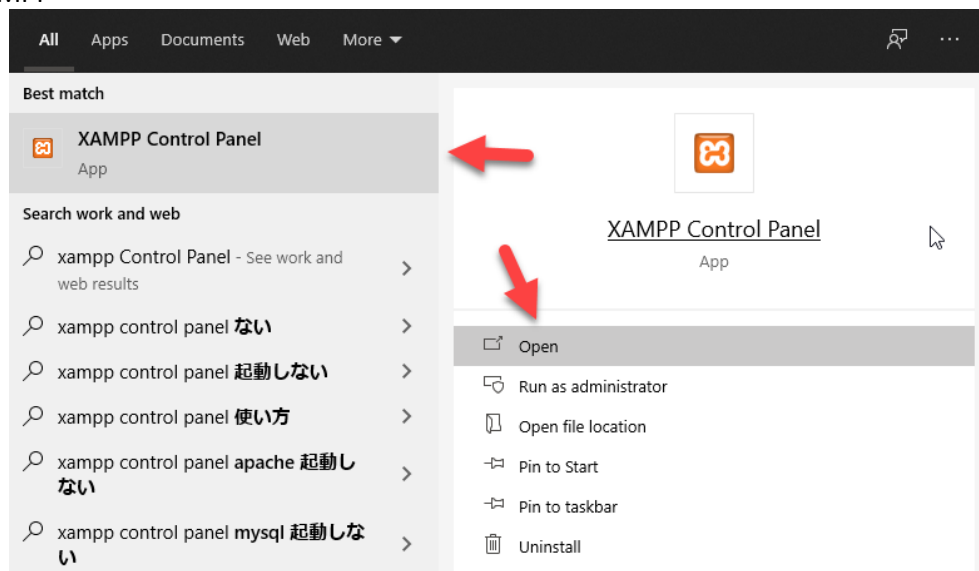


Langkah Instalasi XAMPP dan setup database:

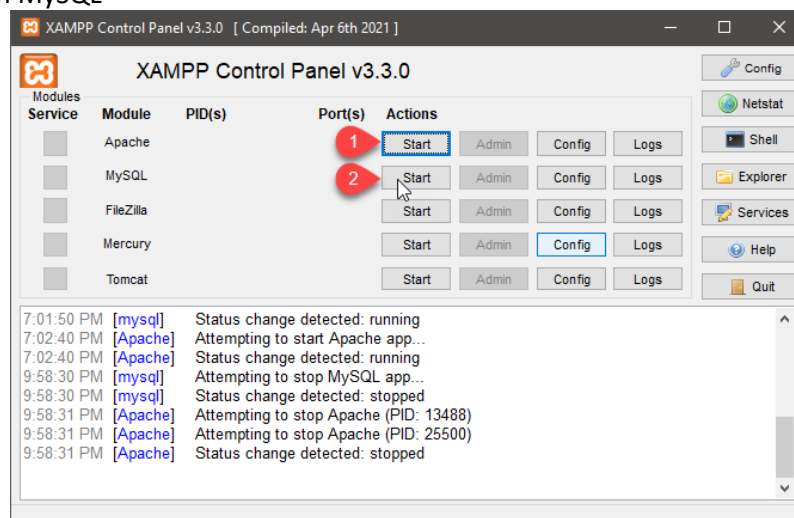
1. Kunjungi www.apachefriends.org
2. Pilih link download yang sesuai dengan sistem operasi Anda



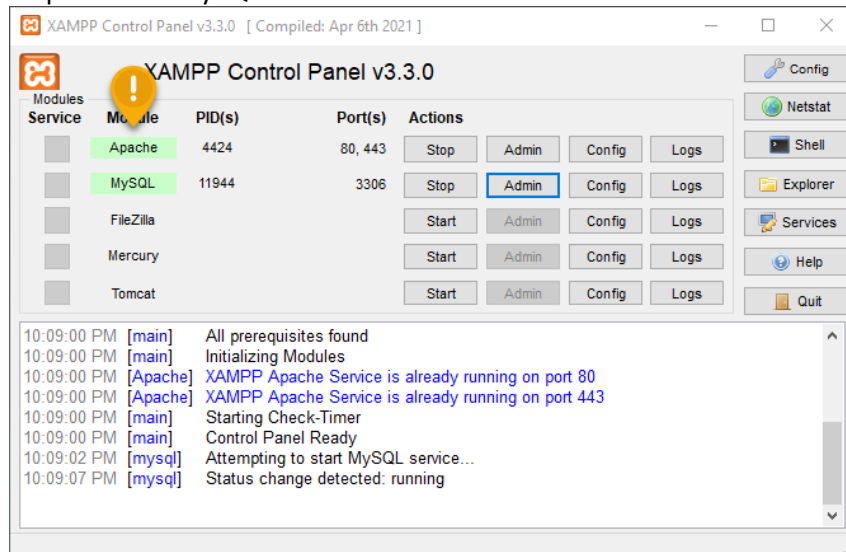
3. Instal XAMPP dari file installer yang telah didownload
4. Buka XAMPP



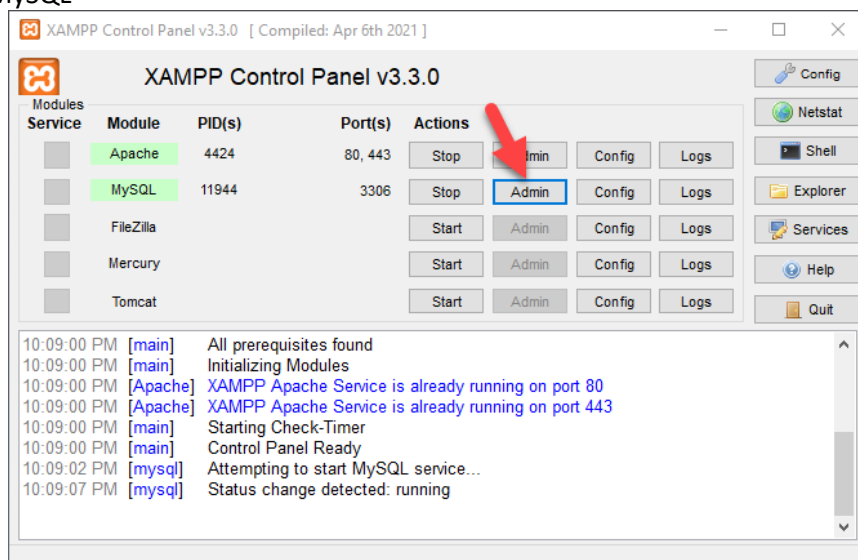
5. Start Apache dan MySQL



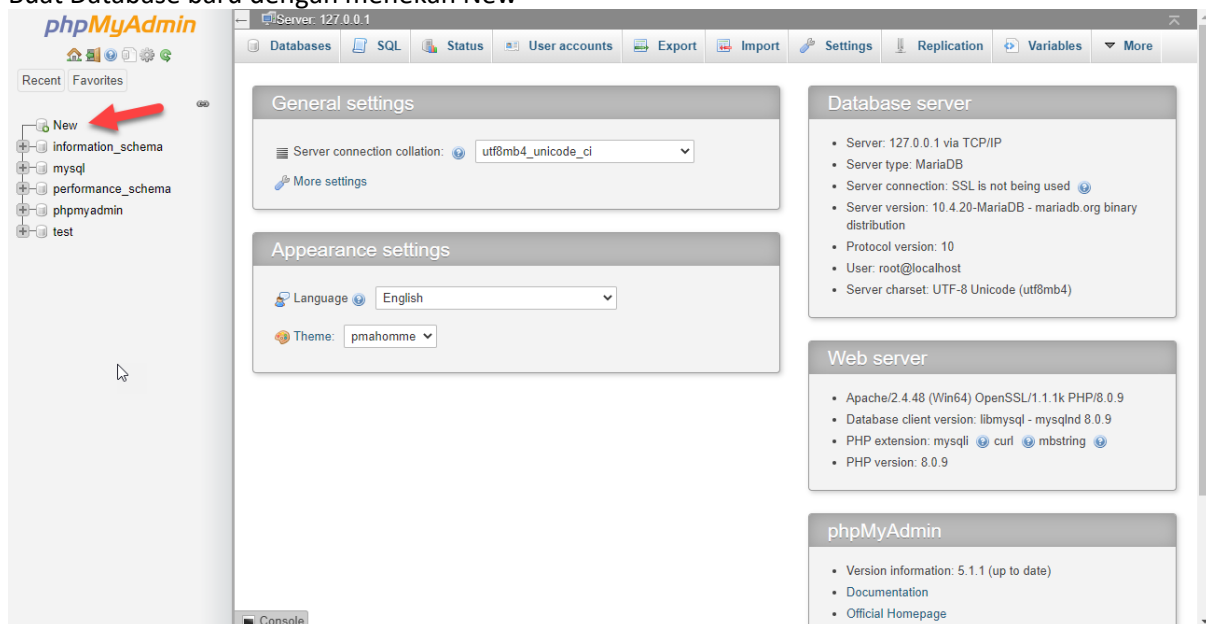
6. Pastikan modul Apache dan MySQL sudah berwarna **HIJAU**



