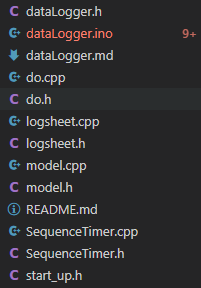
# 3.2 Perancangan dan pembuatan Software

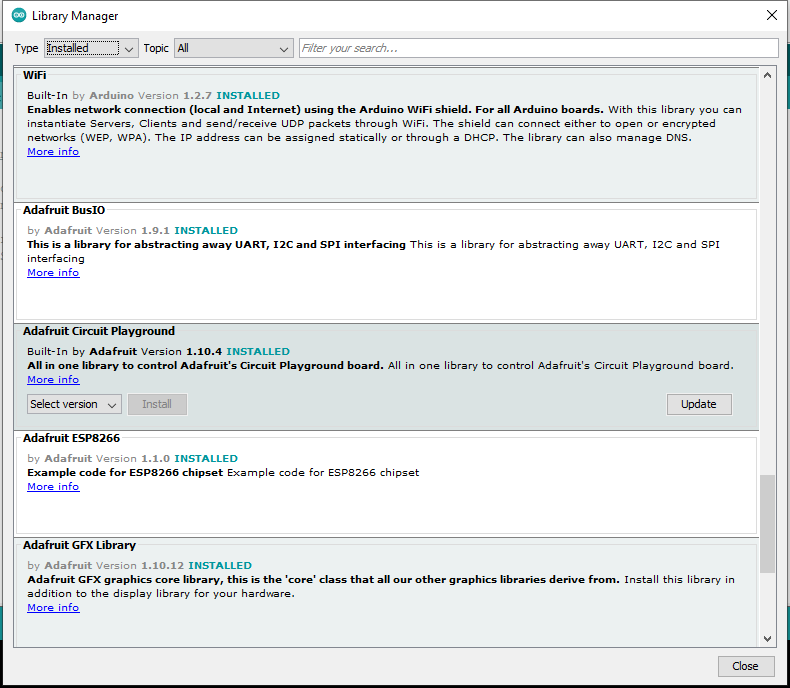
Detail Program

Sudah dijelaskan pada penjelasan umum bahwa dalam perancangan program menggunakan pendekatan oop. Bisa dilihat pada gambar dibawah ini yang memperlihatkan beberapa file dari program yang telah dibuat. Diperlihatkan bahwa program ini menggunakan strategi OOP yaitu membagi program menjadi bagian sesuai dengan tugasnya/kepentinganya. Pada program dibawah ini terdapat 6 bagian (1 main program dan 5 class), datalogger.ino sebagai main program dan yang lainnya sebagai class. Ke 5 class ini sudah tercantum pada file-file dibawah ini yaitu do, logsheet, model, SequenceTimer dan start\_up. File-file ini merupakan library, jadi bisa disimpulkan bahwa class ini sudah terwakili oleh library.

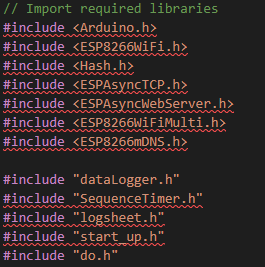


Dalam perancangan dan pembuatan software ini ada urutan tersendiri dimana akhirnya program ini dapat menjalankan sistem data logger sesuai dengan rancangan yang telah dibuat. Berikut ini urutannya:

1. Pada main program (dataLogger.ino) langkah awal yang harus dilakukan adalah memasukkan library pada program. Library-library ini didapatkan dari aplikasi Arduino IDE dan dibuat sendiri (own library). Pada Arduino IDE terdapat menu “Library Manager”, disinilah tempat mengunduh dan menginstall library yang diperlukan. Untuk yang own library terdapat didalam satu folder dengan program dataLogger.ino.



Bisa dilihat pada gambar di bawah ini terdapat perintah #include yang digunakan untuk memanggil library masuk ke dalam program. Bisa dilihat pada gambar program dibawah ini terdapat perintah #include yang dilanjutkan dengan tanda kurung sudut (< >) dan tanda petik dua (""). Untuk tanda kurung sudut (< >) ini merupakan library yang telah di-loading atau diinstall dari library Arduino IDE dan untuk tanda petik dua ("") merupakan library yang berada dalam satu folder dengan dataLogger.ino.



GAMBAR MASIH CONTOH

Berikut ini penjelasan dari library yang terdapat di program diatas:

* #include <Arduino.h>

Library ini memungkinkan pengguna untuk menggunakan fitur jaringan seperti rest dan mqtt. Termasuk beberapa tools untuk ESP8266.Bisa digunakan hanya dengan Arduino Uno WiFi Developer Edition.

* #include <ESP8266WiFi.h>

Library ini berguna untuk memberikan ESP8266 rute Wi-Fi secara specific yang kita panggil pada program untuk terhubung ke jaringan.

* #include <ESPAsyncTCP.h>

Ini adalah library TCP yang sepenuhnya asinkron, yang ditujukan untuk memungkinkan lingkungan jaringan multi-koneksi yang bebas masalah untuk MCU ESP8266 Espressif. Library ini adalah basis untuk ESPAsyncWebServer.

* #include <ESPAsyncWebServer.h>

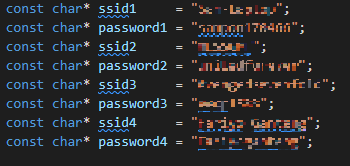
Merupakan Async HTTP dan WebSocket Server untuk ESP8266 Arduino. Untuk menggunakan library ini ESP8266 memerlukan ESPAsyncTCP.

* #include <ESP8266WiFiMulti.h>

Library ini dapat digunakan untuk menghubungkan ke jaringan WiFi dengan sinyal WiFi terkuat (RSSI). Ini memerlukan pendaftaran satu atau lebih titik akses dengan SSID dan kata sandi. Dan bisa secara otomatis beralih ke jaringan WiFi lain ketika koneksi WiFi terputus.

* #include "dataLogger.h"
* #include "SequenceTimer.h"
* #include "logsheet.h"
* #include "start\_up.h"
* #include "do.h"

1. Selanjutnya pada program datalogger.h digunakan untuk menulis nama ssid dan password dari wifi tujuan yang akan dikoneksikan ke alat data logger.



Bisa dilihat pada program ini bisa menulis 4 opsi wifi tujuan. Setiap opsi ini bisa digunakan sebagai cadangan dari opsi yang lain. Jadi Ketika wifi tujuan yang pertama terputus bisa mengkoneksikan langsung ke wifi tujuan yang lainnya selama masih dalam range dari wifi tujuan.

1. SequenceTimer.h
2. logsheet.h
3. start\_up.h
4. do.h