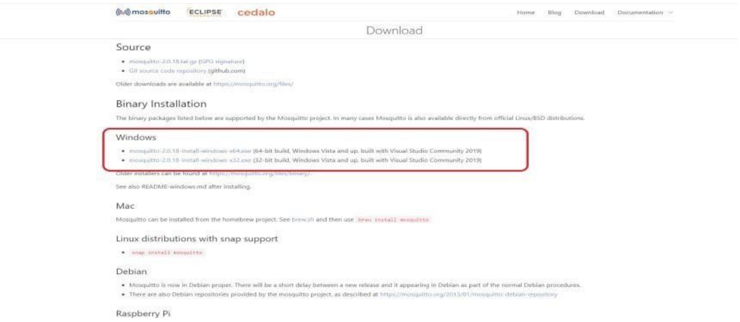
Instalasi MQTT broker (Mosquitto)

1. Unduh mosquitto dari halaman website resmi: https://mosquitto.org/download/. Pilih sistem operasi yang sesuai dengan komputer yang digunakan sebagai server.



Home Blog Download Documentation

Download

Source

- mosquitto-2.0.18.tar.gz (GPG signature)
- Git source code repository (github.com)

Older downloads are available at https://mosquitto.org/files/

Binary Installation

The binary packages listed below are supported by the Mosquitto project. In many cases Mosquitto is also available directly from official Linux/BSD distributions.

Windows

- mosquitto-2.0.18-install-windows-x64.exe (64-bit build, Windows Vista and up, built with Visual Studio Community 2019)
- mosquitto-2.0.18-install-windows-x32.exe (32-bit build, Windows Vista and up, built with Visual Studio Community 2019)

Older installers can be found at https://mosquitto.org/files/binary/.

See also README-windows.md after installing.

Mac

Mosquitto can be installed from the homebrew project. See brew.sh and then use `brew install mosquitto`

Linux distributions with snap support

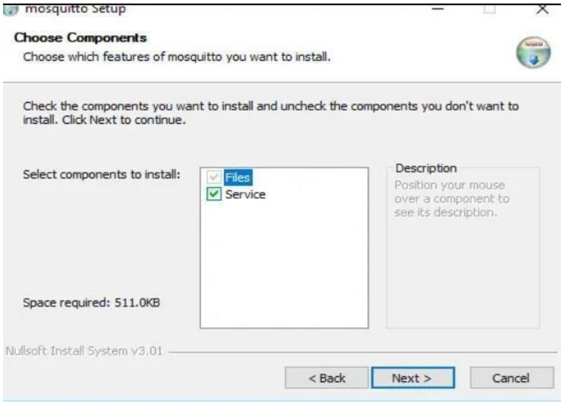
- `snap install mosquitto`

Debian

- Mosquitto is now in Debian proper. There will be a short delay between a new release and it appearing in Debian as part of the normal Debian procedures.
- There are also Debian repositories provided by the mosquitto project, as described at https://mosquitto.org/2013/01/mosquitto-debian-repository

Raspberry Pi

2. Saat Lakukan proses instalasi sesuai arahan, seperti gambar dibawah ini



mosquitto Setup

Choose Components

Choose which features of mosquitto you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

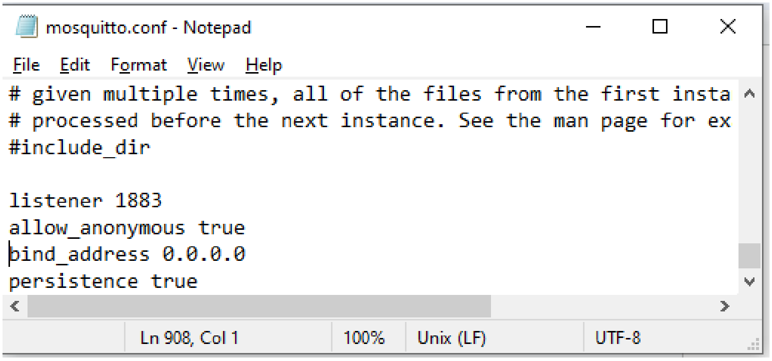
Select components to install: [✓] Files [✓] Service

Description: Position your mouse over a component to see its description.

Space required: 511.0KB

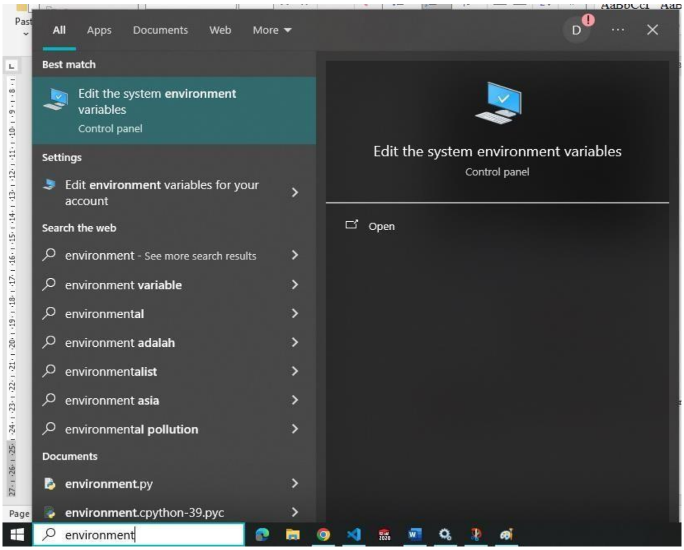
Nullsoft Install System v3.01

3. Konfigurasi broker MQTT pada file **mosquitto.conf** untuk menambahkan port, seperti pada gambar dibawah ini

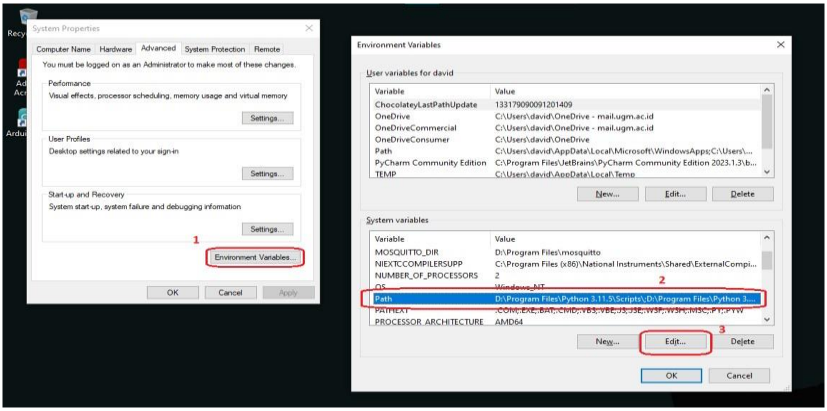


```
# given multiple times, all of the files from the first insta
# processed before the next instance. See the man page for ex
#include_dir

listener 1883
allow_anonymous true
bind_address 0.0.0.0
persistence true
```

4. Periksa Tambahkan mosquitto broker ke path. Pada sistem operasi Windows, buka search box dan cari Environment Variables



5. Pilih Environment Variables. Pada bagian systems variables cari dan pilih variable Path lalu klik tombol edit. Ikuti seperti langkah-langkah yang tertera pada gambar berikut :



You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage and virtual memory

User Profiles
Desktop settings related to your sign-in

Start-up and Recovery
System start-up, system failure and debugging information

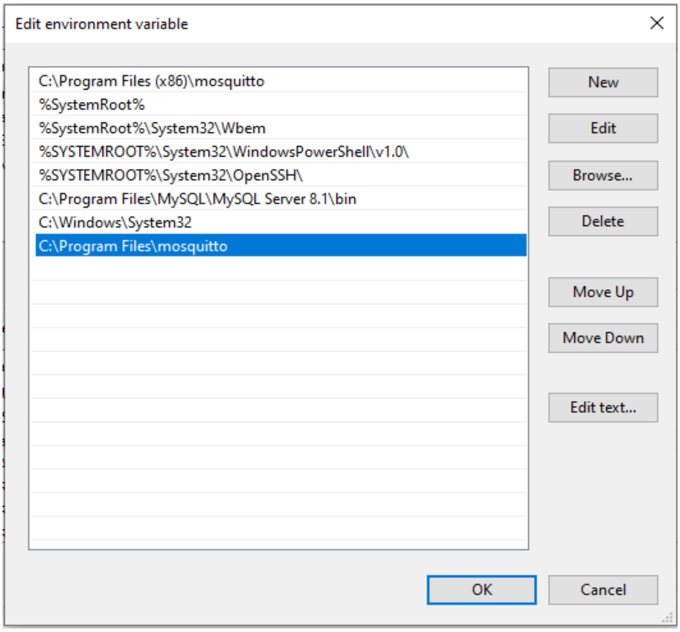
User variables for david

Variable	Value
ChocolateyLastPathUpdate	133179090091201409
OneDrive	C:\Users\david\OneDrive - mail.ugm.ac.id
OneDriveCommercial	C:\Users\david\OneDrive - mail.ugm.ac.id
OneDriveConsumer	C:\Users\david\OneDrive
Path	C:\Users\david\AppData\Local\Microsoft\WindowsApps;C:\Users\...
PyCharm Community Edition	C:\Program Files\JetBrains\PyCharm Community Edition 2023.1.3\b...
TEMP	C:\Users\david\AppData\Local\Temp

System variables

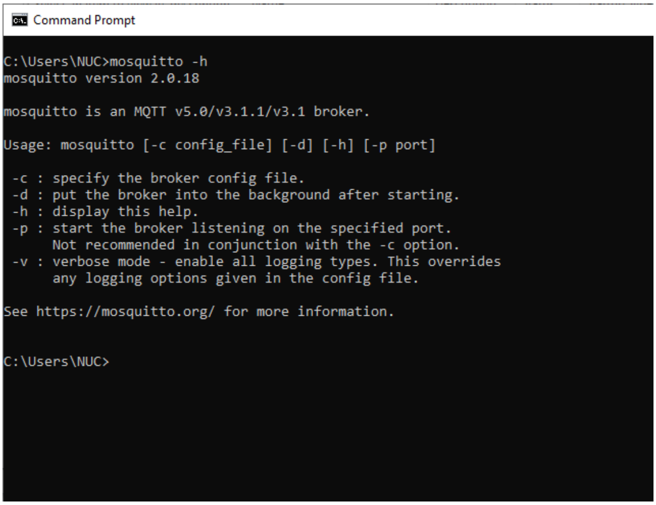
Variable	Value
MOSQUITTO_DIR	D:\Program Files\mosquitto
NIEXTCCOMPILERSUPP	C:\Program Files (x86)\National Instruments\Shared\ExternalCompi...
NUMBER_OF_PROCESSORS	2
OS	Windows_NT
Path	D:\Program Files\Python 3.11.5\Scripts\;D:\Program Files\Python 3....
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTURE	AMD64

6. Untuk Tambahkan Path directory bin mosquitto dengan menekan tombol "New" (sesuaikan dengan folder instalasi)

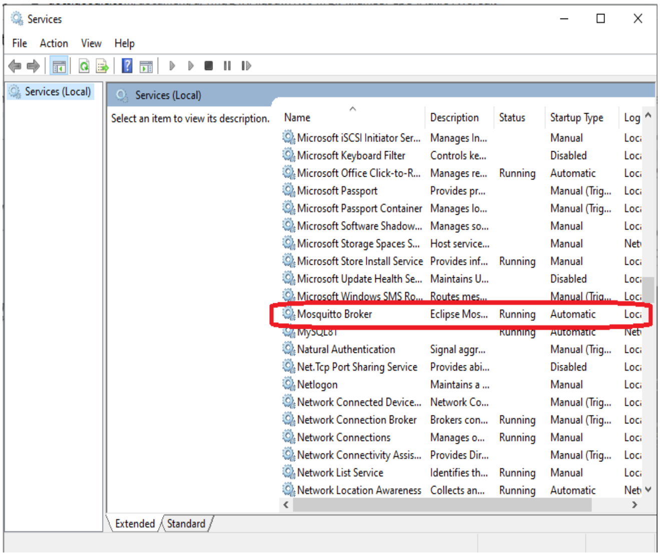


C:\Program Files (x86)\mosquitto
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\MySQL\MySQL Server 8.1\bin
C:\Windows\System32
C:\Program Files\mosquitto

7. Untuk langkah terakhir, verifikasi broker mqtt mosquitto dengan cara melakukan perintah mosquitto –h pada Command Prompt atau dapat dilihat status pada Search-Services seperti kedua gambar dibawah ini


Command Prompt

```
C:\Users\NUC>mosquitto -h
mosquitto version 2.0.18

mosquitto is an MQTT v5.0/v3.1.1/v3.1 broker.

Usage: mosquitto [-c config_file] [-d] [-h] [-p port]

 -c : specify the broker config file.
 -d : put the broker into the background after starting.
 -h : display this help.
 -p : start the broker listening on the specified port.
      Not recommended in conjunction with the -c option.
 -v : verbose mode - enable all logging types. This overrides
      any logging options given in the config file.

See https://mosquitto.org/ for more information.


C:\Users\NUC>
```



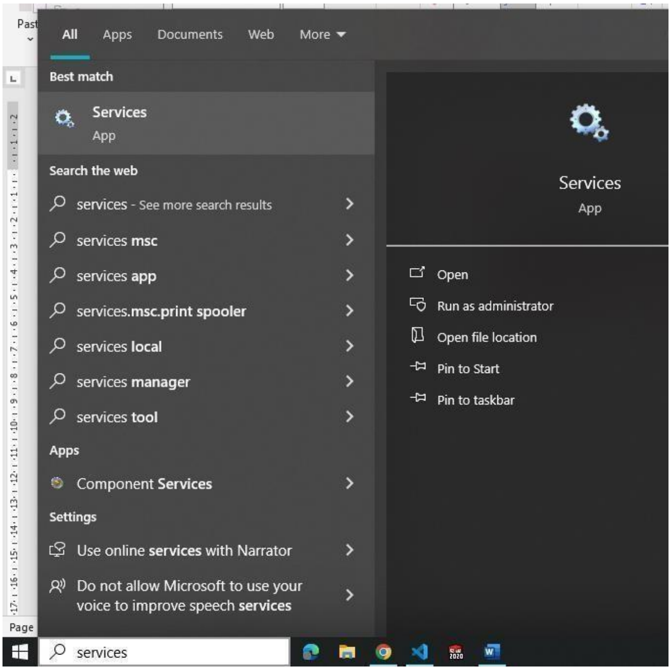
Services (Local)

Select an item to view its description.

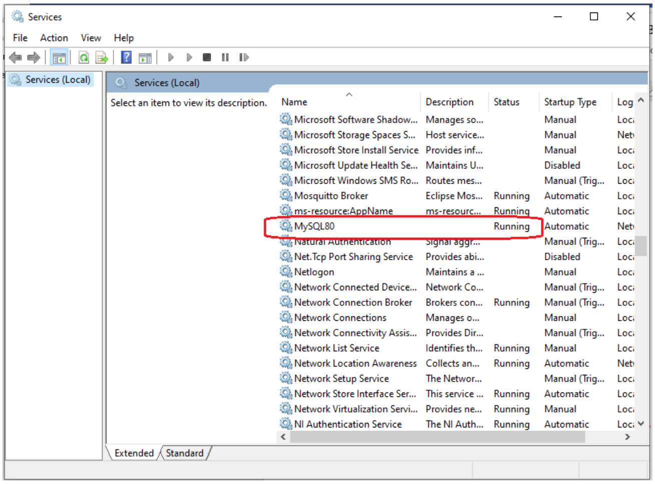
Name	Description	Status	Startup Type	Log
Microsoft iSCSI Initiator Ser...	Manages In...		Manual	Loc
Microsoft Keyboard Filter	Controls ke...		Disabled	Loc
Microsoft Office Click-to-R...	Manages re...	Running	Automatic	Loc
Microsoft Passport	Provides pr...		Manual (Trig...	Loc
Microsoft Passport Container	Manages lo...		Manual (Trig...	Loc
Microsoft Software Shadow...	Manages so...		Manual	Loc
Microsoft Storage Spaces S...	Host service...		Manual	Net
Microsoft Store Install Service	Provides inf...	Running	Manual	Loc
Microsoft Update Health Se...	Maintains U...		Disabled	Loc
Microsoft Windows SMS Ro...	Routes mes...		Manual (Trig...	Loc
Mosquitto Broker	Eclipse Mos...	Running	Automatic	Loc
MySQL81		Running	Automatic	Net
Natural Authentication	Signal aggr...		Manual (Trig...	Loc
Net.Tcp Port Sharing Service	Provides abi...		Disabled	Loc
Netlogon	Maintains a ...		Manual	Loc
Network Connected Device...	Network Co...		Manual (Trig...	Loc
Network Connection Broker	Brokers con...	Running	Manual (Trig...	Loc
Network Connections	Manages o...	Running	Manual	Loc
Network Connectivity Assis...	Provides Dir...		Manual (Trig...	Loc
Network List Service	Identifies th...	Running	Manual	Loc
Network Location Awareness	Collects an...	Running	Automatic	Net

Installing MySQL

1. Download MySQL sesuai dengan sistem operasi yang digunakan pada website MySQL (https://dev.mysql.com/downloads/installer)
2. Buka Installer dengan double-klik pada file installer yang telah diunduh untuk memulai proses instalasi.
3. Start MySQL Server, pada sebagian besar system, server akan menyala omomatis (Pada system operasi windows, kita bisa mengecheck server melalui Search-Services) seperti gambar dibawah



4. Lalu check status MySQL seperti gambar berikut ini



Services

File Action View Help

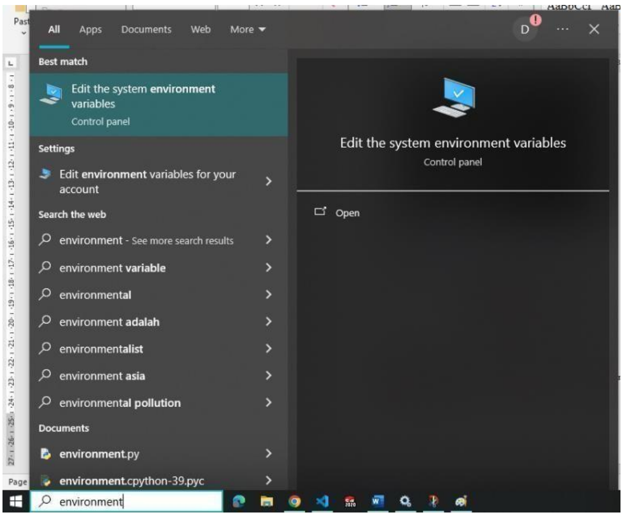
Services (Local)

Select an item to view its description.

Name	Description	Status	Startup Type	Log
Microsoft Software Shadow...	Manages so...		Manual	Loc
Microsoft Storage Spaces S...	Host service...		Manual	Net
Microsoft Store Install Service	Provides inf...		Manual	Loc
Microsoft Update Health Se...	Maintains U...		Disabled	Loc
Microsoft Windows SMS Ro...	Routes mes...		Manual (Trig...	Loc
Mosquitto Broker	Eclipse Mos...	Running	Automatic	Loc
ms-resource:AppName	ms-resourc...	Running	Automatic	Loc
MySQL80		Running	Automatic	Net
Natural Authentication	Signal aggr...		Manual (Trig...	Loc
Net.Tcp Port Sharing Service	Provides abi...		Disabled	Loc
Netlogon	Maintains a ...		Manual	Loc
Network Connected Device...	Network Co...		Manual (Trig...	Loc
Network Connection Broker	Brokers con...	Running	Manual (Trig...	Loc
Network Connections	Manages o...		Manual	Loc
Network Connectivity Assis...	Provides Dir...		Manual (Trig...	Loc
Network List Service	Identifies th...	Running	Manual	Loc
Network Location Awareness	Collects an...	Running	Automatic	Net
Network Setup Service	The Networ...		Manual (Trig...	Loc
Network Store Interface Ser...	This service ...	Running	Automatic	Loc
Network Virtualization Servi...	Provides ne...	Running	Manual	Loc
NI Authentication Service	The NI Auth...	Running	Automatic	Loc

Extended / Standard

5. Tambahkan MySQL ke path. Pada system operasi Windows,buka search box dan cari Environment Variables, seperti gambar dibawah ini



All Apps Documents Web More

Best match

Edit the system **environment** variables
Control panel

Settings

Edit **environment** variables for your account

Search the web

environment - See more search results

environment **variable**

environment**al**

environment **adalah**

environment**alist**

environment **asia**

environment**al pollution**

Documents

environment.py

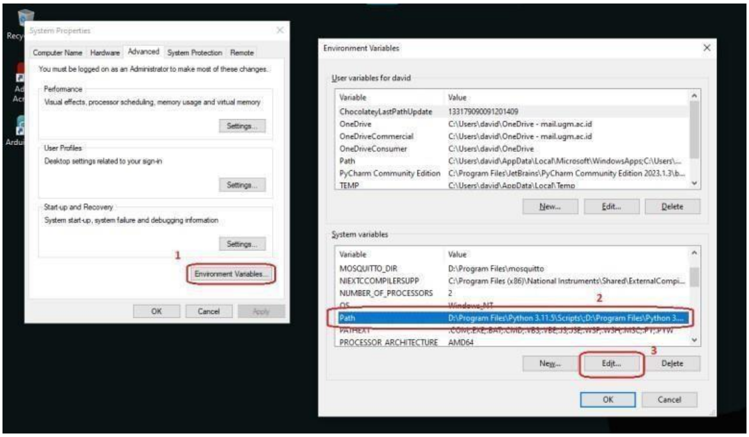
environment.cpython-39.pyc

Edit the system environment variables
Control panel

Open

environment

6. Pilih Environment Variables, pada bagian system variables cari dan pilih variable Path lalu klik tombol edit. Ikuti langkah-langkah pada gambar dibawah ini



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage and virtual memory

User Profiles
Desktop settings related to your sign-in

Start-up and Recovery
System start-up, system failure and debugging information

Environment Variables

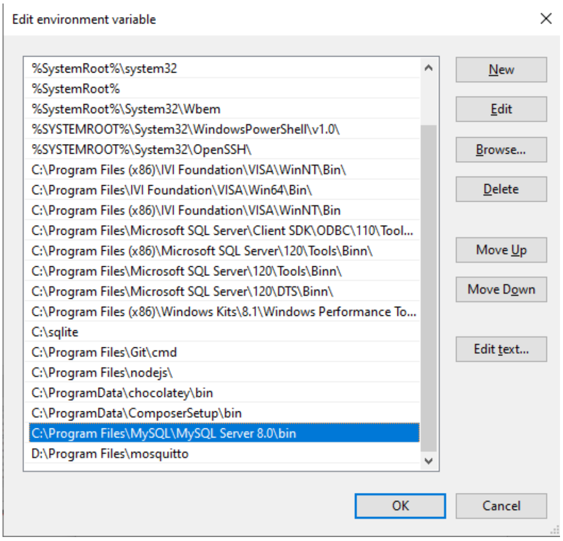
User variables for david

Variable	Value
ChocolateyLastPathUpdate	133179090091201409
OneDrive	C:\Users\david\OneDrive - mail.ugm.ac.id
OneDriveCommercial	C:\Users\david\OneDrive - mail.ugm.ac.id
OneDriveConsumer	C:\Users\david\OneDrive
Path	C:\Users\david\AppData\Local\Microsoft\WindowsApps;C:\Users\...
PyCharm Community Edition	C:\Program Files\JetBrains\PyCharm Community Edition 2023.1.3\b...
TEMP	C:\Users\david\AppData\Local\Temp

System variables

Variable	Value
MOSQUITTO_DIR	D:\Program Files\mosquitto
NIEXTCCOMPILERSUPP	C:\Program Files (x86)\National Instruments\Shared\ExternalCompi...
NUMBER_OF_PROCESSORS	2
OS	Windows_NT
Path	D:\Program Files\Python 3.11.5\Scripts\;D:\Program Files\Python 3....
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTURE	AMD64

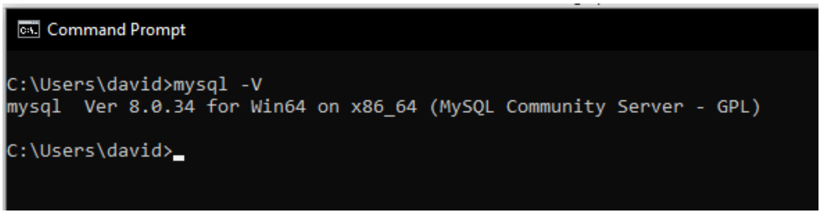
7. Tambahkan Path Directory bin MySQL dengan menekan tombol "New" (sesuaikan dengan folder instalasi seperti pada gambar)



Edit environment variable

%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\Bin\
C:\Program Files\IVI Foundation\VISA\Win64\Bin\
C:\Program Files (x86)\IVI Foundation\VISA\WinNT\Bin
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\110\Tool...
C:\Program Files (x86)\Microsoft SQL Server\120\Tools\Binn\
C:\Program Files\Microsoft SQL Server\120\Tools\Binn\
C:\Program Files\Microsoft SQL Server\120\DTS\Binn\
C:\Program Files (x86)\Windows Kits\8.1\Windows Performance To...
C:\sqlite
C:\Program Files\Git\cmd
C:\Program Files\nodejs\
C:\ProgramData\chocolatey\bin
C:\ProgramData\ComposerSetup\bin
C:\Program Files\MySQL\MySQL Server 8.0\bin
D:\Program Files\mosquitto

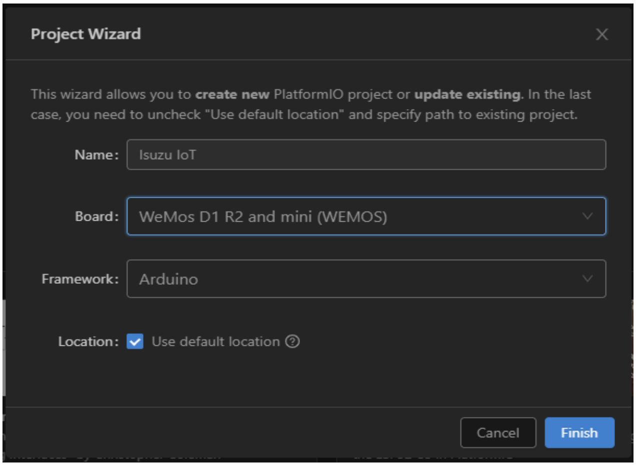
8. Verifikasi instalasi untuk memastikan MySQL telah ditambahkan ke Path dengan benar



```
C:\Users\david>mysql -V
mysql  Ver 8.0.34 for Win64 on x86_64 (MySQL Community Server - GPL)

C:\Users\david>_
```

Setup Program WeMos D1 Mini

1. Buka aplikasi Visual Studio Code. Masuk ke tab PlatformIO lalu pilih **New Project** untuk menambahkan project baru
2. Isi nama project (misal:Isuzu IoT). Lalu pilih board **WeMos D1 R2 Mini (WEMOS)**



Project Wizard

This wizard allows you to **create new** PlatformIO project or **update existing**. In the last case, you need to uncheck "Use default location" and specify path to existing project.

