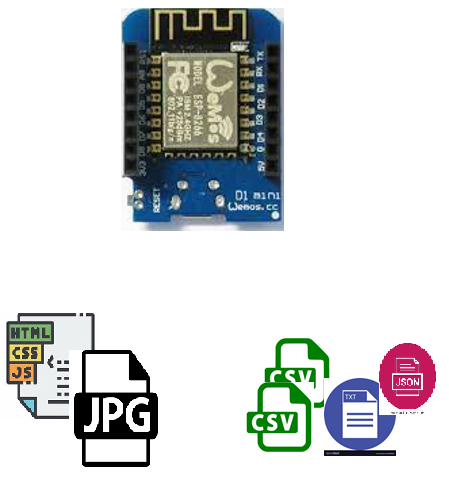
File management

Dalam pembuatan software ini, ESP-8266 terbagi dalam beberapa kelompok penyimpana file :

* File program 2 MB
* Local storage LittleFS 2 MB
  + Konfiguration – Sensor, User
  + Logsheet – data trending, hourly average selama 1 minggu (7 hari), daily average selama 1 tahun
  + Web file – HTML, CSS, JS dan Images



File: /authentication.js

File: /config.html

File: /config.js

File: /gmf.png

File: /index.html

File: /index.js

File: /login.html

File: /login.js

File: /logoGMF.png

File: /users.cfg

File: /logsheet/Friday\_ls.csv

File: /logsheet/Monday\_ls.csv

File: /logsheet/Saturday\_ls.csv

File: /logsheet/Sunday\_ls.csv

File: /logsheet/Thursday\_ls.csv

File: /logsheet/Tuesday\_ls.csv

File: /logsheet/Wednesday\_ls.csv

File: /logsheet/sensors.cfg

**root**

File: /modal-dialog.js

File: /report.html

File: /report.js

File: /style.css

File: /widgetsClass.js

**Log-sheet**

Software Structure

1. Logsheet (data gathering, collection and persistenace from sensor)
   * Get sensor data complete with filtering (alfa ema)
   * Display to OLED
   * Averaging – minute, hourly and daily
   * Saving to local storage (LittleFS) hourly base on day of week, daily base on year
   * Data Services for web
     + Sensor configuration
     + Logsheet hourly
     + Trending data (temperature and humidity)
2. Web services :
   * Server side (back end)
     + Static files services (Images, CSS and JS)
     + HTML
     + AJAX
   * Client side (Front end) pages / User Interface
     + Home
     + Report
     + Configuration
     + Login
     + Logout
     + About
3. Data Model
   * Sensor parameters (High, Low, Unit, Alarm, alfa ema)
   * Loghsheet (temperature and humidity)
   * User (username, password, email, level)
4. Time keeping for synchronizing sampling
5. Start up
   * Display logo during start up
   * Display welcome messages

Adapun fungsi tersebut di atas, dibagi menjadi beberapa bagian software dengan memisahkan sesuai dengan fungsinya (OOP principles) sebagai berikut :

* dataLogger.ino, datalogger.h

Sebagai program utama berikut sebagai pintu masuk (entry point) yang di dalamnya terdapat tulang punggung (skeleton) program Arduino – ESP :

dataLogger.h berisi seluruh definisi global variables dan konstanta dalam berikut fungsi yang digunakan dalam keseluruhan program ESP-8266 ini.

* Importing libraries
* Global variable and objects declaration
* Setup

Dalam function setup ini, ada beberapa hal yang dikerjakan sbb:

* Init OLED 48x64 pixel dengan I2C
* Display logo GMF
* Display Welcome dan program step
* Init pin mode
* Init serial communication dengan baud rate 11520 bit/sec
* Init local storage (LittleFS) untuk server files, configuration dan logsheet
* List files dalam LittleFS
* Load User data (engineer dan operator)
* Setup default user, active user = guest
* Init sensor DHT11 untuk temperature dan humidity dengan one wire communication
* Setup logsheet meliputi :
  + Attach parameter temperature dan humidity
  + Attach sensor DHT11
  + Attach display OLED
  + Set time
* Setup urlController meliputi :
  + Static url (meliputi image files, CSS, JS)
  + HTTP url (meliputi HTML files)
  + AJAX url (data files)
* Start WIFI (MODE\_AP atau Multifiwi)
* Start server
* Loop

Function loop ini yang dijalankan (execution) secara terus-menerus selama ESP8266 ini jalan. Adapun tugas/function yang dijalankan adalah :

* Execute logsheet serta memberikan nilai sampling time
* Execute mainSequence
* Memberikan seting time setiap 1 menit ke logsheet

Adapun detail dari masing-masing fungsi di atas dijelaskan dalam sub bab terpisah.

* logsheet.h, logsheet.cpp

Sebagaimana dijelaskan sepintas dalam Software structure point i. di atas, logSheet dengan entry point function execute, mempunyai tugas yang dijalankan sesuai dengan periode yang berikan pada nilai samplingTime. Nilai ini di-setting pada dataLogger.ino (function loop). Logsheet.h mendefinisikan class Logsheet yang selanjutnya dalam program utama (dataLogger.ino) digunakan untuk membuat object dengan nama logsheet(“logsheet”). Selanjutnya object ini (logsheet) digunakan untuk operasi selanjutnya. Dalam class Logsheet ada beberapa fungsi/method utama yang mendukungnya.

