BAB 2

TINJAUAN PUSTAKA

1.1 Landasan Teori

Landasan teori merupakan teori yang relevan yang digunakan untuk menjelaskan tentang hal apa saja yang akan diteliti dan sebagai dasar untuk memberi jawaban sementara terhadap rumusan masalah yang diajukan (hipotesis), dan penyusunan penelitian. Teori yang digunakan bukan sekedar pendapat dari pengarang atau pendapat lain, tetapi teori yang benar-benar telah teruji kebenarannya.

1.1.1 Definisi Perancangan

Perancangan adalah suatu proses yang bertujuan untuk menganalisis, menilai, memperbaiki dan menyusun suatu sistem, baik sistem fisik maupun non fisik yang optimum untuk waktu yang akan datang dengan memanfaatkan informasi yang ada. Perancangan suatu alat termasuk dalam metode teknik, dengan demikian langkah-langkah pembuatan perancangan akan mengikuti metode teknik.

1.1.2 Definisi Sistem

Sistem adalah sekumpulan unsur / elemen yang saling berkaitan dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan.

Contoh:

Sistem Komputer terdiri dari Software, Hardware, dan Brainware.

Sistem Akuntansi Adapun Menurut para ahli berpendapat tentang sistem:

1. Menurut LUDWIG VON BARTALANFY

   Sistem merupakan seperangkat unsur yang saling terikat dalam suatu antar relasi diantara unsur-unsur tersebut dengan lingkungan.

2. Menurut ANATOL RAPOROT

   Sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain.

3. Menurut L. ACKOF

   Sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satu sama lainnya.

1.1.3 Flowchart

Flowchart adalah bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. Flowchart merupakan cara penyajian dari suatu algoritma.

Tujuan membuat *flowchart* :

- Menggambarkan suatu tahapan penyelesaian masalah.
- Secara sederhana, terurai, rapi dan jelas.
- Menggunakan simbol-simbol standar.

1. *Flowchart* Sistem

   Flowchart sistem adalah bagan yang memperlihatkan urutan prosedur dan proses dari beberapa file di dalam media tertentu. Melalui *flowchart* ini terlihat jenis media penyimpanan yang dipakai dalampengolahandata.

   a. Selain itu juga menggambarkan file yang dipakai sebagai *input* dan *output*.
   b. Tidak digunakan untuk menggambarkan urutan langkah untuk memecahkan masalah.
   c. Hanya untuk menggambarkan prosedur dalam sistem yang berjalan.

2. Simbol-Simbol *FlowChart*

   Simbol-simbol yang dipakai dalam *flowchart* dibagi menjadi 3 kelompok:

   a. *Flow direction symbols*

      Digunakan untuk menghubungkan simbol satu dengan yang lain. Disebut juga *connecting line.*

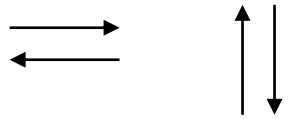
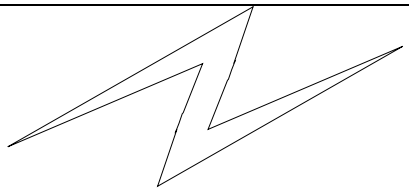
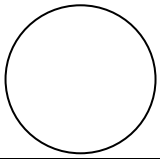
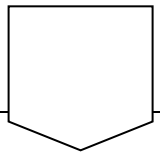
   b. *Processing symbols*

      Menunjukan jenis operasi pengolahan dalam suatu proses/ prosedur.

   c. *Input/ Output symbols*

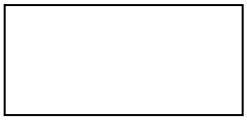
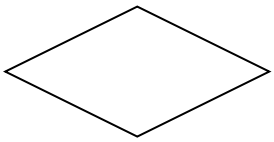
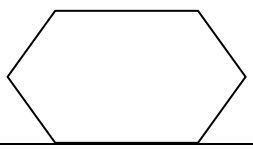
      Menunjukan jenis peralatan yang digunakan sebagai media *input* atau *output.*

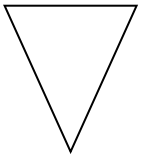
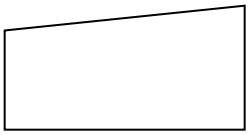
Tabel 0.1 Flow Direction Symbols

	Simbol arus / *flow*, yaitu menyatakan jalannya arus suatu proses.
	Simbol *communication link*, yaitu menyatakan transmisi data dari satu lokasi ke lokasi lain.
	Simbol *connector*, berfungsi menyatakan sambungan dari proses ke proses lainnya dalamhalaman yang sama.
	Simbol*offline connector*, menyatakan sambungan dari proses ke proses lainnya dalamhalaman yang berbeda.

Sumber: Penulis 2021

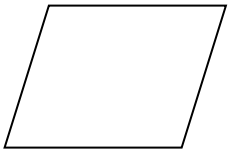
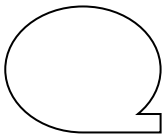
Tabel 0.2 Processing Symbols

	Simbol *process*, yaitu menyatakan suatu tindakan(proses) yang dilakukan oleh komputer.
	Simbol *manual*, yaitu menyatakan suatu tindakan (proses) yang tidak dilakukan oleh komputer.
	Simbol *decision*, yaitu Menujukkan suatu kondisitertentu yang akan menghasilkan dua kemungkinan jawaban : ya / tidak.
	Simbol *predefined process*, yaitu menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberi harga awal.

	Simbol terminal, yaitu menyatakan permulaan atau akhir suatu program.
	Simbol *keying operation*, Menyatakan segala jenis operasi yang diproses dengan menggunakan suatu mesin yang mempunyai *keyboard*.
	Simbol *offline - storage*, Menunjukkan bahwa data dalam simbol ini akan disimpan ke suatu media tertentu.
	Simbol *manual input*, memasukkan datasecara *manual* dengan menggunakan *online keyboard*.

Sumber: Penulis (2021)

Tabel 0.3 Input / OutputSymbols

	Simbol *input/output*, menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya
	Simbol *punched card*, menyatakan *input* berasal darikartu atau *output* ditulis ke kartu.
	Simbol *magnetic tape*, menyatakan *input* berasal dari pita magnetis atau *output* disimpan ke pita magnetis.
	Simbol *diskstorage*, menyatakan *input* berasal dari dari disk atau *output* disimpan kedisk.
	Simbol *document*,mencetak keluaran dalam bentuk dokumen (melalui printer).
	Simbol *display*, mencetak keluarandalam layar monitor.

Sumber: Penulis (2021)

1.1.4 Monitoring

Perananan monitoring yaitu untuk memantau dan mengontrol perkembangan yang sedang terjadi serta mengenali apakah pelaksanaan tindakan sudah sesuai dengan apa yang telah di rencanakan dan mengetahui perubahan dan peningkatan yang telah terjadi dengan tindakan yang telah dilakukan.

1.1.5 Internet Of Things (Iot)

Pengertian IoT (Internet of Things) adalah suatu konsep/skenario dimana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan tanpa memerlukan interaksi manusia ke manusia atau manusia ke komputer dan CPS (Cyber Physical Systems) adalah suatu sistem yang berfungsi untuk melindungi fisik dari sebuah benda maupun lainnya. Menurut Coordinator and support action for global RFID-related activities and standadisation menyatakan internet of things (IoT) sebagai sebuah infrastruktur koneksi jaringan global, yang mengkoneksikan benda fisik dan virtual melalui eksploitasi data capture dan teknologi komunikasi.

1.1.6 Internet

Internet adalah jaringan yang menghubungkan antara komputer yang satu dengan komputer lainya dan menggunakan standar sistem global Transmission Control Protocol atau Internet Protocol Suite (TCP/IP) sebagai protokol pertukaran sehingga kita bisa saling berkomunikasi, berinteraksi, dan saling bertukar informasi meski dalam jarak yang jauh. Internet adalah singkatan dari interconnection networking yang secara sederhana bisa diartikan sebagai a global network of computer networks.

1.1.7 Entity Relationship Diagram (ERD)

1.Entitas

Entiti merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Simbol dari entiti ini biasanya digambarkan dengan persegi panjang.

2.Atribut

Setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendes-kripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasikan isi elemen satu dengan yang lain. Gambar atribut diwakili oleh simbol elips.

3.Atribut Key

Atribut Key adalah satu atau gabungan dari beberapa atribut yang dapat membedakan semua baris data (Row/Record) dalam tabel secara unik. Dikatakan unik jika pada atribut yang dijadikan key tidak boleh ada baris data dengan nilai yang sama

Contoh : Nomor pokok mahasiswa (NPM), NIM dan nomor pokok lainnya

4.Hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda.

1.1.8 Data Flow Diagram (DFD)

Data Flow Diagram adalah diagram sistem yang menggambarkan cara kerja aplikasi secara logic. Mulai dari tikat paling tinggi sampai dengan tingkat paling rendah. Pada perancangan ini terdiri dari perancangan awal (preliminary design) dan perancangan rinci (detailed design) sesuai dengan tahap-tahap rekayasa perangkat lunak.Adapun penjelasan dari perancangan awal adalah perancangan sistem yang menggambarkan tentang hubungan antara sistem dengan lingkungan luar sistem

1.1.9 Telegram

Telegram merupakan aplikasi pesan instan multiplatform berbasis cloud yang gratis dan bersifat nirbala. Telegram dikembangkan oleh perusahaan Telegram Messenger LLP didukung wirausahawan Rusia Pavel Durov. Kode client-side Telegram bersifat gratis, sedangkan server-side tertutup dan hanya dimiliki perusahaan. Layanan Telegram juga menyediakan API untuk pengembang (developers) agar dapat membuat stiker animasi, perubahan tampilan, widgets, hingga bot. Informasi dan kebutuhan API Telegram

1.2 Perangkat Keras Yang Dibutuhkan

1.2.1 Arduino Uno

Arduino Uno adalah *board* mikrokontroler berbasis Atmega328. Memiliki 14 pin*input* dari *outputdigital* dimana 6 pin*input* tersebut dapat digunakan sebagai *output* PWM dan 6 pin*inputanalog*, 16 MHz osilator kristal, koneksi USB, *jack power*, ICSP *header*, dan tombol *reset*. Untuk mendukung mikrokontroler agar dapat digunakan, cukup hanya menghubungkan *Board* Arduino Uno ke komputer dengan menggunakan kabel USB atau listrik dengan AC yang-ke adaptor-DC atau baterai untuk menjalankannya.

Nama "Uno" berarti satu dalam bahasa italia, untuk menandai peluncuran Arduino 1.0. Uno dan versi 1.0 akan menjadi versi referensi dari Arduino. Uno adalah yang terbaru dalam serangkaian USB Arduino, dan sebagai model referensi untuk *platform* Arduino.

Gambar 0.1 Arduino Uno

Sumber: http://electricityofdream.blogspot.com/2018/09/kegunaan-dan-fungsi-arduino.html

Uno berbeda dari semua *board* mikrokontroler diawal-awal yang tidak menggunakan *chip* khusus *driver* FTDI USB-*to-serial*. Sebagai penggantinya penerapan USB-*to-serial* adalah Atmega16U2 versi R2 (versi sebelumnya Atmega8U2). Versi Arduino Uno Rev.2 dilengkapi resistor ke 8U2 ke garis ground yang lebih mudah diberikan ke mode DFU.

Untuk keunggulan *board* Arduino Uno Revision antara lain adalah:

1. 1.0 pin*out*: ditambahkan pin SDA dan SCL di dekat pin AREF dan dua pin lainnya diletakkan dekat tombol *RESET*, fungsi IOREF melindungi kelebihan tegangan pada papan rangkaian. Keunggulan perlindungan ini akan kompatibel

juga dengan dua jenis *board* yang menggunakan jenis AVR yang beroperasi pada tegangan kerja 5V dan Arduino Due tegangan operasi 3.3V.

Tabel 0.4 Spesifikasi Arduino Uno

Mikrokontroler	ATmega328P
Tegangan Operasi	5V
Tegangan *Input* (Rekomendasi)	7-12V
Tegangan *Input* (Batas Maksimum)	6-20V
Jumlah Pin I/O *Digital*	14 (6 pin digunakan sebagai *output* PWM)
Jumlah Pin*Digital* PWM	6
Jumlah Pin*InputAnalog*	6
Arus DC Tiap Pin I/O	20 mA
Arus DC untuk pin 3.3V	50 mA
Flash Memory	32 KB (0,5 KB sebagai *bootloader*)
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Panjang	68.6 mm
Lebar	53.4 mm

Berat	25 g

Sumber : Penulis (2021)

Masing-masing dari 14 pin*digital* Uno dapat digunakan sebagai *input* atau *output*, menggunakan fungsi pin*Mode*(), *digitalWrite*(), dan *digitalRead*(). Pin ini beroperasi pada tegangan 5 volt. Setiap pin dapat memberikan atau menerima maksimum 40 mA dan memiliki resistor *pull-up internal* (terputus secara *default*) dari 20-50 kOhms. Selain itu, beberapa pin memiliki fungsi antara lain:

1. *Serial*: pin 0 (RX) dan 1 (TX) Digunakan untuk menerima (RX) dan mengirimkan (TX) data *serial* TTL. Pin ini terhubung dengan pin ATmega8U2 USB-*to-serial* TTL.
2. Eksternal Interupsi: Pin 2 dan 3 dapat dikonfigurasi untuk memicu *interrupt* pada nilai yang rendah (*low value*), *rising* atau *falling edge*, atau perubahan nilai. Lihat fungsi *attachInterrupt*() untuk rinciannya.
3. PWM: Pin 3, 5, 6, 9, 10, dan 11 Menyediakan 8-bit PWM dengan fungsi *analogWrite*().
4. SPI: pin 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK) mendukung komunikasi SPI dengan menggunakan perpustakaan SPI.
5. LED: pin 13. *Built-in* LED terhubung ke pin *digital* 13. LED akan menyala ketika diberi nilai *HIGH*.

