PENGGERAK AKTUATOR PENGENDALI DAYA LISTRIK DALAM RUANGAN BERBASIS AKTIVITAS MANUSIA

SKRIPSI

TEKNIK ELEKTROKONSENTRASI TEKNIK ELEKTRONIKA

Ditujukan untuk memenuhi prasyarat
memperoleh gelar Sarjana Teknik

RIZQURROHMAN ABDUL AZIZ CHOLIS
NIM. 145060300111045

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2021

RINGKASAN

Rizqurrohman Abdul Aziz Cholis, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Juli 2021, *Sistem Deteksi Manusia dengan Aktuator dalam Ruangan Sebagai Pendekatan Solusi Smart Home,* Dosen Pembimbing: Eka Maulana dan Nurussa'adah.

Smart home merupakan teknologi yang sampai saat ini masih terus dikembangkan. Salah satu aplikasi dari *smart home* ialah seperti identifikasi objek menggunakan kamera yang dihubungkan dengan jaringan *internet* pengguna dalam rumah. Seiring berkembangnya teknologi, identifikasi objek menggunakan kamera ini bahkan sudah bisa dipakai untuk mengenali aktivitas / perilaku manusia seperti berdiri, duduk, berjalan, ataupun berlari. Hal ini menumbuhkan potensi untuk bisa mengembangkan teknologi *smart home* untuk menghubungkan identifikasi perilaku manusia ini dengan suatu aksi yang bermanfaat bagi pengguna teknologi ini seperti menghemat energi listrik ataupun menambah kenyamanan pengguna *smart home* di dalam rumah.

Pada penelitian ini sistem *smart home* didesain menggunakan kamera berjenis *webcam* sebagai input, *Raspberry Pi* sebagai unit prosesor, *web server* jenis *Node.js* dengan bahasa pemrograman *javascript* dan bahasa *C*, *ESP-01* berbasis *chip ESP8266* sebagai unit kontrol, modul *relay IoT (Internet of Things)*, serta beberapa aktuator berupa perangkat listrik dalam rumah seperti lampu, kipas angin, atau TV / monitor. Seluruh perangkat tergabung dalam satu jaringan *internet* lokal secara *wireless*. Sistem akan mendeteksi perilaku manusia di dalam ruangan melalui input dari kamera lalu diproses pada unit prosesor dan kemudian memberikan instruksi ke unit kontrol untuk mengaktifkan / mematikan aktuator dalam ruangan secara *wireless*. Pengguna juga bisa memberikan input kepada *web server* di dalam *Raspberry Pi* dan memberikan instruksi kepada unit kontrol *ESP-01* untuk menyalakan / mematikan beberapa aktuator sekaligus dan dengan demikian sistem *smart home* dapat meningkatkan efisiensi penggunaan energi listrik dalam rumah.

Pada penelitian ini terdapat 3 metode pengujian yaitu pengujian *loading time* mengaktifkan *web server*, pengujian *response time* dari *web server* menuju unit kontrol, dan pengujian efisiensi energi listrik pada aktuator. Dari hasil pengujian ini kemudian dianalisis karakteristik hasil pengujiannya, di mana *loading time web server* untuk aktif dari *Raspberry Pi* berada pada rentang waktu antara 4,35 - 4,6 detik, *response time* web server menuju unit kontrol untuk mengaktifkan / mematikan aktuator paling lambat pada rentang 9-10 detik, dan sistem ini mampu meningkatkan efisiensi penggunaan energi listrik hingga 81% per jam. Diharapkan pada pengembangan dari penelitian ini dapat menjadikan kamera sebagai input pemicu perubahan mode dari sistem *smart home* menurut ada / tidaknya aktivitas manusia di dalam ruangan.

Kata kunci: *smart home, raspberry pi, web server, esp8266,* energi

SUMMARY

Rizqurrohman Abdul Aziz Cholis, *Departement of Electrical Engineering, Faculty of Engineering, University of Brawijaya, July 2021, Human Behavior Detection System with Indoor Actuator as a Smart Home Solution Approach, Academic Supervisor:* Eka Maulana *and* Nurussa'adah.

Smart home is a technology that is still being developed. One application of a smart home is like object identification using a camera that is connected to the user's internet network at home. As technology develops, object identification using this camera can even be used to recognize human activities/behaviors such as standing, sitting, walking, or running. This raises the potential to be able to develop smart home technology to link the identification of human behavior with an action that is beneficial for users of this technology, such as saving electrical energy or increasing the comfort of smart home users at home.

In this study, the smart home system was designed using a webcam type camera as input, Raspberry Pi as a processor unit, Node.js type web server with javascript programming language and C language, ESP-01 based on the ESP8266 chip as a control unit, IoT (Internet of Things), as well as several actuators in the form of electrical devices in the house such as lights, fans, or TV / monitors. All devices are joined in one local internet network wirelessly. The system will detect human behavior in the room through input from the camera and then process it on the processor unit and then give instructions to the control unit to activate/deactivate the indoor actuators wirelessly. Users can also provide input to the web server on the Raspberry Pi and give instructions to the ESP-01 control unit to turn on/off several actuators at once and thus the smart home system can increase the efficiency of electrical energy use in the home.

In this study, there are 3 testing methods, namely testing the loading time to activate the web server, testing the response time from the web server to the control unit, and testing the electrical energy efficiency of the actuator. From the results of this test, the characteristics of the test results are then analyzed, where the loading time for the active web server from the Raspberry Pi is in the range of 4.35 - 4.6 seconds, the response time for the web server to the control unit to activate / deactivate the actuator is the slowest in the range 9-10 seconds, and this system is able to increase the efficiency of using electrical energy by up to 81% per hour. It is hoped that the development of this research can make the camera as an input trigger for changing the mode of the smart home system according to the presence / absence of human activity in the room.

Keywords: *smart home, raspberry pi, web server,esp8266, energy*

PENGANTAR

Puji syukur *Alhamdulillah* saya ucapkan karena masih diberikan kesehatan dan kesempatan sehingga saya dapat menyelesaikan penulisan penelitian skripsi ini. Saya mengucapkan terima kasih banyak kepada ibu dan bapak saya, kakak dan adik saya, keluarga saya, bapak / ibu dosen dan karyawan TEUB – FTUB, teman-teman serta seluruh pihak yang sudah membantu, mendukung, menyemangati dan memotivasi saya dalam proses pengerjaan skripsi ini karena dengan itu semua saya akhirnya bisa menyelesaikan studi saya dan membuat penulisan skripsi ini. Mudah-mudahan apapun usaha, tenaga, waktu, pikiran, dan materi yang telah anda semua berikan kepada saya sekecil apapun itu akan dibalas dengan yang jauh lebih baik dan bisa bermanfaat bagi kehidupan kita semua.

Saya menyadari dan mengakui bahwa apa yang saya pelajari dan saya tuliskan di skripsi ini masih memiliki kesalahan dan kekurangan, namun semoga dengan selesainya penulisan skripsi ini dapat menambah pengetahuan dan menjadi ilmu yang bermanfaat bagi siapapun yang membaca / memahami isi di dalam penulisan skripsi ini. Terima kasih.

Malang, 14 Juli 2021

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Teknologi ciptaan manusia hingga saat ini masih terus dikembangkan agar mampu menambah produktivitas, menghemat energi dan sumberdaya, meningkatkan keamanan dan efisiensi waktu, ataupun sebatas menambah kenyamanan beraktivitas / membantu mengoptimalkan kehidupan sehari-hari. Salah satu konsep teknologi yang mendukung hal-hal tersebut serta masih berkembang hingga saat ini ialah *Smart Home* atau Rumah Pintar, sebuah hunian yang dilengkapi dengan teknologi pintar yang bertujuan untuk menyediakan layanan yang disesuaikan bagi penggunanya (Marikyan et al., 2019). Teknologi pintar tersebut mencakup hal-hal seperti otomatisasi / pengendalian jarak jauh perangkat elektronik, detektor bahaya dalam rumah yang terintegrasi dengan jaringan pengguna (jaringan *wireless* / *internet*) untuk memberikan peringatan, dan lain sebagainya.

Salah satu aplikasi dari *Smart home* ialah identifikasi objek menggunakan kamera yang terhubung dengan jaringan lokal pengguna dalam rumah. Seiring perkembangan teknologi, objek yang bisa diidentifikasi oleh kamera juga termasuk pola perilaku dinamis manusia yang umum ditemui seperti posisi berdiri, duduk, ataupun sedang berjalan. Sebuah riset dari (Wang et al., 2019) menunjukkan tentang beberapa metode deteksi pola perilaku manusia yang dinamis dan klasifikasi berdasarkan data video secara *real-time*. Hal ini menumbuhkan potensi bahwa identifikasi perilaku manusia ini dapat dihubungkan dengan suatu aksi yang bermanfaat untuk membantu kehidupan manusia seperti memberikan kenyamanan lebih dalam beraktivitas ataupun meningkatkan efisiensi penggunaan energi listrik. Salah satu cara untuk mendukung hal tersebut ialah dengan menghubungkan sistem kamera identifikasi perilaku manusia dengan aktuator dalam ruangan seperti menyalakan / mematikan lampu, kipas angin, ataupun jenis aktuator lainnya secara otomatis sehingga orientasi pendekatan solusi *Smart home* dapat dicapai.

Dengan demikian maka penulis ingin merancang sebuah sistem deteksi perilaku manusia dengan aktuator dalam ruangan agar mampu meningkatkan efisiensi energi listrik serta kenyamanan beraktivitas pengguna dalam rumah.

1.2 Rumusan Masalah

1. Bagaimana desain sistem penggerak aktuator pengendali daya listrik dalam ruangan berbasis aktivitas manusia yang dapat meningkatkan efisiensi energi listrik dalam rumah?
2. Bagaimana karakteristik hasil pengujian sistem penggerak aktuator pengendali daya listrik dalam ruangan berbasis aktivitas manusia untuk meningkatkan efisiensi energi listrik dalam rumah?

1.3 Batasan Masalah

1. Unit prosesor untuk sistem *smart home* menggunakan *Raspberry pi* 3B
2. Unit kontrol untuk aktuator adalah *ESP8266*
3. Komunikasi sistem *smart home* menggunakan *web server Node.js*

1.4 Tujuan

1. Untuk membuat desain sistem penggerak aktuator pengendali daya listrik dalam ruangan berbasis aktivitas manusia yang dapat meningkatkan efisiensi energi listrik dalam rumah.
2. Untuk menganalisis karakteristik hasil pengujian sistem penggerak aktuator pengendali daya listrik dalam ruangan berbasis aktivitas manusia untuk meningkatkan efisiensi energi listrik dalam rumah.