* Function execute()

Fungsi ini sebagai titik masuk (entry point) pada class Logsheet (yang selanjutnya digunakan untuk mendefinisakan object logsheet) dengan beberapa tugas sebagai berikut :

* + Melakukan sinkronisasi waktu antara dataLogger.ino (program utama).
  + Melakukan perhitungan jumlah sampling data per menit, dan membatasi 60 sampling per menit.
  + Melakukan beberapa kegiatan saat sampling time telah sesuai (tiba):
    - Mengambil data dari sensor (\_getsensorValue())

Fungsi ini akan mengambil data dari DHT11 berupa data temperature dan humidity serta melakukan filtering pada data yang diambil berdasarkan parameter alfa ema (default 80%). Pengambilan data DHT11 ini dilakukan ketika tidak pada mode SIMULATION. Saat mode SIMULATION data diambil dari random data.

Data temperature dan humidity selanjutnya disimpan dalam data parameter glogal yang di-attach ke logsheet object. Parameter ini baik untuk temperature dan humidity.

Selanjuntya, data filtering ini juga dibandingkan/dikomparasi dengan data konfigurasi untuk masing-masing konfigurasi sensor, baik untuk alarm high, low serta memberikan status untuk alarm yang dimaksud.

* + - Menampilkan data pengukuran pada OLED 64x48 pixel, lengkap dengan unit untuk Temperature dan Humidity.
    - Melakukan perhitungan event/kejadian mulai dari samplingTrending, samplingSecond, samplingMinute, samplingHour
    - Melakukan kegiatan logsheet sesuai dengan event yang terjadi seperti : perhitungan averaging, penyimpanan data ke dalam local storage (LittleFS) sesuai dengan file ynag dimaksud. Penyimpanan file dengan type text dengan format CSV (Coma Seprated Value).
* Function pendukung WEB
  + Mengambil data rata-rata jam (hourlyAverage) selanjutnya dikirim ke WEB untuk ditampilkan sebagai data harian dalam setiap minggunya (Monday – Sunday).
  + Mengambil data trending per sampling, data ini selanjutnya dikirim ke WEB untuk graphic trending temperature dan humidity (24 sampling data).
  + Mengambil data konfigurasi parameter sensor (range high/low, alarm high/low, unit) selanjutnya dikirim ke WEB untuk mebuat graphic dial baik temperature dan humidity.
  + Mengambil data pengukuran per-sampling untuk menampilkan nilai dalam graphic pada WEB.

Data yang diambil tersebut di atas disimpan dalam local storage (LittleFS) kecuali data pengukuran per-sampling.

* Function private untuk keperluan internal loghseet, fungsi ini hanya bisa digunakan dalam lingkup class logsheet sendiri untuk mendukung kegiatan yang diperlukan meliputi :
  + Setup default parameter – untuk setup default saat pengembangan software
  + Setup parameter sensor dari file berupa data parameter (range high/low, alarm high/low, unit).
  + Init random JSON – untuk setup JSON saat pengembangan software
  + Fungsi penyimpanan dan pembacaan file ke dan dari local storage (LittleFS)
* model.h, model.cpp

Secara umum dijelaskan pada Software structure point iii di atas meliputi :

* + Sensor parameters (High, Low, Unit, Alarm, alfa ema)
  + Loghsheet (temperature and humidity)
  + User (username, password, email, level)

Data model ini selanjutnya digunakan untuk penyimpanan data secara global untuk operasional program. Data model yang dimaksud dibentuk dalam class ataupun data struktur biasa. Data model yang dibentuk dalam class juga dilengkapi dengan API (Aplication Program Interfacing) untuk mempermudah penggunaanya.

Class AccesParameter adalah class yang didedikasikan untuk akses parameter sensor dengan API sbb:

* Init
* getJSON – mengambil parameter dalam format JSON
* getOperation – mengambil data operasi untuk temperature dan humidity
* getParameter – mengambil data dalam bentuk struktur
* setParameter – seting parameter dalam bentuk struktur
* sequenceTimer.h, sequenceTimer.cpp

Sebagaimana dala Software structure point iv, class ini mengatur waktu utamanya setiap menit. Jika durasi satu menit telah tercapai, akan memberikan peringatan berupa event/kejadian sehingga dapat digunakan untuk proses eksekusi kegiatan yang ada pada object logsheet (dari class Logsheet) seperti dijelaskan dalam paparan logsheet di atas.

* start\_up.h

Secara garis besar dijelaskan dalam sub bab Software structure point v, StartUp merupakan Class (yang selanjutnya menadi object startUp) yang bertugas untuk :

* Menampilkan logo Garuda, seperti pada gambar berikut.

Gambar ini merupakan konversi dari file PNG menjadi bitmap dengan ukuran pixel 64x48 sesuai dengan ukuran OLED yang dipakai. Selanjutnya hasil konversi dijadikan konstanta dengan nama logo\_gmf. Untuk menampilkan logo ini menggunakan fungsi logoDisplay() yang ada pada class tersebut.

* Menampilkan welcome, seperti pada urutan gambar berikut.

Pesan welcome ditampilkan secara berurutan oleh fungsi welcomeDisplay(). Pesan welcome mulai dari GMF AeroAsia, lalu IoT DataLogger lengkap dengan versinya dan diakhiri dengan pesan Salman Alfarisi serta bulan pembuatan software ini.





* Menampilkan Step start up.

