Module	Features	Update/Get DomoticZ data via	Send/Get data TO/FROM MODULE	Send/Get data VIA	Technical POI
Domoticz					
mqtt_cluster.js	High Availibility Domoticz cluster management based on two servers (active/passive nodes) synchronized via MQTT. Under normal condition, this script running in the backup server monitors the main server and synchronizes its own Domoticz database. If the main server fails, this script manages the backup server in order to continue to deliver the service.	Update --> MQTT[B_domoticz/in] Get <-- MQTT[M_domoticz/in, M_domoticz/out] Get (Heartbeat) <-- JSON_API			
iot_ESP8266_GM43.ino	Lighting management. Lighting can be switched on/off from legacy wall pushbuttons or Domoticz. If lighting is switched on/off using the wall pushbuttons, an MQTT feedback message is sent to DomoticZ.	Update -- > MQTT[domoticz/in] Get <-- MQTT[domoticz/out]	NO		
iot_ESP8266_DHT22.ino	Temperature sensor using DHT22. Answer to HTTP JSON requests.	NO	NO		
iot_ESP8266.js	Polls the temperature sensors and log the values within Domoticz database. Compute and log degrees-days. Monitor the temperature sensors and Raise failure flag if sensors don't answer to requests.	Update -- > JSON_API	iot_ESP8266_DHT22.ino	Get <-- HTPP	
iot_ESP8266_ACS712.ino	Read ACS712 sensors. Send via MQTT its heater power usage (computed energy usage and ESP8266/ADC raw values). Start/stop its heater according to commands received from MQTT. Self Learning of its heater Nominal Power.	Update --> MQTT[domoticz/in] Get <-- MQTT[domoticz/out]	iot_Orchestrator.js	Send --> MQTT[heating/in] Get <-- MQTT[heating/out]	
iot_ACS712.js	Consolidate individual heater consumptions. CalculatesThermal loss and Heating/Cooling Ratios	Update -- > JSON_API	NO		
iot_Orchestrator.js	Manage Heaters and Heating Zones (Scheduled TOP Start/Stop sent by this program to heating/out). Compute and log heaters characteristics (listen heaters consumption log messages at domoticz/in and heating/in). Monitor ESP8266-ACS712/Heaters, ESP8266/Lighting and Raspberry/Alarm server and raise failure flag if one of them die (listen MQTT Will messages at domoticz/in).	Update --> JSON_API Get <--MQTT[domoticz/out]	iot_ESP8266_ACS712.ino	Get <- MQTT[heating/in] Get <- MQTT[domoticz/in] Send -- > MQTT[heating/out]	
iot_ALARM-SVR.js	Alarm server. Manage the CVQ6081 alarm Appliance : arm and disarm the Alarm using the Raspberry GPIO/Relay as a keyswitch and get alarm Alert state at CVQ6081 backpanel	Update --> MQTT[domoticz/in] Get <-- MQTT[domoticz/out]	NO		
~~iot_CVQ6081-ARM.cpp~~	~~Alarm server. Interfaces the CVQ6081 alarm PCB~~	~~NO~~	~~NO~~	~~Send/Get <--> PUBNUB[AlarmUserCommands]~~	
~~iot_CVQ6081.js~~	~~Allow to use DomoticZ Security Panel to arm/disarm the alarm. Monitor the alarm server and raise failure flag if alarm server doesnt answer correctly.~~	~~Update/Get --> JSON_API~~	~~iot_CVQ6081-ARM.cpp~~	~~Send/Get <--> PUBNUB[AlarmUserCommands]~~	
~~MainActivity.java~~ ~~AlarmCommandPanel.java~~ ~~PubnubKeys.java~~	~~Legacy Android app to arm/disarm Alarm and set configuration options~~	~~NO~~	~~iot_CVQ6081-ARM.cpp~~	~~Send/Get <--> PUBNUB[AlarmUserCommands]~~	