1.5 Manfaat

1. Membantu menghemat penggunaan listrik dalam rumah
2. Menambah kenyamanan dalam penggunaan alat listrik dalam rumah dalam hal menghemat waktu dan tenaga

BAB II
TINJAUAN PUSTAKA

2.1 *Smart Home*

Setiap rumah memiliki banyak perangkat rumah seperti sistem kelistrikan (misalnya, Sakelar lampu, televisi, radio, dll.), Sistem mekanis (misalnya, jendela, pintu, kunci pintu, dll.), Sistem komunikasi (misalnya, sistem keamanan, jaringan area lokal (LAN) , dll.), dan sistem hiburan (misalnya, tele vision, sistem home theater, dll.). Rumah pintar (atau rumah yang terhubung) dapat menyatukan semua perangkat rumah ini. Dengan kabel kontrol dari setiap perangkat rumah ke satu titik, umumnya dikenal sebagai *overlay* sistem kontrol, perangkat rumah dapat diintegrasikan dan dibuat *interoperable*. *Overlay* sistem kontrol menyediakan antarmuka tunggal dan terpadu untuk kontrol perangkat rumah individu. Misalnya, perangkat yang sama yang digunakan untuk mengontrol pencahayaan interior rumah juga dapat digunakan untuk mengatur panas di ruang tamu, mematikan stereo, menyalakan televisi, dll. (Ortiz & Xia, 2013)

Menurut (Schiefer, 2015) terdapat 15 kategori perangkat dalam rumah yang bisa dimasukkan dalam kategori perangkat *Smart home* menurut fungsinya. Beberapa kategori diambil sebagai acuan untuk menunjang tujuan penelitian ini yaitu:

- Sistem Pengendalian (*ConS*)

  Produk seperti *base stations*, sistem yang hanya mengontrol *Smart home* seperti tablet atau smartphone dan aplikasi khusus yang sejalan dengan tujuan tersebut termasuk dalam kategori ini.

- Pemanasan, Ventilasi dan Penyejuk Udara (*HVAC*)

  Sistem kategori ini digunakan untuk mengatur suhu ruangan dan ventilasi udara seperti termostat, pengatur suhu unit atau ventilator.

- Cahaya dan Bayangan (*LaS*)

  Perangkat yang memancarkan atau mencegah cahaya, seperti lampu, tenda, dan tirai lipat adalah bagian dari kategori ini.

- Hiburan (*Ent*)

  Perangkat seperti sistem audio, televisi, konsol *game*, robot mainan dan lainnya yang berhubungan dengan *entertainment* masuk dalam kategori ini.

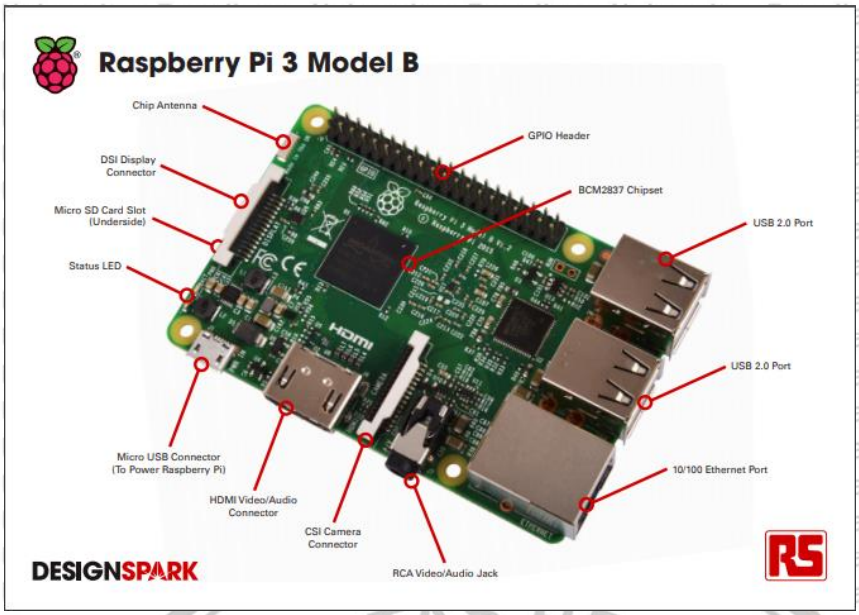
2.2 Deteksi Perilaku Manusia Dengan Kamera

Deteksi perilaku manusia yang dinamis berdasarkan video menjadi topik penelitian hangat dalam *computer vision* dan *machine learning* karena aplikasinya yang luas dalam memahami dan mendeteksi perilaku manusia. Pengaplikasian deteksi perilaku manusia umum digunakan untuk keperluan seperti lingkungan pengawasan, lingkungan hiburan dan sistem perawatan kesehatan. Dalam lingkungan pengawasan, deteksi otomatis aktivitas abnormal dapat digunakan untuk memperingatkan otoritas terkait tentang potensi perilaku kriminal atau berbahaya, seperti pelaporan otomatis dari seseorang dengan tas berkeliaran di bandara atau stasiun. Demikian pula, dalam lingkungan hiburan, pengenalan aktivitas dapat meningkatkan interaksi komputer manusia (HCI), seperti pengenalan otomatis atas tindakan pemain yang berbeda selama pertandingan tenis sehingga dapat membuat avatar di komputer untuk bermain tenis bagi pemain tersebut. Lebih lanjut, dalam sistem kesehatan, pengenalan aktivitas dapat membantu proses rehabilitasi pasien, seperti pengenalan otomatis tindakan pasien untuk memfasilitasi proses rehabilitasi. (Ke et al., 2013)

Deteksi perilaku manusia dengan kamera ini dapat menggunakan kamera dengan jenis *USB webcam* dengan spesifikasi resolusi video maksimal pada *1080p* dengan *frame rate* 30 *frame per second* serta sistem *plug-and-play* didalamnya sehingga tidak memerlukan instalasi perangkat lunak tambahan secara manual dan akan terdeteksi serta terinstal di dalam unit prosesor yang akan digunakan secara otomatis. Dengan spesifikasi ini maka diharapkan kamera dapat menangkap hasil gambar perilaku manusia di dalam ruangan yang optimal untuk kemudian diproses di dalam unit prosesor pada desain sistem *smart home*.

Gambar 2.1. USB Webcam

2.3 *Raspberry pi* 3B



Gambar 2.2. Raspberry pi 3 Model B

Raspberry pi 3B menggunakan chipset Broadcom BCM2387 yang memiliki 64 bit quad core prosesor dengan Cortex-A53 yang dapat berjalan pada kecepatan 1,2 GHz, memiliki RAM sebesar 1GB LPDDR2 dan berjalan dengan sistem operasi Linux atau Windows 10 *IoT* yang di boot melalui kartu Micro SD. *Raspberry pi* 3B dapat dihubungkan dengan perangkat lain seperti 2.4GHz dan 5GHz IEEE 802.11.b/g/n/ac wireless, LAN (Local Access Network), Bluetooth 4.2, BLE, Gigabit Ethernet melalui USB 2.0, dan 4×port USB 2.0. Selain itu, perangkat ini memiliki 40 pin GPIO header. *Raspberry pi* 3B memiliki daya masukan sebesar 5V/2,5A melalui micro USB connector, 5V DC melalui GPIO header, dan Power over Ethernet (PoE) (membutuhkan PoE HAT). Perangkat ini mendukung multimedia H.264, MPEG-4 decode (1080p30), H.264 encode (1080p30), dan OpenGL ES 1.1, 2.0 graphics. (Raspberry, 2012)

2.4 *ESP8266*

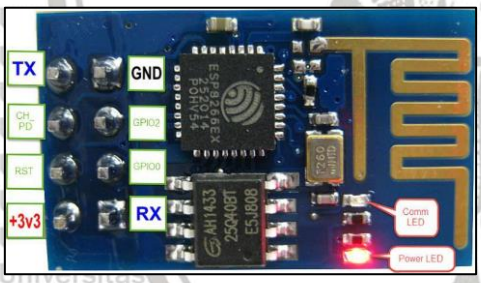
ESP8266 menawarkan solusi jaringan Wi-Fi lengkap dan mandiri, memungkinkannya untuk menghosting aplikasi atau memindahkan semua fungsi jaringan Wi-Fi dari prosesor aplikasi lain. Saat *ESP8266* menghosting aplikasi, dan jika itu adalah satu-satunya prosesor aplikasi di perangkat, ia dapat melakukan boot langsung dari flash eksternal. *ESP8266* memiliki cache terintegrasi untuk meningkatkan kinerja sistem dalam aplikasi tersebut, dan untuk meminimalkan kebutuhan memori. Selain itu *ESP8266* berfungsi sebagai adaptor Wi-Fi, akses *internet* nirkabel dapat ditambahkan ke desain

berbasis mikrokontroler dengan konektivitas sederhana melalui antarmuka UART atau antarmuka jembatan CPU AHB. (Espressif Systems, 2013)

Gambar 2.3. Chip *ESP8266* + WiFi

2.4.1 *ESP-01*

Modul *ESP-01* dikembangkan oleh prosesor inti Tim Ai-thinker *ESP8266*, yaitu *ESP8266*EX, dalam ukuran yang lebih kecil dari modul yang merangkum Tensilica L106 mengintegrasikan mikro MCU 32-bit daya ultra rendah yang terkemuka di industri, dengan mode pendek 16-bit, kecepatan clock mendukung 80 MHz , 160 Mhz, mendukung RTOS, Wi-Fi MAC/bb/RF/PA/LNA terintegrasi, antena terpasang. Pengguna dapat menggunakan modul tambahan ke jaringan perangkat yang ada, atau membangun pengontrol jaringan terpisah.



Gambar 2.4. ESP-01 berbasis chip *ESP8266*

Modul ini memiliki fitur-fitur sebagai berikut:

- 802.11 b/g/n
- MCU 32-bit daya rendah terintegrasi
- ADC 10-bit terintegrasi
- Tumpukan protokol TCP/IP terintegrasi
- Saklar TR terintegrasi, balun, LNA, penguat daya, dan jaringan yang sesuai
- PLL terintegrasi, regulator, dan unit manajemen daya
- Mendukung keragaman antena
- Wi-Fi 2,4 GHz, mendukung WPA/WPA2

- Mendukung mode operasi STA/AP/STA+AP
- Mendukung Fungsi Tautan Cerdas untuk perangkat Android dan iOS
- Mendukung Fungsi Tautan Cerdas untuk perangkat Android dan iOS
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM,
- STBC, 1x1 MIMO, 2x1 MIMO
- Agregasi A-MPDU & A-MSDU dan interval penjaga 0,4 detik
- Daya tidur nyenyak <10uA, Arus bocor daya mati <5uA
- Bangun dan kirim paket dalam <2ms
- Konsumsi daya siaga <1.0mW (DTIM3)
- Daya keluaran +20dBm dalam mode 802.11b
- Kisaran suhu pengoperasian -40C ~ 125C

*ESP8266*EX menawarkan solusi jaringan Wi-Fi yang lengkap dan mandiri; chip ini dapat digunakan untuk meng-host aplikasi atau untuk membongkar fungsi jaringan Wi-Fi dari prosesor aplikasi lain. Ketika ESP826EX menghosting aplikasi, itu boot langsung dari flash eksternal. Chip ini telah terintegrasi cache untuk meningkatkan kinerja sistem dalam aplikasi tersebut. Sebagai alternatif, berfungsi sebagai adaptor Wi-Fi, akses *internet* nirkabel dapat ditambahkan ke desain berbasis mikrokontroler apa pun dengan konektivitas sederhana (antarmuka SPI/SDIO atau I2C/UART). (Espressif Systems, 2013)

2.5 Modul *Relay IoT* untuk *ESP-01*

Relay adalah sakelar yang dioperasikan secara elektromagnetis. Arus penggerak pada kumparan mengoperasikan satu atau lebih kontak atau sirkuit beban yang terpisah secara galvanis. *Relay* elektro mekanik adalah sakelar yang dikendalikan dari jarak jauh yang mampu mengalihkan beberapa rangkaian, baik secara individual, secara bersamaan, atau secara berurutan. (*Relay*s & Design, n.d.)