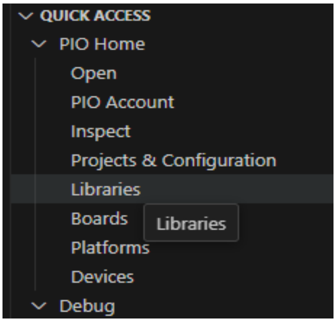
Name: Isuzu IoT

Board: WeMos D1 R2 and mini (WEMOS)

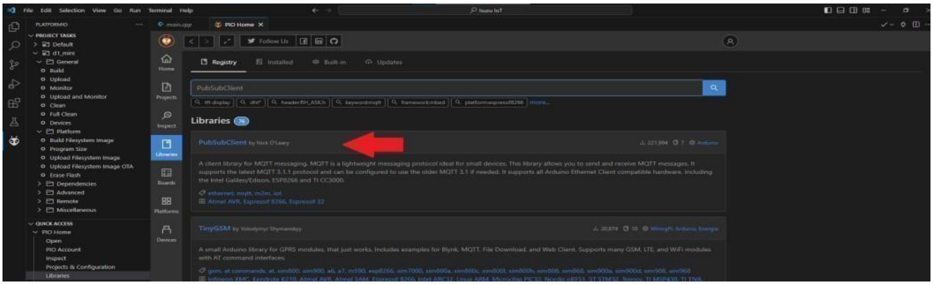
Framework: Arduino

Location: ✓ Use default location

3. Klik **Finish** dan tunggu hingga prosespembuatan project baru selesai.
4. Selanjutnya masuk kembali ke tab PlatformIO kemudian klik Libraries seperti yang terlihat pada gambar dibawah ini



5. Project IoT ini menggunakan library PubSubClient untuk komunikasi menggunakan MQTT. Untuk menginstall library tersebut, cari PunSubClient pada search box dan pilih library tersebut



PubSubClient by Nick O'Leary

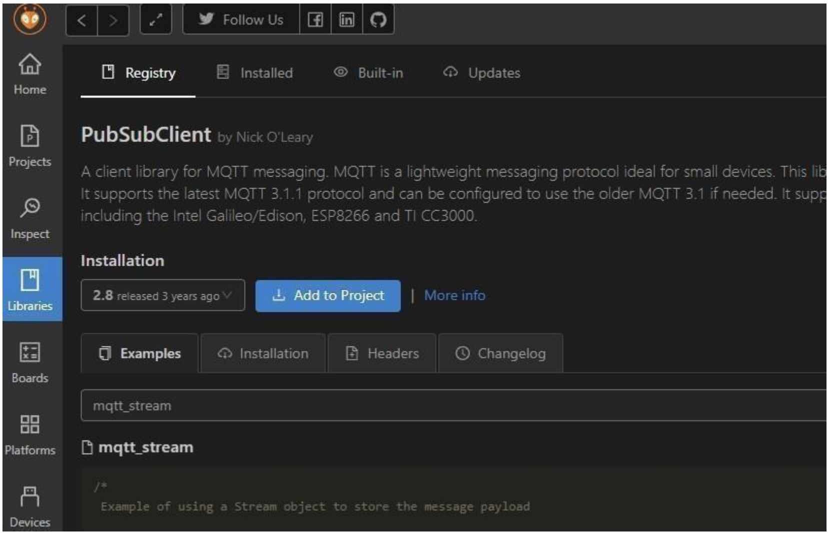
A client library for MQTT messaging. MQTT is a lightweight messaging protocol ideal for small devices. This library allows you to send and receive MQTT messages. It supports the latest MQTT 3.1.1 protocol and can be configured to use the older MQTT 3.1 if needed. It supports all Arduino Ethernet Client compatible hardware, including the Intel Galileo/Edison, ESP8266 and TI CC3000.

ethernet, mqtt, m2m, iot
Atmel AVR, Espressif 8266, Espressif 32

TinyGSM by Volodymyr Shymanskyy

A small Arduino library for GPRS modules, that just works. Includes examples for Blynk, MQTT, File Download, and Web Client. Supports many GSM, LTE, and WiFi modules with AT command interfaces.

6. Klik **Add to Project** dan pilih project yang digunakan seperti pada kedua gambarberikut:

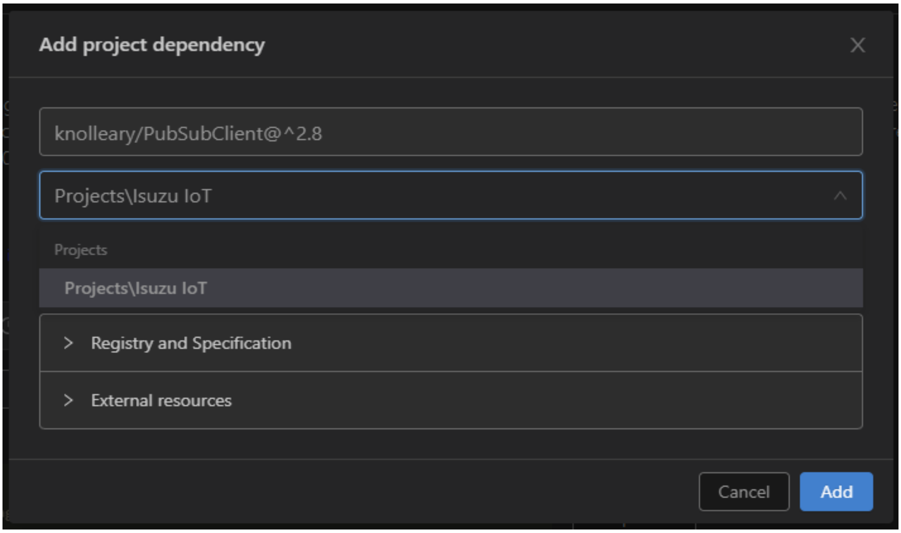


PubSubClient by Nick O'Leary

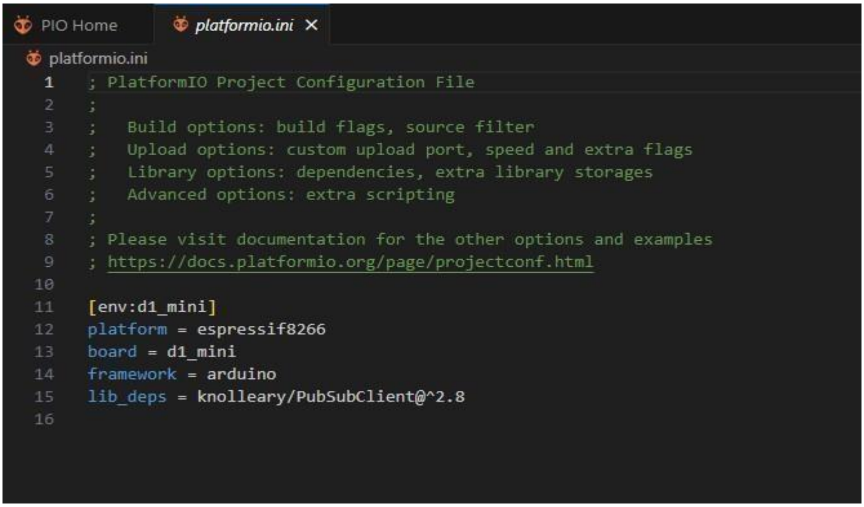
A client library for MQTT messaging. MQTT is a lightweight messaging protocol ideal for small devices. This lib
It supports the latest MQTT 3.1.1 protocol and can be configured to use the older MQTT 3.1 if needed. It supp
including the Intel Galileo/Edison, ESP8266 and TI CC3000.

mqtt_stream

```
/*
 Example of using a Stream object to store the message payload
```

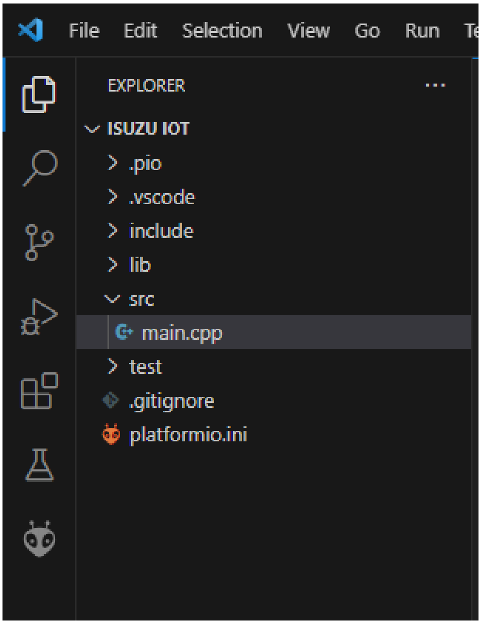


7. Jika berhasil, isi dari file **platformio.ini** akan berubah seperti gambar dibawah ini



```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra flags
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and examples
; https://docs.platformio.org/page/projectconf.html

[env:d1_mini]
platform = espressif8266
board = d1_mini
framework = arduino
lib_deps = knolleary/PubSubClient@^2.8

```

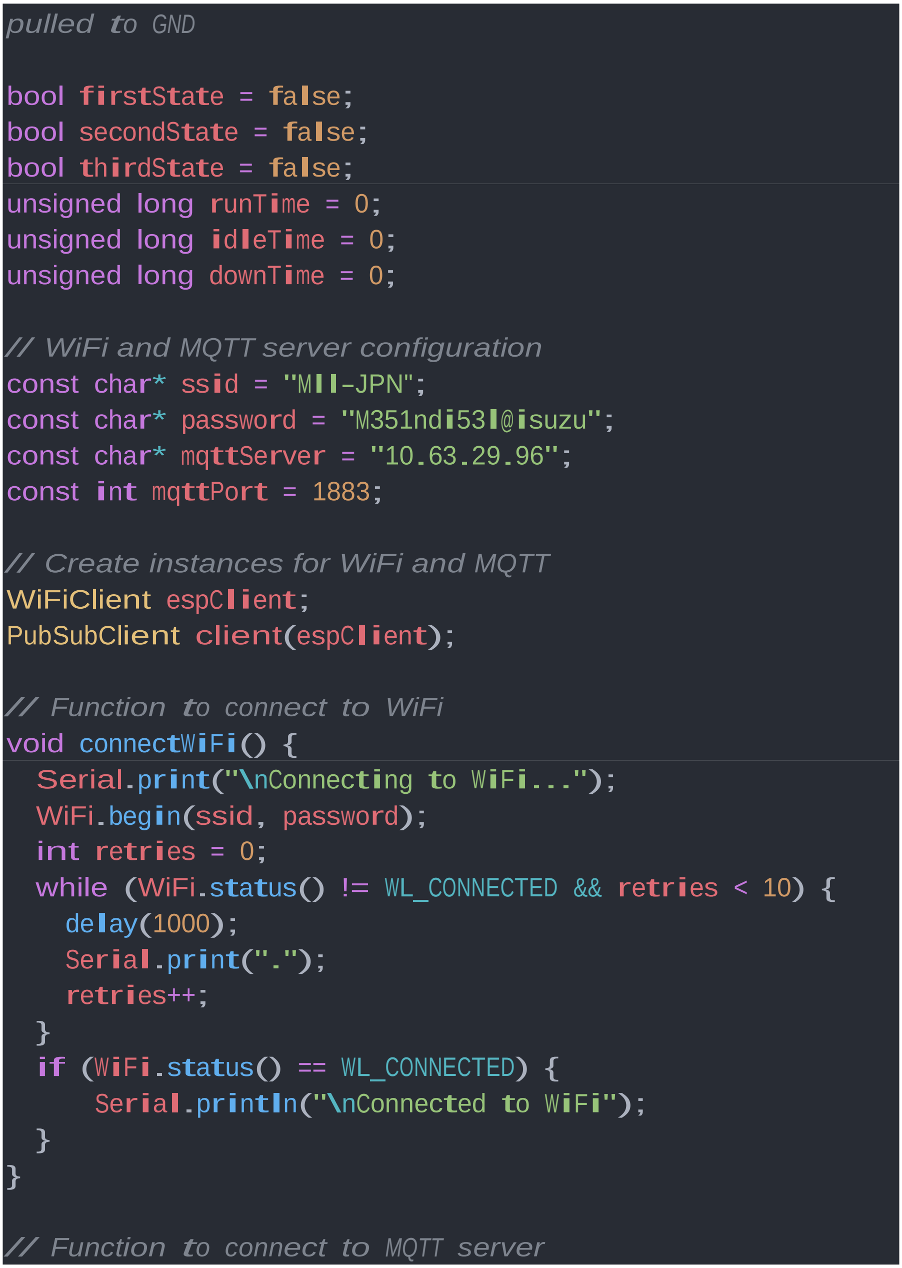
8. Setelah Kembali pada tab Explorer, masuk kefolder "src" kemudian buka file **main.cpp.**



9. Replace code yang ada didalam file main.cpp dengancode dibawah ini

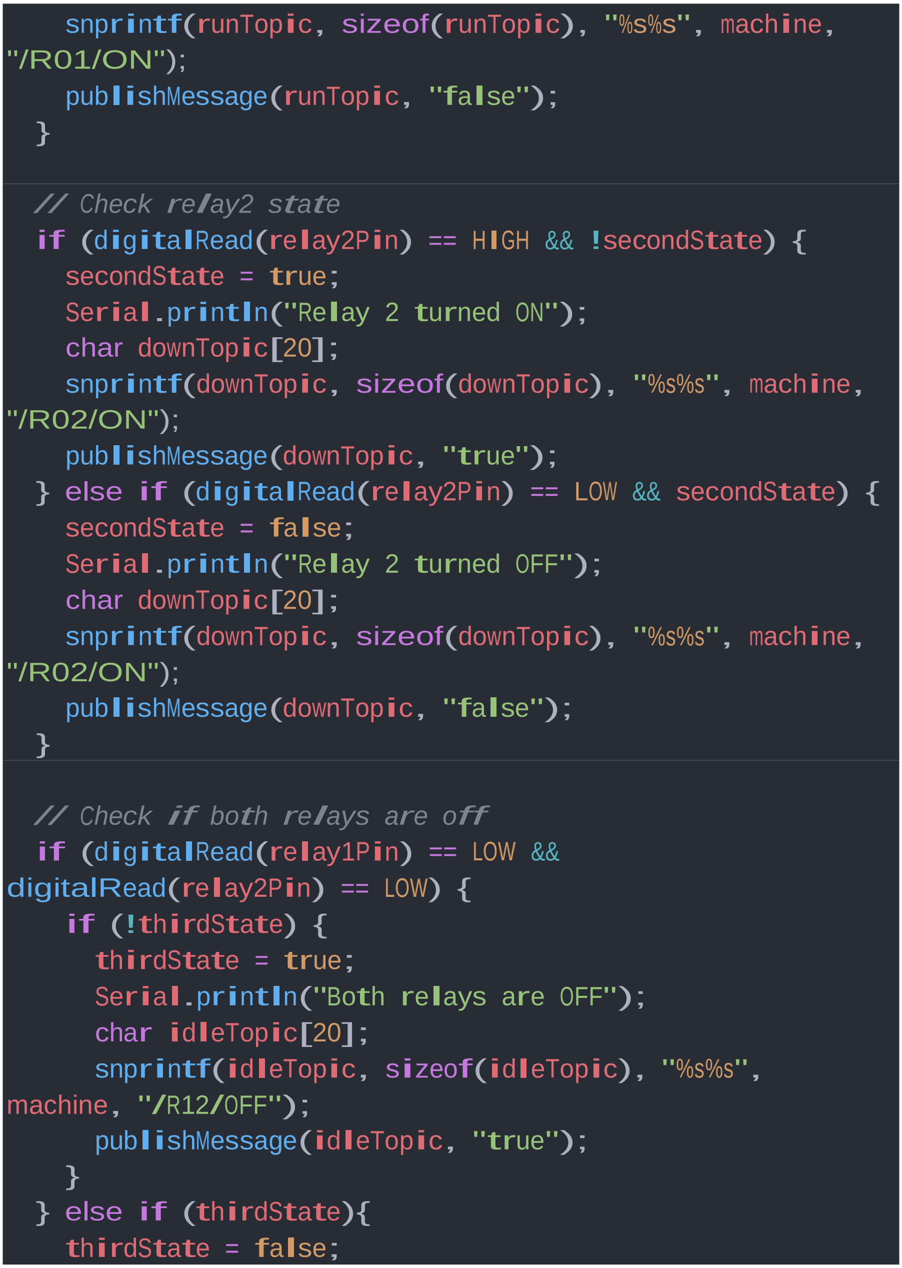
```
    // WEMOS BCR

#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* machine = "fanuc";

const int relay1Pin = D0;           // only GPIO 16 (D0) can
set to INPUT_PULLDOWN
const int relay2Pin = D8;           // input of D8 always
```

PEMBUATAN



```
pulled to GND

bool firstState = false;
bool secondState = false;
bool thirdState = false;
unsigned long runTime = 0;
unsigned long idleTime = 0;
unsigned long downTime = 0;

// WiFi and MQTT server configuration
const char* ssid = "MII-JPN";
const char* password = "M351ndi53l@isuzu";
const char* mqttServer = "10.63.29.96";
const int mqttPort = 1883;

// Create instances for WiFi and MQTT
WiFiClient espClient;
PubSubClient client(espClient);

// Function to connect to WiFi
void connectWiFi() {
  Serial.print("\nConnecting to WiFi...");
  WiFi.begin(ssid, password);
  int retries = 0;
  while (WiFi.status() != WL_CONNECTED && retries < 10) {
    delay(1000);
    Serial.print(".");
    retries++;
  }
  if (WiFi.status() == WL_CONNECTED) {
      Serial.println("\nConnected to WiFi");
  }
}

// Function to connect to MQTT server
```

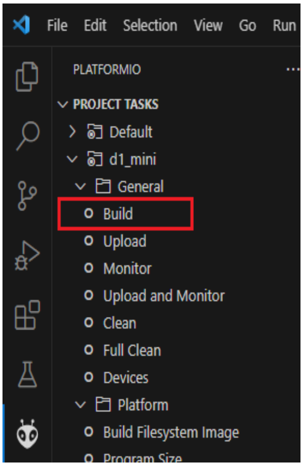
```
void connectMQTT() {
  Serial.println("Connecting to MQTT...");
  client.setServer(mqttServer, mqttPort);
  while (!client.connected()) {
    if (client.connect("ESP8266Client")) {
        Serial.println("Connected to MQTT");
    } else {
        Serial.print("Failed to connect to MQTT: ");
        Serial.println(client.state());
        delay(5000); // Wait 5 seconds before retrying
    }
  }
}

void reconnect() {
  while (!client.connected()) {
    Serial.println("Connecting to MQTT...");
    if (client.connect("ESP8266Client")) {
      Serial.println("Connected to MQTT");
    } else {
      delay(1000);
    }
  }
}

void publishMessage(const char* topic, const char* payload)
{
  if (!client.publish(topic, payload)) {
    Serial.println("Failed to publish message");
  }
}

// Function to format time into HH:MM:SS format
String formatTime(unsigned long milliseconds) {
    unsigned long totalSeconds = milliseconds / 1000;
    int hours = totalSeconds / 3600;
    int minutes = (totalSeconds % 3600) / 60;
    int seconds = totalSeconds % 60;

    char buffer[16];
    snprintf(buffer, sizeof(buffer), "%02d:%02d:%02d",
hours, minutes, seconds);
    return String(buffer);
}

void setup() {
  Serial.begin(9600);
  connectWiFi();
  connectMQTT();
  pinMode(relay1Pin, INPUT_PULLDOWN_16);
  pinMode(relay2Pin, INPUT);
}

void loop() {

  // Check relay1 state
  if (digitalRead(relay1Pin) == HIGH && !firstState) {
    if (digitalRead(relay2Pin) == LOW) {
      firstState = true;
      Serial.println("Relay 1 turned ON");
      char runTopic[20];
      snprintf(runTopic, sizeof(runTopic), "%s%s", machine,
"/R01/ON");
      publishMessage(runTopic, "true");
    }
  } else if ((digitalRead(relay1Pin) == LOW ||
digitalRead(relay2Pin) == HIGH) && firstState) {
    firstState = false;
    Serial.println("Relay 1 turned OFF");
    char runTopic[20];
```

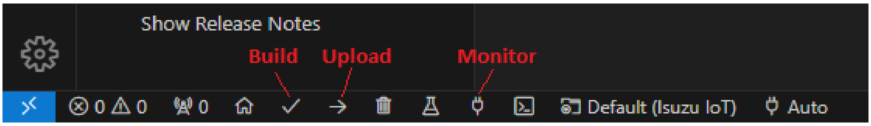


```
    snprintf(runTopic, sizeof(runTopic), "%s%s", machine,
"/R01/ON");
    publishMessage(runTopic, "false");
  }

  // Check relay2 state
  if (digitalRead(relay2Pin) == HIGH && !secondState) {
    secondState = true;
    Serial.println("Relay 2 turned ON");
    char downTopic[20];
    snprintf(downTopic, sizeof(downTopic), "%s%s", machine,
"/R02/ON");
    publishMessage(downTopic, "true");
  } else if (digitalRead(relay2Pin) == LOW && secondState) {
    secondState = false;
    Serial.println("Relay 2 turned OFF");
    char downTopic[20];
    snprintf(downTopic, sizeof(downTopic), "%s%s", machine,
"/R02/ON");
    publishMessage(downTopic, "false");
  }

  // Check if both relays are off
  if (digitalRead(relay1Pin) == LOW &&
digitalRead(relay2Pin) == LOW) {
    if (!thirdState) {
      thirdState = true;
      Serial.println("Both relays are OFF");
      char idleTopic[20];
      snprintf(idleTopic, sizeof(idleTopic), "%s%s",
machine, "/R12/OFF");
      publishMessage(idleTopic, "true");
    }
  } else if (thirdState){
    thirdState = false;
    char idleTopic[20];
    snprintf(idleTopic, sizeof(idleTopic), "%s%s", machine,
"/R12/OFF");
    publishMessage(idleTopic, "false");
  }

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  delay(1000);
}
```

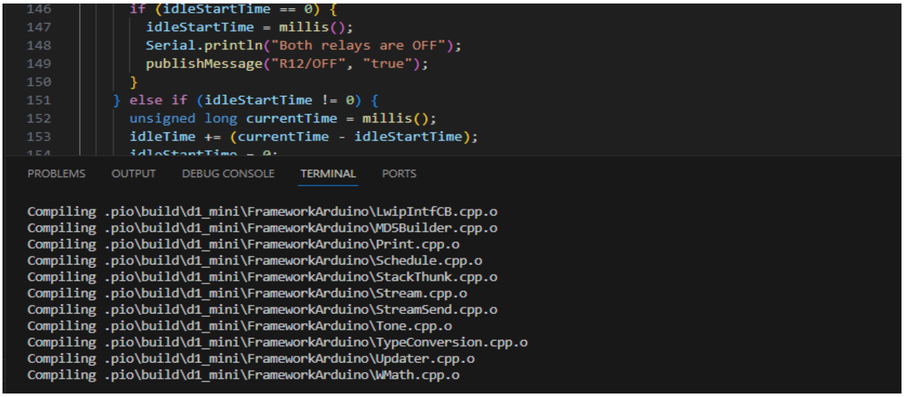
10. Karena dalam project ini mengharuskan kondisi awal relay selalu LOW, pin yang terhubung ke relay adalah D0 dan D8. Pin D0 (GPIO 16) digunakan karena hanya pin tersebut yang dapat diatur ke INPUT_PULLDOWN_16. Pin D8 digunakan karena pin tersebut selalu terhubung ke GROUND ketika tidak menerima input apapun. Jadi, penggunaan kedua pin tersebut dapat mendapatkan hasil kondisi yang akurat.

