This PDF is regarding connecting creating a ESP32 Mesh network.
It uses libraries for connecting like, “painlessMesh.h”, which gives access to the class Schedule.
The scheduler class link below to the library, https://docs.arduino.cc/libraries/scheduler/

With the Mesh class, you’ll create all of the callback methods in which are used by the Mesh object in conjunction with Scheduler class object.

Reading through the code could get a bit confusing, but know each new ESP32 that is introduced with the code will join the mesh and start broadcasting its message to the group.

The Gateway node will be the one you’ll connect to via a simple UART serial connection and relay the message to a network of your choice such as a Raspberry PI, webhook or simple logging node.

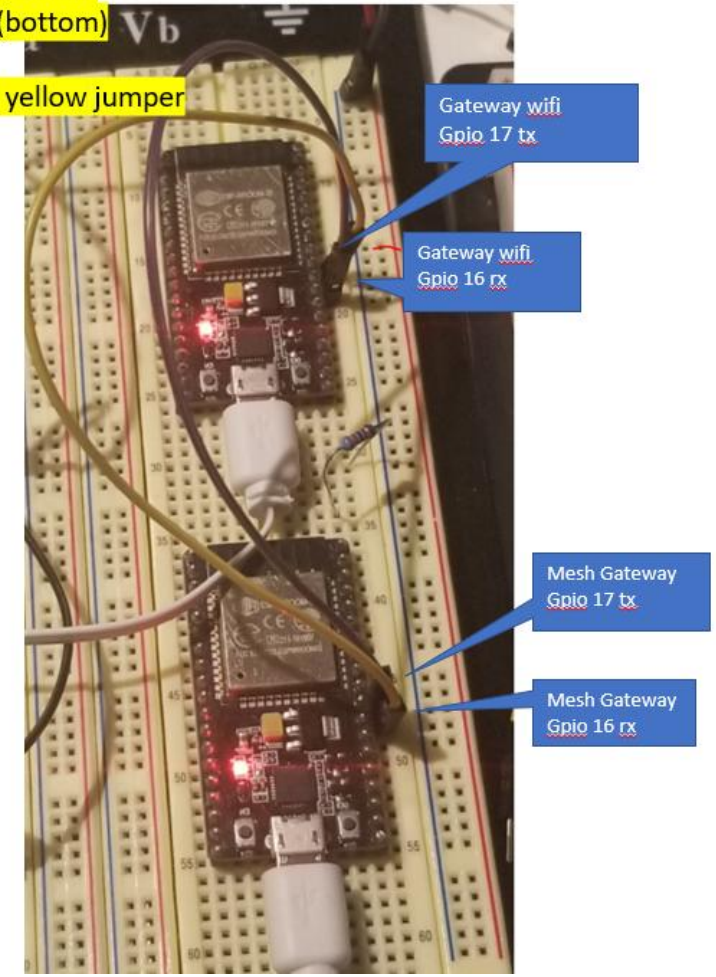
```
************  regular nodes ******************

#include <Arduino.h>
#include "painlessMesh.h"

#define   MESH_PREFIX    "khometekconn"
#define   MESH_PASSWORD   "L8*ITX93IFFKS"
#define   MESH_PORT      5555

Scheduler userScheduler; // to control your personal task
painlessMesh  mesh;

// User stub
void sendMessage() ; // Prototype so PlatformIO doesn't complain

Task taskSendMessage( TASK_SECOND * 1 , TASK_FOREVER, &sendMessage );

void sendMessage() {
  String msg = "Hi from ESP32_mqtt_connect_1 ";
  //msg += mesh.getNodeId();
  mesh.sendBroadcast( msg );
  taskSendMessage.setInterval(TASK_SECOND * 4);
}

// Needed for painless library
void receivedCallback( uint32_t from, String &msg ) {
  Serial.printf("startHere: Received from %u msg=%s\n", from, msg.c_str());
}

void newConnectionCallback(uint32_t nodeId) {
    Serial.printf("--> startHere: New Connection, nodeId = %u\n", nodeId);
}

void changedConnectionCallback() {
  Serial.printf("Changed connections\n");
}

void nodeTimeAdjustedCallback(int32_t offset) {
    Serial.printf("Adjusted time %u. Offset = %d\n", mesh.getNodeTime(),offset);
}


void setup() {
  Serial.begin(115200);
mesh.setDebugMsgTypes( ERROR | STARTUP );  // set before init() so that you can see startup messages

  mesh.init( MESH_PREFIX, MESH_PASSWORD, &userScheduler, MESH_PORT );
  mesh.onReceive(&receivedCallback);
  mesh.onNewConnection(&newConnectionCallback);
  mesh.onChangedConnections(&changedConnectionCallback);
  mesh.onNodeTimeAdjusted(&nodeTimeAdjustedCallback);
  userScheduler.addTask( taskSendMessage );
  taskSendMessage.enable();

}

void loop() {
  mesh.update();
}
```