Aplikasi umum untuk relai termasuk instrumen laboratorium, sistem telekomunikasi, antarmuka komputer, peralatan rumah tangga, AC dan pemanas, listrik otomotif, kontrol lalu lintas, kontrol pencahayaan, kontrol bangunan, kontrol daya listrik, mesin bisnis, kontrol motor dan solenoida, mesin perkakas, peralatan produksi dan pengujian.

Gambar 2.5. Modul *Relay IoT* untuk *ESP-01*

Gambar 2.6. Modul *Relay IoT* untuk *ESP-01* posisi terpasang

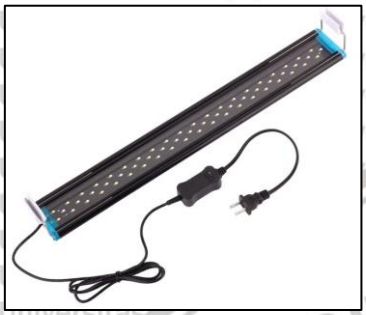
Modul *relay IoT* untuk *ESP-01* ini merupakan sebuah modul yang berisikan rangkaian *relay* DC 5V serta *socket* untuk unit *ESP-01* di mana *ESP-01* ini dapat mengendalikan *relay* DC 5V untuk mati atau nyala sesuai dengan program yang sudah diunggah sebelumnya ke dalam *ESP-01*. *Relay* yang terpasang pada modul ini dapat men-*drive* beban hingga 10A 250V AC atau 10A 30V DC.

2.6 Aktuator

2.6.1 Lampu *LED*

Light Emitting Diode, atau yang biasa disebut *LED*, adalah dioda semikonduktor yang memancarkan spektrum cahaya sempit yang tidak koheren ketika dibias secara elektrik ke arah depan sambungan p-n, seperti pada rangkaian *LED* umum. Efek ini adalah bentuk electroluminescence. *LED* biasanya merupakan sumber cahaya area kecil, seringkali dengan optik yang ditambahkan ke chip untuk membentuk pola radiasinya dan membantu refleksi. *LED* sering digunakan sebagai indikator kecil lampu di perangkat elektronik dan semakin banyak digunakan dalam aplikasi daya yang lebih tinggi seperti lampu senter dan pencahayaan area. Warna cahaya yang dipancarkan tergantung pada komposisi dan kondisi bahan semikonduktor yang digunakan, dan dapat berupa inframerah, sinar tampak, atau ultraviolet. (Singh, 2009)

Gambar 2.7. Lampu *LED* bohlam

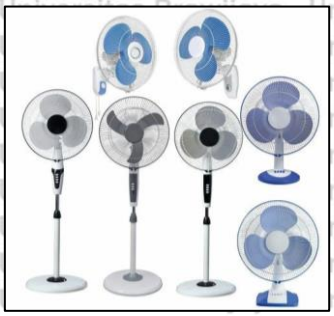


Gambar 2.8. Lampu *LED SMD* / Bar

Lampu *LED* bohlam untuk ruangan / teras rumah pada umumnya serta lampu *LED SMD (Surface Mounting Device)* / bar untuk aquarium ikan hias termasuk sebagai aktuator *smart home* dalam kategori Cahaya dan Bayangan (LaS).

2.6.2 Kipas Angin

Kipas angin adalah mesin bertenaga listrik yang digunakan untuk memindahkan sejumlah volume gas. Dengan kata lain, kipas angin adalah alat penggerak udara. Dalam bentuknya yang paling sederhana, kipas terdiri dari motor, rumah kipas dan *impeller* yang berputar. Desain *impeller* kipas, kecepatan putarannya, dan daya yang disuplai oleh motornya menentukan volume udara yang dapat digerakkan oleh kipas dan jumlah tekanan yang dapat dikembangkannya dalam membantu mengatasi hambatan aliran udara yang ada. (Fan & Impeller, n.d.)



Gambar 2.9. Kipas angin

Tipe kipas angin berdiri yang mampu membantu ventilasi dalam ruangan yang cukup luas termasuk sebagai aktuator *smart home* dalam kategori Pemanasan, Ventilasi, dan Penyejuk Udara (HVAC).

2.6.3 TV / Monitor

Monitor adalah perangkat keras yang dipakai untuk menampilkan output data grafis yang berasal dari sumber-sumber data grafis seperti *CPU*, satelit, dan berbagai sumber data grafis lainnya. Dalam perangkat komputer, monitor sering juga disebut dengan istilah layar komputer. Sedangkan di perangkat jaringan televisi, monitor sering kali disebut dengan istilah layar televisi. TV / monitor termasuk sebagai aktuator *smart home* dalam kategori Hiburan (Ent).

Gambar 2.10. TV / Monitor

2.7 *Web server*

Web server adalah sebuah software yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari *HTTP* atau *HTTPS* pada *client* yang dikenal dan biasanya kita kenal dengan nama *web browser* dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman web dan pada umumnya akan berbentuk dokumen *HTML*. Dalam bentuk sederhana *web server* akan mengirim data HTML kepada permintaan *web browser* sehingga akan terlihat seperti pada umumnya yaitu sebuah tampilan website.

Fungsi utama *web server* adalah untuk melakukan atau akan tranfer berkas permintaan pengguna melalui protokol komunikasi yang telah ditentukan sedemikian rupa. Halaman web yang diminta terdiri dari berkas teks, video, gambar, file dan banyak lagi. pemanfaatan *web server* berfungsi untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman web termasuk yang di dalam berupa teks, video, gambar atau banyak lagi.

Beberapa jenis *web server* di antaranya adalah:

1. Apache Web server / The HTTP Web server

2. *Apache Tomcat*
3. *Microsoft windows Server 2008 IIS (Internet Information Services)*
4. *Lighttpd*
5. *Zeus Web server*
6. *Sun Java System Web server*
7. *Node.js*

2.7.1 *Node.js*

Node.js merupakan *web server* yang menggunakan basis *single thread* untuk mengeksekusi kode aplikasi. Bahasa umum platform ini adalah *Javascript*, di mana *Javascript* adalah bahasa dasar skrip di sisi *client* untuk halaman web. *Node.js* menggunakan model I/O nonblocking yang digerakkan oleh peristiwa yang membuatnya ringan dan efisien. Ini paling cocok untuk aplikasi intensif CPU secara *real-time* yang berjalan di banyak perangkat. Perusahaan besar dari industri web seperti Yahoo telah berkontribusi pada pengembangan platform, dan yang lain seperti Microsoft telah mengintegrasikan dukungan *Node.js* ke dalam platform cloud mereka. (Chitra & Satapathy, 2017)

Gambar 2.11. Logo *Node.js*

BAB III
METODE PENELITIAN

Penyusunan proposal penelitian ini meliputi perencanaan dan perealisasian alat agar dapat bekerja sesuai dengan rencana dengan mengacu pada rumusan masalah. Dalam pengerjaannya diperlukan langkah-langkah sebagai berikut:

1. Melakukan studi literatur tentang teori deteksi manusia dengan kamera
2. Menentukan spesifikasi alat penelitian
3. Melakukan perancangan sistem alat penelitian
4. Pengujian dan evaluasi alat penelitian

3.1 Spesifikasi

3.1.1 Spesifikasi alat

Spesifikasi alat yang digunakan dalam penelitian ini adalah:

1. 1 buah *Raspberry pi* model 3B sebagai unit prosesor,
2. 1 buah catu daya 5V 2,5A DC untuk mencatu *Raspberry pi*,
3. 5 buah modul *ESP-01* berbasis *chip ESP8266* sebagai unit kontrol pada aktuator,
4. 1 buah USB to TTL *uploader* untuk mengunggah program ke *ESP-01*,
5. 5 buah modul *Relay IoT* yang memiliki *relay* 5V (seri SPD-5VDC-SL-C) serta socket untuk meletakkan unit *ESP-01*,
6. 5 buah catu daya 5V 2A DC untuk mencatu *ESP-01* serta *relay* 5V dalam modul *Relay IoT*,
7. 2 buah lampu *LED* bohlam ruangan 7W dan 12W sebagai aktuator,
8. 1 buah lampu *LED SMD* / bar 14W untuk aquarium sebagai aktuator,
9. 1 buah kipas angin ruangan 50W sebagai aktuator,
10. 1 buah TV / Monitor 12W sebagai aktuator,
11. 5 buah steker, kabel listrik AC, dan stop kontak gantung,
12. PC / laptop dan / atau *smartphone* untuk mengakses *web server* secara *remote* melalui *web browser*.

3.1.2 Spesifikasi lain

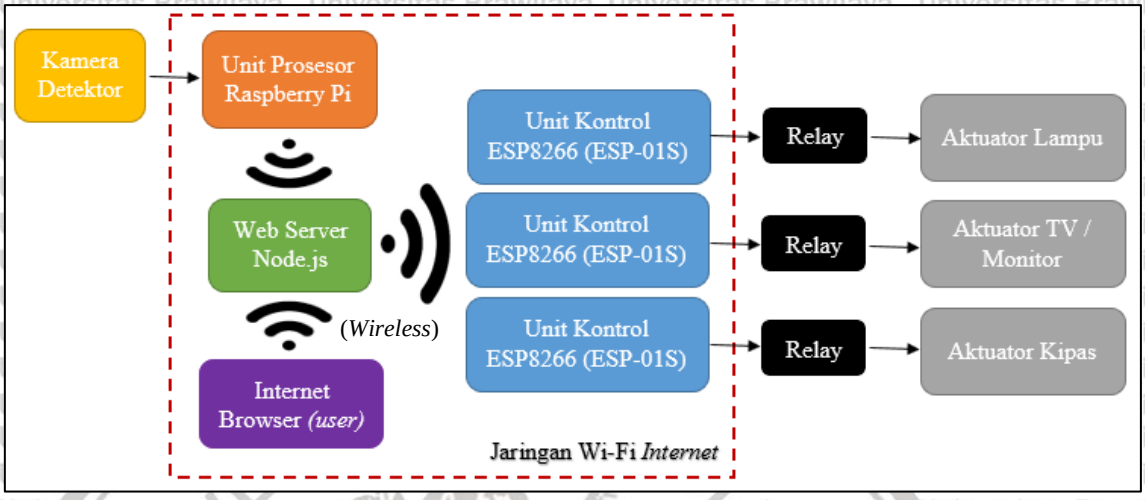
- Jaringan WiFi *internet access point*

Jaringan WiFi *internet* yang digunakan untuk menghubungkan antara unit prosesor *Raspberry pi*, *ESP-01*, dan perangkat *user* adalah jaringan seluler

Telkomsel 4G LTE di daerah Kota Malang, Jawa Timur dengan menggunakan fitur *mobile internet tethering hotspot* dari smartphone yang memiliki *Wireless LAN* berstandar 802.11 a/b/g/n/ac pada frekuensi 2,4GHz.

