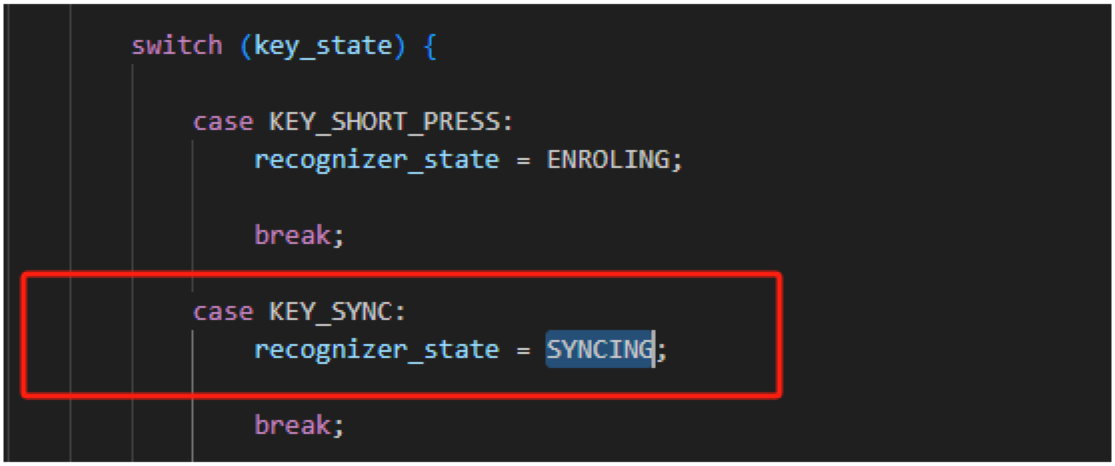
**Sozib Al Mamun**
**Embedded Software Engineer**

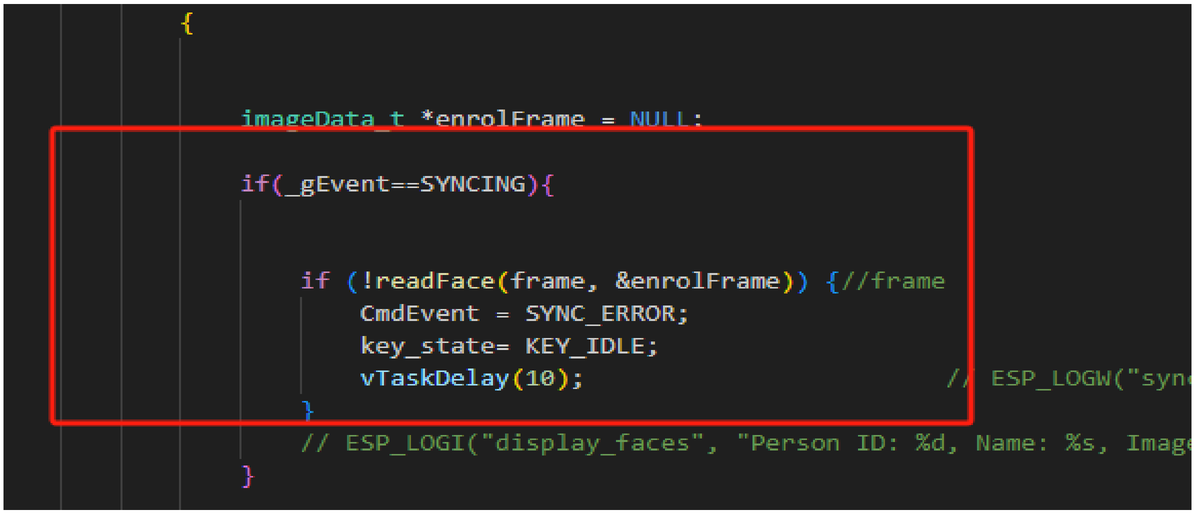
## Sync event

Sync event created for person enrollment and all data came from cloud like image(width, height) and person name. First received image and other info and saved it in sync directory and after successfully ACK server again sent sync command with person info. Then the enrollment process will be started. And again ACK sent to the cloud.



```
switch (key_state) {

    case KEY_SHORT_PRESS:
        recognizer_state = ENROLING;

        break;

    case KEY_SYNC:
        recognizer_state = SYNCING;

        break;
```

