



Automatización y control de bebederos automáticos para especies animales usando LoRa:

*Documentación de la página
web*

31 de agosto de 2021

TRABAJO FIN DE GRADO

Grado en Ingeniería en Sistemas de
Telecomunicaciones

Autora: Lucía Francoso Fernández

Tutor: Juan Pascual García

Índice

Índice de figuras	3
Índice de tablas	4
1 Introducción	5
2 Apariencia de la web	5
Anexo	12

Índice de figuras

1	Captura de la web <i>Open Pet Feeder</i> , donde se muestra el mensaje de error al no detectar la conexión serial con el arduino.	5
2	Captura de la web <i>Open Pet Feeder</i> , donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando <i>ping</i> (botón <i>Comprobar actividad</i>).	6
3	Captura de la web <i>Open Pet Feeder</i> , donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando <i>status</i> (botón <i>Comprobar estado</i>).	7
4	Captura de la web <i>Open Pet Feeder</i> , donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando <i>check_levels</i> (botón <i>Comprobar niveles</i>).	8
5	Captura de la web <i>Open Pet Feeder</i> , donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando <i>rellenar_comedero</i> (botón <i>Rellenar comedero</i>).	9
6	Captura de la web <i>Open Pet Feeder</i> , donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando <i>reset_comedero</i> (botón <i>Reiniciar comedero</i>).	10
7	Captura de la web <i>Open Pet Feeder</i> , donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando <i>rellenar_bebedero</i> (botón <i>Rellenar bebedero</i>).	11

Índice de tablas

1 Introducción

En este documento se recoge una pequeña documentación sobre la página web *Open Pet Feeder*, una web creada bajo el marco del Trabajo Fin de Grado especificado en el título del presente, *Automatización y control de bebederos automáticos para especies animales usando LoRa*. Es una parte fundamental para la consecución de ciertos objetivos del proyecto, entre ellos, la monitorización de la estación que se situará en el refugio de Patitas Unidas Los Alcázares, desde cualquier dispositivo y desde cualquier sitio (dentro o fuera de la red local de la estación donde se sitúa el servidor que almacena la web).

Comprende explicaciones sobre el funcionamiento de la misma, así como explicaciones sobre la creación de la web. Para una introducción a la página web más visual, se incluyen capturas bajo diferentes situaciones.

