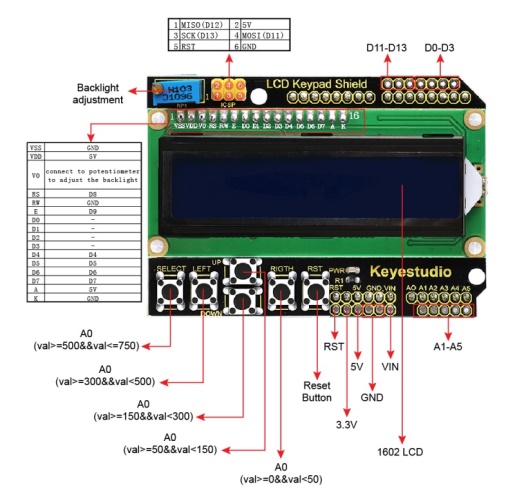
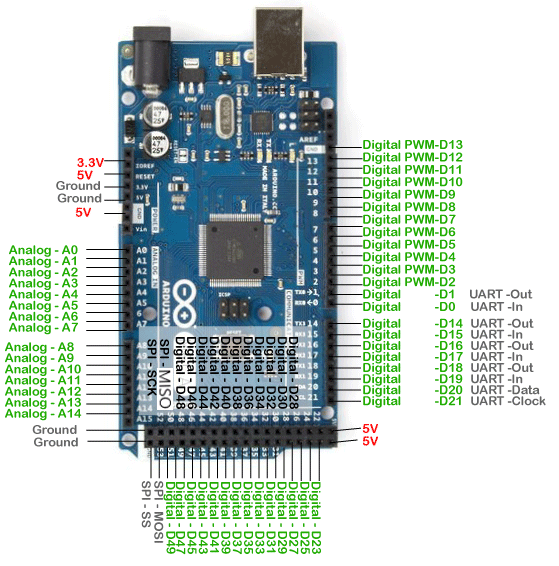
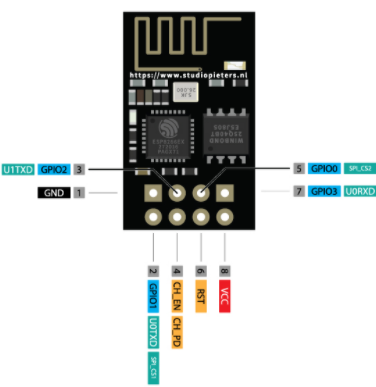
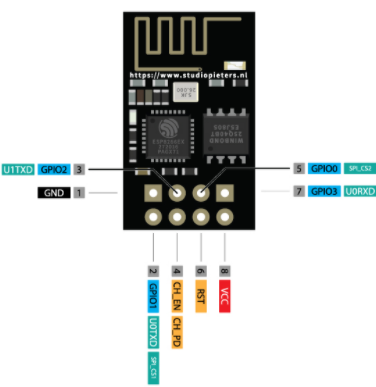
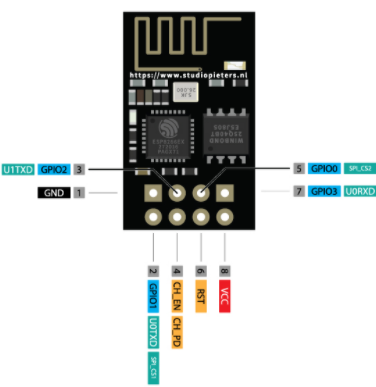
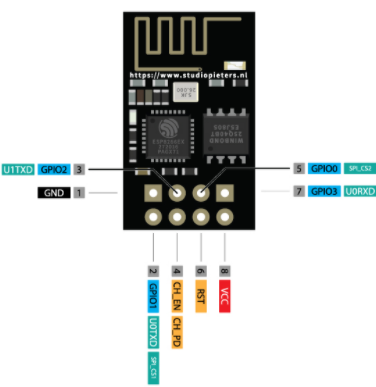
**Fire Protection System Project**

* **Spesifikasi** 
  1. System ini dirancang secara wireless serta mempunyai kemampuan diakses melalui internet sehingga bisa dibuka dari manapun(sepanjang fasilitas internet ada).
  2. System juga dilengkapi display local berupa lcd serta beberapa keypad (button) sehingga bisa diakses secara local.
  3. Sensor api (fire) dipasang pada ruangan (selanjutnya disebut zona) dan memberikan data pengukuran ke server (selanjutnya disebut broker) secara wireless.
  4. Broker ini selanjutnya bertindak sebagai pengendali (controller) serta sebagai pintu masuk dan keluar (gateway) ke jaringan internet
  5. Pada Zona itu juga dilengkapi actuator (berupa solenoid valve) untuk membuka pemadam api (air, powder, foam dll) dalam system ini menggunakan air bertekan (sehingga mampu melakukan penyiraman/spraying).
  6. Actuator menjadi salah satu dari sensor api dan dapat diperintah oleh controller.
  7. Protokol komunikasi antar broker (controller) dan sensor/actuator menggunakan MQTT (Message Queuing Telemetry Transport), sedangkan kejaringan internet menggunakan HTTP (Hypertext Transfer Protocol) sehingga broker (sekaligus sebagai server).
  8. Pengguna (user) bisa melihat dari lcd local beberapa menu, setting parameter serta status masing-masing zona.
  9. Pengguna juga bisa melakukan perubahan setting parameter melalui keypad yang disediakan di local.
  10. Pengguna juga dapat melakukan monitoring status dan perubahan secara remote melalui jaringan internet.
  11. Sytem ini bisa bekerja secara otomatis dan manual.
  12. System ini menggunakan program dengan pendekatan OOP (Object Oriented Programming) serta menggunakan pattern/pola MVC (Model-View-Controller).
  13. Pendekatan ini ditempuh untuk mempermudah dalam pembuatan, mudah dikembangkan serta mudah melakukan pelacakan jika ada masalah.
  14. Ditambahkan lagi…..
* **Hardware**





* **Software**

Pembagian program menjadi bagian-bagian kecil sesuai dengan fungsinya membuat program lebih sederhana dan dapat dikerjakan secara parallel oleh anggota tim. Pembagian ini menggunkan pattern/pola MVC (Model View Controller).

