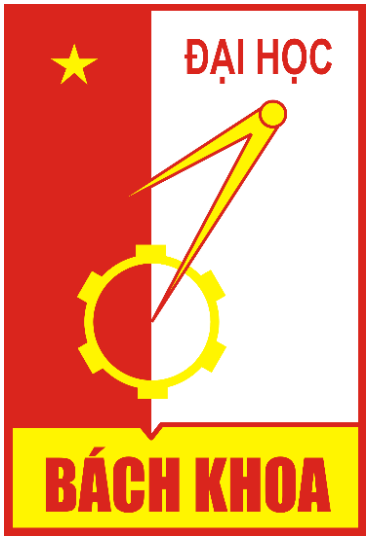
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện công nghệ thông tin & truyền thông



BÁO CÁO MÔN HỌC: PROJECT 3

Đề tài: Lập trình điều khiển module ESP32-camera chụp hình và hiển thị ảnh trên ứng dụng di động

GVHD: TS. Vũ Văn Thiệu

Sinh viên thực hiện: Đinh Trọng Đạt – 20173010

Hà Nội, tháng 1 - 2021

Mục Lục

Danh mục hình ảnh

I. GIỚI THIỆU ĐỀ TÀI

1. Mô tả đề tài

Đề tài này em sẽ lập trình cho module ESP32-camera chụp ảnh và gửi lên server, cùng với đó là lập trình cho server và ứng dụng di động điều khiển ESP32-camera chụp, hiển thị ảnh và lưu những ảnh đã chụp lên Cloud Storage của Google Firebase.

2. Các module phần cứng yêu cầu

- Module ESP32-camera: đây là module chính thực hiện việc chụp và gửi ảnh lên server.

Hình 1: Module ESP32-camera

- Mạch chuyển USB PL2303: dùng để kết nối module ESP32-camera với nguồn điện và máy tính để nạp code.

Hình 2: Mạch chuyển USB PL2303

3. Các ngôn ngữ lập trình sử dụng

- Sử dụng Arduino IDE để lập trình cho module ESP32-camera.

- Server: lập trình bằng PHP với XAMPP.

- Ứng dụng di động: lập trình ứng dụng đa nền tảng sử dụng framework React Native và Expo.

II. QUÁ TRÌNH THỰC HIỆN

1. Ý tưởng thực hiện

Do không thể gửi request trực tiếp từ server đến camera cho nên để điều khiển camera chụp ảnh em sẽ cho camera định kỳ gửi POST request để đọc file json "test.json" trên server, trong file json này sẽ chỉ chứa 1 tham số duy nhất là "capture", bình thường khi không chụp ảnh thì nó sẽ có giá trị {"capture": "off"}, khi đọc được giá trị {"capture": "on"} camera sẽ tiến hành chụp ảnh và upload ảnh chụp được lên server rồi thay đổi lại giá trị "capture" thành "off". Sau đó server sẽ lưu ảnh này vào thư mục "camera_upload".

Về phía ứng dụng điện thoại cũng sẽ dựa vào file "test.json" trên server để có thể điều khiển được việc chụp ảnh của camera, bằng cách gửi request thay đổi giá trị "capture" thành "on" khi muốn chụp ảnh. Sau khi chụp ảnh thành công ứng dụng sẽ lấy ảnh đã được lưu trên server để hiển thị và upload ảnh này lên Firebase Storage thông qua firebase api.

2. Lập trình cho module ESP32-camera

Các thư viện sử dụng:

- "Wifi.h" để kết nối camera với wifi

- "esp_http_client.h" để xử lý việc gửi request về server

- "ArduinoJson.h" để xử lý dữ liệu json đọc được từ server

Kết nối camera với wifi:

```
bool init_wifi()
{
  int connAttempts = 0;
  Serial.println("\r\nConnecting to: " + String(ssid));
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED ) {
    delay(500);
    Serial.print(".");
    if (connAttempts > 10) return false;
    connAttempts++;
  }
  return true;
}
```

