of Things (IoT) yang mengacu pada kemampuan perangkat IoT untuk dikendalikan dari jarak jauh melalui koneksi jaringan, seperti internet atau jaringan lokal. Dengan kata lain, pengguna dapat mengontrol, memonitor, dan berinteraksi dengan perangkat IoT tanpa harus berada di dekatnya secara fisik.

#### 2.2 Gambaran Konsep IoT

Berikut adalah gambaran bagaimana konsep IoT ini berfungsi:

##### 1. Identifikasi perangkat IoT

Setiap perangkat IoT harus memiliki identitas yang unik dalam jaringan, sehingga dapat diidentifikasi dan diakses oleh sistem kontrol.





## 2. Koneksi jaringan

Perangkat IoT harus terhubung ke jaringan, baik melalui Wi-Fi, Ethernet, atau teknologi lainnya, untuk dapat berkomunikasi dengan perangkat pengendali (seperti smartphone, tablet, atau server).

## 3. Protokol komunikasi

Perangkat IoT dan perangkat pengendali harus mengadopsi protokol komunikasi yang sesuai agar dapat saling berkomunikasi. Beberapa protokol umum dalam IoT adalah MQTT (*Message Queuing Telemetry Transport*), CoAP (*Constrained Application Protocol*), dan HTTP (*Hypertext Transfer Protocol*).

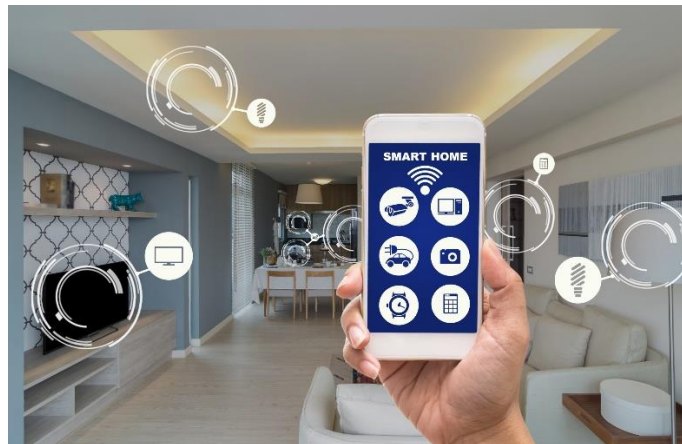
## 4. Penggunaan Aplikasi atau Antarmuka Pengguna

Pengguna dapat mengontrol perangkat IoT melalui aplikasi khusus pada smartphone, tablet, atau antarmuka pengguna lainnya yang terhubung ke perangkat melalui jaringan.

### 2.3 Contoh Aplikasi

Contoh aplikasi praktis dari kontrol melalui jaringan pada proyek IoT:

#### 1. Smart Home



*Gambar 2.1 Smart home*

Dalam konteks rumah pintar (*smart home*), lampu, kipas angin, perangkat AC, atau perangkat lainnya dapat dikendalikan dari jarak jauh melalui aplikasi di *smartphone*. Pengguna dapat menyalakan atau mematikan perangkat, mengatur suhu ruangan, atau mengubah intensitas pencahayaan tanpa harus berada di rumah secara fisik.

#### 2. Smart Security System



*Gambar 2.2 Smart Security System*



Sistem keamanan pintar seperti kamera keamanan dan alarm dapat diakses dan dikendalikan dari jarak jauh. Pengguna dapat memantau keadaan rumah atau kantor mereka melalui feed video langsung dari kamera keamanan, serta mengaktifkan atau menonaktifkan sistem keamanan dengan mudah.

### 3. Industri Otomasi



*Gambar 2.3 Industri Otomasi*

Di lingkungan industri, IoT memainkan peran penting dalam otomasi. Perangkat IoT seperti sensor dan aktuator dapat diprogram dan dikendalikan dari jarak jauh melalui koneksi jaringan. Misalnya, suatu sistem produksi dapat diawasi dan dikelola secara real-time, sehingga dapat dioptimalkan untuk meningkatkan efisiensi dan mengurangi downtime.

### 4. Pertanian Pintar



*Gambar 2.4 Pertanian Pintar*

Pertanian dapat ditingkatkan dengan teknologi IoT. Perangkat IoT seperti sensor kelembaban tanah, sensor suhu, dan irigasi otomatis dapat diintegrasikan ke dalam sistem pertanian pintar. Petani dapat memantau dan mengatur kondisi pertanian mereka dari jarak jauh untuk meningkatkan produktivitas dan efisiensi.

Penerapan kontrol melalui jaringan pada proyek IoT tidak hanya memudahkan pengguna untuk mengelola perangkat mereka dari jarak jauh, tetapi juga membuka peluang untuk mengoptimalkan kinerja dan fungsionalitas perangkat IoT secara keseluruhan. Namun, ketersediaan koneksi internet yang stabil dan keamanan data yang kuat adalah faktor penting untuk memastikan keberhasilan dan penerimaan teknologi ini.

