

به نام خدا



---

سیستم های نهفته بی درنگ

پروژه

---

استاد:

دکتر غلامپور

دانشجویان:

حسین انجیدنی 400100746

تیر ماه 1403

در فایل ویدیویی به بررسی عملکرد برنامه پرداختیم، اکنون به سراغ بررسی کمی تکنیکال تر میرویم.

## Sound Event Detector

فایل Main.py

```
broker_address = "localhost" # or the IP address of your broker
broker_port = 1883

# Define the topic you want to publish to
topic = "SED/label"
client = mqtt.Client()
client.connect(broker_address, broker_port, 60)

def add_event(filename, time, event_name, file_path):
    url = 'http://localhost:8080/add' # Replace with your server address
    params = {
        'filename': filename,
        'time': time,
        'event_name': event_name
    }
    # Read wave data from a file
    with open(file_path, 'rb') as f:
        wave_data = f.read()

    # Make a GET request to the /add endpoint
    try:
        response = requests.get(url, params=params, data=wave_data)
        if response.status_code == 200:
            print("File added successfully")
        else:
            print("Failed to add file. Status code:", response.status_code)
    except requests.exceptions.RequestException as e:
        print("Error:", e)

def add_status():
    url = 'http://localhost:8080/status' # Replace with your server address
    params = {
        'cpu_usage': str(psutil.cpu_percent(interval=1)),
        'ram_usage': str(psutil.virtual_memory().used),
        'time': str(int(time.time()))
    }

    # Make a GET request to the /add endpoint
    try:
        response = requests.get(url, params=params)
        if response.status_code == 200:
            print("Status added successfully")
        else:
            print("Failed to add file. Status code:", response.status_code)
    except requests.exceptions.RequestException as e:
        print("Error:", e)
```

در این بخش از کد به ترتیب به تعریف آدرس ها و توابع مورد نیاز برای اتصال به سرور پرداختیم.

```

99     p = pyaudio.PyAudio()
100     stream = p.open(format=FORMAT,
101                      channels=CHANNELS,
102                      rate=RATE,
103                      input=True,
104                      frames_per_buffer=CHUNK)
105     frames = []
106     for i in range(3):
107         data = stream.read(CHUNK)
108         frames.append(data)
109     # print(frames)
110     while True:
111         try:
112             data = stream.read(CHUNK)
113             # print("start")
114             frames.append(data)
115             # print()
116             bb = b''.join(frames[-3:])
117             waveform = np.frombuffer(bb, dtype=np.int16)
118             waveform = np.array(waveform, dtype=np.float32)
119             waveform = waveform / tf.int16.max
120             tim = time.time()
121             scores, embeddings, spectrogram = model(waveform)
122             scores_np = scores.numpy()
123             spectrogram_np = spectrogram.numpy()
124             max_score = scores_np.mean(axis=0).max()
125             # print(np.shape(scores_np))
126             infered_class = class_names[scores_np.argmax()]
127             print(infered_class, max_score)
128             # and infered_class != "Speech" and infered_class != "Silence"
129             if max_score > 0.5 and infered_class != "Silence":
130                 frames_to_save = frames[-int(RATE / CHUNK * RECORD_SECONDS):]
131                 audio_data = np.frombuffer(b''.join(frames_to_save), dtype=np.int16)
132                 fname = str(tim) + ".wav"
133                 sf.write("out/"+fname, audio_data, RATE)
134                 add_event(fname, str(int(tim)), infered_class, "out/"+fname)
135                 add_status()
136                 client.publish(topic, infered_class)
137                 print(infered_class)
138             except Exception as e:
139                 print(e)
140                 stream.stop_stream()
141                 stream.close()
142                 p.terminate()
143                 time.sleep(2)
144                 p = pyaudio.PyAudio()
145                 stream = p.open(format=FORMAT,
146                                channels=CHANNELS,
147                                rate=RATE,
148                                input=True,
149                                frames_per_buffer=CHUNK)
150                 continue
151     client.disconnect()

```

در این بخش نیز به استفاده از مدل استریم صدا ذخیره کردن و همچنین ارسال داده های نهایی به سرور پرداختیم. معیار زمان از لحظه ای که فایل صوتی ضبط شده مشخص است و معیار زمان نهایی که در video در صفحه سرور با عنوان received time قرار داده شده است

```

26 // Callback when a message arrives
27 void on_message(struct mosquitto *mosq, void *userdata, const struct mosquitto_message *message) {
28     char* payload = (char*) message->payload;
29     cout << "Message arrived on topic: " << message->topic << " Message: " << payload << endl;
30
31     // Insert message into MySQL database
32     try {
33         mysql::MySQL_Driver *driver;
34         Connection *con;
35         PreparedStatement *pstmt;
36
37         driver = mysql::get_mysql_driver_instance();
38         con = driver->connect(DB_HOST, DB_USER, DB_PASS);
39         con->setSchema(DB_NAME);
40
41         pstmt = con->prepareStatement("INSERT INTO micro_status(device, temp, time) VALUES (?, ?, ?)");
42         pstmt->setString(1, "1");
43         pstmt->setString(2, payload);
44         pstmt->setString(3, std::to_string(std::time(0))); // Get current time as string
45         pstmt->execute();
46
47         delete pstmt;
48         delete con;
49     } catch (SQLException &e) {
50         cout << "MySQL error: " << e.what() << endl;
51     }
52 }
53

```

در این فایل یک بخش بر روی ترد اصلی در یک حلقه همواره بررسی میکند که آیا در داخل broker که به آن سابسکرایب ایا داده جدیدی پابلیش شده یا خیر.