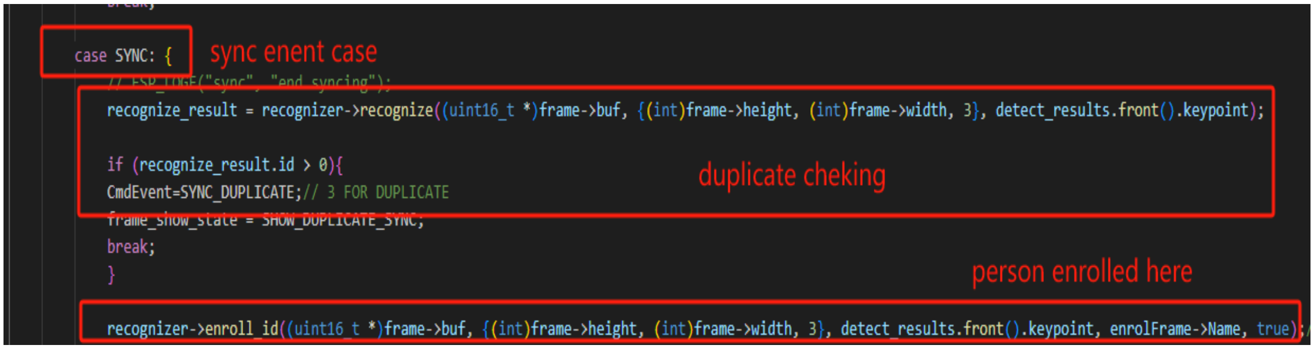
Here the sync event are generated .



```
{


    imageData_t *enrolFrame = NULL;

    if(_gEvent==SYNCING){


        if (!readFace(frame, &enrolFrame)) {//frame
            CmdEvent = SYNC_ERROR;
            key_state= KEY_IDLE;
            vTaskDelay(10);                         // ESP_LOGW("sync
        }
        // ESP_LOGI("display_faces", "Person ID: %d, Name: %s, Image
    }
```

And here check the event type and also read the person info and again back to the idle  event.

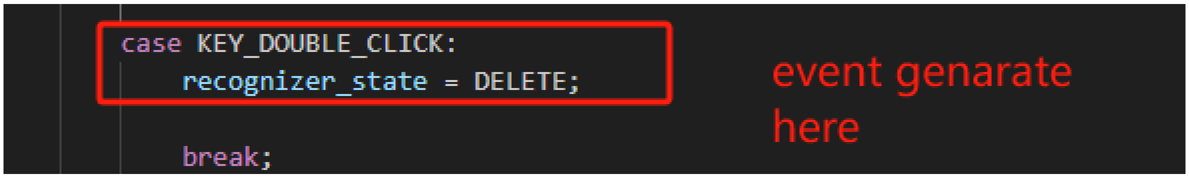
And after that going to face the detection process. And go to syncing event checking duplicate or not , then enrolled person and sent ACK to cloud.



```
    break;

case SYNC: {
    // ESP_LOGE("sync", "end syncing");
    recognize_result = recognizer->recognize((uint16_t *)frame->buf, {(int)frame->height, (int)frame->width, 3}, detect_results.front().keypoint);

    if (recognize_result.id > 0){
    CmdEvent=SYNC_DUPLICATE;// 3 FOR DUPLICATE
    frame_show_state = SHOW_DUPLICATE_SYNC;
    break;
    }

    recognizer->enroll_id((uint16_t *)frame->buf, {(int)frame->height, (int)frame->width, 3}, detect_results.front().keypoint, enrolFrame->Name, true);
```