### 2.4 Implementasi Program



Kali ini disajikan kontrol menyalakan dan mematikan LED melalui jaringan. Di sini disajikan kontrol menyalakan dan mematikan LED melalui jaringan lokal. Skenarionya adalah sebagai berikut:

- ESP dijadikan WiFi mode station, artinya dia konek ke sebuah AP dan memiliki IP sendiri.
- Karena sudah memiliki IP, dia bisa di-kontak oleh komputer lainnya
- Komputer yang berada dalam jaringan yang sama melakukan request ke port HTTP (80) ke IP dari ESP via browser di Laptop
- ESP melakukan replay terhadap request tadi tadi yang berisi tombol kontrol untuk menyalakan dan mematikan LED

Adapun programnya adalah sebagai berikut:

```
LED = 4
gpio.mode(LED, gpio.OUTPUT)
gpio.write(LED, gpio.HIGH)

station_cfg = {}
station_cfg.ssid = "xxxxxxxxxxxxx"
station_cfg.pwd = "xxxxxxxxxxxxx"

wifi.setmode(wifi.STATION)
wifi.sta.config(station_cfg)
wifi.sta.connect()

tmr.alarm(0, 2000, 1, function()
    if wifi.sta.getip() == nil then
        print ("Belum berhasil konek, mengulangi ...")
    else
        tmr.stop(0)
        print ("mode wifi: " .. wifi.getmode())
        print ("MAC address: " .. wifi.ap.getmac())
        print ("Berhasil, IP: " .. wifi.sta.getip())
    end
end)

srv = net.createServer (net.TCP)
srv:listen (80, function (conn)
    conn: on ("receive", function(conn, request)
        local buff = ""

        local _, _, method, path, var = string.find(request,
            "([A-Z]+) (.+) ? (.+) HTTP")
        if (method == nil) then
            _, _, method, path = string.find(request,
                "([A-Z]+) (.+) HTTP")
        end

        local get1 = {}
        if (vars == nil) then
            for k, v in string.gmatch(vars, "(%w+)-(w+t)&*" ) do
                get1[k] = v
            end
        end
    end)
end)
```



```

buff = buff .. "<html>"
buff = buff .. "<head><title>LED on/off</title></head>"
buff = buff .. "<body>"
buff = buff .. "<table border=1 width=11% height=130>"
buff = buff .. "<tr><td width=90 align=center>"
buff = buff .. "<a href='\"?pin=OFF\"'><button>OFF</button></td></tr>"
buff = buff .. "</table>"
buff = buff .. "</body>"
buff = buff .. "</html>"

    if (get1.pin == "ON") then
        gpio.write (LED, gpio.LOW)
    elseif (get1.pin == "OFF") then
        gpio.write (LED, gpio.HIGH)
    end

    conn: send(buff)
end)
conn:on("sent", function(conn) conn:close() end)
end)

```

### ***Penjelasan:***

Kita memakai LED di GPIO 4 yang sudah tersedia di Board ESP. GPIO di-set sebagai mode output. Keadaan awal adalah off.

```

LED = 4
gpio.mode(LED, gpio.OUTPUT)
gpio.write(LED, gpio.HIGH)

```

Selanjutnya, kita mengisi kredensial dari wifi agar ESP bisa konek ke AP. Sesuaikan dengan ssid dan password di tempat Anda.

```

station_cfg = {}
station_cfg.ssid = "xxxxxxxxxxxxx"
station_cfg.pwd = "xxxxxxxxxxxxx"

```

Berikut ini adalah bagian koneksi ke wifi.

```

wifi.setmode(wifi.STATION)
wifi.sta.config(station_cfg)
wifi.sta.connect()

```

Timer secara periodik setiap 2 detik mencoba untuk koneksi, diulangi terus hingga berhasil.

```

tmr.alarm(0, 2000, 1, funtion()
    if wifi.sta.getip() == nil then
        print ("Belum berhasil konek, mengulangi ...")
    else
        tmr.stop(0)
        print ("mode wifi: " .. wifi.getmode())
        print ("MAC address: " .. wifi.ap.getmac())
        print ("Berhasil, IP: " .. wifi.sta.getip())
    end
end)

```



end)

Kemudian dilakukan setting web server di port 80 yang berisi 2 event, on-receive dan on-sent. On receive adalah tempat untuk memproses request dari klien. On sent digunakan untuk menutup koneksi saat request sudah selesai dilayani.

```
srv = net.createServer (net.TCP)
srv:listen (80, function (conn)
    conn: on ("receive", function(conn, request)
    ...
    conn:on("sent", function(conn) conn:close() end)
```

Bagian selanjutnya digunakan untuk melakukan parsing request dengan regular expression untuk mencari string pin=OFF atau pin=ON. Hasilnya dimasukkan dalam variabel get1.

```
local _, _, method, path, var = string.find(request,
"([A-Z]+) (.+) ? (.+) HTTP")
if (method == nil) then
    _, _, method, path = string.find(request,
"([A-Z]+) (.+) HTTP")
end

local get1 = {}
if (vars == nil) then
    for k, v in string.gmatch(vars, "(%w+)-(w+t)&*") do
        get1[k] = v
    end
end
end
```

