Static Software Architecture

(Module-Level Design)

File Structure Rules :

📂 Every Module (Sensor / Protocol) has a folder:

- **Folder_name is capital letter**

```
EX:LM35
```

- Every module :
  - Contain Condition Enable to all file

```
Ex:
#if GPIO_ENABLED == STD_ON
.
. #include All_files here
. All function inside this macro
.
.
#endif
```

  - Contain Debug

```
Ex:
#if GPIO_DEBUG == STD_ON
#define DEBUG_PRINTLN(var) Serial.println(var)
```

📂 Every Module (Sensor / Protocol) divide into :

- `.c` → Implementation
- `.h` → Interface
- **file name is small letter**

```
ex :  /src
        App/
          LM35/
             lm35_sensor.c
             lm35_sensor.h
```

- All Modules have the same config file

📂 **Every Sensor Module include two function :**

- `Init()` : initialize module
- `Main()` : Reading input pin or main task

```
void SensorName_Init(void);
void SensorName_Main( Data_t * data);
```

- Every main function return : Map value
- Max ,Min map value ->> define in config file to manage it

```
ex:
#define MIN_MAP_TEMP 15
#define MAX_MAP_TEMP 45
```

- **Any global variable start with letter :** g_

```
ex:
  int g_TempValue
```

📂 **Config_file.h**

- **content:**

  macros for :
  - Enable/Disable

    ```
    #define GPIO_ENABLE STD_ON
    ```

  - Thresholds

    ```
    #define TEMP_THRESHOLD
    ```

  - define sensor pin

    ```
    #define LM35_PIN
    ```

  - protocols config

    ```
    #define WIFI_PASSWORD
    ```

- **Any config macro** -> capital letter

```
#define MIN_MAP_TEMP
```

Naming Convention

Element	Rule
Files name	`module_name.c / .h`
Functions	`ModuleName_Action()` (first letter in every word is capital)
Global variable	`g_ModuleName_var` (first letter in every word is capital)

Comments

- Add Comments as you like
