

Perancangan dan Implementasi Alat Pemanggil Perawat Nirkabel Berbasis Internet of Things

Mohamad Ali Muflih
Telkom University Kampus
Jakarta
Jakarta, Indonesia
movelhz@student.telkomuniversit
y.ac.id

Muhammad Royhan
Telkom University Kampus
Jakarta
Jakarta, Indonesia
roihani@telkomuniversit.ac.id

Abstrak — Rumah sakit adalah wadah fasilitas kesehatan yang ada pada lingkup masyarakat, hampir setiap daerah memiliki fasilitas kesehatan seperti rumah sakit, selain rumah sakit terdapat klinik klinik yang memiliki fungsi dan tujuan yang sama, pada fasilitas kesehatan terdapat ruang rawat inap dimana dalam keadaan tertentu pasien diharuskan menginap agar dapat di monitor secara langsung oleh perawat hingga kondisi membaik, petugas rumah sakit tidak menjaga langsung ruang rawat inap, ketika ada emergency dari pasien perawat menjadi kurang cepat tanggap dalam menanggapi situasi ini, untuk itu dirancang suatu konsep alat yang bernama Nurse Call yang menjadi solusi dari masalah tersebut, alat ini menjadi sebuah solusi teknologi untuk meningkatkan efisiensi dalam lingkup perawat kesehatan. Tujuan dari perancangan ini adalah mengembangkan sebuah sistem panggilan perawat yang nirkabel dan terintegrasi, memungkinkan pasien untuk dengan cepat meminta bantuan dari perawat atau petugas medis dan memungkinkan perawat merespons panggilan dengan lebih efisien. Terdapat 3 tombol pada rangkaian konsep inovasi nurse call ini, dimana ketika pasien menekal tombol merah akan memanggil perawat yang berpesan emergency, kemudian tombol hijau untuk memanggil perawat ketika membutuhkan bantuan dalam ruangan, dan tombol yang terakhir ditekan oleh perawat ketika sudah datang ke ruangan untuk mematikan command yang dikirimkan. Pengujian prototype ini dilakukan dengan melakukan simulasi percobaan pada ruang pasien dan diterima oleh server yang didalamnya berisi petugas kesehatan. Notifikasi pada telegram, sensor suara dari buzzer yang berbunyi, LED yang menyala, serta menampilkan pesan pada LCD menjadi output ketika push button ditekan, setiap push button akan mengeluarkan output yang berbeda, tergantung tombol mana yang ditekan. Dalam kesimpulannya, perancangan ini bertujuan untuk menciptakan sistem panggilan perawat yang efisien, cepat, dan terintegrasi dengan teknologi terkini. Dengan demikian, sistem ini diharapkan dapat meningkatkan kualitas perawatan pasien dan memastikan respons yang lebih cepat terhadap panggilan bantuan

Kata kunci: *NodeMCU ESP8266, Power Switching, Mikrocontroller, Rumah Sakit, Kesehatan, Nurse Call*

I. PENDAHULUAN

Nurse call berbasis kabel adalah sistem nurse call yang menggunakan kabel fisik untuk menghubungkan perangkat-perangkat dalam jaringan. Dalam sistem ini, kabel digunakan sebagai jalur komunikasi untuk mentransfer sinyal dan informasi antara perangkat-perangkat yang terhubung.

Dalam *nurse call* berbasis kabel, setiap perangkat atau tombol panggilan dihubungkan ke jaringan kabel yang menghubungkan perangkat tersebut dengan stasiun perawat atau pusat kontrol. Saat pasien menggunakan tombol panggilan atau memicu perangkat lainnya, sinyal diteruskan melalui kabel ke stasiun perawat, dan perawat atau staf medis yang bertanggung jawab akan menerima pemberitahuan atau panggilan tersebut.

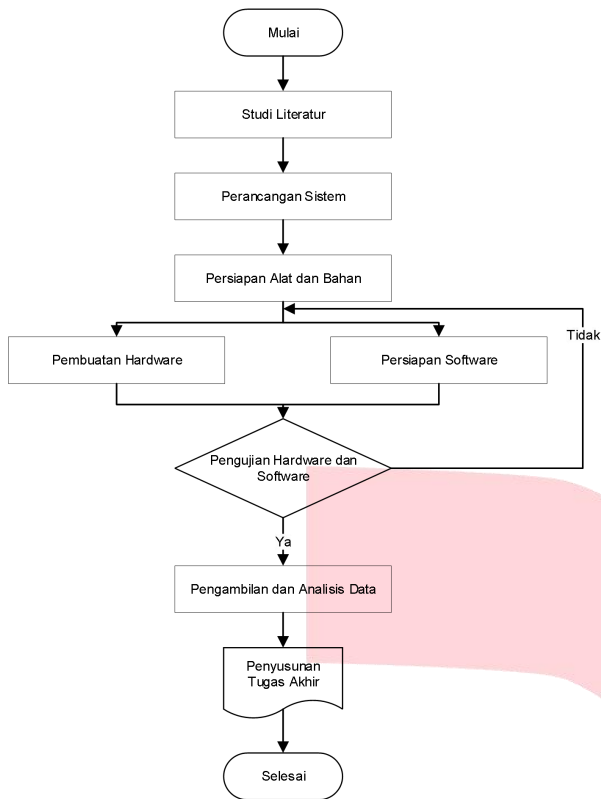
Dalam pemecahan masalah tersebut dibutuhkan suatu inovasi baru yang dapat diimplementasikan untuk memenuhi efisiensi kebutuhan pemantauan kesehatan terhadap pasien, inovasi tersebut adalah *Nurse Call* berbasis *Internet of Things (IoT)* Teknologi tersebut dapat menjadi solusi tanpa adanya penggunaan kabel karena yang digunakan berupa *wireless* yang memungkinkan pengawasan dilakukan dari jarak jauh melalui alat pemanggil perawat menggunakan tombol yang berupa seperti remot.