توجه کنید که در این بخش پس از اطلاع از داده جدید روی یک ترد دیگر تابع on\_message صدا زده میشود و با دیتابیس mysql ارتباط برقرار میکند.

## فولدر WebServer:

در این پوشه که تماماً به زبان سی پیاده سازی شده است اتصال به سرور توسط لایبریری سی بررسی شده است. به علت حجم بالای این کد ها به بررسی آن نمیردازم و تنها نکته حائز اهمیت آن است که در این بخش به دو روش فایل انتقال پیدا میکند که مورد تاکید دستیار آموزشی نیز بود: روش اول با استفاده از Rest API و روش دوم Shared memory که با تغییر آدرس موجود در در متغیر full\_path میتوان به روش دوم رسید که آن را در ویدیو دیدیم و کد روش اول را نیز ضمیمه میکنم

```

130 void serve_audio() {
131     std::string audio_file = req_.target().to_string();
132     std::string full_path = audio_file.erase(0,1);
133     std::cout<<full_path<<std::endl;
134     std::ifstream file(full_path.c_str(), std::ios::in | std::ios::binary);
135     if (!file) {
136         return send_bad_response(http::status::not_found, "Audio file not found");
137     }
138
139     std::stringstream ss;
140     ss << file.rdbuf();
141     std::string content = ss.str();
142
143     http::response<http::string_body> res{http::status::ok, req_.version()};
144     res.set(http::field::content_type, "audio/wav");
145     res.body() = content;
146     res.prepare_payload();
147
148     return send_response(std::move(res));
149 }

```

همچنین در این پوشه یک فایل `index.html` وجود دارد که داده ها را با استفاده از متد گت از `api` های نوشته با کمک جاوا اسکریپت دریافت میکند و به کاربر نمایش میدهد.

## بررسی ESP:

متاسفانه سایت جوان الکترونیک با توجه به کسری قطعه و با اطلاع به بنده مازول **ESP8266** را برایم ارسال کردند و بنده با اطلاع از این مهم که دارای بلوتوث و وایفای است آن را تهیه کردم. اما در پیاده سازی متوجه شدم که از سنسور دمای داخلی میکرو محروم است.

### How to read ESP8266 inside micro temperture



If you're looking to read the internal temperature of the ESP8266 microcontroller itself, it's important to note that the ESP8266 does not have an internal temperature sensor built into the chip. Therefore, directly reading the internal temperature of the ESP8266 is not possible with the hardware alone.

However, you can indirectly monitor the temperature near the ESP8266 chip using an external temperature sensor such as the DS18B20 or the DHT11/DHT22 as mentioned earlier. These sensors can be placed close to the ESP8266 on your circuit board or in the same enclosure, giving you a good approximation of the temperature in the vicinity of the ESP8266.

ادامه این بخش را با گزارش کردن پایه A0 به عنوان جایگزین مقدار دما ادامه دادم.