Untuk mengetahui urutan sebagaimana dijelaskan pada fungsi setup di atas (lihat penjelasan datalogger.h dan datalogger.ino pada sub bab setup) diberikan display pada OLED sehingga tiap Langkah/step start up bisa diketahui. Step mulai dari 00 yaitu pada saat init OLED 64x48 pixel berikut I2C. seperti pada gambar di samping dan dilanjutkan pada urutan yang ada pada fungsi setup.

Pada STEP 05, IoT DataLogger berupaya untuk connect dengan WIFI yang telah dikonfigurasi pada datalogger.h. Jika koneksi berhasil, maka step dilanjutkan sampai dengan step 09.



Step 09 adalah dimulainya aktivasi server sehingga memungkinkan IoT DataLogger bisa diakses via internet/WEB.

Selanjutnya IoT DataLogger bertugas untuk melakukan pengambilan data dari sensor DHT11, menampilkan nilai pada OLED 64x48 pixel serta tugas yang lain seperti dijelaskan dalam logsheet.h dan logsheet.cpp di atas.

Selain dari itu, IoT DataLogger juga siap sebai server untuk menangani permintaan via internet/WEB. Penjelasan terkait dengan ini dipaparkan dalam sub ba tersendiri. Fungsi yang menangani penampilan step ini adalah stepDisplay() dengan memberikan parameter step yang sedang berjalan padanya.

Flowchart – datalogger

Start

Import required libraries

Declare Variables & Objects

Protype functions

Display.begin() – init OLED I2C for 64x48 pixel

startUp.AttachDisplay() – attach OLED to startUp

startUp.logoDisplay – display logo to OLED

startUp.welcomeDisplay – display welcome messages

setup pin for LED

Serial.begin(11520) – baud rate 11520 bit/sec

LittleFS.begin() – init little FS (local storage ESP8266)

listAllFilesInDir() – list file in LittleFS

loadUser() – for engineer and operator

setupDefaultUser() – active user is guest

dhtSensor.begin() – init DHT11 sensor

logsheet.AttachParameter() – temperature & humidity

logsheet.AttachSensor() – sensor DHT11

logsheet.AttachDisplay() – OLED

logsheet.AttachLed() – LED

logsheet.setTime() -synchronize

N

Y

StartWIFI\_AP(); - WIFI AP init

startMultiWifi(); - Multi WIFI init

WIFI\_AP

urlController() – WEB handling (req – resp)

server.begin() – start server

1

1

isMinute

logSheet.setTime() – set time from NTP

Y

N

logSheet.execute() – per sampling time

mainSequence.execute()

End

Logsheet.execute()

Start

\_samplingTime = samplingTime

\_synchronized

\_samplingSec = 0

\_synchronized = true

End

Pelayanan WEB / WEB Services

WEB services adalah bentuk pelayanan/tugas IoT DataLogger terkait dengan tanggapan/respon permintaan dari user (via devices lain : Laptop, Deskto, HP atau gadgets yang lainya). Sebagaimana dijelaskan secara umum pada sub bab Software Structure point ii (WEB Services), Pelayanan WEB ini meliputi :

* Authentication & Authorization user/pengguna
* Menampilkan laman WEB / WEB Page yang responsive

Pelayanan WEB ini terbagi menjadi 2 tempat yaitu (Server Side dan Client Side), server side adalah pelayanan yang dikerjakan oleh IoT DataLogger itu sendiri untuk mendukung permintaan (request dari pengguna/user via internet). Sedangkan client side adalah pelayanan IoT DataLogger yang dikerjakan bersamaan dengan browser yang digunakan (chrome, firefox etc…). Secara fungsi pelayanan ini dibagi menjadi Authentication & Authorization (Auth) dan menampilkan laman WEB/WEB page yang responsive.

Authentication & Authorization (Auth) adalah mekanisme untuk memberikan batasan akses kepada pengguna IoT Data Logger, hal ini diperlukan untuk keamanan program itu sendiri. Tidak setiap user yang akses bisa melakukan pemilihan menu, apalagi menu yang bersifat mengubah data atau reporting ke pejabat yang berwenang. Untuk itu IoT Data Logger juga mempunyai mekanisme Auth yang dimaksud yaitu membatasi pengguna dengan 3 tingkatan :

* Pengguna umum/tamu/Guest : GST – Pengguna ini adalah setiap pengguna yang bisa akses ke laman IoT Data Logger, dengan hanya bisa menampilkan laman induk dan login
* Pengguna Operator : OPR – Pengguna ini bisa melakukan reporting dan login, logout
* Pengguna Engineer : ENG – Pengguna ini bisa melakukan semua menu yang tesedia meliputi reporting, configuration sensor, login, logout

Mekanisme Auth ini berlaku untuk semua laman (index, reporting, login, logout, config).

Pelayanan WEB ini dimulai dengan setup API (Application Program Interface) yang dikerjakan pada fungsi setup (lihat penjelasan dataLogger.h dan dataLogger.cpp pada sub bab setup) saat menjalankan fungsi urlController(). API akan mengatur tatacara melakukan request via WEB, bagaimana IoT melakukan respon terhadapnya serta data apa yang harus dikirimkan terkait request yang dimaksud. urlController membagi respon ini menjadi 2 bagian yaitu :

* Respon terkait dengan permintaan file static (Images, JS, CSS)
* Respon terkait dengan permintaan dynamic (HTML dan data)

Permintaan semua files itu baik static ataupun dynamic tersimpan dalam local storage (LittleFS) lihat penjelasan sub bab Management File di atas.

