```
/**
 *  El-Client REST POST request to PUSHOVER notification service
    (adapted from rest example, following thingspeak example)
 */

#include <ELClient.h>
#include <ELClientRest.h>

/*****************************MODIF**********************************************/
#define MYTOKEN     "ah4y3r7wy7o8yw64kc2yapavpi98yo" // pushover credentials
#define MYUSER      "ue9gs4h56fecq34owfny77cgnapmfj"
#define MYTITLE     "ATS-Alarm"
#define MYMESSAGE   "SensorActivated!"
#define MYSOUND     "alien"

// optional
#define MYPRIORITY  "2"
#define MYRETRY     "60"
#define MYEXPIRE    "120"

#define BUFLEN 266

char path_data[BUFLEN];

/*****************************MODIF ENDS*****************************************/

// Initialize a connection to esp-link using the normal hardware serial port both for
// SLIP and for debug messages.
ELClient esp(&Serial, &Serial);

// Initialize a REST client on the connection to esp-link
ELClientRest rest(&esp);

boolean wifiConnected = false;

// Callback made from esp-link to notify of wifi status changes
// Here we print something out and set a global flag
void wifiCb(void *response) {
  ELClientResponse *res = (ELClientResponse*)response;
  if (res->argc() == 1) {
    uint8_t status;
    res->popArg(&status, 1);

    if(status == STATION_GOT_IP) {
      Serial.println("WIFI CONNECTED");
      wifiConnected = true;
    } else {
      Serial.print("WIFI NOT READY: ");
      Serial.println(status);
      wifiConnected = false;
    }
  }
}

void setup() {
  Serial.begin(115200);   // the baud rate here needs to match the esp-link config
  Serial.println("EL-Client starting!");

  // Sync-up with esp-link, this is required at the start of any sketch and initializes the
  // callbacks to the wifi status change callback. The callback gets called with the initial
  // status right after Sync() below completes.

  esp.wifiCb.attach(wifiCb); // wifi status change callback, optional
                             // (delete if not desired)

  bool ok;
  do {
    ok = esp.Sync();      // sync up with esp-link, blocks for up to 2 seconds
    if (!ok) Serial.println("EL-Client sync failed!");
  } while(!ok);
  Serial.println("EL-Client synced!");

  // Get immediate wifi status info for demo purposes. This is not normally used because the
  // wifi status callback registered above gets called immediately.
  esp.GetWifiStatus();
  ELClientPacket *packet;
  if ((packet=esp.WaitReturn()) != NULL) {
    Serial.print("Wifi status: ");
    Serial.println(packet->value);
  }

  // Set up the REST client to talk to api.pushover.net, this doesn't connect to that server,
  // it just sets-up stuff on the esp-link side
```

```
  /*****************************MODIF************************************************/

  int err = rest.begin("api.pushover.net", 80, true); // https, 443?

/*****************************MODIF ENDS ********************************************/
  if (err != 0) {
    Serial.print("REST begin failed: ");
    Serial.println(err);
    while(1) ;
  }
  Serial.println("EL-REST ready");

/*****************************MODIF (from thingspeak example) ***********************************************/

  // Copy the path and API key into the buffer
  sprintf(path_data, "%s", "token=");
  sprintf(path_data + strlen(path_data), "%s", MYTOKEN);

  // Copy the field number and value into the buffer
  // If you have more than one field to update,
  // repeat and change field1 to field2, field3, ...

  sprintf(path_data + strlen(path_data), "%s", "&user=");
  sprintf(path_data + strlen(path_data), "%s", MYUSER);

       sprintf(path_data + strlen(path_data), "%s", "&title=");
  sprintf(path_data + strlen(path_data), "%s", MYTITLE);

       sprintf(path_data + strlen(path_data), "%s", "&message=");
  sprintf(path_data + strlen(path_data), "%s", MYMESSAGE);

  sprintf(path_data + strlen(path_data), "%s", "&sound=");
  sprintf(path_data + strlen(path_data), "%s", MYSOUND);

/*****************************MODIF ENDS********************************************/

}

void loop() {
  // process any callbacks coming from esp_link
  esp.Process();

  // if we're connected make an HTTP request
  if(wifiConnected) {

/*****************************MODIF************************************************/

    rest.post(path_data, NULL);

/*****************************MODIF ENDS ********************************************/

    char response[BUFLEN];

    memset(response, 0, BUFLEN);
    uint16_t code = rest.waitResponse(response, BUFLEN);
    if(code == HTTP_STATUS_OK){
      Serial.println("ARDUINO: POST successful:"); // modif get->post
      Serial.println(response);
    } else {
      Serial.print("ARDUINO: POST failed: "); // modif get->post
      Serial.println(code);
    }
    delay(5000); // waits 5 s
  }
}
```