Arduino Uno memiliki sejumlah fasilitas untuk berkomunikasi dengan komputer, Arduino lain, atau mikrokontroler lainnya. ATmega328 menyediakan UART TTL (5V) komunikasi *serial*, yang tersedia pada pin *digital* 0 (RX) dan 1 (TX). Pada ATmega16U2 saluran komunikasi *serial* melalui USB dan muncul sebagai *com port virtual* untuk perangkat lunak pada komputer. *Firmware* 16U2 menggunakan standar *driver* USB COM. Namun, pada Windows, diperlukan *file* .inf. Perangkat lunak Arduino termasuk monitor *serial* yang memungkinkan data tekstual sederhana akan dikirim ke dan dari papan Arduino. RX dan TX LED di papan akan berkedip ketika data sedang dikirim melalui *chip* USB-*to-serial* dan koneksi USB komputer (tetapi tidak untuk komunikasi *serial* pada pin 0 dan 1). ATmega328 juga mendukung I2C (TWI) dan komunikasi SPI. Perangkat lunak Arduino termasuk perpustakaan *Wire* berfungsi menyederhanakan penggunaan bus I2C.

Pin listrik yang tersedia pada Arduino Uno adalah sebagai berikut:

1. *Input* tegangan ke *board* Arduino ketika menggunakan sumber daya eksternal. Anda dapat menyediakan tegangan melalui pin ini, atau, jika Anda ingin memasok tegangan melalui colokan listrik, gunakan pin ini.
2. 5V. Pin ini merupakan *output* 5V yang telah diatur oleh regulator papan Arduino. *Board* dapat diaktifkan dengan daya, baik dari colokan listrik DC (7 - 12V), konektor USB (5V), atau pin VIN *board* (7-12V). Jika Anda memasukan tegangan melalui pin 5V atau 3.3V secara langsung (tanpa 8 melewati regulator) dapat merusak papan Arduino. Penulis tidak menyarankan itu.
3. Tegangan pada pin 3.3V. 3.3Volt dihasilkan oleh regulator on-*board*. Menyediakan arus maksimum 50 mA.
4. GND. Pin Ground.
5. IOREF. Pin ini di papan Arduino memberikan tegangan referensi ketika mikrokontroler beroperasi. Sebuah *shield* yang dikonfigurasi dengan benar dapat membaca pin tegangan IOREF sehingga dapat memilih sumber daya yang tepat agar dapat bekerja dengan 5V atau 3.3V.

Solenoid *Doorlock*

Gambar 0.2 Solenoid Doorlock

Sumber: https://robu.in/product/dc-12v-cabinet-door-lock-electric-lock-assembly-solenoid/

Solenoid *doorlock* atau solenoid kunci pintu merupakan alat elektronika yang dirancang khusus untuk mengunci pintu. Solenoid *doorlock* banyak diaplikasikan pada pengaman kunci pintu otomatis. Alat ini bekerja pada tegangan 12V. Sistem kerja pada solenoid *doorlock* ini adalah NC *(Normally Close)*. Pada kondisi normal (tidak ada tegangan) katup solenoid akan memanjang (terkunci) dan sebaliknya katup solenoid akan tertarik apabila diberi tegangan.

Tabel 0.5 Spesifikasi Solenoid Doorlock

Material	*Metal, Electronic Parts*
Tegangan Kerja	DC 12V
Power	8W
Rated Stroke and Force	10mm, 50 g
Mounting Hole Size	6.8 x 3mm
Ukuran total	6.6 x 2.7cm
Berat	147 g

Sumber : Penulis (2021)

1.2.2 Relay

Relay adalah sebuah perangkat elektronik berbentuk saklar *(switch)* yang dioperasikan secara elektromekanikal, terdiri dari 2 bagian utama yaitu *electromagnet* (*coil*) dan mekanikal (seperangkat kontak saklar/ *switch*). Relay menggunakan prinsip *electromagnetic* untuk menggerakan kontak saklar sehingga dengan arus listrik yang *relative* kecil tetap dapat menghantarkan listrik.

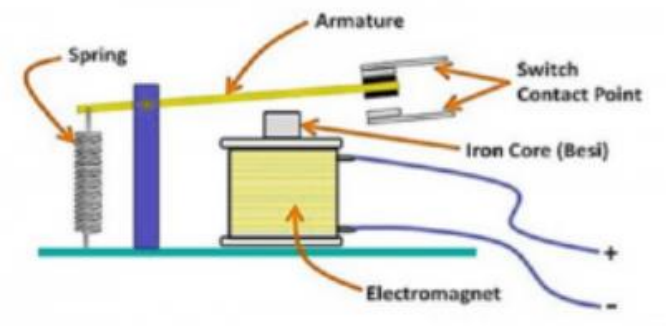
Selain itu relay juga dapat digunakan untuk mengontrol alat dengan daya yang tinggi melalui sirkuit elektronik dengan sinyal daya yang rendah. Sebagai contoh adalah rangkaian *timer* sederhana yang bekerja dibawah 5V DC biasanya tidak dapat mengontrol bola lampu yang memiliki tegangan tinggi, namun dengan penggunaan komponen relay kita dapat mengontrol bola lampu tersebut dengan mudah.

Gambar 0.3 Relay

Sumber: https://www.lazada.co.id/products/47-relay-1-channel-5v-with-opto-isolated-support-high-low-trigger-i966340293-s1456002881.html

Pada umumnya relay memiliki 4 komponen dasar yaitu:

1. *Electromagnet(coil)*
2. Armature
3. *Switch Contact Point* (saklar)
4. *Spring*



Gambar 0.4 Bagian – bagian Relay

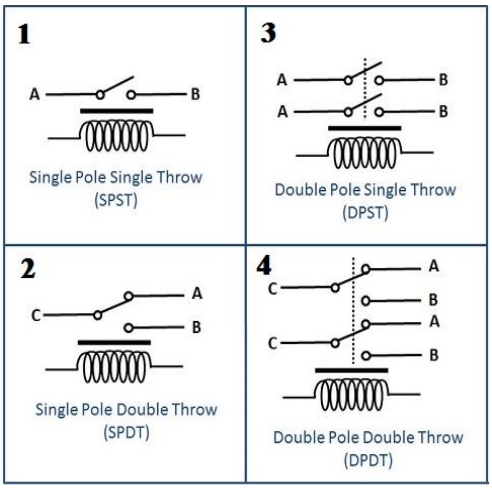
Sumber: https://abi-blog.com/relay-elektronik/

Berdasarkan gambar 2.9 sebuah besi *(iron core)* yang dililit oleh sebuah *coil* memilki fungsi sebagai pengendali. Apabila *coil* dialiri oleh arus listrik, maka akan timbul gaya elektromagnetik yang kemudian menarik armature untuk berpindah dari posisi sebelumnya (NC) ke posisi selanjutnya (NO) sehingga menjadi saklar yang dapat menghantarkan arus listrik pada saat posisi *open* (NO). Saat kondisi arus listrik tidak mengalir, armature akan kembali ke posisi awal (NC). *Coil* yang digunakan oleh relay untuk menarik kontak poin ke posisi *close* pada umumnya hanya membutuhkan arus listrik yang kecil. Selain itu relay juga berfungsi sebagai pelindung dan pemutus sirkuit saat terjadi gangguan *overload voltage* maupun *overload current*.

Kontak poin *(contact point)* pada relay terdiri dari dua jenis yaitu:

1. *Normally Close* (NC) yaitu kondisi awal sebelum perangkat diaktifkan akan selalu berada pada posisi *CLOSE* (tertentu).
2. *Normally Open* (NO) yaitu kondisi awal sebelum perangkat diaktifkan akan selalu berada pada posisi *OPEN*.

Karena relay merupakan salah satu jenis dari saklar, maka relay memiliki istilah *pole and throw* dimana *pole* adalah banyaknya kontak *(contact)* yang dimiliki oleh sebuah relay dan *throw* adalah banyaknya kondisi yang dimiliki oleh sebuah kontak.



Gambar 0.5 Jenis – jenis Relay berdasarkan pole throw

Sumber: https://sinaupedia.com/pengertian-relay/

Berdasarkan penggolongan jumlah *pole and throw* dari sebuah relay, maka relay dapat digolongkan menjadi:

1. *Single Pole Single Throw* (SPST): Relay jenis ini memiliki 4 terminal dimana 2 terminal untuk saklar dan 2 terminal yang lain digunakan untuk *coil*.
2. *Single Pole Double Throw* (SPDT): Relay jenis ini memiliki 5 terminal dimana 3 terminal digunakan untuk saklar dan 2 terminal lain untuk *coil*.
3. *Double Pole Single Throw* (DPST): Relay jenis ini memiliki 6 terminal diantaranya 4 terminal yang terdiri dari 2 pasang terminal saklar sedangkan 2 terminal lainnya untuk *coil*. Relay DPST dapat dijadikan 2 saklar yang dikendalikan oleh 1 *coil*.
4. *Double Pole Double Throw* (DPDT): Relay jenis ini memiliki 8 terminal diantaranya 6 terminal yang terdiri dari 2 pasang relay SPDT yang dikendalikan oleh 1 *(single)coil*. Sedangkan 2 terminal lainnya digunakan untuk *coil*.

Selain jenis-jenis relay diatas, terdapat pula relay yang memiliki jumlah *pole and throw*nya lebih dari dua misalnya 3PDT *(Triple Pole Double)* maupun 4PDT *(Four Pole Double).*

1.2.3 LCD (Liquid Crystal Display) 16x2

LCD atau *Liquid Crystal Display* adalah suatu jenis media *display* (tampilan) yang menggunakan kristal cair *(liquid crystal)* untuk menghasilkan gambar yang terlihat. Teknologi *Liquid Crystal Display* (LCD) atau Penampil Kristal Cair sudah banyak digunakan pada produk-produk seperti layar Laptop, layar Ponsel, layar Kalkulator, layar Jam *Digital*, layar Multimeter, Monitor Komputer, Televisi, layar Game portabel, layar Thermometer *Digital* dan produk-produk elektronik lainnya.

Adapun fitur yang disajikan dalam LCD ini adalah:

1. Terdiri dari 16 karakter dan 2 baris.
2. Mempunyai 192 karakter tersimpan.
3. Terdapat karakter generator terprogram.
4. Dapat dialamati dengan mode 4-bit dan 8-bit
5. Dilengkapi dengan *backlight*.

Gambar 0.6 LCD (Liquid Crystal Display) 16x2

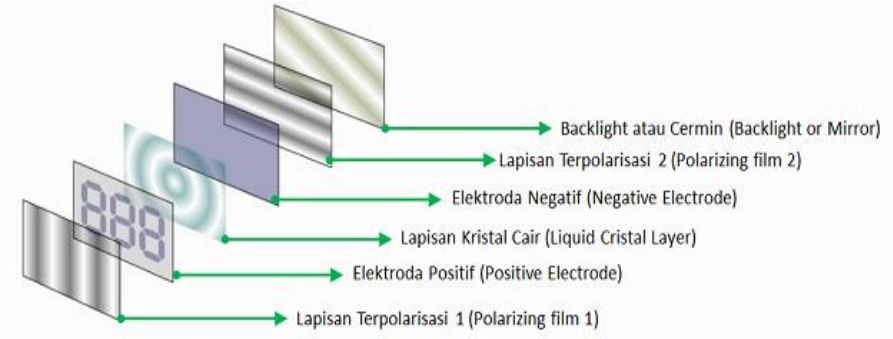
Sumber: https://www.jaycar.com.au/dot-matrix-white-on-blue-lcd-16x2-character/p/QP5521

LCD atau *Liquid Crystal Display* pada dasarnya terdiri dari dua bagian utama yaitu bagian *Backlight* (Lampu Latar Belakang) dan bagian *Liquid Crystal* (Kristal Cair). LCD hanya merefleksikan dan mentransmisikan cahaya yang melewatinya. Oleh karena itu, LCD memerlukan *Backlight* atau Cahaya latar belakang untuk sumber cahayanya. Cahaya *Backlight* tersebut pada umumnya adalah berwarna putih. Sedangkan Kristal Cair *(Liquid Crystal)* sendiri adalah cairan organik yang berada diantara dua lembar kaca yang memiliki permukaan transparan yang konduktif.