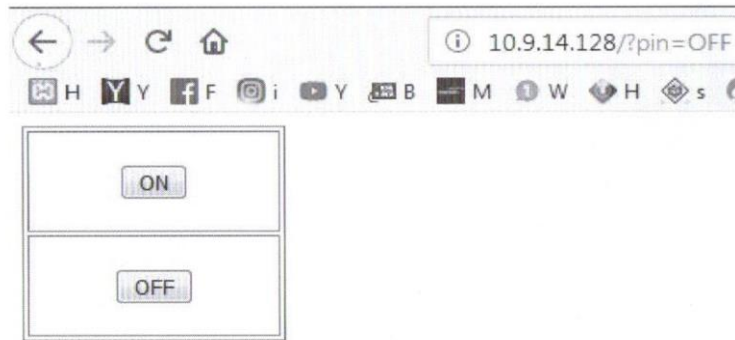
Berikut ini adalah kode HTML untuk menampilkan tombol dan menangani request bertipe GET.

```
buff = buff .. "<html>"
buff = buff .. "<head><title>LED on/off</title></head>"
buff = buff .. "<body>"
buff = buff .. "<table border=1 width=11% height=130>"
buff = buff .. "<tr><td width=90 align=center>"
buff = buff .. "<a href='\"?pin=OFF\"'><button>OFF</button></td></tr>"
buff = buff .. "</table>"
buff = buff .. "</body>"
buff = buff .. "</html>"
```

Dan terakhir digunakan untuk melakukan setting LED on atau off.

```
if (get1.pin == "ON") then
    gpio.write (LED, gpio.LOW)
elseif (get1.pin == "OFF") then
    gpio.write (LED, gpio.HIGH)
end
```

Hasilnya di browser dapat dilihat pada gambar 2.1



*Gambar 2.1 Tampilan di browser*

Untuk mengujinya, klik di tombol ON atau OFF, lalu amati LED yang ada di board.

Kekurangan dari cara ini adalah kontrol hanya bisa dilakukan jika ESP dan laptop berada dalam 1 jaringan.

Pada bab berikutnya kita membuat kontrol lewat internet, sehingga bisa dilakukan dimanapun di seluruh dunia.

### **Pertanyaan 3**

3. Jelaskan pengertian dari kontrol LED pada Internet of Things (IoT) menggunakan program web dan aplikasi Android.

### **Jawaban**

## **BAB III**

## **PROGRAM WEB DAN ANDROID UNTUK KONTROL LED**

### **BAB 3**

### **Program Web dan Android untuk Kontrol LED**

#### **3.1 Pengertian**

Kontrol LED pada Internet of Things (IoT) menggunakan program web dan aplikasi Android adalah suatu sistem yang memungkinkan pengguna untuk mengontrol LED melalui jaringan internet. Sistem ini memanfaatkan teknologi IoT yang memungkinkan perangkat elektronik terhubung ke internet dan saling berkomunikasi.

Pada sistem ini, LED dihubungkan ke mikrokontroler seperti Raspberry Pi atau Arduino yang terhubung ke jaringan internet. Kemudian, pengguna dapat mengontrol LED melalui program web atau aplikasi Android yang terhubung ke mikrokontroler tersebut.

Program web atau aplikasi Android tersebut dapat mengirimkan sinyal ke mikrokontroler untuk menghidupkan atau mematikan LED. Selain itu, pengguna juga dapat mengatur intensitas cahaya LED dengan mengirimkan sinyal PWM (*Pulse Width Modulation*) ke mikrokontroler.

Sistem ini sangat berguna untuk mengontrol lampu atau perangkat elektronik lainnya dari jarak jauh melalui internet. Selain itu, sistem ini juga dapat diintegrasikan dengan sensor dan perangkat lainnya untuk membuat sistem IoT yang lebih kompleks.





### 3.2 Kontrol LED pada Internet of Things

Kontrol LED pada Internet of Things (IoT) menggunakan program web dan aplikasi Android mengacu pada kemampuan untuk mengendalikan atau mengontrol lampu LED (*Light-Emitting Diode*) melalui jaringan internet atau jaringan lokal menggunakan dua jenis antarmuka pengguna: program web dan aplikasi Android.

#### 3.2.1 Kontrol LED Melalui Program Web

Program web adalah sebuah aplikasi yang diakses melalui peramban web (web browser) seperti Google Chrome, Mozilla Firefox, atau Safari. Pada aplikasi web ini, pengguna dapat mengendalikan lampu LED melalui antarmuka yang disediakan secara online.

Langkah-langkahnya meliputi:

Perangkat IoT yang terhubung ke lampu LED harus memiliki koneksi internet dan terhubung ke server atau cloud.

Pengguna membuka peramban web dan masuk ke alamat URL atau situs web khusus yang mengontrol lampu LED.

Di dalam aplikasi web, akan ada tombol atau kontrol lain yang memungkinkan pengguna untuk menyalakan atau mematikan lampu LED, mengubah kecerahan, atau mengubah warna lampu LED jika LED tersebut mendukungnya.

