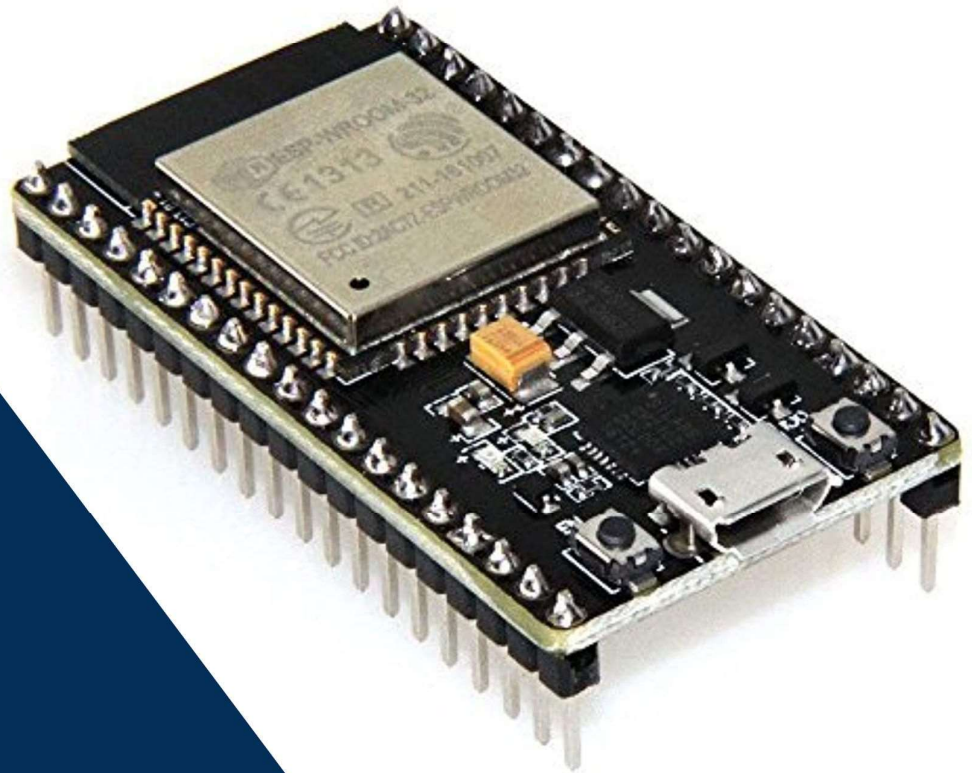




@stc.sultantech



# MANUAL BOOK

Panduan penggunaan sensor  
dan penjelasan program

## DAFTAR ISI

|  |    |
|--|----|
| DAFTAR ISI .....                               | i  |
| BAB I .....                                    | 1  |
| Penjelasan Komponen yang Digunakan .....       | 1  |
| A.    Mikrokontroler ESP32 .....               | 1  |
| B.    Sensor Ph-4502c.....                     | 4  |
| C.    Sensor Turbidity .....                   | 5  |
| D.    Sensor Suhu DS18b20.....                 | 6  |
| E.    Sensor Ultrasonik HC-SR04 .....          | 7  |
| F.    Servo SG90 .....                         | 9  |
| G.    Relay 2-Channel.....                     | 10 |
| H.    RTC DS1307 .....                         | 10 |
| BAB II.....                                    | 12 |
| Cara Menghubungkan ESP32 ke Blynk IoT 2.0..... | 12 |
| 1.    Website Blynk IoT 2.0 .....              | 12 |
| 2.    Upload Program ke ESP32 .....            | 16 |
| BAB III .....                                  | 19 |
| Penjelasan Kode Program .....                  | 19 |
| 1.    Konfigurasi Template pada Blynk .....    | 19 |
| 2.    Library.....                             | 20 |
| 3.    Definisi Pin yang Digunakan .....        | 21 |
| 4.    Inisialisasi Variabel dan Objek.....     | 21 |
| 5.    Deklarasi Variabel .....                 | 22 |
| 6.    Objek Fitur-Fitur Blynk .....            | 22 |
| 7.    Blynk Terkoneksi .....                   | 23 |
| 8.    Mengirim Data Ke Blynk.....              | 23 |
| 9.    Memberi Makan Otomatis .....             | 24 |
| 10.   Menguras air dan mengisi air.....        | 24 |
| 11.   Void setup() .....                       | 25 |
| 12.   Void Loop() .....                        | 27 |
| 13.   Sensor PH.....                           | 27 |
| 14.   Sensor Ultrasonik .....                  | 28 |
| 15.   Sensor Suhu DS18B20.....                 | 28 |

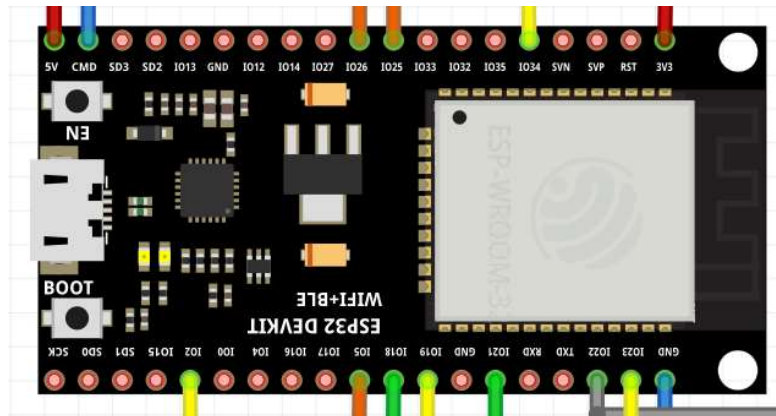
|   |    |
|---|----|
| 16. Sensor Turbidity .....  | 29 |
| 17. Menampilkan Pembacaan Sensor Pada Serial Monitor Arduino IDE..... | 30 |
| 18. RTC .....   | 30 |
| 19. Menambahkan angka 0.....  | 31 |
| BAB IV .....  | 32 |
| Cara Menangani Error dan Kendala pada Sistem .....                    | 32 |
| 1. Error Saat Upload Kode Program ke ESP32 .....                      | 32 |
| 2. Gagal dalam Menghubungkan ESP32 ke Blynk.....                      | 33 |
| 3. Kesalahan Sensor saat Pengukuran.....                              | 34 |

## BAB I

### Penjelasan Komponen yang Digunakan

#### A. Mikrokontroler ESP32

Mikrokontroler ESP32 adalah salah satu platform pengembangan yang populer untuk proyek Internet of Things (IoT). ESP32 dikembangkan oleh perusahaan Tiongkok, Espressif Systems, dan merupakan penerus dari mikrokontroler ESP8266 yang juga sangat populer. Pada proyek ini digunakan ESP32 Devkit yang banyak tersedia dipasar.



Gambar Mikrokontroler ESP32

Spesifikasi yang terdapat pada ESP32 antara lain:

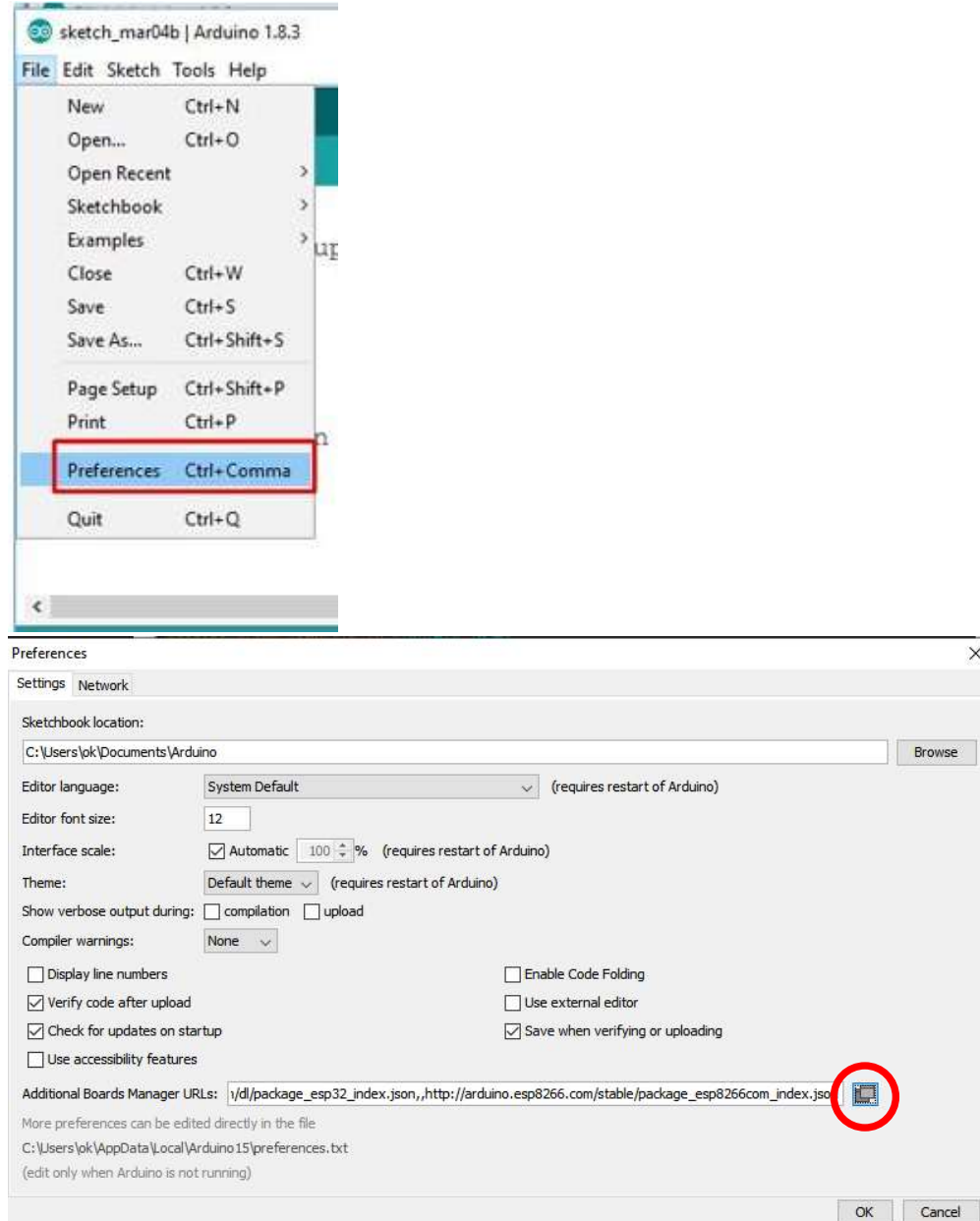
- Microprosesor Xtensa Dual-Core 32 Bit LX6
- Freq Clock up to 240 MHz
- SRAM 520 kB
- Flash memori 4 MB
- 11b/g/n WiFi transceiver
- Bluetooth 4.2/BLE
- 48 pin GPIO
- 15 pin channel ADC (Analog to Digital Converter)
- 25 pin PWM (Pulse Width Modulation)
- 2 pin channel DAC (Digital to Analog Converter)

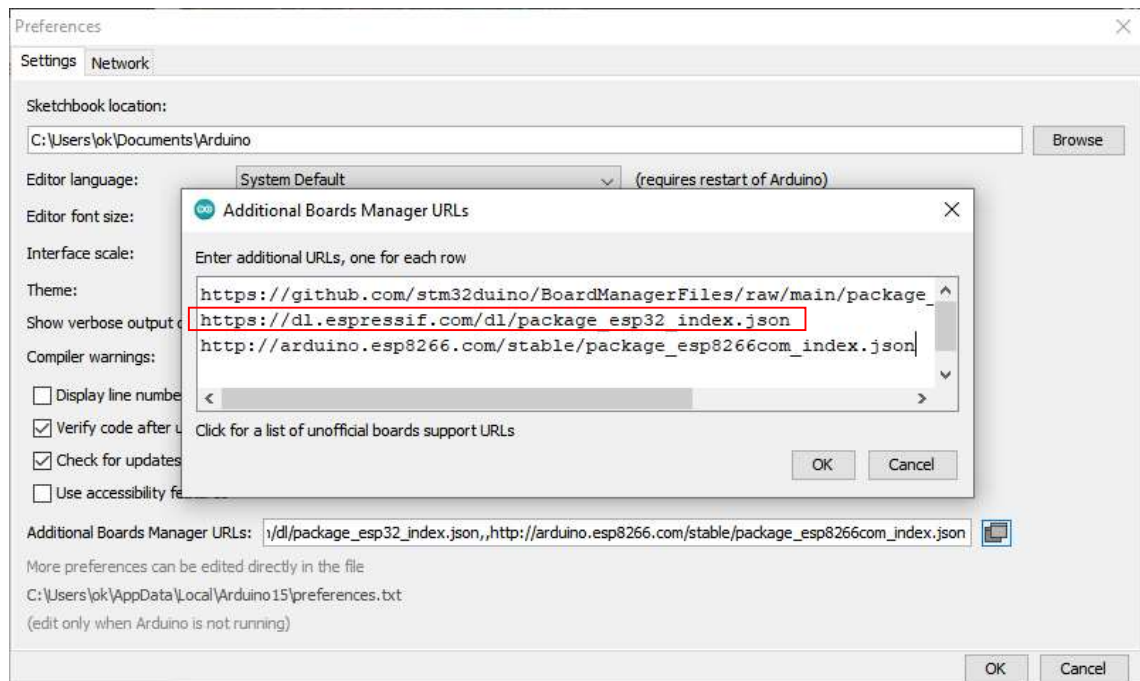
Untuk fitur pada ESP32 Devkit antara lain:

- Jumlah pin : 30 meliputi pin tegangan dan GPIO.
- 15 pin ADC (Analog to Digital Converter)
- 3 UART Interface
- 3 SPI Interface
- 2 I2C Interface
- 16 pin PWM (Pulse Width Modulation)
- 2 pin DAC (Digital to Analog Converter)

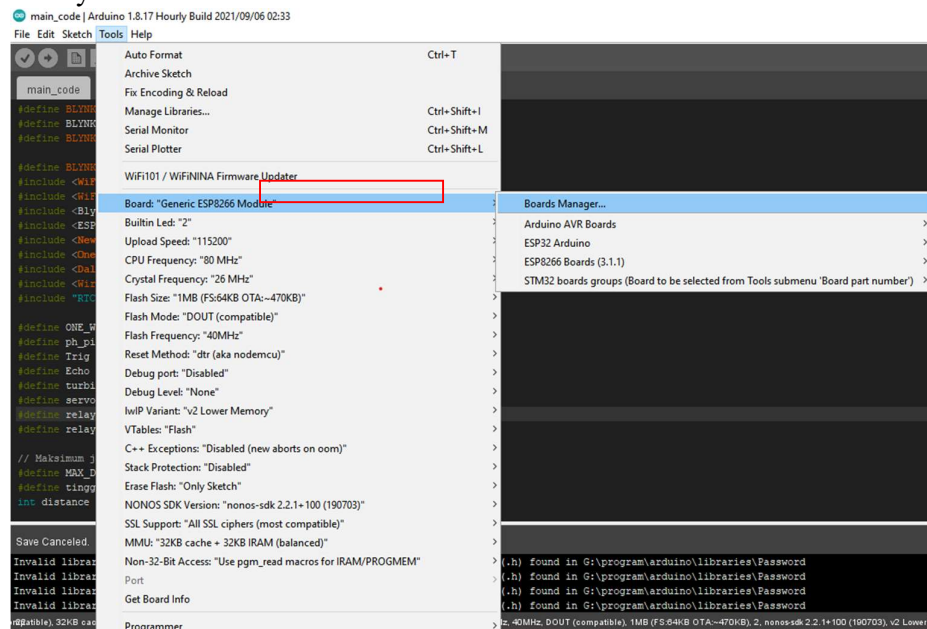
Untuk memprogram ESP32 ini pada arduino IDE, ikuti langkah-langkah berikut:

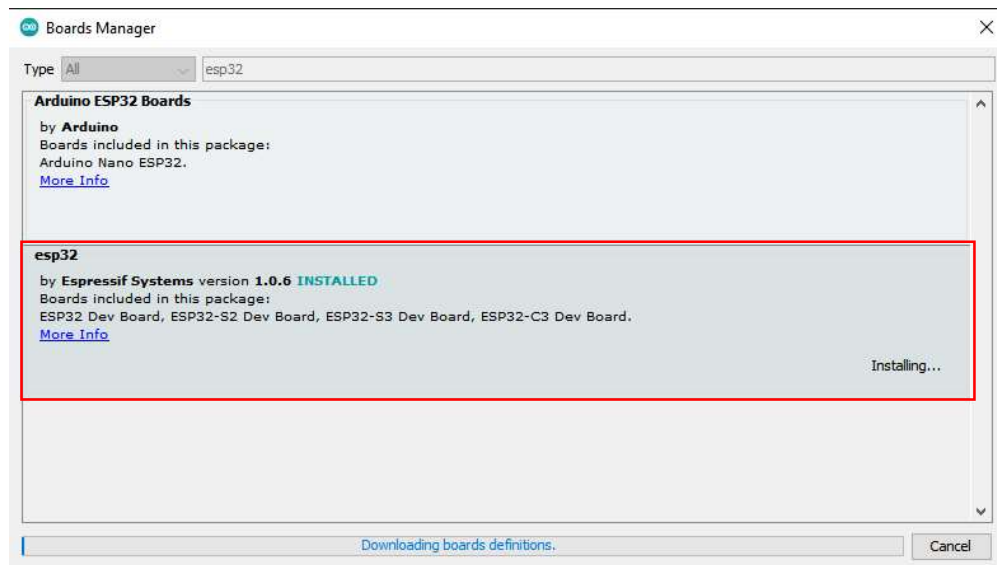
1. Instalasi Arduino IDE: Pastikan Anda telah menginstal Arduino IDE di komputer Anda. Jika belum, Anda dapat mengunduhnya dari situs resmi Arduino: <https://www.arduino.cc/en/software>.
2. Menambahkan Board ESP32 ke Arduino IDE: Buka Arduino IDE dan pergi ke "File" > "Preferences". Di bagian "Additional Boards Manager URLs", tambahkan URL berikut: [https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json). Klik "OK" untuk menyimpan.





3. Instalasi Board ESP32: Setelah menambahkan URL di atas, pergi ke "Tools" > "Board" > "Boards Manager...". Cari "esp32" dan instal "esp32" package yang ditawarkan oleh Espressif Systems.





## B. Sensor Ph-4502c

Sensor pH-4502C adalah sensor pH yang sering digunakan dalam proyek-proyek pengukuran pH. Sensor ini berfungsi untuk mengukur konsentrasi ion hidrogen (pH) dalam suatu larutan atau medium. pH merupakan skala yang digunakan untuk menentukan apakah suatu larutan bersifat asam, basa, atau netral.



Sensor PH dan Module Ph-4502C

**Prinsip Kerja:** Sensor pH-4502C bekerja berdasarkan prinsip elektrokimia. Di dalam sensor terdapat elektroda khusus yang terbuat dari bahan yang peka terhadap perubahan pH. Ketika sensor dicelupkan ke dalam larutan, perubahan pH menyebabkan perubahan potensial listrik di antara elektroda, yang dapat diukur dan diubah menjadi nilai pH. Untuk lebih jelasnya anda dapat kunjungi link berikut: <https://www.youtube.com/watch?v=nsqEcekJ8-E&t=445s>

**Komponen Sensor:** Sensor pH-4502C biasanya terdiri dari dua atau tiga elektroda. Elektroda referensi digunakan sebagai referensi potensial, elektroda penginderaan pH yang sensitif terhadap perubahan pH, dan terkadang elektroda pembanding (counter electrode) untuk menghindari efek polarisasi.

**Rentang Pengukuran:** Sensor pH-4502C memiliki rentang pengukuran pH tertentu, biasanya antara 0 hingga 14 pH. Di sisi atas (pH tinggi), sensor cenderung kurang akurat karena ketersediaan ion hidrogen yang rendah di larutan alkali. Namun, biasanya rentang antara 4 hingga 10 pH lebih akurat dan umum digunakan.

**Kalibrasi:** Untuk mendapatkan hasil pengukuran yang akurat, sensor pH-4502C perlu dikalibrasi secara berkala menggunakan larutan kalibrasi pH yang diketahui nilainya. Biasanya dua titik kalibrasi digunakan, yaitu dengan larutan pH 7 (netral) dan pH 4 atau pH 10 (asam atau basa).

**Aplikasi:** Sensor pH-4502C sering digunakan dalam berbagai bidang seperti akuarium, hidroponik, pertanian, pemantauan air, laboratorium kimia, dan dalam berbagai proyek IoT dan alat pengukuran otomatis.

**Kelebihan:** Sensor pH-4502C memiliki beberapa kelebihan, seperti ukurannya yang kecil, konsumsi daya yang rendah, dan harga yang terjangkau. Hal ini membuatnya menjadi pilihan yang populer untuk berbagai proyek.

**Perawatan:** Sensor pH-4502C memerlukan perawatan yang baik agar tetap akurat dan tahan lama. Setelah digunakan, sensor perlu dibilas dengan air murni untuk menghindari kontaminasi. Selain itu, elektroda referensi mungkin perlu diisi kembali agar tetap berfungsi dengan baik.

Penting untuk diingat bahwa pengukuran pH yang akurat sangat penting dalam banyak aplikasi, terutama di bidang ilmiah dan industri. Penggunaan sensor pH-4502C memerlukan pemahaman yang baik tentang prinsip kerjanya dan perawatan yang tepat agar memberikan hasil pengukuran yang akurat dan konsisten.

### C. Sensor Turbidity

Sensor turbidity (kekeruhan) adalah sebuah sensor yang digunakan untuk mengukur tingkat kekeruhan atau tingkat ketidakjernihan dalam suatu cairan. Tingkat kekeruhan mencerminkan sejauh mana partikel-partikel padat tersuspensi atau koloid dalam larutan, dan hal ini dapat memberikan informasi tentang kualitas air atau kualitas cairan lainnya.



Gambar Sensor Turbidity serta modul



Prinsip Kerja: Sensor turbidity bekerja berdasarkan prinsip hamburan cahaya. Cahaya diarahkan melalui cairan yang akan diukur ke fotodetektor di sisi lain. Jika cairan memiliki sedikit partikel, maka cahaya akan mudah melewati dan sensor akan mendeteksi intensitas cahaya yang tinggi. Namun, jika cairan memiliki banyak partikel, cahaya akan tersebar atau diserap oleh partikel-padatan, sehingga intensitas cahaya yang sampai ke fotodetektor akan lebih rendah. Untuk lebih jelasnya dapat mengikuti link berikut: <https://www.youtube.com/watch?v=Bu8y-6Bpv0U>

Sensor turbidity sering digunakan dalam berbagai aplikasi, seperti Pemantauan Kualitas Air, Industri Makanan dan Minuman, Industri Farmasi dan Sistem Pengolahan Air.

Sensor turbidity memiliki berbagai ukuran dan tingkat sensitivitas tergantung pada aplikasinya. Penting untuk mengkalibrasi sensor secara berkala untuk memastikan keakuratan pengukuran. Sensor ini juga memerlukan perawatan yang tepat untuk menghindari kontaminasi atau kerusakan pada permukaan optik.

#### **D. Sensor Suhu DS18B20**

Sensor suhu DS18B20 adalah sensor suhu digital yang sangat populer dan sering digunakan dalam proyek-proyek DIY dan aplikasi industri. Sensor ini memiliki kemampuan unik karena menggunakan antarmuka One-Wire (1-Wire), yang memungkinkan banyak sensor terhubung ke satu saluran digital menggunakan satu kabel saja.



Gambar Sensor Suhu DS18B20

Berikut adalah beberapa poin penting tentang sensor suhu DS18B20:

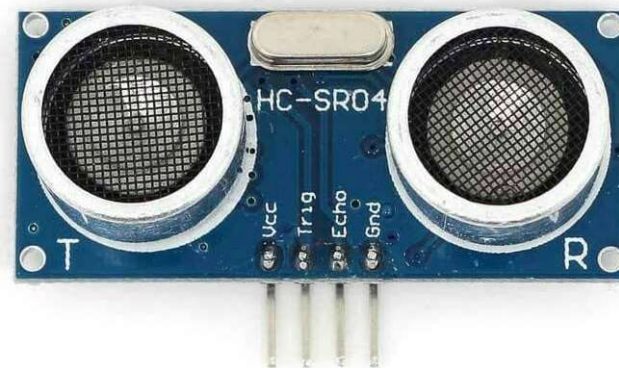
- **Prinsip Kerja:** Sensor suhu DS18B20 bekerja berdasarkan prinsip perubahan resistansi termal pada elemen suhu. Ketika suhu berubah, resistansi pada elemen suhu juga berubah, dan ini akan diubah menjadi nilai digital yang sesuai dengan suhu yang diukur.
- **Antarmuka One-Wire:** Keunggulan utama dari DS18B20 adalah penggunaan antarmuka One-Wire. Ini berarti sensor dapat dihubungkan ke mikrokontroler menggunakan satu kabel saja, membuatnya sangat nyaman untuk digunakan dalam proyek dengan keterbatasan koneksi GPIO.
- **Resolusi Suhu:** Sensor DS18B20 dapat diatur dalam beberapa tingkat resolusi suhu, mulai dari 9 hingga 12 bit. Resolusi yang lebih tinggi akan memberikan pembacaan yang lebih akurat namun memerlukan waktu konversi yang lebih lama.