Bagian-bagian LCD atau *Liquid Crsytal Display* diantaranya adalah:

1. Lapisan Terpolarisasi 1 *(Polarizing Film 1)*
2. Elektroda Positif *(Positive Electrode)*

3. Lapisan Kristal Cair *(Liquid Crystal Layer)*
4. Elektroda Negatif *(Negative Electrode)*
5. Lapisan Terpolarisasi 2 *(Polarizing Film 2)*
6. *Backlight* atau Cermin *(Backlight or Mirror)*



Gambar 0.7 Struktur Dasar LCD (Liquid Crystal Display)

Sumber: https://teknikelektronika.com/pengertian-lcd-liquid-crystal-display-prinsip-kerja2-lcd/

Tabel 0.6 Spesifikasi Kaki LCD 16x2

Pin	**Deskripsi**

1	Ground
2	Vcc
3	Pengatur Kontras
4	"RS" *Instruction/Register Select*
5	"R/W" *Read/Write* LCD *Registers*
6	"EN" *Enable*
7-14	Data I/O Pins
15	Vcc
16	Ground

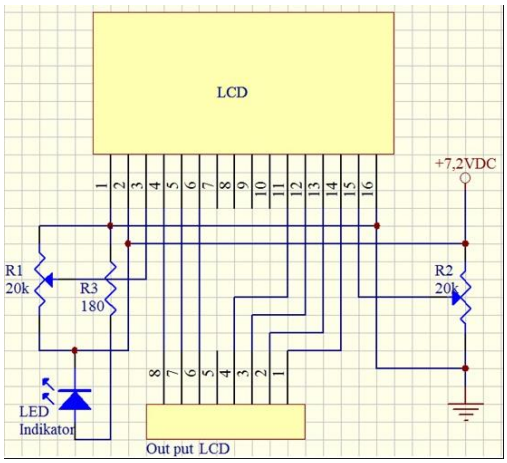
Sumber: Penulis 2021

Cara kerja LCD pada umumnya RW diberi logika rendah "0". *Bus data* terdiri dari 4-bit atau 8-bit. Jika jalur data 4-bit maka yang digunakan ialah DB4 sampai dengan DB7. Sebagaimana terlihat pada table 2.3, *interface* LCD merupakan sebuah parallel bus, dimana hal ini sangat memudahkan dan sangat cepat dalam pembacaan dan penulisan data dari atau ke LCD. Kode ASCII yang ditampilkan sepanjang 8-bit dikirim ke LCD secara 4-bit atau 8-bit pada satu waktu. Jika mode 4-bit yang digunakan, maka 2 nibble data dikirim untuk membuat sepenuhnya 8-bit (pertama dikirim 4-bit MSB lalu 4-bit LSB dengan pulsa *clock* EN setiap nibblenya). Jalur kontrol EN digunakan untuk memberitahu LCD bahwa mikrokontroller mengirimkan data ke LCD. Untuk mengirim data ke LCD program harus menset EN ke kondisi *high* "1" dan kemudian menset dua jalur kontrol lainnya (RS dan R/W) atau juga mengirimkan data ke jalur data bus.

Saat jalur lainnya sudah siap, EN harus diset ke "0" dan tunggu beberapa saat, dan set EN kembali ke *high* "1". Ketika jalur RS berada dalam kondisi *low* "0", data yang dikirimkan ke LCD dianggap sebagai sebuah perintah atau instruksi khusus (seperti bersihkan layar, posisi kursor dll). Ketika RS dalam kondisi *high* atau "1", data yang dikirimkan adalah data ASCII yang akan ditampilkan dilayar. Misal, untuk

menampilkan huruf “A” pada layar maka RS harus diset ke “1”. Jalur kontrol R/W harus berada dalam kondisi *low* (0) saat informasi pada *bus data* akan dituliskan ke LCD. Apabila R/W berada dalam kondisi *high* “1”, maka program akan melakukan *query* (pembacaan) data dari LCD. Instruksi pembacaan hanya satu, yaitu *Get* LCD *status* (membaca status LCD), lainnya merupakan instruksi penulisan. Jadi hampir setiap aplikasi yang menggunakan LCD, R/W selalu diset ke “0”. Jalur data dapat terdiri 4 atau 8 jalur (tergantung mode yang dipilih pengguna), DB0, DB1, DB2, DB3, DB4, DB5, DB6 dan DB7. Mengirim data secara parallel baik 4-bit atau 8-bit merupakan 2 mode operasi primer. Untuk membuat sebuah aplikasi *interface* LCD, menentukan mode operasi merupakan hal yang paling penting.

Mode 8-bit sangat baik digunakan ketika kecepatan menjadi keutamaan dalam sebuah aplikasi dan setidaknya minimal tersedia 11 pin I/O (3 pin untuk kontrol, 8 pin untuk data). Sedangkan mode 4-bit minimal hanya membutuhkan 7-bit (3 pin untuk kontrol, 4 pin untuk data). Bit RS digunakan untuk memilih apakah data atau instruksi yang akan ditransfer antara mikrokontroller dan LCD. Jika bit ini di set (RS = 1), maka byte pada posisi kursor LCD saat itu dapat dibaca atau ditulis. Jika bit ini di reset (RS = 0), merupakan instruksi yang dikirim ke LCD atau status eksekusi dari instruksi terakhir yang dibaca.

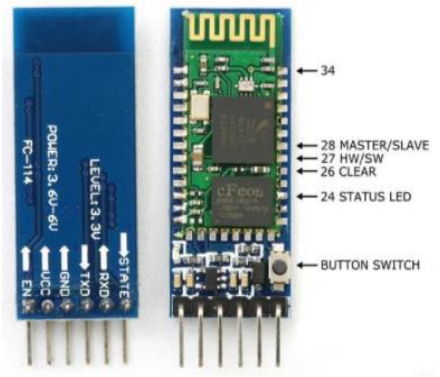


Gambar 0.8 Struktur Dasar LCD (Liquid Crystal Display)

Sumber: http://www.leselektronika.com/2012/06/liguid-crystal-display-lcd-16-x-2.html

1.2.4 Module Bluetooth HC-05

Bluetooth adalah protokol komunikasi *wireless* yang bekerja pada frekuensi radio 2.4 GHz untuk pertukaran data pada perangkat bergerak seperti PDA, laptop, HP, dan lain-lain1 . Salah satu hasil contoh modul *Bluetooth* yang paling banyak digunakan adalah tipe HC-05. Modul *Bluetooth* HC-05 merupakan salah satu modul *Bluetooth* yang dapat ditemukan dipasaran dengan harga yang relatif murah. Modul *Bluetooth* HC-05 terdiri dari 6 pin konektor, yang setiap pin konektor memiliki fungsi yang berbeda - beda.

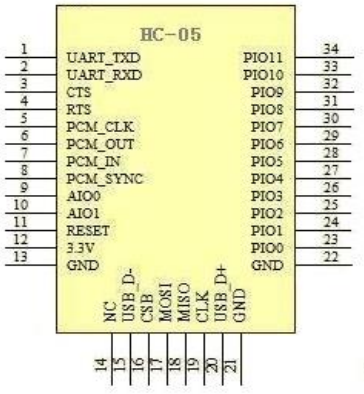


Gambar 0.9 Module Bluetooth HC-05

Sumber: https://www.tokopedia.com/endycom/modul-bluetooth-hc-05-hc05-wireless-serial-ard2uino

Modul *Bluetooth* HC-05 dengan *supply* tegangan sebesar 3,3 V ke pin 12 modul *Bluetooth* sebagai VCC. Pin 1 pada modul *Bluetooth* sebagai *transmitter*. kemudian pin 2 pada *Bluetooth* sebagai *receiver*.

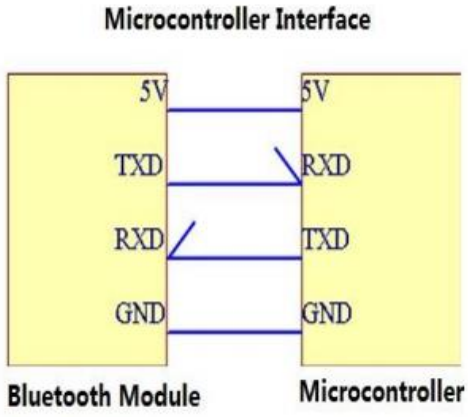
Berikut merupakan konfigurasi pin *bluetoooth* HC-05:



Gambar 0.10 Konfigurasi Pin HC-05

Sumber: https://os.mbed.com/users/edodm85/notebook/HC-05-bluetooth/

Berikut merupakan *Bluetooth-to-Serial-Module* HC-05:



Gambar 0.11 Bluetooth-to-Serial-Module

Sumber: http://tokoone.com/modul-bluetooth-modul-serial)

Konfigurasi pin modul *Bluetooth* HC-05:

Tabel 0.7 Konfigurasi pin Module Bluetooth HC-05

Pin 1	Key	-
Pin 2	VCC	Sumber Tegangan 5V
Pin *3*	GND	GroundTegangan
Pin *4*	TXD	Mengirim data
Pin *5*	RXD	Menerima data
Pin *6*	STATE	-

Sumber: Penulis 2021

Module Bluetooth HC-05 merupakan *module Bluetooth* yang bisa menjadi *slave* ataupun *master*, hal ini dibuktikan dengan bisa memberikan notifikasi untuk melakukan *pairing* keperangkat lain, maupun perangkat lain tersebut yang melakukan *pairing* ke *module Bluetooth* HC-05. Untuk mengeset perangkat *Bluetooth* dibutuhkan perintah-perintah AT *Command* yang mana perintah AT *Command* tersebut akan di respon oleh perangkat *Bluetooth* jika modul *Bluetooth* tidak dalam keadaan terkoneksi dengan perangkat lain.

Berikut adalah keterangan AT *Command Module Bluetooth* HC-05:

Tabel 0.8 AT Command Module Bluetooth HC-05

Test Komunikasi	AT	ON	-
Ganti Nama Bluetooth	AT+NAMEnamaBT	OKnamaBT	-
Ubah Pin *Code*	AT+PINxxxx	Oksetpin	xxxx digit key

Ubah Baudrate	AT+BAUD1 AT+BAUD2 AT+BAUD3 AT+BAUD4 AT+BAUD5 AT+BAUD6	OK1200 OK2400 OK4800 OK9600 OK19200 OK38400	1—1200 2—2400 3—4800 4—9600 5—19200 6—38400 7—57600 8—115200

Sumber: Penulis 2021

1.2.5 *Module* WiFi NodeMCU ESP8266

NodeMCU merupakan sebuah *open source platform* IoT dan pengembangan *kit* yang menggunakan bahasa pemrograman Lua untuk membantu dalam membuat *prototype* produk IoT atau bisa dengan memakai *sketch* dengan arduino IDE. Pengembangan *kit* ini didasarkan pada modul ESP8266, yang mengintegrasikan GPIO, PWM (*Pulse Width Modulation*), IIC, 1-*Wire* dan ADC (*Analog to Digital Converter*) semua dalam satu *board.*

NodeMCU berukuran panjang 57mm , lebar 30mm, dan berat 7 gram. *Board* ini sudah dilengkapi dengan fitur WiFi dan *Firmware*nya yang bersifat *opensource*.