Ketika pengguna berinteraksi dengan tombol atau kontrol, perintah dikirimkan melalui internet ke perangkat IoT yang terhubung ke lampu LED, yang kemudian mengubah status atau pengaturan sesuai dengan perintah tersebut.

#### 3.2.2 Kontrol LED melalui Aplikasi Android

Aplikasi Android adalah aplikasi yang diinstal dan dijalankan pada perangkat berbasis sistem operasi Android, seperti smartphone atau tablet. Dalam konteks ini, aplikasi Android berfungsi sebagai antarmuka untuk mengontrol lampu LED secara nirkabel melalui koneksi internet atau jaringan lokal.

Langkah-langkahnya meliputi:

- Perangkat IoT yang terhubung ke lampu LED harus terhubung ke internet atau jaringan lokal.
- Pengguna menginstal aplikasi Android khusus pada perangkat mereka.
- Setelah aplikasi dijalankan, pengguna akan memiliki antarmuka yang menampilkan tombol atau kontrol untuk mengontrol lampu LED, seperti menghidupkan atau mematikan lampu, mengubah kecerahan, atau mengubah warna (jika lampu LED mendukungnya).
- Ketika pengguna berinteraksi dengan aplikasi Android ini, perintah dikirim melalui internet atau jaringan lokal ke perangkat IoT yang terhubung ke lampu LED. Perangkat IoT kemudian menginterpretasi perintah tersebut dan mengontrol lampu LED sesuai dengan instruksi.

Dengan adanya kemampuan kontrol LED melalui program web dan aplikasi Android, pengguna dapat dengan mudah mengelola dan mengontrol lampu LED mereka dari jarak jauh dan dengan cara yang lebih praktis, memberikan fleksibilitas dan kenyamanan dalam penggunaan perangkat IoT di rumah atau di lingkungan bisnis.

### 3.3 Implementasi Program





Sekarang kita membuat program di bagian user interfacenya. Ada 2 alternatif, yaitu:

- Program berbasis web
- Aplikasi mobile berbasis Android

### 3.3 Program Berbasis Web

Kita bisa membuat kode HTML sederhana dan kemudian menaruh file html tersebut ke hosting.

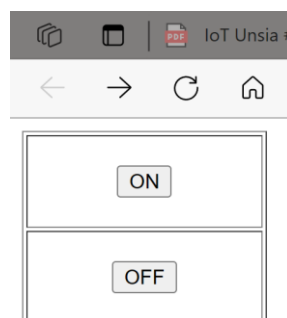
Program berisi 2 tombol, on dan off. Tombol tersebut diberi hyperlink yang akan mengeksekusi API seperti di bab sebelumnya. Agar tampilannya teratur, 2 tombol tersebut diletakkan dalam tabel.

Programnya cukup sederhana seperti ditampilkan di bawah, beri nama led. Html:

```
<html!>
<head><title>LED on/off</title></head>
<body>
<table border=1 width=11% height=130>
<tr><td width=90 align=center>
<a href="http://api.geeknesia.com/api/data?
api_key=15b2459853e0cleelcd7d5bf8024ffe0
&attributes={&quot;LedStat&quot;}&quot;on&quot;}">
<button>ON</button></a></td></tr>
<tr><td width=90 align=center>
<a href="http://api.geeknesia.com/api/data?
Api_key=15b2459853e0cleelcd7d5bf8024ffe0
&attributes={&quot;LedStat&quot;:&quot;off&quot;}">
<button>OFF</button></a></td></tr>
</table>
</body>
</html>
```

Letakkan programnya di sebuah hosting file di folder public\_html atau tergantung konfigurasi hosting yang Anda gunakan. Disini penulis menggunakan hosting sendiri di [www.hardana.com](http://www.hardana.com).

Jalankan di browser <http://www.hardana.com/led.html>, maka tampilannya akan terlihat seperti pada gambar 3.1.



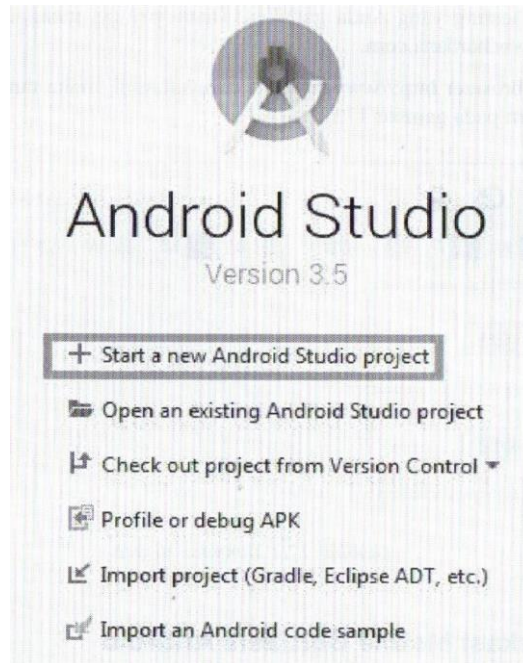


*Gambar 3.1 Tampilan di web*

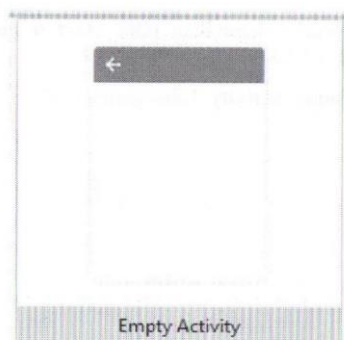
### 3.4 Aplikasi Mobile Berbasis Android