- **Kisaran Pengukuran:** DS18B20 memiliki kisaran pengukuran suhu yang luas, yaitu sekitar  $-55^{\circ}\text{C}$  hingga  $+125^{\circ}\text{C}$ . Rentang yang lebar ini memungkinkan sensor digunakan dalam berbagai aplikasi dengan suhu ekstrem.
- **Presisi:** DS18B20 adalah sensor suhu digital dengan presisi yang tinggi. Tergantung pada resolusi yang diatur, akurasi tipikalnya berkisar antara  $\pm 0,5^{\circ}\text{C}$  hingga  $\pm 0,0625^{\circ}\text{C}$ .
- **Penggunaan Catu Daya Parasit:** Sensor DS18B20 dapat dioperasikan dengan catu daya parasit melalui kabel data. Ini berarti sensor dapat dihidupkan dengan mengambil daya dari kabel data, sehingga mengurangi jumlah kabel yang diperlukan untuk menghubungkannya ke mikrokontroler.
- **Aplikasi:** Sensor suhu DS18B20 banyak digunakan dalam berbagai aplikasi, termasuk pemantauan suhu dalam peralatan elektronik, kendali iklim, sistem HVAC (Heating, Ventilating, and Air Conditioning), kendaraan, dan proyek-proyek IoT.
- **Library Pendukung:** Ada banyak library pendukung yang tersedia untuk sensor suhu DS18B20 di berbagai platform mikrokontroler, termasuk Arduino. Library ini memudahkan penggunaan dan pengintegrasian sensor ke dalam proyek Anda.

Sensor suhu DS18B20 adalah pilihan yang populer untuk proyek-proyek yang membutuhkan pengukuran suhu digital dengan presisi tinggi dan antarmuka komunikasi sederhana. Penggunaan antarmuka One-Wire membuatnya sangat nyaman untuk digunakan dalam proyek dengan keterbatasan koneksi dan banyak sensor dapat dihubungkan secara bersamaan melalui satu kabel, membuatnya cocok untuk aplikasi di mana banyak titik pengukuran suhu diperlukan.

#### E. Sensor Ultrasonik HC-SR04

Sensor ultrasonik HC-SR04 adalah salah satu sensor yang paling populer dan sering digunakan dalam proyek-proyek DIY dan aplikasi robotika. Sensor ini digunakan untuk mengukur jarak antara sensor dan benda di depannya dengan memanfaatkan gelombang ultrasonik.

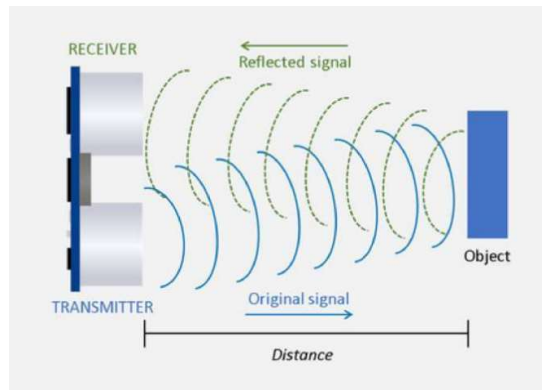


Gambar Sensor Ultrasonik HC-SR04

Berikut adalah beberapa poin penting tentang sensor ultrasonik HC-SR04:

- **Prinsip Kerja:** Sensor ultrasonik HC-SR04 bekerja berdasarkan prinsip pantulan gelombang ultrasonik. Sensor menghasilkan gelombang ultrasonik melalui transduser ultrasonik yang ada di bagian depan sensor. Gelombang ini akan mengalami pantulan saat

mengenai benda di depannya, dan waktu yang diperlukan untuk gelombang kembali ke sensor diukur.



- **Antarmuka:** Sensor ultrasonik HC-SR04 memiliki dua kaki yang digunakan untuk mengirimkan dan menerima gelombang ultrasonik. Satu kaki digunakan untuk mengirimkan gelombang, dan kaki lainnya digunakan untuk menerima gelombang yang dipantulkan. Sensor juga memerlukan dua kaki tambahan untuk daya (Vcc) dan ground (GND).
- **Pengukuran Jarak:** Waktu yang diperlukan untuk gelombang kembali ke sensor digunakan untuk menghitung jarak antara sensor dan benda. Karena gelombang ultrasonik memiliki kecepatan tertentu dalam udara, kita dapat menggunakan waktu perjalanan gelombang (T) untuk menghitung jarak (D) dengan rumus sederhana:  $D = (T \times V) / 2$ , di mana V adalah kecepatan gelombang ultrasonik dalam medium yang diukur (biasanya udara).
- **Rentang Pengukuran:** Rentang pengukuran sensor ultrasonik HC-SR04 tergantung pada kecepatan gelombang ultrasonik dan batas waktu yang dapat diukur oleh mikrokontroler yang digunakan. Biasanya, rentang pengukuran efektifnya adalah sekitar 2 cm hingga 4 meter.
- **Resolusi:** Resolusi sensor tergantung pada periode gelombang ultrasonik yang digunakan dan kecepatan yang diukur. Resolusi yang lebih tinggi dapat dicapai dengan menggunakan periode gelombang yang lebih kecil, namun dengan trade-off pada rentang pengukuran yang lebih rendah.
- **Aplikasi:** Sensor ultrasonik HC-SR04 sering digunakan dalam proyek-proyek robotika untuk menghindari benturan atau menghindari rintangan, pengukuran jarak pada kendaraan, pengukuran air dalam tangki, pengukuran tinggi air, dan berbagai aplikasi lain yang memerlukan pengukuran jarak non-kontak.
- **Penggunaan dengan Mikrokontroler:** Sensor ultrasonik HC-SR04 mudah digunakan dengan mikrokontroler seperti Arduino, ESP atau Raspberry Pi. Dengan menggunakan pin digital pada mikrokontroler, Anda dapat mengirimkan sinyal trig (untuk memulai pengukuran) dan menerima sinyal echo (untuk mengukur waktu pantulan gelombang) untuk menghitung jarak.

Sensor ultrasonik HC-SR04 adalah sensor yang handal dan hemat biaya, membuatnya menjadi pilihan yang populer dalam berbagai proyek elektronik dan robotika. Dengan kemampuannya untuk memberikan informasi jarak non-kontak, sensor ini memungkinkan berbagai aplikasi yang kreatif dan beragam.

## F. Servo SG90

Servo SG90 adalah salah satu servo motor yang paling populer dan banyak digunakan dalam proyek-proyek DIY, robotika, dan model. Servo SG90 dikenal karena ukurannya yang kecil, harga yang terjangkau, dan kemampuan untuk menggerakkan mekanisme dengan presisi tinggi.



Gambar Servo SG90

Berikut adalah beberapa poin penting tentang servo SG90:

- **Prinsip Kerja:** Servo SG90 merupakan jenis servo motor berbasis DC (Direct Current). Servo motor mengandalkan umpan balik posisi untuk menggerakkan porosnya ke posisi yang tepat. Di dalamnya terdapat motor DC, gearbox, dan sebuah potensiometer sebagai sensor umpan balik.
- **Karakteristik Fisik:** Servo SG90 umumnya memiliki ukuran kecil dengan berat sekitar 9 gram. Dimensinya kira-kira 22.2 x 11.8 x 31mm, membuatnya cocok untuk proyek-proyek dengan keterbatasan ruang.
- **Torsi (Torque):** Servo SG90 memiliki torsi yang cukup untuk ukurannya, biasanya sekitar 1,5 kg/cm hingga 2,5 kg/cm. Torsi ini menentukan berapa banyak beban yang dapat diangkat atau digerakkan oleh servo.
- **Kecepatan:** Kecepatan gerakan servo SG90 berkisar antara 0,1 detik hingga 0,14 detik untuk 60 derajat rotasi, tergantung pada kondisi beban.
- **Tegangan Operasi:** Tegangan operasi standar untuk servo SG90 adalah 3V hingga 6V. Penggunaan tegangan di luar rentang ini dapat menyebabkan kerusakan pada servo.
- **Kontrol:** Servo SG90 diatur dengan sinyal PWM (Pulse Width Modulation). Siklus kerja (duty cycle) pulsa PWM mengontrol posisi poros servo. Siklus kerja 0% menghasilkan posisi minimum, 50% menghasilkan posisi tengah, dan 100% menghasilkan posisi maksimum.
- **Aplikasi:** Servo SG90 banyak digunakan dalam berbagai aplikasi, termasuk robotika, kendali pergerakan, pintu otomatis, kamera gimbal, pesawat terbang model, dan banyak lagi.
- **Penggunaan dengan Mikrokontroler:** Servo SG90 mudah diintegrasikan dengan mikrokontroler seperti Arduino, ESP, Raspberry Pi, atau platform lainnya. Biasanya, pin PWM pada mikrokontroler digunakan untuk mengontrol servo.

Meskipun servo SG90 adalah servo motor yang ekonomis, tetapi dapat memberikan kinerja yang andal dalam proyek-proyek sederhana hingga menengah. Penggunaannya yang mudah dan performa yang dapat diandalkan membuatnya menjadi pilihan yang populer bagi para hobiis, pemula, dan pengembang dalam berbagai proyek elektronik dan robotika.

## **G. Relay 2-Channel**

Relay 2 channel adalah modul relay yang memiliki dua saluran atau kanal relay terpisah. Setiap kanal relay dapat berfungsi secara independen untuk mengendalikan perangkat elektronik atau sirkuit listrik. Modul relay ini biasanya digunakan dalam proyek-proyek elektronik, otomasi rumah, dan kendali perangkat dengan tingkat daya yang lebih tinggi.

Berikut adalah beberapa poin penting tentang relay 2 channel:

- **Fungsi Relay:** Relay adalah komponen elektromekanik yang berfungsi sebagai saklar yang dikendalikan oleh sinyal listrik. Ketika diberikan sinyal (biasanya berupa tegangan), relay akan menutup atau membuka sirkuit listrik, sehingga mengizinkan atau menghentikan aliran arus listrik ke perangkat yang dikendalikan.
- **Jumlah Kanal:** Relay 2 channel memiliki dua kanal atau saluran terpisah, yang berarti Anda dapat mengendalikan dua perangkat atau sirkuit listrik secara independen.
- **Tipe Relay:** Relay 2 channel dapat hadir dalam berbagai tipe, termasuk tipe relay elektromekanik (biasanya menggunakan coil elektromagnetik untuk menggerakkan saklar mekanik) dan tipe relay solid-state (SSR) yang menggunakan komponen semikonduktor seperti opto-triac atau opto-MOSFET untuk mengendalikan arus.
- **Kapasitas Beban:** Relay 2 channel biasanya memiliki kapasitas beban yang cukup tinggi, yang memungkinkan mereka mengendalikan perangkat dengan daya yang lebih besar seperti lampu, motor, atau peralatan rumah tangga.
- **Pengendalian:** Untuk mengendalikan relay 2 channel, Anda dapat memberikan sinyal kontrol berupa tegangan dari mikrokontroler seperti Arduino atau Raspberry Pi, atau dari saklar fisik atau tombol.
- **Aplikasi:** Relay 2 channel digunakan dalam berbagai aplikasi, seperti otomasi rumah, kendali peralatan rumah tangga, sistem keamanan, pengendalian motor, dan banyak lagi. Misalnya, Anda dapat menggunakan relay ini untuk menghidupkan atau mematikan lampu, mengendalikan kipas, atau mengaktifkan sistem pengairan otomatis.
- **Keamanan:** Ketika menggunakan relay 2 channel, penting untuk memastikan keamanan dalam mengendalikan perangkat dengan daya tinggi. Pastikan untuk menggunakan perangkat sesuai dengan spesifikasi dan batasan daya relay serta memastikan koneksi dan peralatan listrik lainnya sesuai dengan standar keselamatan.

Relay 2 channel adalah modul yang serbaguna dan berguna dalam berbagai proyek elektronik dan otomasi. Kemampuannya untuk mengendalikan dua perangkat secara independen membuatnya menjadi pilihan yang populer bagi para hobiis dan pengembang dalam berbagai aplikasi yang melibatkan kendali daya listrik.

## **H. RTC DS1307**

RTC DS1307 adalah modul Real-Time Clock (RTC) atau jam waktu nyata yang sangat populer dan sering digunakan dalam proyek-proyek elektronik. RTC DS1307 memungkinkan perangkat mikrokontroler seperti Arduino, ESP, Raspberry Pi, atau platform lainnya untuk mengakses waktu dan tanggal dengan akurasi tinggi bahkan ketika daya listrik telah mati.

Berikut adalah beberapa poin penting tentang RTC DS1307:

- Fungi RTC: RTC DS1307 dirancang untuk menyimpan dan menghitung waktu dan tanggal secara akurat. Ketika perangkat yang menggunakannya dihidupkan, RTC akan memberikan informasi tentang waktu saat ini (jam, menit, dan detik) serta tanggal (hari, bulan, dan tahun).
- Koneksi: RTC DS1307 dihubungkan ke perangkat mikrokontroler melalui antarmuka I2C (Inter-Integrated Circuit) atau dikenal juga sebagai TWI (Two-Wire Interface). I2C memungkinkan komunikasi dua arah antara mikrokontroler dan RTC dengan menggunakan hanya dua kabel, yaitu SDA (Serial Data Line) dan SCL (Serial Clock Line).
- Presisi: RTC DS1307 memiliki presisi yang cukup tinggi dan dapat memberikan akurasi waktu dalam kisaran detik, asalkan telah dikalibrasi dengan benar.
- Perangkat Penyimpanan: RTC DS1307 memiliki beberapa perangkat penyimpanan internal, termasuk register untuk menyimpan informasi waktu dan tanggal, serta register tambahan untuk pengaturan dan pengaturan alarm.
- Tegangan Operasi: RTC DS1307 biasanya beroperasi pada tegangan 5V, tetapi beberapa varian juga dapat beroperasi pada tegangan rendah seperti 3,3V.
- Baterai Cadangan: RTC DS1307 dilengkapi dengan pin VBAT yang dapat dihubungkan ke baterai cadangan, seperti baterai CR2032. Baterai ini berfungsi untuk menyimpan waktu dan tanggal ketika daya listrik utama mati atau perangkat mikrokontroler dalam kondisi mati.
- Aplikasi: RTC DS1307 digunakan dalam berbagai aplikasi yang membutuhkan waktu dan tanggal yang akurat. Beberapa aplikasi termasuk jam digital, logger data dengan penanda waktu, sistem pengatur waktu otomatis, dan berbagai proyek yang memerlukan sinkronisasi waktu.
- Library Pendukung: Ada banyak library yang tersedia untuk mengakses dan mengontrol RTC DS1307 dalam berbagai platform mikrokontroler, termasuk Arduino dan Raspberry Pi.