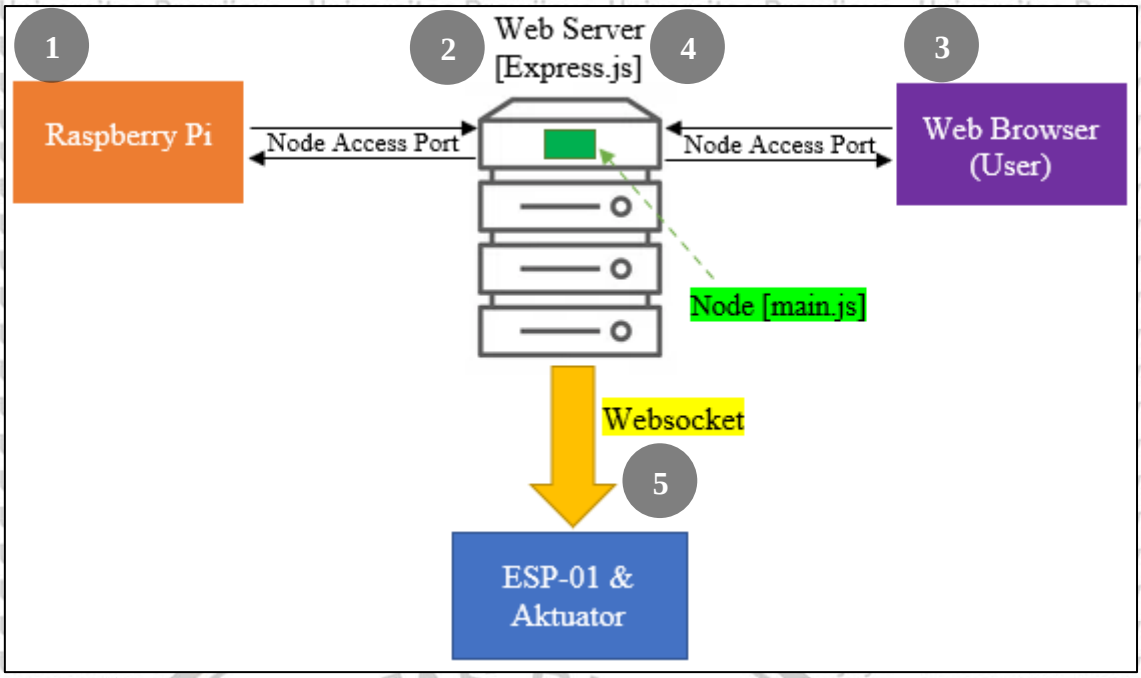
3.2 Perancangan alat

3.2.1 Prinsip kerja alat



Gambar 3.1. Prinsip kerja keseluruhan sistem alat

Agar proses pengiriman data secara *wireless* dapat berjalan, sistem ini membutuhkan satu jaringan Wi-Fi *internet* / *access point di mana Raspberry pi*, seluruh unit *ESP-01*, dan perangkat dari *user* untuk mengakses *internet browser* terhubung di dalam jaringan Wi-Fi *internet* yang sama. Raspberry-pi akan memunculkan sebuah *web server* yang terhubung secara *wireless* dengan perangkat *user* serta seluruh *ESP-01*. Dalam sistem ini yang terhubung secara langsung adalah antara kamera detektor dengan unit prosesor, lalu *ESP-01* dengan *relay* dan aktuator.

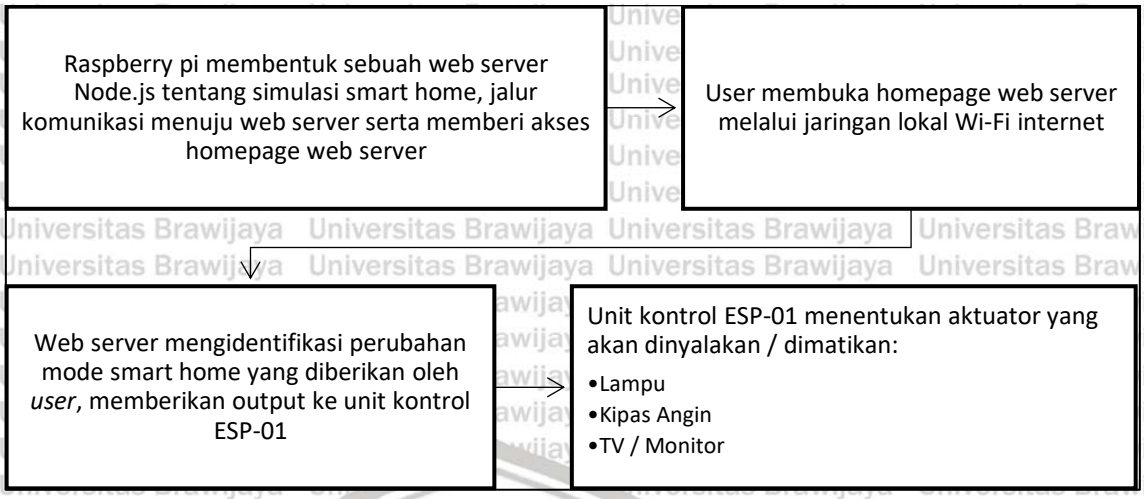


Gambar 3.2. Prinsip kerja *web server*

Web server berfungsi untuk menjalankan simulasi *Smart home*. *Web server* menggunakan *Node.js* sebagai bahasa pemrogaman utama, Express.js sebagai server utama, dan ws.js sebagai server *websocket* yang akan berkomunikasi dengan *ESP-01*. Pada saat ada perubahan event dari homepage simulasi *smart home*, *web server* akan melakukan *broadcast message* melalui *websocket* ke semua *ESP-01*.

ESP-01 berfungsi untuk menerima data dari *Web server* dan menjalankannya ke aktuator. Komunikasi data menggunakan *websocket* sehingga dapat terjadi komunikasi lebih cepat dan realtime. Data yang diterima dari *web server* dalam format json sehingga perlu di parsing menggunakan *library ArduinoJson*. Ketika mendapatkan *broadcast message websocket*, *ESP-01* akan melakukan *parsing* dan menentukan apa yang harus dilakukan oleh aktuator.

3.2.2 Diagram alir perancangan alat



Gambar 3.3. Diagram alir perancangan alat

3.2.3 Objek deteksi perilaku manusia dan tabel logika aktuator

Objek deteksi perilaku manusia ditentukan dari ada / tidaknya orang di dalam ruangan serta kebutuhan akan akses media hiburan oleh pengguna sehingga dirancanglah suatu kondisi mode *web server* untuk simulasi *smart home* yang mampu mempengaruhi kondisi nyala atau tidaknya seluruh aktuator dalam ruangan tersebut. Dengan demikian dibuatlah sebuah tabel logika aktuator yaitu sebuah tabel dengan ketentuan sebagai berikut:

Tabel 3.1.
Logika Aktuator

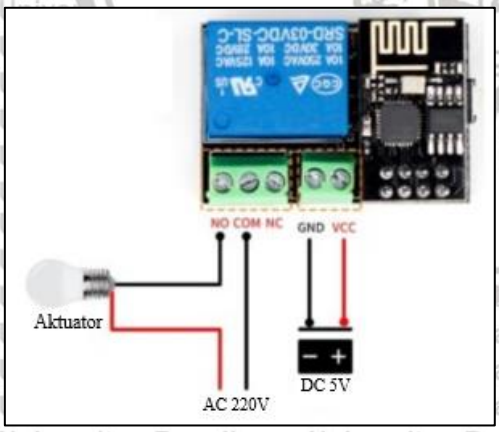
Kondisi dalam ruangan	*Web server*	Lampu Teras Luar	Lampu Ruangan	Kipas Angin	Lampu Aquarium	TV / Monitor
Tidak ada orang	Mode 1	ON	OFF	OFF	OFF	OFF
Ada orang dan akses media	Mode 2	OFF	ON	ON	ON	ON
Ada orang dan tanpa akses media	Mode 3	ON	OFF	OFF	ON	OFF

Pada tabel diatas terdapat 3 kondisi untuk tabel logika pada aktuator yang kemudian digunakan sebagai acuan untuk mengaktifkan salah satu mode *Smart home* dan menginstruksikan unit kontrol di dalam ruangan untuk menghidupkan / mematikan aktuator tertentu. Pada tabel diatas ini menunjukkan hubungan antara mode *smart home* dengan kondisi-kondisi berikut:

- Mode 1 akan dijalankan otomatis jika di dalam rumah tidak terdeteksi ada orang yaitu lampu teras luar akan dinyalakan sedangkan lampu dalam ruangan, lampu aquarium, kipas angin dan TV / Monitor akan dimatikan.
- Mode 2 akan dijalankan otomatis jika di dalam ruangan terdeteksi ada orang dan mengakses media hiburan yaitu lampu teras luar akan dimatikan sedangkan lampu dalam ruangan, lampu aquarium, kipas angin, serta TV/ Monitor akan dinyalakan,
- Mode 3 akan dijalankan otomatis jika di dalam rumah terdeteksi ada orang namun tidak mengakses media hiburan yaitu lampu teras luar dan lampu aquarium akan dinyalakan sedangkan lampu dalam ruangan, kipas angin, dan TV / monitor akan dimatikan,

3.2.4 Pengkabelan alat

Dalam perancangan keseluruhan alat terdapat 5 modul *Relay IoT* yang dipasang terpisah antara satu sama lain. Di tiap modulnya dipasangkan sebuah unit *ESP-01* ke dalam *socket* yang tersedia di dalam modul *Relay IoT*. Sebuah catu daya 5V DC dihubungkan ke port VCC dan GND pada setiap modul *Relay IoT* untuk mencatu *relay* 5V serta *ESP-01*. Sebuah catu daya 220V AC dihubungkan ke port *COM* dan *NO (Normally Open)* pada setiap modul *Relay IoT*, *di mana* diantara jalur pada port *NO* dengan catu daya 220V AC dipasangkan sebuah aktuator. Aktuator dapat berupa lampu *LED* bohlam, lampu *LED SMD* / bar, kipas angin, maupun TV / monitor.



Gambar 3.4. Pengkabelan modul *Relay IoT* dengan aktuator

BAB IV
HASIL DAN PEMBAHASAN

4.1 Pengujian *Loading Time* saat mengaktifkan *Web server*

Untuk mengaktifkan *web server* dari dalam *Raspberry Pi* pasti membutuhkan waktu tertentu sehingga diperlukan suatu pengujian mengenai *loading time* yang dibutuhkan dari kondisi *web server* sebelum aktif hingga aktif. Bersamaan dengan hal tersebut diperlukan juga sebuah pengujian tentang posisi jarak antara *Raspberry Pi* dengan titik *internet access point* sehingga dapat ditemukan apakah ada pengaruh antara jarak unit prosesor dengan *internet access point* dan waktu yang dibutuhkan untuk mengaktifkan *web server* dari *Raspberry Pi.*

4.1.1 Tujuan

Pengujian ini ditujukan untuk mengetahui apakah ada perbedaan waktu untuk mengaktifkan *web server* berdasarkan jarak antara *Raspberry pi* dengan titik *access point* Wi-Fi *internet*.