Masing-masing tugas itu dikelompokkan dalam fungsi yang terpisah yaitu :

* loadStaticFiles()
* urlController()

loadStaticFiles() akan dipanggil oleh urlController() saat dijalankan/dieksekusi pada fungsi setup() pada dataLogger.ino (lihat penjelasan dataLogger.ino di atas serta flowchart terkait).

loadStaticFiles() mempunyai API seperti dalam daftar berikut :

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **API** | **HTTP Method** | **Description** |
| 1 | /index.js | HTTP\_GET | File javascript digunakan oleh index.html |
| 2 | /authentication.js | HTTP\_GET | File javascript digunakan oleh semua html files |
| 3 | /widgetsClass.js | HTTP\_GET | File javascript digunakan oleh index.html untuk graphic |
| 4 | /login.js | HTTP\_GET | File javascript digunakan oleh login.html |
| 5 | /modal-dialog.js | HTTP\_GET | File javascript digunakan oleh semua file html untuk menampilkan dialog windows untuk merespon permintaan About |
| 6 | /login.js | HTTP\_GET | File javascript digunakan oleh login.html |
| 7 | /report.js | HTTP\_GET | File javascript digunakan oleh report.html |
| 8 | /config.js | HTTP\_GET | File javascript digunakan oleh config.html |
| 9 | /logoGMF | HTTP\_GET | File image untuk menampilkan logo GMF pada index.html dan report.html |
| 10 | /report.js | HTTP\_GET | File javascript digunakan oleh report.html |
| 11 | /style.css | HTTP\_GET | File CSS digunakan untuk semua file html |

urlController() secara aktif merespon permintaan user yang diberikan lewat browser melalui laman yang tertera dalam alamat IP. Laman akan ditampilkan sebagaimana permintaan user melalui menu navigasi yang ada di atas setiap laman (top navigation). Selain itu saat pertama kali user masuk dalam alamat IP, maka dataLogger menampilkan home page (laman awal) dan memberikan file yang tersimpan dalam local storage (LittleFS) ke browser untuk ditampilkan dalam display devices user.

API yang ada dalam urlController sebagai berikut :

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **API** | **HTTP Method** | **Description** |
| 1 | / | HTTP\_GET | Menampilkan home page (index.html) |
| 2 | /report | HTTP\_GET | Menampilkan report page (report.html) |
| 3 | /login | HTTP\_GET | Menampilkan login page (login.html) |
| 4 | /login | HTTP\_POST | Kirim data username & password untuk user |
| 5 | /loginStatus | HTTP\_GET | AJAX data untuk mengirimkan status login user |
| 6 | /logout | HTTP\_GET | Menampilkan dialog logout selanjutnya home page |
| 7 | /config | HTTP\_GET | Menampilkan config page (config.html) |
| 8 | /config | HTTP\_POST | Update data konfigurasi sensor |
| 9 | /getTrendingData | HTTP\_GET | AJAX data trending pada graphic home page |
| 10 | /getSensorConfig | HTTP\_GET | AJAX data konfigurasi sensor pada graphic home page dan config page |
| 11 | /getActiveUser | HTTP\_GET | AJAX data aktive user digunakan pada semjua home page untuk authentication dan authorization |
| 12 | /haourlyAvgDay | HTTP\_GET | AJAX data untuk report page |
| 13 | /getSensor | HTTP\_GET | AJAX data untuk menampilkan data pengukuran sensor pada home page |

urlController() dan loadStaticFiles() dikerjakan pada sisi server (Server Side).

Perancangan WEB Page / Laman WEB.

Perancangan WEB page memegang perana yang penting karena melalui WEB page ini pengguna berhubungan dengan IoT DataLogger ini, sehinnga beberapa kriteria WEB page harus diperhatikan :

* Kemudahan akses
* Ramah pengguna (user friendly)
* Sederhana
* Keindahan
* Kecepatan respon

Untuk keperluan di atas, setiap laman yang dibuat melibatkan 3 pilar utama bahasa program WEB yaitu (HTML, CSS, JS). HTML (Hyper Text Markup Language) digunakan untuk keperluan layout laman, mengatur setiap element yang akan ditampilkan. Agar laman tampak indah stylist diperlukan file CSS (Cascading Style Sheet) sehinga tampilan text bisa mempunyai model font serta ukuran yang menarik, berikut warna background dan warna font.

File CSS akan menjadi bagian dari HTML sekalipun ditulis dalam file yang berbeda/terpisah (dan dimungkinkan bisa dalam satu file). Pemisahan file ini untuk kemudahan melakukan perancangan, pengembangan serta pelacakan kesalahan (debuging). Pemisahan ini juga terkait dengan OOP principles dimana file dipisahkan berdasarkan BoC (Business of Concern) – berdasarkan tugasnya masing-masing.

File JS (javascript) menentukan perilaku (behavior) dari laman yang ditampilkan. Dengan JS memungkinkan laman menjadi lebih dinamis dalam merespon permintaan user IoT DataLogger. Hal ini dimungkinkan karena JS dapat melakukan hal-hal berikut ini :

* Memberikan respon tanpa terhubung dengan server lebih dulu
* Melakukan validasi dan memberikan feedback kepada user sebelum dikirim ke server
* Meminta data sebagian dari laman keseluruhan (AJAX)
* Memperkuat fungsi CSS sehingga bisa melakukan perubahan style
* Bisa melakukan animasi sehingga laman bisa sangat impresive

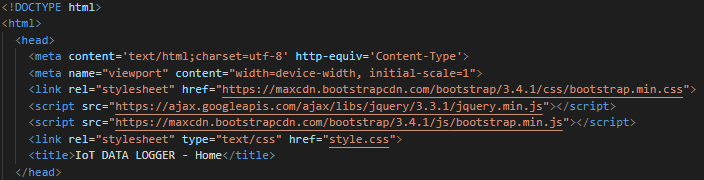
Sebagaimana file CSS, file JS merupakan bagian dari HTML dan bisa ditulis menjadi satu atau terpisah dalam file yang berbeda. Bahkan file CSS dan JS bisa merujuk pada link pada laman WEB.