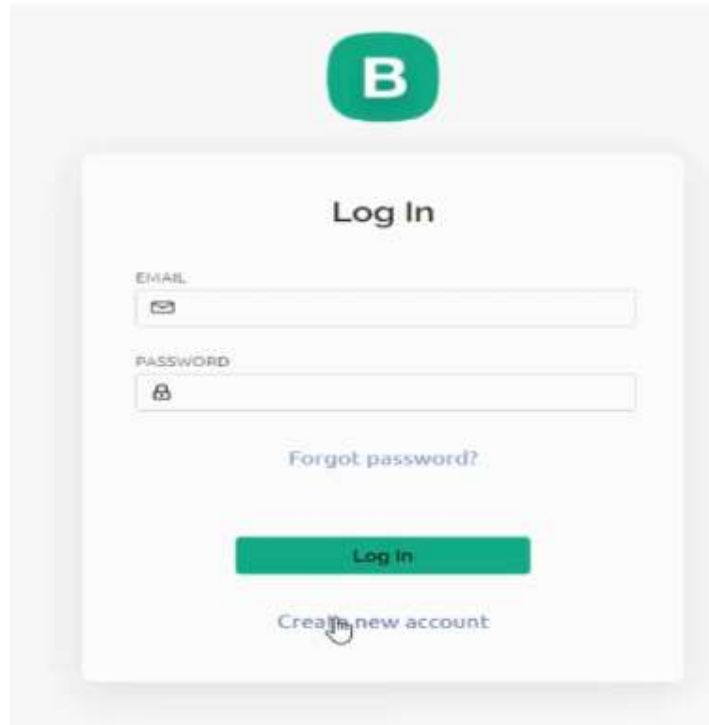
RTC DS1307 adalah modul yang sederhana dan mudah digunakan, tetapi sangat berguna untuk mengatur waktu dalam proyek-proyek elektronik yang memerlukan sinkronisasi waktu atau penanda waktu. Dengan kemampuan untuk berfungsi bahkan ketika daya listrik mati, RTC DS1307 memastikan bahwa perangkat Anda dapat mempertahankan akurasi waktu yang konsisten dalam berbagai kondisi.

## BAB II

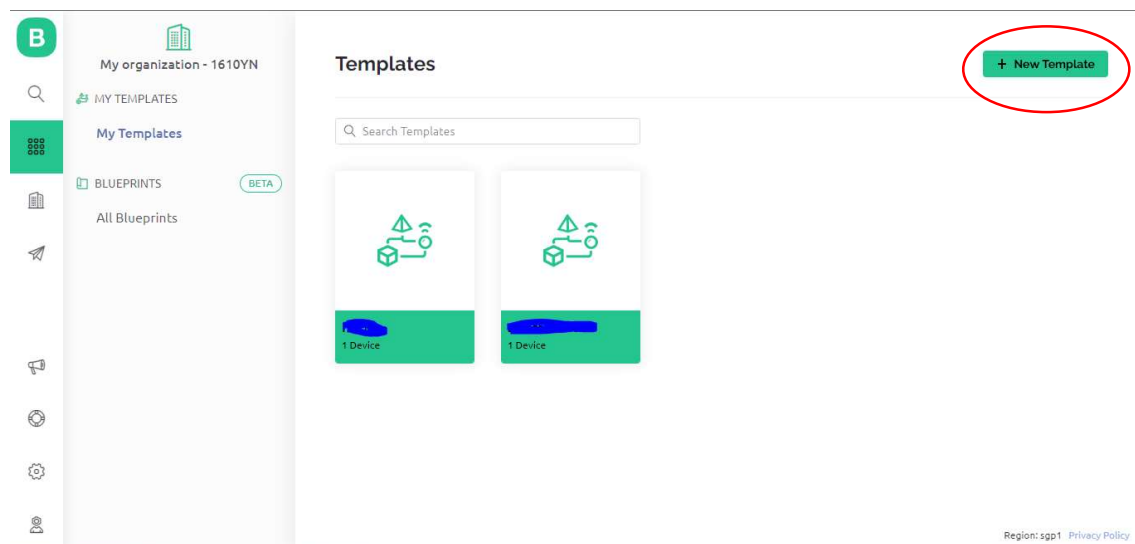
### Cara Menghubungkan ESP32 ke Blynk IoT 2.0

#### 1. Website Blynk IoT 2.0

Hal pertama saat ingin menghubungkan ESP32 ke Blynk IoT 2.0 adalah dengan membuat akun Blynk terlebih dahulu.

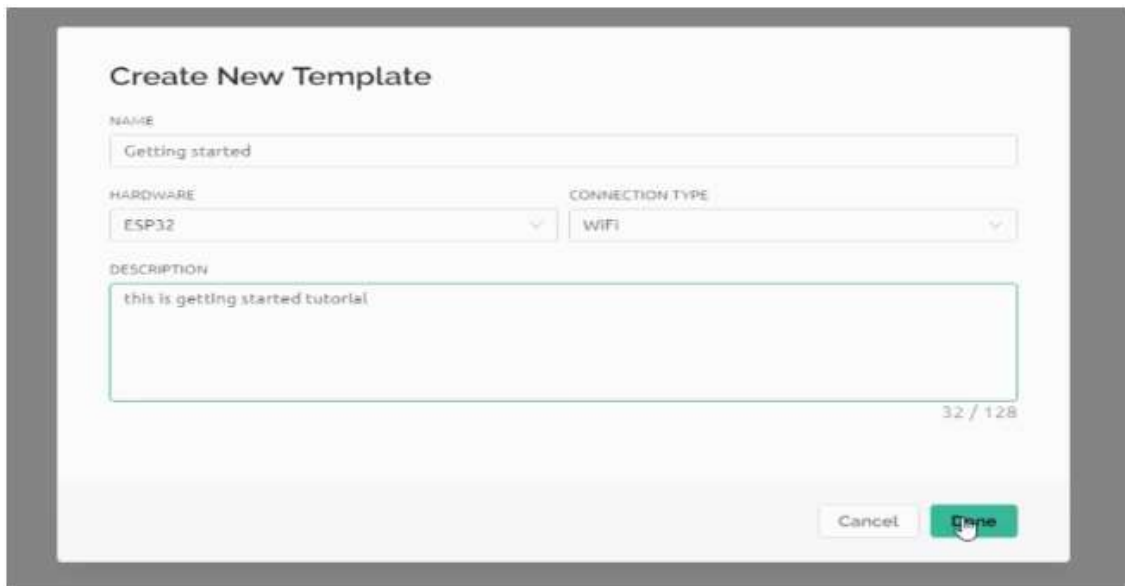


Setelah membuat akun blynk, klik **template** yang ada pada bagian kiri, kemudian klik + **New Template**.

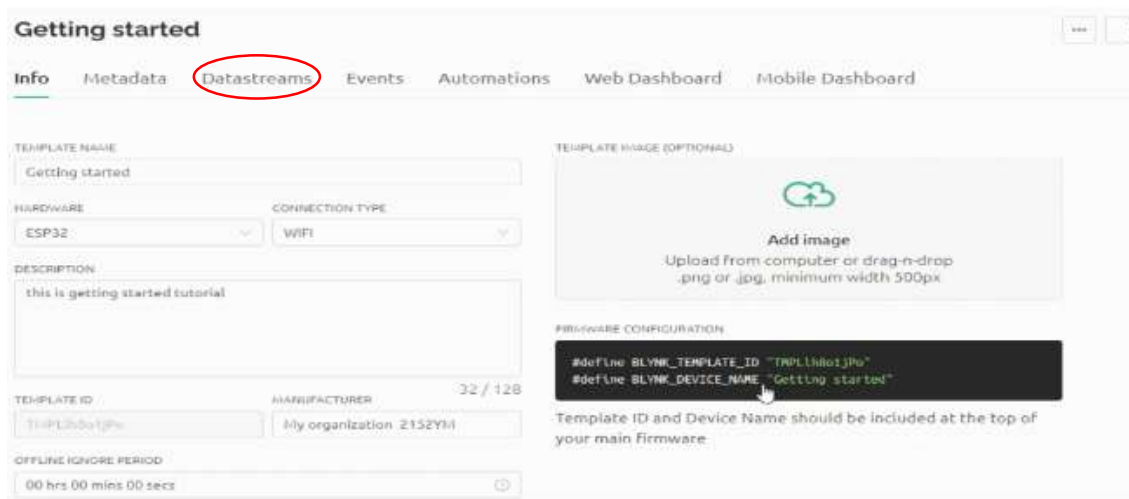


Kemudian isi nama template sesuai keinginan, kemudian pada **HARDWARE**/mikrokontroler pilih ESP32 dan untuk **CONNECTION TYPE** pilih Wifi karena ESP32 akan dihubungkan ke

Blynk IoT melalui wifi (anda juga bisa memberi deskripsi pada template). Kemudian tekan Done.

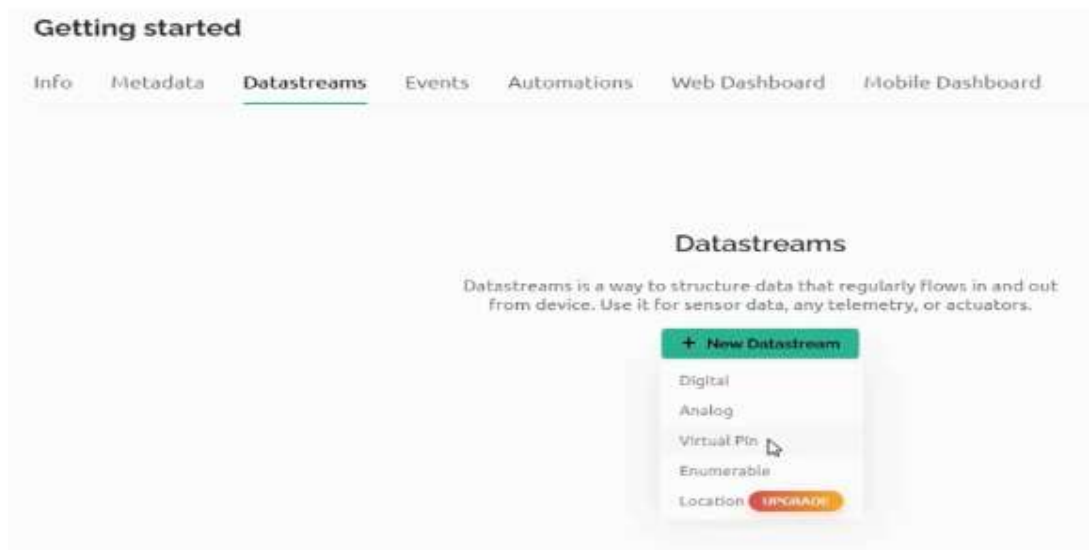


Setelah itu pergi ke datastream untuk menentukan pin yang akan digunakan



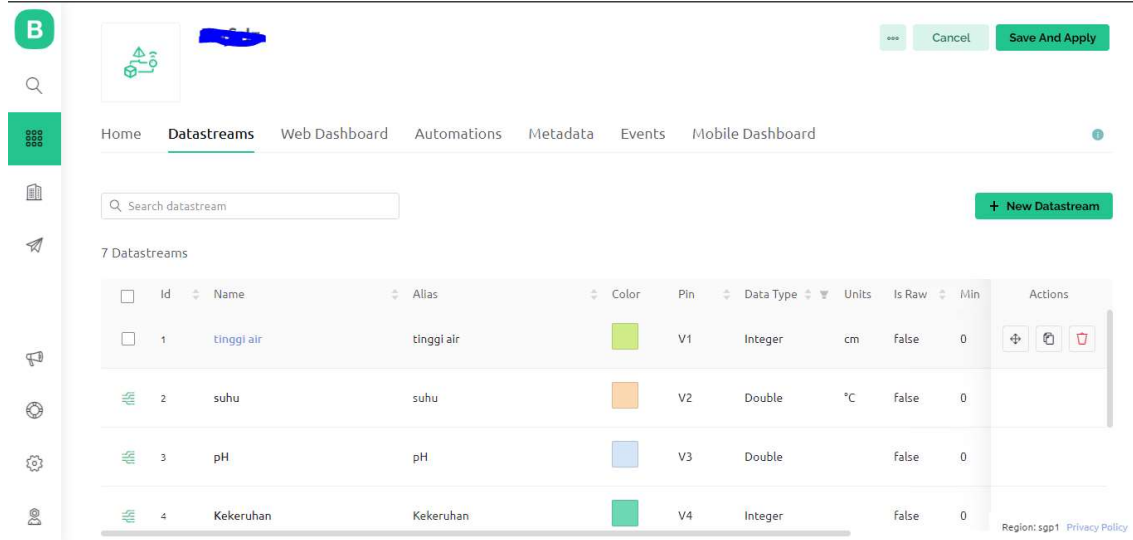
Di Datastreams klik + **New Datastream**, karena pada project ini menggunakan pin V(virtual), maka pilih **Virtual Pin**.