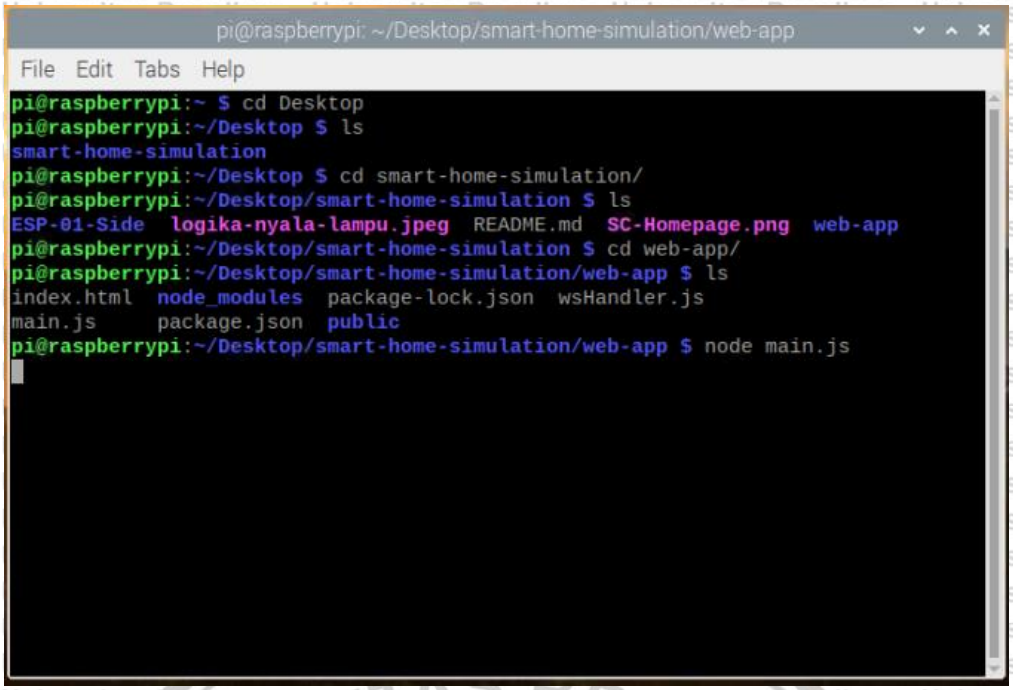
4.1.2 Alat yang digunakan

Alat yang digunakan dalam pengujian ini adalah:

1. Laptop
2. *Raspberry pi* 3B
3. Stopwatch

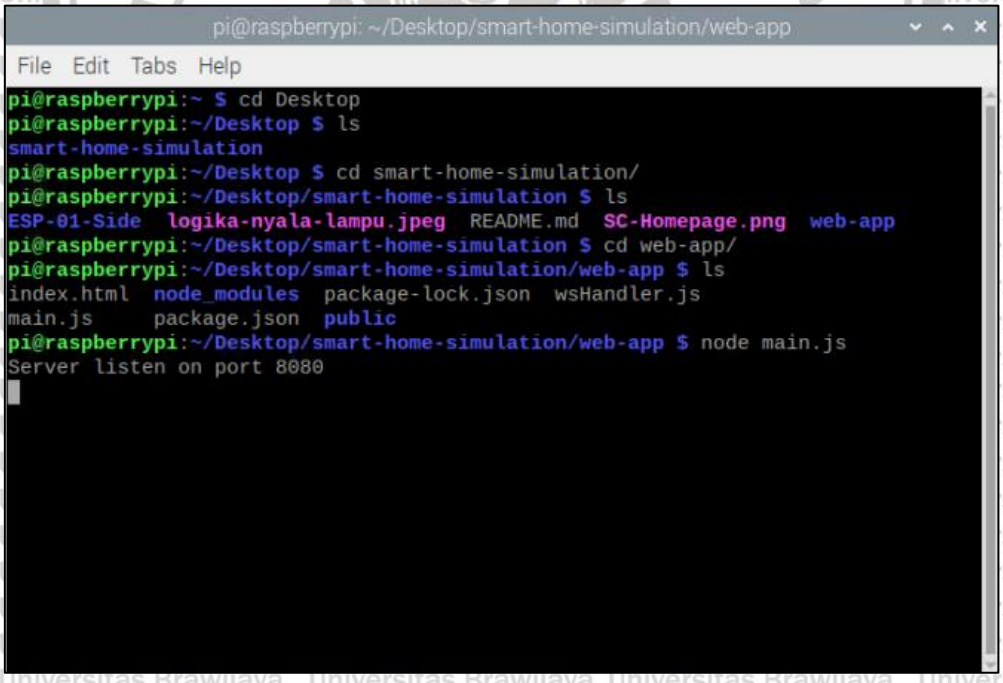
4.1.3 Prosedur pengujian

Untuk mengaktifkan *web server, Raspberry pi* perlu menjalankan sebuah file web app bernama *[main.js]* yang sudah terinstall di dalamnya. Saat *Raspberry pi* mengeksekusi progam *[main.js]* terdapat loading time sebelum server aktif. Berikut ini adalah kondisi sesaat setelah menjalankan instruksi *[node main.js]* pada *command prompt Raspberry pi*:



Gambar 4.1. Command prompt pada *Raspberry pi* sebelum *web server* aktif

Mengaktifkan *web server* memerlukan beberapa detik setelah menjalankan instruksi diatas. Gamber berikut ini menunjukkan bahwa *web server* sudah aktif pada jaringan *internet* lokal dengan *port 8080*:



Gambar 4.2. Command prompt pada *Raspberry pi* setelah *web server* aktif

Prosedur mengukur parameter *loading time* dari pengujian ini adalah menghitung waktu yang dibutuhkan sejak instruksi untuk membuka sebuah node dijalankan hingga

munculnya server aktif pada port 8080 menggunakan stopwatch, lalu data tersebut dicatat dan dimasukkan dalam satu tabel kemudian dianalisis.

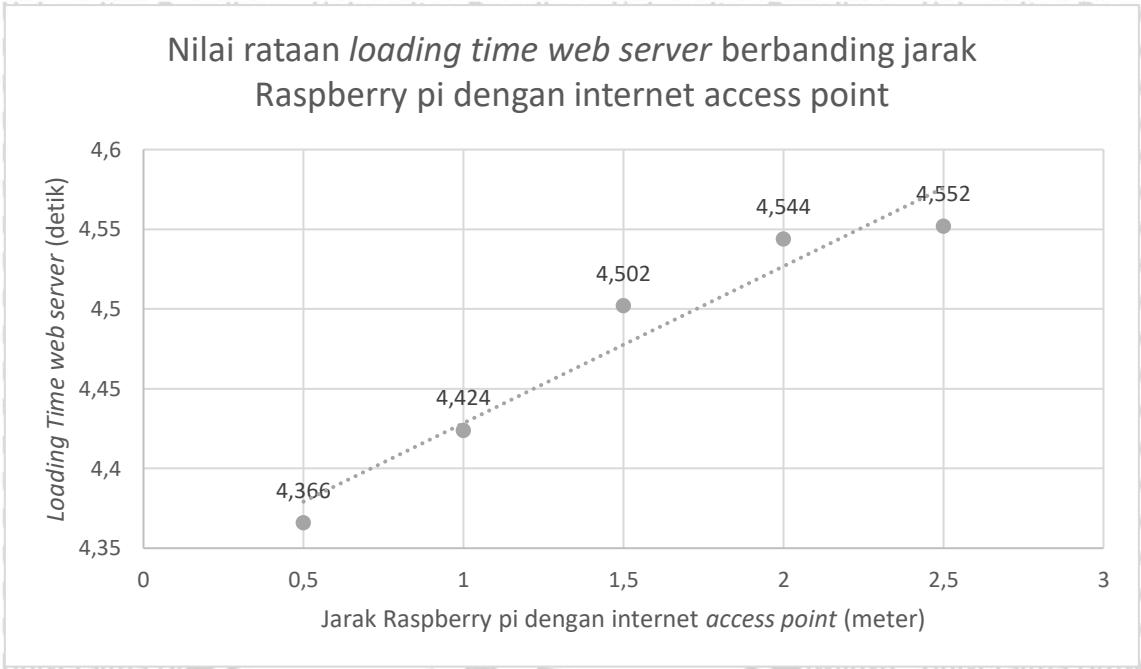
4.1.4 Hasil uji dan analisis

Setelah melakukan beberapa kali pengujian hasil pengukuran waktu *loading time* dengan jarak antara *Raspberry pi* dan *access point* dicatat dalam tabel berikut ini:

Tabel 4.1.
Hasil pengujian 1: Loading Time web server berbanding jarak Access Point

No.	Jarak antara Raspberry-Pi dengan *internet Access Point*	*Loading Time* pada *web server* untuk aktif (detik)					
		Uji ke-1	Uji ke-2	Uji ke-3	Uji ke-4	Uji ke-5	Rataan
1	0,5 meter	4,41	4,39	4,22	4,52	4,29	4,366
2	1 meter	4,64	4,32	4,44	4,21	4,51	4,424
3	1,5 meter	4,57	4,23	4,76	4,45	4,50	4,502
4	2 meter	4,67	4,62	4,34	4,60	4,49	4,544
5	2,5 meter	4,73	4,50	4,36	4,72	4,45	4,552

Jarak 0,5 - 2,5 meter diambil sebagai parameter dengan asumsi bahwa saat di dalam ruangan jarak antara unit prosesor dengan *internet access point* tidak melebihi rentang tersebut. Dari tabel hasil pengujian diatas terlihat bahwa *loading time* yang dibutuhkan untuk mengaktifkan *web server* pada *Raspberry pi* dalam rentang jarak antara 0,5 - 2,5 meter dengan *internet access point* relatif mendekati yaitu sekitar 4,2 - 4,7 detik. Berdasarkan tabel hasil pengujian diatas dapat dibuat sebuah grafik sebagai berikut:



Gambar 4.3. Grafik hasil pengujian 1: Nilai rataan *loading time web server*

Dari grafik hasil pengujian diatas dapat diketahui bahwa nilai rataan *loading time* pada *web server* untuk aktif ada pada rentang 4,35 - 4,6 detik. Dari hasil tersebut dapat diambil kesimpulan bahwa semakin jauh jarak antara *Raspberry pi* dengan *internet access point* maka *loading time* yang dibutuhkan untuk mengaktifkan *web server* semakin tinggi atau semakin lama.

4.2 Pengujian *Response Time* dari *web server* menuju unit kontrol

Setelah *web server* aktif kemudian pengguna dapat melakukan sebuah perubahan mode *smart home* yang diinginkan. Dalam proses ini ada sekumpulan proses yang dilakukan oleh *Raspberry Pi* sebagai *host* dari *web server* sebelum memberikan perintah kepada unit kontrol *ESP-01* untuk mengeksekusi program. Sekumpulan proses ini pasti membutuhkan waktu tertentu agar bisa tereksekusi dari awal hingga akhir. Dengan demikian maka perlu adanya pengujian tentang *response time* yang dibutuhkan setelah sebuah input diberikan ke *web server* hingga program dapat dieksekusi pada unit kontrol *ESP-01* untuk mengaktifkan / mematikan *relay* menuju aktuator.

4.2.1 Tujuan

Pengujian ini dilakukan untuk mengetahui bagaimana *response time* yang dibutuhkan untuk mengirim perintah dari *web server* menuju unit kontrol serta menyalakan aktuator yang diinginkan setelah *web server* aktif mendapatkan *idle time* tertentu.

