

**Nombre del proyecto:** Intercomunicador (Interphone)

**Semestre:** Quinto semestre

**Asignaturas que participan:** Tecnologías e Interfaces de Computadoras, Fundamentos de Bases de Datos, Redes de Computadoras.

**Nombre de los integrantes:**

José Antonio Tapia Cruz

Juan José Ortega García

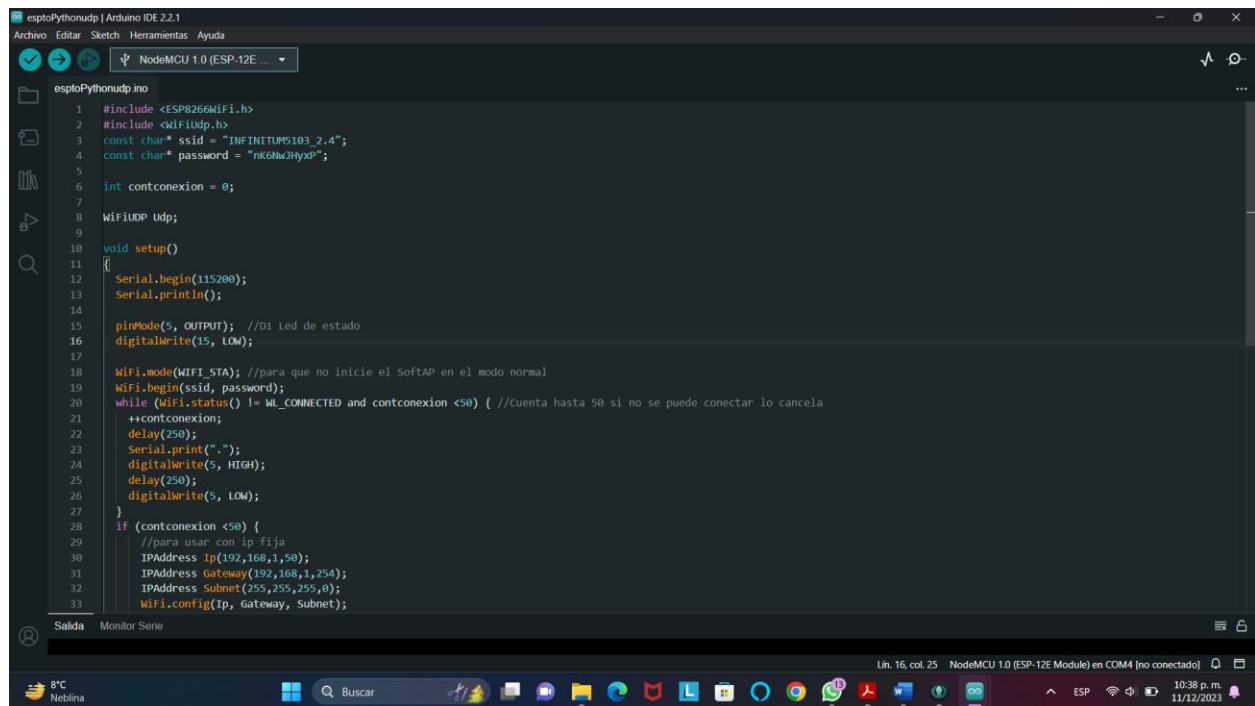
Néstor David Casas Casimiro

Alejandro Vera González

David Felipe Román

**Desarrollo de proyecto:**

Programación, estructuración y codificación del modulo ESP8266 desde el Arduino IDE

