BAB II
TINJAUAN PUSTAKA

1.1 Suhu

Suhu adalah besaran yang menyatakan derajat panas atau dingin suatu benda. Secara mikropis suhu menunjukan energi yang dimiliki oleh suatu zat. Setiap atom dalam suatu benda selalu berada dalam keadaan bergerak, baik itu perpindahan ataupun gerakan ditempat yang berupa getaran. Semakin besar energi atom penyusun suatu benda, maka semakin besar pula suhu benda tersebut. Alat ukur yang digunakan untuk mengukur besaranya suhu adalah thermometer.

Mengacu pada sistem Satuan Internasional (SI), satuan suhu adalah Kelvin (K). Beberapa skala suhu yang biasa digunakan yaitu Kelvin, Celcius, Fahreinheit dan Rankine. Akan tetapi di Indonesia, skala yang umum digunakan untuk mengukur suhu adalah skala Celcius. Pada skala Celcius, 0^oC adalah titik dimana air membeku dan 100^oC adalah titik didih air pada tekanan 1 atmosfer.

Skala Celcius dan Kelvin memiliki tingkatan kenaikan skala yang sama, kenaikan 1 °C sama dengan kenaikan suhu 1 K, yang membedakan hanyalah titik 0 skala. Saat skala Celcius menunjukan 0 °C maka pada skala Kelvin bernilai 273 K (atau 273.15 untuk lebih tepatnya) sehingga untuk mengkonversikan nilai Celcius ke skala Kelvin hanya perlu menambahakan 273 (atau 273.15 untuk lebih tepatnya).

1.2 *Internet of Things*

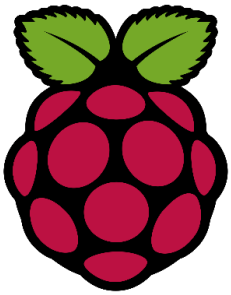
Internet of Things (IoT) adalah konsep dengan lingkungan yang dinamis dan heterogen dimana perangkat penginderaan dapat berinteraksi satu sama lain untuk beberapa tujuan tertentu [4]. IoT melibatkan penggunaan Internet untuk menghubungkan berbagai perangkat, layanan, dan sistem, sehingga membentuk jaringan objek. IoT merupakan sebuah sistem perangkat komputasi dengan konsep yang bertujuan untuk memperluas manfaat dari konektivitas internet dan dilengkapi dengan identifikasi unik antara satu dengan yang lainnya. Beberapa hal yang dapat dilakukan dengan IoT antara lain kemampuan untuk mentransfer data melalui jaringan tanpa campur tangan manusia atau dikenal sebagai komunikasi *Machine-*

to-Machine (M2M), kemampuan mengumpulkan beberapa jenis data dan kemampuan untuk mentransmisikannya ke titik lain. Prinsip utama dari IoT adalah sebagai sarana untuk memudahkan pemantauan dan pengendalian perangkat dengan begitu konsep IoT sangat memungkinkan untuk digunakan hampir pada seluruh kegiatan sehari-hari [5]. Manfaat yang didapatkan dari konsep IoT ialah pekerjaan yang dilakukan menjadi lebih cepat, mudah dan efisien.

Dalam sistem monitoring ini cara kerja IoT yaitu dengan memanfaatkan sebuah algoritma pemrograman yang menghasilkan perintah argumen dimana tiap perintah tersebut menghasilkan sebuah interaksi sesama perangkat yang terhubung secara otomatis tanpa terhalang jarak. Internet menjadi media penghubung antara kedua interaksi perangkat tersebut. Manusia hanya bertugas sebagai pengawas dan pengatur terhadap kerja alat tersebut [6]. Pada tugas akhir ini, IoT menghubungkan semua perangkat ke internet. Perangkat yang dimaksud yaitu perangkat monitoring dan perangkat pengendali. Bagian dari sebuah sistem yang dimonitoring adalah suhu ruangan dan bagian yang dikendalikan adalah AC. IoT akan mengakuisisi data dari sensor kemudian mentransfer data tersebut sehingga dapat diakses dan diolah oleh pengguna.