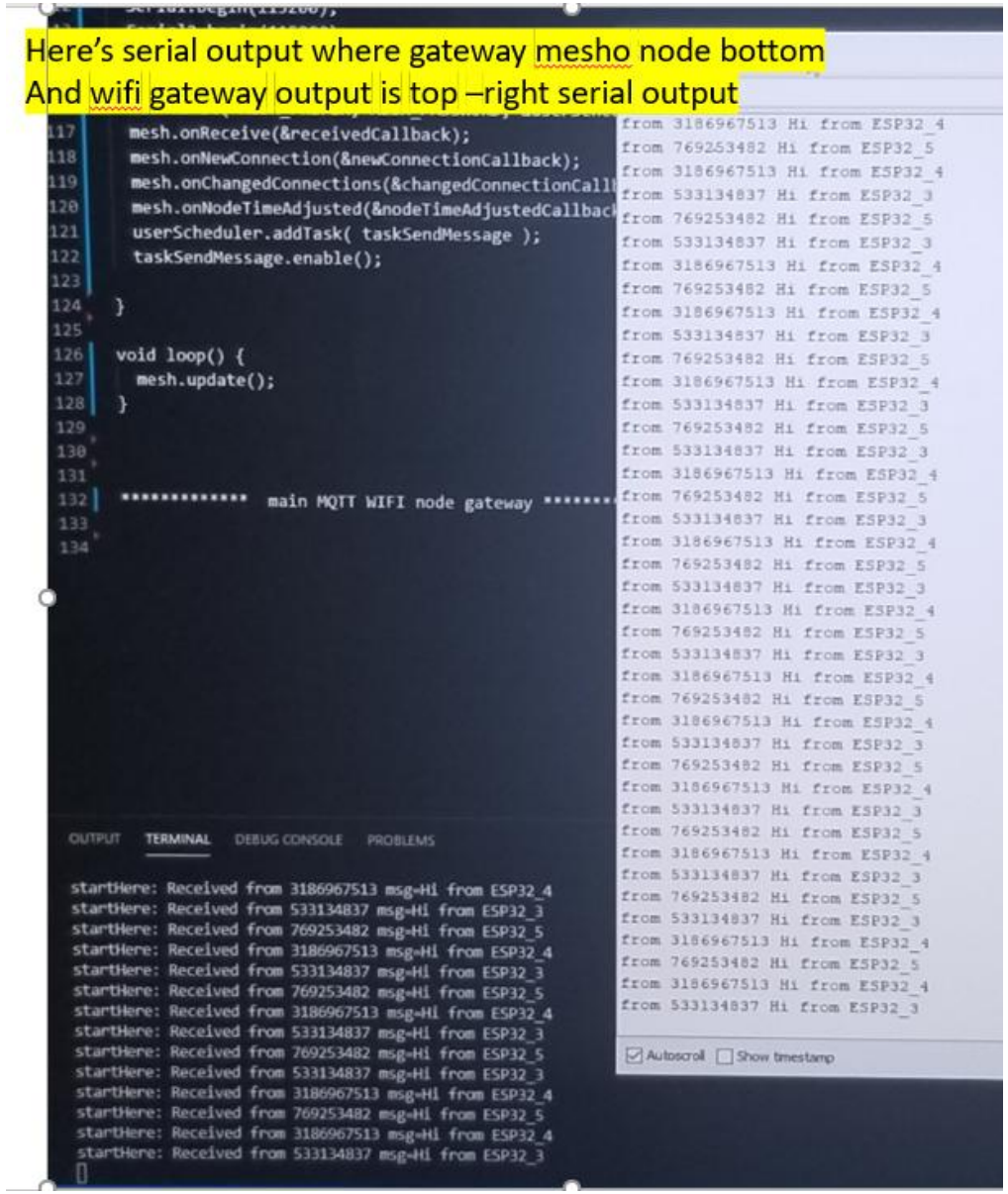
```
******************** gateway node **********************
 #include <Arduino.h>
#include "painlessMesh.h"
 #define   MESH_PREFIX     "khometekconn"
#define   MESH_PASSWORD   "L8*ITX93IFFKS"
#define   MESH_PORT       5555
 Scheduler userScheduler; // to control your personal task
painlessMesh  mesh;
 // User stub
void sendMessage() ; // Prototype so PlatformIO doesn't complain
 Task taskSendMessage( TASK_SECOND * 1 , TASK_FOREVER, &sendMessage );
 void sendMessage() {
  String msg = "Hi from ESP32_gateway_2 ";
  //msg += mesh.getNodeId();
  mesh.sendBroadcast( msg );
  taskSendMessage.setInterval(TASK_SECOND * 4);
  while(Serial2.available()){
    String info = Serial2.readStringUntil('\n');
    info = info+" from mqtt";
    mesh.sendBroadcast(info);
  }
}
 // Needed for painless library
void receivedCallback( uint32_t from, String &msg ) {
  Serial.printf("startHere: Received from %u msg=%s\n", from, msg.c_str());
  String strfrom = String(from);
  String data="from "+strfrom+" "+msg.c_str();
  Serial2.println(data.c_str());
}

void newConnectionCallback(uint32_t nodeId) {
    Serial.printf("--> startHere: New Connection, nodeId = %u\n", nodeId);
}

void changedConnectionCallback() {
  Serial.printf("Changed connections\n");
}

void nodeTimeAdjustedCallback(int32_t offset) {
    Serial.printf("Adjusted time %u. Offset = %d\n", mesh.getNodeTime(),offset);
}

void setup() {
  Serial.begin(115200);
  Serial2.begin(115200);
mesh.setDebugMsgTypes( ERROR | STARTUP );  // set before init() so that you can see startup messages

  mesh.init( MESH_PREFIX, MESH_PASSWORD, &userScheduler, MESH_PORT );
  mesh.onReceive(&receivedCallback);
  mesh.onNewConnection(&newConnectionCallback);
  mesh.onChangedConnections(&changedConnectionCallback);
  mesh.onNodeTimeAdjusted(&nodeTimeAdjustedCallback);
  userScheduler.addTask( taskSendMessage );
  taskSendMessage.enable();

}

void loop() {
  mesh.update();
}
```

```
*************  main MQTT WIFI node gateway *****************

#include <Arduino.h>

//connect to WIFI
//connect to MQTT
//in look publish everything to MQTT so visable by Node Red.
//or instead of loop, implement tashscheduler library.
//options are open on all fronts from the gateway node.

void setup() {
  Serial.begin(115200);
  Serial2.begin(115200);
}

void loop() {
  while(Serial.available()){
    Serial2.write(Serial.read());
  }

  while(Serial2.available()){
    Serial.write(Serial2.read());
  }

}
```