2 Apariencia de la web

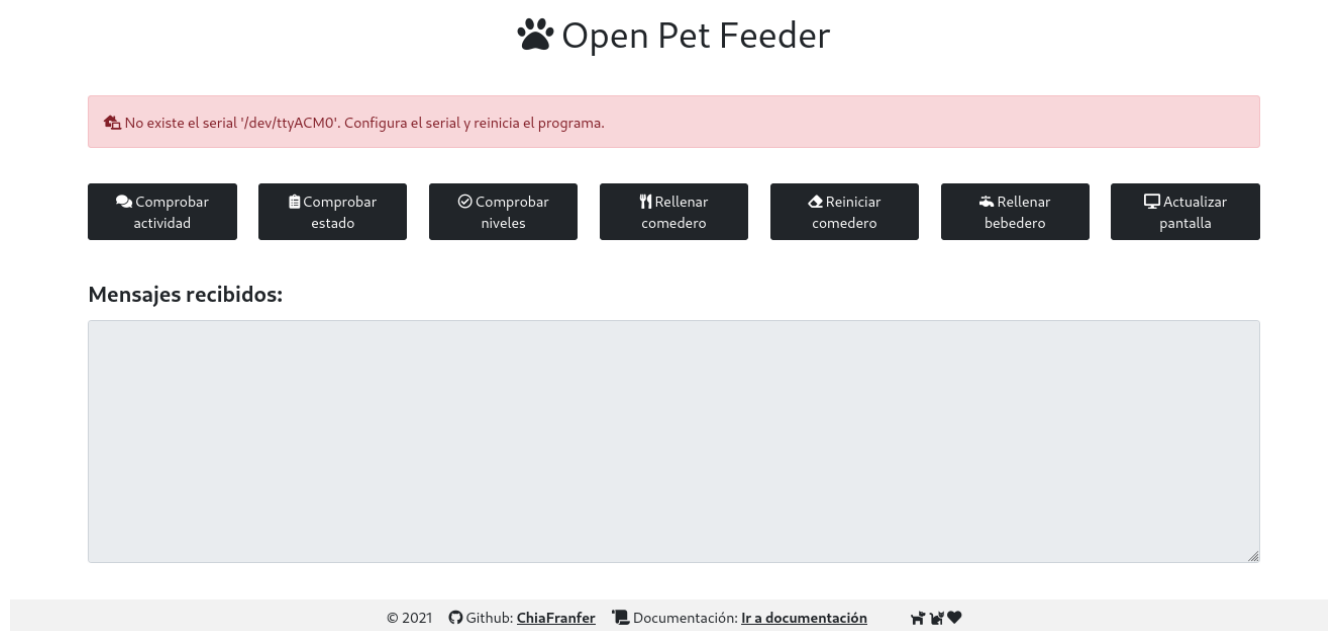





Figura 1: Captura de la web *Open Pet Feeder*, donde se muestra el mensaje de error al no detectar la conexión serial con el arduino.

Open Pet Feeder

 Conectado a GW utilizando el serial: /dev/ttyUSB0!

 Comprobar actividad


 Comprobar estado

 Comprobar niveles

 Rellenar comedero

 Reiniciar comedero

 Rellenar bebedero

 Actualizar pantalla

Mensajes recibidos:

```
21:03:42 >> Comando a gateway: ping
21:03:44 >> Respuesta: [GW] Comando recibido: ping
[GW] Enviando por LoRa: ping
21:03:44 >> Respuesta: [GW] Msg enviado. ID: 9
21:03:45 >> Respuesta: [Refugio] ID: 10
[Refugio] Msg: ack;pong
```


© 2021  Github: [ChiaFranfer](#)  Documentación: [Ir a documentación](#)


 


Figura 2: Captura de la web *Open Pet Feeder*, donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando *ping* (botón *Comprobar actividad*).

Open Pet Feeder

 Conectado a GW utilizando el serial: /dev/ttyUSB0!

 Comprobar actividad


 Comprobar estado

 Comprobar niveles

 Rellenar comedero

 Reiniciar comedero

 Rellenar bebedero

 Actualizar pantalla

Mensajes recibidos:


```
21:05:21 >> Comando a gateway: status
21:05:23 >> Respuesta: [GW] Comando recibido: status
[GW] Enviando por LoRa: status
21:05:23 >> Respuesta: [GW] Msg enviado. ID: 11
21:05:24 >> Respuesta: [Refugio] ID: 12
[Refugio] Msg: ack;status;ok
```


© 2021  Github: [ChiaFranfer](#)  Documentación: [Ir a documentación](#)





Figura 3: Captura de la web *Open Pet Feeder*, donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando *status* (botón *Comprobar estado*).

Open Pet Feeder

 Conectado a GW utilizando el serial: /dev/ttyUSB0!

 Comprobar actividad


 Comprobar estado

 Comprobar niveles

 Rellenar comedero

 Reiniciar comedero

 Rellenar bebedero

 Actualizar pantalla

Mensajes recibidos:


```
21:07:00 >> Comando a gateway: check_levels
21:07:01 >> Respuesta: [GW] Comando recibido: check_lev
21:07:01 >> Respuesta: els
[GW] Enviando por LoRa: check_levels
21:07:02 >> Respuesta: [GW] Msg enviado. ID: 13
21:07:03 >> Respuesta: [Refugio] ID: 14
[Refugio] Msg: ack;water;100
```


© 2021  Github: [ChiaFranfer](#)  Documentación: [Ir a documentación](#)




Figura 4: Captura de la web *Open Pet Feeder*, donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando *check_levels* (botón *Comprobar niveles*).

Open Pet Feeder

 Conectado a GW utilizando el serial: /dev/ttyUSB0!

 Comprobar actividad


 Comprobar estado

 Comprobar niveles

 Rellenar comedero

 Reiniciar comedero

 Rellenar bebedero

 Actualizar pantalla

Mensajes recibidos:

```
21:10:24 >> Comando a gateway: rellenar_comedero
21:10:25 >> Respuesta: [GW] Comando recibido: rellenar_comedero
[GW] Enviando por LoRa: rellenar_comedero
21:10:26 >> Respuesta: [GW] Msg enviado. ID: 21
21:10:43 >> Respuesta: [Refugio] ID: 22
[Refugio] Msg: ack;feeder_refill;ok;num_refill;1
21:10:54 >> Comando a gateway: rellenar_comedero
21:10:55 >> Respuesta: : rellenar_comedero
21:10:55 >> Respuesta: [GW] Msg enviado. ID: 23
21:11:13 >> Respuesta: [Refugio] ID: 24
[Refugio] Msg: ack;feeder_refill;ok;num_refill;2
```


© 2021  Github: [ChiaFranfer](#)  Documentación: [Ir a documentación](#)




Figura 5: Captura de la web *Open Pet Feeder*, donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando *rellenar_comedero* (botón *Rellenar comedero*).