Wireless nurse call adalah sistem *nurse call* yang menggunakan teknologi nirkabel atau *wireless* untuk menghubungkan perangkat-perangkat dalam jaringan. Dalam sistem ini, perangkat-perangkat seperti tombol panggilan atau sensor yang terhubung dengan pasien dapat berkomunikasi dengan stasiun perawat atau pusat kontrol melalui sinyal nirkabel, biasanya menggunakan teknologi seperti WiFi, Bluetooth, atau teknologi nirkabel khusus lainnya.

Dalam *wireless nurse call*, saat pasien memicu perangkat seperti tombol panggilan atau sensor lainnya, sinyal nirkabel dikirimkan ke stasiun perawat, yang kemudian memberi peringatan kepada perawat atau staf medis yang bertanggung jawab. Ini memungkinkan respons yang cepat terhadap panggilan pasien atau situasi darurat.

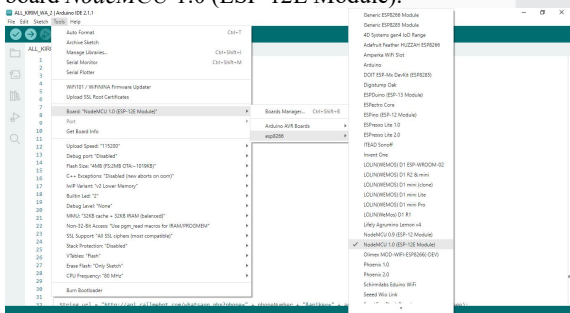
II. METODE

Studi Pustaka, Memahami serta mempelajari materi yang didapat dan diambil dari skripsi jurnal, ataupun tugas akhir, serta buku ilmiah yang berkaitan dengan penelitian sebelumnya yang dilakukan oleh penulis. Perancangan, Melakukan perancangan dan sistem yang akan digunakan sebelum diimplementasikan, pada perancangan ini dilakukan juga pemilihan komponen untuk menunjang pengimplementasian sistem. Implementasi Sistem, Dari perancangan yang telah dibuat akan diimplementasikan berupa prototipe dari sistem. Uji Coba Alat, Metode ini merupakan uji coba alat yang dilakukan pada tanaman untuk melihat apakah sistem yang telah diimplementasikan sudah berjalan dengan baik dan sensor yang diintegrasikan sudah bekerja. Analisa, Pada metode ini merupakan analisis sistem, hasil yang di dapat setelah melakukan uji coba alat tersebut untuk menentukan beroperasi atau tidaknya sistem tersebut yang sudah dibuat.



III. HASIL DAN PEMBAHASAN

1. Pemrograman *NodeMCU* dengan sensor dan koneksi wifi Untuk melakukan pemrograman pada *NodeMCU*, dapat dilakukan dengan menggunakan software Arduino IDE. Pada Arduino IDE, diperlukan pemilihan Board *NodeMCU* dengan benar, pada menu tools default belum ada board *NodeMCU* sehingga harus mengunduh terlebih dahulu library dari *NodeMCU* untuk memunculkan board *NodeMCU* pada menu tools, setelah mengunduh board *NodeMCU* selanjutnya memilih menu Tools -> Boards -> *ESP8266* -> *NodeMCU 1.0* (*ESP-12E Module*). Kemudian, mengklik Processor *NodeMCU*. Berikut adalah gambar pemilihan board untuk pemrograman board Arduino Mega 2560. Gambar Pemilihan board *NodeMCU 1.0* (*ESP-12E Module*).