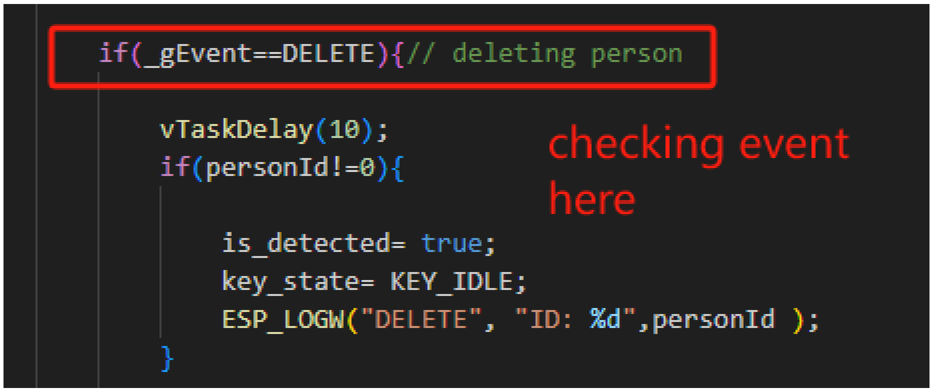
## Delete person by showing name

Event generate here



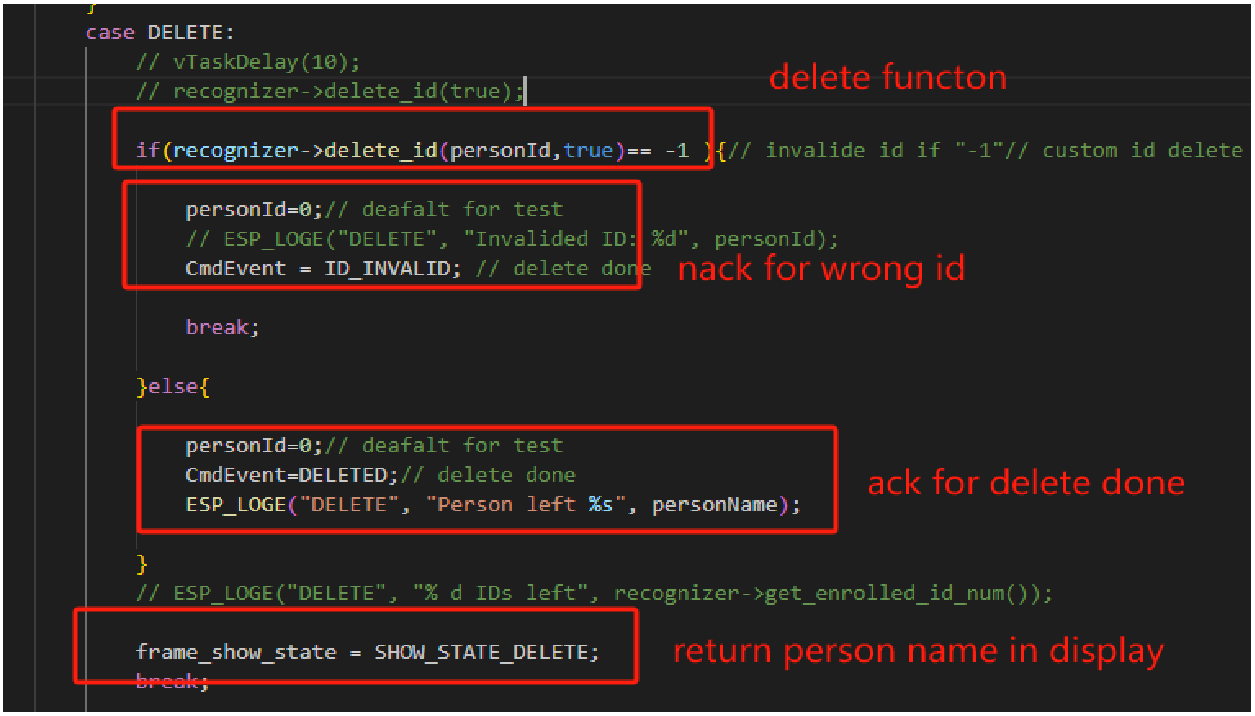
```
case KEY_DOUBLE_CLICK:
    recognizer_state = DELETE;

    break;
```

Event check



```
if(_gEvent==DELETE){// deleting person

    vTaskDelay(10);
    if(personId!=0){

        is_detected= true;
        key_state= KEY_IDLE;
        ESP_LOGW("DELETE", "ID: %d",personId );
    }
```