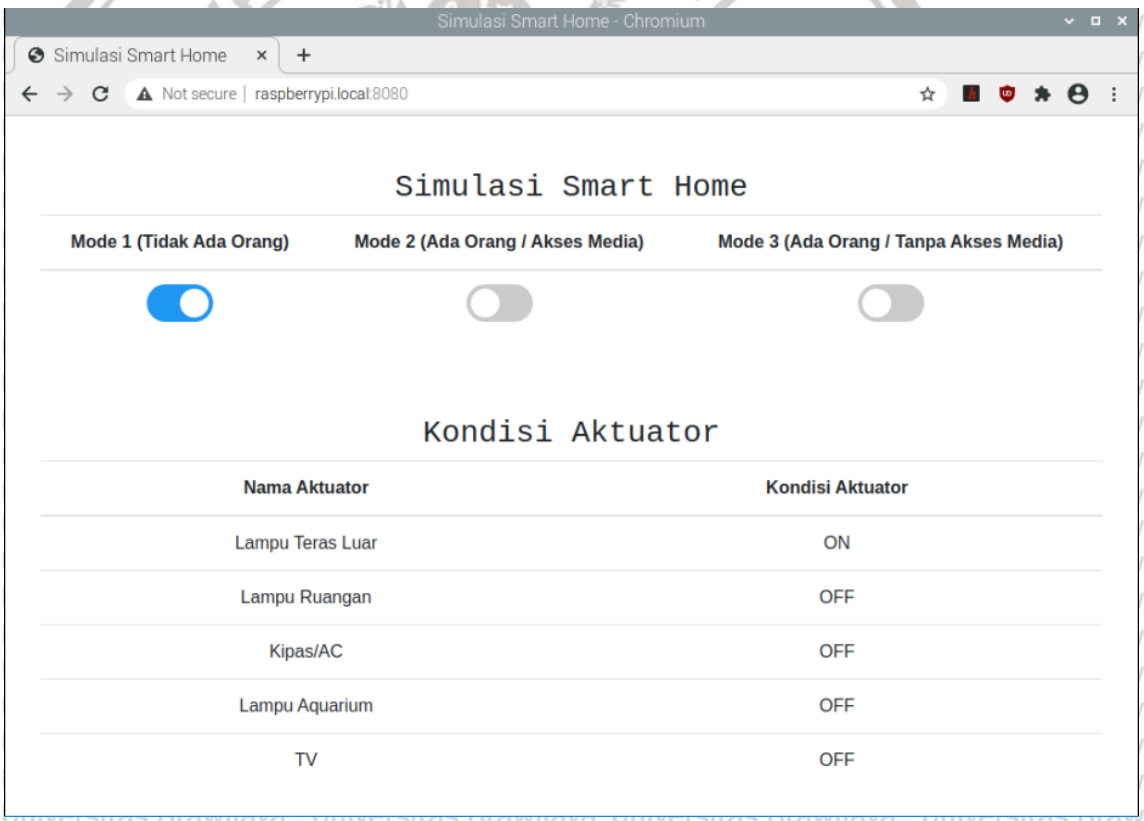
4.2.2 Alat yang digunakan

Alat yang digunakan dalam pengujian ini adalah:

1. Laptop / *smartphone*
2. *Raspberry pi* 3B
3. Stopwatch
4. Modul *Relay IoT* yang terhubung dengan unit *ESP-01*

4.2.3 Prosedur pengujian

User mengakses sebuah *web browser* dan membuka *homepage web server* simulasi *smart home*, lalu setelah server aktif menunggu selama beberapa waktu sebelum *user* melakukan input penggantian mode *smart home* meunuju *ESP-01*, kemudian setelah *ESP-01* menerima instruksi untuk berganti mode dan mengganti kondisi *relay* maka *response time* akan diukur menggunakan stopwatch setiap lalu dicatat dan dimasukkan ke dalam tabel hasil pengujian. Urutan penggantian mode adalah dari Mode 1 - *idle* - Mode 2 - *idle* - Mode 1 - *idle* - Mode 3 - *idle* - Mode 2 - *idle* - Mode 3 - *idle* - Mode 1.



Gambar 4.4. Tampilan *homepage web server* untuk simulasi *smart home*

4.2.4 Hasil uji dan analisis

Setelah melakukan pengujian dan pengambilan data tentang *response time* dari *web server* menuju *ESP-01*, berikut ini adalah tabel hasil pengujian yang didapatkan:

Tabel 4.2.
Hasil pengujian 2: *Response Time web server* menuju *ESP-01* setelah adanya *Idle Time*

No	*Idle Time* setelah mode aktif	*Response Time* perubahan mode dari *web server* menuju *ESP-01* (detik)					
		Mode 1 ke 2	Mode 1 ke 3	Mode 2 ke 1	Mode 2 ke 3	Mode 3 ke 1	Mode 3 ke 2
1	0,5 menit	0	10,49	10,54	10,51	0	0
2	1 menit	10,77	0	0	0	10,54	10,15
3	1,5 menit	10,82	9,97	10,33	10,28	11,27	10,12
4	2 menit	10,67	10,07	9,52	10,31	9,40	10,37
5	2,5 menit	10,34	10,25	10,61	10,42	10,27	10,08

Dari tabel hasil pengujian diatas terdapat kolom yang menunjukkan angka 0 (nol) menandakan bahwa saat melakukan input penggantian mode oleh *user* dalam beberapa kondisi mode tersebut, *ESP-01* dapat langsung menerima instruksi dan segera mengganti kondisi *relay* untuk aktif / mati tanpa ada *delay* / *latency* yang berarti (sangat cepat / *latency* sangat kecil) sedangkan untuk kondisi lainnya memerlukan waktu untuk menerima instruksi dan mengganti kondisi keaktifan pada masing-masing *relay*.

Dari tabel hasil pengujian terlihat bahwa *response time* yang dibutuhkan antara mengganti mode pada *web browser* dengan penggantian kondisi *relay* di sisi *ESP-01* ada pada rentang 9 - 11 detik. Dari tabel hasil pengujian diatas juga terlihat bahwa keadaan yang tidak konsisten memiliki *response time* ada pada pergantian mode setelah *idle time* 0,5 - 1 menit, sedangkan pada pergantian mode setelah *idle time* 1,5 - 2,5 menit seluruh *ESP-01* selalu ada *response time* agar bisa menerima instruksi dan mengganti kondisi *relay* untuk aktif / mati. Hal ini menunjukkan bahwa *web server* pada *Raspberry pi* / *ESP-01* akan mengalami *idle mode* setelah batas rentang waktu tertentu sehingga komunikasi untuk pengiriman data dari *web server* menuju *ESP-01* akan memunculkan sebuah *response time* dan membuat paket data tidak langsung bisa dikirim dan dieksekusi.

4.3 Pengujian efisiensi energi listrik pada aktuator

Untuk menghitung nilai efisiensi energi listrik yang digunakan oleh aktuator dalam rumah pada suatu sistem *smart home* dibutuhkan sebuah pengujian untuk mengukur nilai energi listrik keluaran dari tiap aktuator yang digunakan agar bisa dibandingkan dengan nilai energi listrik keluaran aktuator tanpa menggunakan sistem *smart home* sehingga persentase nilai efisiensi energi listrik sistem dapat ditemukan.

4.3.1 Tujuan

Pengujian ini dilakukan untuk mengetahui besar nilai efisiensi energi listrik yang dihasilkan dari total daya yang dibutuhkan dari komparasi saat sebelum menggunakan sistem *smart home* dengan setelah menggunakan sistem *smart home*.

4.3.2 Alat yang dibutuhkan

Alat yang digunakan dalam pengujian ini adalah:

1. Laptop
2. *Raspberry pi* 3B
3. 5 Modul *Relay IoT* yang terhubung dengan unit *ESP-01*
4. Lampu *LED Ossio* 7W
5. Lampu *LED In-lite* 12W
6. Lampu *LED* bar aquarium 14W
7. Kipas angin berdiri 50W
8. TV / Monitor 12W
9. AVOmeter AC

4.3.3 Prosedur pengujian

Pada pengujian aktuator yang akan diteliti adalah sebagai berikut:

- Tegangan keluaran AC pada masing-masing aktuator lampu *LED*, kipas angin, dan TV / monitor,
- Arus keluaran AC pada masing-masing aktuator lampu *LED*, kipas angin, dan TV / monitor,
- Daya keluaran AC pada masing-masing aktuator lampu *LED*, kipas angin, dan TV / monitor,
- Efisiensi energi listrik di setiap mode *smart home* yang digunakan.

Nilai tegangan dan arus diukur menggunakan AVOmeter AC setiap 1 menit selama 10 menit. Karena nilai yang terukur dari AVOmeter adalah nilai *V.rms* dan *i.rms*, maka perlu dikonversikan menjadi nilai *V.max* dan *i.max* menggunakan rumus-rumus berikut ini yang kemudian hasil konversinya dimasukkan ke dalam tabel hasil pengujian:

$$V.max = \sqrt{2} \times V.rms \quad \text{(4-1)}$$

$$i.max = \sqrt{2} \times i.rms \quad \text{(4-2)}$$

dengan:

V.max = Tegangan maksimum (Volt)

V.rms = Tegangan *root means square* (Volt)

i.max = Arus maksimum (Ampere)

i.rms = Arus *root means square* (Ampere)

Untuk nilai daya didapatkan dari perkalian antara tegangan dan arus. Rumus perhitungan dayanya adalah:

$$P = V.max \times i.max \quad \text{(4-3)}$$

dengan:

P = Daya nyata (Watt)

Setelah perhitungan daya didapatkan kemudian dimasukkan ke dalam tabel hasil pengujian.

4.3.4 Hasil uji dan analisis

Berikut ini merupakan tabel hasil pengukuran nilai tegangan, arus, dan daya pada aktuator:

Tabel 4.3.
Hasil pengujian 3: Nilai pengukuran listrik pada lampu

Waktu Uji (menit ke-)	Lampu Teras			Lampu Ruangan			Lampu Aquarium		
	Tegangan (V)	Arus (A)	Daya (W)	Tegangan (V)	Arus (A)	Daya (W)	Tegangan (V)	Arus (A)	Daya (W)
1	211	0,2	42,2	211	0,2	42,2	211	0,21	44,31
2	210	0,18	37,8	211	0,21	44,31	211	0,2	42,2
3	211	0,18	37,98	210	0,2	42	211	0,21	44,31
4	211	0,2	42,2	211	0,2	42,2	211	0,2	42,2
5	210	0,2	42	210	0,2	42	210	0,2	42
6	211	0,2	42,2	210	0,21	44,1	210	0,2	42
7	211	0,18	37,98	211	0,21	44,31	211	0,2	42,2
8	210	0,2	42	211	0,2	42,2	211	0,2	42,2
9	210	0,2	42	211	0,2	42,2	211	0,2	42,2
10	210	0,2	42	211	0,2	42,2	210	0,2	42
Rataan	210,5	0,194	40,836	210,7	0,203	42,772	210,7	0,202	42,562

Tabel 4.4.
Hasil pengujian 3: Nilai Pengukuran listrik pada kipas angin dan TV / monitor

Waktu Uji (menit ke-)	Kipas Angin			TV / Monitor		
	Tegangan (V)	Arus (A)	Daya (W)	Tegangan (V)	Arus (A)	Daya (W)
1	211	0,24	50,64	210	0,18	37,8
2	211	0,24	50,64	211	0,2	42,2
3	210	0,23	48,3	210	0,2	42
4	211	0,24	50,64	211	0,2	42,2
5	210	0,24	50,4	211	0,2	42,2
6	211	0,24	50,64	210	0,2	42
7	211	0,24	50,64	211	0,2	42,2
8	210	0,24	50,4	211	0,2	42,2
9	211	0,24	50,64	211	0,18	37,98
10	211	0,23	48,53	211	0,2	42,2
Rataan	210,7	0,238	50,147	210,7	0,196	41,298

Setelah mendapatkan tabel hasil pengukuran diatas dapat menghitung nilai daya yang digunakan dalam tiap mode *smart home* sesuai tabel logika aktuator yang dipaparkan sebelumnya. Menggunakan acuan nilai daya rataan dari tiap aktuator dari tabel hasil pengukuran maka perhitungan energi listrik yang digunakan pada setiap mode *smart home* selama 60 menit adalah sebagai berikut:

- Mode 1:
  (Daya Rataan 1 aktuator) x 60 menit = 40,836 x 60 = 2.450,16 Wh = 2,450 kWh
- Mode 2:
  (Daya Rataan 4 aktuator) x 60 menit = 176,779 x 60 = 10.606,74 Wh = 10,607 kWh
- Mode 3:
  (Daya Rataan 2 aktuator) x 60 menit = 83,398 x 60 = 5.003,88 Wh = 5,004 kWh