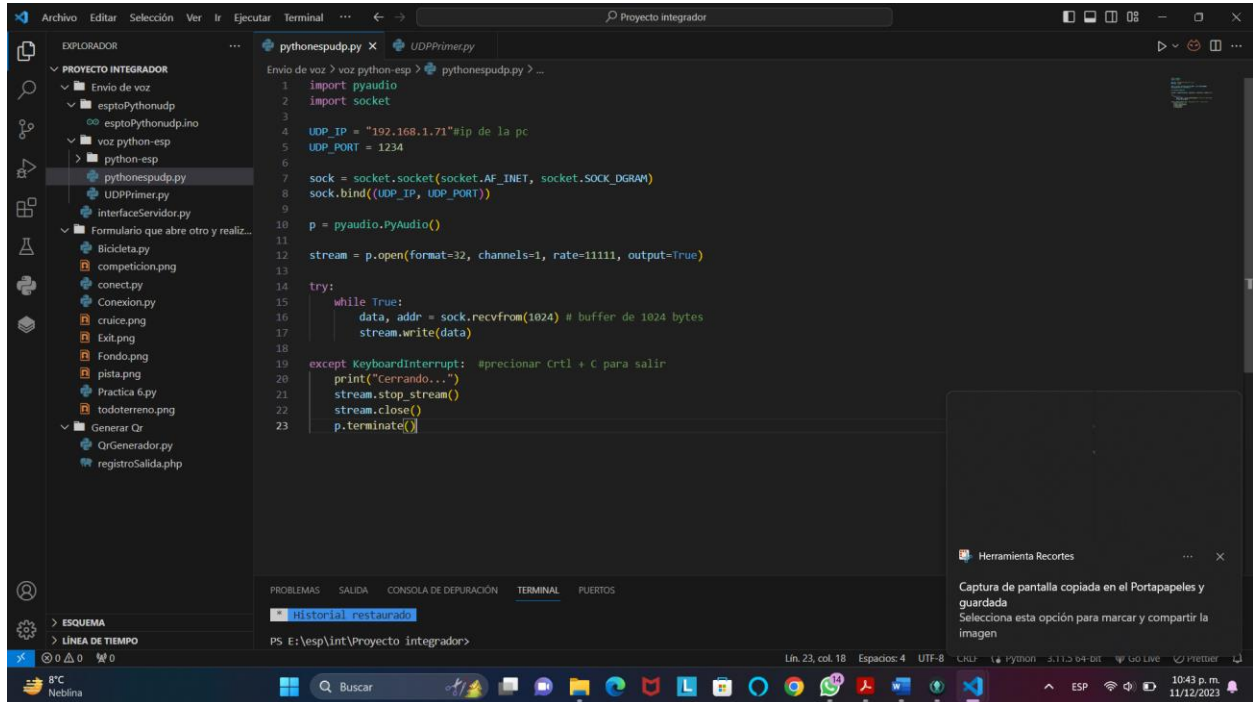


```
esptoPythonudp.ino
1 #include <ESP8266WiFi.h>
2 #include <WiFiUDP.h>
3 const char* ssid = "INFINITUM5103 2.4";
4 const char* password = "nk6NwJhyxP";
5
6 int contconexion = 0;
7
8 WiFiUDP udp;
9
10 void setup()
11 {
12   Serial.begin(115200);
13   Serial.println();
14
15   pinMode(5, OUTPUT); //D1 led de estado
16   digitalWrite(15, LOW);
17
18   WiFi.mode(WIFI_STA); //para que no inicie el SoftAP en el modo normal
19   WiFi.begin(ssid, password);
20   while (WiFi.status() != WL_CONNECTED and contconexion < 50) { //Cuenta hasta 50 si no se puede conectar lo cancela
21     ++contconexion;
22     delay(250);
23     Serial.print(".");
24     digitalWrite(5, HIGH);
25     delay(250);
26     digitalWrite(5, LOW);
27   }
28   if (contconexion < 50) {
29     //para usar con ip fija
30     IPAddress Ip(192,168,1,50);
31     IPAddress Gateway(192,168,1,254);
32     IPAddress Subnet(255,255,255,0);
33     WiFi.config(Ip, Gateway, Subnet);
```