11. Setelah selesai memasukkan code, kita dapat memeriksa code tersebut dengan perintah **Build** yang terdapat pada tab PlatformIO atau icon yang terdapat dibagian bawah VSCode seperti yang terlihat pada kedua gambar berikut





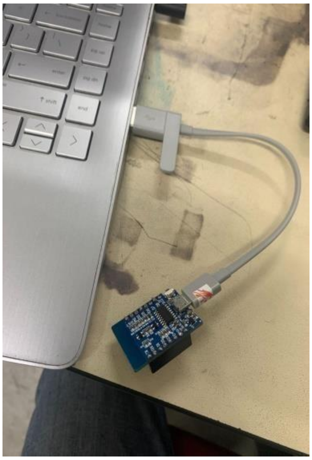
12. Untuk proses build akan terlihat seperti gambar dibawah ini



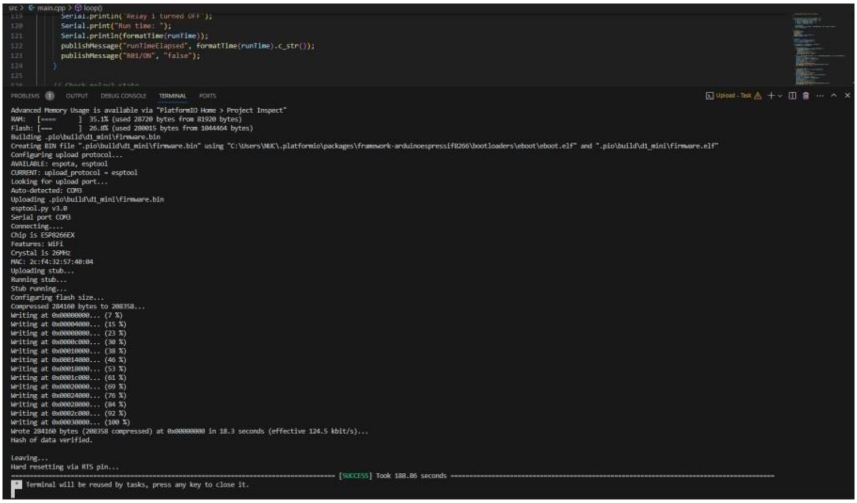
```
        if (idleStartTime == 0) {
          idleStartTime = millis();
          Serial.println("Both relays are OFF");
          publishMessage("R12/OFF", "true");
        }
      } else if (idleStartTime != 0) {
        unsigned long currentTime = millis();
        idleTime += (currentTime - idleStartTime);
```

```
Compiling .pio\build\d1_mini\FrameworkArduino\LwipIntfCB.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\MD5Builder.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\Print.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\Schedule.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\StackThunk.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\Stream.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\StreamSend.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\Tone.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\TypeConversion.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\Updater.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\WMath.cpp.o
```

14. Setelah selesai melakukan build, langkah selanjutnya yaitu **Upload** program ke perangkat WeMos D1 Mini. Sambungkan WeMos ke komputer yang digunakan dengan kabel USB, lalu klik **Upload** atau icon dibagian bawah seperti gambar pada step 11.



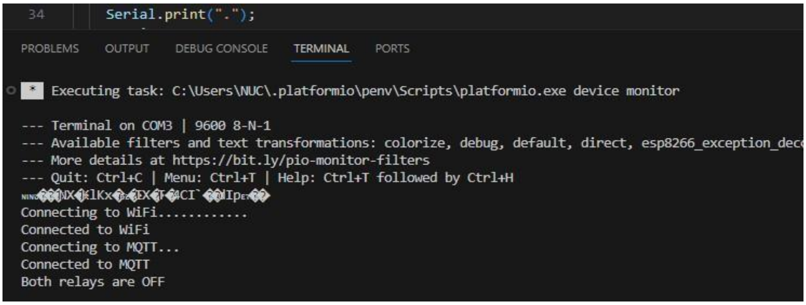
15. Proses upload akan terlihat seperti gambar dibawah ini



```
        Serial.println("Relay 1 turned OFF");
        Serial.print("Run time: ");
        Serial.println(formatTime(runTime));
        publishMessage("runTimeElapsed", formatTime(runTime).c_str());
        publishMessage("R01/ON", "false");
    }

    // Check relay2 state
```

```
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM:   [====      ]  35.1% (used 28720 bytes from 81920 bytes)
Flash: [===       ]  26.8% (used 280015 bytes from 1044464 bytes)
Building .pio\build\d1_mini\firmware.bin
Creating BIN file ".pio\build\d1_mini\firmware.bin" using "C:\Users\NUC\.platformio\packages\framework-arduinoespressif8266\bootloaders\eboot\eboot.elf" and ".pio\build\d1_mini\firmware.elf"
Configuring upload protocol...
AVAILABLE: espota, esptool
CURRENT: upload_protocol = esptool
Looking for upload port...
Auto-detected: COM3
Uploading .pio\build\d1_mini\firmware.bin
esptool.py v3.0
Serial port COM3
Connecting....
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: 2c:f4:32:57:40:84
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Compressed 284160 bytes to 208358...
Writing at 0x00000000... (7 %)
Writing at 0x00004000... (15 %)
Writing at 0x00008000... (23 %)
Writing at 0x0000c000... (30 %)
Writing at 0x00010000... (38 %)
Writing at 0x00014000... (46 %)
Writing at 0x00018000... (53 %)
Writing at 0x0001c000... (61 %)
Writing at 0x00020000... (69 %)
Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 284160 bytes (208358 compressed) at 0x00000000 in 18.3 seconds (effective 124.5 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
=========================================================== [SUCCESS] Took 188.86 seconds ===========================================================
 Terminal will be reused by tasks, press any key to close it.
```

16. Karena pada code sebelumnya sudah ditambahkan fungsi untuk menampilkan konektivitas WiFi dan MQTT pada Serial Monitor, kita dapat mengecek kondisinya dengan klik **Monitor** atau icon dibagian bawah seperti gambar pada step 11. Jika program berhasil terupload, Serial Monitor akan terlihat seperti dibawah ini.

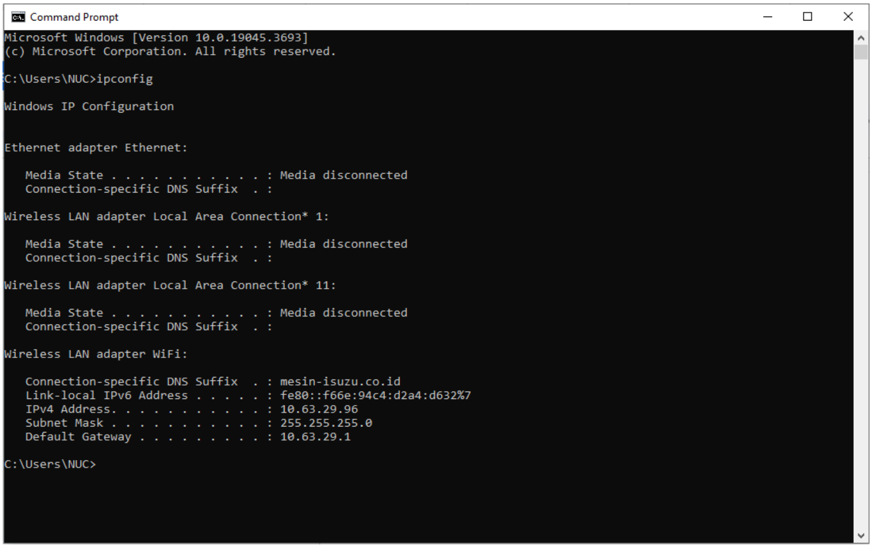


```
34    Serial.print(".");
```

```
 *  Executing task: C:\Users\NUC\.platformio\penv\Scripts\platformio.exe device monitor

--- Terminal on COM3 | 9600 8-N-1
--- Available filters and text transformations: colorize, debug, default, direct, esp8266_exception_dec
--- More details at https://bit.ly/pio-monitor-filters
--- Quit: Ctrl+C | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H
[illegible]
Connecting to WiFi............
Connected to WiFi
Connecting to MQTT...
Connected to MQTT
Both relays are OFF
```

Check IP Address

1. Buka Command Prompt
2. Ketik **ipconfig**



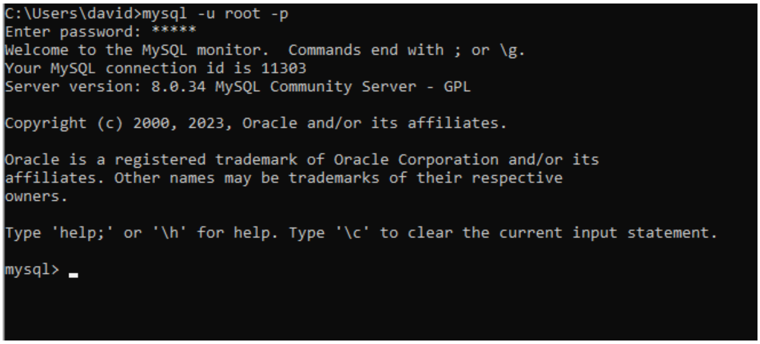
Command Prompt

```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\NUC>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 11:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter WiFi:

   Connection-specific DNS Suffix  . : mesin-isuzu.co.id
   Link-local IPv6 Address . . . . . : fe80::f66e:94c4:d2a4:d632%7
   IPv4 Address. . . . . . . . . . . : 10.63.29.96
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.63.29.1

C:\Users\NUC>
```

3. Alamat IP yang digunakan yaitu yang terlihat pada “Ipv4 Address

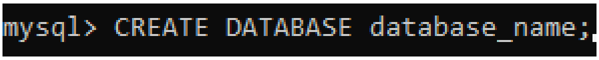
Menyiapkan Database dan Table

1. Login mysql dengan menuliskan perintah pada command prompt“mysql –u root -p”, seperti gambar dibawah ini



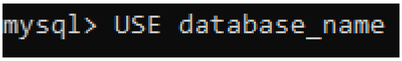
```
C:\Users\david>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11303
Server version: 8.0.34 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> _
```

2. Langkah selanjutnya adalah membuat database dengan menuliskan perintah “CREATE DATABASE database_name”. Rename “database_name” dengan nama database yang diinginkan.



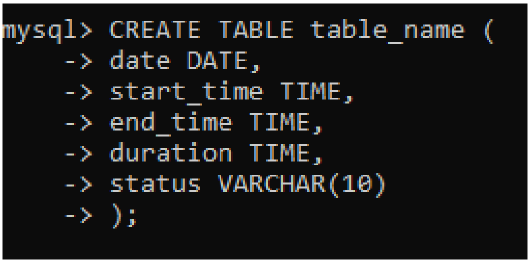
```
mysql> CREATE DATABASE database_name;
```

3. Beralih ke database yang akan digunakan dengan menuliskan perintah “USE database_name”



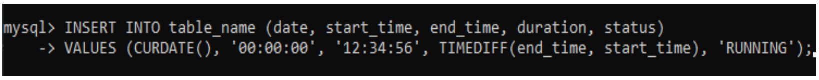
```
mysql> USE database_name
```

4. Lalu mulai untuk membuat tabel dengan menuliskan perintah seperti gambar dibawah ini dan ganti ‘’table_name” dengan tabel yang diinginkan



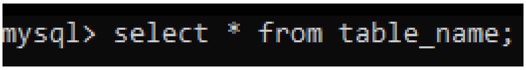
```
mysql> CREATE TABLE table_name (
    -> date DATE,
    -> start_time TIME,
    -> end_time TIME,
    -> duration TIME,
    -> status VARCHAR(10)
    -> );
```

5. Jika ingin memasukkan data secara manual (optional), tuliskan perintah seperti pada gambar dibawah ini



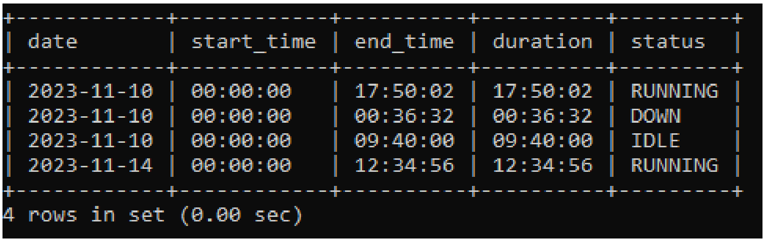
```
mysql> INSERT INTO table_name (date, start_time, end_time, duration, status)
    -> VALUES (CURDATE(), '00:00:00', '12:34:56', TIMEDIFF(end_time, start_time), 'RUNNING');
```

6. Untuk mengecheck isi tabel, tuliskan perintah seperti dibawah ini



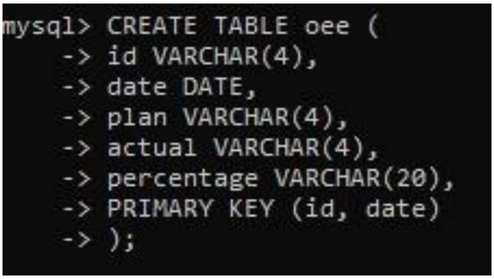
```
mysql> select * from table_name;
```

7. Dan tabel yang telah dibuat akan terlihat seperti gambar dibawah ini



```
+------------+------------+----------+----------+---------+
| date       | start_time | end_time | duration | status  |
+------------+------------+----------+----------+---------+
| 2023-11-10 | 00:00:00   | 17:50:02 | 17:50:02 | RUNNING |
| 2023-11-10 | 00:00:00   | 00:36:32 | 00:36:32 | DOWN    |
| 2023-11-10 | 00:00:00   | 09:40:00 | 09:40:00 | IDLE    |
| 2023-11-14 | 00:00:00   | 12:34:56 | 12:34:56 | RUNNING |
+------------+------------+----------+----------+---------+
4 rows in set (0.00 sec)
```

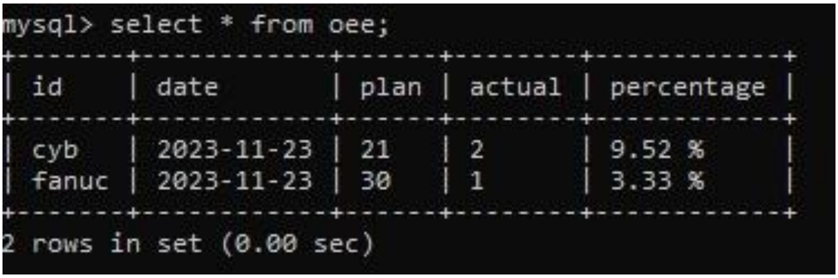
8. Selanjutnya lakukan langkah 2 dan 3 kembali lalu mulai untuk membuat tabel OEE dengan menuliskan perintah seperti gambar dibawah ini dan ganti ‘’table_OEE” dengan tabel yang diinginkan



```
mysql> CREATE TABLE oee (
    -> id VARCHAR(4),
    -> date DATE,
    -> plan VARCHAR(4),
    -> actual VARCHAR(4),
    -> percentage VARCHAR(20),
    -> PRIMARY KEY (id, date)
    -> );
```

9. Untuk mengecheck isi tabel OEE, tuliskan perintah “select * from table_OEE”

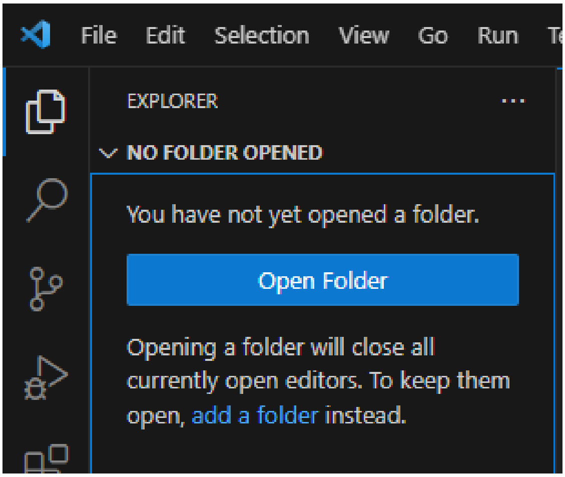
10. Dan tabel yang telah dibuat akan terlihat seperti gambar dibawah ini



```
mysql> select * from oee;
+-------+------------+------+--------+------------+
| id    | date       | plan | actual | percentage |
+-------+------------+------+--------+------------+
| cyb   | 2023-11-23 | 21   | 2      | 9.52 %     |
| fanuc | 2023-11-23 | 30   | 1      | 3.33 %     |
+-------+------------+------+--------+------------+
2 rows in set (0.00 sec)
```

Mengirim Data dari WeMos ke Database

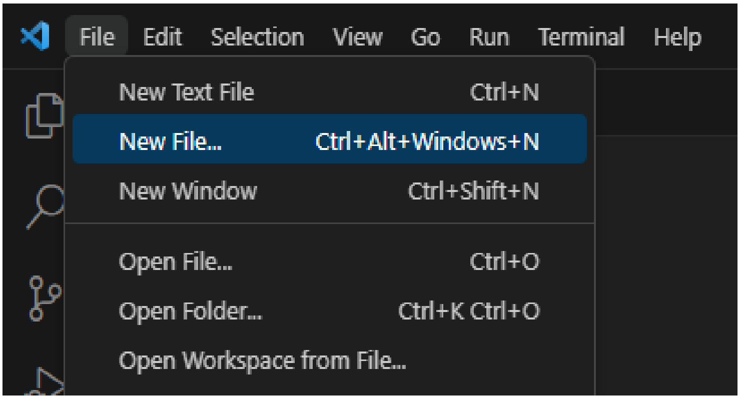
1. Buka Aplikasi text editor Visual Studio Code
2. Pada tab Explorer pilih **Open Folder** untuk menempatkan file yang akan dibuat



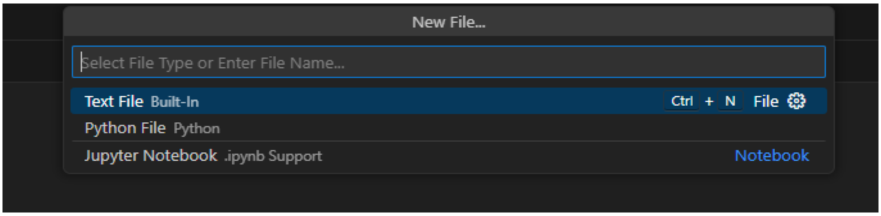
You have not yet opened a folder.

Opening a folder will close all currently open editors. To keep them open, add a folder instead.

3. Setelah folder terbuka, pilih **File – New File**



4. Pilih Python File pada option yang tersedia


New File...
Select File Type or Enter File Name...
Text File Built-In
Ctrl + N
File
Python File Python
Jupyter Notebook .ipynb Support
Notebook

5. Setelah terbuka file baru, masukkan code dibawah ini. Ganti variabel mqttServer sesuai dengan alamat IP pada komputer server. Sesuaikan juga konfigurasi database

```
import paho.mqtt.client as mqtt
import paho.mqtt.publish as publish
import mysql.connector
import threading
import time

# Machine configurations
machines = {
    "fanuc": {"cycle_time": 2.32, "topics": ["R01/ON",
"R02/ON", "R12/OFF"]}
}

mqttServer = "10.63.29.96"

db_config = {
    "host": "localhost",
    "user": "ISUZU",
    "password": "12345",
    "database": "isuzudb"
}

connection = mysql.connector.connect(**db_config)
```

PEMBUATAN

```
cursor = connection.cursor()

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))
    for machine, config in machines.items():
        for topic in config["topics"]:
            client.subscribe(f"{machine}/{topic}")

def on_message(client, userdata, msg):
    print(f"{msg.topic}: {msg.payload.decode()}")
    process_message(msg)

def process_message(msg):
    machine, topic = msg.topic.split("/", 1)
    c = msg.payload.decode()
    if topic == "R01/ON":
        if c == "true":
            qry = f"INSERT INTO {machine} (date, start_time, 
status) VALUES (CURDATE(), CURTIME(), 'RUNNING')"
        else:
            qry = f"UPDATE {machine} SET end_time = 
CURTIME(), duration = TIMEDIFF(CURTIME(), start_time) WHERE 
duration is NULL AND status = 'RUNNING'"
    elif topic == "R02/ON":
        if c == "true":
            qry = f"INSERT INTO {machine} (date, start_time, 
status) VALUES (CURDATE(), CURTIME(), 'DOWN')"
        else:
            qry = f"UPDATE {machine} SET end_time = 
CURTIME(), duration = TIMEDIFF(CURTIME(), start_time) WHERE 
duration is NULL AND status = 'DOWN'"
    elif topic == "R12/OFF":
        if c == "true":
            qry = f"INSERT INTO {machine} (date, start_time, 
status) VALUES (CURDATE(), CURTIME(), 'IDLE')"
```

PEMBUATAN

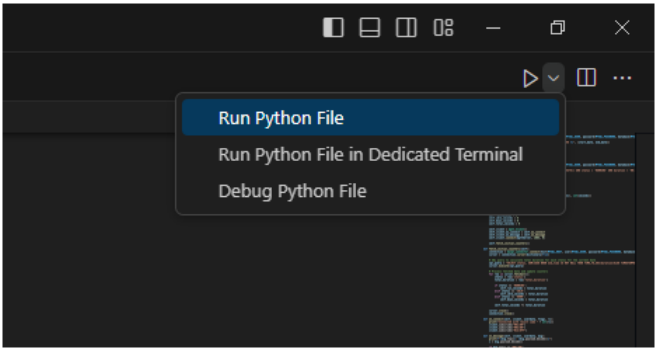
```
        else:
            qry = f"UPDATE {machine} SET end_time =
CURTIME(), duration = TIMEDIFF(CURTIME(), start_time) WHERE
duration is NULL AND status = 'IDLE'"

    cursor.execute(qry)
    connection.commit()

def mqtt_thread_func():
    client = mqtt.Client()
    client.on_connect = on_connect
    client.on_message = on_message
    client.connect(mqttServer, 1883, 0)
    client.loop_forever()

mqtt_thread = threading.Thread(target=mqtt_thread_func)
mqtt_thread.start()

seconds = 0

def calculate_oee(machine, plan_column, actual_column):
    global seconds
    config = machines[machine]
    plan = 0

    cursor.execute(f"SELECT plan FROM oee WHERE date =
CURDATE() AND id = '{machine}'")
    result = cursor.fetchone()

    if result and result[0] is not None:
        plan = int(result[0])
    else:
        plan = 0

    if seconds % int(config["cycle_time"] * 60) == 0:
```

PEMBUATAN

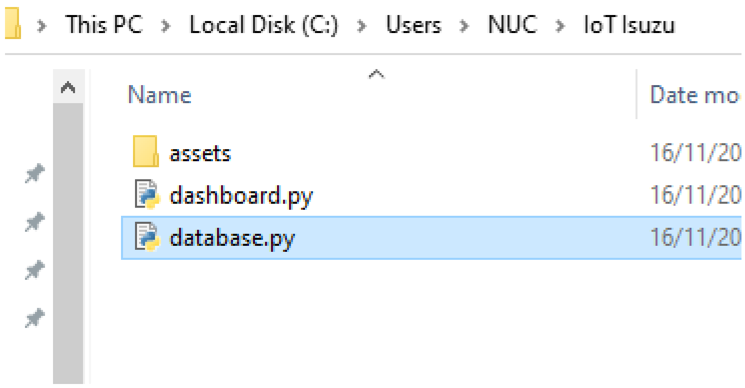
```
        plan += 1

    cursor.execute(f"SELECT COUNT(*) FROM {machine} WHERE 
date = CURDATE() AND status = 'RUNNING' AND duration IS NOT 
NULL AND duration >= '00:02:00'")
    actual = cursor.fetchone()[0]

    percent = (actual / plan) * 100 if plan > 0 else 0
    percentage = f"{percent:.2f} %"

    cursor.execute(
        "INSERT INTO oee (id, date, plan, actual, percentage) 
VALUES (%s, CURDATE(), %s, %s, %s) "
        "ON DUPLICATE KEY UPDATE actual = %s, plan = %s, 
percentage = %s",
        (machine, plan, actual, percentage, actual, plan, 
percentage)
    )

    connection.commit()

while True:
    for machine in machines.keys():
        calculate_oee(machine, f"{machine}_plan", 
f"{machine}_actual")

    current_time = int(time.strftime("%H%M"))
    current_day = int(time.strftime("%w"))

    time_periods = [
        (740, 1000),
        (1010, 1145 if current_day == 5 else 1200),
        (1300 if current_day == 5 else 1245, 1415),
        (1425, 1545 if current_day == 5 else 1615),
        (1630 if current_day == 5 else 0, 1645 if current_day
== 5 else 0)
    ]

    in_time_period = any(start <= current_time < end for
start, end in time_periods)

    if in_time_period:
        time.sleep(1)
        seconds += 1

    if ((current_time == 1615 and current_day != 5) or
        (current_time == 1645 and current_day == 5)):
        for machine, config in machines.items():
            for topic in config["topics"]:
                publish.single(f"{machine}/{topic}",
payload="false", hostname=mqttServer)

        time.sleep(60)  # Delay for a minute to avoid
publishing multiple times
```