Preson delete process:



```
    }
    case DELETE:
        // vTaskDelay(10);
        // recognizer->delete_id(true);

        if(recognizer->delete_id(personId,true)== -1 ){// invalide id if "-1"// custom id delete

            personId=0;// deafalt for test
            // ESP_LOGE("DELETE", "Invalided ID: %d", personId);
            CmdEvent = ID_INVALID; // delete done

            break;

        }else{

            personId=0;// deafalt for test
            CmdEvent=DELETED;// delete done
            ESP_LOGE("DELETE", "Person left %s", personName);

        }
        // ESP_LOGE("DELETE", "% d IDs left", recognizer->get_enrolled_id_num());

        frame_show_state = SHOW_STATE_DELETE;
        break;
```

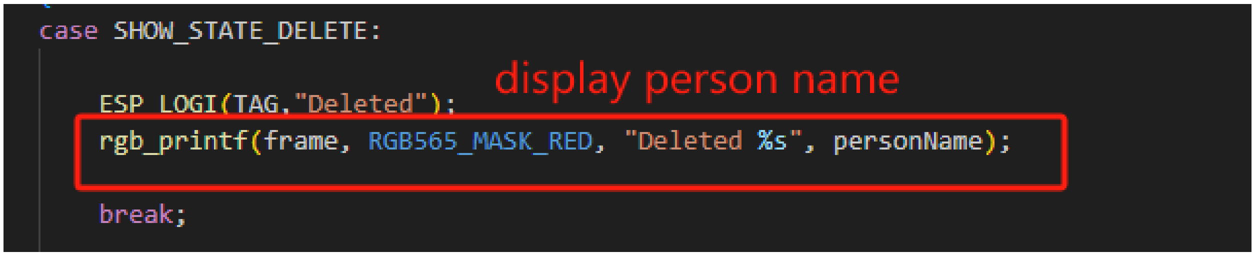
delete functon

nack for wrong id

ack for delete done

return person name in display

Display status

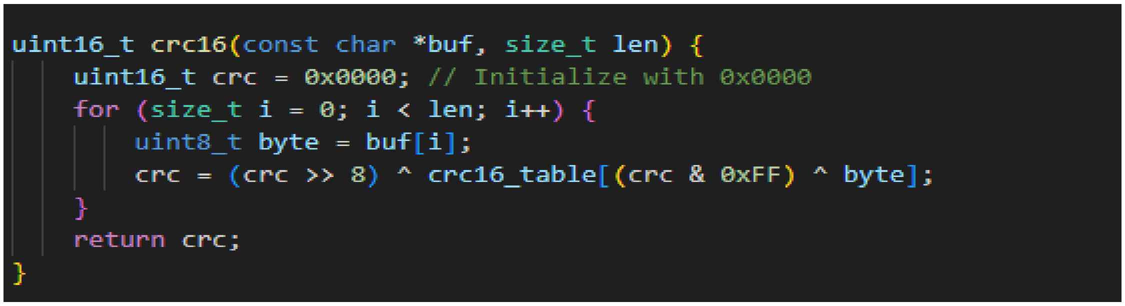


```
case SHOW_STATE_DELETE:

    ESP_LOGI(TAG,"Deleted");
    rgb_printf(frame, RGB565_MASK_RED, "Deleted %s", personName);

    break;
```