Jika tidak menggunakan mode *smart home* dengan asumsi seluruh aktuator dinyalakan bersamaan selama 60 menit, maka nilai energi listrik yang dibutuhkan adalah:

- Tanpa mode *smart home*:
  (Daya Rataan 5 aktuator) x 60 menit = 217,615 x 60 menit = 13.056,90 Wh = 13,057 kWh

Hasil perhitungan energi ini kemudian dapat digunakan untuk menghitung efisiensi penggunaan energi dengan ditentukan sebuah rumus efisiensi di mana pembagian nilai selisih dari total energi menggunakan sistem *smart home* dengan total energi tanpa menggunakan sistem *smart home* dikali dengan 100%, sebagaimana dituliskan dalam rumus berikut:

$$\eta = \frac{selisih\ total\ energi\ menggunakan\ \&\ tanpa\ sistem\ smart\ home}{total\ energi\ tanpa\ sistem\ smart\ home} \times 100\% \text{.....(4-4)}$$

dengan:

η = Koefisien efisiensi energi listrik

Dengan demikian maka efisiensi yang bisa didapatkan dari tiap mode *smart home* adalah:

- Mode 1: $\eta = \frac{(13{,}057 - 2{,}450)\ \text{kWh}}{13{,}057\ \text{kWh}} = \frac{10{,}607\ \text{kWh}}{13{,}057\ \text{kWh}} \times 100\% = 81{,}236\%$
- Mode 2: $\eta = \frac{(13{,}057 - 10{,}607)\ \text{kWh}}{13{,}057\ \text{kWh}} = \frac{2{,}450\ \text{kWh}}{13{,}057\ \text{kWh}} \times 100\% = 18{,}763\%$
- Mode 3: $\eta = \frac{(13{,}057 - 5{,}004)\ \text{kWh}}{13{,}057\ \text{kWh}} = \frac{8{,}053\ \text{kWh}}{13{,}057\ \text{kWh}} \times 100\% = 61{,}675\%$

Dari hasil analisis tersebut terlihat bahwa besarnya efisiensi listrik setiap mode berbeda-beda bergantung pada mode apa yang di aktifkan serta jenis aktuator mana saja yang akan dinyalakan / dimatikan, tetapi jelas terlihat juga bahwa dengan menggunakan sistem *smart home* ini akan menambah efisiensi penggunaan energi listrik dan menghemat penggunaan energi listrik rumahan setiap jamnya.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian serta analisis data yang sudah dilakukan dapat diambil kesimpulan bahwa:

1. Sistem penggerak aktuator pengendali daya listrik dalam ruangan berbasis aktivitas manusia sebagai pendekatan *smart home* yang mampu meningkatkan efisiensi energi listrik dalam rumah dapat didesain dengan menggunakan *Raspberry Pi, ESP8266* serta *web server Node.js.*
2. Desain sistem penggerak aktuator pengendali daya listrik dalam ruangan berbasis aktivitas manusia dapat disimulasikan dan diaplikasikan untuk meningkatkan efisiensi energi listrik rumahan sebagai sistem *smart home* dengan karakteristik *loading time web server* untuk aktif dari *Raspberry Pi* berada pada rentang waktu antara 4,35 - 4,6 detik, *response time* web server menuju unit kontrol untuk mengaktifkan / mematikan aktuator paling lambat pada rentang 9-10 detik, serta sistem ini mampu meningkatkan efisiensi penggunaan energi listrik hingga 81% per jam.

5.2 Saran

Penggantian mode simulasi *smart home* belum berhasil untuk dipicu secara otomatis melalui input yang didapatkan dari kamera detektor sehingga mode *smart home* dalam sistem ini masih butuh untuk diaktifkan secara manual dari pengguna. Diharapkan untuk penelitian lebih lanjut dapat menggunakan kamera sebagai input pemicu otomatis untuk perubahan mode *smart home* menurut ada / tidaknya aktivitas perilaku manusia yang terdeteksi oleh kamera.

Penelitian ini masih memiliki peluang untuk dikembangkan lebih lanjut terutama pada variasi objek deteksi yang dapat dijadikan acuan untuk variasi mode *smart home* yang digunakan atau variasi aksi yang dilakukan oleh unit kontrol kepada aktuator yang tidak terbatas pada *relay* sebagai saklar ON / OFF saja.

DAFTAR PUSTAKA

Chitra, L. P., & Satapathy, R. (2017). Performance comparison and evaluation of Node.js and traditional web server (IIS). *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies, ICAMMAET 2017*, *2017-Janua*, 1–4.

Espressif Systems. (2013). Data Sheet Espressif Smart Connectivity Platform: Esp8266. *WiFi Allience*, 23. https://cdn-shop.adafruit.com/datasheets/ESP8266_Specifications_English.pdf

Ke, S. R., Thuc, H. L. U., Lee, Y. J., Hwang, J. N., Yoo, J. H., & Choi, K. H. (2013). A review on video-based human activity recognition. In *Computers* (Vol. 2, Issue 2).

Marikyan, D., Papagiannidis, S., & Alamanos, E. (2019). A systematic review of the smart home literature: A user perspective. *Technological Forecasting and Social Change*, *138*(September 2018), 139–154.

Ortiz, G., & Xia, W. (2013). *(12) United States Patent*. *2*(12).

Raspberry, P. F. (2012). Raspberry Pi 3 Model B - Raspberry Pi. *Datasheet*, 4–13. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/?resellerType=home%0Ahttps://www.raspberrypi.org/products/raspberry-pi-3-model-b/

Schiefer, M. (2015). Smart Home Definition and Security Threats. *Proceedings - 9th International Conference on IT Security Incident Management and IT Forensics, IMF 2015*, 114–118.

Singh, S. C. (2009). Basics of light emitting diodes, characterizations and applications. *Handbook of Light Emitting and Schottky Diode Research*, *December 2009*, 133–168.

Wang, S., Gao, J. Z., Lin, H., Shitole, M., Reza, L., & Zhou, S. (2019). Dynamic human behavior pattern detection and classification. *Proceedings - 5th IEEE International Conference on Big Data Service and Applications, BigDataService 2019, Workshop on Big Data in Water Resources, Environment, and Hydraulic Engineering and Workshop on Medical, Healthcare, Using Big Data Technologies*, 159–166.

LAMPIRAN

- ***Datasheet*** **Raspberry Pi 3B**

Raspberry Pi

Raspberry Pi 3 Model B

Product Name	**Raspberry Pi 3**
Product Description	The Raspberry Pi 3 Model B is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B. Whilst maintaining the popular board format the Raspberry Pi 3 Model B brings you a more powerful processor, 10x faster than the first generation Raspberry Pi. Additionally it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs.
RS Part Number	**896-8660**

RS

www.rs-components.com/raspberrypi

Raspberry Pi

Raspberry Pi 3 Model B

Specifications

Processor	Broadcom BCM2387 chipset. 1.2GHz Quad-Core ARM Cortex-A53 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE)
GPU	Dual Core VideoCore IV® Multimedia Co-Processor. Provides Open GL ES 2.0, hardware-accelerated OpenVG, and 1080p30 H.264 high-profile decode. Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure
Memory	1GB LPDDR2
Operating System	Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IoT
Dimensions	85 x 56 x 17mm
Power	Micro USB socket 5V1, 2.5A

Connectors:

Ethernet	10/100 BaseT Ethernet socket
Video Output	HDMI (rev 1.3 & 1.4 Composite RCA (PAL and NTSC)
Audio Output	Audio Output 3.5mm jack, HDMI USB 4 x USB 2.0 Connector
GPIO Connector	40-pin 2.54 mm (100 mil) expansion header: 2x20 strip Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2)
Display Connector	Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane
Memory Card Slot	Push/pull Micro SDIO

Key Benefits

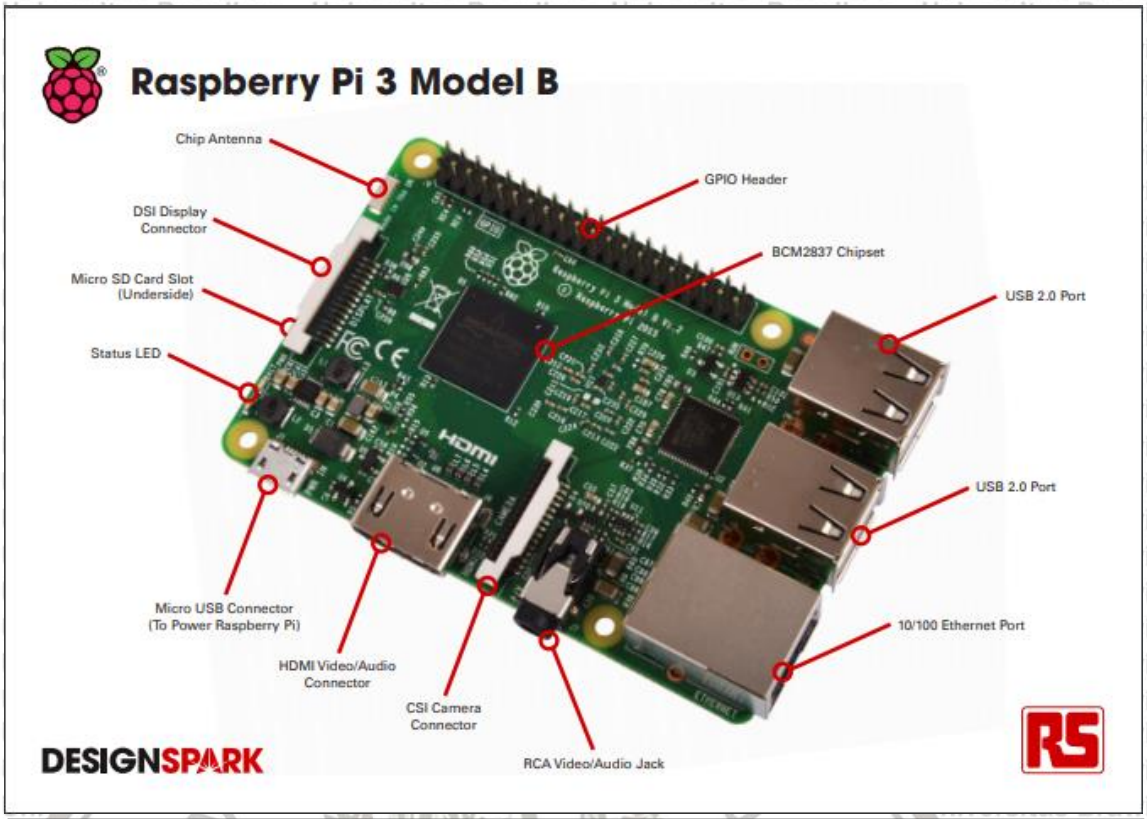
- Low cost
- 10x faster processing
- Consistent board format
- Added connectivity