Selanjutnya akan dibahas program Android menggunakan bahasa Kotlin dengan IDE Android Studio 3.5. Berikut adalah langkah-langkahnya:

1. Jalankan Android Studio, kemudian pilih *Start a new Android Studio project*. Lihat gambar 3.2
2. Selanjutnya pilih *Empty Activity*. Lihat gambar 3.3.



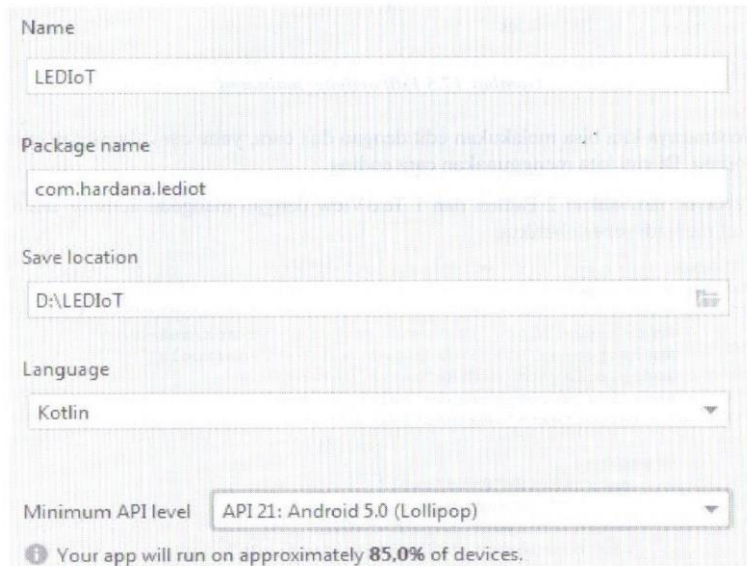
*Gambar 3.2 Start a new Android Studio Project*



*Gambar 3.3 Empty Activity*

3. Klik tombol *Next*, lalu isikan pada form project sesuai keinginan Anda, contohnya:
  - **Name:** LEDIoT adalah nama project.
  - **Package name:** com.hardana.lediot adalah nama domain Anda (bebas) + nama project.
  - **Save location:** disesuaikan dengan komputer Anda, bebas.
  - **Language:** Kotlin (karena disini kita menggunakan bahasa Kotlin).
  - **Minimum API level:** API 21: Android 5.0 (Lollipop)

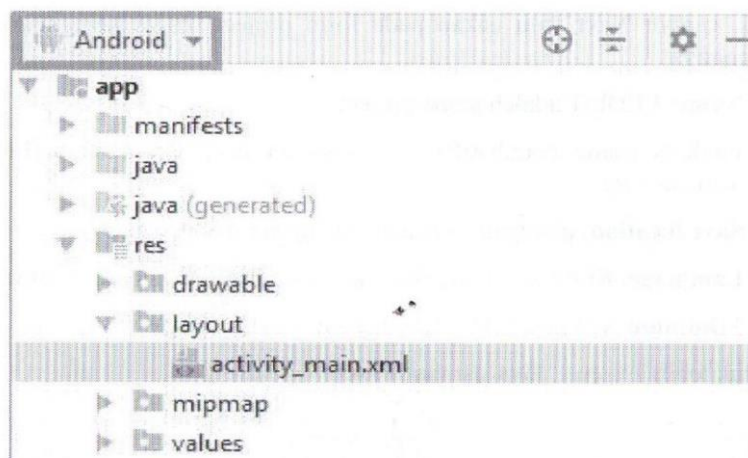
Untuk lebih jelasnya, lihat gambar 3.4.



**Gambar 3.4 Form isian project**

1. Klik tombol **Finish**. Tunggu hingga proses selesai.

Tahap selanjutnya, kita menuju tahap desain form. Pilih tab Android, lalu klik 2x pada **activity\_main.xml**. Lihat gambar 3.5.



**Gambar 3.5 Edit activity\_main.xml**

Sebenarnya kita bisa melakukan edit dengan dua cara, yaitu cara visual dan cara coding. Di sini kita menggunakan cara coding.