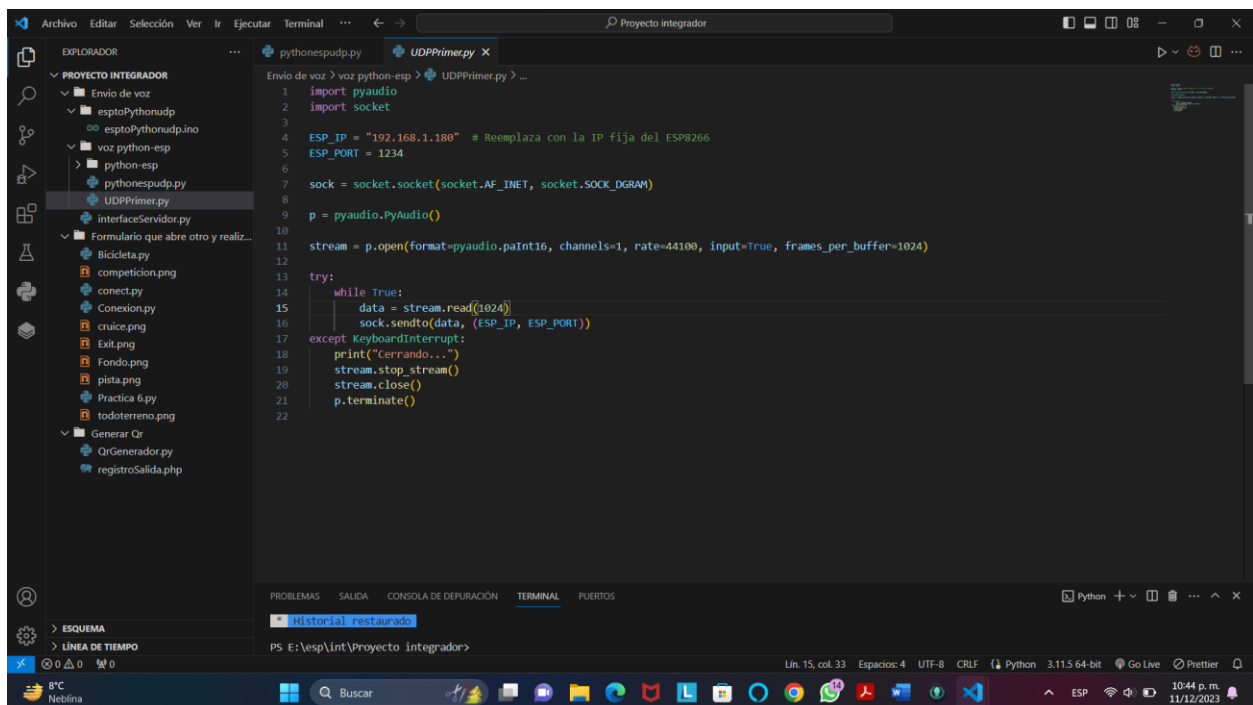
```
esptPythonudp | Arduino IDE 2.2.1
Archivo Editar Sketch Herramientas Ayuda
NodeMCU 1.0 (ESP-12E)
esptPythonudp.ino
14
15 pinMode(5, OUTPUT); //D1 Led de estado
16 digitalWrite(15, LOW);
17
18 WiFi.mode(WIFI_STA); //para que no inicie el SoftAP en el modo normal
19 WiFi.begin(ssid, password);
20 while (WiFi.status() != WL_CONNECTED and contconexion < 50) { //Cuenta hasta 50 si no se puede conectar lo cancela
21   ++contconexion;
22   delay(250);
23   Serial.print(",");
24   digitalWrite(5, HIGH);
25   delay(250);
26   digitalWrite(5, LOW);
27 }
28 if (contconexion < 50) {
29   //para usar con ip fija
30   IPAddress Ip(192,168,1,50);
31   IPAddress Gateway(192,168,1,254);
32   IPAddress Subnet(255,255,255,0);
33   WiFi.config(Ip, Gateway, Subnet);
34
35   Serial.println("");
36   Serial.println("WiFi conectado");
37   Serial.println(WiFi.localIP());
38   digitalWrite(5, HIGH);
39 }
40 else {
41   Serial.println("");
42   Serial.println("Error de conexion");
43   digitalWrite(15, LOW);
44 }
45 }
46
Salida Monitor Serie
Lin. 16, col. 25 NodeMCU 1.0 (ESP-12E Module) en COM4 [no conectado]
8°C Nebina
```

```
esptPythonudp | Arduino IDE 2.2.1
Archivo Editar Sketch Herramientas Ayuda
NodeMCU 1.0 (ESP-12E)
esptPythonudp.ino
37 Serial.println(WiFi.localIP());
38 digitalWrite(5, HIGH);
39 }
40 else {
41   Serial.println("");
42   Serial.println("Error de conexion");
43   digitalWrite(15, LOW);
44 }
45 }
46
47 void loop()
48 {
49   udp.beginPacket("192.168.1.71", 1234);
50
51   for(int i=0; i<1024;i++){
52     int old=micros();
53
54     float analog = analogRead(17);
55
56     analog = ((analog / 1) - 385);
57     if (analog > 255){
58       analog = 255;
59     }
60     else if (analog < 0){
61       analog = 0;
62     }
63
64     udp.write(int(analog));
65     while(micros()-old<87); // 90uSec = 1Sec/11111Hz - 3uSec para los otros procesos
66   }
67   udp.endPacket();
68   delay(5);
69 }
indexing: 73/86
Lin. 16, col. 25 NodeMCU 1.0 (ESP-12E Module) en COM4 [no conectado]
8°C Nebina
```