Gambar 0.12 Module WiFi NodeMCU ESP8266

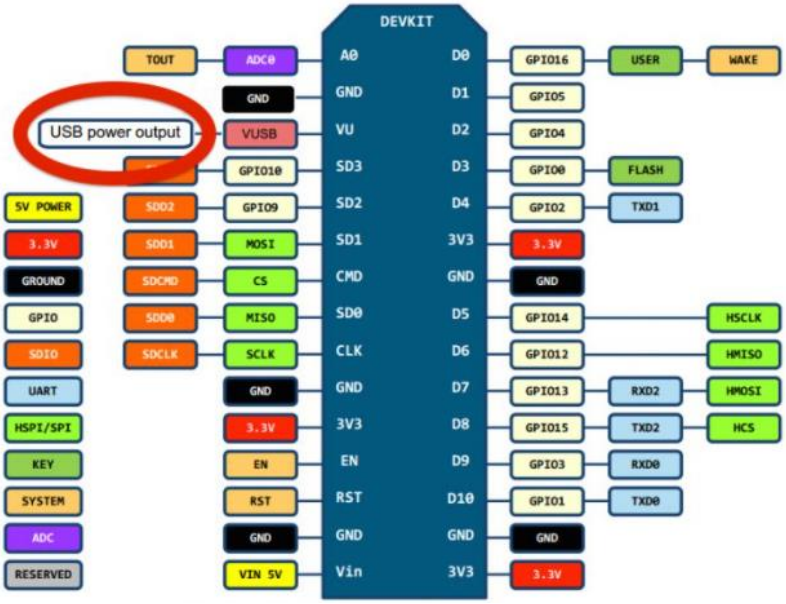
Sumber: https://www.amazon.co.uk/AZDelivery-NodeMCU-Module-ESP8266-12E-Soldered/dp/B07ZRRWP6G

Tabel 0.9 Spesifikasi NodeMCU ESP8266 V3

Mikrokontroler	ESP8266
Ukuran *Board*	57mm x 30mm
Tegangan *Input*	3.3 ~ 5V
GPIO	13 PIN
Kanal PWM	10 Kanal
10 bit ADC Pin	1 Pin
Flash Memory	4 MB
Clock Speed	40/26/24 MHz
WiFi	IEEE 802.11 b/g/n
Frekuensi	2.4 GHz ~ 22.5 GHz
USB *Port*	*Micro* USB
Card Reader	Tidak Ada
USB *to Serial Converter*	CH340G

Sumber: Penulis 2021

Berikut adalah gambar dari GPIO NodeMCU ESP8266:



Gambar 0.13 GPIO NodeMCU ESP8266 v3

Sumber: https://embeddednesia.com/v1/tutorial-nodemcu-pertemuan-pertam1a/

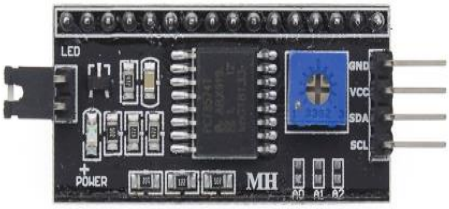
Berikut adalah masing-masing fungsi pin pada GPIO NodeMCU ESP8266 v3:

1. RST: berfungsi me*reset* modul
2. ADC: *Analog Digital Converter*. Rentang tegangan masukan 0-1v, dengan skup nilai *digital* 0-1024
3. EN: *Chip Enable, Active High*
4. IO16: GPIO16, dapat digunakan untuk membangunkan *chipset* dari mode *deep sleep*.
5. IO14: GPIO14; HSPI_CLK
6. IO12: GPIO12; HSPI_MISO
7. IO13: GPIO13; HSPI_MOSI; UART0_CTS
8. VCC: Catu daya 3.3V (VDD)
9. CS0: *Chip selection*
10. MISO: *Slave output, Main input*
11. IO9: GPIO9
12. IO10: GPIO10

13. MOSI: *Main output, Slave input*
14. SCLK: *Clock*
15. GND: Ground
16. IO15: GPIO15; MTDO; HSPICS; UART0_RTS
17. IO2: GPIO2; UART1_TXD
18. IO0: GPIO0
19. IO4: GPIO4
20. IO5: GPIO5
21. RXD: UART0_RXD; GPIO3
22. TXD: UART0_TXD; GPIO1

1.2.6 I2C (Inter Integrated Circuit)

Inter Integrated Circuit atau sering disebut I^2C adalah standar komunikasi *serial* dua arah menggunakan dua saluran yang didisain khusus untuk mengirim maupun menerima data. Sistem I^2C terdiri dari saluran SCL (*Serial Clock*) dan SDA (*Serial Data*) yang membawa informasi data antara I^2C dengan pengontrolnya. Piranti yang dihubungkan dengan sistem I2C Bus dapat dioperasikan sebagai *Master* dan *Slave*. *Master* adalah piranti yang memulai *transfer* data pada I^2C Bus dengan membentuk sinyal *Start*, mengakhiri *transfer* data dengan membentuk sinyal *Stop*, dan membangkitkan sinyal *clock*. *Slave* adalah piranti yang dialamati *master*.

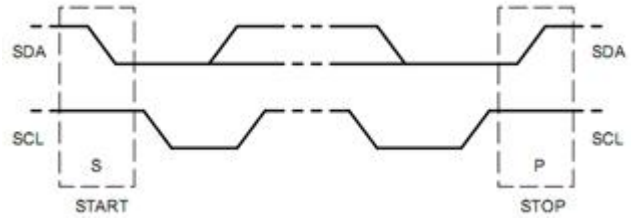


Gambar 0.14 I2C (Inter Integrated Circuit)

Sumber: https://shopee.co.id/I2C-IIC-Modul-Board-Interface-Serial-untuk-Arduino-R3-LCD-1602-Display-i.296542820.4246951012?gclid=CjwKCAjw_sn8BRBrEiwAnUGJDqqCzBnwciUyW3qE19IG6bCsjVH556MP3gqheDEsC3--mKKE00o3ogvshoC5hwQAvD_BwE

Sinyal *Start* merupakan sinyal untuk memulai semua perintah, didefinisikan sebagai perubahan tegangan SDA dari “1” menjadi “0” pada saat SCL “1”. Sinyal *Stop* merupakan sinyal untuk mengakhiri semua perintah, didefinisikan sebagai perubahan tegangan SDA dari “0” menjadi “1” pada saat SCL “1”.

Berikut merupakan kondisi sinyal *Start* dan sinyal *Stop*:

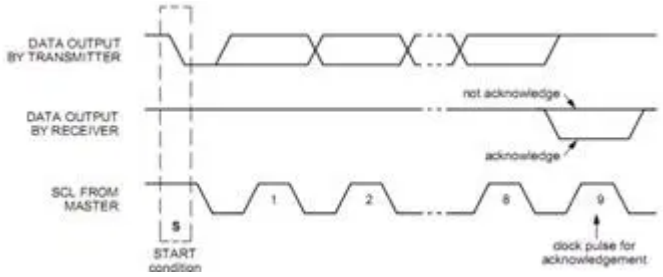


Gambar 0.15 Kondisi sinyal start dan stop

Sumber: https://purnomosejati.wordpress.com/2011/08/25/mengenal-komunikasi-i2cinter-integrated-circuit/#:~:text=Inter%20Integrated%20Circuit%20atau%20sering,untuk%20mengirim%20maupun%20meneri6636ma%20data.&text=Piranti%20yang%20dihubungkan%20dengan%20sistem,dioperasikan%20sebagai%20Master%20dan%20Slave.

Sinyal dasar yang lain dalam I^2C Bus adalah sinyal *acknowledge* yang disimbolkan dengan ACK setelah *transfer* data oleh *master* berhasil diterima *slave*, *slave* akan menjawabnya dengan mengirim sinyal *acknowledge*, yaitu dengan membuat SDA menjadi "0" selama siklus *clock* ke 9. Ini Menunjukkan bahwa *Slave* telah menerima 8 bit data dari *Master*.

Berikut merupakan kondisi sinyal *acknowledge*:

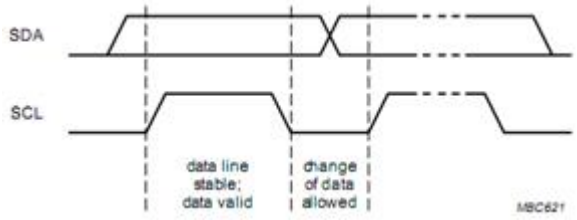


Gambar 0.16 Sinyal ACK dan NACK

Sumber: https://purnomosejati.wordpress.com/2011/08/25/mengenal-komunikasi-i2cinter-integrated-circuit/#:~:text=Inter%20Integrated%20Cir878cuit%20atau%20sering,untuk%20mengirim%20maupun%20menerima%20data.&text=Piranti%20yang%20dihubungkan%20dengan%20sistem,dioperasikan%20sebagai%20Master%20dan%20Slave.

Dalam melakukan *transfer* data pada I^2C Bus, kita harus mengikuti tata cara yang telah ditetapkan yaitu:

1. *Transfer* data hanya dapat dilakukan ketika Bus tidak dalam keadaan sibuk.
2. Selama proses *transfer* data, keadaan data pada SDA harus stabil selama SCL dalam keadan tinggi. Keadaan perubahan "1" atau "0" pada SDA hanya dapat dilakukan selama SCL dalam keadaan rendah. Jika terjadi perubahan keadaan SDA pada saat SCL dalam keadaan tinggi, maka perubahan itu dianggap sebagai sinyal *Start* atau sinyal *Stop*.



Gambar 0.17 Transfer bit pada I2C bus

Sumber: https://purnomosejati.wordpress.com/2011/08/25/mengenal-komunikasi-i2cinter-integrated-circuit/#:~:text=Inter%20Integrated%20Circuit%20atau%20sering,untuk%20mengirim%20maupun%20menerima%20data.&text=Piranti%20yang%20dihubungkan%20dengan%20sistem,dioperasikan%20sebagai%20Master%20dan%20Slave.

1.2.7 QR Code (Quick Response Code)

QRCode(Quick Response Code) adalah *barcode* dua dimensi yang dapat menyimpan data. *QRCode* dikembangkan oleh Denso Corporation, Jepang dan dapat digunakan secara gratis, bahkan untuk keperluan komersial.

Gambar 0.18 QRCode

Sumber: Penulis (2021)

QRCode dapat dibaca menggunakan berbagai *software* gratis yang tersedia pada berbagai *platform*. Besaran data yang dapat disimpan bervariasi, tergantung pada versi *QRCode*, ukuran *QRCode* dan tingkat *Error Correction Capability*-nya.

Tabel 0.10 Variasi Besaran Data QRCode

Versi *QRCode*	**Modul**	**Tingkat *Error Correction Capability***	**Jumlah Data (bits)**
1	21x21	L	152
		M	128

		Q	104
		H	72
2	25x25	L	272
		M	224
		Q	176
		H	128
3	29x29	L	440
		M	352
		Q	272
		H	208

Sumber: Penulis 2021

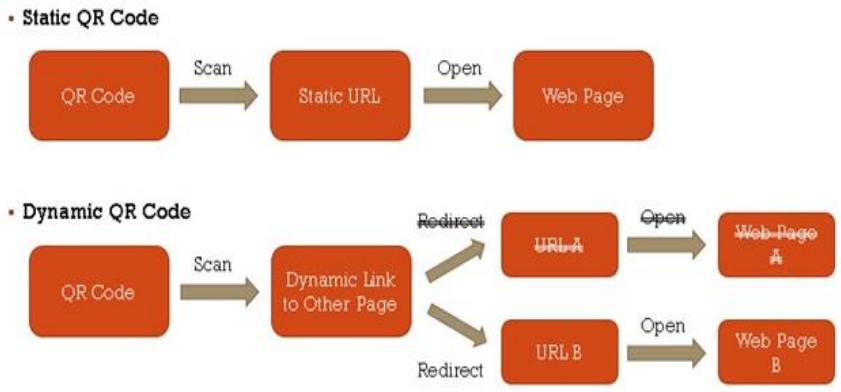
Error Correction Capability (ECC) Menunjukkan batasan *QRCode* masih dapat terbaca ketika terjadi kerusakan pada *QRCode* tersebut.*QRCode* dapat rusak ketika dicetak dan diletakkan pada kemasan produk.

Keuntungan *QRCode* antara lain:

1. Gratis dalam pembuatan dan penggunaannya
2. Tersedia *QRCode* scanner gratis
3. Menghemat kertas
4. Ukuran kecil
5. Tidak perlu membeli perangkat khusus scan *QRCode*
6. Sistem dapat cepat memberikan respon terkait hasil scan

Sedangkan kekurangan *QRCode* antara lain:

1. *QRCode* hanya mudah diakses oleh pengguna *smartphone.*
2. Pengguna *smartphone* harus *download* aplikasi *QRCode Scanner* terlebih dahulu.



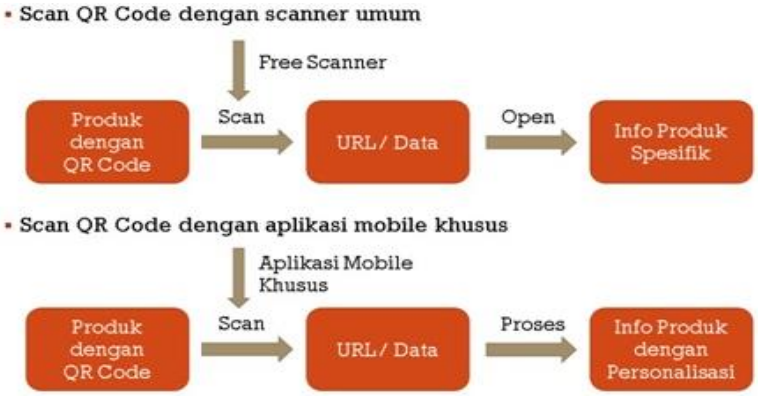
Gambar 0.19 Static and Dynamic QRCode

Sumber: https://socs.binus.ac.id/2018/12/15/pengenalan-qr-code/

Gambar 2.19 mengilustrasikan *static* dan *dynamic QRCode*. *Static QRCode* adalah *QRCode* yang berisi tautan kesebuah halaman *web* yang tetap. Penggunaan *static QRCode* menyebabkan konten *QRCode* tidak dapat diubah.

Dynamic QRCode adalah *QRCode* berisi sebuah URL singkat yang kemudian dialihkan ke halaman *web* yang lain. Penggunaan *dynamic QRCode* menyebabkan *QRCode* dapat digunakan ulang terus menerus.