Pertama, tambahkan 2 Button dan 1 TextView dengan mengedit activity main.xml! menjadi seperti berikut:

```
<?xml: version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns: android=http://schemas.android.com/apk/res/android
    xmlns: tools=http://schemas.android.com/tools"
    android: layout_width="match_parent"
    android: layout_height="match_parent"
    tools: context=".MainActivity">

    <TextView
```

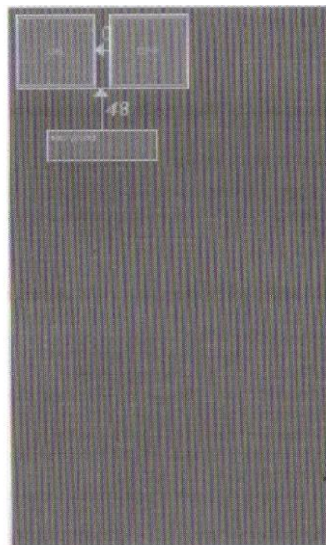


```
        android:id="@+id/txt1"
        android:layout_width="140dp"
        android:layout_height="40dp"
        android:layout_bellow="@id/buttonOn"
        android:layout_margin="@android:dimen/app_icon_size"
        android:bacground ="@android:color/holo_blue_bright"
        android:text="Hello World" />

<Button
    android:id="@+id/buttonOn"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:layout_margin="10dp"
    android:text="On" />

<Button
    android:id="@+id/buttonOff"
    android:layout_width="100dp"
    android:layout_height="100dp"
    android:layout_margin="10dp"
    android:layout_toEndOf="@id/buttonOn"
    android:text="Off" />
</RelativeLayout>
```

Dalam bentuk desain, tampilannya akan terlihat seperti pada gambar 3.6.



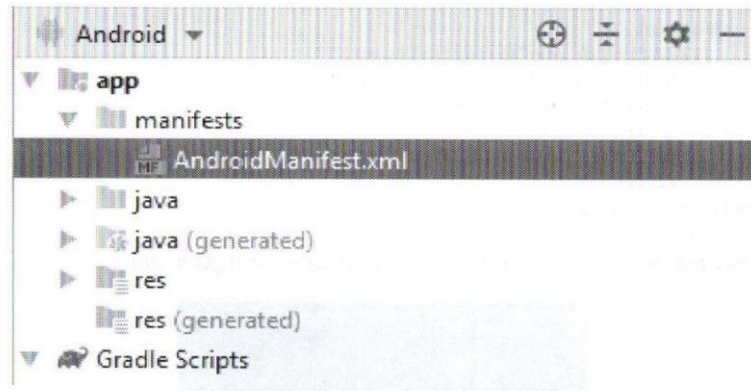
**Gambar 3.6 Desain form**

Deasin selesai. Lanjut ke langkah berikutnya.

Karena kita nantinya menggunakan Internet, maka tambahkan permission internet di AndroidManifest.xml.

```
<uses-permission android:name="android.permission.INTERNET" />
```

Oh ya, letak file AndroidManifest.xml dapat dilihat pada gambar 3.7.



**Gambar 3.7 Edit AndroidManifest.xml**

Adapun coding lengkapnya akan terlihat seperti berikut:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.hardana.lediot">

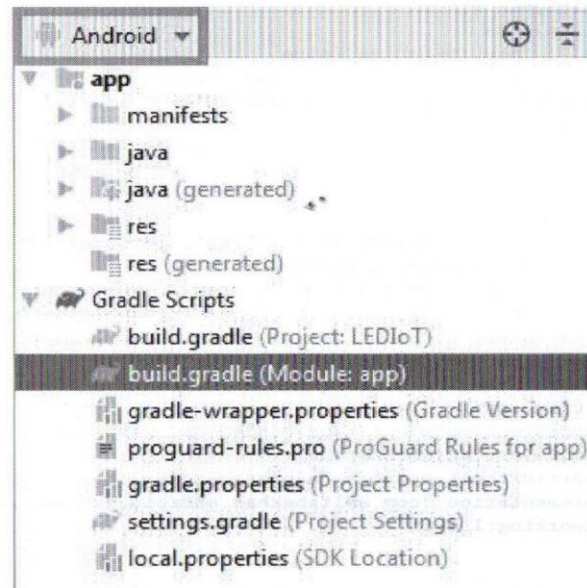
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundicon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Kita juga perlu menambahkan modul library eksternal untuk request http. Ada banyak pilihan, disini kita menggunakan library jackson android networking.

Untuk menambahkannya, buka dibagian build.gradle (Module: app). Lihat gambar 3.8.



**Gambar 3.8 Edit build.gradle**

Tambahkan code berikut:

```
implementation 'com.amitshekhar.android:jackson-android-networking:1.0.1'
```

Adapun coding lengkapnya akan terlihat seperti berikut:

```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.1"
    defaultConfig {
        applicationId "com.hardana.lediot"
        minSdkVersion 21
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
            "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile
                (';proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'org.jetbrains.kotlin:
        kotlin-stdlib-jdk7:$kotlin_version'
```