Hình 3: Kết nối camera với wifi

Gửi POST request lên server:

```
static esp_err_t send_post_request()
{
   char *rev_data =(char*) malloc(20);
    if (rev_data == NULL) {
        ESP_LOGE(TAG, "Cannot malloc http receive buffer");
    }

  esp_http_client_handle_t http_client;

  esp_http_client_config_t config_client = {0};
  config_client.url = post_url_json;
  config_client.event_handler = _http_event_handler;
  config_client.method = HTTP_METHOD_POST;

  http_client = esp_http_client_init(&config_client);

  esp_err_t err = esp_http_client_perform(http_client);
  if (err == ESP_OK) {
    Serial.print("esp_http_client_get_status_code: ");
    Serial.println(esp_http_client_get_status_code(http_client));
  }

   int read_len = esp_http_client_read(http_client,rev_data,content_length);
  data_esp32=String(rev_data);
  esp_http_client_cleanup(http_client);
}
```

Hình 4: Gửi POST request lên server

Xử lý data nhận được:

```
void process_command()
{

       String result = data_esp32;
// ===============Parse JSON================
      int size = result.length()+1;
      char json[size];
      result.toCharArray(json, size);
      DynamicJsonBuffer jsonBuffer(size);
      JsonObject& json_parsed = jsonBuffer.parseObject(json);
      if (!json_parsed.success())
      {
        Serial.println("parseObject() failed");
      }
      else Serial.println("Parse OK");
          if (strcmp(json_parsed["capture"], "on") == 0)
              {
          Serial.println("TAKE A PHOTO");
              take_send_photo();
              }

          if (strcmp(json_parsed["capture"], "off") == 0)
              {
          Serial.println("NO TAKING PHPTO");
              }


}
```

Hình 5:Xử lý data nhận được

Chụp và gửi ảnh lên server:

```
static esp_err_t take_send_photo()
{
  Serial.println("Taking picture...");
  camera_fb_t * fb = NULL;
  esp_err_t res = ESP_OK;

  fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("Camera capture failed");
    return ESP_FAIL;
  }

  esp_http_client_handle_t http_client;

  esp_http_client_config_t config_client = {0};
  config_client.url = post_url;
  config_client.event_handler = _http_event_handler;
  config_client.method = HTTP_METHOD_POST;

  http_client = esp_http_client_init(&config_client);

  esp_http_client_set_post_field(http_client, (const char *)fb->buf, fb->len);

  esp_http_client_set_header(http_client, "Content-Type", "image/jpg");

  esp_err_t err = esp_http_client_perform(http_client);
  if (err == ESP_OK) {
    Serial.print("esp_http_client_get_status_code: ");
    Serial.println(esp_http_client_get_status_code(http_client));
  }
  int content_length =  esp_http_client_fetch_headers(http_client);
  Serial.print("length of data receive:");
   Serial.print(content_length);
  esp_http_client_cleanup(http_client);

  esp_camera_fb_return(fb);
}
```

Hình 6: Chụp và gửi ảnh lên server

3. Lập trình PHP server

Ở đây em sử dụng XAMPP để tạo server.

File camera_upload.php xử lý dữ liệu hình ảnh gửi lên từ camera:

```php
<?php
$jsonString = file_get_contents( filename: "test/test.json");
$data = json_decode($jsonString,  assoc: true);
$received = file_get_contents( filename: 'php://input');
$fileToWrite = "camera_upload/upload_esp32.jpg";
file_put_contents($fileToWrite, $received);

$data['capture'] = "off";
$newJsonString = json_encode($data);
file_put_contents( filename: "test/test.json", $newJsonString);
?>
```

Hình 7: File camera_upload.php

Còn lại là các file test chứa giao diện cơ bản và các chức năng chụp, hiển thị ảnh để tiện cho việc xây dựng ứng dụng di động.

4. Lập trình ứng dụng di động

Trước hết cần cài đặt một số thứ liên quan để có thể bắt đầu lập trình như:

- Expo CLI: để tạo ứng dụng React Native.

- Nodejs để quản lý package với npm.