Berikut merupakan bagan *QRCode* dengan aplikasi *mobile* khusus:



Gambar 0.20 Skema ScanQRCode

Sumber: https://socs.bin56us.ac.id/2018/12/15/pengenalan-qr-code/

QRCode dapat berisi konten yang umum, seperti URL ke sebuah halaman *web* atau unduhan file, *business card*, dan lain-lain. *QRCode* sejenis ini dapat di-*scan* dengan *QR-scanner* apapun. Namun demikian, terdapat *QRCode* yang dikhususkan untuk di-*scan* menggunakan aplikasi tertentu. Jika *QRCode* jenis ini di-scan menggunakan sembarang *QR-scanner*, akan menghasilkan pesan yang tidak dimengerti.

Berikut adalah contoh *QR-code* untuk https://web.whatsapp.com ketika di-*scan* menggunakan *QRscanner* pada umumnya:

Gambar 0.21 Encrypted code

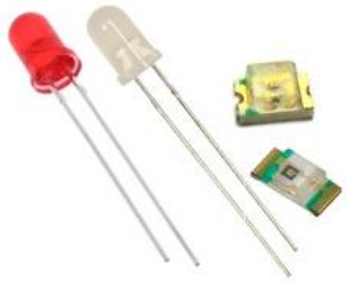
Sumber: https://socs.binus.ac.id/2018/12/15/pengenalan-qr-code/

Pesan tersebut tidak dapat dimengerti, karena adalah sebuah pesan *encrypted* dari aplikasi Whatsapp. Jika di-*scan* menggunakan menggunakan aplikasi Whatsapp, akan terjadi navigasi ke Whatsapp *web* suatu akun. Oleh karena itu, *QRCode* sejenis ini hanya dapat digunakan dengan melakukan *scan* dengan aplikasi tertentu.

1.2.8 LED (Light Emitting Diode)

Light Emitting Diode atau sering disingkat dengan LED adalah komponen elektronika yang dapat memancarkan cahaya monokromatik ketika diberikan tegangan maju. LED merupakan keluarga Dioda yang terbuat dari bahan semikonduktor. Warna-warna Cahaya yang dipancarkan oleh LED tergantung pada jenis bahan semikonduktor yang dipergunakannya. LED juga dapat memancarkan sinar inframerah yang tidak tampak oleh mata seperti yang sering kita jumpai pada *Remote Control* TV ataupun *Remote Control* perangkat elektronik lainnya.

Bentuk LED mirip dengan sebuah bohlam (bola lampu) yang kecil dan dapat dipasangkan dengan mudah ke dalam berbagai perangkat elektronika. Berbeda dengan Lampu Pijar, LED tidak memerlukan pembakaran filamen sehingga tidak menimbulkan panas dalam menghasilkan cahaya. Oleh karena itu, saat ini LED *(Light Emitting Diode)* yang bentuknya kecil telah banyak digunakan sebagai lampu penerang dalam LCD TV yang mengganti lampu tube.



Gambar 0.22 Bentuk LED

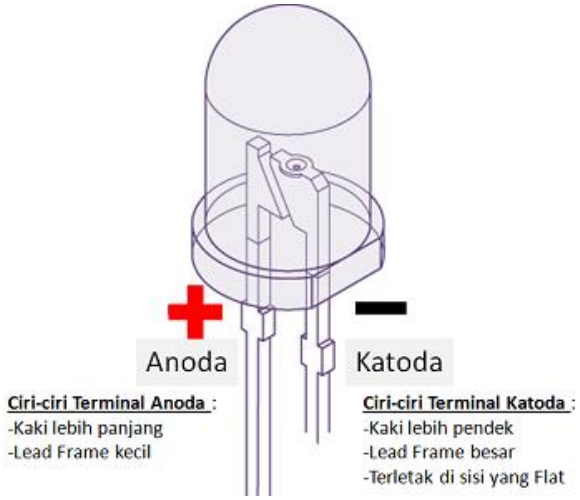
Sumber: https://teknikelektronika.com/pengertian-led-light-emitting-diode-cara-kerja/

Cara kerja LED hampir sama dengan Dioda yang memiliki dua kutub yaitu kutub Positif (P) dan Kutub Negatif (N). LED hanya akan memancarkan cahaya apabila dialiri tegangan maju *(bias forward)* dari Anoda Menuju ke Katoda.

LED terdiri dari sebuah *chip* semikonduktor yang di doping sehingga menciptakan *junction* P dan N. Yang dimaksud dengan proses doping dalam semikonduktor adalah proses untuk menambahkan ketidakmurnian *(impurity)* pada semikonduktor yang murni sehingga menghasilkan karakteristik kelistrikan yang diinginkan. Ketika LED dialiri tegangan maju atau *bias forward* yaitu dari Anoda (P) Menuju ke Katoda (K), Kelebihan Elektron pada N-Type material akan berpindah ke

wilayah yang kelebihan *Hole* (lubang) yaitu wilayah yang bermuatan positif (P-Type material). Saat Elektron berjumpa dengan *Hole* akan melepaskan photon dan memancarkan cahaya monokromatik (satu warna).

LED atau *Light Emitting Diode* yang memancarkan cahaya ketika dialiri tegangan maju ini juga dapat digolongkan sebagai *Transduser* yang dapat mengubah Energi Listrik menjadi Energi Cahaya.



Gambar 0.23 Polaritas LED

Sumber: https://teknikelektronika.com59/pengertian-led-light-emitting-diode-cara-kerja/

Untuk mengetahui polaritas terminal Anoda (+) dan Katoda (-) pada LED. Kita dapat melihatnya secara fisik berdasarkan gambar 2.28. Ciri-ciri Terminal Anoda pada LED adalah kaki yang lebih panjang dan juga *Lead Frame* yang lebih kecil. Sedangkan ciri-ciri Terminal Katoda adalah Kaki yang lebih pendek dengan *Lead Frame* yang besar serta terletak di sisi yang *flat*.

LED telah memiliki beranekaragam warna, diantaranya seperti warna merah, kuning, biru, putih, hijau, jingga dan infra merah. Keanekaragaman Warna pada LED tersebut tergantung pada *wavelength* (panjang gelombang) dan senyawa semikonduktor yang dipergunakannya.

Berikut ini adalah Tabel Senyawa Semikonduktor yang digunakan untuk menghasilkan variasi warna pada LED:

Tabel 0.11 Senyawa Semikonduktor LED

Bahan Semikonduktor	***Wavelength***	**Warna**
Gallium Arsenide (GaAs)	850-940nm	Infra Merah
Gallium Arsenide Phosphide (GaAsP)	630-660nm	Merah
Gallium Arsenide Phosphide (GaAsP)	605-620nm	Jingga
Gallium Arsenide Phosphide Nitride (GaAsP:N)	585-595nm	Kuning
Aluminium Gallium Phosphide (AIGaP)	550-570nm	Hijau
Silicon Carbide (SiC)	430-505nm	Biru
Gallium Indium Nitride (GaInN)	450nm	Putih

Sumber: Penulis 2021

Masing-masing Warna LED *(Light Emitting Diode)* memerlukan tegangan maju *(Forward Bias)* untuk dapat menyalakannya. Tegangan Maju untuk LED tersebut tergolong rendah sehingga memerlukan sebuah Resistor untuk membatasi Arus dan Tegangannya agar tidak merusak LED yang bersangkutan. Tegangan Maju biasanya dilambangkan dengan tanda V_F.

Tabel 0.12 Tegangan Maju LED

Warna	**Tegangan Maju @20mA**
Infra Merah	1,2V

Merah	1,8V
Jingga	2,0V
Kuning	2,2V
Hijau	3,5V
Biru	3,6V
Putih	4,0V

Sumber: Penulis 2021

1.2.9 DC Power Supply (Adaptor)

Catu Daya adalah bagian dari setiap perangkat elektronika yang berfungsi sebagai sumber tenaga. Catudaya sebagai sumber tenaga dapat berasal dari ; baterai , *accu* , *solar cell* dan adaptor. Komponen ini akan mencatu tegangan sesuai dengan tegangan yang diperlukan oleh rangkaian elektronika.

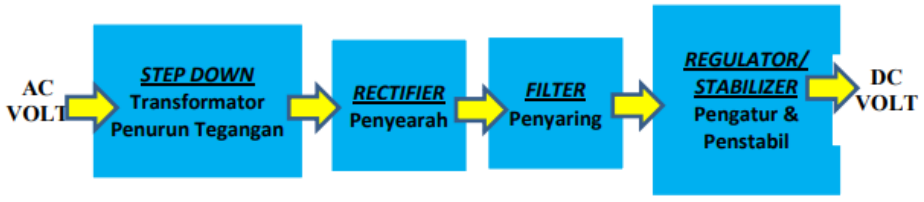
Prinsip Kerja DC *Power Supply* (Adaptor) adalah: Arus Listrik yang kita gunakan di ruma, kantor dan pabrik pada umumnya adalah dibangkitkan, dikirim dan didistribusikan ke tempat masing-masing dalam bentuk Arus Bolak-balik atau arus AC *(Alternating Current).*

Hal ini dikarenakan pembangkitan dan pendistribusian arus listrik melalui bentuk arus bolak-balik (AC) merupakan cara yang paling ekonomis dibandingkan dalam bentuk arus searah atau arus DC *(Direct Current).* Akan tetapi, peralatan elektronika yang kita gunakan sekarang ini sebagian besar membutuhkan arus DC dengan tegangan yang lebih rendah untuk pengoperasiannya. Oleh karena itu, hampir setiap peralatan Elektronika memiliki sebuah rangkaian yang berfungsi untuk melakukan konversi arus listrik dari arus AC menjadi arus DC dan juga untuk menyediakan tegangan yang sesuai dengan rangkaian Elektronika-nya. Rangkaian yang mengubah arus listrik AC menjadi DC ini disebut dengan DC *Power Supply* atau dalam bahasa Indonesia disebut dengan Catu daya DC. DC *Power Supply* atau Catu Daya ini juga sering dikenal dengan nama “Adaptor”.

Catu daya Adaptor adalah perangkat elektronika yang berfungsi Menurunkan dan mengubah tegangan AC *(Alternating Current)* menjadi tegangan DC *(Dirrect Current)* yang dapat di

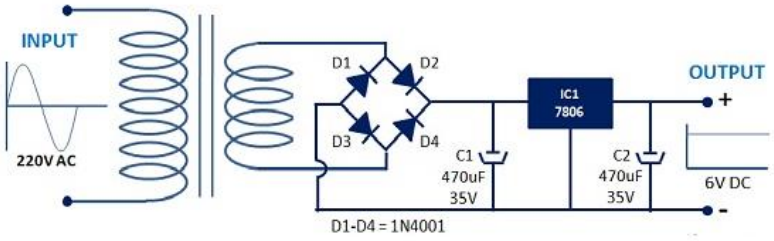
gunakan sebagai sumber tenaga peralatan elektronika. Sebuah catu daya adaptor yang baik memiliki bagian-bagian atau blok rangkaian. Sebuah DC *Power Supply* atau Adaptor pada dasarnya memiliki 4 bagian utama agar dapat menghasilkan arus DC yang stabil. Keempat bagian utama tersebut diantaranya adalah *Transformer* Penurun Tegangan, *Rectifier*, *Filter* dan *Voltage Regulator*. Sebelum kita membahas lebih lanjut mengenai Prinsip Kerja DC *Power Supply*, sebaiknya kita mengetahui Blok-blok dasar yang membentuk sebuah DC *Power Supply* atau Pencatu daya ini.

Berikut adalah Diagram Blok DC *Power Supply* (Adaptor) pada umumnya:



Gambar 0.24 Diagram Blok DC Power Supply

Sumber: Penulis (2021)



Gambar 0.25 Rangkaian Sederhana DC Power Supply

Sumber: https://teknikelektronika.com/prinsip-596kerja-dc-power-supply-adaptoir/

Keterangan:

1. *Stepdown* (Penurun Tegangan)

Bagian ini berfungsi Menurunkan tegangan AC 110/220V menjadi tegangan AC yang lebih rendah yang diperlukan (5V, 9V, 12V, dll). Bagian ini terdiri dari sebuah *transformer* (trafo).

2. *Rectifier* (Penyearah)

Bagian ini merupakan bagian penyearah arus dari arus AC (bolak-balik) menjadi arus DC (searah). Bagian ini terdiri dari sebuah dioda silikon , germanium , selenium yang disusun secara *bridge* dioda sebagai penyearah gelombang penuh *(full wave).*

3. *Filter* (Penyaring)

Bagian ini berfungsi untuk menyaring arus DC yang masih berdenyut sehingga menjadi rata. Komponen yang digunakan yaitu gabungan dari kapasitor elektrolit dengan resistor atau induktor.

4. *Stabilizer* (Penstabil)

Bagian ini berfungsi menstabilkan tegangan DC agar tidak terpengaruh oleh tegangan beban.Komponen ini berupa Dioda Zener atau IC yang didalamnya berisi rangkaian penstabil.