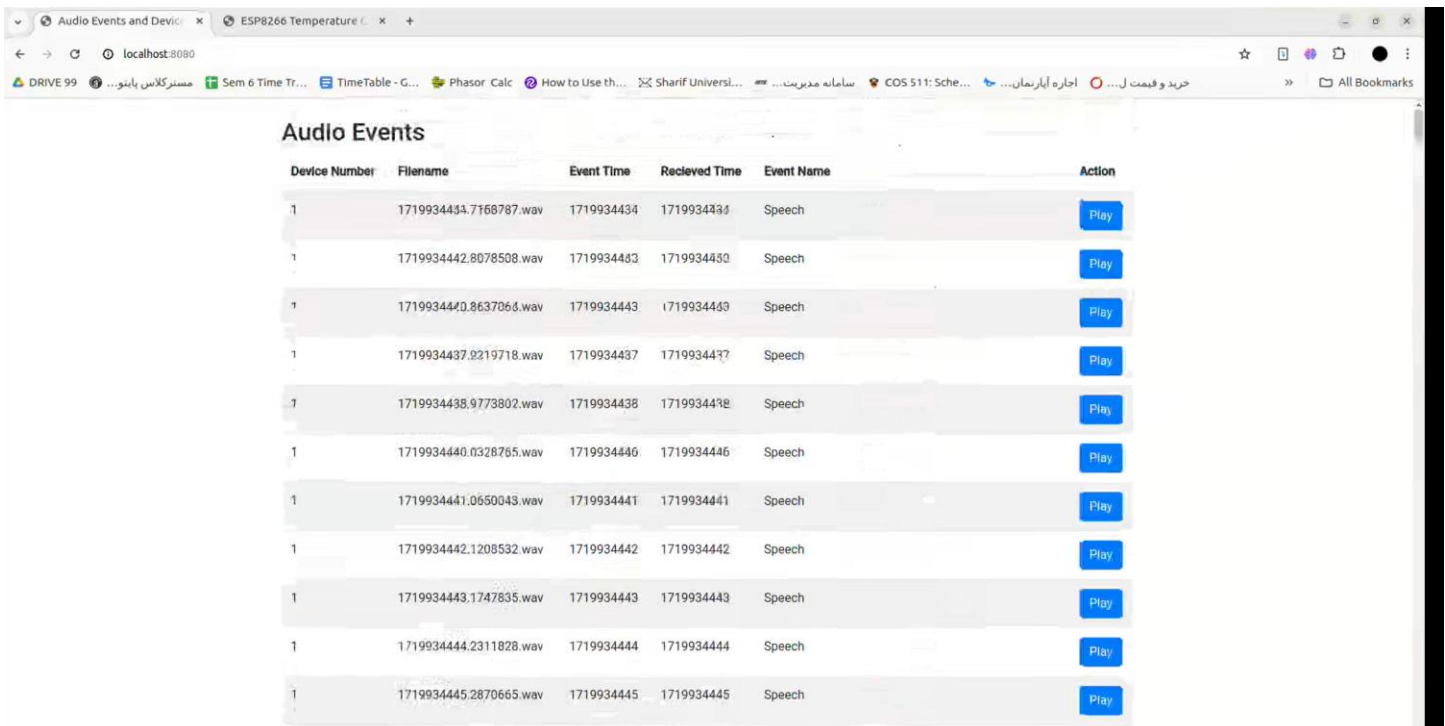
کد های مرتبط با ESP در پوشه ESP موجود است.

اکنون چند تصویر از محیط های عملیاتی برنامه میآورم:

```
hosein@hoseinontheho:~/Emb/project/server/WebServer/build$ sudo systemctl status web.service
[sudo] password for hosein:
● web.service - Web Server
   Loaded: loaded (/etc/systemd/system/web.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2024-07-02 14:19:10 +0330; 4h 43min ago
 Main PID: 622 (WebServer)
    Tasks: 1 (limit: 18939)
   Memory: 5.7M
      CPU: 43ms
   CGroup: /system.slice/web.service
           └─622 /home/hosein/Emb/project/server/WebServer/build/WebServer

14:19:10 02 جولای hoseinontheho systemd[1]: Started Web Server.
hosein@hoseinontheho:~/Emb/project/server/WebServer/build$
```

سرویس اجرا کننده WebServer



Device Number	Filename	Event Time	Recieved Time	Event Name	Action
1	1719934434.7166787.wav	1719934434	1719934434	Speech	Play
1	1719934442.8078508.wav	1719934443	1719934450	Speech	Play
1	1719934440.8637064.wav	1719934443	1719934443	Speech	Play
1	1719934437.9219718.wav	1719934437	1719934437	Speech	Play
1	1719934438.9773802.wav	1719934438	1719934438	Speech	Play
1	1719934440.0328765.wav	1719934446	1719934446	Speech	Play
1	1719934441.0650043.wav	1719934441	1719934441	Speech	Play
1	1719934442.1208532.wav	1719934442	1719934442	Speech	Play
1	1719934443.1747635.wav	1719934443	1719934443	Speech	Play
1	1719934444.2311828.wav	1719934444	1719934444	Speech	Play
1	1719934445.2870665.wav	1719934445	1719934445	Speech	Play

صفحه اصلی سرور دارای UI پیاده سازی شده با Bootstrap و JS (شایان ذکر است که AJAX پلتفرم بهتری برای سینک و نمایش داده ها با mysql اراته میکرد اما به علت مضیقه زمانی فرصت پیاده سازی آن نبود).









## Device Status

CPU Usage	RAM Usage	Time
1	4958896128	1719934485
1	4965994496	1719934438
1	4978984064	1719934432
1	4977102848	1719934438
1	4984897536	1719934439
1	4979499008	1719934441

وضعیت دستگاه شامل مقدار رم استفاده شده به بایت و درصد استفاده از cpu



صفحه تنظیم نرخ ارسال داده (از طراحی دیفالت جاوا اسکریپت و HTML استفاده شده).

بنده دو فایل ویدیویی ضمیمه کرده ام که یک فایل دارای صدا و از موبایل ضبط شده و یک فایل از کامپیوتر ضبط شده است.

زمان زیادی برای هر بخش این پروژه صرف کرده ام فلذا در صورت امکان از دستیاران آموزشی خواهمشدم در صورت بروز هرگونه ابهام در مورد پیاده سازی بنده لطفاً به بنده اطلاع دهند و بنده تمام آمادگی لازم به جهت ارائه حضوری و محازی و تهیه و فیلم و عکس و توضیحات بیشتر را دارم.