1.3 Raspberry Pi

Raspberry Pi merupakan sebuah mini komputer yang dapat digunakan seperti sebuah *Personal Computer* (PC) atau di sebut juga sebagai komputer mini. Raspberry Pi menggunakan *Operating System* (OS) agar bisa di operasikan. Berbeda dengan mikrokontroler, Raspberry Pi dapat mengontrol lebih dari 1 unit *device*. Untuk pengontrolan unit *device*, Raspberry Pi menggunakan bahasa Python sebagai bahasa pemrogramannya [7]. Raspberry Pi telah dilengkapi dengan semua fungsi layaknya sebuah komputer lengkap, menggunakan *System-on-a-chip* (SoC) ARM yang dikemas dan diintegrasikan diatas PCB [8].



Gambar 2.1. Logo Raspberry Pi

Raspberry Pi memiliki dua model yaitu model A dan model B. Perbedaan model A dan B terletak pada memori yang digunakan. Model A menggunakan memori 256 MB dan model B 512 MB. Selain itu model B juga sudah dilengkapai dengan *ethernet port* (kartu jaringan) yang tidak terdapat di model A. Raspberry Pi bersifat *open source* (berbasis Linux), Raspberry Pi bisa dimodifikasi sesuai kebutuhan penggunanya. Sistem operasi utama Raspberry Pi menggunakan Debian GNU/Linux dan bahasa pemrograman Python. Salah satu pengembang OS untuk Raspberry Pi telah meluncurkan sistem operasi yang dinamai Raspbian, Raspbian diklaim mampu memaksimalkan perangkat Raspberry Pi. Sistem operasi tersebut dibuat berbasis Debian yang merupakan salah satu distribusi Linux OS [8].

2.3.1 Raspberry Pi 3 Model B+

Raspberry Pi 3 Model B+ adalah produk terbaru dalam rangkaian Raspberry Pi 3 dan merupakan generasi ketiga dari Raspberry Pi. Raspberry Pi 3 Model B+ memiliki prosesor *quad core* 64-bit yang berjalan pada 1.4 GHz, *dual-band* 2.4 GHz dan LAN nirkabel 5 GHz, *Bluetooth* 4.2 / BLE, *Ethernet* lebih cepat dan kapabilitas PoE melalui PoE HAT terpisah [9]. Raspberry Pi 3 Model B+ direkomendasikan untuk digunakan dalam proyek-proyek yang membutuhkan daya yang sangat rendah. Raspberry Pi 3 model B+ terdiri dari 40 pin I/O digital port dan USB untuk mengkoneksikan berbagai perangkat USB seperti LCD *display*, *keyboard*, *mouse*, dan lain-lain. Dari 40 pin, 26 pin GPIO dan yang lain adalah pin *power* atau *ground* (ditambah dua pin ID EEPROM). Mini USB port digunakan untuk menghubungkan ke power adaptor. Raspberry Pi sudah mendukung

audio/video. Untuk mengkoneksikan ke monitor/tv dapat menggunakan HDMI atau RCA. Dalam koneksi ke jaringan bisa menggunakan port Ethernet/LAN atau pada Raspberry Pi 3 Model B+ sudah dilengkapi dengan *wifi built-in* [10]. Raspberry Pi 3 Model B+ mempertahankan jejak mekanis yang sama seperti Raspberry Pi 2 Model B dan Raspberry Pi 3 Model B.

Gambar 2.2. Raspberry Pi 3 Model B+
(*Sumber : raspberrypi.org*)

