

Foro colaborativo – Modulo 2 – Lección 8

Integrantes:

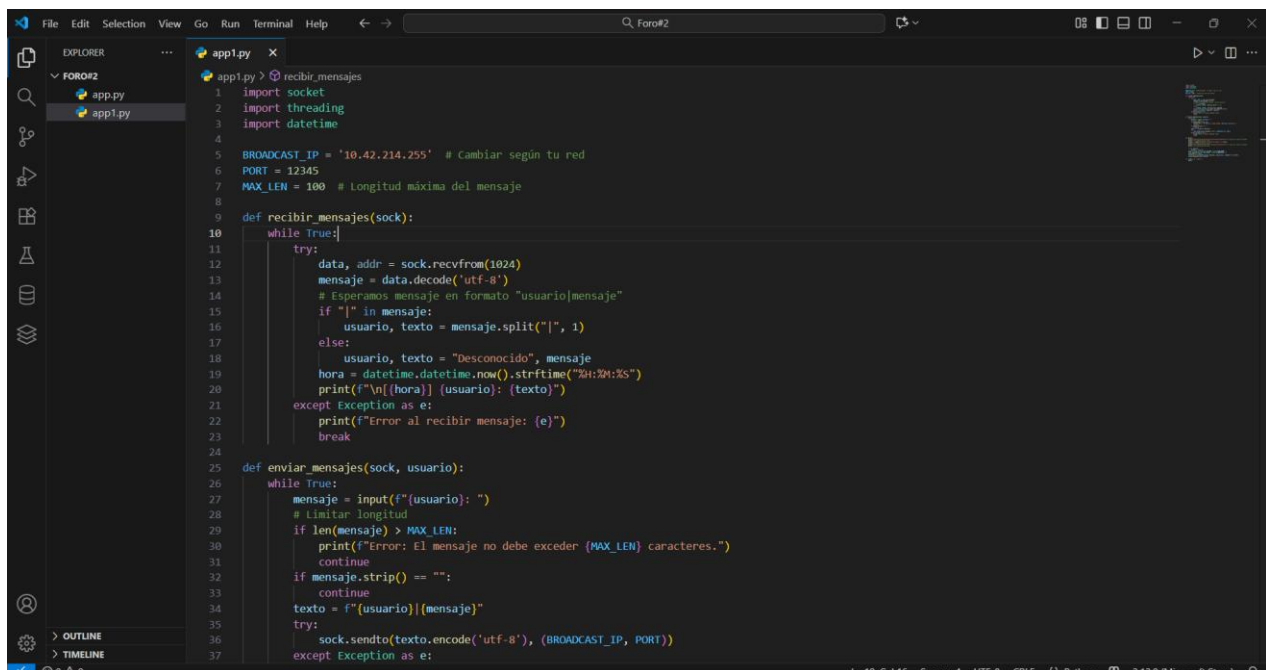
Benyahir Y. Martínez

Adriana M. Marrero

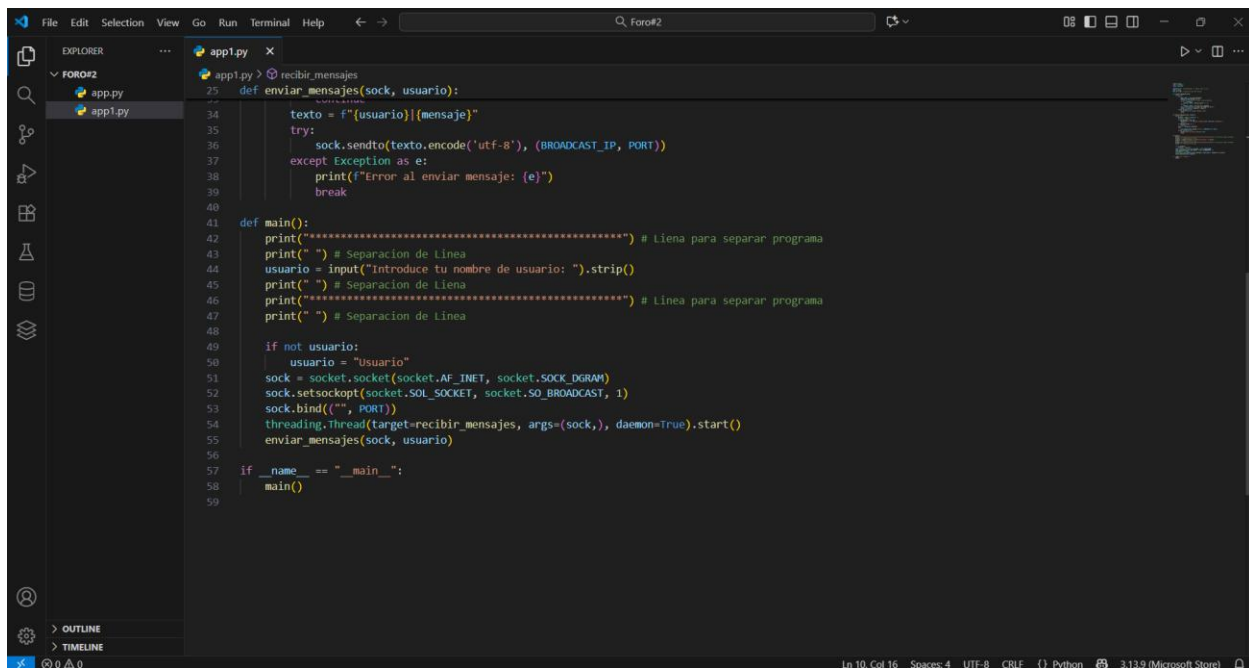
Descripción del proceso:

El programa permite que un usuario al correrlo pueda enviar un mensaje. Este mensaje es enviado usando una dirección broadcast a cualquier persona que también está utilizando el programa con esa misma dirección. Cuando el mensaje es recibido por el recipiente a este le aparecerá: el nombre de usuario del emisor, el tiempo en que este fue enviado y lo que el emisor le quiere comunicar. También permite que recipiente conteste al mensaje de este y viceversa permitiendo una interacción fluida. Unas adiciones que fueron añadidas a este fue la capacidad de permitir que el emisor y receptor puedan incluir un nombre de usuario para que estos puedan identificarse entre sí, una máxima capacidad de 100 caracteres para el mensaje y cronológica el tiempo exacto en el que el mensaje fue enviado.

Capturas del código (Benyahir):

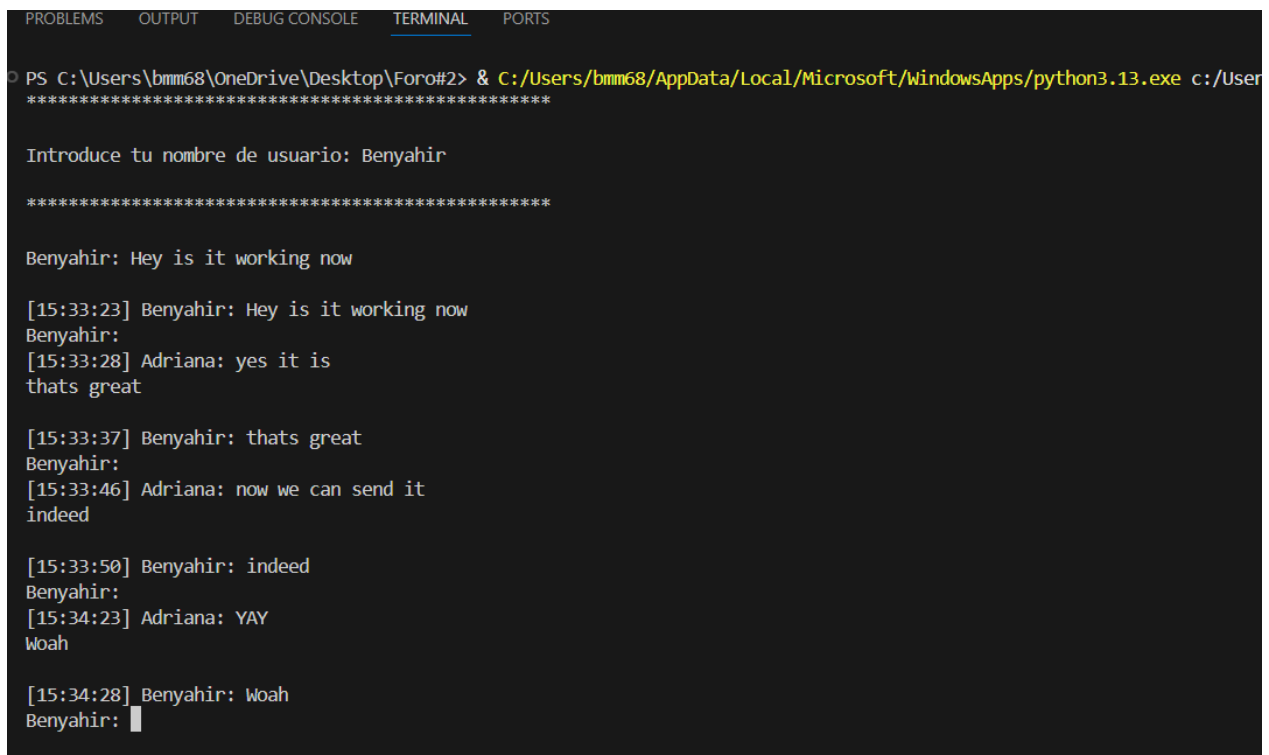
A screenshot of a code editor window titled 'app1.py'. The editor shows a Python script for a chat application. The script includes imports for socket, threading, and datetime. It defines a BROADCAST_IP, PORT, and MAX_LEN. Two functions are defined: 'recibir_mensajes(sock)' and 'enviar_mensajes(sock, usuario)'. The 'recibir_mensajes' function receives data from a socket, decodes it, and prints the user and message. The 'enviar_mensajes' function prompts the user for a message, checks its length, and sends it to the broadcast IP. The script is written in a dark-themed editor with syntax highlighting.

```
1 import socket
2 import threading
3 import datetime
4
5 BROADCAST_IP = '10.42.214.255' # Cambiar según tu red
6 PORT = 12345
7 MAX_LEN = 100 # Longitud máxima del mensaje
8
9 def recibir_mensajes(sock):
10     while True:
11         try:
12             data, addr = sock.recvfrom(1024)
13             mensaje = data.decode('utf-8')
14             # Esperamos mensaje en formato "usuario|mensaje"
15             if "|" in mensaje:
16                 usuario, texto = mensaje.split("|", 1)
17             else:
18                 usuario, texto = "Desconocido", mensaje
19             hora = datetime.datetime.now().strftime("%H:%M:%S")
20             print(f"\n[hora] (usuario): {texto}")
21         except Exception as e:
22             print(f"Error al recibir mensaje: {e}")
23             break
24
25 def enviar_mensajes(sock, usuario):
26     while True:
27         mensaje = input(f"(usuario): ")
28         # Limitar longitud
29         if len(mensaje) > MAX_LEN:
30             print(f"Error: El mensaje no debe exceder {MAX_LEN} caracteres.")
31             continue
32         if mensaje.strip() == "":
33             continue
34         texto = f"{usuario}|{mensaje}"
35         try:
36             sock.sendto(texto.encode('utf-8'), (BROADCAST_IP, PORT))
37         except Exception as e:
```



```
app1.py
25 def enviar_mensajes(sock, usuario):
26     while True:
27         texto = f"[usuario] {mensaje}"
28         try:
29             sock.sendto(texto.encode('utf-8'), (BROADCAST_IP, PORT))
30         except Exception as e:
31             print(f"Error al enviar mensaje: {e}")
32             break
33
34 def main():
35     print("*****") # Línea para separar programa
36     print(" ") # Separación de Línea
37     usuario = input("Introduce tu nombre de usuario: ").strip()
38     print(" ") # Separación de Línea
39     print("*****") # Línea para separar programa
40     print(" ") # Separación de Línea
41
42     if not usuario:
43         usuario = "Usuario"
44     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
45     sock.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
46     sock.bind(("", PORT))
47     threading.Thread(target=recibir_mensajes, args=(sock,), daemon=True).start()
48     enviar_mensajes(sock, usuario)
49
50 if __name__ == "__main__":
51     main()
52
53
54
55
56
57
58
59
```