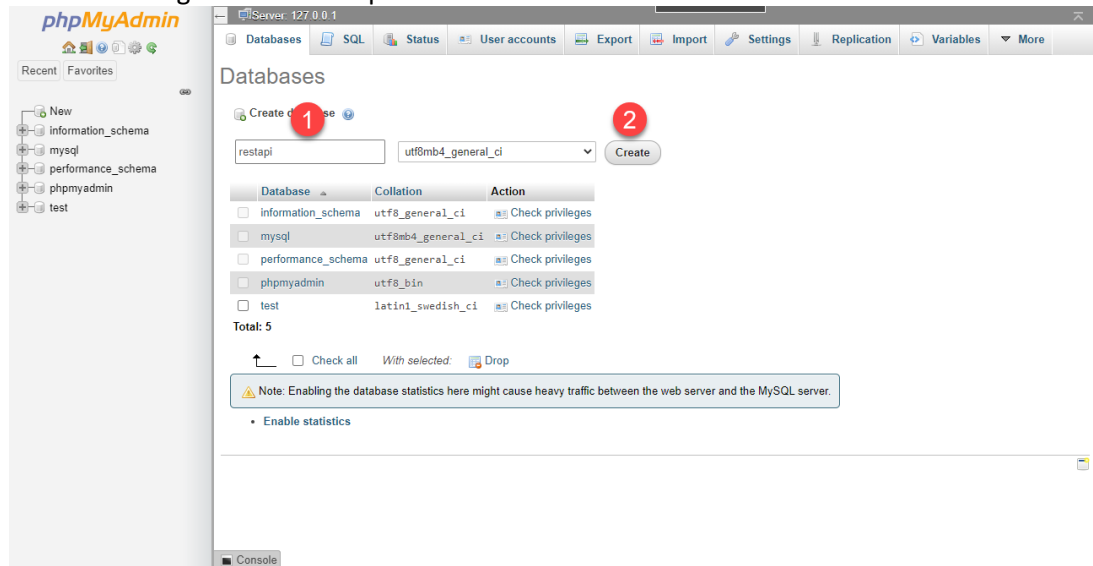
7. Buka Admin MySQL



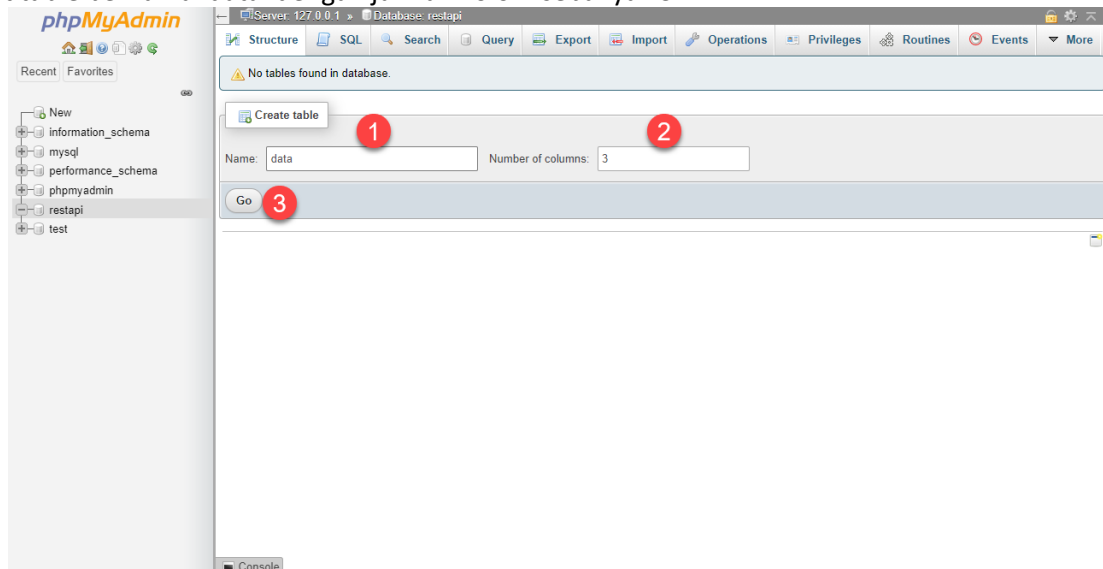
8. Buat Database baru dengan menekan New



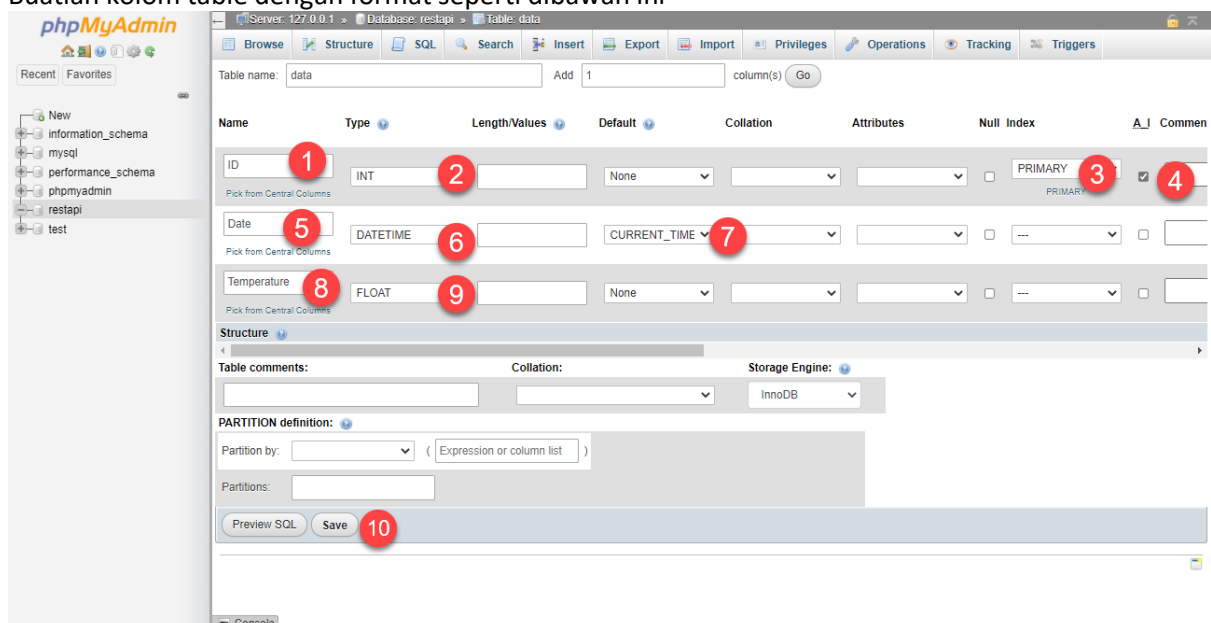
9. Buat Database dengan nama 'restapi' dan tekan create