Open Pet Feeder

 Conectado a GW utilizando el serial: /dev/ttyUSB0!

 Comprobar actividad


 Comprobar estado

 Comprobar niveles

 Rellenar comedero

 Reiniciar comedero

 Rellenar bebedero

 Actualizar pantalla

Mensajes recibidos:


```
21:12:05 >> Comando a gateway: reset_comedero
21:12:06 >> Respuesta: [GW] Comando recibido: reset_comedero
[GW] Enviando por LoRa: reset_comedero
21:12:06 >> Respuesta: [GW] Msg enviado. ID: 25
21:12:08 >> Respuesta: [Refugio] ID: 26
[Refugio] Msg: ack;reset_comedero
21:12:13 >> Comando a gateway: rellenar_comedero
21:12:14 >> Respuesta: [GW] Comando recibido: rellenar_comedero
[GW] Enviando por LoRa: rellenar_comedero
21:12:15 >> Respuesta: [GW] Msg enviado. ID: 27
21:12:32 >> Respuesta: [Refugio] ID: 28
[Refugio] Msg: ack;feeder_refill;ok;num_refill;1
```


© 2021  Github: [ChiaFranfer](#)  Documentación: [Ir a documentación](#)





Figura 6: Captura de la web *Open Pet Feeder*, donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando *reset_comedero* (botón *Reiniciar comedero*).

Open Pet Feeder

 Conectado a GW utilizando el serial: /dev/ttyUSB0!

 Comprobar actividad


 Comprobar estado

 Comprobar niveles

 Rellenar comedero

 Reiniciar comedero

 Rellenar bebedero

 Actualizar pantalla

Mensajes recibidos:

```
21:14:05 >> Comando a gateway: rellenar_bebedero
21:14:06 >> Respuesta: [GW] Comando recibido: rellenar_bebedero
[GW] Enviando por LoRa: rellenar_bebedero
21:14:06 >> Respuesta: [GW] Msg enviado. ID: 29
21:14:13 >> Respuesta: [Refugio] ID: 30
[Refugio] Msg: ack;water_refill;ok
```

© 2021  Github: [ChiaFranfer](#)  Documentación: [Ir a documentación](#)



Figura 7: Captura de la web *Open Pet Feeder*, donde se muestra en la consola la interacción entre estación del campo y estación del refugio a través del comando *rellenar_bebedero* (botón *Rellenar bebedero*).

El botón *Actualizar pantalla* (comando asociado *update_oled*) no realiza la función que describe su nombre, ya que no se implementó a nivel arduino por ocupar más espacio del disponible en el arduino nano (la librería *OLED* ocupa bastante espacio). Si se envía este comando, la estación del refugio devuelve un string con “*to do*”, indicando que hay que implementarlo.

Anexo

En este anexo, se incluye el código empleado para crear la web.