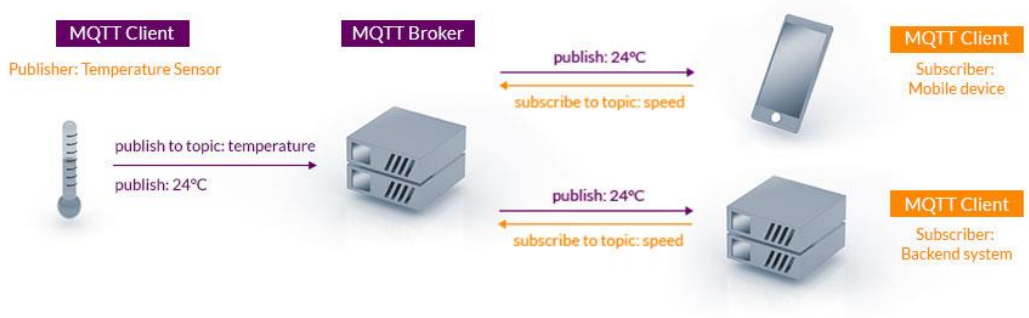
2.3.2 Raspberry OS

Sistem operasi yang digunakan oleh Raspberry Pi biasa disebut Rasbian. Raspbian adalah sistem operasi bebas berbasis Debian GNU / LINUX dan dioptimalkan untuk perangkat keras Raspberry Pi (arsitektur prosesor ARMHF) [11]. Raspbian dilengkapi dengan lebih dari 35.000 paket, atau perangkat lunak *precompiled* paket dalam format yang bagus untuk kemudahan instalasi pada Raspberry Pi. Awal di rilis sejak Juni 2012, menjadi distribusi yang terus aktif dikembangkan dengan penekanan pada peningkatan stabilitas dan kinerja sebanyak mungkin. Meskipun Debian menghasilkan distribusi untuk arsitektur lengan, *Rasbian* hanya kompatibel dengan versi yang lebih baru dari yang digunakan pada Raspberry Pi (ARMv7 CPU-A dan vs Raspberry Pi ARMv6 CPU yang lebih tinggi).

1.4 *Message Queuing Telemetry Transport*

Message Queuing Telemetry Transport (MQTT) adalah protokol perpesanan standar OASIS untuk *Internet of Things* (IoT). MQTT menggunakan prinsip *publish/subscribe*, yang menghasilkan info tertentu dan menerbitkan info tersebut disebut publisher dan sangat sederhana dengan protokol pesan yang ringan (*lightweight*) [12]. Dirancang untuk memberikan latency rendah, pengiriman pesan yang terjamin dan distribusi yang efisien ke satu atau banyak penerima. MQTT adalah protokol konektivitas *machine-to-machine* (M2M) atau *Internet of Things* (IoT) sangat ideal untuk perangkat yang terhubung dan aplikasi mobile. Protokol ini berfokus pada meminimalkan jumlah bytes pada sebuah jalur dan penggunaan daya yang rendah.

Pada penerapannya, *device* akan terhubung pada sebuah **broker** dan mempunyai *topic* tertentu. Broker pada MQTT berfungsi untuk meng*handle* data *publish* dan *subscribe* dari berbagai *device*, bisa diibaratkan sebagai *server* yang memiliki alamat IP khusus. Beberapa contoh dari *Broker* yang ada seperti Mosquitto, HiveMQ dan Mosca. Pada sistem kerja MQTT protokol ini, *topic* bersifat wajib. Pada setiap transaksi data antara *Publisher* dan *Subscriber* harus memiliki suatu *topic* tertentu.



Gambar 2.3. Arsitektur MQTT *Publish/Subcribe*
(Sumber: https://mqtt.org/)

Pada **Gambar 2.3** di atas, terdapat tiga bagian penting yaitu *Publisher, Broker dan Subscriber*. Setiap bagian memiliki tugas masing-masing sebagai berikut :

1. Blok *Publisher*

   Pada blok Publisher, terdapat beberapa sensor, seperti contoh terdapat sensor A, B dan C. Setiap sensor akan terhubung ke suatu *gateway* seperti *Ethernet shield*, ESP8266, SIM800L atau *device* sejenisnya yang dapat menghubungkan kita ke MQTT *Broker*. Antara *gateway* dan sensor pasti terdapat *controller*, *controller* yang dapat digunakan seperti Arduino, Raspberry Pi atau sejenisnya. Tugas dari Blok *Publisher* yaitu mengirim data yang diambil dari sensor A, B dan C ke suatu MQTT *Broker* dengan Topic data1.

2. MQTT *Broker*

   MQTT *Broker* memiliki suatu alamat yang dapat diakses oleh Publisher dan Subscriber. Tugas dari MQTT *Broker* yaitu sebagai penghubung transaksi data antara *publisher* dan *subscriber*. MQTT *Broker* juga mengenali suatu data lewat sebuah pengelompokan atau biasa disebut topic. Ketika *Publisher* mengirim data sensor A, B, C dengan topic data1, dan suatu saat terdapat *Subscriber* yang melakukan *subscribe* dengan *topic* yang sama data1, maka bisa dipastikan *Subscriber* akan menerima data sensor A, B dan C dari *Publisher*.

