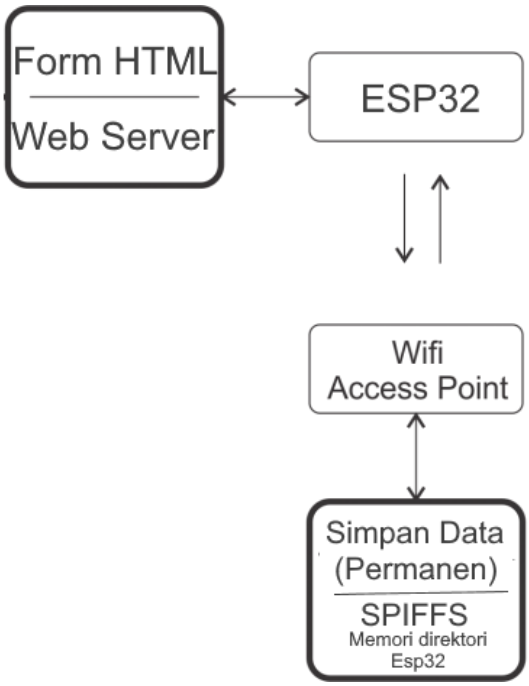
INTERNET OF THINGS LANJUT

Projek 2 (Input Data pada Web Server Menggunakan Form HTML Disimpan Permanen pada Memori ESP32/SPIFFS)

Konsep Projek 2 :

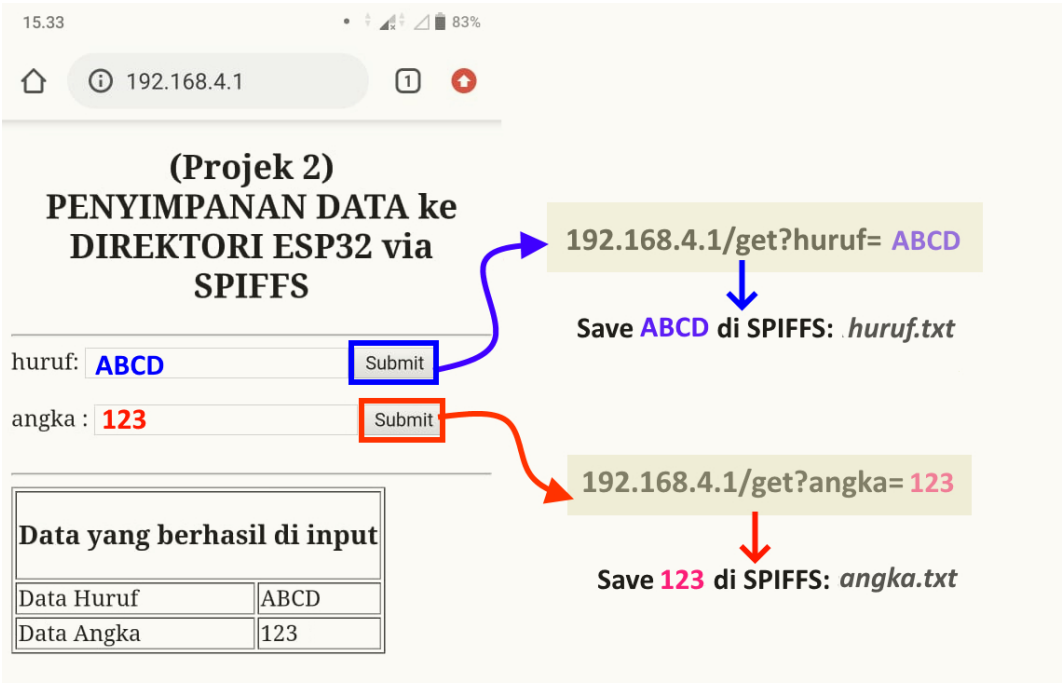


Dalam projek kedua ini, penulis akan mempelajari cara membuat server web ESP32 dengan dua jenis tipe data inputan untuk meneruskan nilai ke ESP menggunakan form HTML dan disimpan ke memori internal ESP32 berupa file txt. Kemudian, kita dapat menggunakan nilai tersebut sebagai variabel. Kita akan menggunakan Arduino IDE untuk memprogram ESP32 Board.