Home Page – laman utama/induk

Home page akan tampil pertama kali ketika user menulis pada WEB browser alamat yang tertera dalam IP. Misalkan dalam simulasi yang dilakukan kita mengetikkan <http://192.168.137.235/> atau sekedar <http://192.168.137.235> atau bahkan 192.168.137.235 saja. Web browser akan mengirimkan pesan ini melalui jaringan internet via WIFI pada alamat terdekat (utamanya yang terhubung dalam satu jaringan LAN – Local Area Network). Jika ditemukan, maka server dalam hal ini ESP-8266 yang berfungsi sebagai server akan memberikan respon terkait permintaan itu. Respon yang diberikan sesuai dengan daftar API tersebut di atas (yang tertera dalam urlController lihat penjelasan WEB Services di atas).

Dalam API, permintaan itu artinya server (ESP-8266 – IoT Server) merespon dengan memberikan file index.html yang berada dalam local storage (LittleFS pada root folder) ke browser untuk diproses lebih lanjut oleh browser peminta.

Pada index.html ini terdiri dari beberapa element :

Type dokumen, dalam hal ini adalah html. Ditandai dengan tag <!DOCTYPE html>

Seluruh dokument html berada dalam cakupan tag pembuka <html> dan tag penutup </html>. Selanjutnya tag <html> terbagi menjadi dua bagian yang ditandai dengan tag <head> …… </head> dan <body> ……. </body>.

Secara umum file html mempunyai struktur sbb :

<!DOCTYPE html>

<html>

    <head>

        <title>Page Title</title>

    </head>

    <body>

        <h1>This is a Heading</h1>

        <p>This is a paragraph.</p>

    </body>

</html>

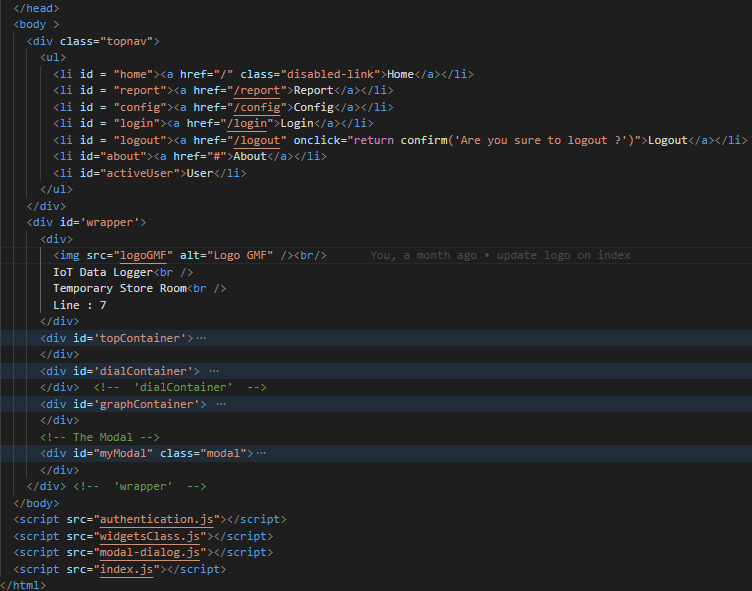
Dokument yang tampak dalam laman adalah yang berada pada tag <body> …… </body>. Pada gambar di atas menampilkan tag <head>, dalam bagian itu menampilkan beberapa hal meliputi meta data link CSS, JS yang digunakan berikut title dari laman yang ditampilkan ( IoT DATA LOGGER – HOME). Sebagaimana penjelasan file CSS, JS di atas, file-file itu bisa terpisah dalam file yang berbeda atau link ke alamat web.

Untuk mempercantik penampilan CSS merujuk pada link web dengan tag <link……> (<link rel = "stylesheet" href = "https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">) serta local storage (<link rel="stylesheet" type="text/css" href="style.css">).

File JS untuk memberikan respons yang cepat (responsive), merujuk dari WEB dengan tag yang ditandai dengan <srcipt….>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

Selain dari WEB, file JS juga merujuk dari local storage yang dipanggil pada akhir sebelum tag penutup </html>. Peletakan pada awal atau akhir itu tergantung dari apakah file itu dikerjakan lebih dulu atau kemudian.Body html, merupakan bagian htlm yang ditampilkan dalam laman web.

Seluruh file yang terlibat html, CSS dan JS sebagian besar fokus pada penampilan element yang ada pada tag body ini.