- 1 IDE bất kỳ có thể lập trình được javascript.

Do sử dụng expo để tạo ứng dụng nên không cần thiết phải cài đặt Android Studio.

Ứng dụng sẽ gồm 2 phần chính là chụp ảnh và xem ảnh đã chụp được lưu trên firebase storage.

Để chụp ảnh, ở đây em sử dụng thư viện axios để gửi POST request lên server thay đổi giá trị của file “test.json”:

```
capture = ()=>{
    axios.post(
        "http://"+host+"/ESP32/esp32_cam_upload_control.php",
        "CAPTURE_ON="
    )
        .then((res)=>{
            this.setState( state: {
                isCapturing: true,
                captureState: "Taking photo..."
            });
            t = setInterval( handler: ()=>{
                if(this.state.isCapturing){
                    axios.post(
                        "http://"+host+"/ESP32/sync_allpages.php",
                        "val_button="
                    )
                        .then((res)=>{
                            if(res.data == "DONE!"){
                                this.setState( state: {
                                    captureState: "Done!",
                                    isCapturing: false
                                });
                                this.upload();
                                clearInterval(t);
                            }
                        })
                        .catch((err)=>{
                            console.error(err);
                        })
                }
            }, timeout: 1000)
        })
        .catch((err)=>{
            console.error(err);
        })
}
```

Hình 8: Gửi request lên server từ app di động

Sau đó dùng firebase api để lưu ảnh đã chụp được lên cloud storage:

```
upload = async ()=>{
    const response = await fetch( input: 'http://'+host+'/ESP32/camera_upload/upload_esp32.jpg?time='+Date.now());
    const blob = await response.blob();
    let metadata = {
        contentType: 'image/jpeg',
    };
    let currentDate = new Date();
    let datetime = "" + currentDate.getDate() + "-"
        + (currentDate.getMonth()+1)  + "-"
        + currentDate.getFullYear() + " "
        + currentDate.getHours() + ":"
        + currentDate.getMinutes() + ":"
        + currentDate.getSeconds();
    let fileName = datetime + ".jpg";
    let ref = firebaseApp.storage().ref().child( path: 'captured/' + fileName);
    return ref.put(blob, metadata);
}
```