3. Blok *Subscriber*

   Blok *Subscriber* bertugas untuk melakukan subscribe data pada topic data1. Setelah mendapatkan data yang berupa nilai sensor A, B dan C dari *publisher*, kita dapat mengolah data-data tersebut untuk dimasukkan ke dalam *database*, dianalisis atau dapat kita proses menjadi sebuah sistem monitoring yang terstruktur.

2.4.1 *Quality of Service*

Tingkat *Quality of Service* (QoS) adalah kesepakatan antara pengirim pesan dan penerima pesan yang menentukan jaminan pengiriman untuk pesan tertentu. QoS nerupakan fitur utama dari protokol MQTT. QoS memberi klien kekuatan

untuk memilih tingkat layanan yang sesuai dengan keandalan jaringan dan logika aplikasinya karena MQTT mengelola transmisi ulang pesan dan menjamin pengiriman. QoS membuat komunikasi dalam jaringan yang tidak dapat diandalkan menjadi jauh lebih mudah. Ada 3 level QoS di MQTT:

1. QoS 0 atau *At most once*

   Tingkat layanan ini menjamin pengiriman dengan upaya terbaik. Tidak ada jaminan pengiriman. Penerima tidak mengakui telah menerima pesan tersebut dan pesan tersebut tidak disimpan dan dikirim ulang oleh pengirim. QoS 0 sering disebut dengan *fire and forget*.

2. QoS 1 atau *At least once*

   QoS 1 menjamin bahwa pesan dikirim setidaknya satu kali ke penerima. Pengirim menyimpan pesan sampai mendapatkan paket PUBACK dari penerima yang menyatakan penerimaan pesan tersebut. Mungkin saja sebuah pesan dikirim beberapa kali.

   Pengirim menggunakan pengidentifikasi paket di setiap paket untuk mencocokkan paket PUBLIKASIKAN dengan paket PUBACK yang sesuai. Jika pengirim tidak menerima paket PUBACK dalam jangka waktu yang wajar, pengirim mengirim ulang paket PUBLIKASIKAN. Ketika penerima mendapat pesan dengan QoS 1, ia dapat segera memprosesnya. Misalnya, jika penerima adalah *broker*, *broker* mengirim pesan ke semua klien yang berlangganan dan kemudian membalas dengan paket PUBACK.

   Jika klien penerbitan mengirim pesan lagi itu menetapkan bendera duplikat (DUP). Di QoS 1, flag DUP ini hanya digunakan untuk keperluan internal dan tidak diproses oleh *broker* atau klien. Penerima pesan mengirim PUBACK, apa pun bendera DUPnya.

3. QoS 2 atau *Exactly once*

   QoS 2 adalah tingkat layanan tertinggi di MQTT. Level ini menjamin bahwa setiap pesan hanya diterima satu kali oleh penerima yang dituju. QoS 2 adalah

kualitas tingkat layanan yang paling aman dan paling lambat. Jaminan diberikan oleh setidaknya dua aliran permintaan / tanggapan antara pengirim dan penerima. Pengirim dan penerima menggunakan pengidentifikasi paket dari pesan PUBLIKASIKAN asli untuk mengoordinasikan pengiriman pesan. Ketika penerima mendapatkan paket QoS 2 PUBLISH dari pengirim, ia memproses pesan publikasi yang sesuai dan membalas pengirim dengan paket PUBREC yang mengakui paket PUBLIKASIKAN. Jika pengirim tidak mendapatkan paket PUBREC dari penerima, pengirim akan mengirimkan paket PUBLISH lagi dengan flag duplikat (DUP) sampai menerima pengakuan. Setelah pengirim menerima paket PUBREC dari penerima, pengirim dapat dengan aman membuang paket PUBLISH awal. Pengirim menyimpan paket PUBREC dari penerima dan merespons dengan paket PUBREL.