6. Simpan file tersebut dengan cara File – Save atau CTRL + S pada keyboard. (misal : database.py)

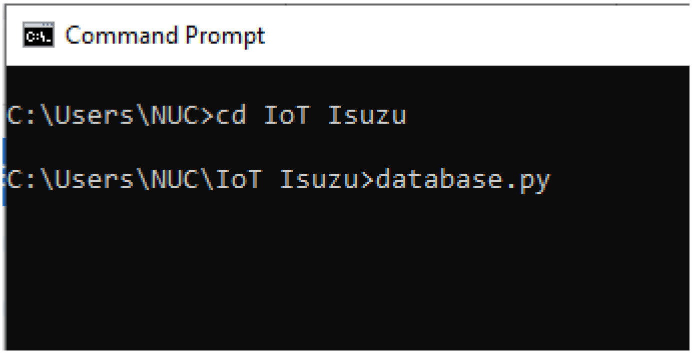
7. Simpan file tersebut dengan cara File – Save atau CTRL + S pada keyboard. (misal : database.py)
8. Langkah selanjutnya yaitu running program dashboard.py yang dapat dilakukan dengan 3 cara antara lain :
9.
   a. Menekan tombol run yang terdapat pada Visual Studio Code



   b. Buka folder menggunkaan file explorer lalu klik 2x database.py



c. Buka folder menggunakan commandprompt kemudian ketik nama file (database.py)



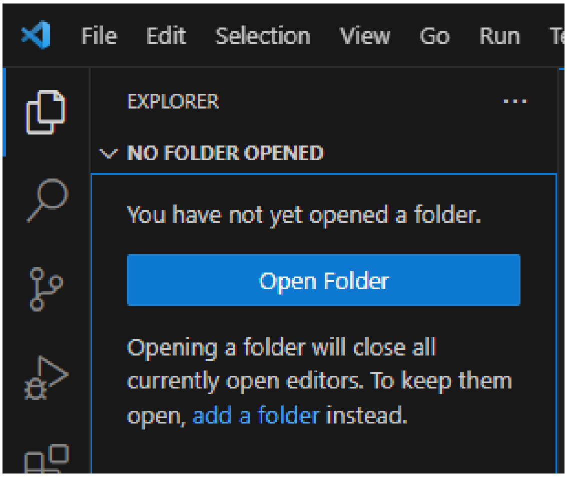
Command Prompt

```
C:\Users\NUC>cd IoT Isuzu

C:\Users\NUC\IoT Isuzu>database.py
```

10. Pastikan run program database sebelum run program dashboard.

Setup GUI

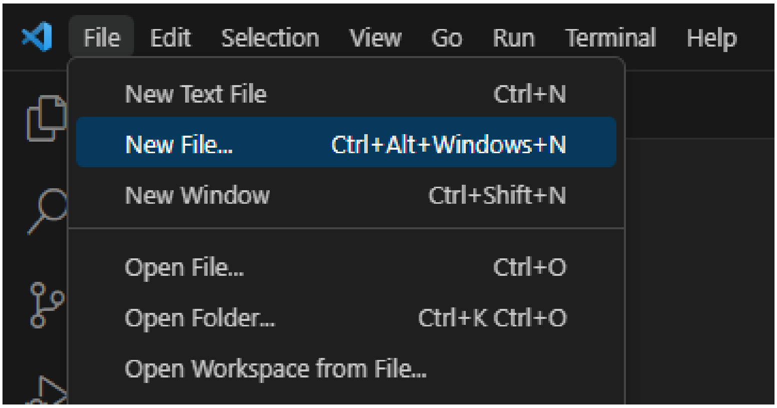
1. Buka aplikasi text editornVisual Studio Code
2. Pada tab Explorer pilih **Open Folder** untuk menempatkan file yang akan dibuat



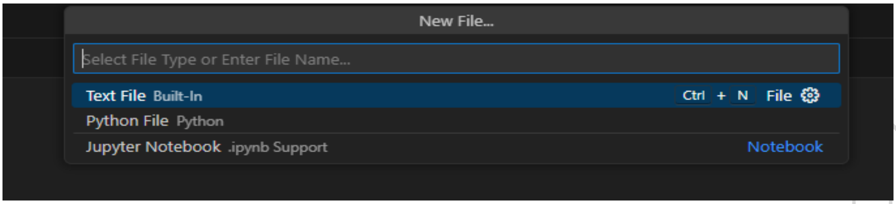
You have not yet opened a folder.

Opening a folder will close all currently open editors. To keep them open, add a folder instead.

3. Setelah membuka foldernya, pilih **File- New File**



4. Pilih Python File pada option yang tersedia



New File...

Select File Type or Enter File Name...

Text File Built-In — Ctrl + N — File

Python File Python

Jupyter Notebook .ipynb Support — Notebook

5. Setelah file baru terbuka, masukkan code dibawah ini. Ganti variable mqtt server sesuai dengan alamat IP pada Komputer server. Sesuaikan juga konfigurasi database.

```
import dash
from dash import dcc, html
from dash import dash_table as dt
from dash.dependencies import Input, Output, State
import dash_bootstrap_components as dbc
import paho.mqtt.client as mqtt
import mysql.connector
import time
import pandas as pd

machine_configs = {
    "fanuc": {"mqtt_topics": ["R01/ON", "R02/ON",
"R12/OFF"]},
    # Add configurations for other machines as needed
    "toyoda": {"mqtt_topics": ["R01/ON", "R02/ON",
"R12/OFF"]},
    "makino": {"mqtt_topics": ["R01/ON", "R02/ON",
"R12/OFF"]},
}

mqtt_server = "172.20.10.3"

mysql_config = {
```

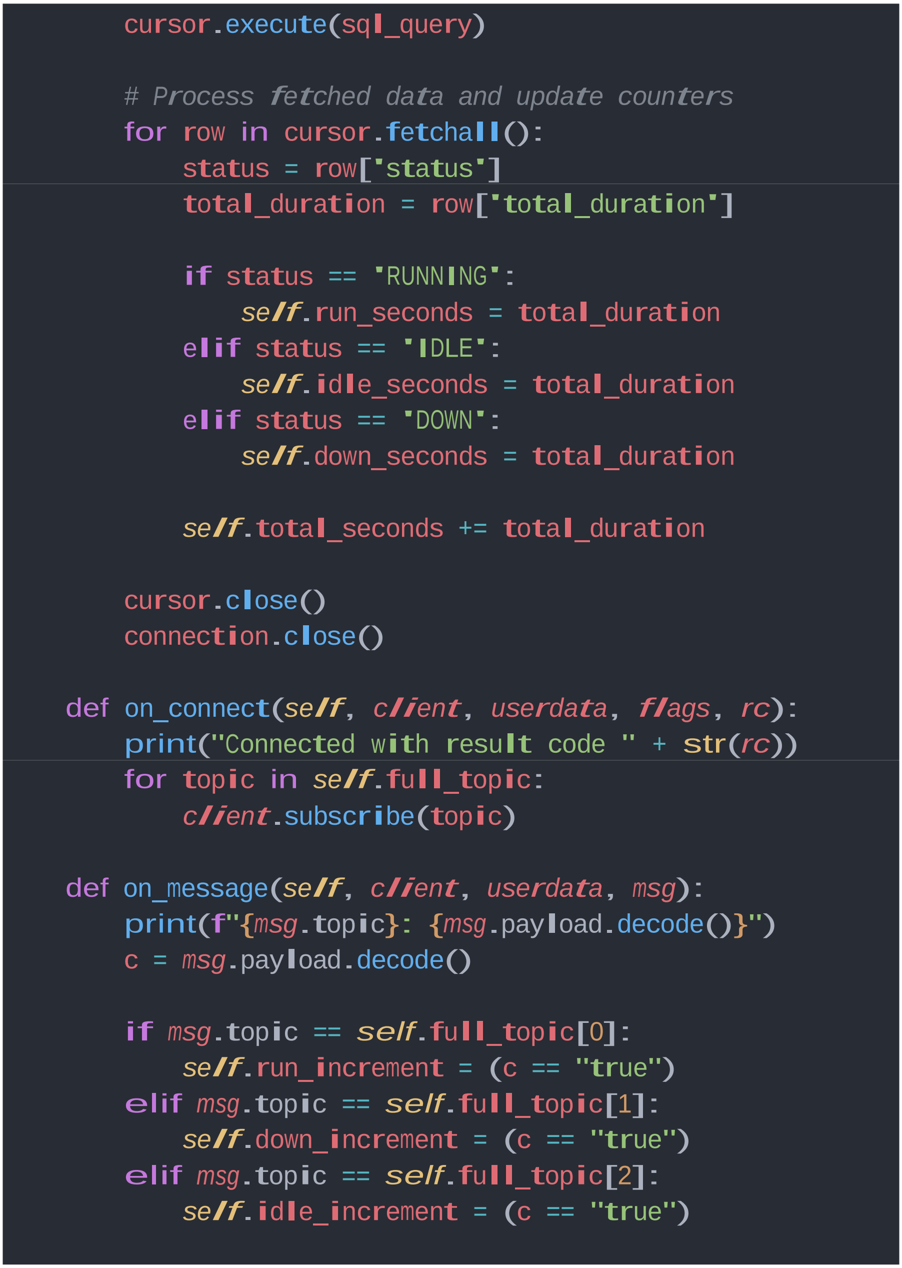
PEMBUATAN

```
    "host": "localhost",
    "user": "root",
    "password": "12345",
    "database": "relaydata",
}

app = dash.Dash(__name__, suppress_callback_exceptions=True,
external_stylesheets=[dbc.themes.BOOTSTRAP,
dbc.icons.BOOTSTRAP])

def format_time(seconds):
        hours, remainder = divmod(seconds, 3600)
        minutes, seconds = divmod(remainder, 60)
        return '{:02}:{:02}:{:02}'.format(int(hours),
int(minutes), int(seconds))

class RealTimeMonitor:
    def __init__(self, machine_name, mqtt_server,
mysql_config, mqtt_topics):
        self.machine_name = machine_name
        self.run_increment = False
        self.idle_increment = False
        self.down_increment = False
        self.run_seconds = 0
        self.idle_seconds = 0
        self.down_seconds = 0
        self.total_seconds = 0

        self.client = mqtt.Client()
        self.client.on_connect = self.on_connect
        self.client.on_message = self.on_message
        self.client.connect(mqtt_server, 1883, 0)
        self.full_topic = [f"{machine_name}/{topic}" for
topic in mqtt_topics]
        self.client.loop_start()
```

PEMBUATAN

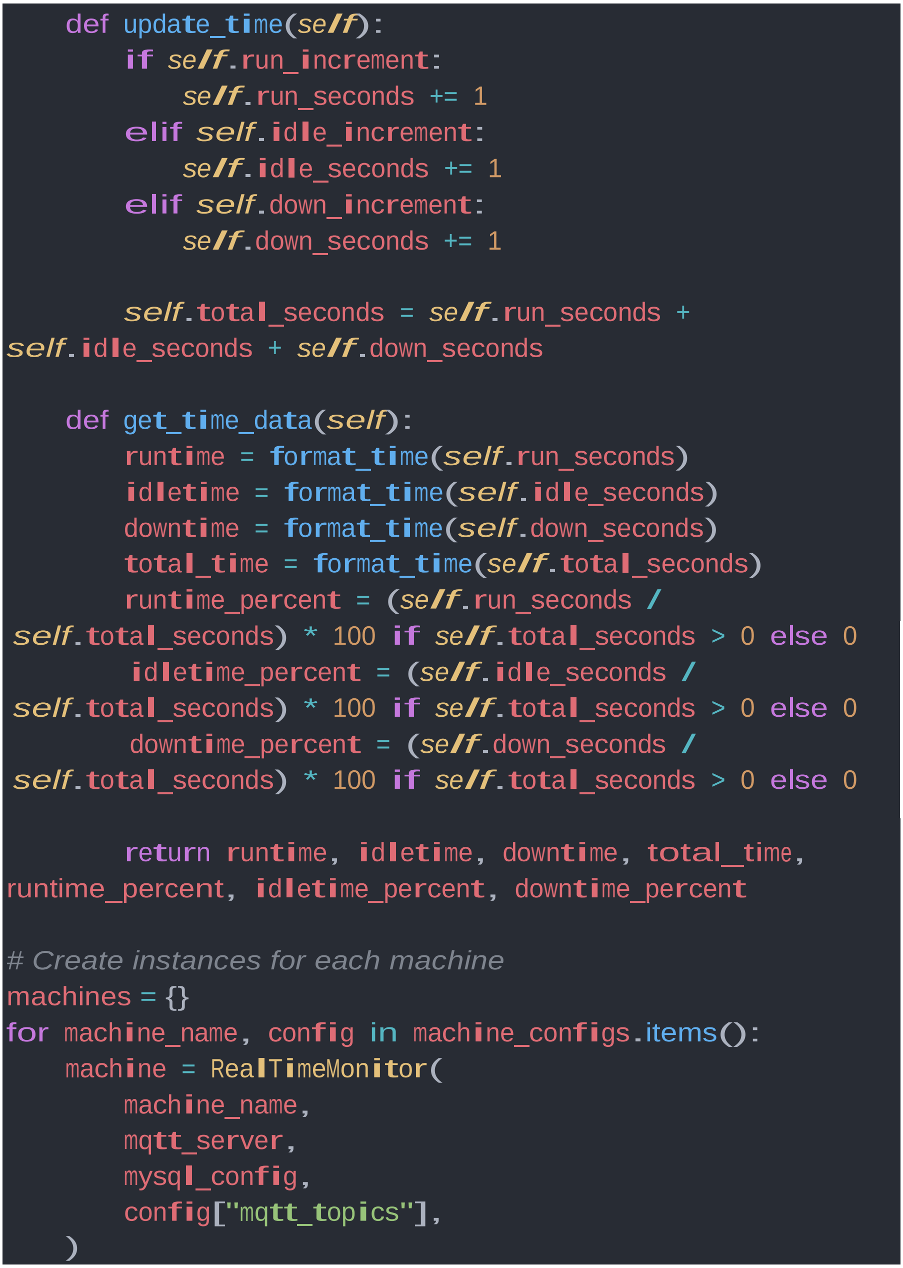
```
        self.fetch_initial_counters()

    # Fetch the required data from the MySQL database
    def fetch_data_from_mysql(self, start_date, end_date):
        connection = mysql.connector.connect(**mysql_config)
        cursor = connection.cursor()
        cursor.execute(f"SELECT * FROM {self.machine_name}
WHERE date BETWEEN %s AND %s", (start_date, end_date))
        data = cursor.fetchall()
        cursor.close()
        connection.close()
        return data

    # Fetch the oee
    def fetch_oee_data(self):
        connection = mysql.connector.connect(**mysql_config)
        cursor = connection.cursor()
        cursor.execute(f"SELECT * FROM oee WHERE date =
CURDATE() AND id = {self.machine_name}")
        data = cursor.fetchall()
        cursor.close()
        connection.close()
        return data

    def fetch_initial_counters(self):
        connection = mysql.connector.connect(**mysql_config)
        cursor = connection.cursor(dictionary=True)

        # SQL query to calculate total duration for each
status for the current date
        sql_query = f"SELECT status, SUM(CASE WHEN end_time
IS NOT NULL THEN TIME_TO_SEC(duration)ELSE
TIMESTAMPDIFF(SECOND, start_time, NOW()) END) as
total_duration FROM {self.machine_name} GROUP BY status;"
```

PEMBUATAN

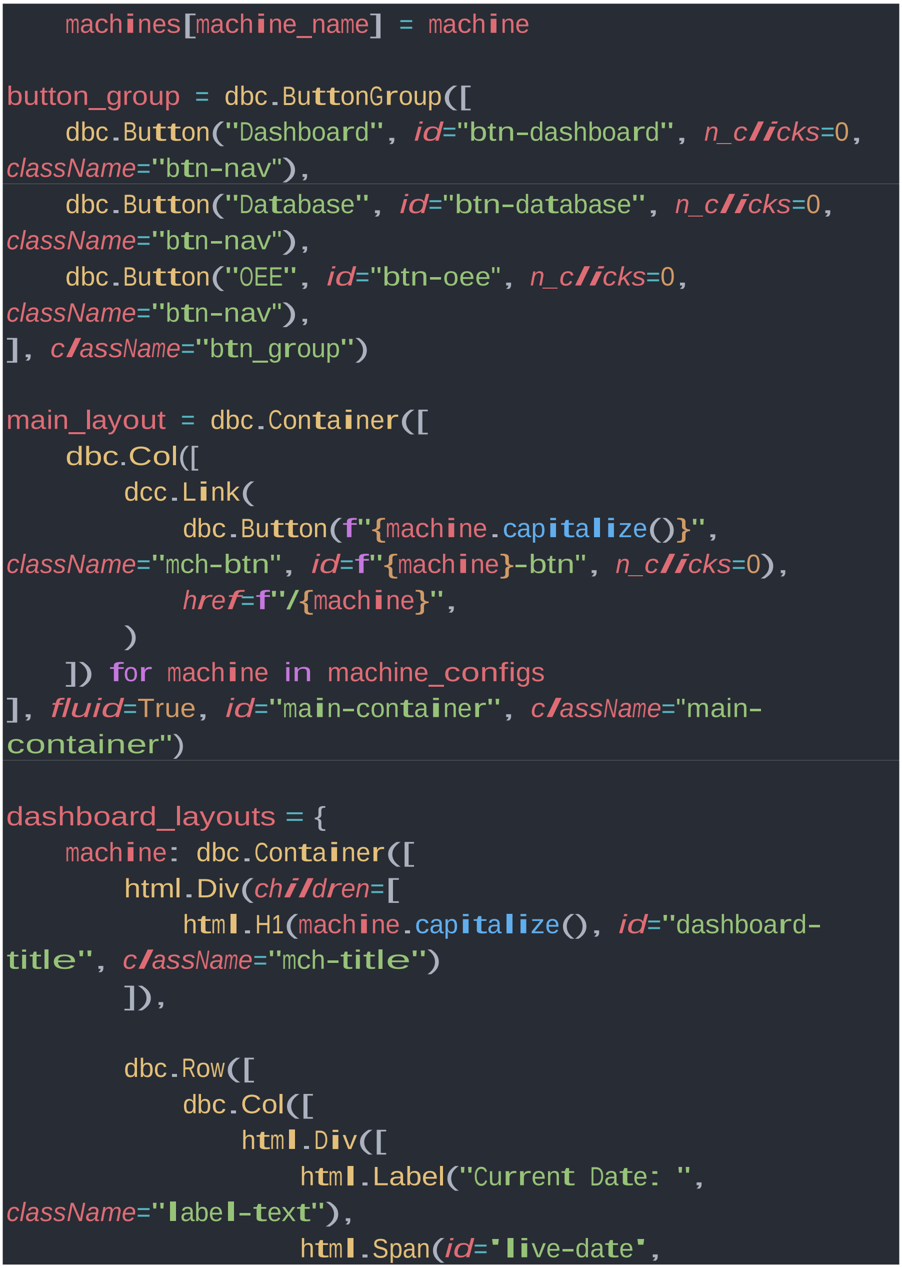


```
        cursor.execute(sql_query)

        # Process fetched data and update counters
        for row in cursor.fetchall():
            status = row['status']
            total_duration = row['total_duration']

            if status == 'RUNNING':
                self.run_seconds = total_duration
            elif status == 'IDLE':
                self.idle_seconds = total_duration
            elif status == 'DOWN':
                self.down_seconds = total_duration

            self.total_seconds += total_duration

        cursor.close()
        connection.close()

    def on_connect(self, client, userdata, flags, rc):
        print("Connected with result code " + str(rc))
        for topic in self.full_topic:
            client.subscribe(topic)

    def on_message(self, client, userdata, msg):
        print(f"{msg.topic}: {msg.payload.decode()}")
        c = msg.payload.decode()

        if msg.topic == self.full_topic[0]:
            self.run_increment = (c == "true")
        elif msg.topic == self.full_topic[1]:
            self.down_increment = (c == "true")
        elif msg.topic == self.full_topic[2]:
            self.idle_increment = (c == "true")
```

PEMBUATAN



```
    def update_time(self):
        if self.run_increment:
            self.run_seconds += 1
        elif self.idle_increment:
            self.idle_seconds += 1
        elif self.down_increment:
            self.down_seconds += 1

        self.total_seconds = self.run_seconds +
self.idle_seconds + self.down_seconds

    def get_time_data(self):
        runtime = format_time(self.run_seconds)
        idletime = format_time(self.idle_seconds)
        downtime = format_time(self.down_seconds)
        total_time = format_time(self.total_seconds)
        runtime_percent = (self.run_seconds /
 self.total_seconds) * 100 if self.total_seconds > 0 else 0
         idletime_percent = (self.idle_seconds /
 self.total_seconds) * 100 if self.total_seconds > 0 else 0
         downtime_percent = (self.down_seconds /
 self.total_seconds) * 100 if self.total_seconds > 0 else 0

        return runtime, idletime, downtime, total_time,
runtime_percent, idletime_percent, downtime_percent

# Create instances for each machine
machines = {}
for machine_name, config in machine_configs.items():
    machine = RealTimeMonitor(
        machine_name,
        mqtt_server,
        mysql_config,
        config["mqtt_topics"],
    )
```



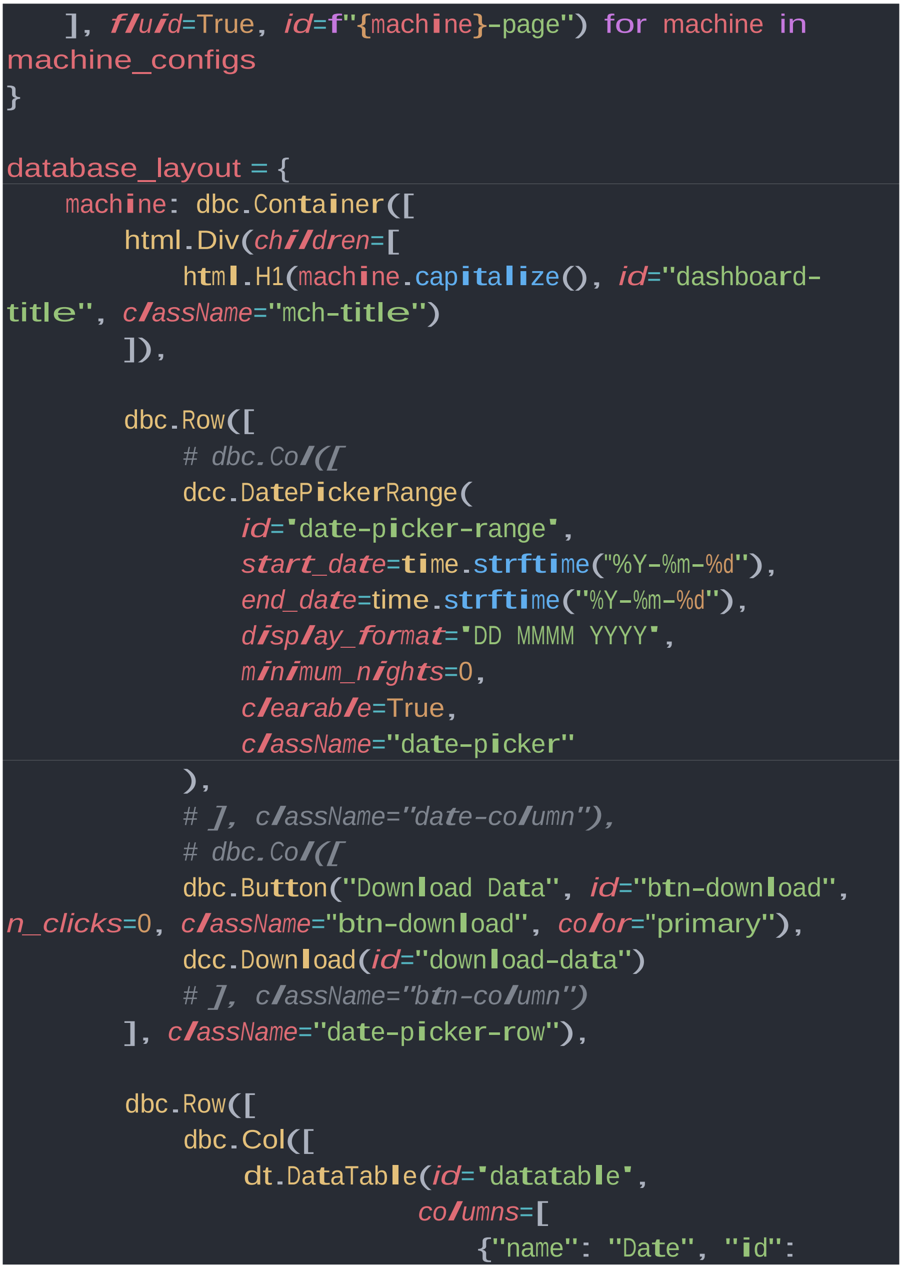
```
    machines[machine_name] = machine

button_group = dbc.ButtonGroup([
    dbc.Button("Dashboard", id="btn-dashboard", n_clicks=0,
className="btn-nav"),
    dbc.Button("Database", id="btn-database", n_clicks=0,
className="btn-nav"),
    dbc.Button("OEE", id="btn-oee", n_clicks=0,
className="btn-nav"),
], className="btn_group")

main_layout = dbc.Container([
    dbc.Col([
        dcc.Link(
            dbc.Button(f"{machine.capitalize()}",
className="mch-btn", id=f"{machine}-btn", n_clicks=0),
            href=f"/{machine}",
        )
    ]) for machine in machine_configs
], fluid=True, id="main-container", className="main-
container")

dashboard_layouts = {
    machine: dbc.Container([
        html.Div(children=[
            html.H1(machine.capitalize(), id="dashboard-
title", className="mch-title")
        ]),

        dbc.Row([
            dbc.Col([
                html.Div([
                    html.Label("Current Date: ",
className="label-text"),
                    html.Span(id='live-date',
```