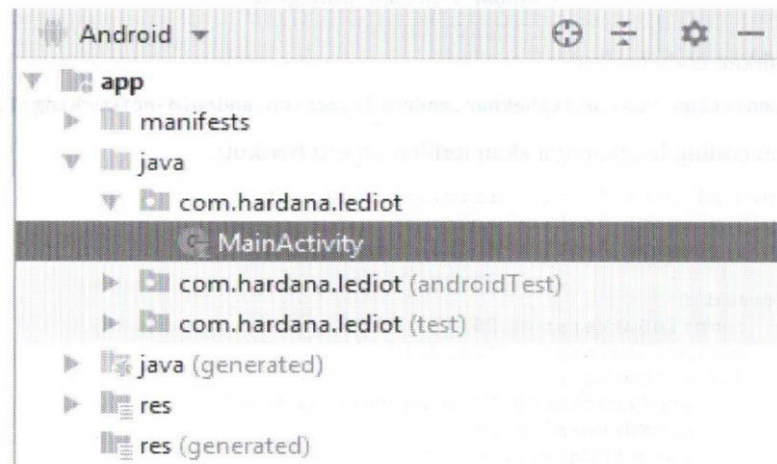


```

implementation 'androidx.appcompat:appcompat:1.0.2'
implementation 'androidx.core:core-ktx:1.0.2'
implementation 'androidx.constraintlayout:
constraintlayout:1.1.3'
testImplementation 'junit:junit:4.12'
androidTestImplementation 'androidx.test:runner:1.1.1'
androidTestImplementation
'androidx.test.espresso:espresso-core:3.1.1'
implementation 'com.amithshekhar.android:jackson-android-
networking:1.0.1'
}

```

Terakhir, kita ke bagian program utama. Edit **MainActivity**. Lihat gambar 3.9



**Gambar 3.9 Edit MainActivity**

Selanjutnya kita menuju ke program utama. Tambahkan event pada tombol ON dan OFF di MainActivity.kt di bagian **buttonOn.setOnClickListener** dan **buttonOn.setOnOffClickListener**. Keduanya mirip, hanya beda di bagian URL-nya.

```

package com.hardana.lediot
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.androidnetworking.AndroidNetworking
import com.androidnetworking.error.ANError
import com.androidworking.interface
.JSONObjectttRequestListener
import kotlinx.android.synthetic.main.activity_main.*
import org.json.JSONObject

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        buttonOn.setOnClickListener {
            val urlOn = "http://api.geeknesia.com/api/data?
            api_key=15b2459853e0cleelcd7d5bf8024ffe0
            &attributeView={\"LedStat\": \"on\"}"
            AndroidNetworking.get(urlOn)
                .setPriority(Priority.MEDIUM)

```





```
        .build()
    .getAsJSONObject(object : JSONObjectRequestListener {
        override fun onResponse(response: JSONObject?) {
            txt1.text = "Berhasil"
        }
        override fun onError(anError: ANError?) {
            txt1.text = if (anError?.errorCode == 0)
            "Berhasil" else anError?.errorDetail
        }
    })
}

buttonOff.setOnClickListener {
    val urlOff = "http://api.geeknesia.com/api/data?
    api_key=15b2459853eocleelcd7d5bf8024ffe0
    &attributes=(\"LedStat\": \"off\")"
    AndroidNetworking.get(urlOff)
        .setPriority(Priority.MEDIUM)
        .build()
    .getAsJSONObject(object : JSONObjectRequestListener {
        override fun onResponse(response: JSONObject?) {
            txt1.text = "Berhasil"
        }
        override fun onError(anError: ANError?) {
            txt1.text = if (anError?.errorCode == 0)
            "Berhasil" else anError?.errorDetail
        }
    })
}
}
```



**Gambar 3.10 Tampilan saat dijalankan**



### Penjelasan:

- Eksekusi: dilakukan dengan perintah **AndroidNetworking.get(URL)** dengan parameter **Priority.MEDIUM**
- Response ditangkap oleh **override fun onResponse** dan jika terjadi error ditangkap di **override fun onError**.
- **txt1.text** di-assign dengan ekspresi if, jika tidak terjadi error maka di isi dengan "Berhasil" dan jika gagal di isi dengan **anError?.errorDetail**.

Saat dijalankan, hasilnya akan terlihat seperti pada gambar 3.10.

### 3.5 Menambahkan Tombol Cek Status

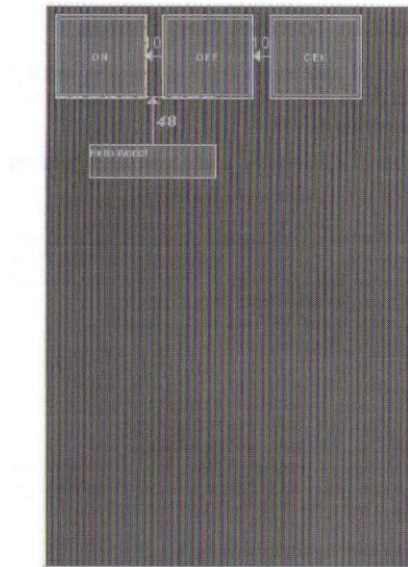
Secara fungsi, program di atas sudah dapat digunakan untuk menyalakan dan mematikan LED. Kita perlu satu fungsi atau tombol lagi, yaitu untuk mengecek status LED, apakah menyala atau mati.