Setelah penerima mendapatkan paket PUBREL, penerima dapat membuang semua status yang disimpan dan menjawab dengan paket PUBCOMP (hal yang sama berlaku saat pengirim menerima PUBCOMP). Sampai penerima menyelesaikan pemrosesan dan mengirim paket PUBCOMP kembali ke pengirim, penerima menyimpan referensi ke pengidentifikasi paket dari paket PUBLISH asli. Langkah ini penting untuk menghindari pemrosesan pesan untuk kedua kalinya. Setelah pengirim menerima paket PUBCOMP, pengidentifikasi paket dari pesan yang dipublikasikan tersedia untuk digunakan kembali. Ketika aliran QoS 2 selesai, kedua belah pihak yakin bahwa pesan terkirim dan pengirim memiliki konfirmasi pengiriman.

Jika sebuah paket hilang di sepanjang jalan, pengirim bertanggung jawab untuk mengirimkan kembali pesan tersebut dalam jangka waktu yang wajar. Ini juga benar jika pengirimnya adalah klien MQTT atau broker MQTT. Penerima memiliki tanggung jawab untuk menanggapi setiap pesan perintah yang sesuai.

Saat akan mengimplementasikan QoS di MQTT, perlu mempertimbangkan dua sisi pengiriman pesan sebagai berikut :

1. Pengiriman pesan dari *publishing client* ke *broker*
2. Pengiriman pesan dari *broker* ke *subcribing client* yang *subscribe*

Klien yang mempublikasikan pesan ke broker menentukan tingkat QoS dari pesan ketika mengirim pesan ke broker. Broker mengirimkan pesan ini ke *subcribing client* menggunakan tingkat QoS yang ditentukan oleh setiap klien yang *subscribe* selama proses berlangganan. Jika klien yang *subscribe* menentukan QoS yang lebih rendah dari pada *publishing client*, broker mengirimkan pesan dengan kualitas layanan yang lebih rendah.

Berikut beberapa pedoman yang dapat membantu dalam proses pengambilan keputusan untuk menentukan QoS yang tepat:

1. QoS 0 digunakan ketika:
   a. Memiliki koneksi yang sepenuhnya atau sebagian besar stabil antara pengirim dan penerima. Kasus penggunaan klasik untuk QoS 0 menghubungkan klien uji atau aplikasi ujung depan ke broker MQTT melalui koneksi kabel.
   b. Tidak keberatan jika beberapa pesan hilang sesekali. Hilangnya beberapa pesan dapat diterima jika datanya tidak terlalu penting atau bila data dikirim dalam interval pendek.
   c. Tidak membutuhkan antrean pesan. Pesan hanya diantrekan untuk klien yang terputus jika mereka memiliki QoS 1 atau 2 dan sesi tetap.

2. QoS 1 digunakan ketika:
   a. Perlu mendapatkan setiap pesan dan kasus penggunaan dapat menangani duplikat. QoS level 1 adalah tingkat layanan yang paling sering digunakan karena menjamin pesan sampai setidaknya sekali tetapi memungkinkan untuk beberapa pengiriman. Aplikasi harus mentolerir duplikat dan dapat memprosesnya sesuai kebutuhan.
   b. Tidak dapat menanggung *overhead* QoS 2. QoS 1 mengirimkan pesan jauh lebih cepat daripada QoS 2.

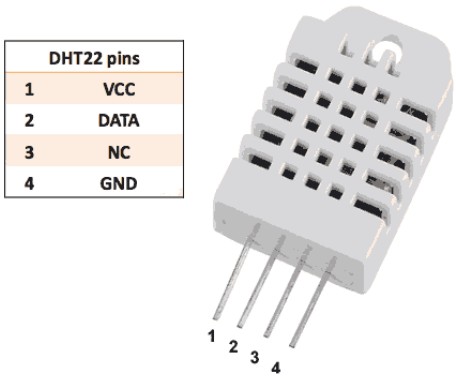
3. QoS 2 digunakan ketika aplikasi harus menerima semua pesan tepat satu kali. Hal ini sering terjadi jika pengiriman duplikat dapat merugikan pengguna aplikasi atau klien yang berlangganan. Waspadai *overhead* dan interaksi QoS 2 membutuhkan lebih banyak waktu untuk diselesaikan.