5. *Regulator* (Pengatur)

Bagian ini mengatur kestabilan arus yang mengalir ke rangkaian elektronika. Komponen yang di gunakan merupakan gabungan dari transistor, resistor dan kapasitor.

1.3 Teori-Teori Tentang Aplikasi Yang Dibahas

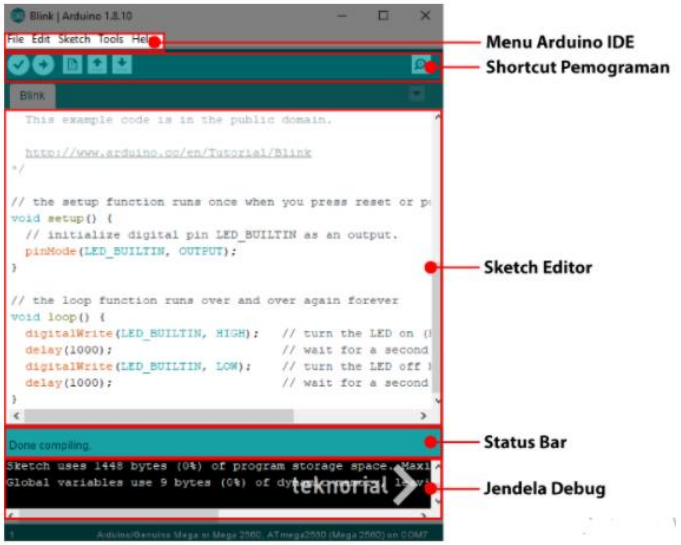
1.3.1 *Software* Arduino IDE

Arduino IDE *(Integrated Development Environment)*, atau secara bahasa mudahnya merupakan lingkungan terintegrasi yang digunakan untuk melakukan pengembangan. Disebut sebagai lingkungan karena melalui *software* inilah Arduino dilakukan pemrograman untuk melakukan fungsi-fungsi yang dibenamkan melalui sintaks pemrograman.

Arduino menggunakan bahasa pemrograman sendiri yang menyerupai bahasa C. Bahasa pemrograman Arduino (*Sketch*) sudah dilakukan perubahan untuk memudahkan pemula dalam melakukan pemrograman dari bahasa aslinya. Sebelum dijual ke pasaran, IC mikrokontroler Arduino telah ditanamkan suatu program bernama *Bootloader* yang berfungsi sebagai penengah antara *compiler* Arduino dengan mikrokontroler.

rduino IDE dibuat dari bahasa pemrograman JAVA. Arduino IDE juga dilengkapi dengan library C/C++ yang biasa disebut *Wiring* yang membuat operasi *input* dan *output* menjadi lebih mudah. Arduino IDE ini dikembangkan dari software *processing* yang dirombak menjadi Arduino IDE khusus untuk pemrograman dengan Arduino.

Berikut merupakan fungsi-fungsi *tools* dan Menu-Menu pada Arduino IDE:

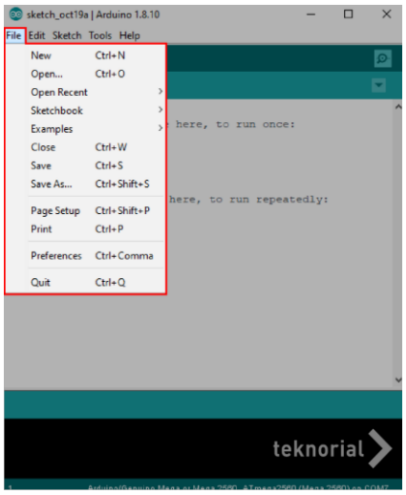


Gambar 0.26 Tampilan Arduino IDE

Sumber: https://teknorial.com/tutorial/pengenalan-Menu-tampilan-arduino-ide

2. *Menu* Arduino IDE

1. *File*:



Gambar 0.27 MenuFile Arduino IDE

*Sumber: https://teknorial.com/tutorial/pengenalan-*Menu*-tampilan-arduino-ide*

2) *New*: berfungsi untuk membuka lembar (*Sketch*) kerja baru.

3) *Open*: berfungsi untuk membuka *Sketch* yang pernah dibuat pada media penyimpanan.

4) *Open Recent*: berfungsi untuk mempersingkat waktu dalam membuka *Sketch*, *file* yang di buka merupakan *Sketch* terbaru atau yang terakhir tersimpan.

5) *Sketchbook:* berfungsi untuk menampilkan *Sketch* yang sudah tersimpan di dalam *folder* Arduino.

6) *Example:* berisi berbagai contoh kode program yang disediakan oleh pengembang.

7) *Close:* berfungsi Menutup jendela Arduino IDE sekaligus menghentikan Aplikasi.

8) *Save:* berfungsi menyimpan *Sketch* yang telah dibuat.

9) *Save as:* berfungsi menyimpan *Sketch* dengan nama yang berbeda.

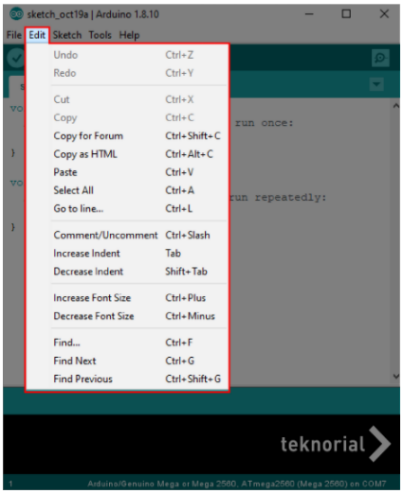
10) *Page Setup:* berfungsi mengatur tampilan *page* seperti *paper*, *orientation*, *margins* untuk pencetakan *Sketch*.

11) *Print:* berfungsi mengirimkan *file Sketch* ke mesin cetak untuk dicetak atau di buatkan menjadi *file* pdf.

12) *Preferences:* berfungsi merubah tampilan *interface* Arduino IDE.

13) *Quit:* berfungsi untuk Menutup semua jendela Arduino IDE. *Sketch* yang masih terbuka pada saat tombol *Quit* ditekan, secara otomatis akan terbuka pada saat Arduino IDE dijalankan.

b. *Edit:*



Gambar 0.28 MenuEdit Arduino IDE

*Sumber: https://teknorial.com/tutorial/pengenalan-*Menu*-tampilan-arduino-ide*

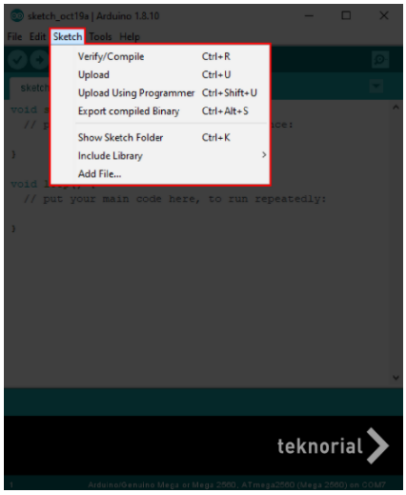
1) *Undo:* berfungsi untuk memundur satu *step* perubahan.
2) *Redo:* berfungsi untuk memajukan satu *step* perubahan.
3) *Cut:* berfungsi untuk menghapus kode program pada *editor* yang sudah terpilih sebelumnya untuk di pindahkan ke tempat lain, bila di inginkan menggunakan *paste.*
4) *Copy:* berfungsi menduplikasi kode program pada *sketch* yang sudah terpilih sebelumnya untuk di perbanyak ke tempat lain, bila di inginkan menggunakan *paste.*
5) *Copy for Forum:* berfungsi melakukan *copy* kode dari *editor* dan melakukan *formating* agar sesuai untuk ditampilkan dalam *forum*.
6) *Copy as HTML:* berfungsi menduplikasi teks yang terpilih kedalam *editor* dan menempatkan teks tersebut pada *clipboard* dalam bentuk atau *format* HTML.
7) *Paste:* berfungsi menyalin data yang terdapat pada *clipboard*, kedalam *Sketch.*
8) *Select All:* berfungsi untk memilih semua teks atau kode dalam halaman *Sketch*.
9) *Comment/Uncomment:* berfungsi untuk mengubah teks atau kode yang sudah terpilih menjadi berstatus “komen” yang di tandakan muncul tanda //.
10) *Increase Indent:* berfungsi untuk menambahkan Indentasi pada baris tertentu.
11) *Decrease Indent:* berfungsi untuk mengurangi Indentasi pada baris tertentu.

12) *Find:* berfungsi untuk memanggil jendela window "*find and replace*", di sini kamu dapat mencari teks atau kode di halaman *Sketch*, bisa juga sekalian untuk mengganti teks atau kode sebelumnya dengan kode baru.

13) *Find Next:* berfungsi menemukan kata setelahnya dari kata pertama yang berhasil ditemukan.

14) *Find Previous:* berfungsi menemukan kata sebelumnya dari kata pertama yang berhasil ditemukan.

c. *Sketch:*



Gambar 0.29 MenuSketch Arduino IDE

Sumber: https://teknorial.com/tutorial/pengenalan-Menu-tampilan-arduino-ide

1) *Verify/Compile:* berfungsi untuk mengecek *Sketch* yang kamu buat apa ada kekeliruan dalam segi bahasa programnya *(error)*. jika tidak ada, program yang kamu buat akan di *compile*.

2) *Upload:* berfungsi mengirimkan program yang sudah *compile* ke Arduino *Board.*

3) *Upload Using Programmer:* berfungsi untuk Menuliskan *bootloader* kedalam IC Mikrokontroler Arduino. Pada kasus ini kamu membutuhkan perangkat tambahan seperti USBasp untuk menjembatani penulisan program *bootloader* ke IC Mikrokontroler.

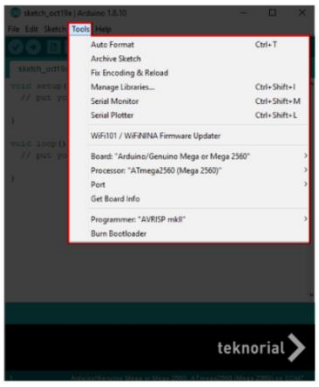
4) *Export Compiled Binary:* berfungsi untuk menyimpan *file* dengan ekstensi .hex, yang nantinya *file* ini dapat di *upload* ke *board* lain dengan aplikasi yang berbeda.

5) *Show Sketch Folder:* berfungsi membuka lokasi *folderSketch* yang saat ini sedang di dikerjakan.

6) *Include Library:* berfungsi untuk menambahkan *library*/pustaka yang sudah di sediakan pengembang ke dalam *Sketch* yang sedang di kerjakan.

7) *Add File:* berfungsi untuk menambahkan *file* kedalam *Sketch* Arduino. *File* akan muncul sebagai tab baru dalam jendela *Sketch*.

Tools



Gambar 0.30 MenuTools Arduino IDE

*Sumber: https://teknorial.com/tutorial/pengenalan-*Menu*-tampilan-arduino-ide*

1. *Auto Format:* berfungsi melakukan pengaturan *format* kode pada jendela *Sketch*.

2. *Archive Sketch:* berfungsi menyimpan *Sketch* kedalam *file* .zip .

3. *Fix Encoding & Reload:* berfungsi memperbaiki kemungkinan perbedaan antara pengkodean karakter *Sketch* dan peta karakter sistem operasi yang lain.

4.Manage Libraries: berfungsi untuk meng*installlibrary* tambahan dari Arduino *team*, atau pengembang pihak ke-tiga yang sudah mendaftarkan *library*nya di Arduino.

5.Serial Monitor: berfungsi membuka jendela *serial monitor* untuk melihat pertukaran data *interface* komunikasi *serial* dari program yang telah di buat.

6.Serial plotter: berfungsi untuk menanampikan gelombang sinus.

7.WiFi101/WiFiNINA *Firmware Updater:* berfungsi untuk meng*update* WiFi101/ WiFiNINA.

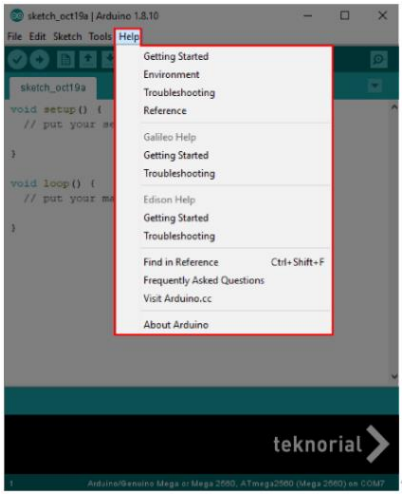
8,Board: berfungsi memilih dan melakukan konfigurasi *board* yang ingin digunakan.

9. *Port:* berfungsi untuk menyesuaikan *port* sebagai jalur komunikasi antara *software* dengan *hardware*.

*Programmer:*Menu ini digunakan ketika kamu hendak melakukan pemrograman *chip* mikrokontroler tanpa menggunakan koneksi *Onboard* USB-*Serial*. Biasanya digunakan pada proses *burning bootloader*.