Gambar 4. 1. Tampilan menu tools dan board pada Arduino IDE

```

19: else if (digitalRead(btn_tengah) == LOW) {
20:   Serial.println("Client.....");
21:   Serial.println("Send Command = ");
22:   if (btn_press == true)
23:     Commands = "LED_ON";
24:   Serial.println(Commands);
25:   send_commands();
26: }

```

Pemrograman *Client Node* yang dihubungkan dengan Server

Pemrograman *NodeMCU* yang dihubungkan ke server melalui Wifi yang berada dalam 1 jaringan dengan

menghubungkan ke 1 jaringan wifi dan memasukan IP

```

1: #include <ESP8266WiFi.h>
2:
3: const char* wifiName = "Amerta";
4: const char* wifiPass = "Mabunget";
5:
6: // the setup function runs once when you press reset or power the board
7: void setup() {
8:   Serial.begin(115200);
9:   delay(10);
10:  // we start by connecting to a WiFi network
11:  Serial.println();
12:  Serial.println("connecting to ");
13:  Serial.println(wifiName);
14:  WiFi.begin(wifiName, wifiPass);
15:  while (WiFi.status() != WL_CONNECTED) {
16:    delay(500);
17:    Serial.print(".");
18:  }
19:  Serial.println("");
20:  Serial.println("WiFi connected");
21:  Serial.println("IP address: ");
22:  Serial.println(WiFi.localIP()); //You can get IP address assigned to ESP
23: }

```

address ke software Arduino IDE. Koding untuk menghubungkan ke wifi dan IP address dijelaskan pada gambar 4.2

Gambar 4. 2. Tampilan Pemrograman Koneksi Wifi dan IP Address

```

const char * host = "192.168.183.152"; // IP Server
const int httpPort = 80;

Serial.println("");
Serial.println("Client.....");
Serial.println("connecting to network");
WiFi.mode(WIFI_STA); // Mode Station
WiFi.begin(ssid, password); // Matching the SSID and Password
delay(1000);

// Waiting to connect to wifi
while (WiFi.status() != WL_CONNECTED) {
  Serial.print(".");
  delay(500);
}
Serial.println("");
Serial.println("Successfully Connecting");
Serial.println("Status : Connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
Serial.println(".....");
Serial.println("");

```

Gambar 4. 3 Coding Input IP Address

Untuk menghubungkan ke server node harus memasukan IP address server node kedalam koding Arduino IDE.

Gambar 4. 4 Coding Menghubungkan dari Client ke Server

Setelah Wifi terkoneksi, untuk selanjutnya menghubungkan koneksi dari client ke server seperti pada gambar 4.4

Pemrograman Command Button Hijau

Pemrograman *NodeMCU* Button Hijau yaitu button yang ditengah pada rangkaian dilakukan untuk mengirim *command* ke server node agar server bisa menerima *command* untuk mengaktifkan output

```

1: loop() {
2:   // put your main code here, to run repeatedly:
3:   if (digitalRead(btn_tengah) == LOW) {
4:     Serial.println("Client.....");
5:     Serial.println("Send Command = ");
6:     if (btn_press == true)
7:       Commands = "LED_ON";
8:     Serial.println(Commands);
9:     send_commands();
10:  }

```

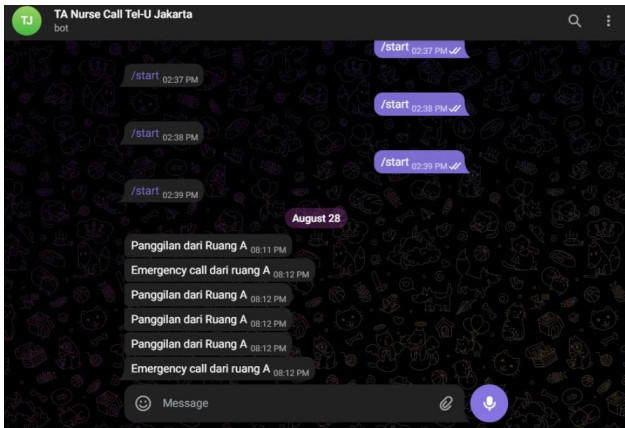
Gambar 4. 5. Tampilan Pemrograman Button Hijau Ketika Button Hijau ditekan akan mengirimkan *command* berupa notifikasi panggilan ke server berupa LED Hijau Menyala, LCD dan Telegram akan memunculkan pesan " Panggilan dari ruang A " serta buzzer akan berbunyi

Pemrograman Command Button Merah

Pemrograman *NodeMCU* Button Merah yaitu button yang berada paling kiri pada rangkaian dilakukan untuk mengirim *command* ke server node agar server bisa menerima *command* untuk mengaktifkan output

Gambar 4. 6. Tampilan Pemrograman Button Merah Output yang dihasilkan ketika Push button yang ditengah ditekan akan memberikan *command* berupa notifikasi emergency call ke server yang berupa LED Merah akan menyala, Buzzer akan

berbunyi, serta LCD dan telegram akan memunculkan pesan "Emergency Call dari Ruang A"