****** pin setup ****
SERIAL2 inbuilt UART pins
Gateway node -> gateway node wifi connected
P17 (tx) -> P16(rx)
P16 (rx) -> P17(tx)

Here's the illustration screenshot I tested connection between mesh node (bottom) to wifi node (top) speaking via Serial Uart (Serial2) Purple and yellow jumper Wires connected to other node.




Here's serial output where gateway mesho node bottom
And wifi gateway output is top –right serial output
mesh.onReceive(&receivedCallback);
mesh.onNewConnection(&newConnectionCallback);
mesh.onChangedConnections(&changedConnectionCall
mesh.onNodeTimeAdjusted(&nodeTimeAdjustedCallbac
userScheduler.addTask(taskSendMessage);
taskSendMessage.enable();
}
void loop() {
mesh.update();
}
********** main MQTT WIFI node gateway ********
OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS
startHere: Received from 3186967513 msg=Hi from ESP32_4
startHere: Received from 533134837 msg=Hi from ESP32_3
startHere: Received from 769253482 msg=Hi from ESP32_5
startHere: Received from 3186967513 msg=Hi from ESP32_4
startHere: Received from 533134837 msg=Hi from ESP32_3
startHere: Received from 769253482 msg=Hi from ESP32_5
startHere: Received from 3186967513 msg=Hi from ESP32_4
startHere: Received from 533134837 msg=Hi from ESP32_3
startHere: Received from 769253482 msg=Hi from ESP32_5
startHere: Received from 533134837 msg=Hi from ESP32_3
startHere: Received from 3186967513 msg=Hi from ESP32_4
startHere: Received from 769253482 msg=Hi from ESP32_5
startHere: Received from 3186967513 msg=Hi from ESP32_4
startHere: Received from 533134837 msg=Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 3186967513 Hi from ESP32_4
from 533134837 Hi from ESP32_3
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 3186967513 Hi from ESP32_4
from 533134837 Hi from ESP32_3
from 769253482 Hi from ESP32_5
from 3186967513 Hi from ESP32_4
from 533134837 Hi from ESP32_3
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
from 3186967513 Hi from ESP32_4
from 769253482 Hi from ESP32_5
from 533134837 Hi from ESP32_3
Autoscroll
Show timestamp