Tambahkan tombol Cek di file **activity\_main.xml** sehingga menjadi:

```
<?xml version="1.0" encoding="utf-8" ?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res-android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/txt1"
        android:layout_width="140dp"
        android:layout_height="40dp"
        android:layout_below="@id/buttonOn"
        android:layout_margin="@android:dimen/app_icon_size"
        android:background="@android:color/holo_blue_bright"
        android:text="Hello World!" />
    <Button
        android:id="@+id/buttonOn"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_margin="10dp"
        android:text="On" ?>
    <Button
        android:id="@+id/buttonOff"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_margin="10dp"
        android:layout_toEndOf="@id/buttonOn"
        android:text="Off" />
    <Button
        android:id="@+id/buttonCek"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_margin="10dp"
        android:layout_toEndOf="@id/buttonOff"
        android:text="Cek" />
</RelativeLayout>
```



Sekarang tampilan desainnya akan terlihat seperti pada gambar 3.11.



*Gambar 3.11 Desain form setelah ditambahkan tombol Cek*

Tambahkan **buttonCek.onClickListener** di **MainActivity.kt**, sehingga menjadi:

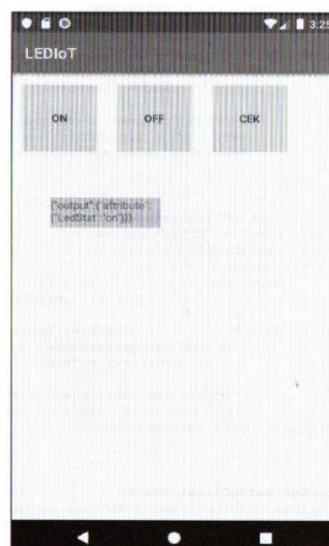
```
package com.hardana.lediot
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.androidnetworking.AndroidNetworking
import com.androidnetworking.common.Priority
import com.androidnetworking.error.ANError
import com.androidworking.interfaces
.JSONObjectRequestListener
import kotlin.android.synthetic.main.activity_main.*
import org.json.JSONObject
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        buttonOn.setOnClickListener {
            val urlOn = "http://api.geeknesia.com/api/data?
            api_key=15b2459853e0cleelcd7d5df8024ffe0
            &attributes={\"LedStat\": \"on\"}"
            AndroidNetworking.get(urlOn)
                .setPriority(Priority.MEDIUM)
                .build()
                .getAsJSONObject(object : JSONObjectRequestListener {
                    override fun onResponse(response: JSONObject?) {
                        txt1.text = "Berhasil"
                    }
                    override fun onError(anError: ANError?) {
                        txt1.text = if (anError?.errorCode == 0)
                        "Berhasil" else anError?.errorDetail
                    }
                })
        }
    }
}
```



```
buttonOff.setOnClickListener {
    val urlOff = "http://api.geeknesia.com/api/data?
    api_key=15b2459853e0cleelcd7d5bf8024ffe0
    &attributes={\"LedStat\": \"off\"}"
    AndroidNetworking.get(urlOff)
        .setPriority(Priority.MEDIUM)
        .build()
    .getAsJSONObject (object : JSONObjectRequestListener {
        override fun onResponse(response: JSONObject?) {
            txt1.text = "Berhasil"
        }
        override fun onError (anError: ANError?) {
            txt.text = if (anError?.errorCode == 0)
"Berhasil" else anError?.errorDetail
        }
    })
}

buttonCek.setOnClickListener {
    val urlCek = "http://api.geeknesia.com/api/attribute/
    LedStat?api_key=15b2459853e0cleelcd7d5bf8024ffe0"
    AndroidNetworking.get(urlCek)
        .setPriority(Priority.MEDIUM)
        .build()
    .getAsJSONObject (object : JSONObjectRequestListener {
        override fun onResponse (response: JSONObject?) {
            txt.text = response.toString()
        }
        override fun onError(anError: ANError?) {
            //txt.text = if (anError?.errorCode == 0)
            "Berhasil" else anError?.errorDetail
        }
    })
}
}
```

Adapun hasilnya dapat dilihat pada gambar 3.12.



**Gambar 3.12 Hasil penambahan CEK Status**





### Referensi

Hardana, R. F. (2019). Membuat Aplikasi IoT (Internet of Things). Dalam Hardana, *Membuat Aplikasi IoT (Internet of Things)* (hal. 2-8). Yogyakarta: CV. LOKOMEDIA.

Indrajit, N. d. (2022). *Internet of Things (IoT)*. Yogyakarta: CV ANDI.

### Link File Arduino

[https://drive.google.com/file/d/1ICCj-FABXom\\_evwnFDxD729Dx\\_Ca0GJb/view?usp=sharing](https://drive.google.com/file/d/1ICCj-FABXom_evwnFDxD729Dx_Ca0GJb/view?usp=sharing)

Nilai	Tanda Tangan Dosen Pengampu	Tanda Tangan Mahasiswa
	 (Anjrah Mintana,M.Kom)	 (Hendro Gunawan)
Diserahkan pada Tanggal :		Tanggal Mengumpulkan :
		03/07/2023