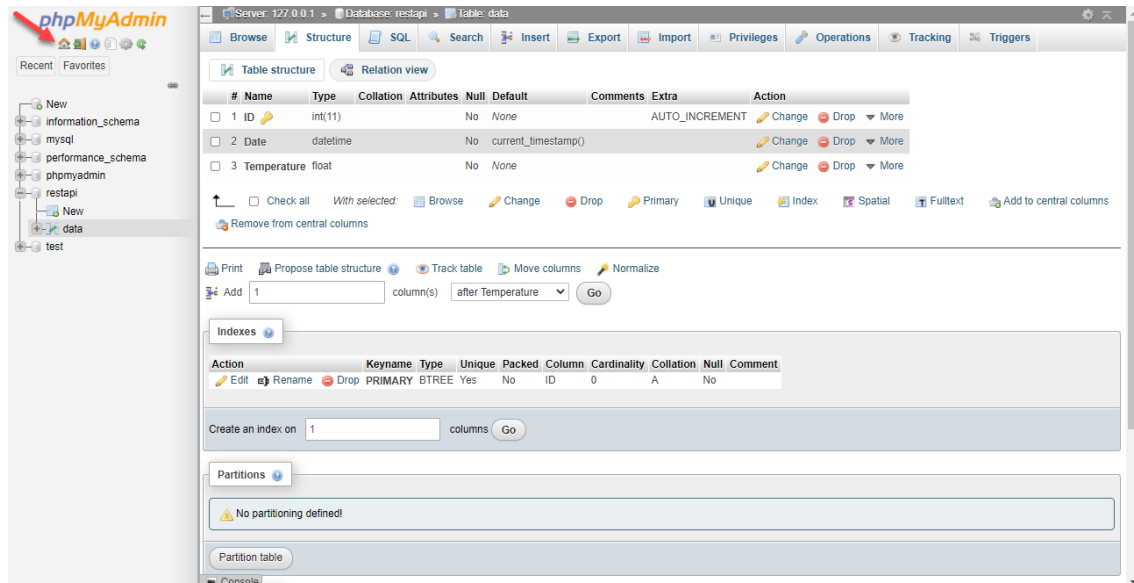
10. Buat table bernama 'data' dengan jumlah kolom sebanyak 3



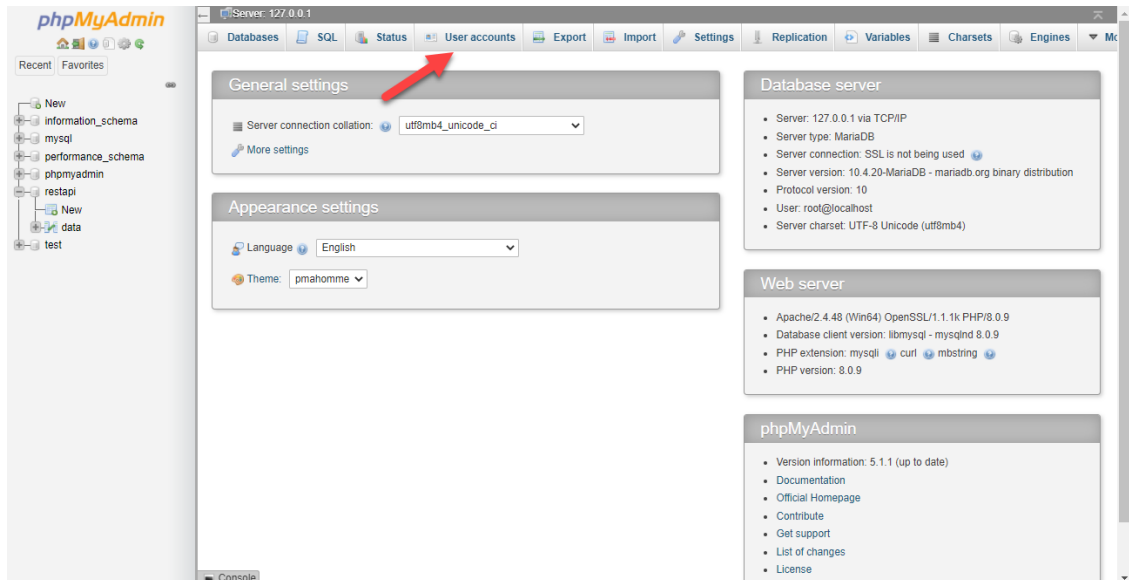
11. Buatlah kolom table dengan format seperti dibawah ini



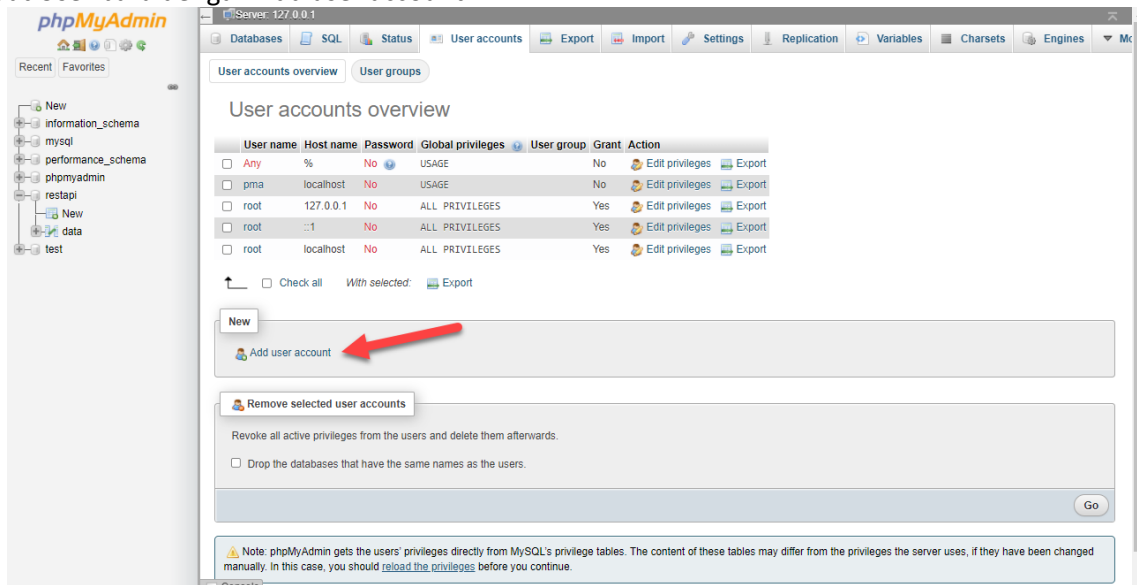
12. Jika sudah selesai Kembali ke home



13. Buka User accounts



14. Buat User baru dengan Add user account



15. Buat user dengan username RestDemo dan pass demoRest, serta check all untuk global privileges

The screenshot shows the 'Add user account' form in phpMyAdmin. The form is divided into several sections: 'Login Information', 'Database for user account', 'Global privileges', 'Resource limits', and 'SSL'. Red circles with numbers 1 through 5 highlight specific fields and buttons. Annotation 1 points to the 'User name' field containing 'RestDemo'. Annotation 2 points to the 'Password' field containing 'demoRest'. Annotation 3 points to the 'Re-type' field. Annotation 4 points to the 'Check all' checkbox under 'Global privileges'. Annotation 5 points to the 'Go' button at the bottom of the form. The 'Global privileges' section is expanded, showing checkboxes for various privileges like SELECT, INSERT, UPDATE, DELETE, FILE, CREATE, ALTER, INDEX, DROP, CREATE TEMPORARY TABLES, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, EXECUTE, CREATE VIEW, EVENT, TRIGGER, GRANT, SUPER, PROCESS, RELOAD, SHUTDOWN, SHOW DATABASES, LOCK TABLES, REFERENCES, REPLICATION CLIENT, REPLICATION SLAVE, and CREATE USER. The 'Resource limits' section has input fields for MAX QUERIES PER HOUR, MAX UPDATES PER HOUR, MAX CONNECTIONS PER HOUR, and MAX USER_CONNECTIONS, all set to 0. The 'SSL' section has radio buttons for REQUIRE NONE (selected), REQUIRE SSL, REQUIRE X509, and SPECIFIED, along with input fields for REQUIRE CIPHER, REQUIRE ISSUER, and REQUIRE SUBJECT.

phpMyAdmin

Server: 127.0.0.1

Databases SQL Status User accounts Export Import Settings Replication Variables Charsets Engines

Add user account

Login Information

User name: 1

Host name: % 2

Password: 3

Re-type:

Authentication plugin:

Generate password:

Database for user account

☐ Create database with same name and grant all privileges.
☐ Grant all privileges on wildcard name (username_%).