1.5 DHT22

DHT22 merupakan sensor yang digunakan untuk mengetahui nilai suhu dan kelembaban. Sensor DHT22 terdiri dari 2 buah sensor didalamnya yaitu sensor kelembaban yang berupa *capacitive-type humidity* untuk pengukur kelembaban sensor ini bekerja berdasarkan perubahan kapasistas kapasitor apabila ada objek yang berada dalam daerah deteksinya yaitu adanya molekul air di udara dan sebuah *temperature module* untuk mengatur suhu yang terbuat dibuat dari campuran bahan semikonduktor yang dapat menghasilkan hambatan interistik yang akan berubah terhadap *temperature*. Sensor DHT22 merupakan sensor pengukur suhu dan kelembaban relatif dengan keluaran berupa sinyal digital [13].

Sensor ini memiliki kalibrasi akurat dengan kompensasi suhu ruang penyesuaian dengan nilai koefisien tersimpan dalam memori OTP terpadu. Sensor DHT22 memiliki rentang pengukuran suhu dan kelembaban yang luas, DHT22 mampu mentransmisikan sinyal keluaran melewati kabel hingga 20 meter sehingga sesuai untuk ditempatkan di mana saja . Tapi jika kabel yang panjang di atas 2 meter harus ditambahkan *buffer capacitor* 0,33µF antara pin#1 (VCC) dengan pin#4 (GND).

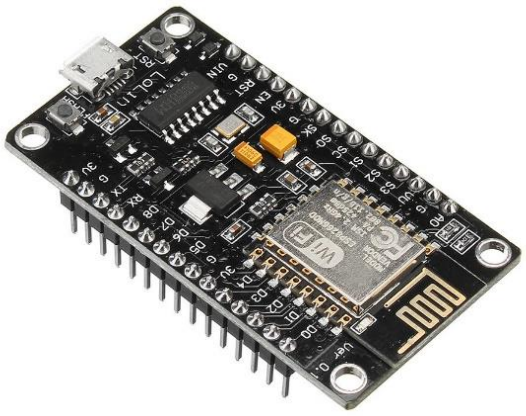
Cara kerja dari sensor DHT22 yaitu bahan semikonduktor pada sensor suhu dan kelembaban membaca nilai suhu dan kelembaban kemudian data dikirimkan ke *wemos* dalam bentuk digital secara beriringan, waktu pengiriman data antara data suhu dan kelembaban sangatlah singkat yaitu kurang dari 40ms, sehingga pembacaan akan terlihat seperti bersamaan. Untuk dapat mengirim data digital pengukuran pertama mikrokontroler mengirimkan data hasil pengukuran kelembaban kemudian suhu seacara beriringan.



Gambar 2.4 Sensor DHT22

1.6 NodeMCU ESP8266

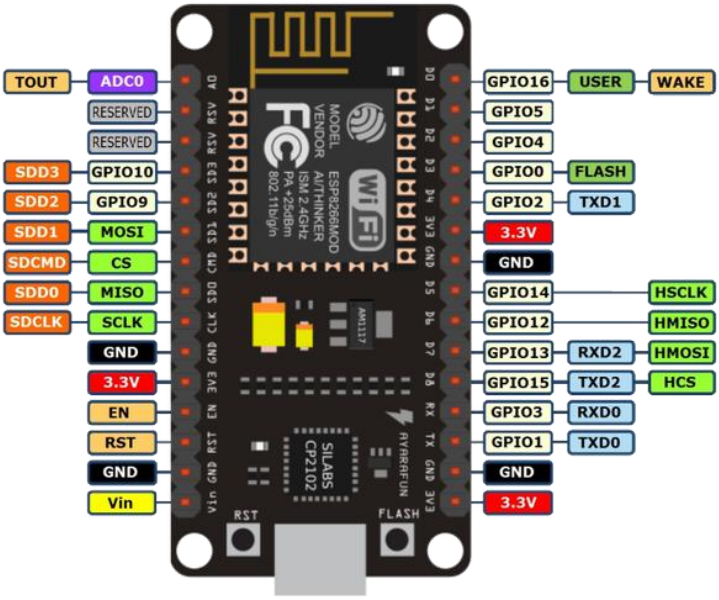
NodeMCU ESP8266 merupakan sebuah mikrokontroler pengembangan kit berbasis ESP8266 yang sudah berisikan GPIO, PWM (Pulse Width Modulation), IIC, 1 *Wire* dan ADC (Analog to Digital Converter) semua dalam satu *board*.. Dengan mengguanakn Node MCU ESP8266 membantu pembuat dalam membuat *prototype* produk IoT.