Pada bagian akhir html terdapat rujukan file JS yang berada pada local storage (LittleFS) yaitu :

<script src="authentication.js"></script>

<script src="widgetsClass.js"></script>

<script src="modal-dialog.js"></script>

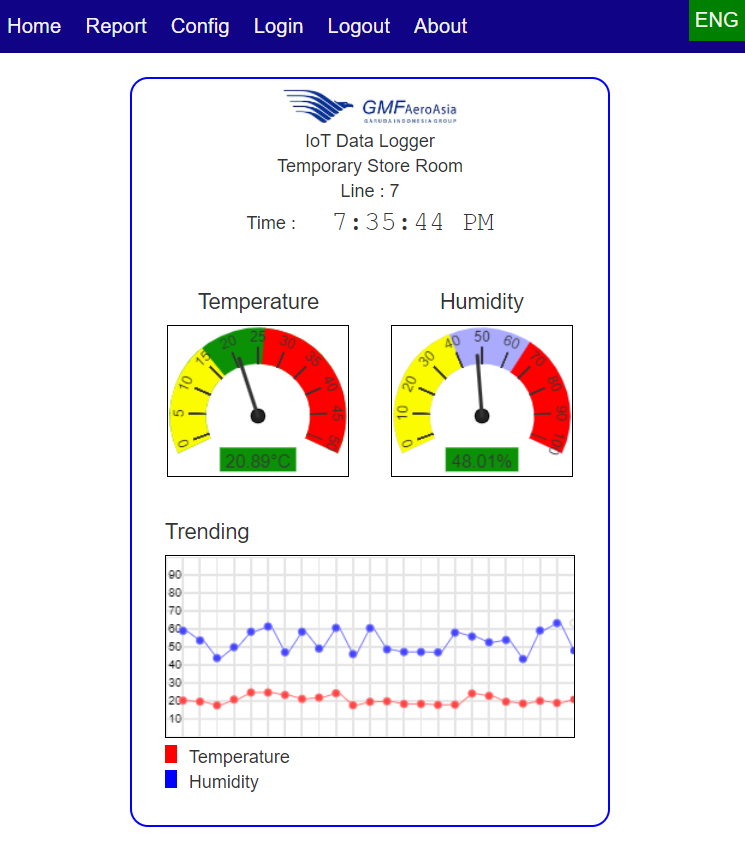
<script src="index.js"></script>

Pada index.html, terbagi menjadi 2 bagian penting yaitu bagian atas yang memuat seluruh menu pada laman index.html dan laman web yang lain pada IoT Data Logger ini. Menu ini menggunakan class topnav pada botstrap. Masing-masing menu merujuk pada API pada urlController sebagaimana dijelaskan pada sub bab sebelumnya, dan mengimplementasikan mekanisme Auth seperti dijalaskan pada sub bab Authentication & Authorization di atas.

Berikut tabulasi dari Auth yang dimaksud :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Menu - Topnavigation** | **GST - Guest** | **OPR - Operator** | **ENG - Engineer** |
| 1 | Home | V | V | V |
| 2 | Report |  | V | V |
| 3 | Config |  |  | V |
| 4 | Login | V | V | V |
| 5 | Logout |  | V | V |
| 6 | About | V | V | V |

Mekanisme Auth dikerjakan oleh javascript yang tertuang dalam file authentication.js, file ini tersimpan dalam local storage (LittleFS) dan dikirim ke browser setelah ada permintaan dari laman utama/induk melalui file index.html. File authentication.js akan dipanggil oleh fungsi yang terdapat dalam file index.js saat pertama kali laman tampil. Mekanisme Auth ini akan dipakai oleh keseluruhan laman yang ada pada IoT Data logger ini.



**1a**

**1**

**1b**

**2a**

**2b**

**2c**

**2**

Jika semua file yang terlibat di atas (mulai dari html, css dan js) telah terkirim ke browser, selanjutnya tugas browser untuk menampilkan dalam layar/display user, sehingga tampak seperti gambar di atas (pada browser chrome).

Laman induk/utama terdiri dari beberapa bagian :

* Topnav (1 navigasi atas) terdapat 2 bagian (menu 1a dan active user 1b)
* Element yang menjadi satu dengan id wraper (2) terdiri dari 3 element :
  + Top container (2a) – menampilkan logo, IoT Data Logger sampai dengan waktu
  + Dial container (2b) – menampilkan indikasi (gauge) untuk temperature dan humidity
  + Graph container (2c) – menampilkan trending temperature dan humidity
* Window modal dialog saat menu About active, seperti gambar berikut ini.

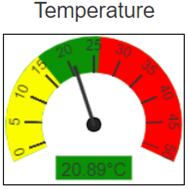
Topnav – Topnavigation berikut berfungsi sebagai menu (kecuali indikasi User yang aktif).

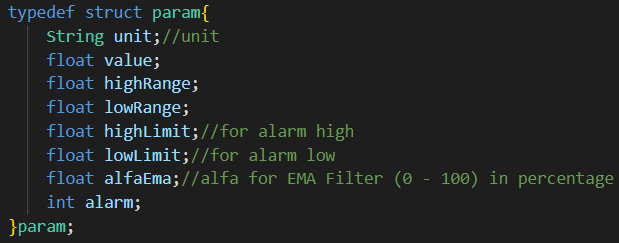
Menu yang aktif pada topnav ini mengikuti mekanisme Auth (seperti dijelaskan di atas) dan dikerjakan oleh authentication.js dikerjakan saat laman pertama kali muncul. Saat menu dipilih, maka akan memberikan perintah/request ke server untuk memberikan file html yang bersesuaian, sesuai dengan API yang telah didefinisikan dalam fungsi urlController.