Global privileges ☒ Check all 4

Note: MySQL privilege names are expressed in English.

Data

- ☒ SELECT
- ☒ INSERT
- ☒ UPDATE
- ☒ DELETE
- ☒ FILE

Structure

- ☒ CREATE
- ☒ ALTER
- ☒ INDEX
- ☒ DROP
- ☒ CREATE TEMPORARY TABLES
- ☒ SHOW VIEW
- ☒ CREATE ROUTINE
- ☒ ALTER ROUTINE
- ☒ EXECUTE
- ☒ CREATE VIEW
- ☒ EVENT
- ☒ TRIGGER

Administration

- ☒ GRANT
- ☒ SUPER
- ☒ PROCESS
- ☒ RELOAD
- ☒ SHUTDOWN
- ☒ SHOW DATABASES
- ☒ LOCK TABLES
- ☒ REFERENCES
- ☒ REPLICATION CLIENT
- ☒ REPLICATION SLAVE
- ☒ CREATE USER

Resource limits

Note: Setting these options to 0 (zero) removes the limit.

MAX QUERIES PER HOUR:

MAX UPDATES PER HOUR:

MAX CONNECTIONS PER HOUR:

MAX USER_CONNECTIONS:

SSL

☒ REQUIRE NONE
☐ REQUIRE SSL
☐ REQUIRE X509
☐ SPECIFIED

REQUIRE CIPHER:

REQUIRE ISSUER:

REQUIRE SUBJECT:

5

Console

16. Setup database telah selesai

Langkah membuat server untuk REST

1. Install Python
2. Install dependensi flask, flask-restful, pandas, dan mysql.connector dengan:

```
pip install -U flask flask-restful pandas mysql.connector
```

3. Jika telah menginstall dependensi diatas, salin kode dibawah ini:

```
import mysql.connector
from datetime import date
import pandas as pd

today = date.today().strftime("%Y-%m-%d")

mydb = mysql.connector.connect(
    host = "localhost",
    user = "RESTdemo",
    password = "demoRest",
    database = "restapi"
)

def postData(sensorVal):
    mycursor = mydb.cursor()
    sql = "INSERT INTO data (Temperature) VALUES (%f)" %(sensorVal)
    mycursor.execute(sql)
    mydb.commit()

def getTodayData():
    mycursor = mydb.cursor()
    sql = "SELECT * FROM data WHERE CAST(Date AS DATE) = CAST('%s' AS DATE)
ORDER BY ID" %(str(today))
    mycursor.execute(sql)
    return mycursor.fetchall()
```

4. Simpan kode diatas dengan nama 'sqlConnector.py'
5. Salin juga kode dibawah ini:

```
from flask import Flask
from flask_restful import Api, Resource, reqparse
import pandas as pd
import sqlConnector

app = Flask(__name__)
app.config["DEBUG"] = True
api = Api(app)

class ESP(Resource):
    def get(self):
        data = sqlConnector.getTodayData()
        data = pd.DataFrame(data, columns=["ID", "Date", "Temp"])
        data["Date"] = data["Date"].dt.strftime("%Y-%m-%d %H:%M:%S")
        data = data.to_dict('records')
        return {'data' : data}, 200

    def post(self):
        parser = reqparse.RequestParser()
        parser.add_argument('temperature', required=True)
```

```

args = parser.parse_args()
new_data = pd.DataFrame({
    'temperature' : [args['temperature']],
})
sqlConnector.postData(float(args['temperature']))
return {'data' : new_data.to_dict('records')}, 201

# Add URL endpoints
api.add_resource(ESP, '/esp')

if __name__ == '__main__':
    app.run()

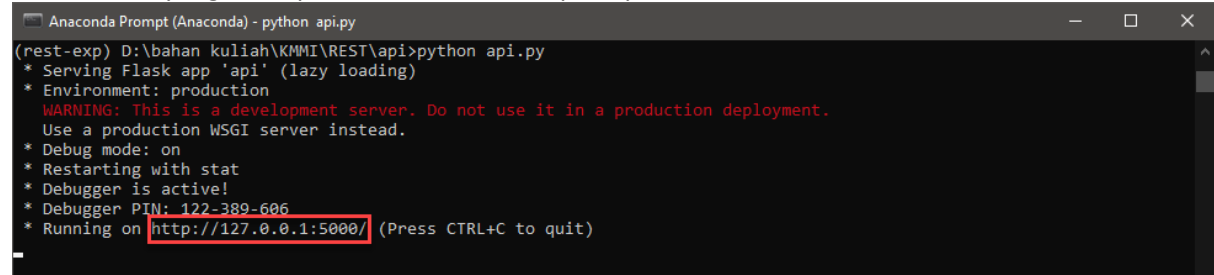
```

6. Simpan dengan nama 'api.py' di folder yang sama dengan 'sqlConnector.py'
7. Jika sudah, jalankan program 'api.py' dengan perintah:

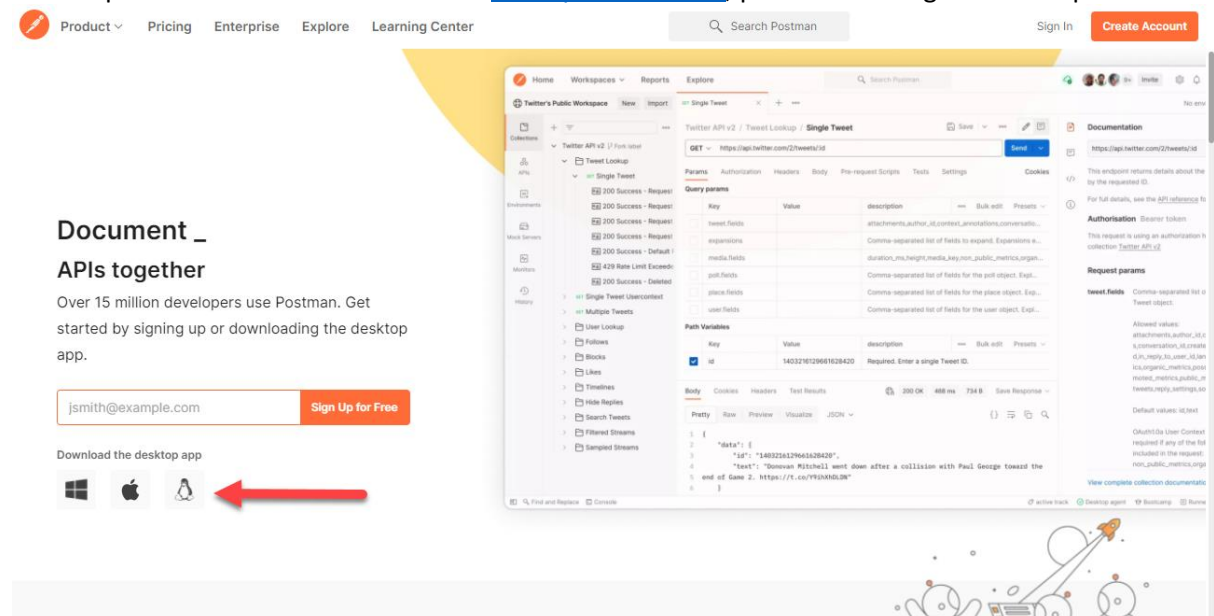
```
python api.py
```



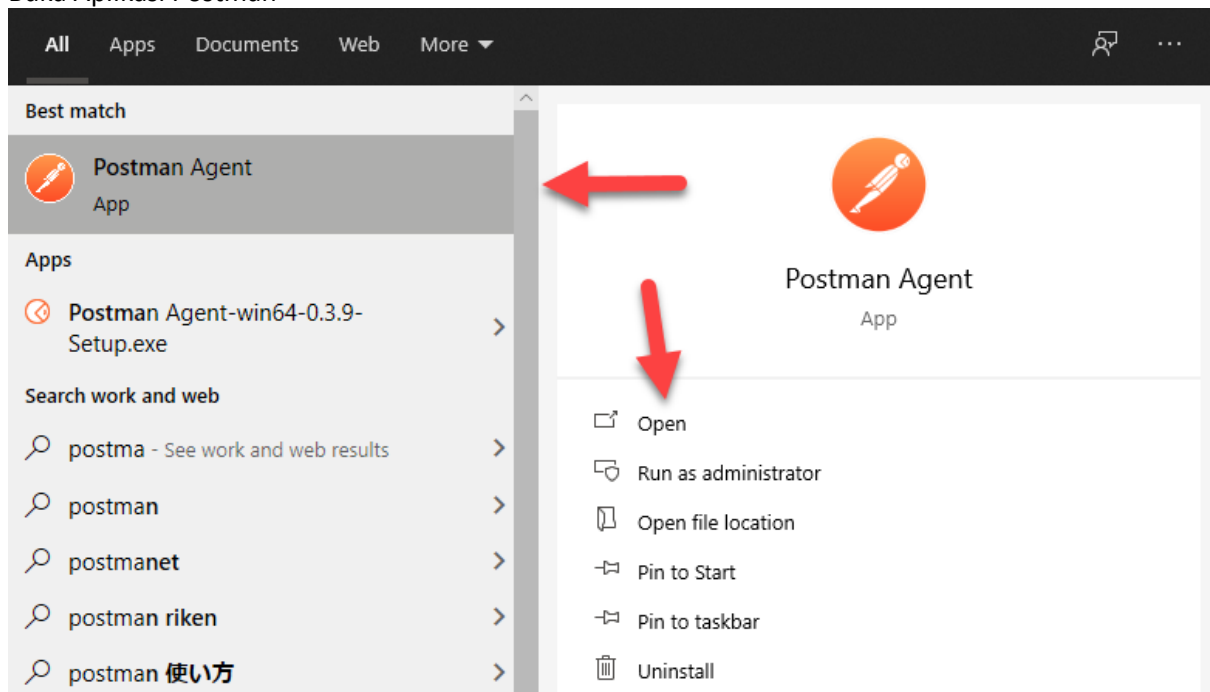
8. Salin alamat yang ditampilkan oleh command prompt



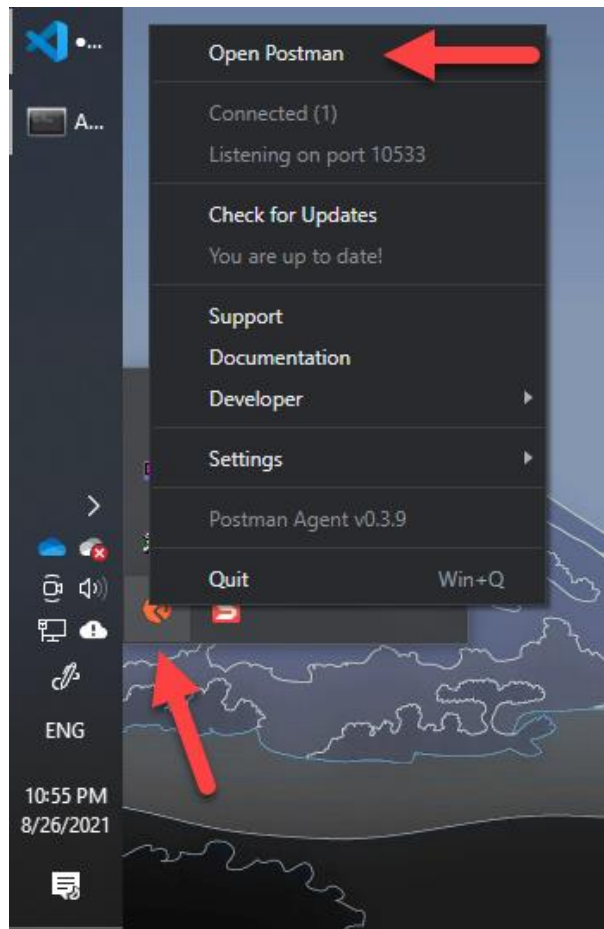
9. Install aplikasi Postman dari link berikut www.postman.com, pilih sesuai dengan sistem operasi Anda



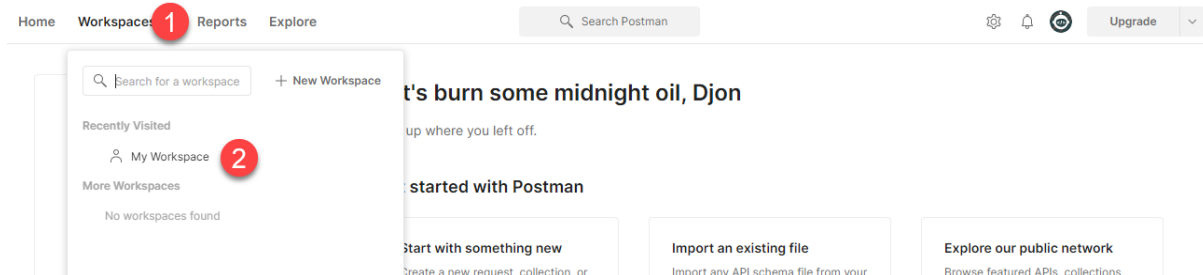
10. Buka Aplikasi Postman



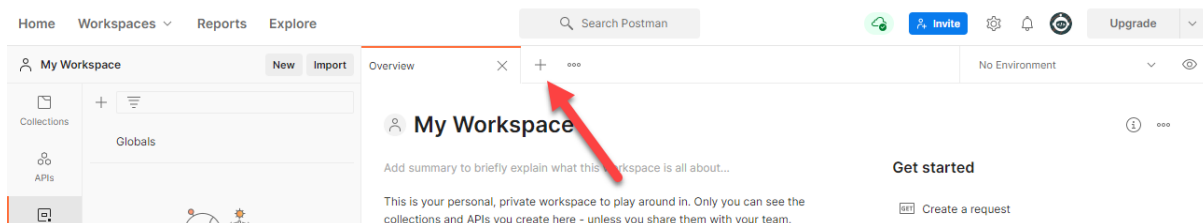
11. Jika tidak muncul tampilan postman, silahkan cek di 'system tray' windows, klik kanan pada ikon postman dan klik 'Open Postman'