Kita dapat mengaksesnya dengan browser apa pun di jaringan yang kita punya. Saat kita mengetik nilai baru dan menekan tombol "Submit", ESP32 akan memperbarui variabel dengan nilai baru.

Jika kita perlu memperbarui variabel melalui webserver ESP, kita bisa belajar dari proyek sederhana ini. Dengan metode ini, kita bisa menghindari variabel pengkodean rumit karena cukup hanya dengan membuat bidang input di halaman web kita sudah bisa memperbarui variabel apa pun dengan nilai baru. Ini dapat sangat berguna untuk mengatur nilai ambang batas, mengatur SSID / kata sandi, mengubah Kunci API, dll ...

Disini penulis akan menunjukkan cara menyimpan variabel tersebut secara permanen menggunakan SPIFFS dan cara mengaksesnya. Berikut cara kerjanya.



Halaman web ini memungkinkan kita untuk menginputkan dua jenis variabel: String, dan Int. Kemudian, setiap kali kita mengirimkan nilai baru, nilai tersebut disimpan dalam file SPIFFS. Halaman web ini juga berisi tempat penampung untuk memperlihatkan nilai saat ini.

Prasyarat

Pastikan kita memeriksa semua prasyarat di bagian ini sebelum melanjutkan proyek untuk mengompilasi kode.

1. Instal ESP32 Board di Arduino IDE
   Kita akan memprogram ESP32 menggunakan Arduino IDE. Jadi, kita harus menginstal Add-On ESP32.

1. Menginstal Library
   Untuk membangun asynchronous web server, kita perlu memasang library ini.

2. ESP32: instal ESPAsyncWebServer dan library AsyncTCP.
   Library ini tidak tersedia jika kita mengakses langsung melalui Manajer Library Arduino, jadi kita perlu menyalin file library ke folder Instalasi Arduino. Atau, di Arduino IDE kita dapat langsung membuka Sketch> Include Library> Add .zip Library dan memilih file library yang baru saja kita unduh.

2. Bagian yang Diperlukan
   Untuk mengikuti projek ini kita hanya membutuhkan ESP32. Tidak ada sirkuit lain untuk proyek ini.

Projek ini akan menyimpan data yang disisipkan di kolom input secara permanen di SPIFFS. Kita juga akan menambahkan placeholder di halaman web untuk menampilkan nilai saat ini.