Capturas del Proceso (Benyahir):



```
PS C:\Users\bmm68\OneDrive\Desktop\Foro#2> & C:/Users/bmm68/AppData/Local/Microsoft/WindowsApps/python3.13.exe c:/User
*****

Introduce tu nombre de usuario: Benyahir

*****

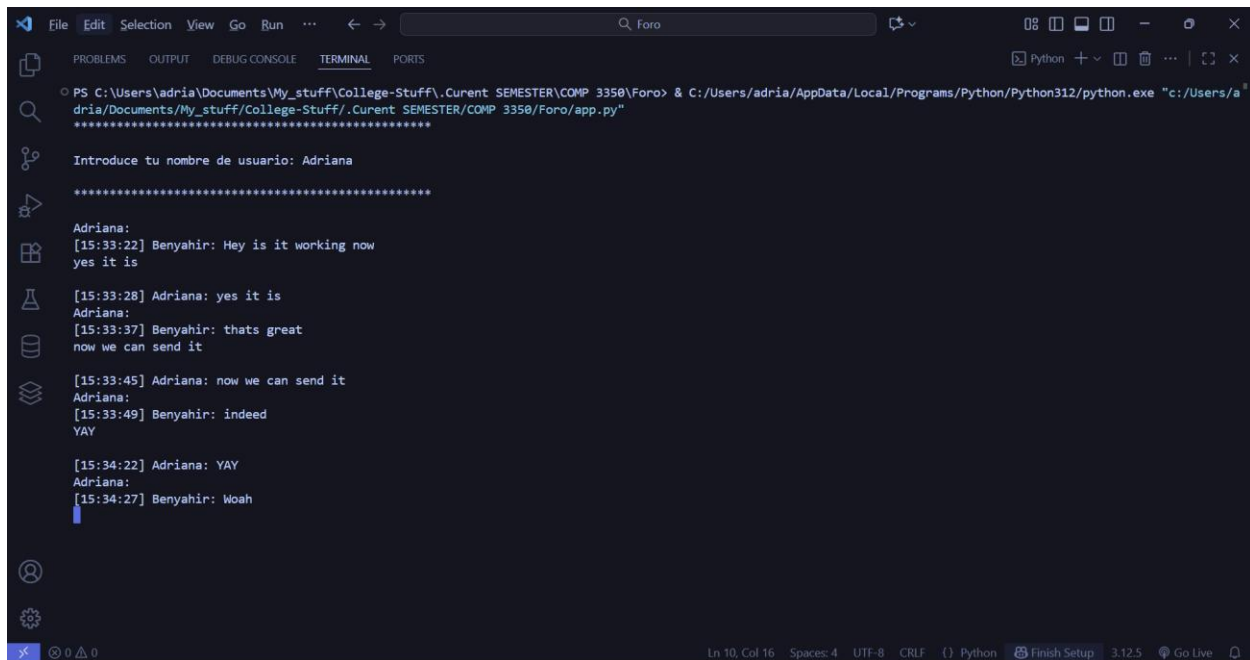
Benyahir: Hey is it working now

[15:33:23] Benyahir: Hey is it working now
Benyahir:
[15:33:28] Adriana: yes it is
thats great

[15:33:37] Benyahir: thats great
Benyahir:
[15:33:46] Adriana: now we can send it
indeed

[15:33:50] Benyahir: indeed
Benyahir:
[15:34:23] Adriana: YAY
Woah

[15:34:28] Benyahir: Woah
Benyahir: 
```

```
PS C:\Users\adria\Documents\My_stuff\College-Stuff\Current SEMESTER\COMP 3350\Foro> & C:/Users/adria/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/a
dria/Documents/My_stuff/College-Stuff/Current SEMESTER/COMP 3350/Foro/app.py"
*****
Introduce tu nombre de usuario: Adriana
*****
Adriana:
[15:33:22] Benyahir: Hey is it working now
yes it is
[15:33:28] Adriana: yes it is
Adriana:
[15:33:37] Benyahir: thats great
now we can send it
[15:33:45] Adriana: now we can send it
Adriana:
[15:33:49] Benyahir: indeed
YAY
[15:34:22] Adriana: YAY
Adriana:
[15:34:27] Benyahir: Woah
```

Reflexión final: “¿Que aprendieron sobre los dominios de dominios de broadcast y su relación con UDP?”

Se demostró muy bien como los dominios de broadcast funcionan al estar utilizados para él envió de los mensajes que la compañera y yo estuvimos realizando. En esto pudimos ver como el mensaje se podía enviar a los dispositivos conectados y que tenían la misma dirección de broadcast. Ocurrieron varias fallas en el desarrollo, desde puertos que tuvieron que ser creado, firewalls que estaban interfiriendo con él envió y llegada de los mensajes, hasta nuestro mayor problema la fuerza de la red. En ese caso notamos que si la fuerza de la red no era la suficiente uno de nosotros podía enviar un mensaje, pero cuando queríamos responder no aparecía en la terminal y esto fue algo que estuvo ocurriendo varias veces, pero de milagro pudimos tener una conexión a una red con cuya fue lo suficiente para demostrar la conversación que se ilustra en este documento.