Gambar 4. 7 Tampilan Notifikasi pada Telegram

Pemrograman Command Reset Button

Pemrograman *NodeMCU* Reset Button yaitu button yang berada paling kanan pada rangkaian dilakukan untuk mematikan buzzer yang berbunyi tombol ini ditekan oleh perawat rumah sakit yang sedang bertugas.

```

70 else if (digitalRead(buttons) == LOW) {
71   Serial.println("Client.....");
72   Serial.print("Send Command : ");
73   if (data_press == true)
74     Commands = "LED_Off";
75   Serial.println(Commands);
76   send_commands();
77 }

```

Gambar 4. 8. Tampilan Pemrograman Reset Button

2. Pemrograman Mengirim Perintah ke Server

Untuk mengaktifkan output dari push button, di butuhkan command yang dikirim ke server, apabila command sudah berhasil terkirim maka server akan merespon dengan mengaktifkan output yang dihasilkan berupa suara dari buzzer serta notifikasi pada LCD dan lampu LED, berikut syntax mengirim perintah ke server.

```

void send_commands(){
  Serial.println("Sending command...");
  Serial.println("Don't press the button for now...");
  Serial.println("");
  Serial.print("Connecting to ");
  Serial.println(host);

  // Use WiFiClient class to create TCP connections
  WiFiClient client;

  if (!client.connect(host, httpPort)) {
    Serial.println("Connection failed");
    return;
  }

  // We now create a URI for the request
  Serial.print("Requesting URL : ");
  Serial.println(Commands);

  // This will send the request to the server
  client.print(String("GET ") + Commands + " HTTP/1.1\r\n" +
    "Host: " + host + "\r\n" + "Connection: Close\r\n\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println(">>> Client Timeout!");
      client.stop();
      return;
    }
  }
}

```

```

Serial.print("Server Reply = ");
// Read all the lines of the reply from server and print them to
// Serial
while(client.available()){
  String line = client.readStringUntil("\r");
  Serial.print(line);
}
Serial.println("Now you can press the button ...");
Serial.println("-----");
Serial.println("");

```

Ketika koding perintah server belum diaktifkan maka tombol belum bisa ditekan, apabila tetap ditekan dalam keadaan belum tersambung ke server maka output yang dihasilkan tidak akan keluar, karena itu dengan mengaktifkan command mengirim ke server maka, tombol bisa ditekan dan akan mengirimkan *feedback* dari server berupa suara dari buzzer serta notifikasi di LCD dan telegram.

3. Pemrograman *NodeMCU* dengan seluruh komponen

Pemrograman dimulai dari memasukkan hasil library yang telah diunduh untuk masing-masing komponen baik sensor, button, wifi, maupun perintah ke server. Dilakukan pengisian input pin pada masing-masing komponen yang memerlukan data baik pin analog maupun digital. Setelah itu penginputan tipe data pada masing-masing komponen untuk memanggil hasil pada akhir pemrograman.

Setelah itu diintegrasikan *NodeMCU* dengan *Wi-Fi* dengan memasukkan SSID dan password yang satu jaringan antara *client* dengan server agar *NodeMCU* dapat saling terhubung. Sebelum mengkoneksikan client dan server pada jaringan *Wi-Fi* dilakukan pengecekan pada masing masing IP, yang kemudian IP tersebut di input pada program Arduino.

Pemrograman untuk menghubungkan client dengan server dilakukan setelah pengecekan masing masing *NodeMCU* yang terkoneksi pada jaringan *Wi-Fi* yang sama, command push button di input kedalam program arduino IDE setelah client dengan server berhasil terhubung, Berikut merupakan gambar tampilan Arduino IDE dengan program lengkap yang dibuat oleh penulis.

```

1 #include <ESP8266WiFi.h>
2
3 const char* ssid = "Amarta"; // Your wifi name
4 const char* password = "MauBangeT"; // Your wifi Password
5
6 const char* host = "192.168.183.152"; // IP Server
7
8 const int httpPort = 80;
9
10 const char* Commands; // The command variable that is sent to the server
11
12 int button = D5; // push button is connected
13 int button2 = D4;
14 int button3 = D6;
15 bool btn_press = true; // The variable to detect the button has been pressed
16 int led = 5; // Variables for mode
17
18 void setup() {
19   // put your setup code here, to run once:
20   pinMode(button, INPUT_PULLUP); // initialize the pushbutton pin as an input:
21   pinMode(button2, INPUT_PULLUP);
22   pinMode(button3, INPUT_PULLUP);
23   Serial.begin(115200); // initialize serial:
24 }

```

Gambar 4. 9 Tampilan Pemrograman Seluruh Sistem 1

```

25 Serial.println("");
26 Serial.println("Client-----");
27 Serial.print("Connecting to Network");
28 WiFi.mode(WIFI_STA); // Mode Station
29 WiFi.begin(ssid, password); // Matching the SSID and Password
30 delay(1000);
31
32 // waiting to connect to wifi
33 while (WiFi.status() != WL_CONNECTED) {
34   Serial.print(".");
35   delay(500);
36 }
37 Serial.println("");
38 Serial.println("Successfully Connecting");
39 Serial.println("Status : Connected");
40 Serial.print("IP address: ");
41 Serial.println(WiFi.localIP());
42 Serial.println("-----");
43 Serial.println("");