Gambar 2.5 NodeMCU ESP8266

NodeMCU dapat dianalogikan sebagai *board* arduino-nya ESP8266. Program ESP8266 sedikit merepotkan karena diperlukan beberapa teknik wiring serta tambahan modul *USB to serial* untuk mengunduh program. Namun NodeMCU telah *me-package* ESP8266 ke dalam sebuah *board* yang kompak dengan berbagai fitur layaknya mikrokontroler + kapabilitas akses terhadap WiFi

juga chip komunikasi *USB to serial* sehingga kita bisa melakukan *programming* langsung ke ESP8266 tanpa membutuhkan mikrokontroler tambahan. Kelebihan lainnya ESP8266 ini dapat menjalankan peran sebagai *adhoc* akses poin maupun klien sekaligus.



Gambar 2.6 Skema Pin NodeMCU ESP8266

Gambar di atas merupakan kaki pin yang ada pada NodeMCU ESP8266. Berikut keterangan dari pin-pin tersebut.

Tabel 2.1 Keterangan Pin-pin pada NodeMCU ESP8266

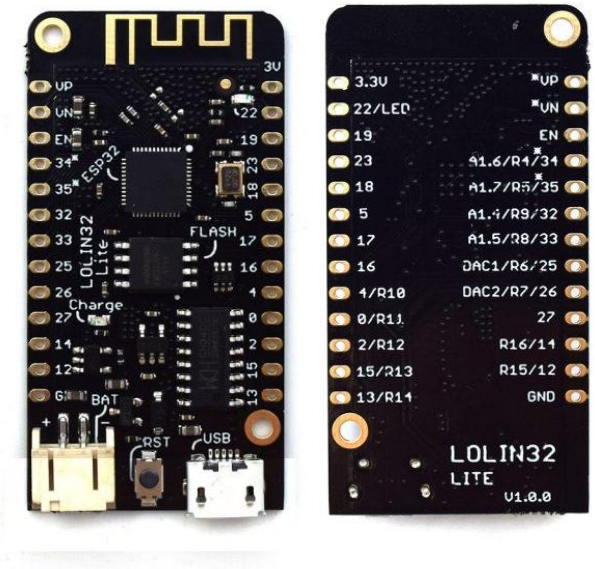
Spesifikasi	**Keterangan**
ADC	Analog Digital Converter. Rentang tegangan masukan 0-1v, dengan skup nilai digital 0-1024
MOSI	Main output slave input
CS	Chip selection
MISO	Main input slave output
SCLK	Clock
RST	Berfungsi mereset modul
EN	Chip Enable, Active High

Vin	Catu daya 5V
GND	Ground
RXD	UART0_RXD; GPIO3
TXD	UART0_TXD; GPIO1
GPIO	General-purpose input/output merupakan pin generic pada suatu chip yang karakternya pada pin input ataupun output dapat dikontrol dan deprogram oleh pengguna. Adanya pin GPIO dapat menghemat dan mempermudah saat mengatur sirkuit tambahan.

Alasan dipilihnya NodeMCU ESP8266 karena mudah deprogram dan memiliki pin I/O yang memadai dan dapat mengakses jaringan Internet untuk mengirim atau mengambil data melalui konesi WiFi.

1.7 Wemos LOLIN32 Lite – ESP32

Wemos LOLIN32 Lite merupakan *board* berbasis ESP32 dengan WiFi dan *Bluetooth*. Pada mikrokontroler ini sudah tersedia modul WiFi dalam chip sehingga sangat mendukung untuk membuat sistem aplikasi *Internet of Things*. Wemos LOLIN32 Lite memiliki koneksi USB mikro yang dapat diprogram atau diberi daya. *Board* ini memiliki chip ESP32 dengan memori *flash* 4 MB. ESP32 versi Lite ini memiliki lebih sedikit pin digital dan analog dibandingkan versi standar.