Salin sketsa berikut ke Arduino IDE. Kemudian, sebelum mengunggah ketikkan kredensial jaringan Anda (SSID dan kata sandi).

```
#include <ESPmDNS.h>
#include <ESPAsyncWebServer.h>
#include <AsyncTCP.h>
#include <SPIFFS.h>

AsyncWebServer server(80); // server port 80

const char index_html[] PROGMEM = R"rawliteral(
<!DOCTYPE HTML><html><head>
  <title>Database via SPIFFS</title>
  <meta name="viewport" content="width=device-width, initial-
scale=1">
  <script>
    function PesanSubmit() {
      alert("Data akan disimpan");
      setTimeout(function(){ document.location.reload(false); },
500);
    }
  </script></head><body>

  <H2><b><center>(Projek 2)<br>PENYIMPANAN DATA ke DIREKTORI
ESP32 via SPIFFS</center></b></H2>
  <hr>

   <form action="/get" target="hidden-form" class="center">
    huruf: <input type="text" name="huruf">
    <input type="submit" placeholder="Masukkan Data Huruf"
value="Submit" onclick="PesanSubmit()">
  </form><br>
  <form action="/get" target="hidden-form" class="center">
    angka : <input type="number " name="angka">
    <input type="submit" placeholder="Masukkan Data Angka"
value="Submit" onclick="PesanSubmit()">
  </form><br>
  <iframe style="display:none" name="hidden-form"></iframe>
  <hr>

  <table border="1" class="center">
    <tr>
      <td colspan="2"><H3><b><center>Data yang berhasil di
input</center></b></H3></td>
    </tr>
      <td>Data Huruf</td>
      <td>%huruf%</td>
    </tr>
    <tr>
      <td>Data Angka</td>
      <td>%angka%</td>
  </table>
```

```
</body></html>)rawliteral";

const char* INPUT_STRING = "huruf";
const char* INPUT_INT = "angka";
int LED_BUILTIN = 2;
String readFile(fs::FS &fs, const char * path){
  File file = fs.open(path, "r");
  if(!file || file.isDirectory()){
    Serial.println("- File kosong atau tidak ditemukan");
    return String();
  }
  String fileContent;
  while(file.available()){
    fileContent+=String((char)file.read());
  }
  Serial.println(fileContent);
  return fileContent;
}

void writeFile(fs::FS &fs, const char * path, const char *
message){
  Serial.printf("Writing file: %s\r\n", path);
  File file = fs.open(path,"a+");
  if(!file){
    Serial.println("- failed to open file for writing");
    return;
  }
  if(file.print(message)){
    Serial.println("- isi file berhasil ditulis");
  } else {
    Serial.println("- Penulisan Gagal");
  }
}

// Replaces placeholder with stored values
String processor(const String& var){
  //Serial.println(var);
  if(var == "huruf"){
    return readFile(SPIFFS, "/huruf.txt");
  }
  else if(var == "angka"){
    return readFile(SPIFFS, "/angka.txt");
  }
  return String();
}

void notFound(AsyncWebServerRequest *request)
{
  request->send(404, "text/plain", "not found");
}

void setup(void)
{

  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  WiFi.softAP("FiryanulRizky ESP Wifi", "");
  Serial.println("FiryanulRizky ESP Wifi");
```

```
  Serial.println(WiFi.softAPIP());


  if (MDNS.begin("esp")) { //esp.local/
    Serial.println("MDNS responder started");
  }

  if(!SPIFFS.begin(true)){
      Serial.println("Error saat memuat SPIFFS");
      return;
    }

  // Membuka file HTML ke client
  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/html", index_html, processor);
  });

  server.on("/get", HTTP_GET, [] (AsyncWebServerRequest *request)
{
    String inputMessage;
    // Mengakses data <ESP_IP>/get?huruf=<inputMessage>
    if (request->hasParam(INPUT_STRING)) {
      inputMessage = request->getParam(INPUT_STRING)->value();
      writeFile(SPIFFS, "/huruf.txt", inputMessage.c_str());
      digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH
is the voltage level)
      delay(500);                       // wait for a second
      digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by
making the voltage LOW
      delay(500);                       // wait for a second
    }
    // Mengakses data <ESP_IP>/get?angka=<inputMessage>
    else if (request->hasParam(INPUT_INT)) {
      inputMessage = request->getParam(INPUT_INT)->value();
      writeFile(SPIFFS, "/angka.txt", inputMessage.c_str());
       digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH
is the voltage level)
       delay(500);                       // wait for a second
       digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by
making the voltage LOW
       delay(500);                       // wait for a second
    }
    else {
      inputMessage = "Pesan gagal";
      digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by
making the voltage LOW
    }
    Serial.println(inputMessage);
    request->send(200, "text/text", inputMessage);
  });

  server.onNotFound(notFound);

  server.begin();  // it will start webserver
}


void loop(void)
{
```

```
  String datahuruf = readFile(SPIFFS, "/huruf.txt");
  Serial.print("*** Data huruf: ");
  Serial.println(datahuruf);

  int dataangka = readFile(SPIFFS, "/angka.txt").toInt();
  Serial.print("*** Data angka: ");
  Serial.println(dataangka);
  delay(5000);
}
```