Berikut API yang dimaksud :

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Menu - Topnavigation** | **API - urlController** | **Description** |
| 1 | Home | / | Laman utama |
| 2 | Report | /report | Laman report |
| 3 | Config | /config | Laman config sensor |
| 4 | Login | /login | Laman login |
| 5 | Logout | Dialog window | Dialog confirmation |
| 6 | About | Modal dialog window | Modal dialog |

Topcontainer terdapat logo, title sampai dengan waktu. Waktu diupdate setiap 1000 mili second oleh funsi interfal yang terdapat pada file index.js. File index.js ini sebagai induk seluruh file JS (javascript) yang digunakan pada laman ini.

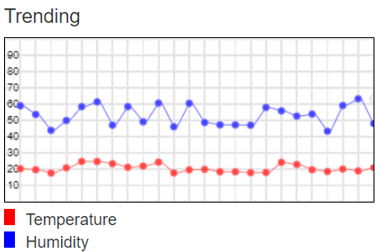
Dialcontainer terdapat 2 dial untuk pengukuran temparature dan humidity, untuk menampilkan gambar (graphic), html menggunakan tag canvas yang selanjutnya dikerjakan oleh fungsi yang terdapat dalam file widgetsClass.js. Untuk melakukan gambar di samping diperlukan data konfigurasi sensor yang terdapat dalam IoT Data Logger (server) disusun dalam data terstruktur yang telah menjadi kelas AccessParam (Access Parameter sensor). Definisi itu ada pada file model.h dan model.cpp seperti dijelaskan di sub bab di atas. Pemodelan seperti memudahkan pengembangan program lebih lanjut sebagaimana OOP principles. Adapun parameter yang digunakan untuk kedua sensor tampak dalam gambar berikut ini :

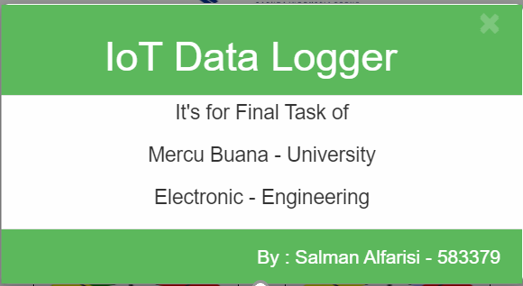
Data yang dikirim ke browser dalam bentuk JSON format dan akan diparsing oleh browser sebelum digunakan untuk menggambar Dial. Data alarm digunakan untuk memberikan warna background pada indikasi. Jika terdapat alarm baik low/high diberikan warna merah, sedangkan jika normal diberikan warna hijau seperti gambar di atas.

Data configurasi sensor ini, diterima oleh browser dari server saat laman utama pertama kali tampil melalui API yang telah didefinisikan dalam urlController. Pengiriman data seperti di atas dapat dilakukan dengan technologi AJAX yang memungkinkan pengiriman data sebagian dari seluruh laman yang ada. Dan ini membuat laman lebih responsive.

Selanjutnya update nilai pengukuran berikut graphic trending dikerjakan oleh fungsi interval yang terdapat dalam index.js. Periode update ditentukan oleh nilai sampling time yang diberikan dari 2000 ms sampai dengan 60000 ms (1 menit). Sehingga untuk membuat laman utama ini (index.html) diperlukan API yang lain sbb:

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **API** | **HTTP Method** | **Description** |
| 1 | / | HTTP\_GET | Menampilkan home page (index.html) |
| 2 | /getTrendingData | HTTP\_GET | AJAX data trending pada graphic home page |
| 3 | /getSensorConfig | HTTP\_GET | AJAX data konfigurasi sensor pada graphic home page dan config page |
| 4 | /getActiveUser | HTTP\_GET | AJAX data aktive user digunakan pada semjua home page untuk authentication dan authorization |
| 5 | /getSensor | HTTP\_GET | AJAX data untuk menampilkan data pengukuran sensor pada home page |
| 6 | /style.css | HTTP\_GET | CSS file for styling |

Graphic trending, graphic ini juga dikerjakan pada fungsi yang terdapat dalam widgetsClass.js dan nilai diupdate setiap peride nilai sampling. Jumlah data trending adalah 24 sampling, artinya jika periode tiap 10000 ms (10 detik) maka akan menampung data 240 detik (4 menit).

Dalam setiap laman IoT Data Logger, diberikan menu About yang jika dipilih/di-click/diaktifkan akan menampilkan gambar dialog modal window di samping. Untuk menampilkan window yangdimaksud tanpa melibatkan server, hal inilah yang membuat dengan peranan JS laman menjadi lebih responsive.

Keseluruhan mekanisme yang diuraikan di atas adlah kolaborasi antara server (server side/back end) dan client (client side/front end).

Report Page – laman report (reporting)

Laman report dapat muncul setelah memilih pada menu topnav, namun menu ini hanya bisa diakses oleh user yang login dengan level OPR ataupun ENG (lihat pembahasan Auth pada sub bab di atas). Laman ini memberikan fungsi pelaporan (reporting) berupa rerata data per jam yang dikemas dalam hari tiap pekan (Sunday – Saturday). Dengan penyimpanan data selama satu pekan (7 hari) memberikan kehandalan bahwa tidak ada data logsheet yang tidak terecord. Bentuk report dalam file PDF dan bisa di-share via WA atau email dalam bentuk attachment. Reporting dalam bentuk pdf ini memberikan keunggulan di antaranya :

* Tidak bisa diubah/diedit
* Lebih flexible dalam pengiriman data
* Telah terformat dalam bentuk tabel.