Gambar 2.7 Wemos LOLIN32 Lite

Tabel 2.2 Spesifikasi Wemos LOLIN32 Lite

Spesifikasi	**Keterangan**
Operating Voltage	3.3 V
SoC	Espressive ESP32-DOWD6Q Rev 1.0. Tensilica Xtensa LX6 (Prosesor inti ganda 240 MHz)
Static RAM	520 kB
Connectivity	WiFi 802. 11 b/g/n, Bluetooth LE 4.x
Access	I/O via connector 2 x 13 pin for UART, 12C, SPI, VP/VN, DAC.
Input Power	Power 5V via micro USB connector or LiPo battery on JST 2.0 Connector (dapat mengisi daya hingga 500mA) 1x mikro USB Type B untuk power, Lipo battery charging, programming and debug
Flash	4 MB
Operating	MicroPython
Dimension	40 x 25 mm (tanpa *header*)

1.8 OpenHAB

Open Home Automation Bus (OpenHAB) adalah software untuk menggabungkan teknologi atau standard dari *Home Automation System* yang berbeda menjadi satu kesatuan, dan menawarkan *User Interface* yang seragam. Karena sistem yang bersifat *opensource* dan dapat digunakan untuk pengembangan pemrograman *Home Automation* maka OpenHAB dapat disebut sebuah *framework* untuk *Home Automation*. OpenHAB tidak mencoba untuk menggantikan solusi dari *Home Automation* yang sudah ada, tetapi lebih kepada meningkatkannya [14]. OpenHAB sangat sesuai dengan konsep untuk penerapan IoT.

OpenHAB ditulis dalam Java. OpenHAB memerlukan JVM dan dapat digunakan pada server yang menjalankan berbagai sistem operasi, instansi Raspberry Pi khusus , atau beberapa sistem penyimpanan yang terhubung dengan jaringan. OpenHAB mendukung sejumlah *backend* persistensi untuk menyimpan dan menanyakan data, termasuk basis data relasional dan time series. Agar dapat memberikan akses jarak jauh yang aman dan memungkinkan pengguna OpenHAB untuk memonitor, mengontrol, dan mengarahkan perangkat dari jarak jauh melalui internet terdapat OpenHAB Cloud yang merupakan layanan *cloud* pendamping dan *backend* untuk OpenHAB.

1.9 MicroPython

MicroPython adalah implementasi yang lebih ramping dan efisien dari bahasa pemrograman Python 3 yang mencakup sebagian kecil dari pustaka standar Python dan dioptimalkan untuk perangkat mikrokontroler [15]. Micropython berjalan diatas "*bare-metal*" mikrokontroler secara langsung, tidak seperti Python yang memang dirancang berjalan diatas sistem operasi. Beberapa optimasi dibuat guna membuat micropython dapat berjalan pada perangkat dengan resource yang terbatas. Dalam mengimplementasikan pilihan pustaka inti Python, MicroPython menyertakan modul seperti "mesin" untuk mengakses perangkat keras tingkat rendah.

1.10 PuTTY

PuTTY adalah sebuah *software remote console* atau terminal yang bersifat *open source* yang dapat digunakan untuk melakukan protokol jaringan SSH dan Telnet. Protokol ini dapat digunakan untuk menjalankan sesi r*emote* pada sebuah komputer melalui sebuah jaringan, baik itu LAN, maupun internet. Program ini banyak digunakan oleh para pengguna komputer tingkat menengah ke atas, yang biasanya digunakan untuk menyambungkan, mensimulasi, atau mencoba berbagai hal yang terkait dengan jaringan [7].

1.11 Wireshark

Wireshark adalah aplikasi penganalisis protokol jaringan [16]. Cara kerja Wireshark yaitu menangkap paket-paket data dari protokol-protokol yang berbeda dari berbagai tipe jaringan yang umum ditemukan di dalam trafik jaringan internet. Paket-paket data tersebut ditangkap lalu ditampilkan di jendela hasi *capture* secara *realtime*. Aplikasi ini digunakan sebagai alat *troubleshoot* pada jaringan yang bermasalah, selain itu digunakan untuk pengujian *software* karena kemampuannya untuk membaca konten tiap paket trafik data.

1.12 Komponen yang digunakan

Berikut merupakan komponen yang digunakan dalam rancang bangun sistem monitoring suhu ruangan dengan menggunakan protokol MQTT dan OpenHAB:

Tabel 2.3 Komponen yang digunakan

No	Nama Komponen
1	Raspberry Pi
2	DHT22
3	NodeMCU ESP8266
4	Wemos Lolin32 Lite - ESP32
5	Infrared LED
6	Infrared Receiver