**Model** adalah bagian program yang memuat model data yang terlibat dalam program. Dalam WlFPS ini meliputi data-data yang ditampilkan dalam menu, data parameter setting masing-masing zona. Data tersusun berdasar struktur-data sehingga memudahkan dalam penangananya.

**Struktur data menu** :

**typedef struct dataMenu{**

**int id;//**

**boolean hasParameter;//is a Parameter**

**String pesan1;//pesan untuk baris pertama (1)**

**String pesan2;//pesan untuk baris kedua (2)**

**}dataMenu;**

**Struktur data parameter – Untuk masing-masing Zona :**

**typedef struct parameter{**

**String id;//**

**String unit;//unit**

**int indexMenu;//index menu**

**float value;**

**float highRange;**

**float lowRange;**

**float highLimit;//for alarm high**

**float lowLimit;//for alarm low**

**float increment;**

**boolean highAlarm = false;**

**boolean lowAlarm = false;**

**}parameter;**

Pada model ini juga dilengkapi dengan mothode/fungsi untuk melakukan akses pada masing-masing struktur data serta dijadikan dalam satu class agar menjadikan satu object. Seluruh model data dikumpulkan dalam file *Model.h* (header file) serta *Model.cpp* (uraian masing-masing method/fungsi).

**View** adalah bagian program yang bertugas menampilkan pesan ke LCD, Serial port, serta home-page (termasuk LED – jika diperlukan). Pesan ini meliputi tampilan menu utama, data parameter serta status dari WlFPS (alarm masing-masing zona). Dalam view ini juga bertugas untuk menampilkan operasional WlFPS dan menjadi antarmuka antara pengguna dan system. Seluruh class (object) view dikumpulkan ke dalam file View.h (header file) serta View.cpp (uraian masing-masing method/fungsi).

**Controller** adalah bagian program yang melakukan pengendalian/pengaturan dari seluruh kegiatan meliputi :

1. Memberikan respon pada setiap perintah yang diberikan oleh pengguna.
2. Memilih data yang ditampilkan dalam LCD, Serial Port, Web-page ataupun LED
3. Menentukan tindakan berdasar input yang masuk

Controller ini dilengkapi dengan method/fungsi beberapa menu :

1. Menu Utama
2. Menu Parameter
3. Menu Ubah Parameter

Controller juga bisa melakukan akses ke beberapa object/class :

**void attachCmdIn(command\*);//Pasang command, penggunaan pointer ini sangat mutlak (JIKA TIDAK ERROR !!!!)**

**void pasangModelMenu(AccessDataMenu\*);//Pasang Model menu, penggunaan pointer ini sangat mutlak (JIKA TIDAK ERROR !!!!)**

**void pasangModelParameter(AccessParameter\*);//Pasang Model parameter, penggunaan pointer ini sangat mutlak (JIKA TIDAK ERROR !!!!)**

**void pasangView(View\*);//Pasang View, penggunaan pointer ini sangat mutlak (JIKA TIDAK ERROR !!!!)**

**void setMenuIndex(int);**

**int getMenuIndex();**

**int getParameterIndex();**

**void menu();**

Perintah diakses melalui void attachCmdIn(command\*) – pasang input-command/perintah yang meliputi perintah dari Keypad, Serial, Bluetooth, Wifi ataupun via Web page serta dari sensor yang terpasang. Dari perintah ini kemudian controller melakukan proses untuk pemilihan data, memproses serta menampilkan dalam LCD, Serial Webpage serta LED.

Model diakses melalui void pasangModelMenu (AccessDataMenu\*), void pasangModelParameter (AccesssDataParameter\*) yang memungkinkan controller untuk melakukan akses (read/write).

View diakses melalui void pasangView(View\*) yang memungkinkan controller untuk memberikan erintah pada View untuk menampilkan data menu, parameter setelah dilakukan proses sesuai logic/algoritma yang dibuat dalam WlFPS.

Seluruh kegiatan controller dilakukan dalam void menu() didalamnya terdapat mekanisme pengambilan perintah/command dari masing-masing inputan (Keypad, Serial, Bluetooth, Web-page) menentukan menu-mana yang aktif berdasar perintah yang masuk. Menu ini meliputi Menu Utama, Menu Parameter, Menu Ubah Parameter. Dalam masing-masing menu itu ditentukan data yang ditampilkan dalam LCD, LED, Web-page serta fungsi untuk masing-masing Keypad.

Seluruh mekanisme controller ini ada pada file Controller.h (header file) dan Controller.cpp (uraian masing-masing method/fungsi).

Selain hal di atas, WlFPS juga mempunyai object/class yang bertugas untuk mengatur beberapa fungsi Watch-dog (berupa LED yang berkedip setiap satu detik pada Built-in LED). Watch-dog ini berfungsi sebagai tanda bahwa system bekerja normal. Fungsi itu ada pada file SequenceTimer.h (header file) dan SequenceTimer.cpp (uraian masing-masing method/fungsi).

Sensor adalah bagian dari input command/perintah, sensor ini memberikan masukan melalui WIFI dengan protocol MQTT ke broker (mirip dengan server).

Actuator berupa solenoid adalah salah satu bentuk keluaran merupakan penggerak untuk mengalirkan media pemadaman (air).