```

Gambar 4. 10 Tampilan Pemrograman Seluruh Sistem 2


```

87 void loop() {
88 // put your main code here, to run repeatedly:
89 if (digitalRead(button) == LOW) {
90 Serial.println("Client-----");
91 Serial.println("Send Command = ");
92 if (btn_press == true)
93 Commands="LED_On";
94 Serial.println(Commands);
95 send_command();
96 }
97
98 else if (digitalRead(button) == LOW) {
99 Serial.println("Client-----");
100 Serial.println("Send Command = ");
101 if (btn_press == true)
102 Commands="LED_On?";
103 Serial.println(Commands);
104 send_command();
105 }
106
107 else if (digitalRead(button) == LOW) {
108 Serial.println("Client-----");
109 Serial.println("Send Command = ");
110 if (btn_press == true)
111 Commands="LED_Off";
112 Serial.println(Commands);
113 send_command();
114 }
115 }

```

Gambar 4. 11 Tampilan Pemrograman Seluruh Sistem 3

```

152 void send_command() {
153 Serial.println("Sending command...");
154 Serial.println("Don't press the button for now...");
155 Serial.println("");
156 Serial.println("Connecting to ");
157 Serial.println(host);
158
159 // Use WiFiClient class to create TCP connections
160 WiFiClient client;
161
162 if (!client.connect(host, httpPort)) {
163 Serial.println("Connection failed");
164 return;
165 }
166
167 // An http GET request
168 Serial.println("Requesting URL : ");
169 Serial.println(Commands);
170
171 // This will send the request to the server
172 client.print(String("GET ") + Commands + " HTTP/1.1\r\n" + "Host: " + host + "\r\n" + "Connection: close\r\n\r\n");
173 unsigned long timeout = millis();
174 while (client.available() == 0) {
175 if (millis() - timeout > 5000) {
176 Serial.println(">>> client timeout !");
177 client.stop();
178 return;
179 }
180 }