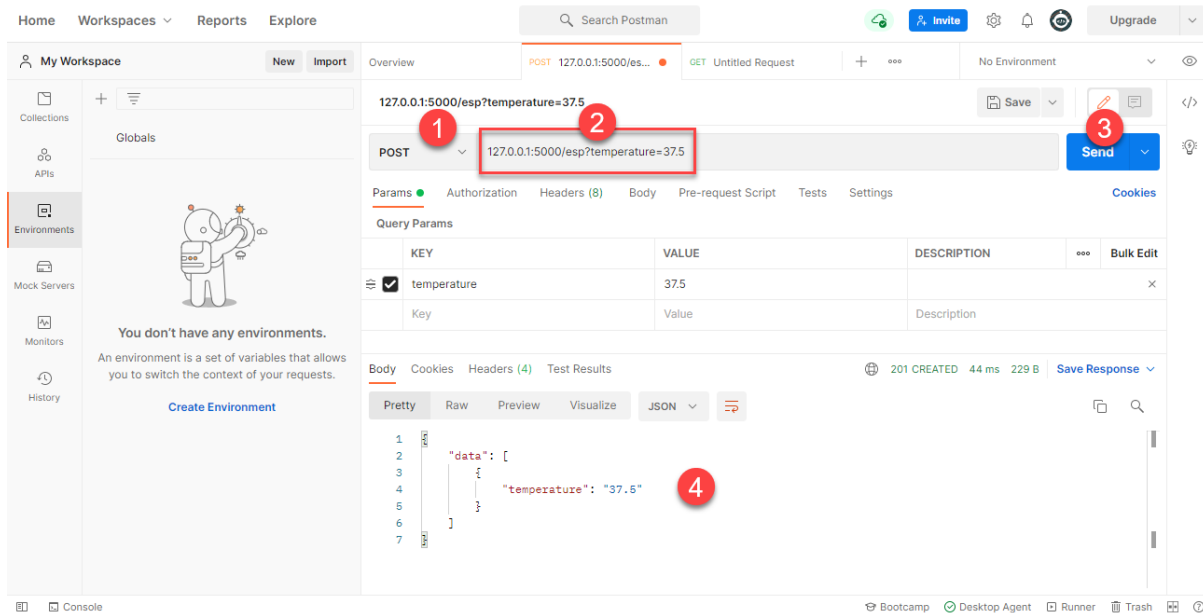
12. Jika tampilan postman sudah terbuka, buka lembar workspace



13. Jika lembar workspace telah terbuka, tekan tombol '+'

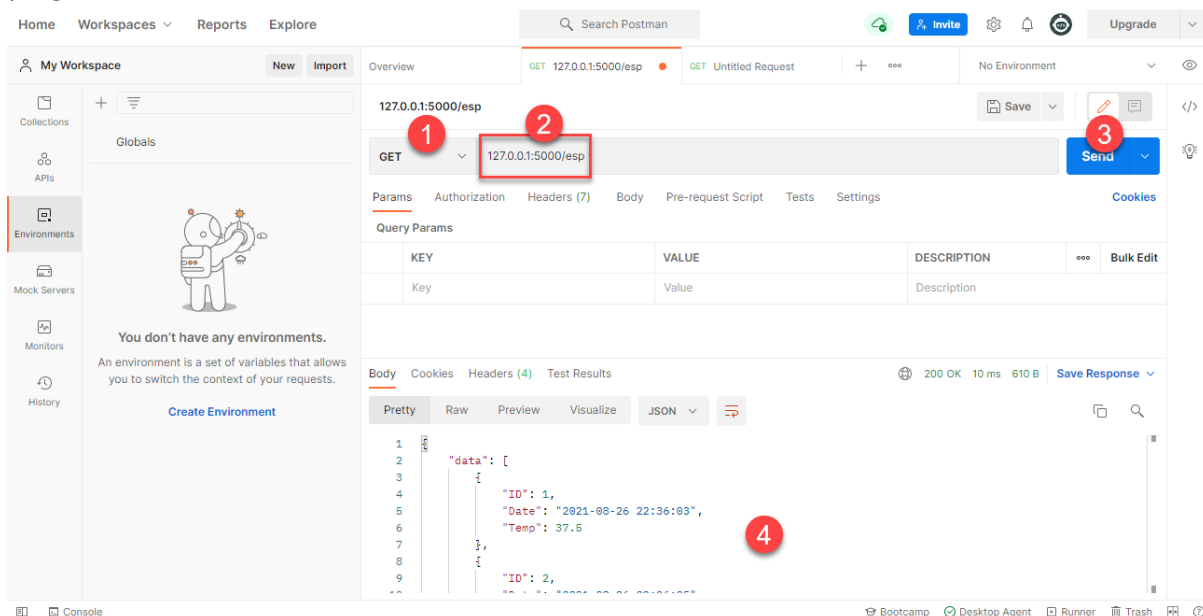


14. Masukkan alamat yang telah didapatkan pada step 8 pada kolom yang ditandai merah serta tambahkan '/esp?temperature=37.5' diakhir alamat, dan pastikan perintah yang dimasukkan adalah POST, dan tekan send

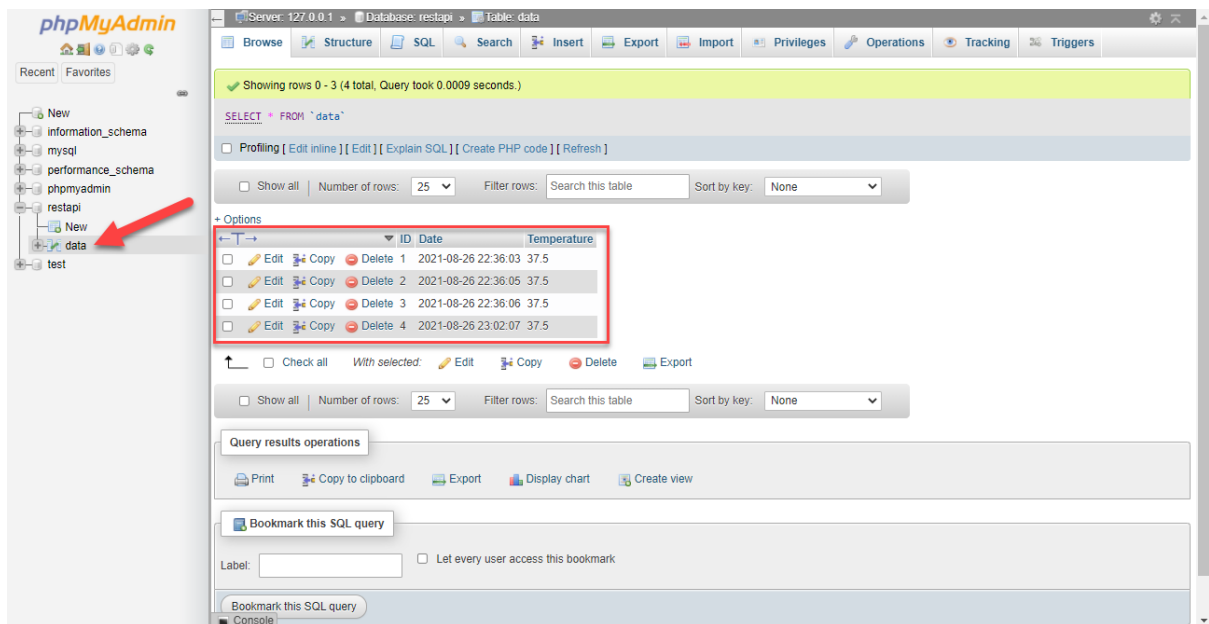


15. Jika data berhasil masuk ke database maka muncul tampilan seperti di step 14 point 4.

16. Untuk mengecek data yang sudah ada di database masukan alamat yang telah didapatkan pada step 8 pada kolom yang ditandai merah serta tambahkan '/esp' diakhir alamat, dan pastikan perintah yang dimasukan adalah GET, dan tekan send