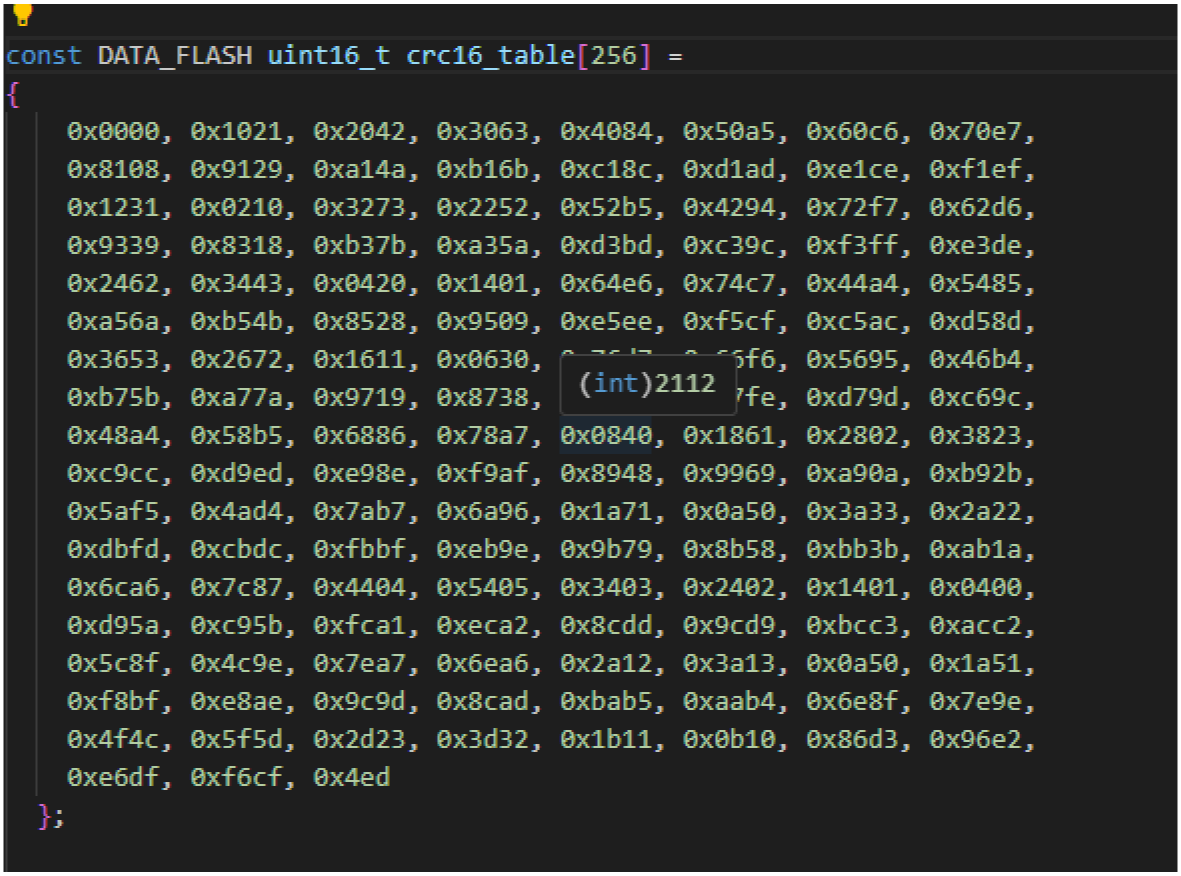
display person name

**CRC was some problem change crc table & function**

Previously calculated crc was changing so it's a problem. So change the crc table and function logic so it works.



```
uint16_t crc16(const char *buf, size_t len) {
    uint16_t crc = 0x0000; // Initialize with 0x0000
    for (size_t i = 0; i < len; i++) {
        uint8_t byte = buf[i];
        crc = (crc >> 8) ^ crc16_table[(crc & 0xFF) ^ byte];
    }
    return crc;
}
```



```
const DATA_FLASH uint16_t crc16_table[256] =
{
    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
    0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
    0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
    0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
    0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
    0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
    0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
    0xb75b, 0xa77a, 0x9719, 0x8738,  (int)2112  7fe, 0xd79d, 0xc69c,
    0x48a4, 0x58b5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
    0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
    0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a22,
    0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
    0x6ca6, 0x7c87, 0x4404, 0x5405, 0x3403, 0x2402, 0x1401, 0x0400,
    0xd95a, 0xc95b, 0xfca1, 0xeca2, 0x8cdd, 0x9cd9, 0xbcc3, 0xacc2,
    0x5c8f, 0x4c9e, 0x7ea7, 0x6ea6, 0x2a12, 0x3a13, 0x0a50, 0x1a51,
    0xf8bf, 0xe8ae, 0x9c9d, 0x8cad, 0xbab5, 0xaab4, 0x6e8f, 0x7e9e,
    0x4f4c, 0x5f5d, 0x2d23, 0x3d32, 0x1b11, 0x0b10, 0x86d3, 0x96e2,
    0xe6df, 0xf6cf, 0x4ed
  };
```