PEMBUATAN

```
className="live-date"),
                ], className="text-center my-2"),

                html.Div([
                    html.Label("Current Time: ",
className="label-text"),
                    html.Span(id='live-time',
className="live-time"),
                ], className="text-center my-2"),
            ], width=8)
        ], justify="center"),

        dbc.Row([
            dbc.Col([
                html.Div([
                    html.Label("Production Time: ",
className="label-text"),
                    html.Span(id='total-time',
className="live-time"),
                ], className="text-center my-2")
            ], width=8)
        ], justify="center"),

        dbc.Container([
            dbc.Row([
                dbc.Col([
                    dbc.Card(className="card", children=[
                        dbc.CardBody([
                            html.P("Run Time",
className="card-text text-center"),
                            dbc.Progress(id="runtime-
progress", value=50, max=100, striped=True, className="card-
progress"),
                            html.P(id='runtime-time',
className="card-text text-center")
```

```
                        ], style={"border": "solid green"})
                    ])
                ], width=4),

                dbc.Col([
                    dbc.Card(className="card", children=[
                        dbc.CardBody([
                            html.P("Idle Time",
className="card-text text-center"),
                            dbc.Progress(id="idletime-
progress", value=30, max=100, striped=True, className="card-
progress"),
                            html.P(id='idletime-time',
className="card-text text-center")
                        ], style={"border": "solid yellow"})
                    ])
                ], width=4),

                dbc.Col([
                    dbc.Card(className="card", children=[
                        dbc.CardBody([
                            html.P("Down Time",
className="card-text text-center"),
                            dbc.Progress(id="downtime-
progress", value=20, max=100, striped=True, className="card-
progress"),
                            html.P(id='downtime-time',
className="card-text text-center")
                        ], style={"border": "solid red"})
                    ])
                ], width=4)
            ], justify="center", className="container")
        ]),

        button_group
```

PEMBUATAN



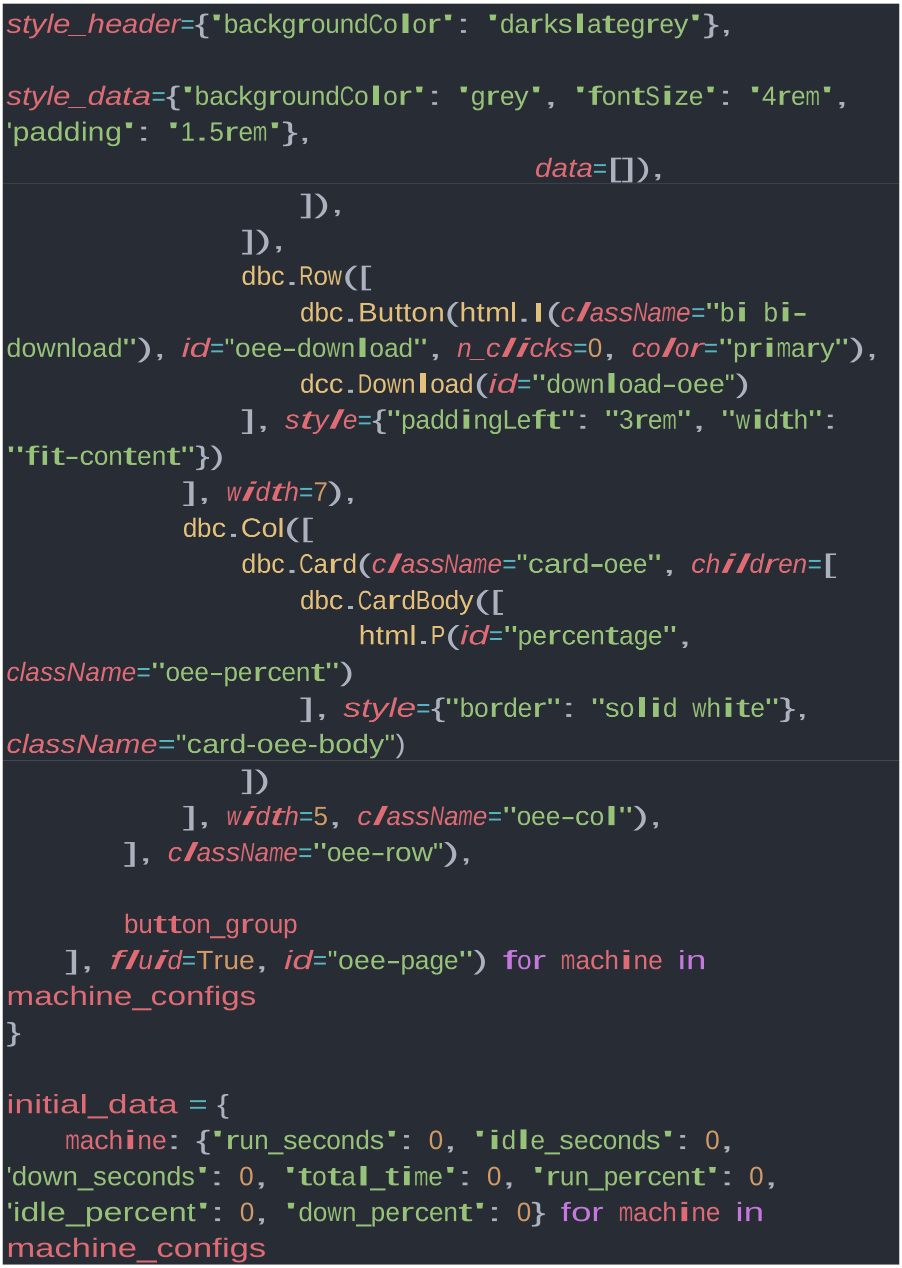
```
    ], fluid=True, id=f"{machine}-page") for machine in
machine_configs
}

database_layout = {
    machine: dbc.Container([
        html.Div(children=[
            html.H1(machine.capitalize(), id="dashboard-
title", className="mch-title")
        ]),

        dbc.Row([
            # dbc.Col([
            dcc.DatePickerRange(
                id='date-picker-range',
                start_date=time.strftime("%Y-%m-%d"),
                end_date=time.strftime("%Y-%m-%d"),
                display_format='DD MMMM YYYY',
                minimum_nights=0,
                clearable=True,
                className="date-picker"
            ),
            # ], className="date-column"),
            # dbc.Col([
            dbc.Button("Download Data", id="btn-download",
n_clicks=0, className="btn-download", color="primary"),
            dcc.Download(id="download-data")
            # ], className="btn-column")
        ], className="date-picker-row"),

        dbc.Row([
            dbc.Col([
                dt.DataTable(id='datatable',
                             columns=[
                                 {"name": "Date", "id":
```

PEMBUATAN

```
"date"},
                                {"name": "Start Time", "id":
"start_time"},
                                {"name": "End Time", "id":
"end_time"},
                                {"name": "Duration", "id":
"duration"},
                                {"name": "Status", "id":
"status"}
                            ],
                            cell_selectable=False,
                            style_table={'bordered': True,
'marginBottom': '6rem'},   # Add border to the table
                            style_cell={'textAlign':
'center', 'color': 'white'},   # Center align the cell
content
                            style_header={'backgroundColor':
'darkslategrey'},
                            style_data={'backgroundColor':
'grey'},
                            style_data_conditional=[
                                {
                                    'if': {'column_id':
'status', 'filter_query': '{status} = "RUNNING"'},
                                    'backgroundColor':
'seagreen',
                                },
                                {
                                    'if': {'column_id':
'status', 'filter_query': '{status} = "IDLE"'},
                                    'backgroundColor':
'goldenrod',
                                },
                                {
                                    'if': {'column_id':
```

```
'status', 'filter_query': '{status} = "DOWN"'},
                                                              'backgroundColor':
'firebrick',
                                                       }
                                                ],
                                                data=[]),
                        ]),
                  ]),
                  button_group
            ], fluid=True, id="database-page") for machine in
machine_configs
}

oee_layout = {
      machine: dbc.Container([
            dbc.Row([
                  dbc.Col([
                        html.H3(machine.capitalize(),
 className="part-header"),
                        dbc.Row([
                              dbc.Col([
                                    dt.DataTable(id='andontable',
                                                       columns=[
                                                              {"name": "Plan",
"id": "plan"},
                                                              {"name": "Actual",
"id": "actual"}
                                                       ],
                                                       cell_selectable=False,
                                                       style_table={'bordered':
True, 'marginBottom': '1rem'},
                                                       style_cell={'textAlign':
'center', 'color': 'white'},  # Center align the cell
content
```

PEMBUATAN



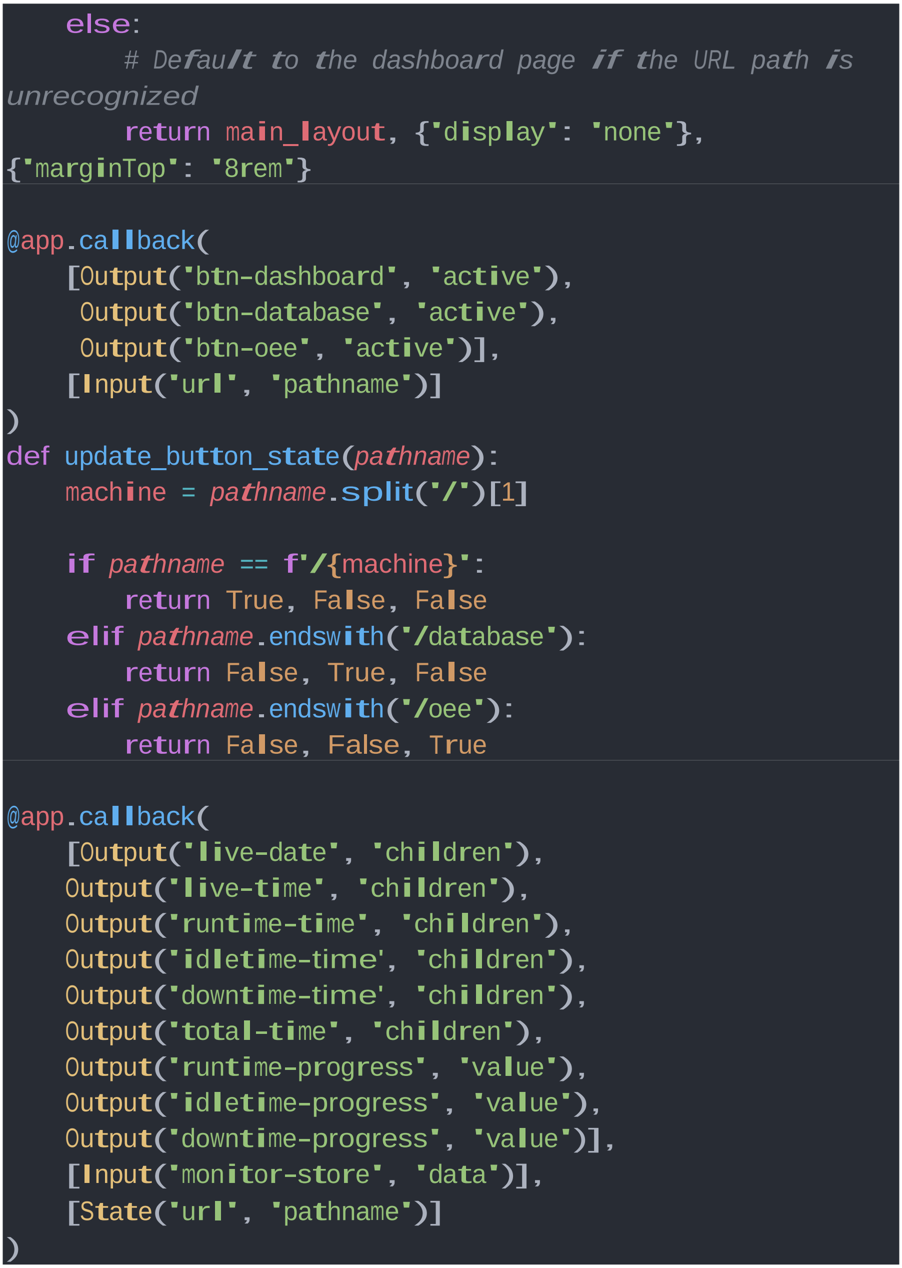
```
style_header={'backgroundColor': 'darkslategrey'},

style_data={'backgroundColor': 'grey', 'fontSize': '4rem',
'padding': '1.5rem'},
                                    data=[]),
                    ]),
                ]),
                dbc.Row([
                    dbc.Button(html.I(className="bi bi-
download"), id="oee-download", n_clicks=0, color="primary"),
                    dcc.Download(id="download-oee")
                ], style={"paddingLeft": "3rem", "width":
"fit-content"})
            ], width=7),
            dbc.Col([
                dbc.Card(className="card-oee", children=[
                    dbc.CardBody([
                        html.P(id="percentage",
className="oee-percent")
                    ], style={"border": "solid white"},
className="card-oee-body")
                ])
            ], width=5, className="oee-col"),
        ], className="oee-row"),

        button_group
    ], fluid=True, id="oee-page") for machine in
machine_configs
}

initial_data = {
    machine: {'run_seconds': 0, 'idle_seconds': 0,
'down_seconds': 0, 'total_time': 0, 'run_percent': 0,
'idle_percent': 0, 'down_percent': 0} for machine in
machine_configs
```

PEMBUATAN

```
}

app.layout = dbc.Container([
    html.Div(className="header", children=[
        dbc.Button(html.I(className="bi bi-house-door"),
href="/", id="home-btn", className="home-btn",
color="primary"),
        html.H1("ISUZU Real-time Monitoring System",
id='main-title')
    ]),
    dcc.Store(id='monitor-store', storage_type='session',
data=initial_data),
    dcc.Location(id='url', refresh=False),
    html.Div(id='page-content'),
    dcc.Interval(id='interval-component', interval=1000,
n_intervals=0),
], fluid=True, id="app-container")

@app.callback(
    [Output('url', 'pathname')],
    [Input('btn-dashboard', 'n_clicks'),
    Input('btn-database', 'n_clicks'),
    Input('btn-oee', 'n_clicks')],
    [State('url', 'pathname')]
)
def update_url(btn_dashboard, btn_database, btn_oee,
pathname):
    ctx = dash.callback_context
    if not ctx.triggered_id:
        button_id = 'btn-dashboard'
    else:
        button_id = ctx.triggered_id.split('.')[0]

    machine = pathname.split('/')[1]
    current_path = f'/{machine}'
```

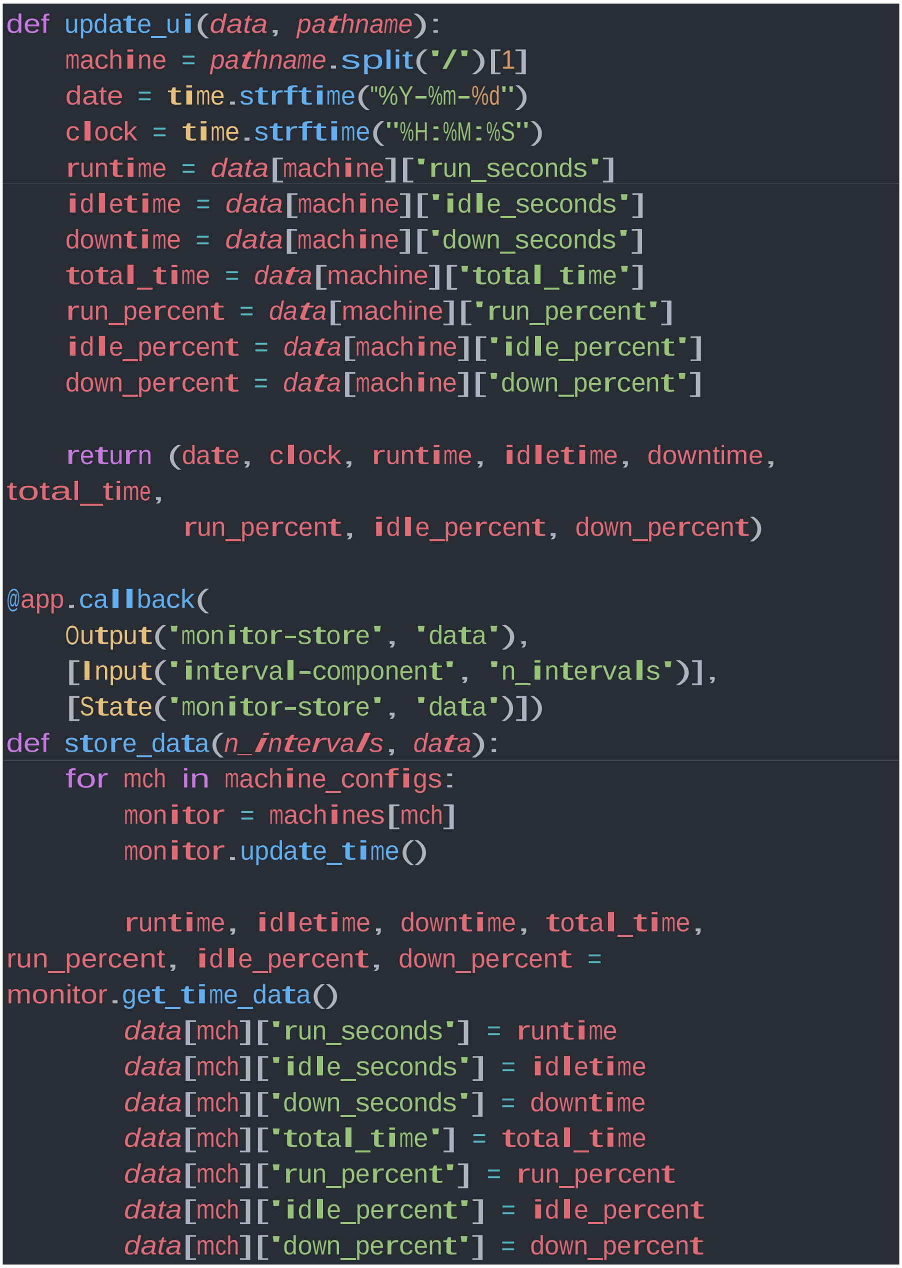
PEMBUATAN

```
    if button_id == 'btn-dashboard':
        return [f'{current_path}']
    elif button_id == 'btn-database':
        return [f'{current_path}/database']
    elif button_id == 'btn-oee':
        return [f'{current_path}/oee']

# Callback to update machine and display the corresponding
dashboard_layout
@app.callback([Output("page-content", "children"),
               Output('home-btn', 'style'),
               Output('main-title', 'style')],
              [Input('url', 'pathname')])
def display_page(pathname):
    machine = pathname.split('/')[1]
    if machine and machine in machine_configs:
        if machine in dashboard_layouts:
            if pathname.endswith('/database'):
                if machine in database_layout:
                    return database_layout[machine],
{'display': 'flex'}, {}
                else:
                    # If database hasn't been created
                    return dbc.Container([
                        html.H1(f"No Database Layout for
{machine}", id="dashboard-title", style={"color": "red"})
                    ])
            elif pathname.endswith('/oee'):
                if machine in oee_layout:
                    return oee_layout[machine], {'display':
'flex'}, {}
            else:
                return dashboard_layouts[machine],
{'display': 'flex'}, {}
```

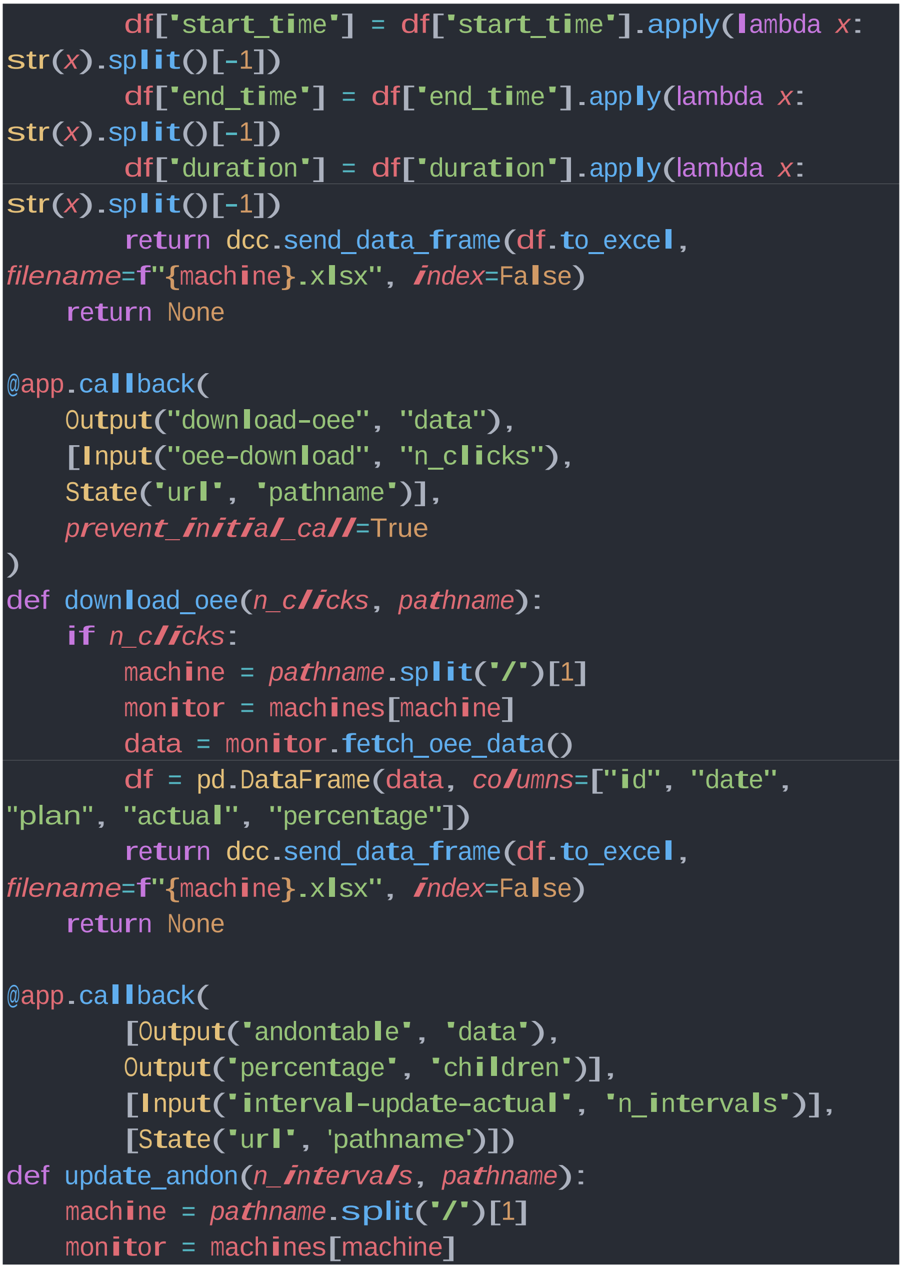


```
    else:
        # Default to the dashboard page if the URL path is
unrecognized
        return main_layout, {"display": "none"},
{"marginTop": "8rem"}

@app.callback(
    [Output("btn-dashboard", "active"),
     Output("btn-database", "active"),
     Output("btn-oee", "active")],
    [Input("url", "pathname")]
)
def update_button_state(pathname):
    machine = pathname.split("/")[1]

    if pathname == f"/{machine}":
        return True, False, False
    elif pathname.endswith("/database"):
        return False, True, False
    elif pathname.endswith("/oee"):
        return False, False, True

@app.callback(
    [Output("live-date", "children"),
    Output("live-time", "children"),
    Output("runtime-time", "children"),
    Output("idletime-time', "children"),
    Output("downtime-time', "children"),
    Output("total-time", "children"),
    Output("runtime-progress", "value"),
    Output("idletime-progress", "value"),
    Output("downtime-progress", "value")],
    [Input("monitor-store", "data")],
    [State("url", "pathname")]
)
```