Codificación y programación de un simulador en Python que permite establecer comunicación con el modulo ESP8266 y recibir todos los paquetes UDP que este emite y poderlos decodificar para posteriormente ser procesados y reproducidos a modo de audio desde la interfaz.

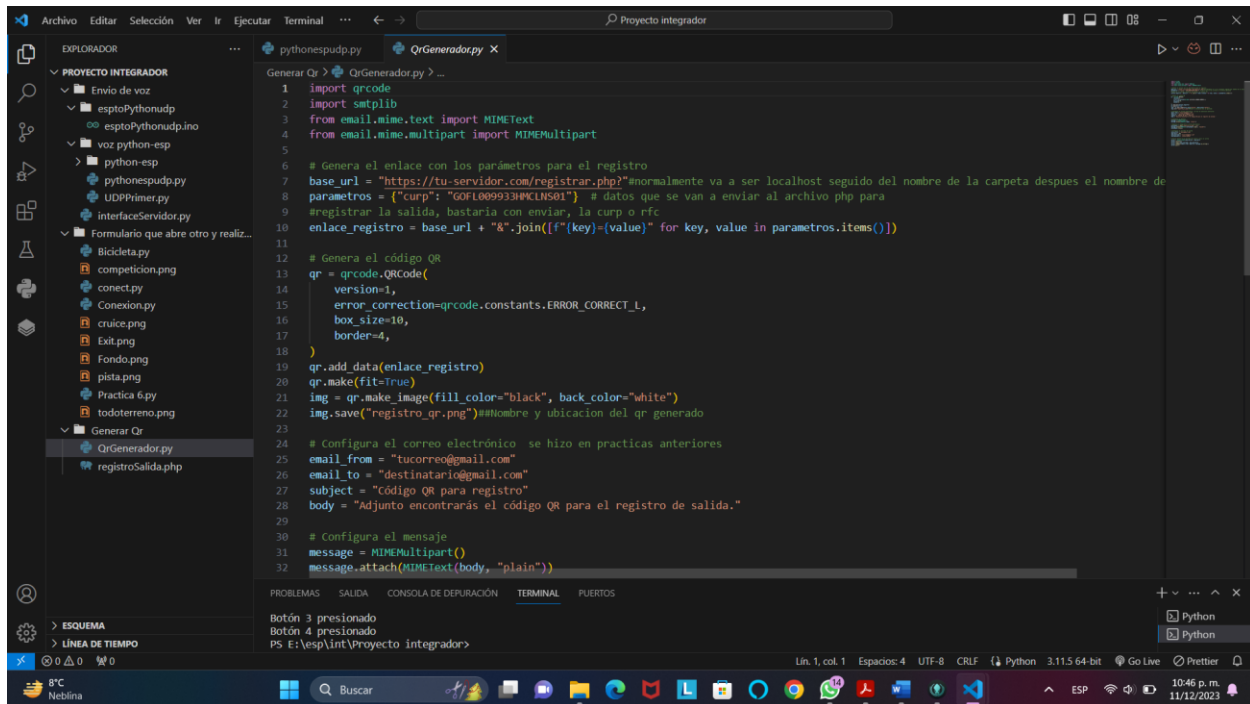


```
1 import pyaudio
2 import socket
3
4 UDP_IP = "192.168.1.71" # ip de la pc
5 UDP_PORT = 1234
6
7 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
8 sock.bind((UDP_IP, UDP_PORT))
9
10 p = pyaudio.PyAudio()
11
12 stream = p.open(format=32, channels=1, rate=11111, output=True)
13
14 try:
15     while True:
16         data, addr = sock.recvfrom(1024) # buffer de 1024 bytes
17         stream.write(data)
18
19 except KeyboardInterrupt: #precionar ctrl + C para salir
20     print("Cerrando...")
21     stream.stop_stream()
22     stream.close()
23     p.terminate()
```