Penjelasan per Baris Kode :

Ini adalah beberapa Library yang disebutkan, jika kita menggunakan ESP32. kita perlu memuat library SPIFFS untuk menulis ke SPIFFS.

```
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
#include <AsyncTCP.h>
#include <SPIFFS.h>
```

Mengubah ESP32 menjadi Access Point

Untuk mempermudah client mengakses webserver, maka disini kita buatkan esp32 Menjadi wifi (access point) dengan perintah/fungsi *softAP*, dan *MDNS Responder* agar client bisa mendapatkan IP default dari ESP32 yaitu 192.168.4.1, sehingga ketika user sebagai client ingin mengakses webserver, perlu menghubungkan access point esp dan memasukkan ip 192.168.4,1 pada web browser.

```
WiFi.softAP("FiryanulRizky ESP Wifi", "");
  Serial.println("FiryanulRizky ESP Wifi");
  Serial.println(WiFi.softAPIP());


  if (MDNS.begin("esp")) { //esp.local/
    Serial.println("MDNS responder started");
  }
```

LED pada pin 2 (LED bawaan ESP32) akan menyala setiap fungsi server.on() dijalankan, ini mengindikasikan ketika sistem berhasil menangkap inputan dari HTTP_GET, dan memberitahukan ke user bahwa data berhasil disimpan ke SPIFFS :

```
int LED_BUILTIN = 2;
pinMode(LED_BUILTIN, OUTPUT);

server.on("/get", HTTP_GET, [] (AsyncWebServerRequest *request) {
    String inputMessage;
    // Mengakses data <ESP_IP>/get?huruf=<inputMessage>
    if (request->hasParam(INPUT_STRING)) {
      inputMessage = request->getParam(INPUT_STRING)->value();
```

```
      writeFile(SPIFFS, "/huruf.txt", inputMessage.c_str());
      digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH
is the voltage level)
      delay(500);                       // wait for a second
      digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by
making the voltage LOW
      delay(500);                       // wait for a second
    }
    // Mengakses data <ESP_IP>/get?angka=<inputMessage>
    else if (request->hasParam(INPUT_INT)) {
      inputMessage = request->getParam(INPUT_INT)->value();
      writeFile(SPIFFS, "/angka.txt", inputMessage.c_str());
       digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH
is the voltage level)
       delay(500);                       // wait for a second
       digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by
making the voltage LOW
       delay(500);                       // wait for a second
    }
```

Form HTML

Dalam contoh ini, ketika kita mengirimkan nilai, akan ada jendela pop up terbuka yang mengatakan data telah disimpan ke SPIFFS.

Untuk itu, kita perlu menambahkan fungsi JavaScript, dalam hal ini disebut submitMessage () yang memunculkan pesan peringatan yang mengatakan bahwa nilai telah disimpan ke SPIFFS. Setelah itu muncul, maka halaman web akan dimuat ulang sehingga dapat menampilkan nilai saat ini.

```
<script>
    function PesanSubmit() {
      alert("Data akan disimpan");
      setTimeout(function(){ document.location.reload(false); },
500);
    }
</script>
```

Ini adalah form untuk memasukkan data.

```
<form action="/get" target="hidden-form" class="center">
    huruf: <input type="text" name="huruf">
    <input  type="submit"  placeholder="Masukkan  Data  Huruf"
value="Submit" onclick="PesanSubmit()">
  </form><br>
  <form action="/get" target="hidden-form" class="center">
    angka : <input type="number " name="angka">
    <input  type="submit"  placeholder="Masukkan  Data  Angka"
value="Submit" onclick="PesanSubmit()">
  </form><br>
  <iframe style="display:none" name="hidden-form"></iframe>
  <hr>
```

```
  <table border="1" class="center">
    <tr>
      <td colspan="2"><H3><b><center>Data yang berhasil di
input</center></b></H3></td>
    </tr>
      <td>Data Huruf</td>
      <td>%huruf%</td>
    </tr>
    <tr>
      <td>Data Angka</td>
      <td>%angka%</td>
  </table>
```

Dalam kasus ini, atribut target dan <iframe> digunakan sehingga kita tetap berada di halaman yang sama setelah mengirimkan data form.