Aplicación web: app.py

```
1 import engineio
2 from flask import Flask, render_template
3 from flask_socketio import SocketIO
4 from flask_serial import Serial
5 from datetime import datetime
6
7 import json
8 import socketio
9 import os
10
11 app = Flask(__name__)
12
13 ## Configuration Serial GW
14 serial_port = '/dev/ttyUSB0' # Modificar aqui el serial del
    Arduino, lo dice el IDE
15 app.config['SERIAL_PORT'] = serial_port
16 app.config['SERIAL_TIMEOUT'] = 0.1
17 app.config['SERIAL_BAUDRATE'] = 115200
18 app.config['SERIAL_BYTESIZE'] = 8
19 app.config['SERIAL_PARITY'] = 'N'
20 app.config['SERIAL_STOPBITS'] = 1
21
22 socketio = SocketIO(app, async_mode="gevent", ping_timeout=5, logger=False,
    engineio_logger=False)
23 ser = Serial(app)
24
25 #
26 #   Enpoints
27 #
28
29 # Endpoint home page
30 @app.route("/")
31 def index():
32     # Comprobamos que el serial existe
33     if os.path.exists(serial_port):
34         serial_exists = True
35     else:
36         serial_exists = False
37
38     return render_template("index.html", serial_port=serial_port, serial_exists=
        serial_exists)
39
40 # Endpoint documentation page
41 @app.route("/doc")
42 def doc():
43     return render_template("doc.html")
```

```

44
45 #
46 #   SocketIO
47 #
48
49 # Handler SocketIO [messages of web]
50 @socketio.on('send')
51 def handle_send(json_str):
52     data = json.loads(json_str) ["message"]
53     print("Boton pulsado: {}".format(data))
54
55 # Enviamos por serial
56 try:
57     ser.on_send(data)
58 except Exception as ex:
59     print("Error on serial: {}".format(ex))
60
61 # Enviamos en el socket 'receive_message' [hacer en recepcion serial]
62 now = datetime.now()
63 msg = "{} >> Comando a gateway: {}".format(now.strftime("%H:%M:%S"), data)
64 socketio.emit("receive_message", data={"message":str(msg)})
65
66 #
67 #   Flask-Serial
68 #
69
70 # Handler incoming serial messages
71 @ser.on_message()
72 def handle_message(msg):
73     try:
74         print("receive message: {}".format(msg))
75         now = datetime.now()
76         msg = "{} >> Respuesta: {}".format(now.strftime("%H:%M:%S"), msg.decode("utf-8").
            strip('\n'))
77         print("Send socket msg: {}".format(msg))
78         socketio.emit("receive_message", data={"message":str(msg)})
79     except Exception as ex:
80         print(ex)
81
82 # Log thread serial
83 @ser.on_log()
84 def handle_logging(level, info):
85     print(level, info)
86
87 if __name__ == '__main__':
88     socketio.run(app, debug=False, log_output=True)
89

```

Página principal de la web: index.html

```

1 <!doctype html>
2 <html lang="es">
3 <head>

```

```

4 <!-- Required meta tags -->
5 <meta charset="utf-8">
6 <meta name="viewport" content="width=device-width, initial-scale=1">
7 <link rel="icon" type="image/png" href="static/favicon.ico">
8
9 <!-- Bootstrap CSS -->
10 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css
    " rel="stylesheet" integrity="sha384-EVSTQN3/
    azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWfFspd3yD65VohhpuuCOMLASjC" crossorigin="anonymous
    ">
11 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome
    /5.15.3/css/all.min.css" integrity="sha512-iBBXm8fW90+nuLcSKlbnrPcLa00T92x01BIsZ
    +ywDWZCvqsWgccV3gFoRBv0z+8dLJgyAHlR35VZc2oM/gIlw==" crossorigin="anonymous"
    referrerpolicy="no-referrer" />
12 <script src="https://cdn.socket.io/3.1.3/socket.io.min.js" integrity="sha384-
    cPwlPLvBTa3sKAgddT6krw0cJat7egBga3DJepJyrLl4Q9/5WLra3rrnMcyTyOnh" crossorigin="
    anonymous"></script>
13 <script src="https://code.jquery.com/jquery-3.6.0.min.js" integrity="sha256-/xUj+3
    OJU5yExlq6GSYGSk7tPXikynS7ogEvDej/m4=" crossorigin="anonymous"></script>
14
15 <title>Open Pet Feeder</title>
16 </head>
17 <body>
18 <h1 style="text-align: center; margin-top: 2%;"><a href="/" style="text-decoration:
    none; color: inherit;"><i class="fas fa-paw"></i> Open Pet Feeder</a></h1>
19
20
21 <div class="container" style="margin-top: 3%;">
22 {% if serial_exists == True %}
23 <div class="alert alert-success" role="alert" style="margin-bottom: 3%;">
24 <i class="fas fa-laptop-house"></i> Conectado a GW utilizando el serial: {{
    serial_port}}!
25 </div>
26 {% else %}
27 <div class="alert alert-danger" role="alert" style="margin-bottom: 3%;">
28 <i class="fas fa-laptop-house"></i> No existe el serial '{{serial_port}}'.
    Configura el serial y reinicia el programa.
29 </div>
30 {% endif %}
31
32 <div class="row" style="text-align: center;">
33 <div class="col">
34 <button type="button" class="btn btn-dark mt-1" id="send_ping" value="ping"><
    i class="fas fa-comments"></i> Comprobar actividad</button>
35 </div>
36 <div class="col">
37 <button type="button" class="btn btn-dark mt-1" id="send_status" value="
    status"><i class="fas fa-clipboard-list"></i> Comprobar estado</button>
38 </div>
39 <div class="col">
40 <button type="button" class="btn btn-dark mt-1" id="send_check_levels" value=
    "check_levels"> <i class="fas fa-check-circle"></i> Comprobar niveles</button>
41 </div>
42 <div class="col">