PEMBUATAN

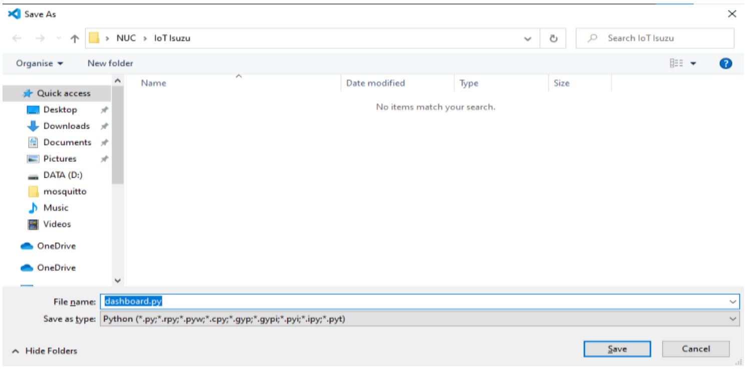


```
def update_ui(data, pathname):
    machine = pathname.split('/')[1]
    date = time.strftime("%Y-%m-%d")
    clock = time.strftime("%H:%M:%S")
    runtime = data[machine]['run_seconds']
    idletime = data[machine]['idle_seconds']
    downtime = data[machine]['down_seconds']
    total_time = data[machine]['total_time']
    run_percent = data[machine]['run_percent']
    idle_percent = data[machine]['idle_percent']
    down_percent = data[machine]['down_percent']

    return (date, clock, runtime, idletime, downtime,
total_time,
            run_percent, idle_percent, down_percent)

@app.callback(
    Output('monitor-store', 'data'),
    [Input('interval-component', 'n_intervals')],
    [State('monitor-store', 'data')])
def store_data(n_intervals, data):
    for mch in machine_configs:
        monitor = machines[mch]
        monitor.update_time()

        runtime, idletime, downtime, total_time,
run_percent, idle_percent, down_percent =
monitor.get_time_data()
        data[mch]['run_seconds'] = runtime
        data[mch]['idle_seconds'] = idletime
        data[mch]['down_seconds'] = downtime
        data[mch]['total_time'] = total_time
        data[mch]['run_percent'] = run_percent
        data[mch]['idle_percent'] = idle_percent
        data[mch]['down_percent'] = down_percent
    return data

# Define the callback to update the table content
@app.callback(
        Output('datatable', 'data'),
        [Input('interval-component', 'n_intervals'),
        Input('date-picker-range', 'start_date'),
        Input('date-picker-range', 'end_date')],
        [State('url', 'pathname')])
def update_table(n_intervals, start_date, end_date,
pathname):
    machine = pathname.split('/')[1]
    monitor = machines[machine]
    data = monitor.fetch_data_from_mysql(start_date,
end_date)
    return [{'date': str(row[0]), 'start_time': str(row[1]),
'end_time': str(row[2]), 'duration': str(row[3]), 'status':
row[4]} for row in data]

@app.callback(
    Output("download-data", "data"),
    [Input("btn-download", "n_clicks"),
    State('date-picker-range', 'start_date'),
    State('date-picker-range', 'end_date'),
    State('url', 'pathname')],
    prevent_initial_call=True
)
def download_data(n_clicks, start_date, end_date, pathname):
    if n_clicks:
        machine = pathname.split('/')[1]
        monitor = machines[machine]
        data = monitor.fetch_data_from_mysql(start_date,
end_date)
        df = pd.DataFrame(data, columns=["date",
"start_time", "end_time", "duration", "status"])
```



```
        df['start_time'] = df['start_time'].apply(lambda x:
str(x).split()[-1])
        df['end_time'] = df['end_time'].apply(lambda x:
str(x).split()[-1])
        df['duration'] = df['duration'].apply(lambda x:
str(x).split()[-1])
        return dcc.send_data_frame(df.to_excel,
filename=f"{machine}.xlsx", index=False)
    return None

@app.callback(
    Output("download-oee", "data"),
    [Input("oee-download", "n_clicks"),
    State('url', 'pathname')],
    prevent_initial_call=True
)
def download_oee(n_clicks, pathname):
    if n_clicks:
        machine = pathname.split('/')[1]
        monitor = machines[machine]
        data = monitor.fetch_oee_data()
        df = pd.DataFrame(data, columns=["id", "date",
"plan", "actual", "percentage"])
        return dcc.send_data_frame(df.to_excel,
filename=f"{machine}.xlsx", index=False)
    return None

@app.callback(
        [Output('andontable', 'data'),
        Output('percentage', 'children')],
        [Input('interval-update-actual', 'n_intervals')],
        [State('url', 'pathname')])
def update_andon(n_intervals, pathname):
    machine = pathname.split('/')[1]
    monitor = machines[machine]
    data = monitor.fetch_oee_data()
    for row in data:
        plan = row[2]
        actual = row[3]
        percentage = row[4]
    return [{'plan': plan, 'actual': actual}], percentage

if __name__== '__main__':
    try:
        app.run(debug=True, host="0.0.0.0", port=8080)
    except KeyboardInterrupt:
        print("\nScript terminated.")
```

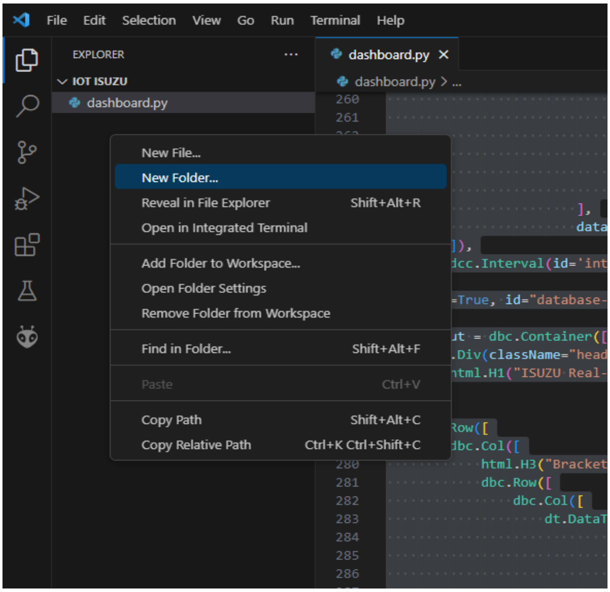
6. Simpan file tersebut dengan cara **File – Save** atau **CTRL + S** pada keyboard



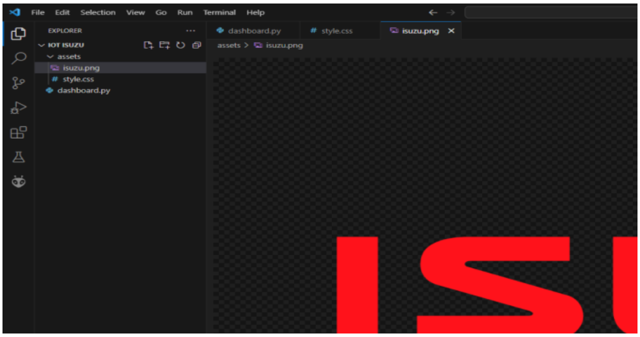
No items match your search.

File name: dashboard.py
Save as type: Python (*.py;*.rpy;*.pyw;*.cpy;*.gyp;*.gypi;*.pyi;*.ipy;*.pyt)

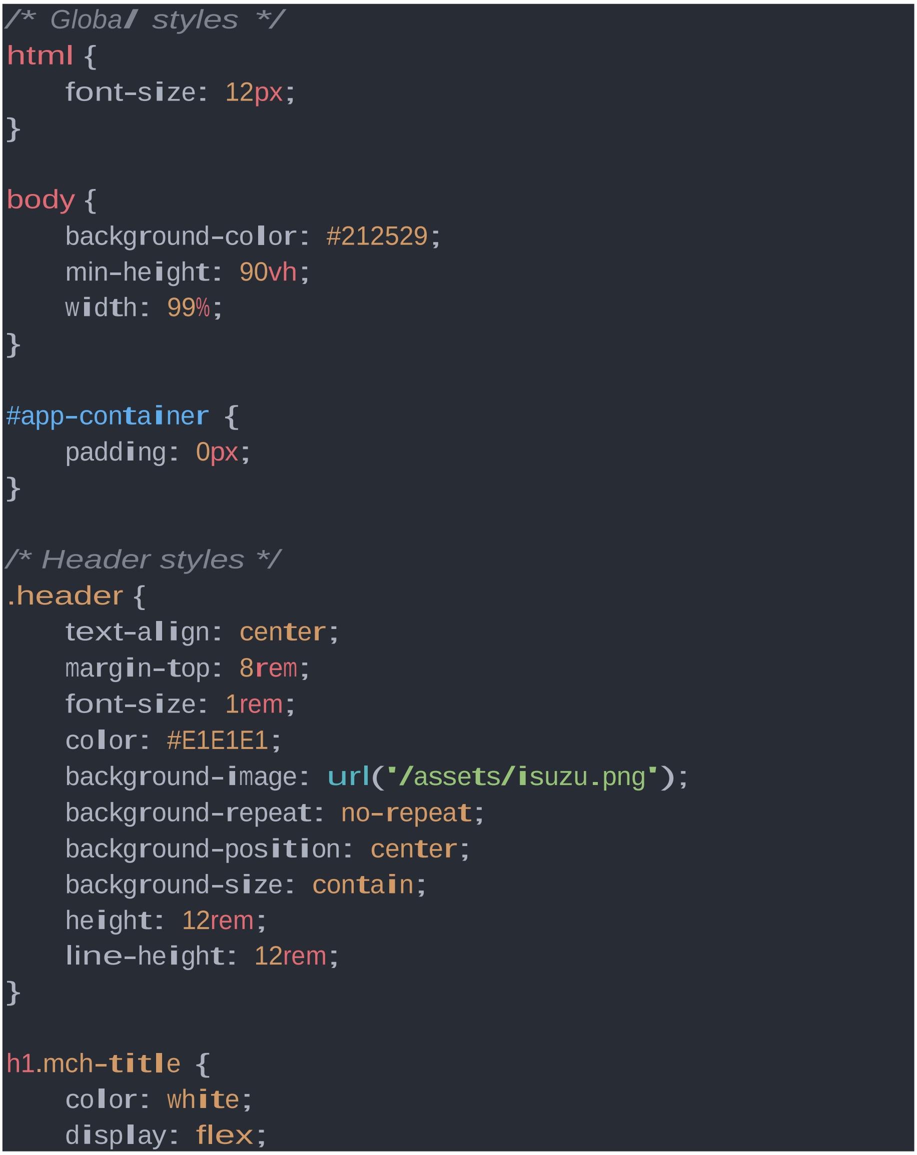
7. Langkah selanjutnya yaitu membuat folder baru bernama **assets** yang akan berisi file pendukung dashboard untuk memperindah tampilannya


File Edit Selection View Go Run Terminal Help
EXPLORER
IOT ISUZU
dashboard.py
New File...
New Folder...
Reveal in File Explorer Shift+Alt+R
Open in Integrated Terminal
Add Folder to Workspace...
Open Folder Settings
Remove Folder from Workspace
Find in Folder... Shift+Alt+F
Paste Ctrl+V
Copy Path Shift+Alt+C
Copy Relative Path Ctrl+K Ctrl+Shift+C

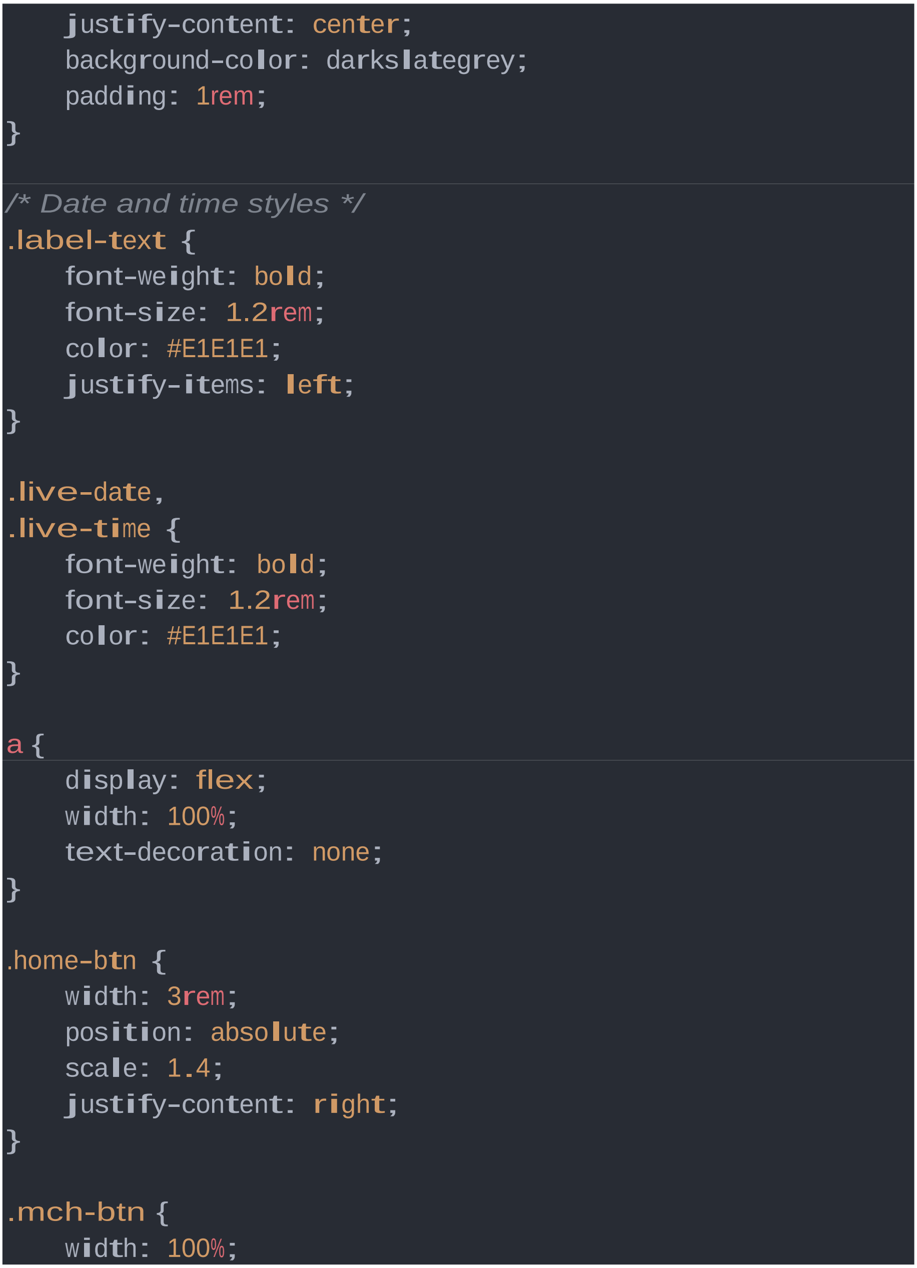
8. Masukkan file logo isuzu. File dapat diunduh padahalaman website https://1000logos.net/isuzu-logo/.



9. Buat file baru bernama style.css dan masukkan code dibawah



```css
/* Global styles */
html {
    font-size: 12px;
}

body {
    background-color: #212529;
    min-height: 90vh;
    width: 99%;
}

#app-container {
    padding: 0px;
}

/* Header styles */
.header {
    text-align: center;
    margin-top: 8rem;
    font-size: 1rem;
    color: #E1E1E1;
    background-image: url("/assets/isuzu.png");
    background-repeat: no-repeat;
    background-position: center;
    background-size: contain;
    height: 12rem;
    line-height: 12rem;
}

h1.mch-title {
    color: white;
    display: flex;
```

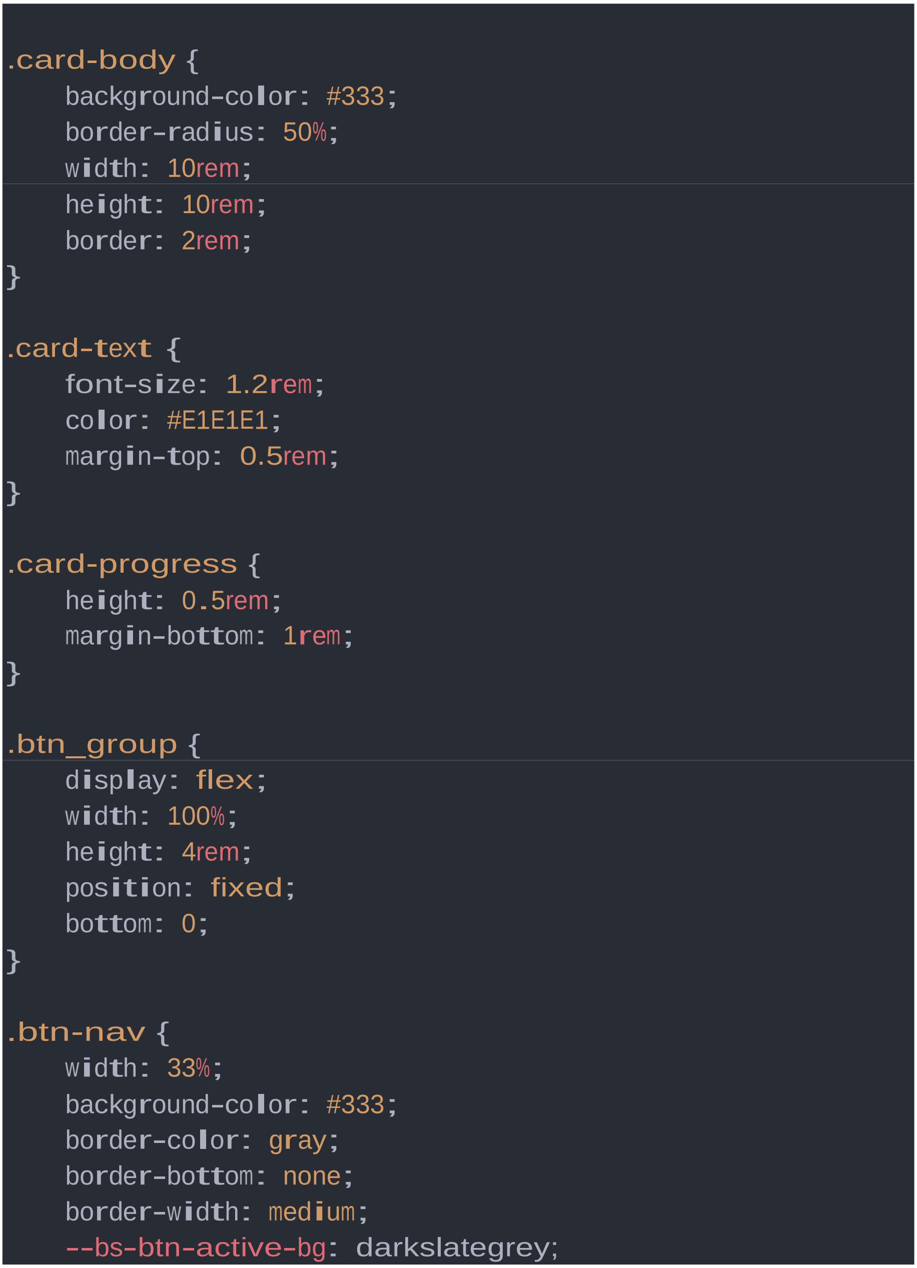


```css
    justify-content: center;
    background-color: darkslategrey;
    padding: 1rem;
}

/* Date and time styles */
.label-text {
    font-weight: bold;
    font-size: 1.2rem;
    color: #E1E1E1;
    justify-items: left;
}

.live-date,
.live-time {
    font-weight: bold;
    font-size: 1.2rem;
    color: #E1E1E1;
}

a {
    display: flex;
    width: 100%;
    text-decoration: none;
}

.home-btn {
    width: 3rem;
    position: absolute;
    scale: 1.4;
    justify-content: right;
}

.mch-btn {
    width: 100%;
```

PEMBUATAN

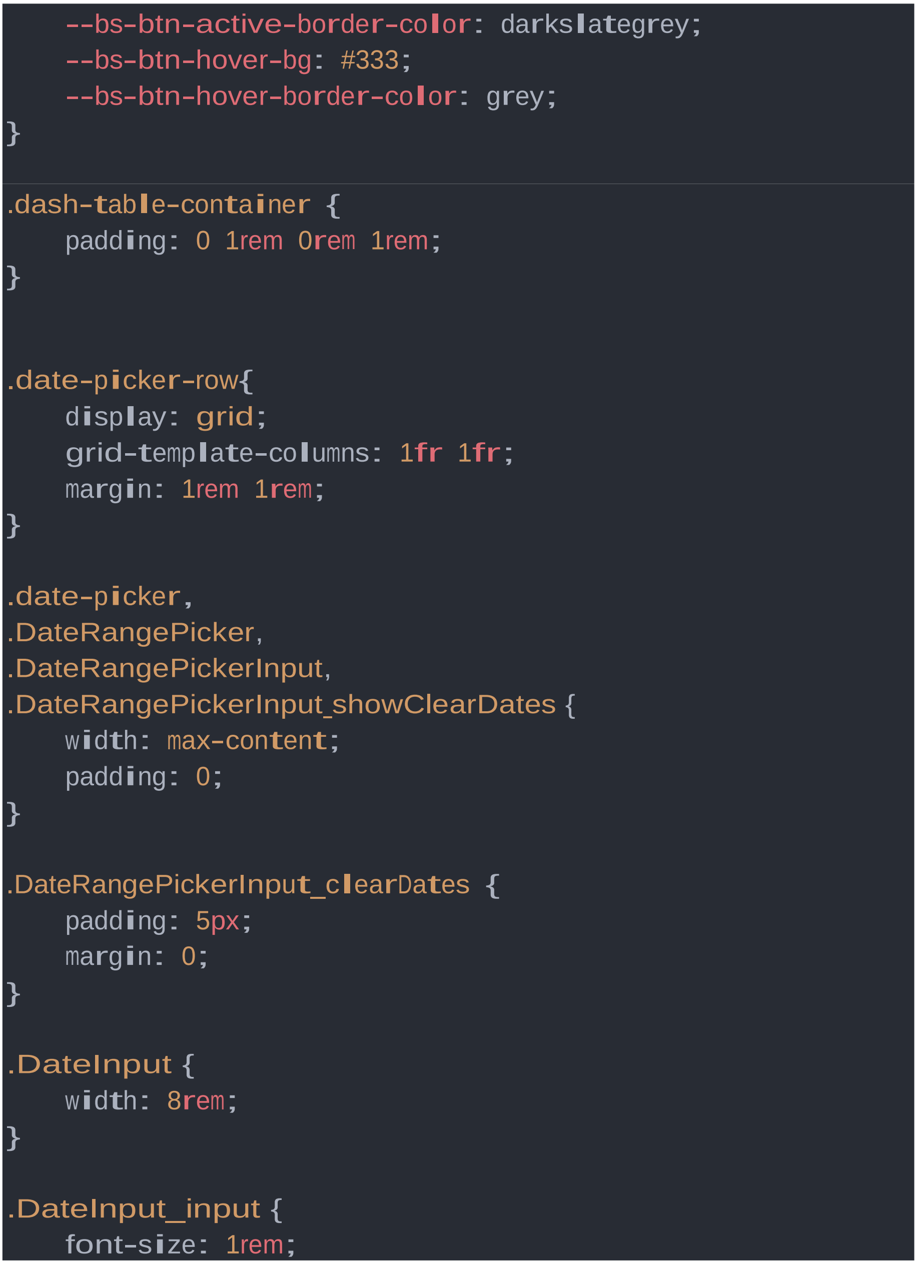
```
    height: 10rem;
    font-size: 2rem;
    background-color: darkslategrey;
    --bs-btn-border-color: none;
}

.main-container{
    margin: 0;
    max-width: 100vw;
    width: 100%;
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(350px,
1fr));
    gap: 2rem;
}

.container{
    padding: 3rem 0 0 0;
}

.col-4{
    display: flex;
    justify-content: center;
    width: fit-content;
    padding: 0 0.4rem;
}

/* Card styles */
.card {
    background-color: #333;
    border-radius: 50%;
    width: 10rem;
    height: 10rem;
    margin-bottom: 6rem;
}
```