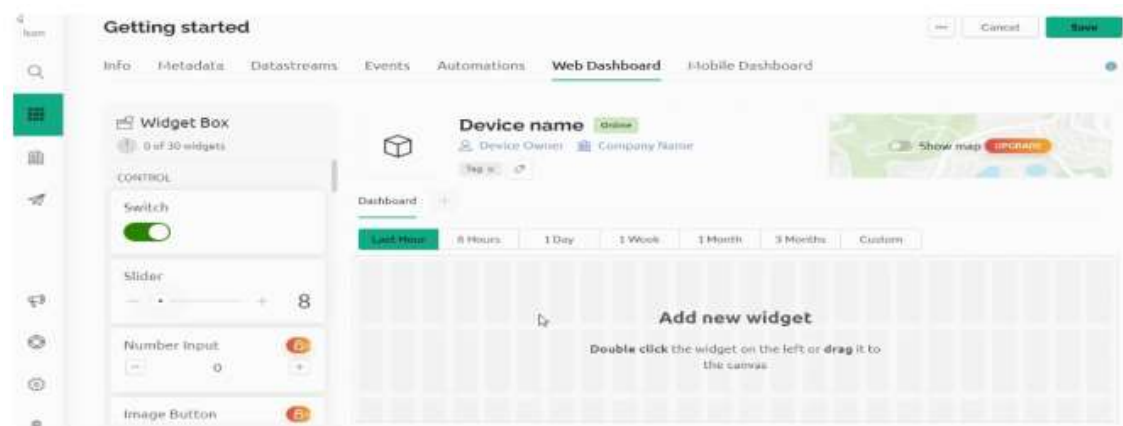


Tulis nama pin, pilih Pin yang akan digunakan, kemudian tentukan tipe data untuk pinnya. Anda juga bisa menambahkan units/satuan pada sensor yang anda gunakan atau memberi nilai maksimum dan minimum nilai yang dapat ditampilkan. Ketika semua parameter telah diisi, maka klik **Create**.

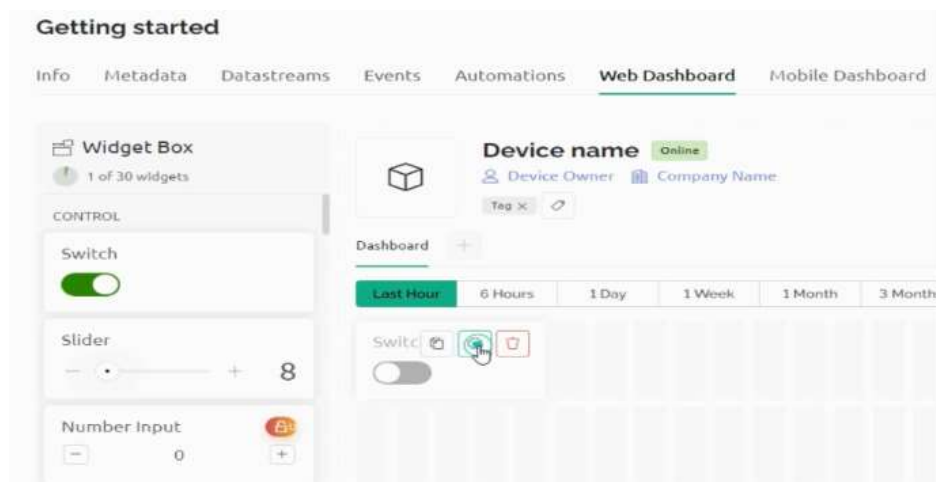
Buat pin-pin yang lain sesuai yang ada pada kode program arduino Ide.



Kemudian langkah selanjutnya adalah menentukan widget yang akan ditampilkan pada web, dengan mengeklik **Web Dashboard**.



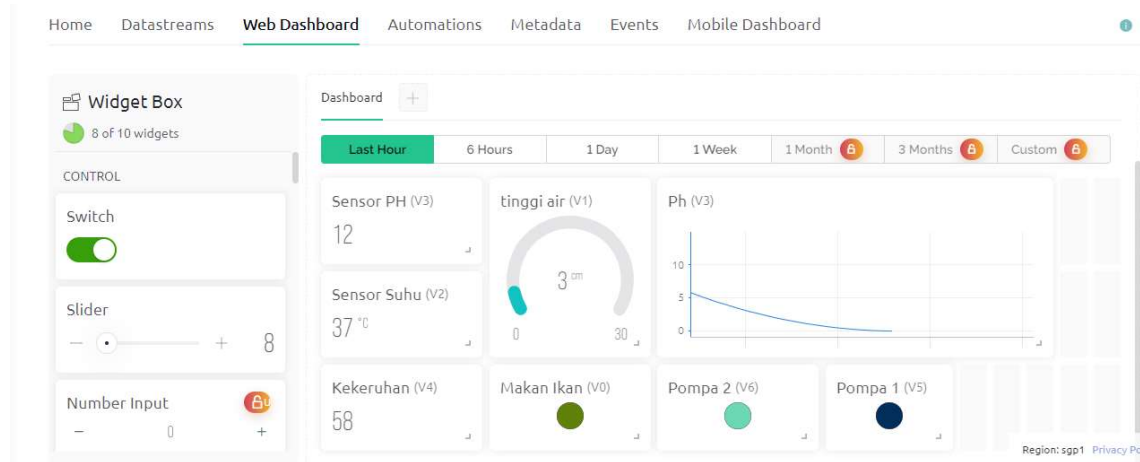
Pada Blynk IoT 2.0 ada beberapa widget yang berbayar yaitu widget yang ada icon gembok. Untuk itu gunakan widget yang gratis. Drag and drop widget yang dibutuhkan. Kemudian klik icon gear.



Pilih datastream yang telah dibuat. Setelah itu klik **Save**



Kemudian pilih widget yang lain sesuai datastream yang telah dibuat.



## 2. Upload Program ke ESP32

Sebelum melakukan upload program, ada beberapa hal yang perlu dipastikan terlebih dahulu.

A. Pastikan Template Blynk pada kode program sudah sesuai dengan template yang akan digunakan.

```
#define BLYNK_TEMPLATE_ID " "
#define BLYNK_TEMPLATE_NAME " "
#define BLYNK_AUTH_TOKEN " "
```

Anda dapat menemukan template blynk ini pada bagian **Search-> Pilih Device-> Device Info**.



Jika pada serial monitor muncul seperti ini:

[19]

```
  / _ ) / / _ _ _ _ / / _  
 / _ / / / / _ \ / ' _/  
 / _ _ / _ \ _ , / _ / / _ \ _ \  
      / _ _ / v0.4.6 on Arduino Uno
```

```
[603] Connecting to vfrsion:1.0.0/0 (Apr 16 2006 13:02:45) □  
[13624] AT vfrsion:1.0.0/0 (Apr 16 2006 13:02:45) □  
SDK version:1.5.3(aec341  
[17702] +CIFSR:ST@IP,"234.204/202.153"  
+CIFSR:STAMAC,"201.10.30.100"  
[17714] Connected to WiFi
```

Itu artinya ESP32 telah terhubung ke Blynk IoT 2.0.

## BAB III

### Penjelasan Kode Program

#### 1. Konfigurasi Template pada Blynk

```
#define BLYNK_TEMPLATE_ID " "
#define BLYNK_TEMPLATE_NAME " "
#define BLYNK_AUTH_TOKEN " "
```

Template pada Blynk memungkinkan Anda untuk menyimpan dan mengelola proyek Anda dengan mudah dan dapat digunakan oleh banyak perangkat dengan token otentikasi yang sama.

Berikut adalah penjelasan dari setiap baris kode:

- `#define BLYNK_TEMPLATE_ID " _ ":` Ini mendefinisikan ID Template proyek Blynk yang digunakan dalam program ini. ID ini unik untuk setiap proyek dan menandakan proyek tertentu yang digunakan.
- `#define BLYNK_TEMPLATE_NAME " _ ":` Ini mendefinisikan nama Template proyek Blynk yang digunakan dalam program ini. Nama ini adalah label untuk memudahkan pengenalan proyek.
- `#define BLYNK_AUTH_TOKEN " _ ":` Ini adalah token otentikasi Blynk yang diperlukan untuk menghubungkan perangkat ke proyek Blynk tertentu. Token ini berfungsi sebagai kunci untuk mengidentifikasi perangkat yang diperbolehkan terhubung ke proyek tersebut.

Dengan menggunakan Template pada Blynk, Anda dapat dengan mudah mengakses dan mengatur ulang konfigurasi proyek di banyak perangkat hanya dengan menggunakan token otentikasi yang sama. Hal ini berguna jika Anda ingin menggunakan proyek Blynk yang sama di beberapa perangkat atau saat Anda ingin berbagi proyek dengan pengguna lain. Template mempermudah Anda untuk mendistribusikan dan memanfaatkan proyek Blynk secara lebih efisien.

Untuk mengetahui template blynk, anda dapat menemukannya pada search > pilih template yang dibuat > device info.

The screenshot shows the Blynk mobile app interface. On the left, there's a sidebar with a search icon and a list of devices. The main screen displays the 'Device Info' for a device named 'Mitigasi banjir dan listrik tegan...'. The device is currently 'Offline'. The 'Device Info' section includes the following details:

- STATUS:** Offline
- LAST UPDATED:** 11:06 AM May 3, 2023
- LAST ONLINE:** 11:06 AM May 3, 2023
- LATEST METADATA UPDATE:** 11:11 AM Mar 18, 2023
- DEVICE ACTIVATED:** 1:13 PM Jan 20, 2023
- ORGANIZATION:** My organization - 1610YN
- AUTH TOKEN:** DjiU - \*\*\*\* - \*\*\*\*
- TEMPLATE NAME:** smartHome77
- MANUFACTURER:** IP

A 'FIRMWARE CONFIGURATION' section is also visible, showing a snippet of code that defines the Blynk template ID, name, and auth token. The code is highlighted in blue:

```
#define BLYNK_TEMPLATE_ID " "
#define BLYNK_TEMPLATE_NAME " "
#define BLYNK_AUTH_TOKEN " "
```

Below the code snippet, a note states: 'Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.'

## 2. Library

```
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <ESP32Servo.h>
#include <NewPing.h>
#include <OneWire.h> //Memanggil
#include <DallasTemperature.h> //
#include <Wire.h>
#include "RTClib.h"// memanggil
```

Berikut adalah penjelasan tentang kode diatas:

- `#define BLYNK_PRINT Serial`: Ini mendefinisikan output untuk mencetak pesan debug dari Blynk ke monitor serial. Dengan cara ini, Anda dapat melihat pesan debug atau kesalahan yang mungkin terjadi saat perangkat terhubung ke layanan Blynk.
- `#include <WiFi.h>`: Ini adalah pustaka untuk mengaktifkan fitur WiFi pada mikrokontroler ESP32. Diperlukan untuk menghubungkan perangkat ke jaringan WiFi.
- `#include <WiFiClient.h>`: Pustaka ini mendukung koneksi WiFi Client untuk perangkat ESP32. Diperlukan untuk melakukan koneksi ke server Blynk.
- `#include <BlynkSimpleEsp32.h>`: Pustaka Blynk untuk platform ESP32 yang menyediakan fungsi-fungsi dasar untuk berkomunikasi dengan server Blynk dan mengelola data dari dan ke Blynk app.
- `#include <ESP32Servo.h>`: Pustaka untuk mengendalikan motor servo pada ESP32. Digunakan dalam program untuk menggerakkan servo yang digunakan dalam memberi makan ikan.
- `#include <NewPing.h>`: Pustaka untuk mendukung sensor ultrasonik NewPing pada ESP32. Digunakan dalam program untuk mengukur jarak dan mengetahui tinggi air di akuarium.
- `#include <OneWire.h>`: Pustaka yang diperlukan sebagai dependensi untuk menggunakan sensor suhu DS18B20. OneWire digunakan untuk berkomunikasi dengan sensor menggunakan satu kabel data.
- `#include <DallasTemperature.h>`: Pustaka untuk mendukung sensor suhu DS18B20 DallasTemperature pada ESP32. Digunakan untuk membaca suhu air di akuarium.
- `#include <Wire.h>`: Pustaka untuk mendukung komunikasi I2C (Inter-Integrated Circuit) pada ESP32. Digunakan untuk berkomunikasi dengan beberapa perangkat seperti RTC DS1307 dalam program ini.
- `#include "RTClib.h"`: Pustaka untuk mendukung RTC DS1307 (Real Time Clock) pada ESP32. Digunakan dalam program untuk mendapatkan data waktu yang tepat.