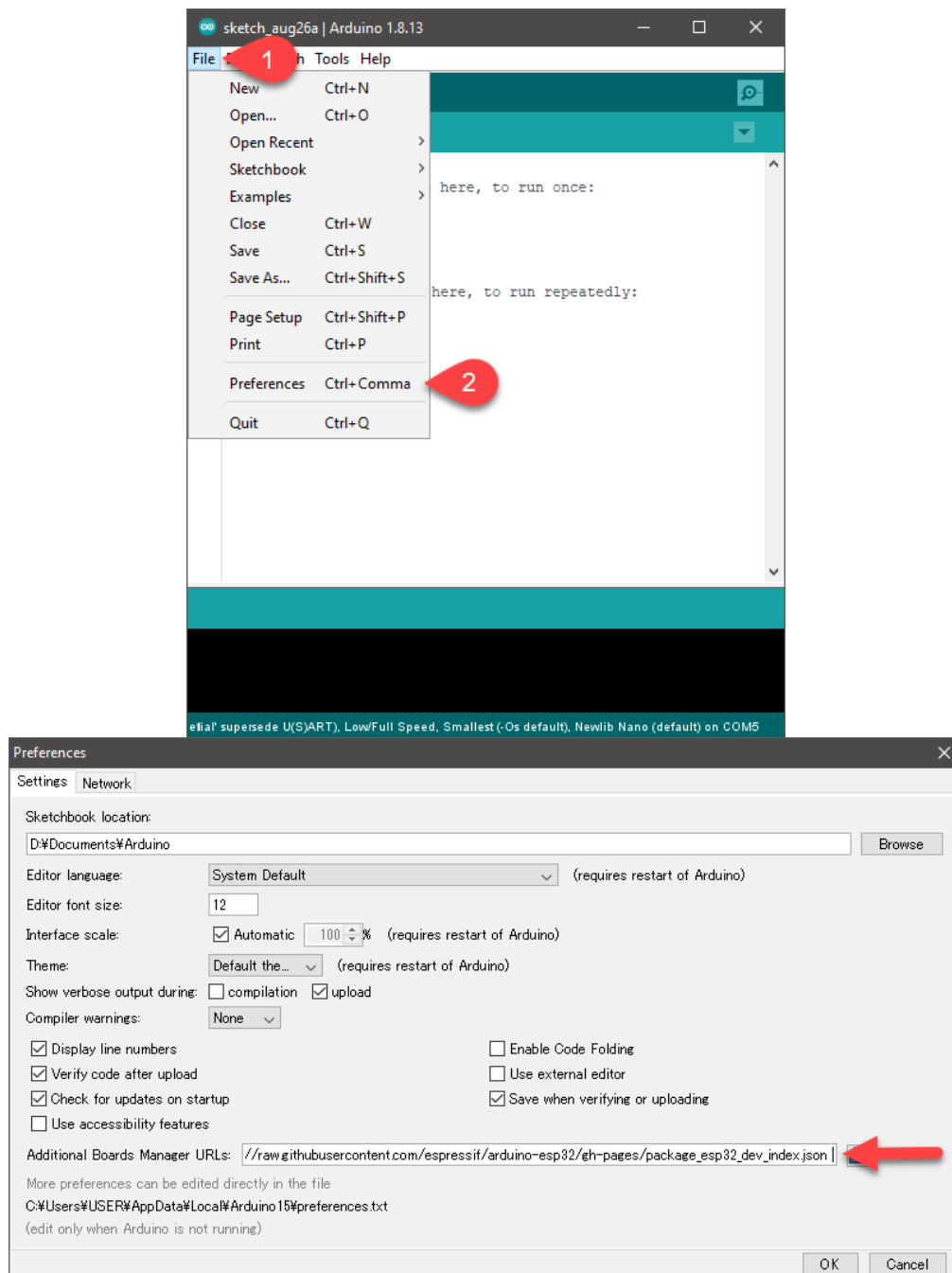
17. Data juga dapat dilihat pada halaman admin mySQL



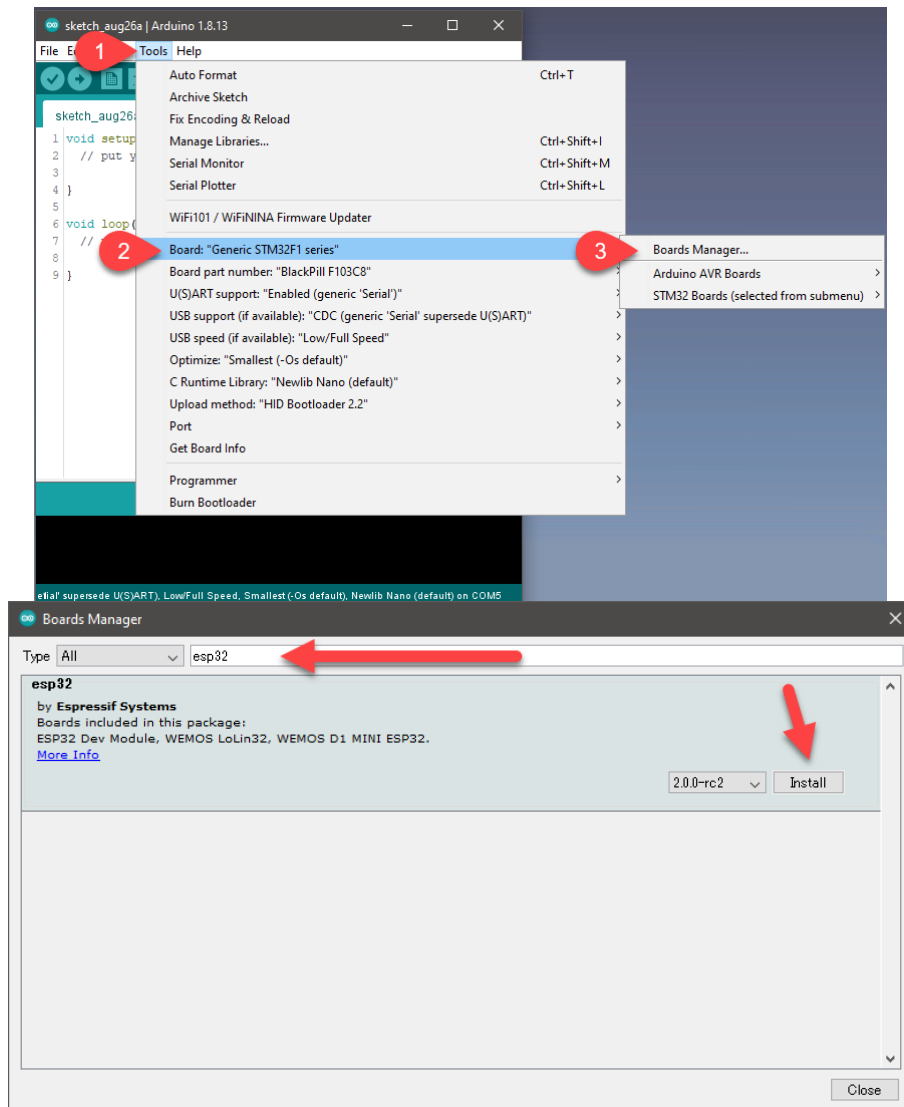
18. Pembuatan server REST telah berhasil, dan data dapat ditransmisikan

Langkah Pemrograman ESP32 untuk mengakses server REST

1. Instal Arduino
2. Instal Board ESP32 dengan menambahkan Board Manager Arduino dengan alamat berikut:
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json
 Alamat tersebut dapat dilihat di: <https://github.com/espressif/arduino-esp32>