Laman reporting didapat dari file report.html yang mempunyai format umum seperti dijelaskan pada file index.html (Dokumen type, Head, Body dll – lihat penjelasan pada sub bab laman utama). Pada element head serta bagian atas dari body untuk keseluruhan laman IoT Data Logger ini adalah sama, sehingga tidak perlu dijelaskan ulang.

**1**

**1a**

**1b**

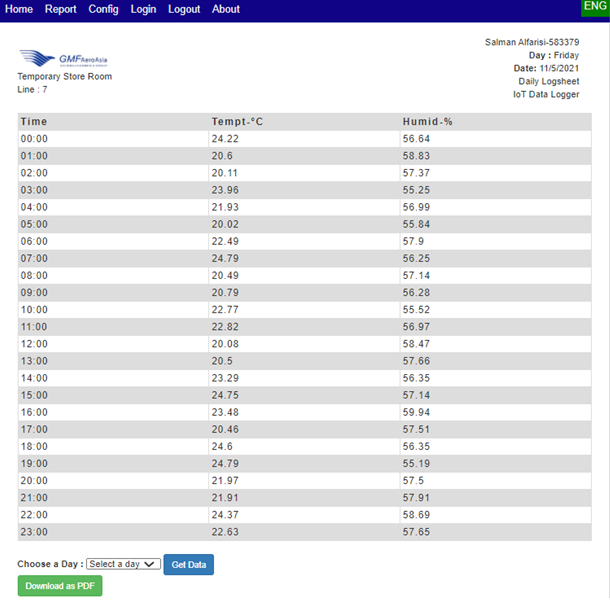
**2a**

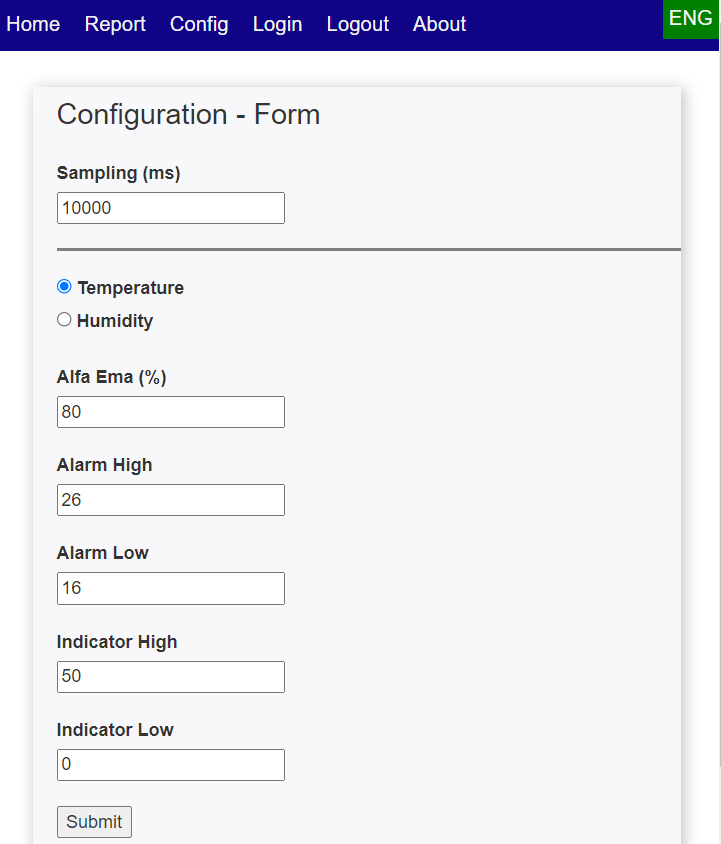
**2**

**2b**

**2c**

**2d**





**1**

**1a**

**1b**

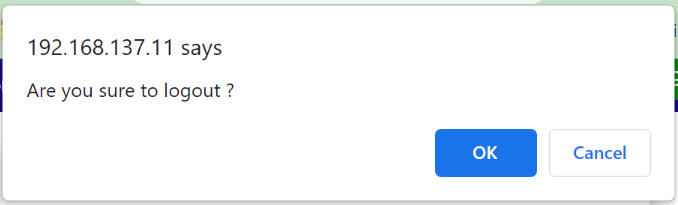
**2a**

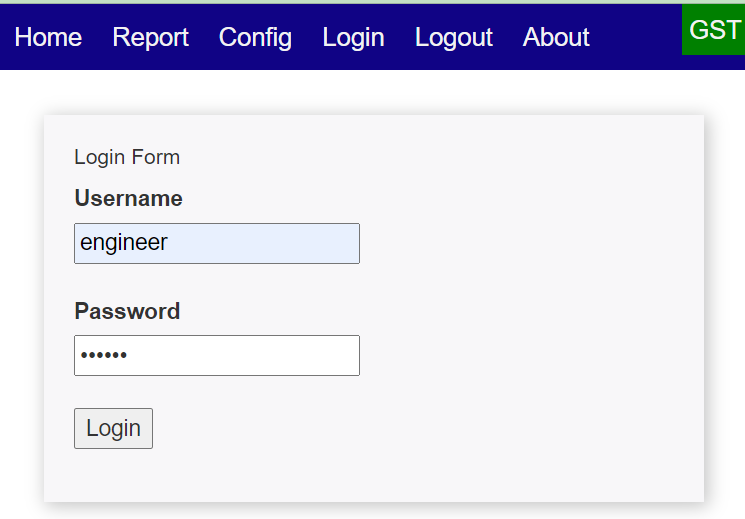
**2b**

**2**

**2c**

**2d**





**1**

**1a**

**1b**

**2**

**2a**