Untuk versi library yang digunakan adalah sebagai berikut:

- Blynk : 1.2.0
- ESP32Servo : 0.13.0
- NewPing : 1.9.7
- OneWire : 2.3.7
- DallasTemperature : 3.9.0

Untuk library “RTCLib.h” merupakan library dari download di google, jadi anda perlu include sendiri secara manual. Untuk cara include library, anda dapat menggunakan link berikut : <https://www.youtube.com/watch?v=VvE4KLt49OM>.

### 3. Definisi Pin yang Digunakan

```
#define ONE_WIRE_BUS 23 // pin sensor DS18B20.
#define ph_pin 35 // Pin Sensor PH
#define Trig 26 // pin trig ultrasonik
#define Echo 25 // pin echo ultrasonik
#define turbidityPin 34 // pin sensor turbidity
#define servoPin 19 // pin Servo
#define relay1 18 // pin relay1
#define relay2 5 // pin relay2
```

Kode di atas mendefinisikan beberapa pin yang digunakan untuk menghubungkan sensor dan aktuator pada mikrokontroler ESP32. Dengan mendefinisikan pin-pin ini, program dapat mengonfigurasi penggunaan pin yang tepat untuk setiap sensor dan aktuator yang terhubung ke mikrokontroler ESP32. Pastikan bahwa koneksi fisik dari sensor dan aktuator sesuai dengan definisi pin yang telah ditentukan. Jika anda melakukan perubahan atau menambahkan suatu sensor pada sistem ESP32. Maka anda harus mendefinisikan pin baru atau merubahnya.

### 4. Inisialisasi Variabel dan Objek

```
OneWire oneWire(ONE_WIRE_BUS); //
DallasTemperature sensor(&oneWire);
RTC_DS1307 rtc; // membuat variabel
DateTime now;
NewPing sonar(Trig, Echo, MAX_DISTANCE);
Servo myservo;
```

Bagian kode yang Anda berikan adalah inisialisasi variabel dan objek yang akan digunakan dalam program. Mari kita bahas setiap baris secara lebih detail:

- `OneWire oneWire(ONE_WIRE_BUS);`: Ini adalah inisialisasi objek `oneWire` dari kelas `OneWire`. Objek ini digunakan untuk berkomunikasi dengan sensor suhu DS18B20. Konstruktor `OneWire` menerima argumen pin data (data pin) yang telah didefinisikan sebelumnya dengan `ONE_WIRE_BUS`. Jadi, objek `oneWire` akan menggunakan pin yang ditentukan (`ONE_WIRE_BUS`) untuk berkomunikasi dengan sensor suhu.
- `DallasTemperature sensor(&oneWire);`: Ini adalah inisialisasi objek sensor dari kelas `DallasTemperature`. Objek ini akan digunakan untuk membaca data suhu dari sensor suhu DS18B20 yang telah terhubung melalui objek `oneWire`. Konstruktor `DallasTemperature` menerima argumen `&oneWire`, yang merupakan alamat dari objek `oneWire`, sehingga sensor dapat berkomunikasi dengan sensor suhu melalui koneksi `OneWire`.
- `RTC_DS1307 rtc;`: Ini adalah inisialisasi objek `rtc` dari kelas `RTC_DS1307`. Objek ini akan digunakan untuk mengakses modul RTC DS1307 (Real Time Clock) yang telah terhubung ke mikrokontroler. Modul RTC membantu mendapatkan waktu yang tepat untuk memantau perangkat secara waktu nyata.
- `DateTime now;`: Ini adalah deklarasi objek `now` dari kelas `DateTime`. Objek ini akan digunakan untuk menyimpan data waktu yang diperoleh dari modul RTC. Data waktu ini nantinya akan digunakan dalam berbagai fungsi terkait dengan waktu dalam program.



- `NewPing sonar(Trig, Echo, MAX_DISTANCE);`: Ini adalah inisialisasi objek sonar dari kelas `NewPing`. Objek ini akan digunakan untuk mengukur jarak menggunakan sensor ultrasonik. Konstruktor `NewPing` menerima tiga argumen: `Trig` (pin trigger), `Echo` (pin echo), dan `MAX_DISTANCE` (jarak maksimum yang dapat diukur oleh sensor). Objek sonar akan digunakan untuk melakukan pengukuran jarak menggunakan sensor ultrasonik yang terhubung melalui pin-pins yang telah ditentukan.
- `Servo myservo;`: Ini adalah inisialisasi objek `myservo` dari kelas `Servo`. Objek ini digunakan untuk mengontrol motor servo.

## 5. Deklarasi Variabel

```
float suhuDS18B20; //deklarasi variable suhu DS18B20 den
String tanggal, bulan, tahun, jam, menit, detik;
int itahun;
int nilai_analog_PH; // menampung hasil pembacaan ADC se
float nilai_PH; // menampung hasil nilai PH yang sebenar
int turbidityADC = 0;
int turbidityValue = 0;|
char auth[] = BLYNK_AUTH_TOKEN;
```

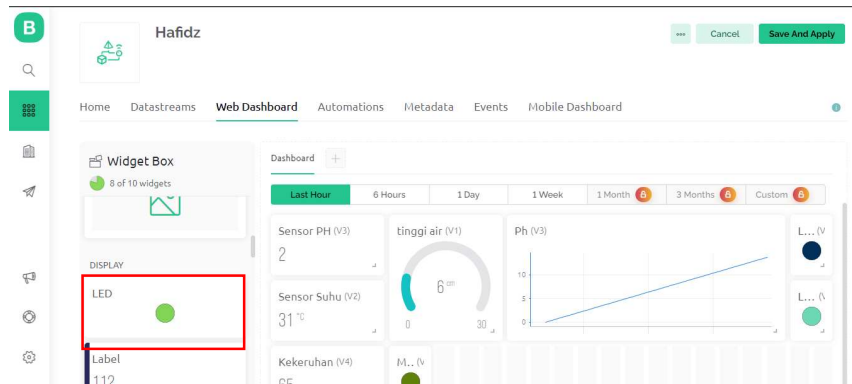
Dalam bagian kode tersebut, terdapat beberapa deklarasi variabel yang berfungsi untuk menyimpan nilai atau data dalam program. Dengan mendeklarasikan variabel-variabel ini, program akan dapat menyimpan dan memanipulasi data suhu, waktu, nilai pH, dan kekeruhan air yang diperlukan dalam proyek pemantauan akuarium ikan berbasis IoT menggunakan ESP32 dan Blynk.

## 6. Objek Fitur-Fitur Blynk

```
BlynkTimer timer;
WidgetLED makanIkan(V0);
WidgetLED pompa1(V5);
WidgetLED pompa2(V6);
```

Dalam bagian kode tersebut, terdapat definisi dari beberapa objek yang digunakan untuk berinteraksi dengan fitur-fitur khusus Blynk.

- `BlynkTimer timer;`: Ini adalah definisi objek timer dari kelas `BlynkTimer`. Objek ini digunakan untuk mengatur jadwal dan waktu untuk menjalankan suatu fungsi secara berkala dalam program. Anda dapat menggunakan objek timer untuk membuat fungsi-fungsi yang dijalankan pada interval tertentu, seperti membaca sensor secara berkala atau mengirim data ke server Blynk pada waktu tertentu.
- `WidgetLED`: ini digunakan untuk mengontrol LED virtual di aplikasi Blynk.



Gambar Wigdet Led

## 7. Blynk Terkoneksi

```
BLYNK_CONNECTED()
{
  // Change Web Link Button message to "Congratulations!"
  Blynk.setProperty(V3, "offImageUri", "https://static-image.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulations.png");
  Blynk.setProperty(V3, "onImageUri", "https://static-image.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulations_pressed.png");
  Blynk.setProperty(V3, "url", "https://docs.blynk.io/en/getting-started/what-do-i-need-to-blynk/how-quickstart-device-was-made");
}
```

Fungsi BLYNK\_CONNECTED() adalah salah satu fitur khusus Blynk yang akan dipanggil oleh pustaka Blynk ketika perangkat berhasil terhubung ke server Blynk. Pada bagian kode di atas, fungsi BLYNK\_CONNECTED() digunakan untuk mengubah tampilan Web Link Button pada aplikasi Blynk setelah perangkat berhasil terhubung. Dengan menggunakan fungsi BLYNK\_CONNECTED(), Anda dapat melakukan berbagai perubahan tampilan atau aksi lain di aplikasi Blynk setelah perangkat berhasil terhubung ke server Blynk. Ini dapat membantu memberikan umpan balik visual kepada pengguna dan memberikan tanda bahwa perangkat telah berhasil terhubung ke layanan Blynk.

## 8. Mengirim Data Ke Blynk

```
void myTimerEvent()
{
  Blynk.virtualWrite(V1, tinggiAir);
  Blynk.virtualWrite(V2, suhuDS18B20);
  Blynk.virtualWrite(V3, nilai_PH);
  Blynk.virtualWrite(V4, turbidityValue);
}
```

Fungsi myTimerEvent() digunakan dalam proyek yang menggunakan platform Blynk untuk mengirim data ke dashboard Blynk. Setiap kali timer berakhir, fungsi ini akan dieksekusi dan mengirimkan data ke empat widget (V1, V2, V3, dan V4) di dashboard Blynk.

Dalam kode di atas, fungsi Blynk.virtualWrite() digunakan untuk mengirimkan data ke widget Blynk yang sesuai. Fungsi ini menerima dua parameter: nomor pin virtual (V1, V2, V3, atau V4) yang sesuai dengan widget di dashboard Blynk, dan nilai variabel yang ingin Anda kirimkan.

## 9. Memberi Makan Otomatis

```
if (jam.equals("8") && menit.equals("30")) {  
    Serial.println("Waktunya memberi makan ikan");  
    makanIkan.on();  
    for (int pos = 0; pos < 180; pos += 1) //fungsi perulangan yang akan dijadikan PWM deng  
    {  
        myservo.write(pos); //prosedur penulisan data PWM ke motor servo  
        delay(15); //waktu tunda 15 ms  
    }  
    delay(1000);  
    for (int pos = 180; pos >= 1; pos -= 1) //fungsi perulangan yang akan dijadikan PWM de  
    {  
        myservo.write(pos);  
        delay(15);  
    }  
    makanIkan.off();  
}
```

Kode diatas merupakan kode yang digunakan untuk memberi makan ikan dengan menggunakan motor servo sebagai mekanisme pemberi makan otomatis. Kode ini berisi instruksi untuk menggerakkan motor servo untuk membuka dan menutup makanan pada waktu tertentu, dalam hal ini ketika jam 08.30 maka servo akan otomatis bergerak. Anda bisa mengatur waktu makan dengan merubah parameter jam.equals() dan menit.equals() berupa string. Fungsi makanIkan.on() dan makanIkan.off() digunakan untuk pemberitahuan pada blynk bahwa waktu makan telah tiba dan servo otomatis bergerak. Widget yang digunakan pada aplikasi blynk berupa widget led yang ada pada V0.

## 10. Menguras air dan mengisi air

```
if (turbidityValue > 25) {  
    pompal.on();  
    while (tinggiAir > (tinggiAkuarium * 0.5)) {  
        Serial.println("Sedang Menguras Air");  
        digitalWrite(relay1, LOW);  
        sensorUltrasonic();  
        Serial.print("tinggi Air :");  
        Serial.println(tinggiAir);  
        Blynk.virtualWrite(V1, tinggiAir);  
        delay(500);  
    }  
    pompal.off();  
    digitalWrite(relay1, HIGH);  
    pompa2.on();  
    while (tinggiAir < tinggiAkuarium * 0.9) {  
        Serial.println("Sedang Mengisi Air");  
        digitalWrite(relay2, LOW);  
        sensorUltrasonic();  
        Serial.print("tinggi Air :");  
        Serial.println(tinggiAir);  
        Blynk.virtualWrite(V1, tinggiAir);  
        delay(500);  
    }  
    pompa2.off();  
    digitalWrite(relay2, HIGH);  
}
```

Kode diatas merupakan kode untuk mengendalikan aliran air di dalam akuarium berdasarkan nilai sensor turbidity (kekeruhan air). Dalam kode ini, ketika nilai kekeruhan air (turbidityValue) melebihi 25, maka akan dilakukan pengaturan aliran air untuk menjaga kebersihan air dalam akuarium. Anda dapat merubah nilai 25 tersebut sesuai dengan keinginan anda.

Kode ini menggunakan loop while untuk mengatur aliran air selama tinggiAir masih berada di luar batas yang diinginkan. Selama proses menguras air atau mengisi air, nilai tinggiAir akan terus dibaca menggunakan fungsi sensorUltrasonic() dan dikirimkan ke widget V1 di dashboard Blynk untuk memantau tinggi air dalam akuarium.

Pada while loop pertama, ketika tinggiAir masih lebih besar dari pada 50%(0.5) tinggi akuarium, maka relay1(pompa penguras air) hidup dan akan menguras air hingga 50% setelah itu relay1 mati. Selanjutnya pada while loop kedua, ketika tinggi air kurang dari 90% tinggi akuarium, maka relay2 (pompa pengisi air) hidup dan akan mengisi air hingga 90% tinggi akuarium. Setelah mencapai 90% maka relay 2 akan mati. (anda dapat merubah pengkondisian tersebut).