PEMBUATAN



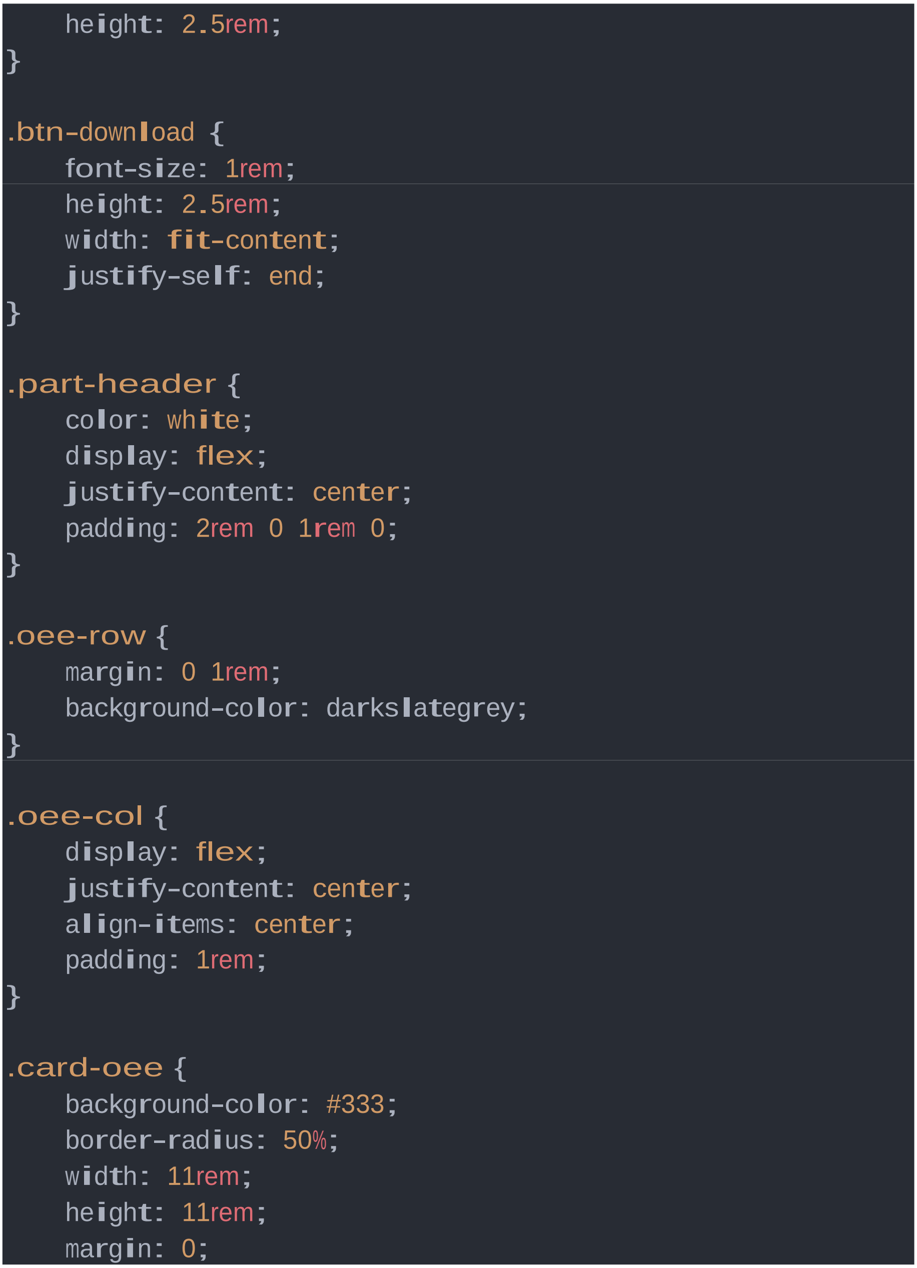
```
.card-body {
    background-color: #333;
    border-radius: 50%;
    width: 10rem;
    height: 10rem;
    border: 2rem;
}

.card-text {
    font-size: 1.2rem;
    color: #E1E1E1;
    margin-top: 0.5rem;
}

.card-progress {
    height: 0.5rem;
    margin-bottom: 1rem;
}

.btn_group {
    display: flex;
    width: 100%;
    height: 4rem;
    position: fixed;
    bottom: 0;
}

.btn-nav {
    width: 33%;
    background-color: #333;
    border-color: gray;
    border-bottom: none;
    border-width: medium;
    --bs-btn-active-bg: darkslategrey;
```

PEMBUATAN

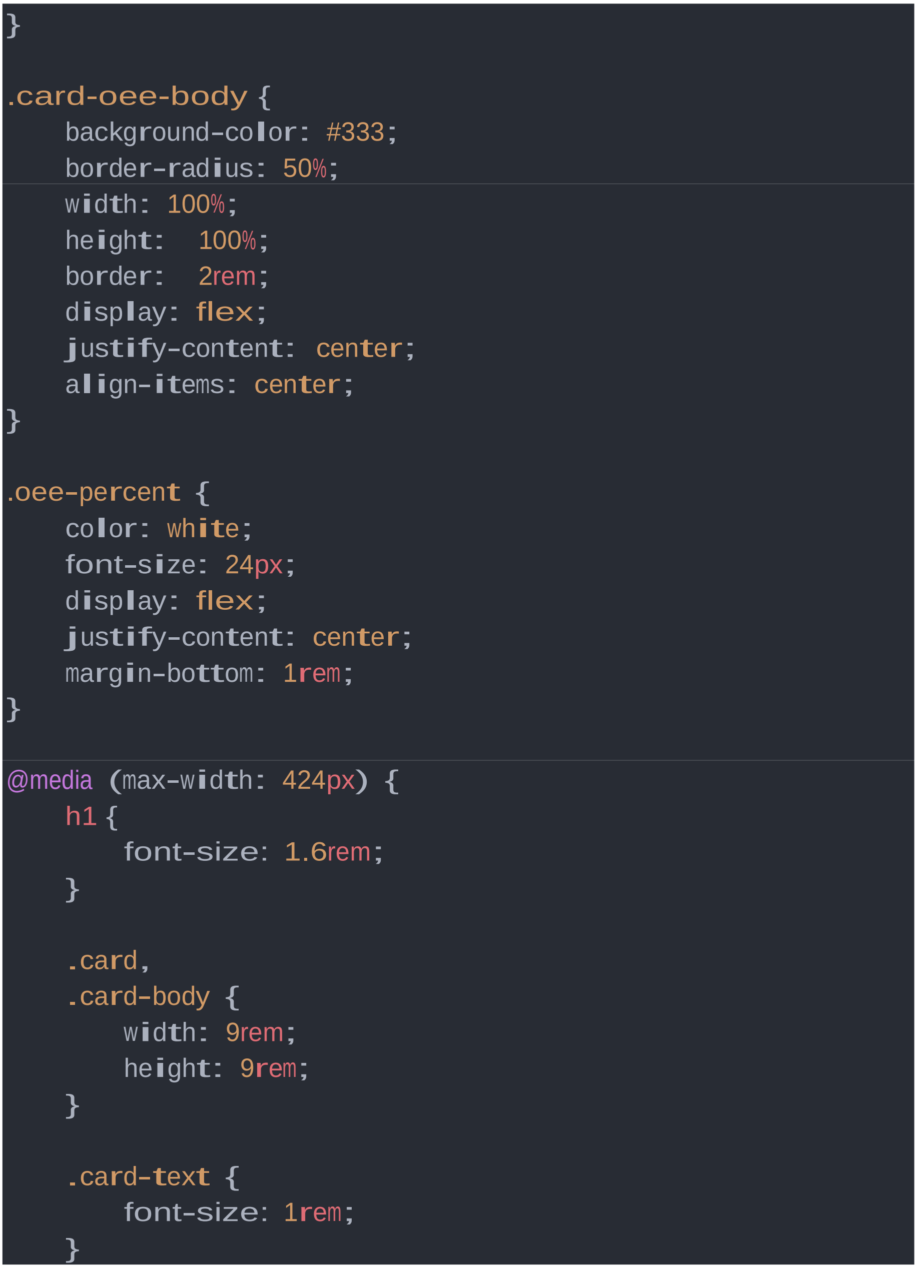


```
    --bs-btn-active-border-color: darkslategrey;
    --bs-btn-hover-bg: #333;
    --bs-btn-hover-border-color: grey;
}

.dash-table-container {
    padding: 0 1rem 0rem 1rem;
}


.date-picker-row{
    display: grid;
    grid-template-columns: 1fr 1fr;
    margin: 1rem 1rem;
}

.date-picker,
.DateRangePicker,
.DateRangePickerInput,
.DateRangePickerInput_showClearDates {
    width: max-content;
    padding: 0;
}

.DateRangePickerInput_clearDates {
    padding: 5px;
    margin: 0;
}

.DateInput {
    width: 8rem;
}

.DateInput_input {
    font-size: 1rem;
```

PEMBUATAN

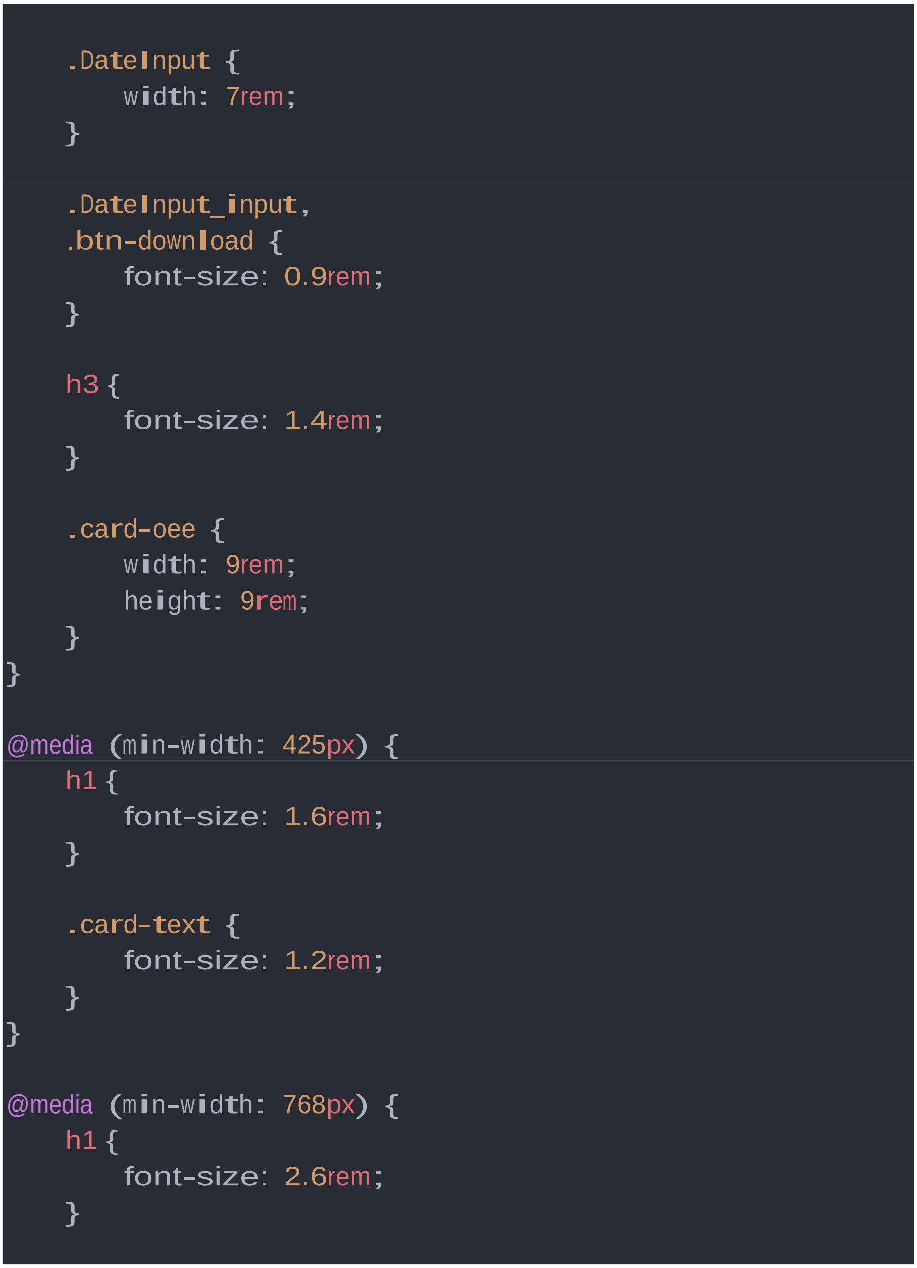


```
    height: 2.5rem;
}

.btn-download {
    font-size: 1rem;
    height: 2.5rem;
    width: fit-content;
    justify-self: end;
}

.part-header {
    color: white;
    display: flex;
    justify-content: center;
    padding: 2rem 0 1rem 0;
}

.oee-row {
    margin: 0 1rem;
    background-color: darkslategrey;
}

.oee-col {
    display: flex;
    justify-content: center;
    align-items: center;
    padding: 1rem;
}

.card-oee {
    background-color: #333;
    border-radius: 50%;
    width: 11rem;
    height: 11rem;
    margin: 0;
```



```
}

.card-oee-body {
    background-color: #333;
    border-radius: 50%;
    width: 100%;
    height:  100%;
    border:  2rem;
    display: flex;
    justify-content: center;
    align-items: center;
}

.oee-percent {
    color: white;
    font-size: 24px;
    display: flex;
    justify-content: center;
    margin-bottom: 1rem;
}

@media (max-width: 424px) {
    h1 {
        font-size: 1.6rem;
    }

    .card,
    .card-body {
        width: 9rem;
        height: 9rem;
    }

    .card-text {
        font-size: 1rem;
    }
```

PEMBUATAN

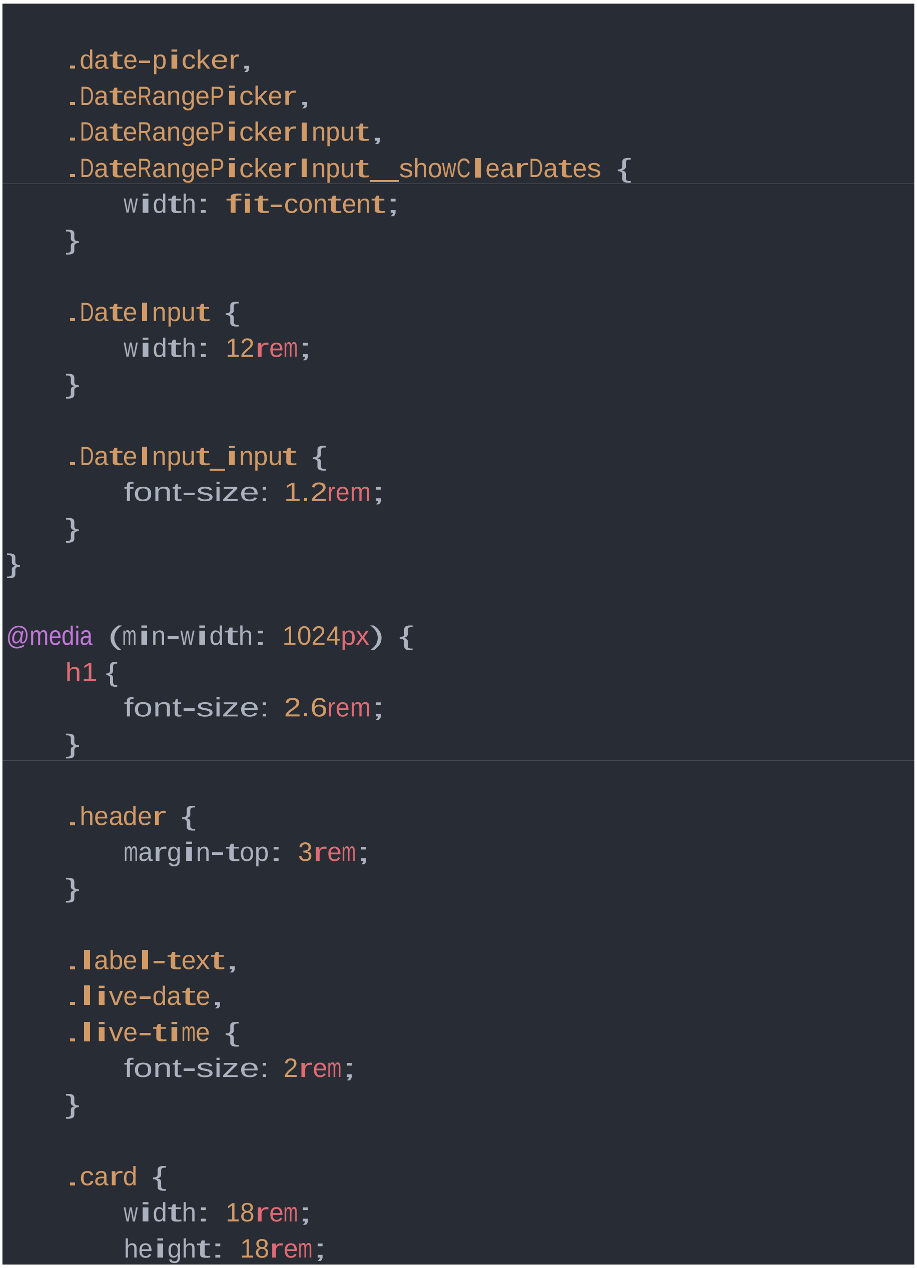


```css
    .DateInput {
        width: 7rem;
    }

    .DateInput_input,
    .btn-download {
        font-size: 0.9rem;
    }

    h3 {
        font-size: 1.4rem;
    }

    .card-oee {
        width: 9rem;
        height: 9rem;
    }
}

@media (min-width: 425px) {
    h1 {
        font-size: 1.6rem;
    }

    .card-text {
        font-size: 1.2rem;
    }
}

@media (min-width: 768px) {
    h1 {
        font-size: 2.6rem;
    }
```

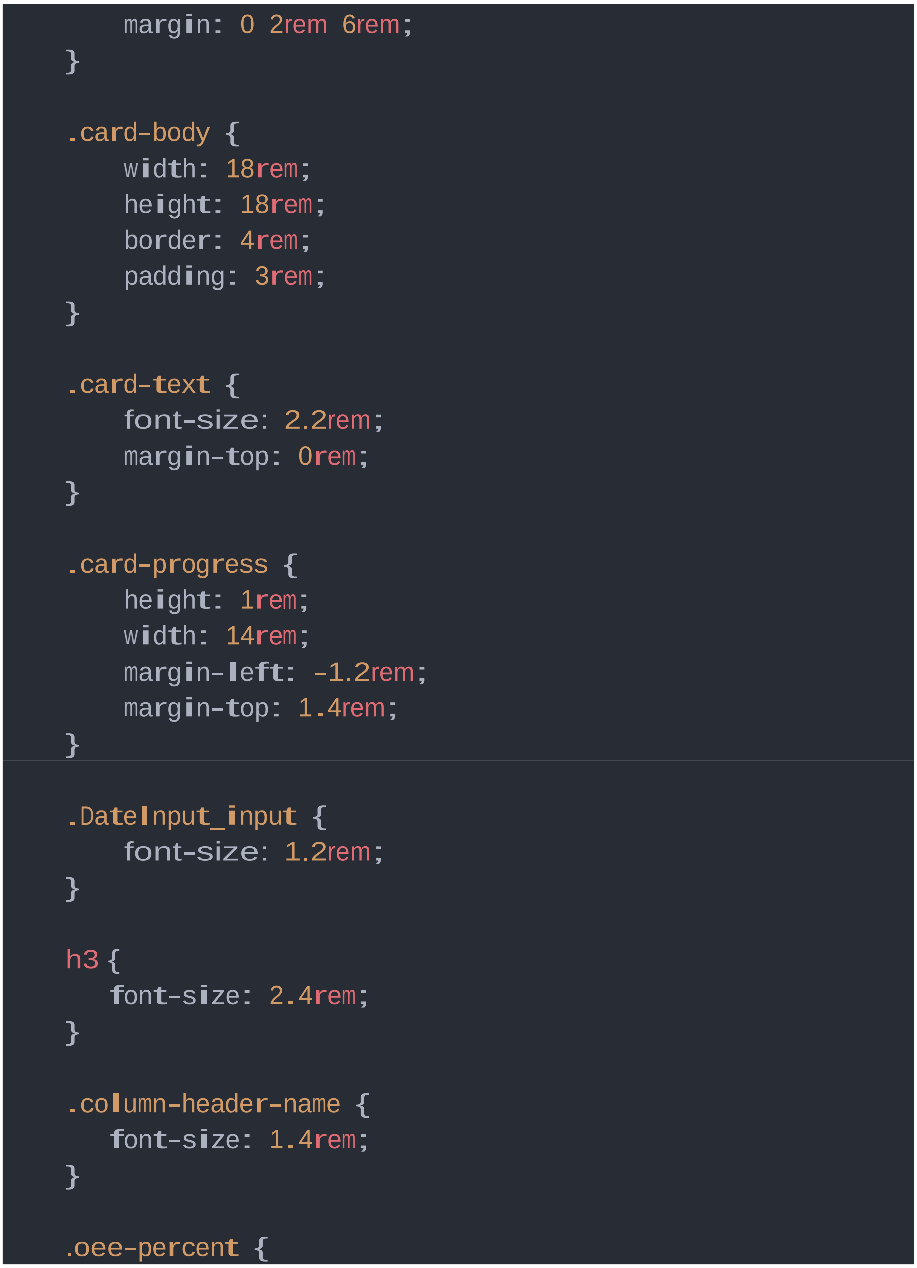
PEMBUATAN

```css
.home-btn {
    width: 4rem;
    scale: 2;
}

.label-text,
.live-date,
.live-time {
    font-size: 2rem;
}

.card,
.card-body {
    width: 17rem;
    height: 17rem;
    border: 4rem;
}

.card-text {
    font-size: 2.2rem;
    margin-top: 1.5rem;
}

.card-progress {
    height: 1rem;
    margin-top: 2rem;
}

.dash-table-container {
    padding: 0 2rem 0 2rem;
}

.date-picker-row{
    margin: 1rem 2rem;
}
```