Burn Bootloader: mengizinkan kamu untuk mengkopikan program *bootloader* kedalam IC mikrokontroler.

Help



Gambar 0.31 MenuTools Arduino IDE

Sumber: https://teknorial.com/tutorial/pengenalan-Menu-tampilan-arduino-ide

Menu *help* berisikan *file-file* dokumentasi yang berkaitan dengan masalah yang sering muncul, serta penyelesaiannya. Walaupun kamu *offline*, kamu masih dapat mengaksesnya.

Berikut sub Menu *help*:

a. *Getting Starter:* menjelaskan cara memulai peng*install*an Arduino dan juga cara mengujinya sudah siap di gunakan atau tidak.

b. *Environment:* menjelaskan mengenai Menu-Menu dan *shortcut* yang ada di Arduino IDE.

c. *Troubleshooting:* menjelaskan cara memecahkan masalah yang sering muncul saat menggunakan Arduino IDE.

d. *Reference:* menjelaskan mengenai bahasa *Sketch* yang digunakan saat memulai Menulis di *text editor* seperti *structure*, *variables*, *functions*.

e. *Find in Reference:* menjelaskan tentang cara membuat *comments* yang baik dan tips penggunaannya untuk mempermudah Menulis program di *text editor*.

f. *Frequently asked questions:* Pihak pengembang akan membantu menjelaskan untuk menjawab beberapa pertanyaan yang sering muncul atas ketidak tahuan *user.*

g. *Visit* Arduino.cc: masuk kedalam halaman *website* Arduino.cc.

h. *About* Arduino: versi Arduino IDE.

Shortcut Pemrograman

Gambar 0.32 Shortcut Pemrograman Arduino IDE

*Sumber: https://teknorial.com/tutorial/pengenalan-*Menu*-tampilan-arduino-ide*

Tabel 0.13 Shortcut Pemrograman

Gambar	**Menu**	**Keterangan**
	*Verify*	Berfungsi untuk menge*check* program yang sudah kita buat sudah sesuai dengan kaidah tata cara penulisan bahasa C++ *for* Arduino atau belum. Biasakan *verify* terlebih dahulu program yang sudah di buat sebelum di *upload.*

	*Upload*	Berfungsi untuk mengirimkan program yang sudah kita buat dari *software* Arduino IDE ke *memory board* Arduino.
	*New*	Berfungsi untuk Membuka lembar kerja baru atau *Sketch*.
	*Open*	Berfungsi untuk membuka lembar kerja atau *Sketch* yang sudah pernah di buat untuk melakukan pengeditan atau pun untuk *upload* ulang.
	*Save*	Berfungsi untuk menyimpan *Sketch* yang telah di buat ke dalam media penyimpanan kita.
	*Serial Monitor*	Berfungsi untuk melihat *data interface* komunikasi *serial* dari program yang telah kita buat, yang dikirimkan atau dipertukarkan antara Arduino dengan *Sketch* pada *port serial*nya. *Serial monitor* ini dapat digunakan untuk menampilkan nilai proses, nilai pembacaan, bahkan pesan *error* terdapat pada program yang telah kita buat.

Sumber: Penulis 2021

ii. Sketch Editor

Gambar 0.33 Sketch Editor Arduino IDE

Sumber: https://teknorial.com/tutorial/pengenalan-Menu-tampilan-arduino-ide

Sketch Editor merupakan tempat membangun logika pemrograman dengan menggunakan bahasa pemrograman yang digunakan adalah bahasa C/C++. Program pada Arduino terbagi menjadi tiga bagian utama yaitu *Structure*, *Values* (berisi variabel dan konstantata) dan yang terakhir *function.*

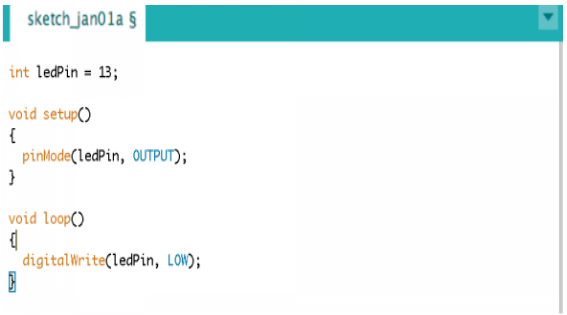
Berikut penjelasan tentang *Structure, Values,* dan *function:*

Structure: Struktur kode pada arduino yaitu berisi fungsi *setup()* dan *loop().*

1) *Setup():* fungsi ini dipanggil pertama kali ketika menjalankan *sketch*. digunakan sebagai tempat inisialisasi *variable*, *pin mode*, penggunaan *library* dan lainnya. fungsi ini dijalankan sekali ketika *board* dinyalakan atau di *reset*.

2) *Loop():* Setelah membuat fungsi *setup()* sebagai tempat inisialisasi *variable* dan menetapkan nilai maka selanjutnya fungsi *loop()* seperti namanya fungsi ini akan

melakukan perulangan berturut-turut, memungkinan program untuk mengubah dan menanggapi. digunakan untuk mengontrol *board* Arduino.



Gambar 0.34 Structure code Arduino

Sumber: http://allgoblog.com/apa-itu-arduino-ide-dan-arduino-sketch/#:~:text=Arduino%20IDE%20(Integrated%20Development%20Environment,di%20website%20resmi%20Arduino%20IDE.

b. *Values:* Berisi *variable* atau konstanta sesuai dengan *type* data yang didukung oleh Arduino.

c. *Function:* Segmentasi kode ke fungsi memungkinkan *programmer* untuk membuat potongan-potongan modular kodeyang melakukan tugas yang terdefinisi dan kemudian kembali ke asal kode dari mana fungsi itu “dipanggil”. Umumnya menggunakan fungsi adalah ketika salah satu kebutuhan untuk melakukan tindakan yang sama beberapa kali dalam sebuah program.

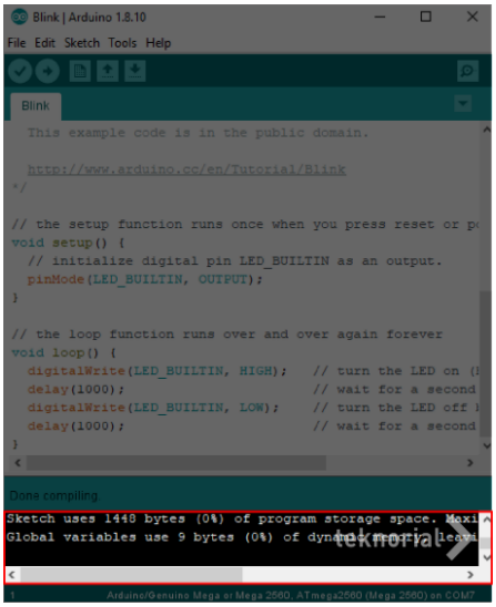
Status Bar

Gambar 0.35 Status Bar Arduino IDE

*Sumber: https://teknorial.com/tutorial/pengenalan-*Menu*-tampilan-arduino-ide*

Status bar merupakan jendela informasi, fungsinya untuk memberi notifikasi jika melakukan sesuatu di Arduino IDE. Misalnya saat melakukan *verify Sketch* untuk meng*compilingSketch*, dan meng*upload Sketch* ke *board* Arduino IDE. Sukses tidaknya kita akan mendapatkan notifikasi lanjutan sebagai acuan langkah selanjutnya.

Jendela *Debug*



Gambar 0.36 Jendela Debug Arduino IDE

Sumber: https://teknorial.com/tutorial/pengenalan-Menutampilan-arduino-ide

Jendela *debug* berfungsi untuk memunculkan kesalahan apa saja yang dilakukan *user* saat Menulis *Sketch* dan juga baris yang error dapat di ketahui disini. Dengan adanya jendela ini memudahkan *user* untuk mengedit dan memperbaiki *Sketch* yang sedang di kerjakan.

Bahasa Pemrograman Arduino IDE

Bahasa pemrograman yang digunakan oleh Arduino IDE didalam mengembangkan aplikasi mikrokontroler adalah C/C++.

Dengan beberapa perbedaan yaitu:

- *void main(void)* sebagai fungsi program utama diganti dengan *void loop()*. Perbedaannya pada c biasa tidak terjadi *loop*, jadi harus ada *looping* yang ditambahkan misalnya *while*(1){……}. Dalam arduino secara otomatis fungsi *loop*() akan kembali lagi dari awal jika sudah dieksekusi intruksi paling bawahnya.

- Penambahan fungsi *void setup(void)*, fungsi ini digunakan untuk inisialisasi mikrokontroler sebelum fungsi utama *loop*() dieksekusi.
- Tidak ada *settingregister-register*, karena arduino sudah memasukkannya kedalam *library* dan secara otomatis disesuaikan dengan jenis *board* arduino berkenaan dengan jenis mikrokontrolernya.

Struktur dasar Bahasa Pemrograman Arduino IDE:

a.

1) *#include*

Library File header / *header file* yaitu *file* yang berisi deklarasi fungsi dan definisi konstanta. Beberapa *fileheader* sudah disediakan di Arduino IDE sesuai *Library* yang sudah di *install*. File-file ini mempunyai ciri bereksistensi .h.File-file *header* ini biasanya dipanggil menggunakan fungsi *include*.

Bentuk umum dari #*include,* yaitu:

```
#include
void setup() {
// semua kode yang disini akan dibaca sekali oleh Arduino
}
void loop() {
//semua kode yang ada disini akan dibaca berulang kali (terus menerus) oleh Arduino
}
```

2) *void setup*

void setup di Arduino IDE merupakan isi kode berupa perintah untuk menentukan fungsi pada sebuah pin.

Contoh kodenya seperti:

pinMode(13, *OUTPUT*); // menentukan pin 13 sebagai *OUTPUT*

pinMode(3, *INPUT*); // menentukan pin 3 sebagai *INPUT*

untuk komunikasi antara Arduino dengan komputer, menggunakan:

Serial.begin(9600); // untuk komunikasi Arduino dengan komputer

3) *void loop*

void loop akan dibaca setelah void setup dan akan dibaca terus menerus oleh Arduino. Isinya berupa kode-kode perintah kepada pin *INPUT* dan *OUTPUT* pada Arduino.

Contoh kode :

digitalWrite(13, HIGH); //untuk memberikan 5V (nyala) kepada pin 13.

digitalWrite(13, LOW); //untuk memberikan 0V (mati) kepada pin 13.

analog*Write*(3, 225); //untuk memberikan nilai 225 (setara dengan 5V) kepada pin 3.

b.

1) Catatan Pada Program

Catatan program, dengan cara ketik // catatan,:

```
void loop() {
  // catatan pada baris ini tidak akan dibaca oleh program
}
```

Pemakaian tanda // hanya berfungsi untuk catatan satu baris, catatan yang panjang berupa paragraf. /* ketik catatan, dan jika selesai tutup dengan kode */.

Contoh:

```
void loop() {

 /* apapun yang kamu mau ketikan disini tidak
    akan dibaca oleh program
    sepanjang apapun kamu mengetiknya
*/
}
```

2) Kurung Kurawal {}

Digunakan untuk menentukan awal dan akhir dari program. Karena seperti bahasa pemrograman pada umumnya, Arduino membaca mulai dari atas hingga kebawah.

Contoh:

```
void loop()
{
....program
....program
....program
}
```

3) Titik Koma ;

Setiap baris kode pada Arduino harus diakhiri dengan tanda titik koma (;)

```
void setup(){
pinMode(13, OUTPUT);
}

void loop(){
digitalWrite(13, HIGH);
}
```

4) *Variable*

Variable adalah kode program yang digunakan untuk menyimpan suatu nilai pada sebuah nama.

Variable yang sering digunakan:

- *Int (interger)*

Variabel yang paling sering digunakan dan dapat menyimpan suatu nilai pada sebuah nama.

- *Long (long)*

Biasa digunakan jika nilai datanya lebih besar dari *integer*. Menggunakan 4 bytes(32 bits).

- *Boolean (boolean)*

Variabel yang hanya menyimpan nilai *TRUE* dan *FALSE* saja. Hanya menggunakan 1 bitsaja.

- *Float (float)*

Digunakan untuk *floating point* pada nilai decimal. *Memory* yang digunakan 4 bytes(32 bits)

- *Char (character)*

Menyimpan character berdasarkan ASCII kode (contoh: 'A'=65). Menggunakan 1 byte (8 bits).

5) Struktur Pengendali

Struktur Pengendali atau fungsi *if* digunakan untuk menentukan sebuah kondisi, dan jika kondisinya sudah terpenuhi maka akan melaksanakan perintah yang sudah ditentukan.

Contoh:

```
if(kondisi A)
{
  Kode Perintah A
}
elseif(kondisi B)
{
  Kode Perintah B
}
else
{
```

Kode Perintah C

}

c. Kode

1) Kode *Digital*

Digunakan untuk pemrograman yang menggunakan Pin *Digital*.

pinMode(pin, mode);

Kode ini digunakan untuk setting mode pin.