```

```

43     <button type="button" class="btn btn-dark mt-1" id="send_comedero" value="
rellenar_comedero"><i class="fas fa-utensils"></i> Rellenar comedero</button>
44 </div>
45 <div class="col">
46 <button type="button" class="btn btn-dark mt-1" id="send_reset_comedero"
value="reset_comedero"><i class="fas fa-eraser"></i> Reiniciar comedero</button>
47 </div>
48 <div class="col">
49 <button type="button" class="btn btn-dark mt-1" id="send_bebedero" value="
rellenar_bebedero"><i class="fas fa-faucet"></i> Rellenar bebedero</button>
50 </div>
51
52 <div class="col">
53 <button type="button" class="btn btn-dark mt-1" id="send_oled" value="
update_oled"><i class="fas fa-desktop"></i> Actualizar pantalla</button>
54 </div>
55 </div>
56 </div>
57
58 <div class="container" style="margin-top: 3%;">
59 <div class="console">
60 <label class="form-label" for="receive"> <h4> <b>Mensajes recibidos:</b> </h4>
> </label>
61 <textarea readonly class="form-control" name="receive" id="receive" rows="8"
style="width: 100%; height: auto;"></textarea>
62 </div>
63 </div>
64
65 <!-- Footer -->
66 <div class="footer fixed-bottom mt-4 py-4">
67 <div class="text-center p-2" style="background-color: rgba(0, 0, 0, 0.05);
margin-bottom: -1%;">
68 C 2021 <i class="fab fa-github" style="margin-left: 1%;"></i> Github:
69 <a class="text-reset fw-bold" href="https://github.com/ChiaFranfer">
ChiaFranfer</a>
70 <i class="fas fa-scroll" style="margin-left: 1%;"></i> Documentacion: <a
class="text-reset fw-bold" href="{url_for('doc')}"> Ir a documentacion</a>
71 <i class="fas fa-dog" style="margin-left: 3%;"></i> <i class="fas fa-cat"></i>
> <i class="fas fa-heart"></i>
72 </div>
73 </div>
74
75 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.
bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8Nl+
NtUVF0sA7MsXsPlUyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM" crossorigin="anonymous"></script>
>
76 <script type="text/javascript" charset="utf-8">
77 $(document).ready(function() {
78     var socket = io.connect('http://' + document.domain + ':' + location.port);
79
80     $('#send_status').on('click', function() {
81         var message = $(this)[0].value;
82         console.log('Enviando: ' + message);
83         var data = '{ ' + '"message": ' + message + ' }';

```

```

84     socket.emit('send', data=data);
85 });
86
87 $('#send_ping').on('click', function() {
88     var message = $(this)[0].value;
89     console.log('Enviando: ' + message);
90     var data = '{' + '"message": "' + message + '"}';
91     socket.emit('send', data=data);
92 });
93
94 $('#send_reset_comedero').on('click', function() {
95     var message = $(this)[0].value;
96     console.log('Enviando: ' + message);
97     var data = '{' + '"message": "' + message + '"}';
98     socket.emit('send', data=data);
99 });
100
101 $('#send_comedero').on('click', function() {
102     var message = $(this)[0].value;
103     console.log('Enviando: ' + message);
104     var data = '{' + '"message": "' + message + '"}';
105     socket.emit('send', data=data);
106 });
107
108 $('#send_bebedero').on('click', function() {
109     var message = $(this)[0].value;
110     console.log('Enviando: ' + message);
111     var data = '{' + '"message": "' + message + '"}';
112     socket.emit('send', data=data);
113 });
114
115 $('#send_check_levels').on('click', function() {
116     var message = $(this)[0].value;
117     console.log('Enviando: ' + message);
118     var data = '{' + '"message": "' + message + '"}';
119     socket.emit('send', data=data);
120 });
121
122 $('#send_oled').on('click', function() {
123     var message = $(this)[0].value;
124     console.log('Enviando: ' + message);
125     var data = '{' + '"message": "' + message + '"}';
126     socket.emit('send', data=data);
127 });
128
129 socket.on('receive_message', function(data){
130     console.log(data);
131     var text = data['message'];
132     var $textarea = $('#receive');
133     $textarea.val($textarea.val() + text + '\n');
134     $textarea.scrollTop($textarea[0].scrollHeight);
135 })
136 });
137

```



```
138     </script>
139     </body>
140     </html>
141
```