```

Gambar 4. 12 Tampilan Pemrograman Seluruh Sistem 4

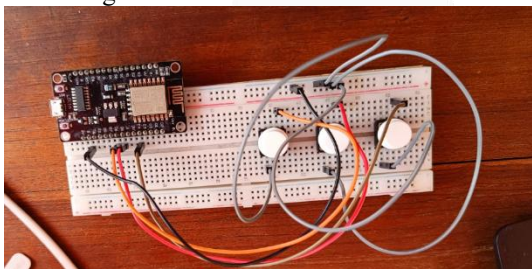
```

181 }
182
183 pinMode(buzzer, OUTPUT);
184 digitalWrite(buzzer, LOW);
185
186 pinMode(buzzer, OUTPUT);
187 digitalWrite(buzzer, HIGH);
188
189 digitalWrite(buzzer, LOW);
190
191 digitalWrite(buzzer, HIGH);
192
193 digitalWrite(buzzer, LOW);
194
195 digitalWrite(buzzer, HIGH);
196
197 digitalWrite(buzzer, LOW);
198
199 digitalWrite(buzzer, HIGH);
200
201 digitalWrite(buzzer, LOW);
202
203 digitalWrite(buzzer, HIGH);
204
205 digitalWrite(buzzer, LOW);
206
207 digitalWrite(buzzer, HIGH);
208
209 digitalWrite(buzzer, LOW);
210
211 digitalWrite(buzzer, HIGH);
212
213 digitalWrite(buzzer, LOW);
214
215 digitalWrite(buzzer, HIGH);
216
217 digitalWrite(buzzer, LOW);
218
219 digitalWrite(buzzer, HIGH);
220
221 digitalWrite(buzzer, LOW);
222
223 digitalWrite(buzzer, HIGH);
224
225 digitalWrite(buzzer, LOW);
226
227 digitalWrite(buzzer, HIGH);
228
229 digitalWrite(buzzer, LOW);
230
231 digitalWrite(buzzer, HIGH);
232
233 digitalWrite(buzzer, LOW);
234
235 digitalWrite(buzzer, HIGH);
236
237 digitalWrite(buzzer, LOW);
238
239 digitalWrite(buzzer, HIGH);
240
241 digitalWrite(buzzer, LOW);
242
243 digitalWrite(buzzer, HIGH);
244
245 digitalWrite(buzzer, LOW);
246
247 digitalWrite(buzzer, HIGH);
248
249 digitalWrite(buzzer, LOW);
250
251 digitalWrite(buzzer, HIGH);
252
253 digitalWrite(buzzer, LOW);
254
255 digitalWrite(buzzer, HIGH);
256
257 digitalWrite(buzzer, LOW);
258
259 digitalWrite(buzzer, HIGH);
260
261 digitalWrite(buzzer, LOW);
262
263 digitalWrite(buzzer, HIGH);
264
265 digitalWrite(buzzer, LOW);
266
267 digitalWrite(buzzer, HIGH);
268
269 digitalWrite(buzzer, LOW);
270
271 digitalWrite(buzzer, HIGH);
272
273 digitalWrite(buzzer, LOW);
274
275 digitalWrite(buzzer, HIGH);
276
277 digitalWrite(buzzer, LOW);
278
279 digitalWrite(buzzer, HIGH);
280
281 digitalWrite(buzzer, LOW);
282
283 digitalWrite(buzzer, HIGH);
284
285 digitalWrite(buzzer, LOW);
286
287 digitalWrite(buzzer, HIGH);
288
289 digitalWrite(buzzer, LOW);
290
291 digitalWrite(buzzer, HIGH);
292
293 digitalWrite(buzzer, LOW);
294
295 digitalWrite(buzzer, HIGH);
296
297 digitalWrite(buzzer, LOW);
298
299 digitalWrite(buzzer, HIGH);
300
301 digitalWrite(buzzer, LOW);
302
303 digitalWrite(buzzer, HIGH);
304
305 digitalWrite(buzzer, LOW);
306
307 digitalWrite(buzzer, HIGH);
308
309 digitalWrite(buzzer, LOW);
310
311 digitalWrite(buzzer, HIGH);
312
313 digitalWrite(buzzer, LOW);
314
315 digitalWrite(buzzer, HIGH);
316
317 digitalWrite(buzzer, LOW);
318
319 digitalWrite(buzzer, HIGH);
320
321 digitalWrite(buzzer, LOW);
322
323 digitalWrite(buzzer, HIGH);
324
325 digitalWrite(buzzer, LOW);
326
327 digitalWrite(buzzer, HIGH);
328
329 digitalWrite(buzzer, LOW);
330
331 digitalWrite(buzzer, HIGH);
332
333 digitalWrite(buzzer, LOW);
334
335 digitalWrite(buzzer, HIGH);
336
337 digitalWrite(buzzer, LOW);
338
339 digitalWrite(buzzer, HIGH);
340
341 digitalWrite(buzzer, LOW);
342
343 digitalWrite(buzzer, HIGH);
344
345 digitalWrite(buzzer, LOW);
346
347 digitalWrite(buzzer, HIGH);
348
349 digitalWrite(buzzer, LOW);
350
351 digitalWrite(buzzer, HIGH);
352
353 digitalWrite(buzzer, LOW);
354
355 digitalWrite(buzzer, HIGH);
356
357 digitalWrite(buzzer, LOW);
358
359 digitalWrite(buzzer, HIGH);
360
361 digitalWrite(buzzer, LOW);
362
363 digitalWrite(buzzer, HIGH);
364
365 digitalWrite(buzzer, LOW);
366
367 digitalWrite(buzzer, HIGH);
368
369 digitalWrite(buzzer, LOW);
370
371 digitalWrite(buzzer, HIGH);
372
373 digitalWrite(buzzer, LOW);
374
375 digitalWrite(buzzer, HIGH);
376
377 digitalWrite(buzzer, LOW);
378
379 digitalWrite(buzzer, HIGH);
380
381 digitalWrite(buzzer, LOW);
382
383 digitalWrite(buzzer, HIGH);
384
385 digitalWrite(buzzer, LOW);
386
387 digitalWrite(buzzer, HIGH);
388
389 digitalWrite(buzzer, LOW);
390
391 digitalWrite(buzzer, HIGH);
392
393 digitalWrite(buzzer, LOW);
394
395 digitalWrite(buzzer, HIGH);
396
397 digitalWrite(buzzer, LOW);
398
399 digitalWrite(buzzer, HIGH);
400
401 digitalWrite(buzzer, LOW);
402
403 digitalWrite(buzzer, HIGH);
404
405 digitalWrite(buzzer, LOW);
406
407 digitalWrite(buzzer, HIGH);
408
409 digitalWrite(buzzer, LOW);
410
411 digitalWrite(buzzer, HIGH);
412
413 digitalWrite(buzzer, LOW);
414
415 digitalWrite(buzzer, HIGH);
416
417 digitalWrite(buzzer, LOW);
418
419 digitalWrite(buzzer, HIGH);
420
421 digitalWrite(buzzer, LOW);
422
423 digitalWrite(buzzer, HIGH);
424
425 digitalWrite(buzzer, LOW);
426
427 digitalWrite(buzzer, HIGH);
428
429 digitalWrite(buzzer, LOW);
430
431 digitalWrite(buzzer, HIGH);
432
433 digitalWrite(buzzer, LOW);
434
435 digitalWrite(buzzer, HIGH);
436
437 digitalWrite(buzzer, LOW);
438
439 digitalWrite(buzzer, HIGH);
440
441 digitalWrite(buzzer, LOW);
442
443 digitalWrite(buzzer, HIGH);
444
445 digitalWrite(buzzer, LOW);
446
447 digitalWrite(buzzer, HIGH);
448
449 digitalWrite(buzzer, LOW);
450
451 digitalWrite(buzzer, HIGH);
452
453 digitalWrite(buzzer, LOW);
454
455 digitalWrite(buzzer, HIGH);
456
457 digitalWrite(buzzer, LOW);
458
459 digitalWrite(buzzer, HIGH);
460
461 digitalWrite(buzzer, LOW);
462
463 digitalWrite(buzzer, HIGH);
464
465 digitalWrite(buzzer, LOW);
466
467 digitalWrite(buzzer, HIGH);
468
469 digitalWrite(buzzer, LOW);
470
471 digitalWrite(buzzer, HIGH);
472
473 digitalWrite(buzzer, LOW);
474
475 digitalWrite(buzzer, HIGH);
476
477 digitalWrite(buzzer, LOW);
478
479 digitalWrite(buzzer, HIGH);
480
481 digitalWrite(buzzer, LOW);
482
483 digitalWrite(buzzer, HIGH);
484
485 digitalWrite(buzzer, LOW);
486
487 digitalWrite(buzzer, HIGH);
488
489 digitalWrite(buzzer, LOW);
490
491 digitalWrite(buzzer, HIGH);
492
493 digitalWrite(buzzer, LOW);
494
495 digitalWrite(buzzer, HIGH);
496
497 digitalWrite(buzzer, LOW);
498
499 digitalWrite(buzzer, HIGH);
500