Hình 9: Upload ảnh lên firebase storage

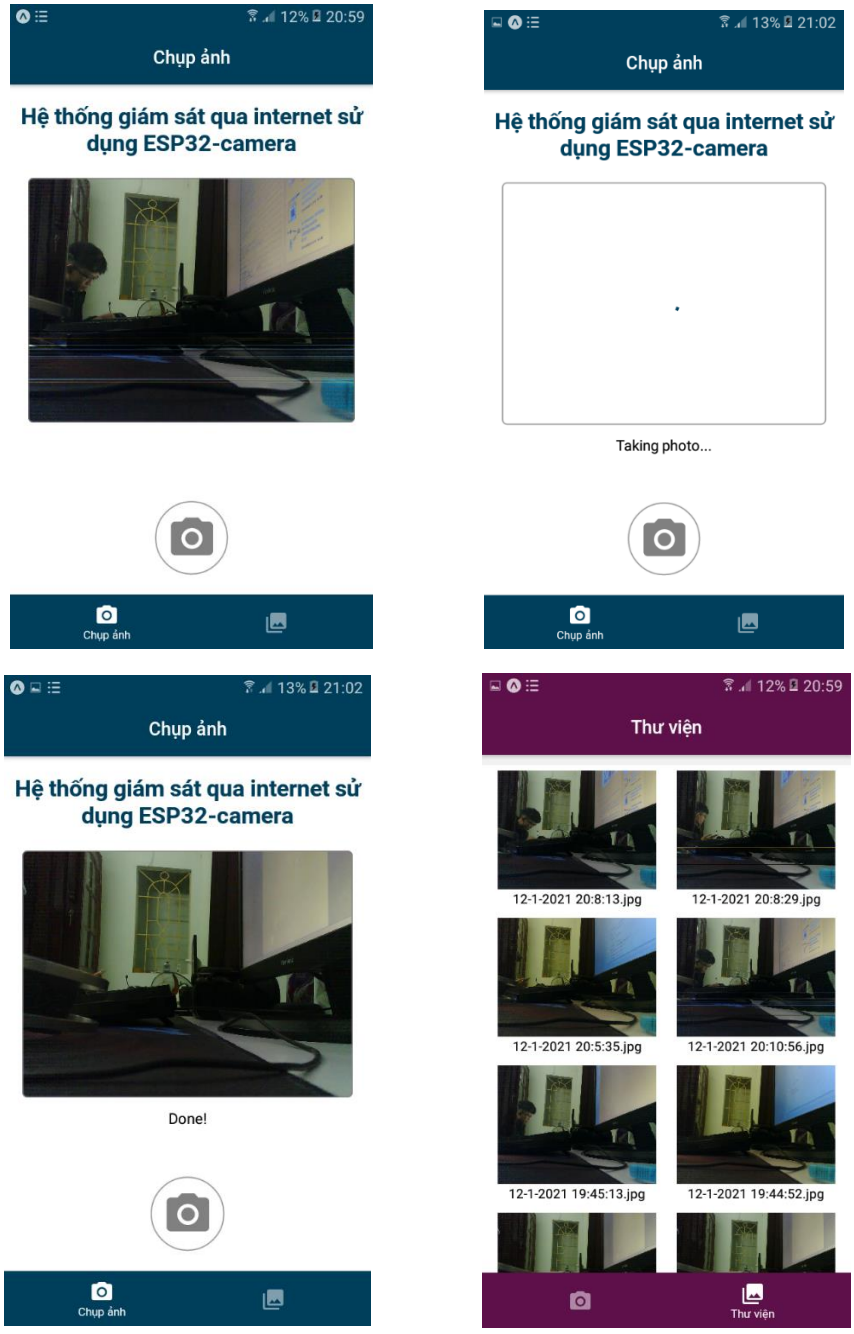
Để hiện thị ảnh trong thư viện em cũng dùng firebase api để lấy url của các ảnh đã chụp và hiển thị các ảnh này trên ứng dụng.

```
let listRef = firebaseApp.storage().ref().child( path: 'captured');
listRef.listAll().then(function(res : ListResult ) {
    let count = 0;
    let check = 0;
    res.items.forEach(function(itemRef : Reference ) {
        count++;
    });
    res.items.forEach(function(itemRef : Reference ) {
        itemRef.getDownloadURL().then(url=>{
            item.unshift(JSON.parse(JSON.stringify( value: {"url": url, "name": itemRef.name})));
            check++;
            if(check === count) isLoaded = true;
        });
    });
}).catch(function(e) {
    console.error(e);
});
```

Hình 10: Lấy dữ liệu ảnh từ firebase

III. KẾT QUẢ ĐẠT ĐƯỢC

Sau đây là ảnh của một số màn hình ứng dụng di động mà em đã làm được:



Hình 11: Demo ứng dụng

Source code của ứng dụng và các phần liên quan:
https://github.com/trongdat1512/project3_esp32

IV. TÀI LIỆU THAM KHẢO

1. Hướng dẫn nối dây, cài đặt môi trường cho Arduino IDE:

https://luuvachiase.net/index.php/2019/12/27/esp32-camera-cai-dat-moi-truong-arduino-ide-va-nap-chuong-trinh/

2. Tài liệu liên quan đến lập trình cho module ESP32-camera:

https://randomnerdtutorials.com/esp32-cam-take-photo-display-web-server/

https://luuvachiase.net/index.php/2020/01/11/esp32-camera-dieu-khien-chup-hinh-esp32-camera-qua-ineternet-hien-thi-hinh-anh-len-trinh-duyet/

3. Tài liệu liên quan đến react native:

https://reactnative.dev/docs/getting-started

https://docs.expo.io/

4. Tài liệu liên quan đến firebase api:

https://firebase.google.com/docs/storage/web/start