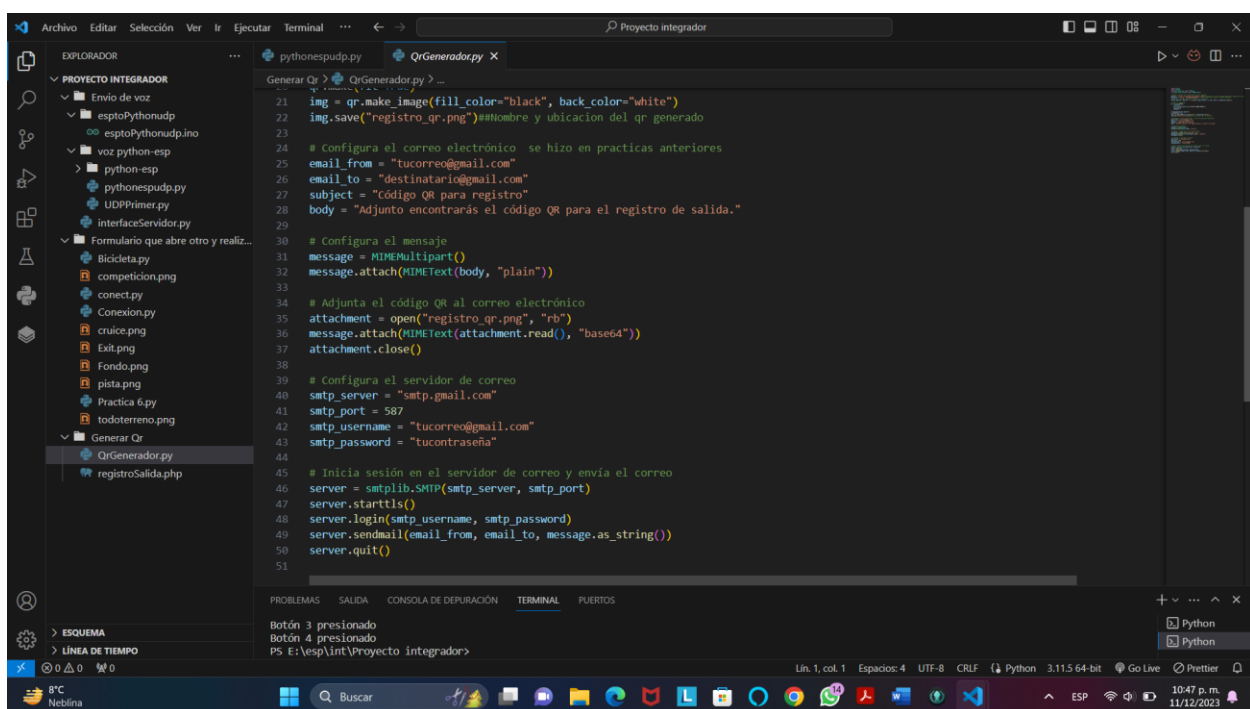


```
1 import pyaudio
2 import socket
3
4 ESP_IP = "192.168.1.180" # Reemplaza con la IP fija del ESP8266
5 ESP_PORT = 1234
6
7 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
8
9 p = pyaudio.PyAudio()
10
11 stream = p.open(format=pyaudio.paInt16, channels=1, rate=44100, input=True, frames_per_buffer=1024)
12
13 try:
14     while True:
15         data = stream.read(1024)
16         sock.sendto(data, (ESP_IP, ESP_PORT))
17
18 except KeyboardInterrupt:
19     print("Cerrando...")
20     stream.stop_stream()
21     stream.close()
22     p.terminate()
```