```

Gambar 4. 13 Tampilan Pemrograman Seluruh Sistem 5

4. Tampilan Sistem yang telah diimplementasikan Implementasi Rangkaian Alat Client Node

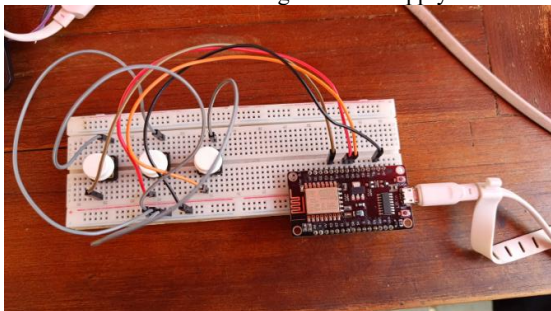


Proses Implementasi system dibuat dengan rancangan alat sesuai rencana yang dibuat, yang terdiri dari NodeMCU ESP8266 sebagai pusat pemroses, kemudian dirangkai dengan button serta kabel jumper yang dihubungkan pada breadboard.

Gambar 4. 14 Implementasi Rangkaian Client Node

Implementasi Pengujian Client node yang dihubungkan dengan server

Implementasi ini dilakukan untuk pengujian rangkaian alat apakah terhubung dengan server dimana perantara penghubungnya menggunakan kabel USB yang disambungkan dengan Baterai atau Power Bank sebagai Power Supply.



Gambar 4. 15 Pengujian Pada Client dengan Server Node

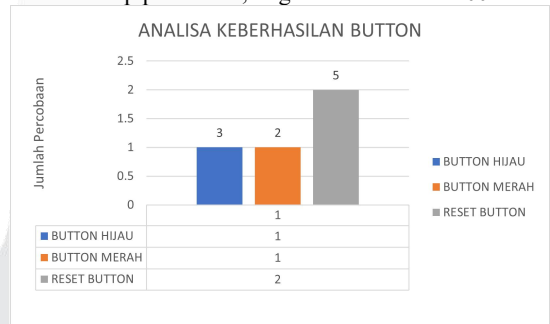
Saat Button ditekan dengan kondisi terhubung ke server maka akan mengaktifkan output yang berupa Sensor Suara dari Buzzer, sensor cahaya dari LCD serta notifikasi telegram.

Tabel 4. 1 Tabel Pengujian Client Node

Waktu	Jenis Button	Terminal Arduino IDE
12.05	Button Hijau	Command Sent
12.10	Reset Button	Command Sent
12.30	Button Merah	Command Sent
12.34	Reset Button	Command Sent
13.10	Button Merah	Command Sent
13.12	Reset Button	Command Sent
14.20	Button Hijau	Command Sent
14.24	Reset Button	Command Sent
15.30	Button Merah	Command Sent
15.33	Reset Button	Command Sent

5. Analisa Hasil Pengujian

Berdasarkan hasil pengujian pada masing masing button dalam rangkaian client node dalam keadaan terhubung ke server dengan jaringan wifi yang sama, masing masing tombol berfungsi dengan baik dari setiap percobaan, tingkat keberhasilan 100%.