Contoh:

pinMode(13, *OUTPUT*); // artinya pin 13 digunakan sebagai *OUTPUT*

pinMode(7, *INPUT*); // artinya pin 7 digunakan sebagai *INPUT*

digitalRead(pin);

Kode ini digunakan pin *INPUT*, untuk membaca nilai sensor yang ada pada pin. Dan nilainya hanya terbatas pada 1 (*TRUE*), atau 0 (*FALSE*).

Contoh:

digitalRead(13); //kode membaca nilai sensor pada pin 13

Kode *digitalRead* masuk dalam void loop.

*digitalWrite(*pin, nilai);

Kode ini digunakan untuk pin *OUTPUT* yang sudah kita setting apakah akan diberikan *HIGH* (+5V), atau *LOW* (Ground).

Contoh:

digitalWrite(13, HIGH);

digitalWrite(13, LOW);

2) Kode Analog

analogWrite(pin, nilai);

Dengan pin analog nilai yang dihasilkan menjadi bervariasi dari 0-225, itu setara dengan 0-5V.

Contoh:

analogWrite(3, 150); *//* artinya pin 3 diberikan nilai sebesar 150

Kode *analogWrite* juga dimasukan dalam *void loop*.

Kode analog digunakan ketika ingin menggunakan pin Analog. Pin Analog hanya bisa kita gunakan sebagai *INPUT*.

analogRead(pin);

Kode diatas digunakan untuk membaca nilai pada sensor analog. Yaitu antara 0-1024.

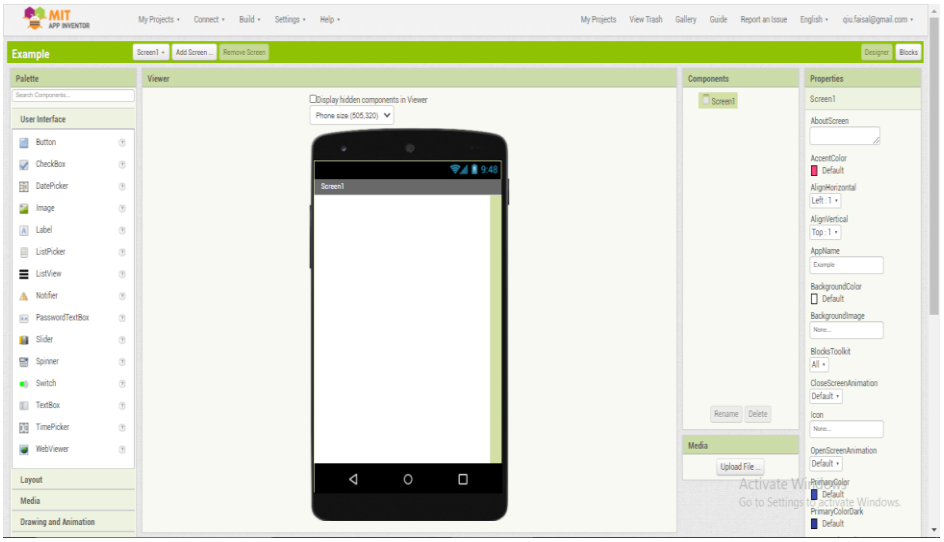
Contoh:

analogRead(A0); *//* artinya kode akan membaca nilai sensor pada pin AO.

1.3.2 MIT App Inventor

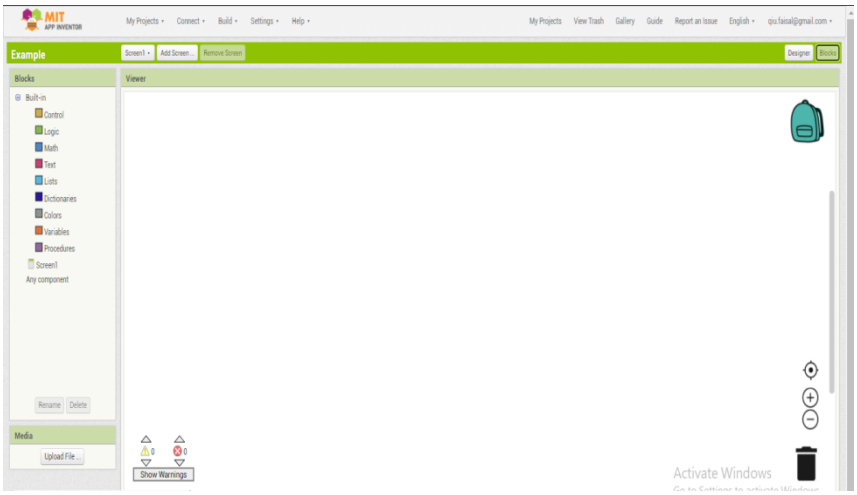
MIT *App Inventor* merupakan *platform* untuk memudahkan proses pembuatan aplikasi sederhana tanpa harus mempelajari atau menggunakan bahasa pemrograman yang terlalu banyak. Kita dapat mendesain aplikasi android sesuai keinginan dengan menggunakan berbagai macam layout dan komponen yang tersedia.

Pada MIT *App Inventor* terdapat dua halaman utama, yaitu halaman *designer* dan halaman *blocks*. Halaman *designer* digunakan untuk mendesain tampilan aplikasi dengan berbagai komponen dan *layout* yang disediakan sesuai dengan keinginan. Sedangkan halaman *blocks* digunakan untuk memprogram jalannya aplikasi android sesuai dengan tujuan.



Gambar 0.37 Tampilan Halaman Designer

Sumber: Penulis



Gambar 0.38 Halaman Tampilan Blocks

Sumber: Penulis (2021)

1. Halaman *Designer*

Pada halaman *designer* terdapat beberapa jendela seperti *Palette, Viewer, Components, Media,* dan *Properties.Tools* tersebut berfungsi untuk mendesain tampilan aplikasi android sesuai keinginan.

Palette: Jendela tempat mengambil komponen-komponen yang dikategorikan dalam beberapa kategori untuk dimasukkan dalam aplikasi yang dibuat. Terdapat kategori *User Interface, Layout, Media, Drawing and Animation, Maps, Sensors, Social, Storage, Connectivity,* LEGO MINDSTORMS, *Experimental,* dan *Extension.*

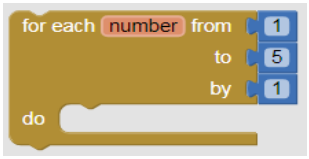
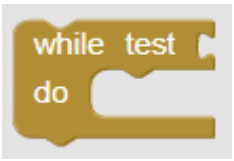
Viewer: Tempat untuk mengatur tampilan komponen pada aplikasi nantinya.

Components: Tempat untuk mengatur komponen-komponen yang telah diletakkan di *viewer,* seperti misalnya mengganti nama komponen, dan menghapus komponen.

Properties: Tempat untuk mengatur properti layar, dan komponen-komponen yang digunakan pada aplikasi yang sedang dibuat seperti lebar, tinggi, warna latar, besar huruf, dll.

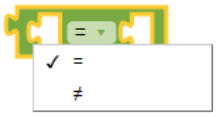
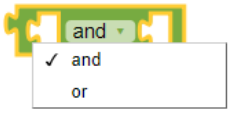
Media: Tempat untuk mengunggah gambar untuk digunakan pada aplikasi yang sedang dibuat.

Tabel 0.14 Komponen-Komponen Control

Blok Kode	**Fungsi**
	*If* kondisional. Jika "if" meMenuhi syarat, maka blok yang ada setelah "then" dieksekusi.
	*Looping* dari angka pertama hingga angka terakhir dengan suatu *interval*. Maka gambar disamping berarti *loop* dari angka 1 hingga 5 dengan *interval* 1: 1, 2, 3, 4, 5.
	*For* bertingkat. Untuk setiap objek dalam *list*, dilakukan *looping*.
	Jika nilai *test true*, maka *loop while* berjalan.

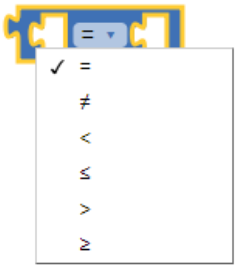
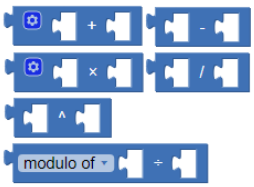
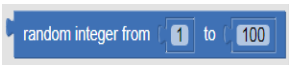
Sumber: Penulis 2021

Tabel 0.15 Komponen-Komponen Logic

Blok Kode	**Fungsi**
	Boolean *true/ false*.
	Jika dipasangkan dengan *true/ false* maka *true* menjadi *false*, dan *false* menjadi *true*.
	Memeriksa apakah satu objek sama dengan/tidak sama dengan objek di kanan. Jika sesuai kriteria, maka blok akan bernilai *true*, dan *false* jika tidak sesuai.
	Pada logika *and* jika kedua syarat terpenuhi, maka nilainya menjadi *true*. Sedangkan pada logika *or*, jika salah satu syarat atau keduanya terpenuhi, maka nilainya menjadi *true*.

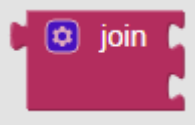
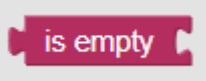
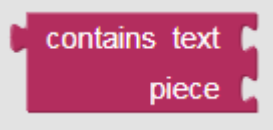
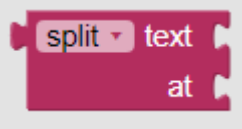
Sumber: Penulis 2021

Tabel 0.16 Komponen-Komponen Math

Blok Kode	**Fungsi**
	Digunakan untuk menginput angka.
	Digunakan untuk membandingkan dua angka. Perbandingan dapat berupa sama dengan, tidak sama dengan, lebih dari, kurang dari, lebih dari sama dengan, dan kurang dari sama dengan.
	Operasi matematika dasar, yaitu tambah, kurang, kali, bagi, pangkat, dan modulus.
	Mengambil nilai *integer* secara acak dari *range* yang ditentukan.
	Operasi trigonometri sin, cos, tan.

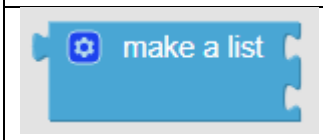
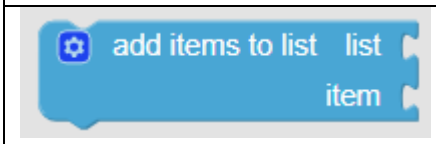
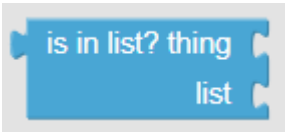
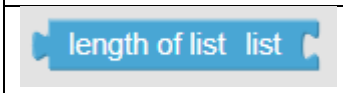
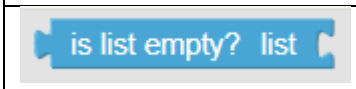
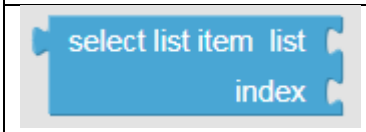
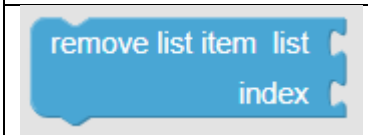
Sumber: Penulis 2021

Tabel 0.17 Komponen-Komponen Text

Blok Kode	**Fungsi**
	Teks kosong.
	Menggabungkan dua atau lebih teks.
	Memeriksa jika teks kosong atau tidak.
	Memeriksa apakah ada bagian tertentu dalam suatu teks.
	Memisahkan teks pada penanda tertentu dan membuatnya menjadi *list*.

Sumber: Penulis 2021

Tabel 0.18 Komponen-Komponen List

Blok Kode	**Fungsi**
	Membuat *list* kosong.
	Membuat *list*.
	Menambahkan elemen pada *list*.
	Memeriksa apakah ada elemen tertentu pada *list*.
	Memeriksa banyaknya elemen pada *list*.
	Memeriksa apakah *list* kosong atau tidak.
	Mengambil elemen dari suatu *list*.
	Menghapus elemen dari suatu *list*.

Sumber: Penulis 2021

Tabel 0.19 Komponen-Komponen Colors

Blok Kode	Fungsi
	Pilihan warna yang disediakan MIT AI.
	Membuat warna sendiri dengan menggunakan kode RGB warna.

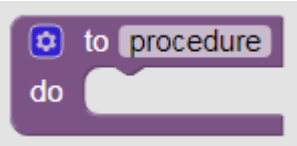
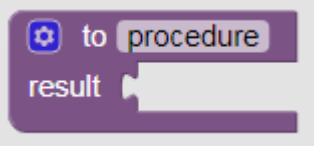
Sumber: Penulis 2021

Tabel 0.20 Komponen-Komponen Variables

Blok Kode	Fungsi
	Membuat suatu variabel global.
	Mengambil variabel global.
	Memodifikasi isi dari variabel global.

Sumber: Penulis 2021

Tabel 0.21 Komponen-Komponen Procedure

Blok Kode	Fungsi
	Membuat suatu prosedur.
	Membuat suatu fungsi yang mengembalikan hasil tertentu.

Sumber: Penulis 2021