Programa que permite generar el código QR del registro de la persona visitante que desea tener acceso en la interfaz, además de mandarlo por correo adjunto.

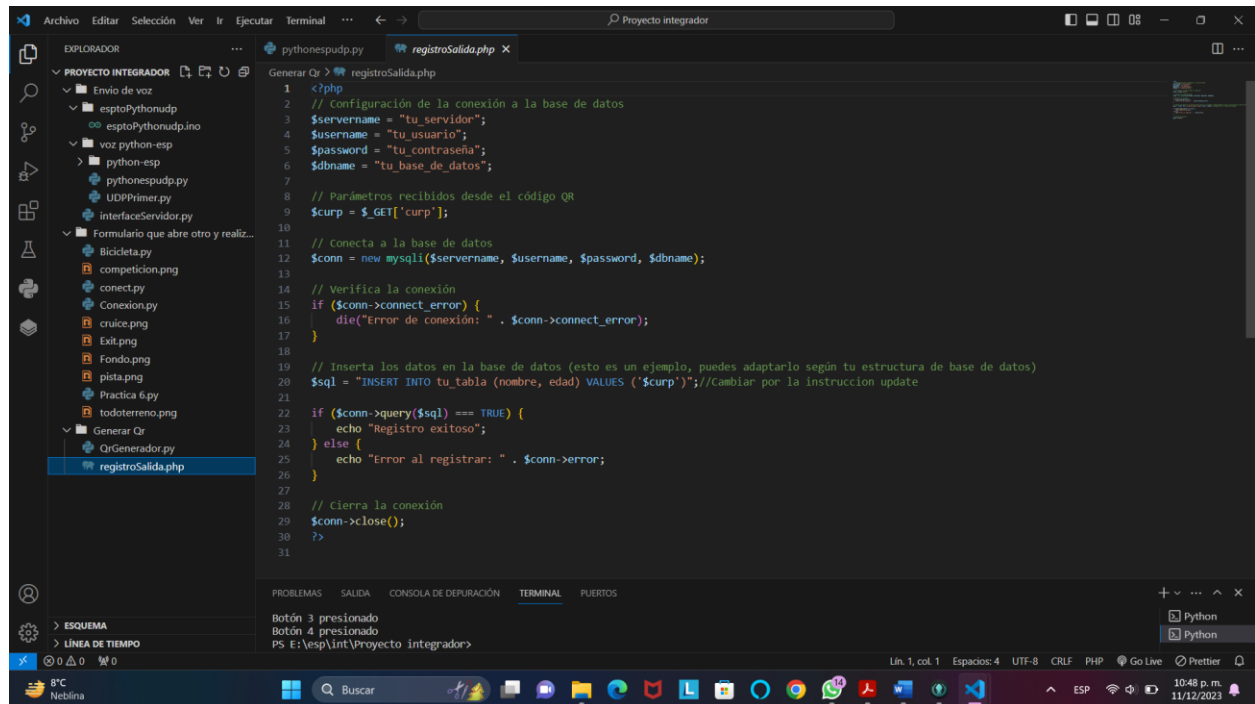


```
1 import qrcode
2 import smtplib
3 from email.mime.text import MIMEText
4 from email.mime.multipart import MIMEMultipart
5
6 # Genera el enlace con los parámetros para el registro
7 base_url = "https://tu-servidor.com/registro.php?" # normalmente va a ser localhost seguido del nombre de la carpeta despues el nombre de
8 parametros = {"curp": "GOF100993344CUN501"} # datos que se van a enviar al archivo php para
9 # registrar la salida, bastaria con enviar, la curp o rfc
10 enlace_registro = base_url + "&".join([f"{key}={value}" for key, value in parametros.items()])
11
12 # Genera el código QR
13 qr = qrcode.QRCode(
14     version=1,
15     error_correction=qrcode.constants.ERROR_CORRECT_L,
16     box_size=10,
17     border=4,
18 )
19 qr.add_data(enlace_registro)
20 qr.make(fit=True)
21 img = qr.make_image(fill_color="black", back_color="white")
22 img.save("registro_qr.png") # Nombre y ubicación del qr generado
23
24 # Configura el correo electrónico se hizo en practicas anteriores
25 email_from = "tucorreo@gmail.com"
26 email_to = "destinatario@gmail.com"
27 subject = "Código QR para registro"
28 body = "Adjunto encontrarás el código QR para el registro de salida."
29
30 # Configura el mensaje
31 message = MIMEMultipart()
32 message.attach(MIMEText(body, "plain"))
```



```
21 img = qr.make_image(fill_color="black", back_color="white")
22 img.save("registro_qr.png") # Nombre y ubicación del qr generado
23
24 # Configura el correo electrónico se hizo en practicas anteriores
25 email_from = "tucorreo@gmail.com"
26 email_to = "destinatario@gmail.com"
27 subject = "Código QR para registro"
28 body = "Adjunto encontrarás el código QR para el registro de salida."
29
30 # Configura el mensaje
31 message = MIMEMultipart()
32 message.attach(MIMEText(body, "plain"))
33
34 # Adjunta el código QR al correo electrónico
35 attachment = open("registro_qr.png", "rb")
36 message.attach(MIMEText(attachment.read(), "base64"))
37 attachment.close()
38
39 # Configura el servidor de correo
40 smtp_server = "smtp.gmail.com"
41 smtp_port = 587
42 smtp_username = "tucorreo@gmail.com"
43 smtp_password = "tucontraseña"
44
45 # Inicia sesión en el servidor de correo y envía el correo
46 server = smtplib.SMTP(smtp_server, smtp_port)
47 server.starttls()
48 server.login(smtp_username, smtp_password)
49 server.sendmail(email_from, email_to, message.as_string())
50 server.quit()
51
```