Pada pengujian pertama diawali dengan button merah yang ditekan pada rangkaian client node dengan command sukses yang dilampirkan pada gambar dibawah ini

```

Client-----
Send Command = LED_On2
Sending command...
Don't press the button for now...

Connecting to 192.168.183.152
Requesting URL : LED_On2
Server Reply = GET LED Status : On HTTP/1.1
Host: 192.168.183.58
Connection: close

```

Gambar 4. 16 Command Sent Button Merah

Kemudian dilakukan percobaan kedua dengan menekan Push Button hijau yang ditekan pada rangkaian client node juga sukses terkirim seperti dijelaskan pada gambar 4.17

```

Client-----
Send Command = LED_On
Sending command...
Don't press the button for now...

Connecting to 192.168.183.152
Requesting URL : LED_On
Server Reply = GET LED Status : On HTTP/1.1
Host: 192.168.183.58
Connection: close

```

Gambar 4. 17 Command Sent Button Hijau

Setelah percobaan dari masing masing button, baik button merah dan button Hijau untuk mematikan command perintah ditekan push button pada reset button yang berada di sebelah kanan rangkaian client node, perintah dijelaskan dalam gambar 4.18

```

Client-----
Send Command = LED_Off
Sending command...
Don't press the button for now...

Connecting to 192.168.183.152
Requesting URL : LED_Off
Server Reply = GET LED Status : Off HTTP/1.1
Host: 192.168.183.58
Connection: close

```

Gambar 4. 18 Command Reset Button

IV. Kesimpulan

Pada proyek akhir ini mendapat kesimpulan yaitu : Perancangan alat pemanggil perawat yang memanfaatkan NodeMCU ESP8266 memudahkan pasien dalam memanggil bantuan perawat ketika dalam ruangan, alat ini berguna bagi pasien rawat inap, alat ini dirancang dengan wireless sehingga menghemat penggunaan kabel, alat ini juga mudah untuk dibawa kemana mana dan dapat digunakan sebagai remot panggilan perawat. Perancangan alat pemanggil perawat yang memanfaatkan

NodeMCU ESP8266 yang kemudian dihubungkan dengan jaringan *Wi-Fi* mampu mengirimkan notifikasi panggilan kepada server, agar ketika client menekan tombol, perawat dari server dapat segera mengatasi masalah dari client. Alat pemanggil perawat berbasis *wireless* yang dirangkai menggunakan NodeMCU lebih efisien dibanding nurse call berbasis kabel, dimana pada ruangan pasien tidak membutuhkan kabel yang banyak sehingga mengganggu kenyamanan pasien, selain itu alat ini dapat dibawa dan digenggam oleh pasien

V. REFERENSI

- [1] M FAHMI ERFAN, " SMART NURSE CALL BERBASIS MIKROKONTROLER ARDUINO UNTUK KOMUNIKASI ANTARA KAMAR PASIEN DENGAN PERAWAT MENGGUNAKAN ANDROID," Universitas Muhammadiyah Surabaya, 2020.
- [2] Ahmad Firdausi, " Perancangan Internet of Things Nurse Call System pada Area Rawat Inap Rumah Sakit Berbasis Arduino menggunakan Metode FIFO," Universitas Mercu Buana, Jakarta 2022.
- [3] S. N. Astriana Rahma Putri, "Perancangan Wireless Nurse Call Berbasis Mikrokontroler NodeMCU," Seminar Nasional Inovasi dan Aplikasi Teknologi di Industri 2019, pp. 150-154, 2019.
- [4] J. S. WAKUR, "ALAT PEMANGGIL PERAWAT MENGGUNAKAN ARDUINO UNO," POLITEKNIK NEGERI MANADO, MANADO, 2015.
- [5] FEBRIYANDA NURFAHRI, "RANCANG BANGUN NURSE CALL BERBASIS MIKROKONTROLLER UNTUK LANJUT USIA," Universitas Brawijaya, Malang, 2021.
- [6] Cahya Vikasari, " TEKNOLOGI APLIKASI NURSE CALL BERBASIS CLIENT SERVER PADA RUMAH SAKIT," Politeknik Negeri Cilacap, Cilacap, 2018.
- [7] M. G. Defriza, "RANCANG BANGUN SMART NURSE CALL BERBASIS ANDROID," Politeknik Negeri Malang, Malang, 2021.
- [8] A. R. RAMADAN, "SMART NURSE CALL BERBASIS MIKROKONTROLER ARDUINO UNTUK KOMUNIKASI ANTAR KAMAR PASIEN DENGAN MENGGUNAKAN ANDROID," UNIVERSITAS MUHAMMADIYAH SURABAYA, SURABAYA, 2020.
- [9] S. M. ARFAN, "RANCANG BANGUN NURSE CALL BERBASIS SMARTPHONE," UNIVERSITAS ISLAM NEGERI ALAUDIN MAKASAR, JAKARTA, 2022.