Nama yang muncul untuk bidang input berisi variabel %huruf% %angka% yang akan menampilkan input nilai saat ini dari masing-masing variabel.

Onclick = "pesanSubmit()" memanggil fungsi JavaScript pesanSubmit() setelah mengklik tombol "Submit".

Membaca dan Menulis ke SPIFFS
Kemudian, kita punya beberapa fungsi untuk membaca dan menulis dari SPIFFS.

ReadFile () membaca konten dari file:

```
String readFile(fs::FS &fs, const char * path){
  File file = fs.open(path, "r");
  if(!file || file.isDirectory()){
    Serial.println("- File kosong atau tidak ditemukan");
    return String();
  }
  String fileContent;
  while(file.available()){
    fileContent+=String((char)file.read());
  }
  Serial.println(fileContent);
  return fileContent;
}
```

WriteFile () menulis konten ke file :

```
void writeFile(fs::FS &fs, const char * path, const char *
message){
  Serial.printf("Writing file: %s\r\n", path);

  File file = fs.open(path, "a+");

  if(!file){
    Serial.println("- failed to open file for writing");
    return;
  }
  if(file.print(message)){
    Serial.println("- isi file berhasil ditulis");
```

```
  } else {
    Serial.println("- Penulisan Gagal");
  }
}
```

Fungsi *Prosesor()* bertanggung jawab untuk mencari placeholder dalam teks HTML dan akan selalu membaca nilai aktual yang disimpan di SPIFFS.

```
String processor(const String& var){
  //Serial.println(var);
  if(var == "huruf"){
    return readFile(SPIFFS, "/huruf.txt");
  }
  else if(var == "angka"){
    return readFile(SPIFFS, "/angka.txt");
  }
  return String();
}
```

Kemudian, kita perlu menangani permintaan HTTP GET.

Saat kita mengakses URL, kita mengirim halaman HTML ke klien. Dalam kasus ini, teks HTML disimpan di variabel index_html.

```
// Membuka file HTML ke client
  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/html", index_html, processor);
  });
```

Kemudian, Anda perlu menangani apa yang terjadi ketika Anda menerima permintaan pada rute / get.

```
server.on("/get", HTTP_GET, [] (AsyncWebServerRequest *request) {
```

Dan selanjutnya akan menyimpan data variabel di SPIFFS.

Misalnya, untuk form input Data Huruf :

```
if (request->hasParam(INPUT_STRING)) {
      inputMessage = request->getParam(INPUT_STRING)->value();
      writeFile(SPIFFS, "/huruf.txt", inputMessage.c_str());
```

Ketika permintaan request berisi huruf (yaitu PARAM_STRING), kita akan memprogram variabel inputMessage ke nilai yang dikirimkan pada form huruf.

```
inputMessage = request->getParam(INPUT_STRING)->value();
```

Kemudian, simpan nilai tersebut ke SPIFFS.

```
writeFile(SPIFFS, "/huruf.txt", inputMessage.c_str());
```

Proses serupa terjadi untuk form angka.

Mengakses data/variabel Melalui Serial Monitor

Dengan adanya fungsi loop(), kita dapat mengakses dan melihat variabel/data dari Serial Monitor Arduino IDE..

Misal untuk variabel String dataHuruf yang akan membaca konten file huruf.txt di SPIFFS.

```
String datahuruf = readFile(SPIFFS, "/huruf.txt");
  Serial.print("*** Data huruf: ");
```

Kemudian, cetak variabel itu di Serial Monitor.

```
Serial.println(datahuruf);
```

Referensi :

https://randomnerdtutorials.com/

https://www.instructables.com/IOT-Made-Simple-Playing-With-the-ESP32-on-Arduino-/