Untuk variabel tinggiAkuarium dapat anda temukan pada bagian bawah define pin. Anda dapat merubah nilai tinggiAkuarium sesuai tinggi akuarium yang anda akan gunakan.

```
#define ONE_WIRE_BUS 23 // pin sensor DS18B20.
#define ph_pin 2 // Pin Sensor PH
#define Trig 26 // pin trig ultrasonik
#define Echo 25 // pin echo ultrasonik
#define turbidityPin 34 // pin sensor turbidity
#define servoPin 19 // pin Servo
#define relay1 18 // pin relay1
#define relay2 5 // pin relay2

// Maksimum jarak yang dapat diukur oleh sensor ultrasonik (dalam centimeter)
#define MAX_DISTANCE 200
#define tinggiAkuarium 30 // masukkan tinggi akuarium dalam satuan cm
int distance = 0, tinggiAir; // variabel untuk menampung nilai jarak sensor ultrasonik
```

## 11. Void setup()

Pada fungsi setup(), beberapa inisialisasi dan pengaturan awal dilakukan sebelum program utama berjalan secara terus-menerus di dalam fungsi loop(). Berikut adalah penjelasan dari setiap baris kode dalam fungsi setup():

```
Serial.begin(115200);
```

Memulai komunikasi Serial dengan baud rate 115200 untuk digunakan dalam Serial Monitor (untuk debugging dan informasi lainnya).

```
pinMode(ph_pin, INPUT);
pinMode(turbidityPin, INPUT);
myservo.attach(servoPin);
sensor.begin(); //Menginisiasikan sensor One-Wire DS18B20
pinMode(relay1, OUTPUT);
pinMode(relay2, OUTPUT);
```

Mengatur pin ph\_pin dan turbidityPin sebagai input, karena keduanya akan digunakan untuk membaca data dari sensor pH dan turbidity. Menghubungkan motor servo ke pin servoPin untuk mengontrol mekanisme pemberi makan ikan. Memulai komunikasi dengan sensor

DS18B20 (One-Wire) untuk membaca data suhu. Mengatur pin relay1 dan relay2 sebagai output untuk mengontrol valve atau relay yang mengatur aliran air.

```
digitalWrite(relay1, HIGH);  
digitalWrite(relay2, HIGH);
```

Pada kode diatas relay1, dan relay2 dalam keadaan HIGH sehingga pompa1 dan pompa2 mati, alasan mengapa kondisi HIGH membuat pompa mati karena kami (developer STC) berasumsi relay dipasang dengan pompa dengan kondisi Normally Close (NC). Jika relay dan pompa terpasang secara Normally Open (NO), maka kode diatas harus dirubah menjadi LOW agar pompa mati. Untuk lebih jelasnya anda mengikuti link berikut: <https://arduinogetstarted.com/tutorials/arduino-relay>.

```
if (!rtc.begin())  
{  
  Serial.println("Couldn't find RTC");  
  while (1)  
  ;  
}
```

Maksud dari kode diatas adalah untuk memulai komunikasi dengan modul RTC (Real-Time Clock) dan melakukan pemeriksaan apakah RTC dapat ditemukan. Jika tidak, program akan berhenti dalam loop.

```
Blynk.begin(auth, ssid, pass);  
timer.setInterval(500L, myTimerEvent);
```

Memulai koneksi ke server Blynk dengan menggunakan token autentikasi (auth), serta memberikan informasi SSID dan password WiFi. Mengatur interval waktu untuk pemanggilan fungsi myTimerEvent() dengan menggunakan timer Blynk. Fungsi myTimerEvent() kemudian akan dipanggil secara berkala sesuai dengan interval waktu yang telah ditentukan.

```
rtc.adjust(DateTime(F(__DATE__), F(__TIME__))); // Set waktu langsung dari waktu PC  
bacaRTC();
```

Mengatur waktu RTC dengan waktu saat ini dari waktu kompilasi program (F(\_\_\_\_DATE\_\_\_\_) dan F(\_\_\_\_TIME\_\_\_\_) memberikan waktu kompilasi program) sesuai waktu yang ada pada laptop. Gunakan rtc.adjust() satu kali saja, setelah anda UPLOAD sekali, komen kode tersebut dan UPLOAD ulang kode ini. Sedangkan untuk fungsi bacaRTC(); merupakan fungsi membaca waktu rtc yang akan dijelaskan pada pembahasan berikutnya.

## 12. Void Loop()

```
void loop() {  
  bacaRTC();  
  sensorPh();  
  sensorUltrasonic();  
  sensorSuhu();  
  sensorTurbidity();  
  tampilkanSemuaSensor();  
  
  delay(1000);  
  Blynk.run();  
  timer.run();  
}
```

Pada void loop(), hanya berisi kode untuk menjalankan berbagai fungsi-fungsi sensor yang telah dibuat dan akan dibahas dipembahasan berikutnya.

Untuk Blynk.run(): Menjalankan fungsi Blynk untuk memastikan koneksi dan interaksi dengan server Blynk. Ini diperlukan agar widget Blynk dapat berfungsi dan menerima pembaruan data dari sensor.

timer.run(): Menjalankan fungsi timer Blynk yang telah diatur di dalam setup(). Fungsi ini memastikan bahwa myTimerEvent() (fungsi yang mengirimkan data ke dashboard Blynk) dieksekusi sesuai dengan interval waktu yang telah ditentukan.

## 13. Sensor PH

```
void sensorPh() {  
  nilai_analog_PH = analogRead(ph_pin);  
  // Serial.print("Nilai ADC Ph: ");  
  // Serial.println(nilai_analog_PH);  
  nilai_PH = (nilai_analog_PH - 5223.3) / -285.68;  
  // Serial.print("Nilai PH: ");  
  // Serial.println(nilai_PH);  
}
```

Kode diatas merupakan fungsi untuk membaca nilai pH air. Perhitungan diatas didapat dari hasil kalibrasi dengan metode linear regresion. Jika anda merasa pembacaan sensor masih kurang akurat. Anda dapat melakukan kalibrasi ulang dengan menyediakan 3 sample air dengan pH yang berbeda. Jika anda kesulitan untuk mengetahui nilai pH air, anda dapat membeli pH buffer serbuk yang ada pada toko online. Untuk cara kalibrasi dengan metode liniear regression anda dapat melihatnya pada link berikut: <https://www.youtube.com/watch?v=MK6FcvWWCcc&t=1210s>

Jika anda ingin menggunakan metode lain, anda dapat merubahnya pada fungsi tersebut.



## 14. Sensor Ultrasonik

```
void sensorUltrasonic() {  
    distance = sonar.ping_cm();  
    tinggiAir = tinggiAkuarium - distance;  
    if (tinggiAir < 0) {  
        tinggiAir = 0;  
    }  
}
```

Fungsi `sensorUltrasonic()` adalah fungsi yang digunakan untuk mengukur tinggi air dalam akuarium menggunakan sensor ultrasonik. Sensor ultrasonik biasanya memiliki kemampuan untuk mengirimkan gelombang ultrasonik dan mendeteksi pantulannya kembali untuk mengukur jarak objek. Dalam kode ini, kami menggunakan library `NewPing` yang mengizinkan penggunaan sensor ultrasonik dengan mudah. Library ini dibuat khusus untuk sensor ultrasonik sehingga hasil pembacaan sensor lebih akurat. Akan tetapi sensor hanya bisa membaca dengan nilai integer. Jika anda ingin hasil pembacaan sensor berupa bilangan float. Anda tidak perlu menggunakan library `NewPing`. Anda hanya perlu memasukkan rumus jarak ke dalam fungsi tersebut.

Didalam fungsi tersebut terdapat pengkondisian dimana ketika tinggi air kurang dari 0 (-1,-2,-3 ....) sensor ultrasonik tidak dapat mendeteksi permukaan air atau objek lain di atasnya (di luar jangkauan). Untuk menghindari nilai negatif yang tidak masuk akal, `tinggiAir` diatur menjadi nol.

## 15. Sensor Suhu DS18B20

```
void sensorSuhu() {  
    sensor.setResolution(11);  
    sensor.requestTemperatures(); // Perintah konversi suhu  
    suhuDS18B20 = sensor.getTempCByIndex(0); //Membaca data suhu da  
}
```

`sensor.setResolution(11);`: Mengatur resolusi sensor suhu menjadi 11 bit. Resolusi yang lebih tinggi menghasilkan akurasi yang lebih baik, tetapi memerlukan waktu lebih lama untuk melakukan konversi suhu.

`sensor.requestTemperatures();`: Meminta sensor suhu untuk melakukan konversi suhu dan mengukur suhu aktual. Proses konversi ini biasanya memakan waktu beberapa ratus milidetik tergantung pada resolusi yang diatur.

`suhuDS18B20 = sensor.getTempCByIndex(0);`: Menggunakan fungsi `getTempCByIndex(0)` dari library "`DallasTemperature`" untuk membaca data suhu dalam satuan Celsius dari sensor indeks 0 dan menyimpannya di variabel `suhuDS18B20`. Jika Anda menggunakan lebih dari satu sensor DS18B20 (dalam rangkaian paralel), Anda dapat menggunakan indeks 1, 2, dan seterusnya untuk membaca data dari sensor-sensor tambahan tersebut.

Setelah fungsi `sensorSuhu()` dieksekusi, variabel `suhuDS18B20` akan berisi nilai suhu aktual dalam satuan Celsius.

## 16. Sensor Turbidity

```
void sensorTurbidity() {  
    turbidityADC = analogRead(turbidityPin);  
    // Serial.print("ADC Sensor: ");  
    // Serial.println(turbidityADC);  
    turbidityValue = map(turbidityADC, 0, 2100, 100, 0);  
    if (turbidityValue < 0) {  
        turbidityValue = 0;  
    }  
}
```

Dalam kode di atas, ada beberapa hal yang dilakukan:

- `turbidityADC = analogRead(turbidityPin);`: Menggunakan fungsi `analogRead()` untuk membaca nilai ADC dari sensor turbidity yang terhubung ke pin `turbidityPin`. Nilai ADC tersebut akan berada dalam rentang 0 hingga 4095 (pada platform Arduino atau sejenisnya yang menggunakan ADC dengan resolusi 12 bit).
- `turbidityValue = map(turbidityADC, 0, 2100, 100, 0);`: Menggunakan fungsi `map()` untuk mengubah nilai ADC menjadi nilai kekeruhan dalam satuan persen (dari 0 hingga 100). Nilai ADC yang terbaca dari sensor turbidity (0 hingga 2100) akan dikonversi ke dalam rentang nilai kekeruhan (0 hingga 100).
- `if (turbidityValue < 0) turbidityValue = 0;`: Setelah mengkonversi nilai ADC menjadi nilai kekeruhan, dilakukan pemeriksaan apakah nilainya lebih kecil dari nol (atau di luar jangkauan ADC). Jika demikian, artinya sensor turbidity tidak dapat memberikan nilai valid (misalnya, sensor terlepas atau mengalami masalah). Untuk menghindari nilai negatif yang tidak masuk akal, `turbidityValue` diatur menjadi nol.

Perlu diketahui, dikarenakan keterbatasan alat ukur, kami tidak bisa menjamin pengukuran yang dilakukan valid atau tidak. Kami tidak bisa melakukan kalibrasi terhadap sensor dan hanya menggunakan metode min max. jika anda memiliki alat ukur turbidity meter, anda kami sarankan untuk melakukan kalibrasi sendiri agar pembacaan sensor valid. Anda dapat menggunakan metode linier regresi seperti pada sensor pH. Untuk cara kalibrasi anda dapat melihatnya pada link berikut: <https://www.youtube.com/watch?v=KucFuwptGfl&t=315s>



## 17. Menampilkan Pembacaan Sensor Pada Serial Monitor Arduino IDE

```
void tampilkanSemuaSensor() {  
  // menampilkan waktu  
  Serial.println("\n" + konversi_jam(jam) + ":" + konversi_jam(menit) + ":" + konversi_jam(detik));  
  // menampilkan ketinggian air  
  Serial.print("Tinggi Air : ");  
  Serial.print(tinggiAir);  
  Serial.println(" cm");  
  //menampilkan nilai suhu air  
  Serial.print("Suhu : ");  
  Serial.print(suhuDS18B20, 3); //Presisi 3 digit  
  Serial.println(" C");  
  
  // menampilkan nilai PH air  
  Serial.print("Nilai PH: ");  
  Serial.println(nilai_PH);  
  // menampilkan nilai turbidity  
  Serial.print("Turbidity: ");  
  Serial.print(turbidityValue);  
  Serial.println(" NTU");  
  Serial.println();  
}
```

Kode di atas merupakan fungsi untuk menampilkan hasil pembacaan seluruh sensor ke Serial monitor arduino IDE.

## 18. RTC

```
void bacaRTC()  
{  
  DateTime now = rtc.now(); // Ambil data waktu dari DS1307  
  tanggal = String(now.day(), DEC);  
  bulan = String(now.month(), DEC);  
  itahun = now.year() - 2000;  
  tahun = String(itahun);  
  jam = String(now.hour(), DEC);  
  menit = String(now.minute(), DEC);  
  detik = String(now.second(), DEC);  
}
```

Fungsi bacaRTC() adalah fungsi yang digunakan untuk membaca data waktu dari modul RTC (Real-Time Clock) DS1307 dan menyimpannya ke dalam variabel-variabel terkait, seperti tanggal, bulan, tahun, jam, menit, dan detik. Fungsi ini memastikan bahwa data waktu selalu terbaru dari RTC untuk digunakan dalam program.