Programa que permite tener acceso a la base de datos, para posteriormente generar las consultas necesarias, agregar registros entre otros procesos.



The screenshot shows a code editor with a dark theme. On the left, a file explorer shows a project named 'PROYECTO INTEGRADOR' with various subfolders and files. The main editor area displays a PHP file named 'registroSalida.php'. The code is as follows:

```
1 <?php
2 // Configuración de la conexión a la base de datos
3 $servername = "tu_servidor";
4 $username = "tu_usuario";
5 $password = "tu_contraseña";
6 $dbname = "tu_base_de_datos";
7
8 // Parámetros recibidos desde el código QR
9 $curp = $_GET['curp'];
10
11 // Conecta a la base de datos
12 $conn = new mysqli($servername, $username, $password, $dbname);
13
14 // Verifica la conexión
15 if ($conn->connect_error) {
16     die("Error de conexión: " . $conn->connect_error);
17 }
18
19 // Inserta los datos en la base de datos (esto es un ejemplo, puedes adaptarlo según tu estructura de base de datos)
20 $sql = "INSERT INTO tu_tabla (nombre, edad) VALUES ('$curp')"; //Cambiar por la instrucción update
21
22 if ($conn->query($sql) === TRUE) {
23     echo "Registro exitoso";
24 } else {
25     echo "Error al registrar: " . $conn->error;
26 }
27
28 // Cierra la conexión
29 $conn->close();
30 ?>
31
```

At the bottom, a terminal window shows the command prompt with the path 'E:\esp\Int\Proyecto Integrador>'.

Link de acceso al repositorio en GitHub: <https://github.com/JoseTapia05/PROYECTO-INTEGRADOR/tree/master>