3. Kemudian install Board ESP32 dari Board Manager



4. Jika sudah selesai maka salin kode dibawah ini

```
#include <WiFi.h>
#include <HTTPClient.h>

#ifdef __cplusplus
extern "C" {
#endif
uint8_t temprature_sens_read();
#ifdef __cplusplus
}
#endif
uint8_t temprature_sens_read();

//Masukan ssid dan password dari router
const char* ssid = "ssid";
const char* password = "pass";

//Alamat IP server dengan portnya
```

```

unsigned long lastTime = 0;
unsigned long timerDelay = 1000;

void setup() {
    Serial.begin(115200);

    WiFi.begin(ssid, password);
    Serial.println("Connecting");
    while(WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.print("Connected to WiFi network with IP Address: ");
    Serial.println(WiFi.localIP());
}

void loop() {
    if ((millis() - lastTime) > timerDelay) {
        //cek koneksi
        if(WiFi.status()== WL_CONNECTED){
            WiFiClient client;
            HTTPClient http;
            float temp = (temperature_sens_read() - 32) / 1.8;
            char serverName[100];
            sprintf(serverName,"http://192.168.0.102:5000/esp?temperature=%3.2f",
temp);
            http.begin(client, serverName);

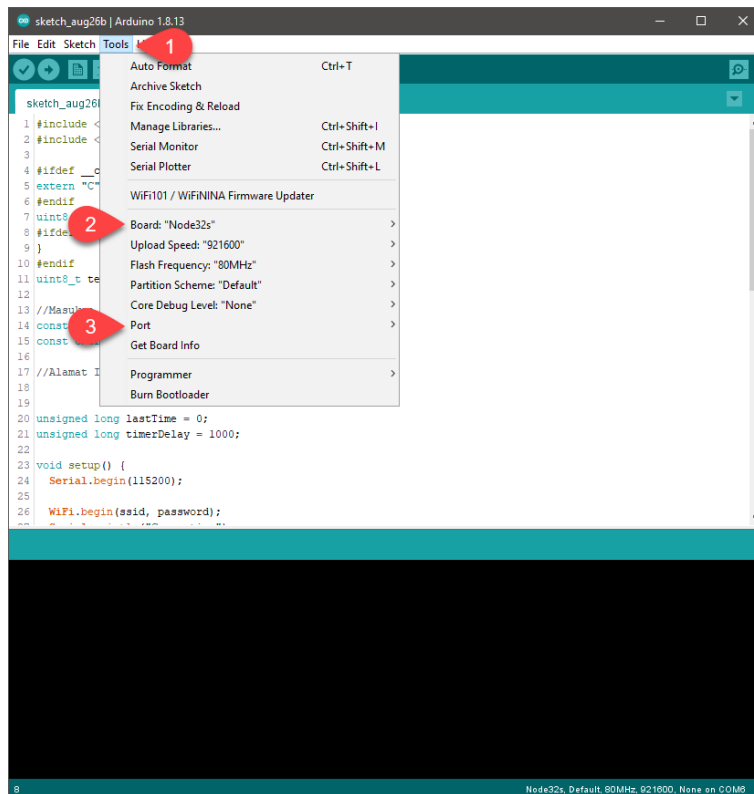
            // kirimkan HTTP POST
            int httpResponseCode = http.POST();

            Serial.print("HTTP Response code: ");
            Serial.println(httpResponseCode);

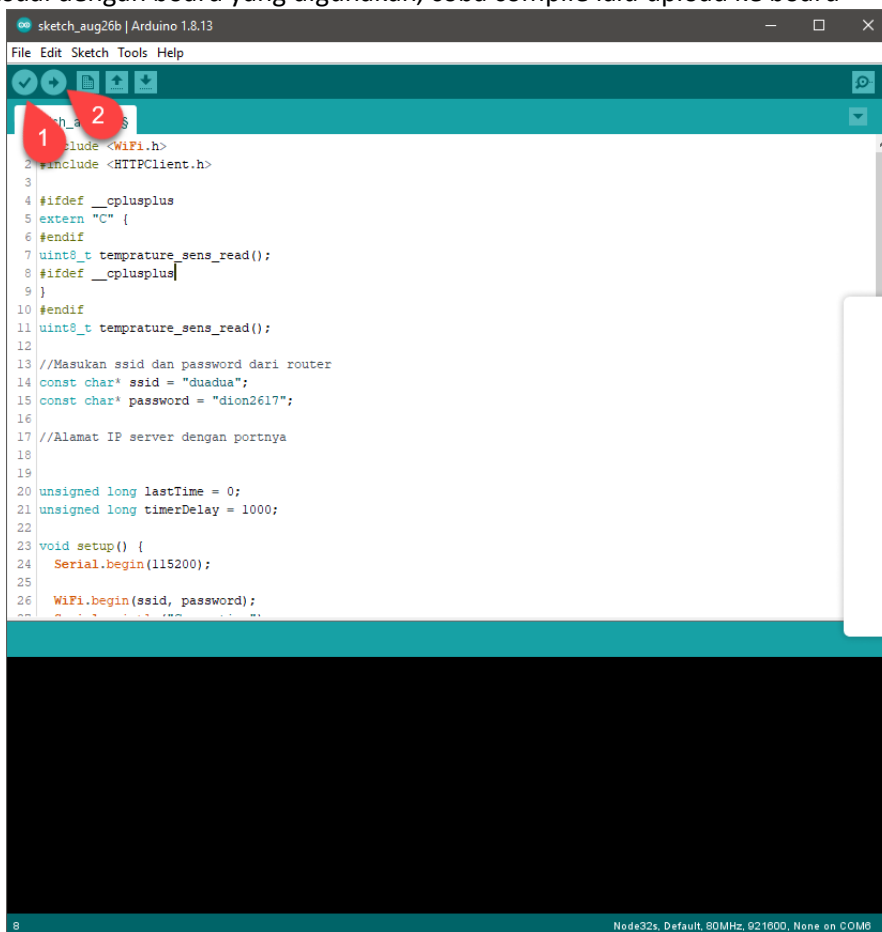
            http.end();
        }
        else {
            Serial.println("WiFi Disconnected");
        }
        lastTime = millis();
    }
}

```

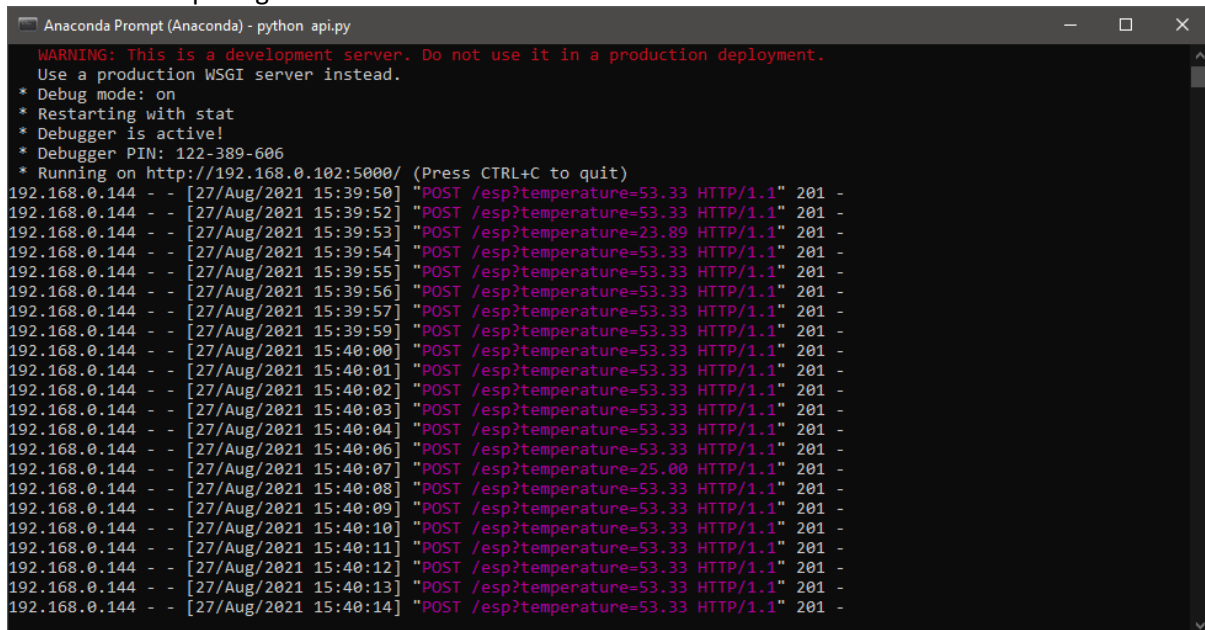
5. Set Board dan Port yang digunakan



6. Jika sudah sesuai dengan board yang digunakan, coba compile lalu upload ke board



7. Jika program telah terupload ke Board, perhatikan command prompt dari api.py, jika koneksi berhasil akan muncul seperti gambar berikut:



```
Anaconda Prompt (Anaconda) - python api.py
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 122-389-606
* Running on http://192.168.0.102:5000/ (Press CTRL+C to quit)
192.168.0.144 - - [27/Aug/2021 15:39:50] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:52] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:53] "POST /esp?temperature=23.89 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:54] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:55] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:56] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:57] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:39:59] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:00] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:01] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:02] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:03] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:04] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:06] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:07] "POST /esp?temperature=25.00 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:08] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:09] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:10] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:11] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:12] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:13] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
192.168.0.144 - - [27/Aug/2021 15:40:14] "POST /esp?temperature=53.33 HTTP/1.1" 201 -
```

8. Proses koneksi telah berjalan dan server mampu menerima data.