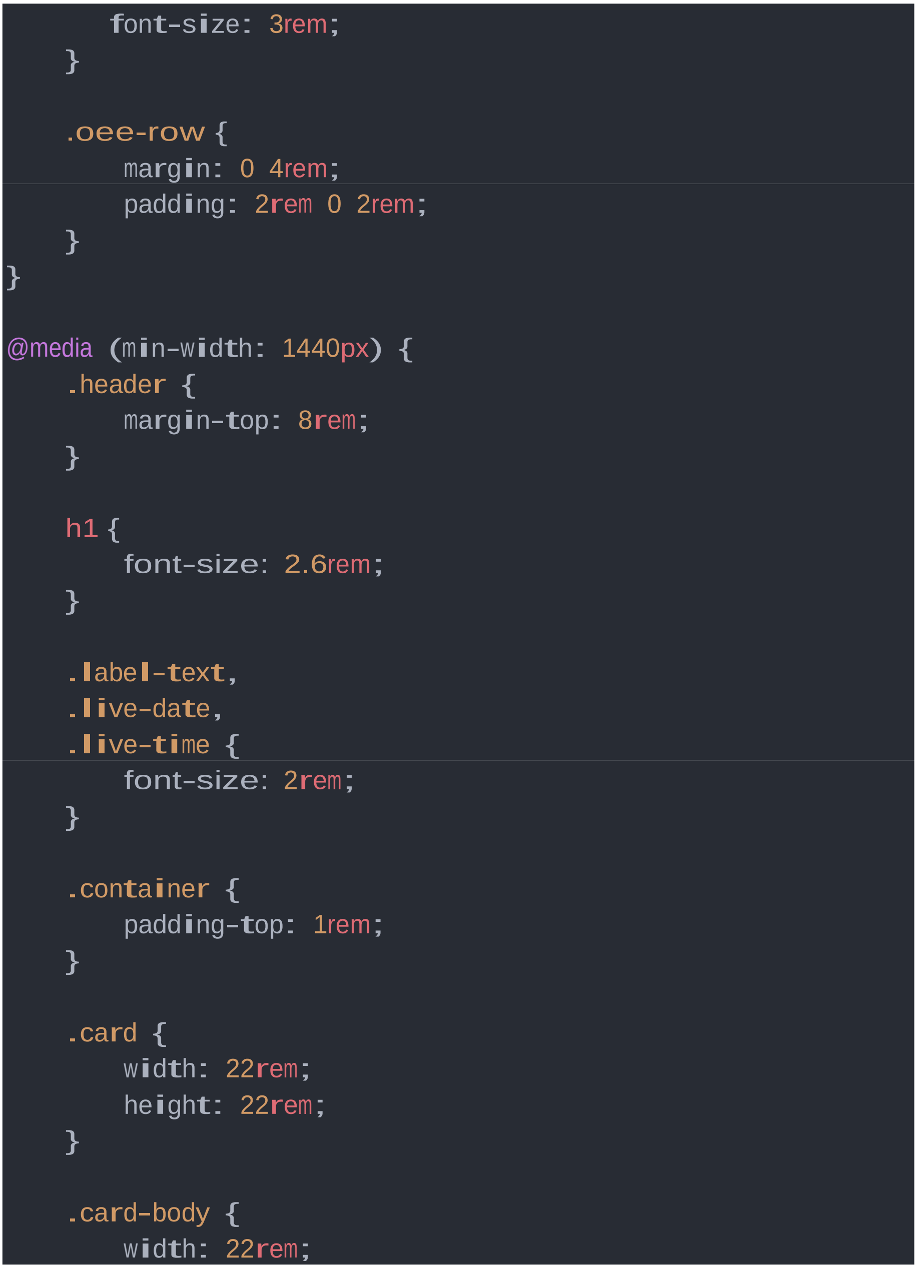
PEMBUATAN



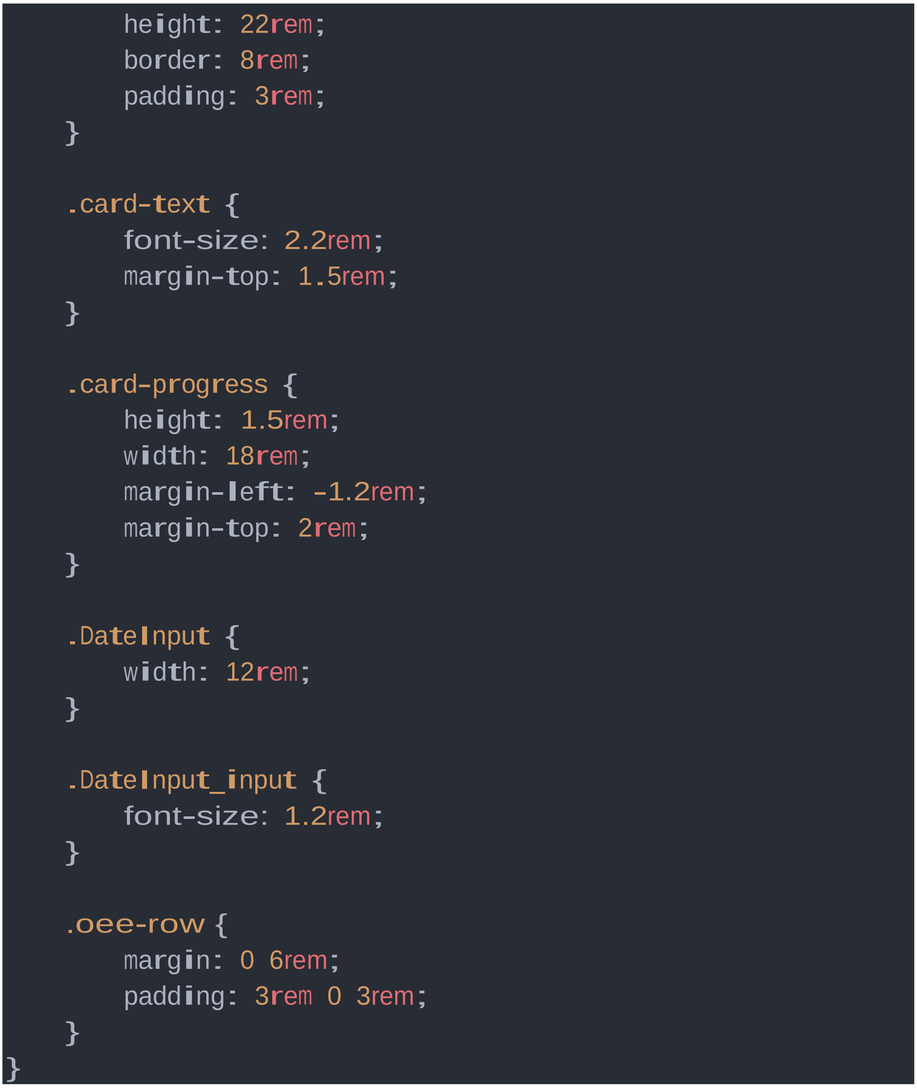
```
    .date-picker,
    .DateRangePicker,
    .DateRangePickerInput,
    .DateRangePickerInput__showClearDates {
        width: fit-content;
    }

    .DateInput {
        width: 12rem;
    }

    .DateInput_input {
        font-size: 1.2rem;
    }
}

@media (min-width: 1024px) {
    h1 {
        font-size: 2.6rem;
    }

    .header {
        margin-top: 3rem;
    }

    .label-text,
    .live-date,
    .live-time {
        font-size: 2rem;
    }

    .card {
        width: 18rem;
        height: 18rem;
```



```
    margin: 0 2rem 6rem;
}

.card-body {
    width: 18rem;
    height: 18rem;
    border: 4rem;
    padding: 3rem;
}

.card-text {
    font-size: 2.2rem;
    margin-top: 0rem;
}

.card-progress {
    height: 1rem;
    width: 14rem;
    margin-left: -1.2rem;
    margin-top: 1.4rem;
}

.DateInput_input {
    font-size: 1.2rem;
}

h3 {
   font-size: 2.4rem;
}

.column-header-name {
   font-size: 1.4rem;
}

.oee-percent {
```

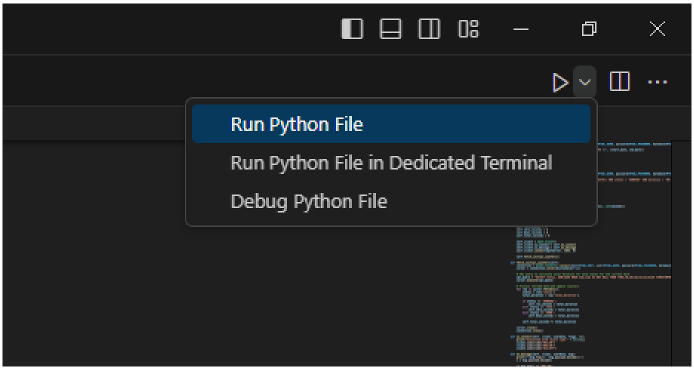


```
        font-size: 3rem;
    }

    .oee-row {
        margin: 0 4rem;
        padding: 2rem 0 2rem;
    }
}

@media (min-width: 1440px) {
    .header {
        margin-top: 8rem;
    }

    h1 {
        font-size: 2.6rem;
    }

    .label-text,
    .live-date,
    .live-time {
        font-size: 2rem;
    }

    .container {
        padding-top: 1rem;
    }

    .card {
        width: 22rem;
        height: 22rem;
    }

    .card-body {
        width: 22rem;
```

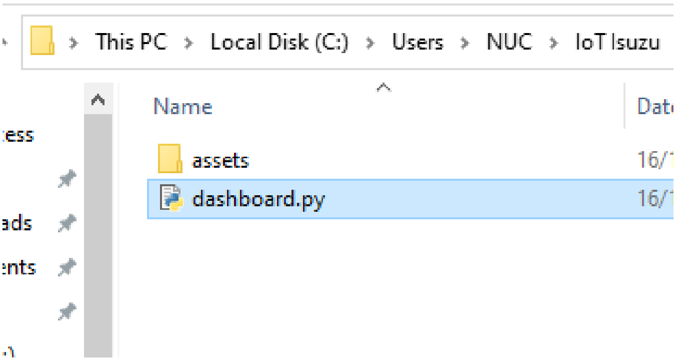


```
        height: 22rem;
        border: 8rem;
        padding: 3rem;
    }

    .card-text {
        font-size: 2.2rem;
        margin-top: 1.5rem;
    }

    .card-progress {
        height: 1.5rem;
        width: 18rem;
        margin-left: -1.2rem;
        margin-top: 2rem;
    }

    .DateInput {
        width: 12rem;
    }

    .DateInput_input {
        font-size: 1.2rem;
    }

    .oee-row {
        margin: 0 6rem;
        padding: 3rem 0 3rem;
    }
}
```

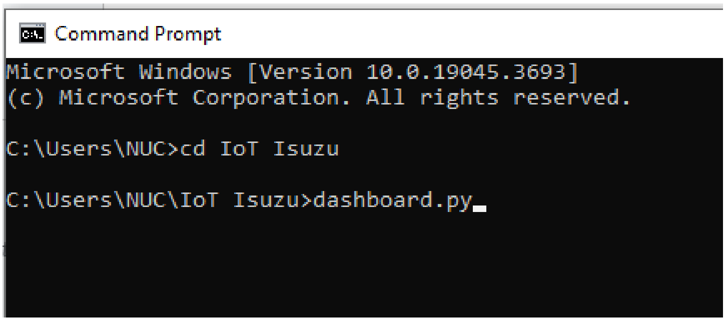
10. Langkah selanjutnya yaitu running program dashboard.py yang dapat dilakukan dengan 3 cara antara lain :
    a. Menekan tombol run yang terdapat pada Visual Studio Code



    b. Buka folder menggunkaan file explorer lalu klik 2x dashboard.py



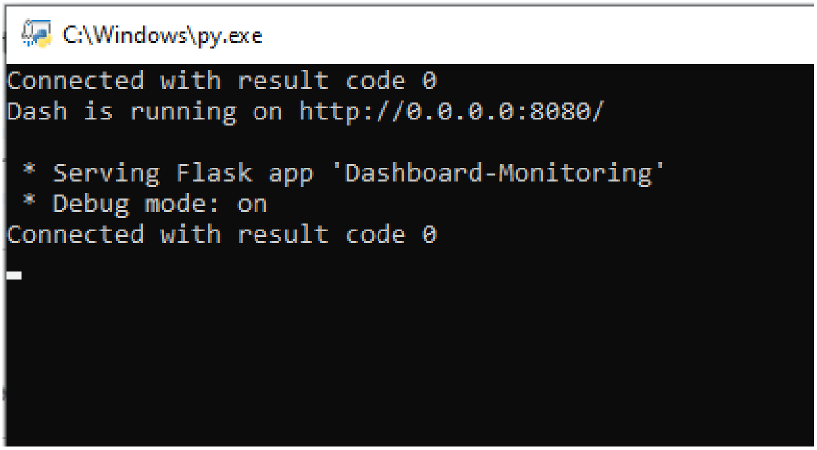
c. Buka folder menggunakan commandprompt kemudian ketik nama file (dashboard.py)



Command Prompt

```
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\NUC>cd IoT Isuzu

C:\Users\NUC\IoT Isuzu>dashboard.py_
```

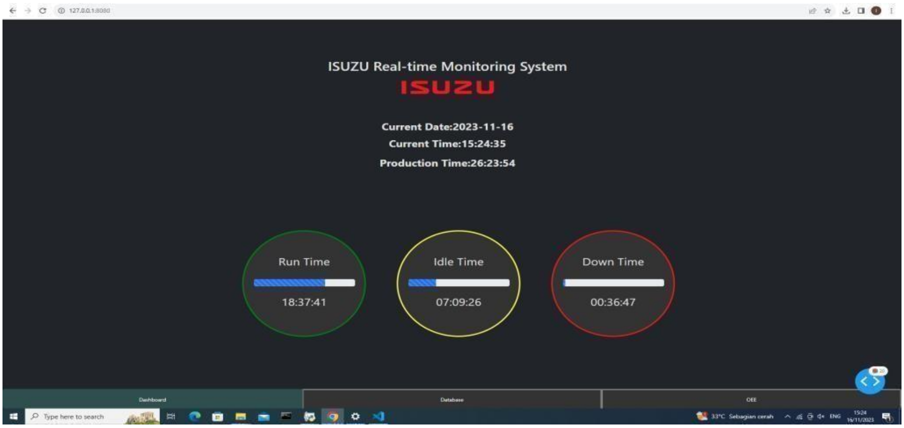
11. Jika berhasil akan muncul tampilan seperti gambardibawah ini



C:\Windows\py.exe

```
Connected with result code 0
Dash is running on http://0.0.0.0:8080/

 * Serving Flask app 'Dashboard-Monitoring'
 * Debug mode: on
Connected with result code 0
```

12. Copy alamat yang tertera (http://0.0.0.:8080/ atau http://127.0.0.1:8080 atau http://localhost:8080) kedalam web browser untuk melihat tampilan dashboard seperti gambar dibawah ini



13. Jika ingin mengakses dari perangkat lain, perangkat yang digunakan wajib terhubung dengan jaringan yang sama dengan komputer server. Lalu pada web browser ketikkan http://alamat_ip:8080. Ganti alamat_ip dengan alamat IPserver

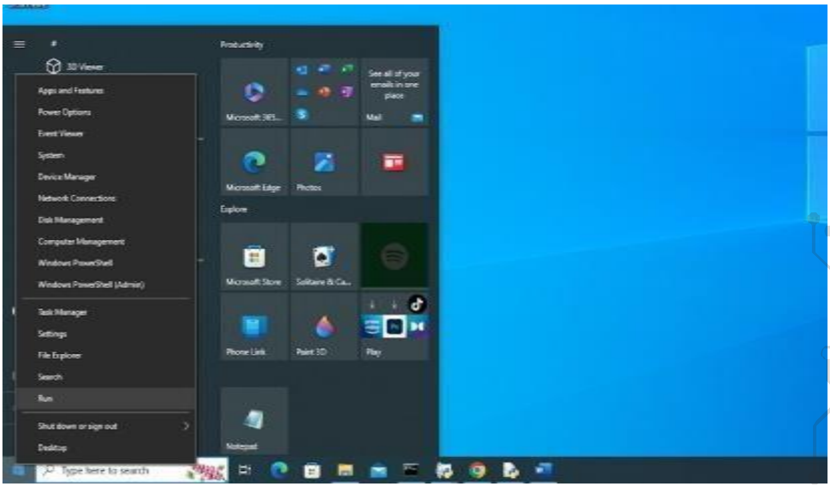
Instalasi

1. Putar saklar mesin ke posisi off
2. Buka penutup mesin pada bagian belakang
3. Hubungkan terminal positif relay 1 atau relay run time ke soket Y22
4. Hubungkan terminal positif relay 2 atau relay down time ke soket Y23
5. Hubungkan kedua terminal negatif relay ke 0V
6. Hubungkan terminal positif buck converter ke soket 24V
7. Hubungkan terminal negatif buck converter ke 0V
8. Hubungkan pin nomor 9 pada relay 1 dan relay 2 ke soket 3V pada wemos
9. Hubungkan pin nomor 5 pada relay 1 ke soket D0 pada wemos
10. Hubungkan pin nomor 5 pada relay 2 ke soket D8 pada wemos
11. Hubungkan USB port wemos ke USB port Buck Converter
12. Setelah komponen terhubung dengan benar kemudian nyalakan mesin

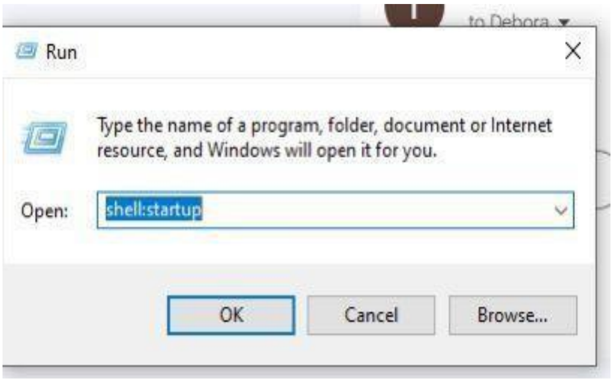
Pemantauan Sistem

Nyalakan Mini PC yang sudah terhubung ke monitor

1. Setelah terhubung, klik windows yang terletak pada kiri bawah monitor lalu klik kanan pada kursor
2. Akan tampil seperti gambar dibawah ini dan klik "run"



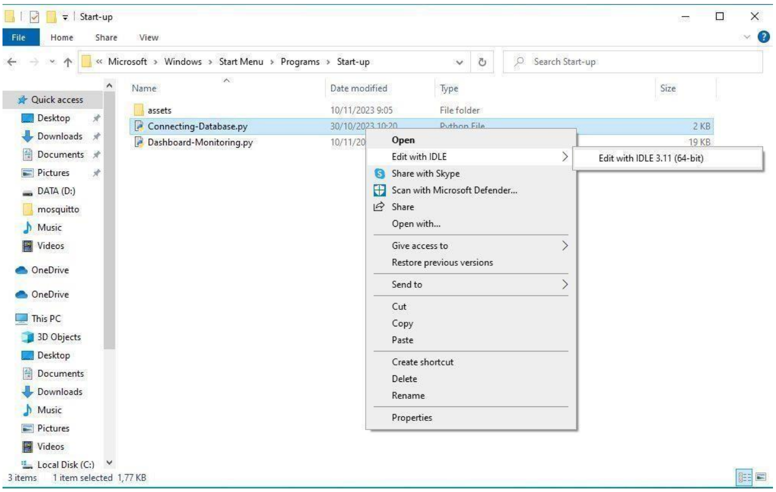
4. Kemudian akan tampil seperti gambar dibawah, lalu klik "OK"



Type the name of a program, folder, document or Internet resource, and Windows will open it for you.

Open: shell:startup

5. Maka akan muncul python file yang akan di run melalui Python IDE, alasan mengapa menyimpan python file di microsoft adalah agar tidak perlu run python file setiap kali menghidupkan monitor karena akan otomatis run
6. Klik kanan pada file “Connecting-Database.py”, lalu klik “Edit with IDLE” seperti gambar dibawah ini

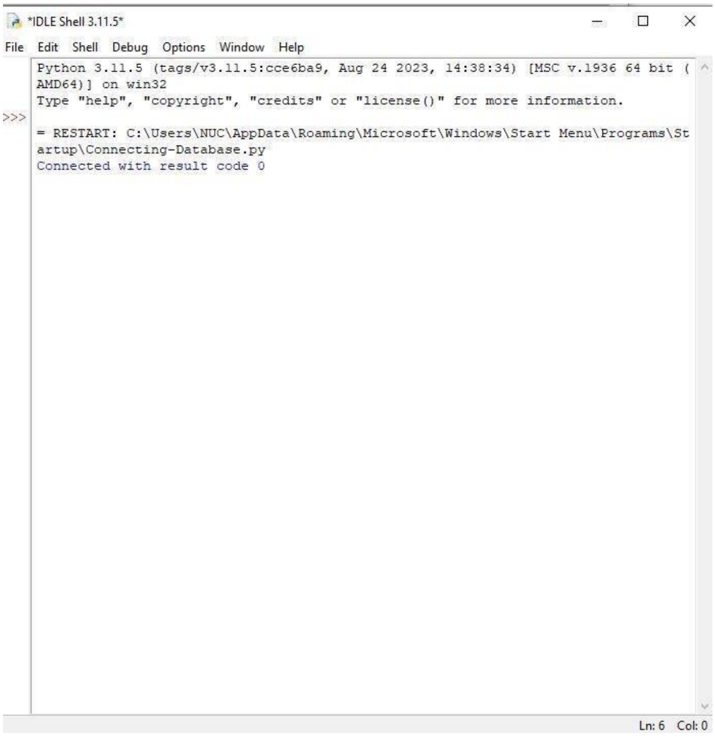


Name	Date modified	Type	Size
assets	10/11/2023 9:05	File folder	
Connecting-Database.py	30/10/2023 10:20	Python File	2 KB
Dashboard-Monitoring.py	10/11/20		19 KB

3 items 1 item selected 1,77 KB

7. Akan muncul python code pada monitor lalu klik “Run”, dan pilih option “Run Module”

8. Tunggu sampai muncul seperti gambar dibawah ini



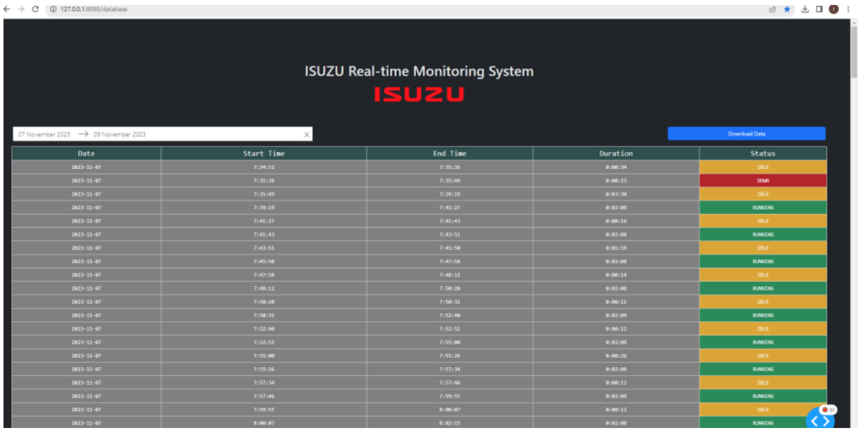
```
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.1936 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\NUC\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\St
artup\Connecting-Database.py
Connected with result code 0
```

9. Setelahnya, klik kanan pafa file "Dashboard-Monitoring.py", lalu klik "Edit with IDLE" seperti saat kita run file "Connecting-Database.py"
10. Akan muncul python code pada monitor lalu klik "Run" dan pilih option "Run Module"
11. Untuk melihat hasil monitoring klik shortcut bernama dashboard atau dengan mengetik alamat web secara manual yaitu pada alamat http://127.0.0.1:8080/
12. Hasil pembacaan relay dapat dipantau secara real time melalui web tersebut

CARA PENGGUNAAN

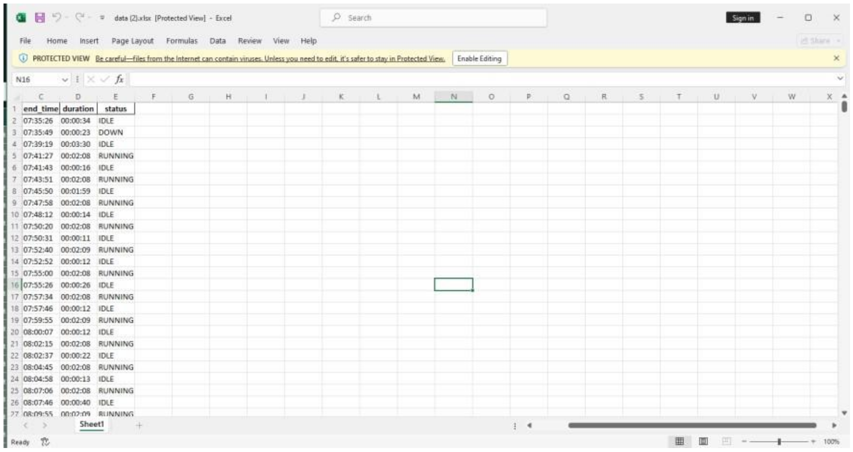
Mengekspor Data

1. Untuk mengekspor data hasil pemantauan klik export data pada kanan atas laman dashboard web


127.0.0.1:8080/database
ISUZU Real-time Monitoring System
ISUZU
07 November 2023 → 09 November 2023
Download Data

Date	Start Time	End Time	Duration	Status
2023-11-07	7:34:52	7:35:26	0:00:34	IDLE
2023-11-07	7:35:26	7:35:49	0:00:23	DOWN
2023-11-07	7:35:49	7:39:19	0:03:30	IDLE
2023-11-07	7:39:19	7:41:27	0:02:08	RUNNING
2023-11-07	7:41:27	7:41:43	0:00:16	IDLE
2023-11-07	7:41:43	7:43:51	0:02:08	RUNNING
2023-11-07	7:43:51	7:45:50	0:01:59	IDLE
2023-11-07	7:45:50	7:47:58	0:02:08	RUNNING
2023-11-07	7:47:58	7:48:12	0:00:14	IDLE
2023-11-07	7:48:12	7:50:20	0:02:08	RUNNING
2023-11-07	7:50:20	7:50:31	0:00:11	IDLE
2023-11-07	7:50:31	7:52:40	0:02:09	RUNNING
2023-11-07	7:52:40	7:52:52	0:00:12	IDLE
2023-11-07	7:52:52	7:55:00	0:02:08	RUNNING
2023-11-07	7:55:00	7:55:26	0:00:26	IDLE
2023-11-07	7:55:26	7:57:34	0:02:08	RUNNING
2023-11-07	7:57:34	7:57:46	0:00:12	IDLE
2023-11-07	7:57:46	7:59:55	0:02:09	RUNNING
2023-11-07	7:59:55	8:00:07	0:00:12	IDLE
2023-11-07	8:00:07	8:02:15	0:02:08	RUNNING

2. Buka hasil ekspor data melalui microsoft excel dan akan muncul tampilan seperti dibawah ini


data (2).xlsx [Protected View] - Excel
Search
Sign in
File Home Insert Page Layout Formulas Data Review View Help
PROTECTED VIEW Be careful—files from the Internet can contain viruses. Unless you need to edit, it's safer to stay in Protected View.
Enable Editing
N16
Sheet1
Ready
100%

end_time	duration	status
07:35:26	00:00:34	IDLE
07:35:49	00:00:23	DOWN
07:39:19	00:03:30	IDLE
07:41:27	00:02:08	RUNNING
07:41:43	00:00:16	IDLE
07:43:51	00:02:08	RUNNING
07:45:50	00:01:59	IDLE
07:47:58	00:02:08	RUNNING
07:48:12	00:00:14	IDLE
07:50:20	00:02:08	RUNNING
07:50:31	00:00:11	IDLE
07:52:40	00:02:09	RUNNING
07:52:52	00:00:12	IDLE
07:55:00	00:02:08	RUNNING
07:55:26	00:00:26	IDLE
07:57:34	00:02:08	RUNNING
07:57:46	00:00:12	IDLE
07:59:55	00:02:09	RUNNING
08:00:07	00:00:12	IDLE
08:02:15	00:02:08	RUNNING
08:02:37	00:00:22	IDLE
08:04:45	00:02:08	RUNNING
08:04:58	00:00:13	IDLE
08:07:06	00:02:08	RUNNING
08:07:46	00:00:40	IDLE
08:09:55	00:02:09	RUNNING