Setelah fungsi bacaRTC() dieksekusi, variabel-variabel tanggal, bulan, tahun, jam, menit, dan detik akan berisi data waktu terkini dari modul RTC DS1307. Data waktu ini dapat digunakan untuk menampilkan waktu di Serial Monitor, mengatur waktu pada RTC, atau operasi lain yang memerlukan data waktu dalam program Anda.

## 19. Menambahkan angka 0

```
String konversi_jam(String angka) // Fungsi untuk
{
    if (angka.length() == 1)
    {
        angka = "0" + angka;
    }
    else
    {
        angka = angka;
    }
    return angka;
}
```

Fungsi `konversi_jam()` adalah fungsi yang digunakan untuk mengubah angka satuan menjadi format dua digit dengan menambahkan angka 0 di depannya. Misalnya, jika angka yang diberikan adalah "1", maka fungsi ini akan mengembalikan "01". Dengan cara ini, fungsi `konversi_jam()` dapat digunakan untuk mengubah angka satuan menjadi format dua digit sehingga sesuai untuk ditampilkan pada LCD atau tampilan lainnya yang memerlukan format waktu yang lebih standar.

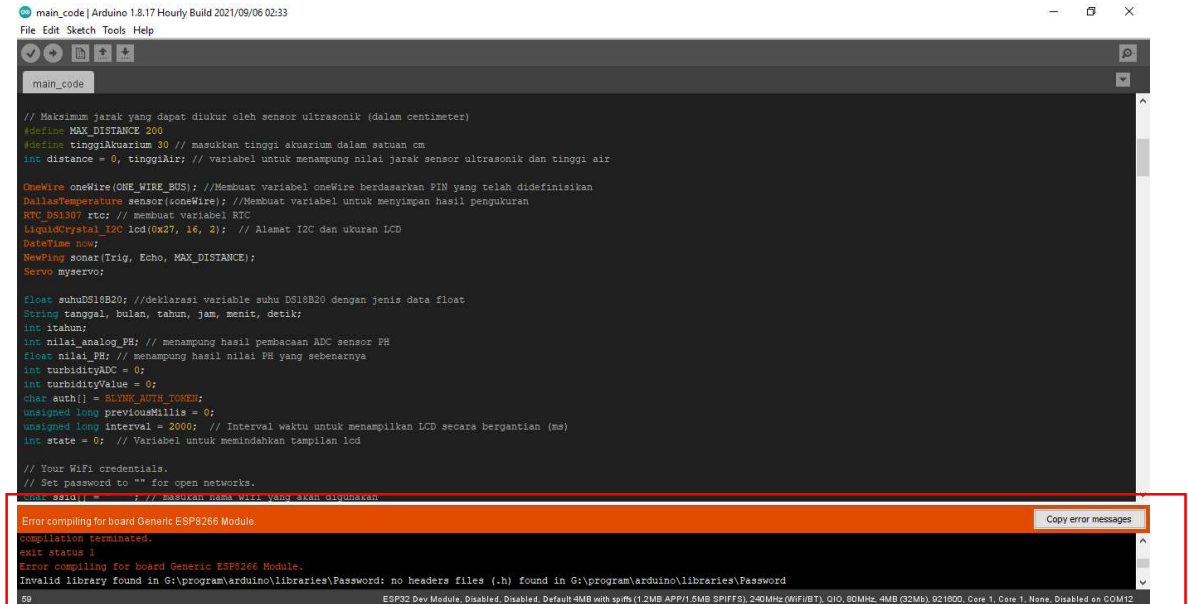
## BAB IV

### Cara Menangani Error dan Kendala pada Sistem

#### 1. Error Saat Upload Kode Program ke ESP32

Jika terjadi error saat upload program, yang perlu anda lakukan adalah sebagai berikut:

- A. Baca error yang muncul pada bagian bawah arduino IDE, disana akan diperlihatkan error apa yang terjadi pada program



```
// Maksimum jarak yang dapat diukur oleh sensor ultrasonik (dalam centimeter)
#define MAX_DISTANCE 200
#define tinggiAkuarium 30 // masukkan tinggi akuarium dalam satuan cm
int distance = 0, tinggiAir; // variabel untuk menampung nilai jarak sensor ultrasonik dan tinggi air

OneWire oneWire(ONE_WIRE_BUS); //Membuat variabel oneWire berdasarkan PIN yang telah didefinisikan
DallasTemperature sensor(oneWire); //Membuat variabel untuk menyimpan hasil pengukuran
RTC_DS18B20 rtc; // membuat variabel RTC
LiquidCrystal_I2C lcd(0x27, 16, 2); // Alamat I2C dan ukuran LCD
DateTime now;
NewPing sonar(Trig, Echo, MAX_DISTANCE);
Servo myservo;

float suhuDS18B20; //deklarasi variable suhu DS18B20 dengan jenis data float
String tanggal, bulan, tahun, jam, menit, detik;
int itahun;
int nilai_analog_PH; // menampung hasil pembacaan ADC sensor PH
float nilai_PH; // menampung hasil nilai PH yang sebenarnya
int turbidityADC = 0;
int turbidityValue = 0;
char auth[] = "BLYNK_AUTH_TOKEN";
unsigned long previousMillis = 0;
unsigned long interval = 2000; // Interval waktu untuk menampilkan LCD secara bergantian (ms)
int state = 0; // Variabel untuk memindahkan tampilan lcd

// Your WiFi credentials.
// Set password to "" for open networks.
// =====
// =====

Error compiling for board Generic ESP8266 Module.
compilation terminated.
Error compiling for board Generic ESP8266 Module.
Invalid library found in G:\program\arduino\libraries\Password: no headers files (.h) found in G:\program\arduino\libraries\Password
```

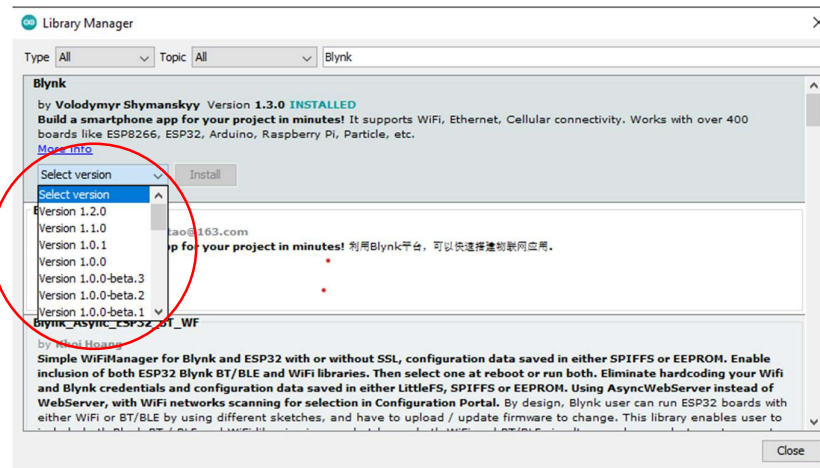
- B. Jika error yang terjadi adalah kesalahan sintak kode program, maka perbaiki kode program(karena arduino menggunakan bahasa C/C++ jadi kesalahan sedikit saja bisa menyebabkan error). Di arduino IDE, kesalahan sintak ditandai dengan blok merah. Berikut adalah contoh kesalahan syntaks.

```
String tanggal, bulan, tahun, jam, menit, detik;
int itahun;
int nilai_analog_PH; // menampung hasil pembacaan ADC sensor PH
float nilai_PH; // menampung hasil nilai PH yang sebenarnya
```

Diatas menandakan adanya kurang ; pada int itahun. Untuk memperbaikinya tinggal menambahkan ; pada variabel tersebut.

```
String tanggal, bulan, tahun, jam, menit, detik;
int itahun;
int nilai_analog_PH; // menampung hasil pembacaan ADC sensor PH
```

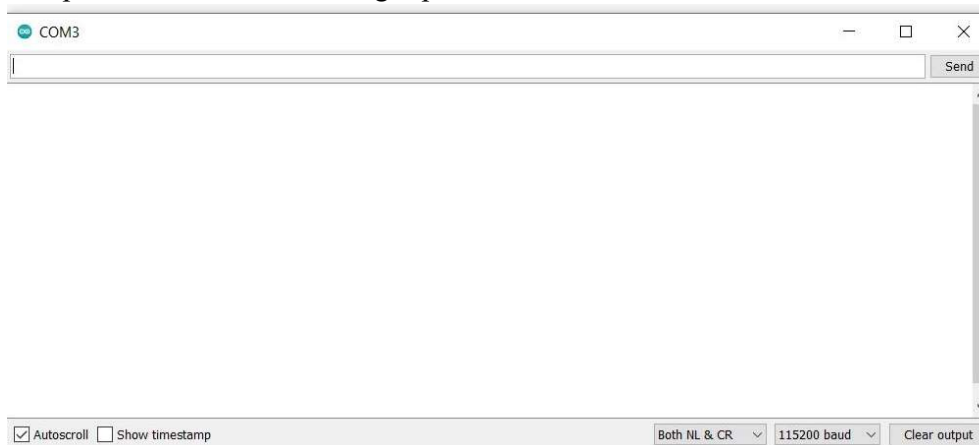
- C. Kesalahan pada library yang digunakan, kesalahan ini sering terjadi ketika asal membuat kode program tanpa memperhatikan library yang digunakan. Versi library juga penting untuk diperhatikan. Perbedaan pada versi library biasanya ada beberapa kode program pada library tersebut yang dirubah. Sehingga ketika menggunakan library yang sama tapi beda versi kadang juga bisa membuat program error. Untuk memperbaikinya, install library sesuai dengan versi library yang digunakan. Saat anda download library pada arduino IDE maka anda disuruh memilih versi library yang akan anda gunakan.



## 2. Gagal dalam Menghubungkan ESP32 ke Blynk

Jika anda gagal atau belum bisa menghubungkan ESP32 ke Blynk IoT. Hal pertama yang anda harus lakukan adalah melihat serial monitor yang terdapat pada arduino IDE.

A. Jika pada serial monitor kosong seperti ini:



Maka yang anda harus lakukan adalah tekan tombol EN pada board mikrokontroler ESP32.



B. Jika gambar yang muncul serial monitor adalah seperti ini:



Itu artinya ESP32 sedang berusaha untuk menghubungkan wifi, jika proses ini terlalu lama, ada kemungkinan bahwa nama SSID/Password wifi pada kode program yang anda masukan salah atau wifi yang anda gunakan tidak ada atau dibatasi. Untuk mengatasi kendala ini, silahkan cek SSID/Password wifi pada kode program yang anda masukan dan lakukan pengecekan pada wifi yang digunakan, apakah wifi mati atau dibatasi pada perangkat.

C. Jika serial monitor menunjukkan hasil sebagai berikut:

```

COM8

[517] Connecting to Storm Fiber
[13550] AT session:1.2.0.0(Jul 1 2016 20:04:45)
SDK version:1.5.4.1(39cbl
[23611] +CIFSR:STAIP,"[redacted]"
+CIFSR:STAMAC,"[redacted]"

[23612] Connected to WiFi
[34062] Ready (ping: 11ms).
[69619] Ready (ping: 11ms).
[105178] Ready (ping: 12ms).

☒ Autoscroll ☐ Show timestamp
Both NL & CR 115200 baud Clear output

```

Itu artinya ESP32 berhasil terhubung ke server Blynk IoT, tapi jika pada website Blynk IoT tidak ada perubahan (perubahan terjadi hanya pada bagian **Search->Device** bukan pada bagian **Templates**). Maka besar kemungkinan anda menggunakan virtual pin yang salah atau widget yang salah. hal yang anda lakukan adalah memeriksa virtual pin pada Blynk IoT dan pada kode program sudah sesuai atau belum. Periksa juga widget yang digunakan apakah sudah sesuai dengan yang anda inginkan.

Berikut adalah virtual pin yang digunakan pada sistem ini.

```

WidgetLED makanIkan(V0);
WidgetLED pompal(V5);
WidgetLED pompa2(V6);

Blynk.virtualWrite(V1, tinggiAir);
Blynk.virtualWrite(V2, suhuDS18B20);
Blynk.virtualWrite(V3, nilai_PH);
Blynk.virtualWrite(V4, turbidityValue);

```

Sesuaikan virtual pin pada website blynk IoT dengan virtual pin tersebut.

### 3. Kesalahan Sensor saat Pengukuran

Jika sensor salah dalam melakukan pengukuran atau tidak adanya perubahan saat pengukuran dengan media yang berbeda. Itu terjadi karena wiring kabel dimana kabel kurang kencang atau salah dalam menyambungkan kabel ke pin ESP32. Atau ada kabel yang putus. Untuk mengatasinya anda dapat menggunakan multitester untuk memeriksa sambungan kabel anda ada yang bermasalah atau tidak.

Terima Kasih Telah  
Menggunakan Jasa Kami.