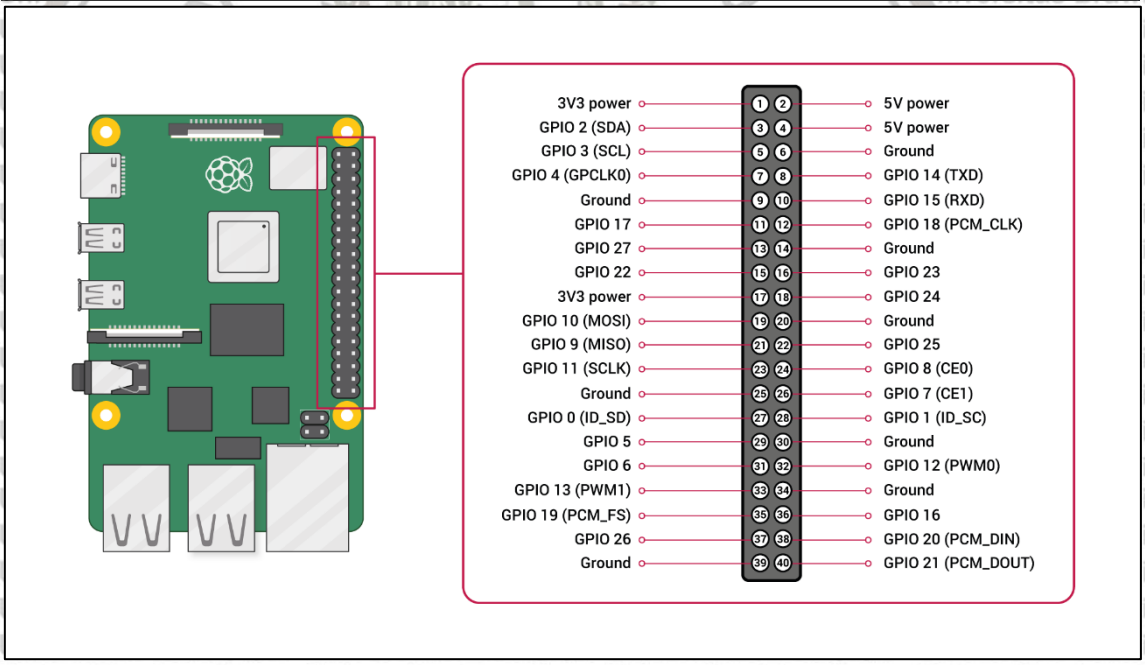
Key Applications

- Low cost PC/tablet/laptop
- Media centre
- Industrial/Home automation
- Print server
- Web camera
- Wireless access point
- Environmental sensing/monitoring (e.g. weather station)
- IoT applications
- Robotics
- Server/cloud server
- Security monitoring
- Gaming

RS

www.rs-components.com/raspberrypi





- ***Webpage html script code***

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link rel="stylesheet" href="static/bootstrap-4.1.3-dist/css/bootstrap.min.css">
  <link rel="stylesheet" href="static/toggle.css">
  <script src="static/bootstrap-4.1.3-dist/js/bootstrap.min.js"></script>
  <script src="static/jquery-3.6.0.min.js"></script>

  <title>Simulasi Smart Home</title>
</head>
<body>
  <div class="container mt-5">
    <div class="row d-flex justify-content-center">
      <h3 class="h3 text-monospace">Simulasi Smart Home</h3>
    </div>
    <div class="row d-flex-justify-content-center">
      <table class="table">
        <thead>
          <tr>
            <th class="text-center" scope="col">Mode 1 (Tidak Ada Orang)</th>
            <th class="text-center" scope="col">Mode 2 (Ada Orang / Akses Media)</th>
            <th class="text-center" scope="col">Mode 3 (Ada Orang / Tanpa Akses
Media)</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td class="text-center">
              <label class="switch">
                <input type="checkbox" id="mode_1">
                <span class="slider round"></span>
              </label>
            </td>
            <td class="text-center">
              <label class="switch">
                <input type="checkbox" id="mode_2">
                <span class="slider round"></span>
              </label>
            </td>
            <td class="text-center">
              <label class="switch">
                <input type="checkbox" id="mode_3">
                <span class="slider round"></span>
              </label>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
```

```
    <div class="row d-flex justify-content-center mt-5">
      <h3 class="h3 text-monospace">Kondisi Aktuator</h3>
    </div>
    <div class="row d-flex-justify-content-center">
      <table class="table">
        <thead>
          <tr>
            <th class="text-center" scope="col">Nama Aktuator</th>
            <th class="text-center" scope="col">Kondisi Aktuator</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td class="text-center" >Lampu Teras Luar</td>
            <td class="text-center" id="lampu_teras">ON</td>
          </tr>
          <tr>
            <td class="text-center" >Lampu Ruangan</td>
            <td class="text-center" id="lampu_ruangan">OFF</td>
          </tr>
          <tr>
            <td class="text-center" >Kipas/AC</td>
            <td class="text-center" id="kipas_ac">OFF</td>
          </tr>
          <tr>
            <td class="text-center" >Lampu Aquarium</td>
            <td class="text-center" id="lampu_aquarium">OFF</td>
          </tr>
          <tr>
            <td class="text-center" >TV</td>
            <td class="text-center" id="tv">OFF</td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>

</body>

<script>
  $(document).ready(() => {
    var mode1checked = true;
    var mode2checked = false;
    var mode3checked = false;

    $("#mode_1").prop('checked', true);

    $("#mode_1").change(()=>{
      if(mode1checked){
        $("#mode_1").prop('checked', true);
      }
      else{
        mode1checked = true;
        mode2checked = false;
        mode3checked = false;
```

```
        $("#mode_2").prop('checked', false);
        $("#mode_3").prop('checked', false);

        $.ajax({
          type : 'POST',
          url : window.location.origin + '/action',
          data : JSON.stringify({
            mode : 'mode_1'
          }),
          dataType : 'json',
          contentType: 'application/json',
          success : (response) => {
            console.log(response)
          }
        })

        $("#lampu_teras").text("ON");
        $("#lampu_ruangan").text("OFF");
        $("#kipas_ac").text("OFF");
        $("#lampu_aquarium").text("OFF");
        $("#tv").text("OFF");
    }
});

$("#mode_2").change(()=>{
    if(mode2checked){
        $("#mode_2").prop('checked', true);
    }
    else{
        mode1checked = false;
        mode2checked = true;
        mode3checked = false;

        $("#mode_1").prop('checked', false);
        $("#mode_3").prop('checked', false);

        $.ajax({
          type : 'POST',
          url : window.location.origin + '/action',
          data : JSON.stringify({
            mode : 'mode_2'
          }),
          dataType : 'json',
          contentType: 'application/json',
          success : (response) => {
            console.log(response)
          }
        })

        $("#lampu_teras").text("OFF");
        $("#lampu_ruangan").text("ON");
        $("#kipas_ac").text("ON");
        $("#lampu_aquarium").text("ON");
        $("#tv").text("ON");
```

```
    }
  });
  $("#mode_3").change(()=>{
    if(mode3checked){
      $("#mode_3").prop('checked', true);
    }
    else{
      mode1checked = false;
      mode2checked = false;
      mode3checked = true;

      $("#mode_1").prop('checked', false);
      $("#mode_2").prop('checked', false);

      $.ajax({
        type : 'POST',
        url : window.location.origin + '/action',
        data : JSON.stringify({
          mode : 'mode_3'
        }),
        dataType : 'json',
        contentType: 'application/json',
        success : (response) => {
          console.log(response)
        }
      })

      $("#lampu_teras").text("ON");
      $("#lampu_ruangan").text("OFF");
      $("#kipas_ac").text("OFF");
      $("#lampu_aquarium").text("ON");
      $("#tv").text("OFF");
    }
  });
 })
</script>
</html>
```

- **Paket instruksi dalam *web server* [main.js]**

```
const express = require('express');
var app = express();
const bodyParser = require('body-parser')
const Websocket = require('ws');

const wss = new Websocket.Server({ port: 3001 });
var wssClient = []

const SMART HOME_MODE = {
  mode_1 : {
    lampu_teras : 'OFF',
    lampu_ruangan : 'ON',
    tirai_jendela : 'TERBUKA',
    kipas_ac : 'ON',
    lampu_aquarium : 'ON',
    tv : 'OFF'
  },
  mode_2 : {
    lampu_teras : 'OFF',
    lampu_ruangan : 'ON_REDUP',
    tirai_jendela : 'TERTUTUP',
    kipas_ac : 'ON',
    lampu_aquarium : 'ON',
    tv : 'ON'
  },
  mode_3 : {
    lampu_teras : 'ON',
    lampu_ruangan : 'OFF',
    tirai_jendela : 'TERTUTUP',
    kipas_ac : 'OFF',
    lampu_aquarium : 'OFF',
    tv : 'OFF'
  }
}

wss.on('connection', function connection(ws) {
  wssClient.push(ws);
});

function broadcastMessage(message){
  for(var i=0; i<wssClient.length; i++){
    wssClient[i].send(JSON.stringify(message));
  }
}
app.use(bodyParser.json());
app.use('/static', express.static(__dirname + '/public'));

app.get('/', (req, res) => {
  res.sendFile(__dirname + '/', 'index.html');
})

app.post('/action', (req, res) => {
```

```
    var reqBody = req.body;
    if(reqBody.mode == 'mode_1'){
      broadcastMessage(SMART HOME_MODE.mode_1);
    }
    else if(reqBody.mode == 'mode_2'){
      broadcastMessage(SMART HOME_MODE.mode_2);
    }
    else if(reqBody.mode == 'mode_3'){
      broadcastMessage(SMART HOME_MODE.mode_3);
    }
    res.status(200).send(JSON.stringify({
      status : 'success'
    }));
})

app.listen(8080, () => {
    console.log("Server listen on port 8080");
})
```

- **Program *ESP-01***

```
#include <Arduino.h>
#include <ArduinoWebsockets.h>
#include <ESP8266WiFi.h>
#include <ArduinoJson.h>

const char* ssid = "BISA";
const char* password = "bayu1234";
const char* websockets_server = "ws://raspberrypi.local:3001";

using namespace websockets;

void onMessageCallback(WebsocketsMessage message);
void onEventsCallback(WebsocketsEvent event, String data);

WebsocketsClient client;

String namaAktuator = "lampu_teras";

#define RELAY_PIN 0

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  pinMode(RELAY_PIN, OUTPUT);

  for(int i = 0; i < 10 && WiFi.status() != WL_CONNECTED; i++) {
    Serial.print(".");
    delay(1000);
  }

  if(WiFi.status() == WL_CONNECTED){
    Serial.println("WiFi Connected");
    Serial.println(WiFi.localIP());
  }

  client.onMessage(onMessageCallback);
  client.onEvent(onEventsCallback);

  client.connect(websockets_server);

  client.send("Hi Server!");
  client.ping();
}
void loop() {
  client.poll();
}
void onMessageCallback(WebsocketsMessage message) {
  DynamicJsonDocument doc(1024);
  deserializeJson(doc, message.data());

  const char* actuatorAction = doc[namaAktuator];
```

```
    String actuatorActionS = actuatorAction;
    Serial.println(actuatorAction);
    if(actuatorActionS == "ON"){
      digitalWrite(RELAY_PIN, HIGH);
      Serial.print("Menyalakan ");
      Serial.println(namaAktuator);
    }
    else{
      digitalWrite(RELAY_PIN, LOW);
      Serial.print("Mematikan ");
      Serial.println(namaAktuator);
    }
}
void onEventsCallback(WebsocketsEvent event, String data) {
    if(event == WebsocketsEvent::ConnectionOpened) {
        Serial.println("Connnection Opened");
    } else if(event == WebsocketsEvent::ConnectionClosed) {
        Serial.println("Connnection Closed");
    } else if(event == WebsocketsEvent::GotPing) {
        Serial.println("Got a Ping!");
    } else if(event == WebsocketsEvent::GotPong) {
        Serial.println("Got a Pong!");
    }
}
```