Class Activity 3 : Class Work

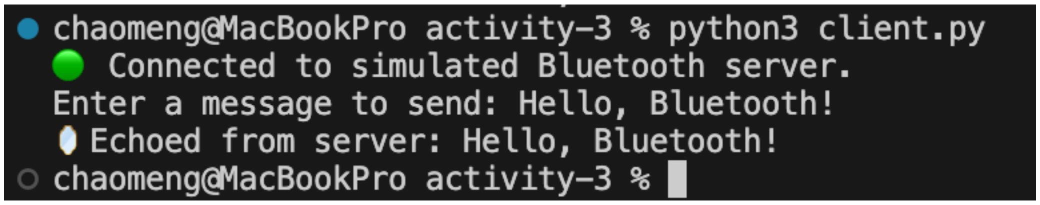
Group 6

Student Name: David Abiola   Student ID: 100974509

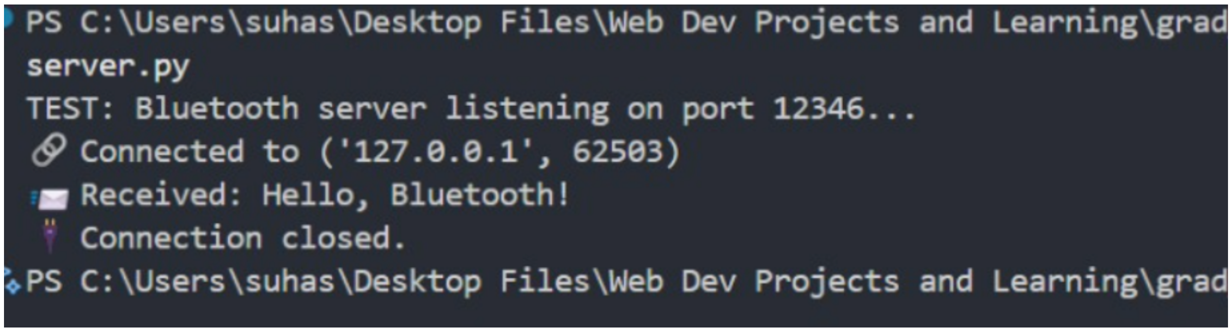
Student Name: Suhas Sunder Student ID:100548159

Student Name: Chao Meng   Student ID: 100997810

Testing

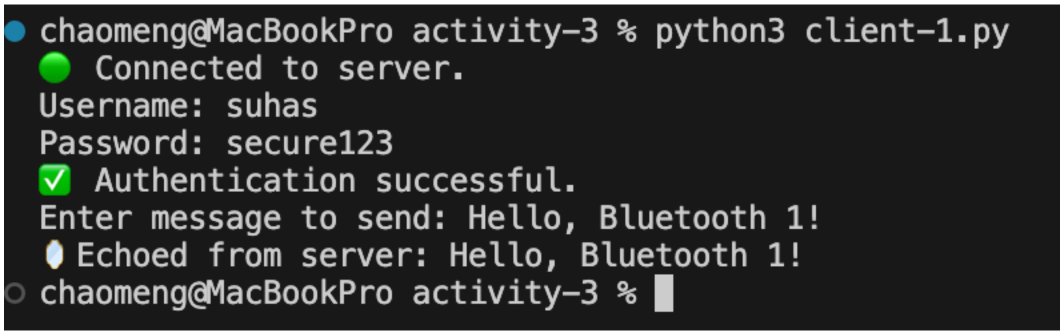


```
chaomeng@MacBookPro activity-3 % python3 client.py
 🟢 Connected to simulated Bluetooth server.
 Enter a message to send: Hello, Bluetooth!
 Echoed from server: Hello, Bluetooth!
chaomeng@MacBookPro activity-3 % 
```

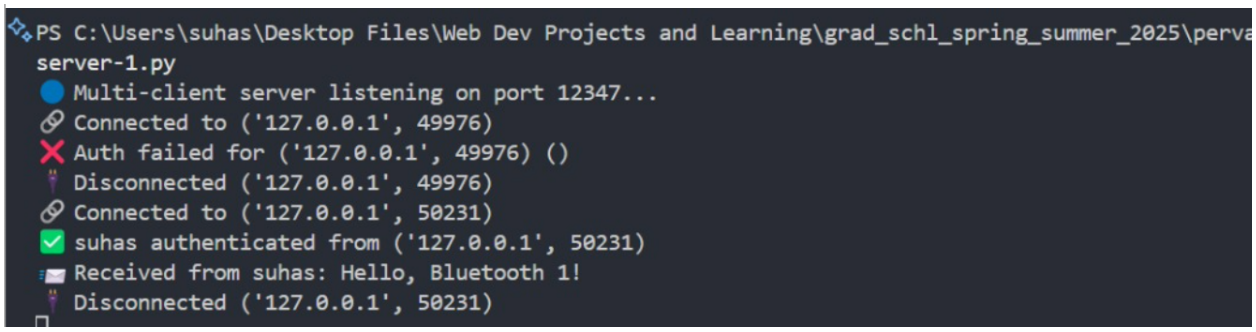


```
PS C:\Users\suhas\Desktop Files\Web Dev Projects and Learning\grad
server.py
TEST: Bluetooth server listening on port 12346...
🔗 Connected to ('127.0.0.1', 62503)
📨 Received: Hello, Bluetooth!
🔌 Connection closed.
PS C:\Users\suhas\Desktop Files\Web Dev Projects and Learning\grad
```

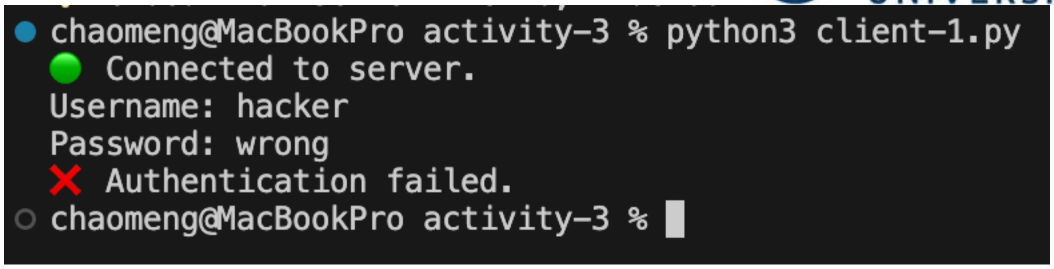
Challenge



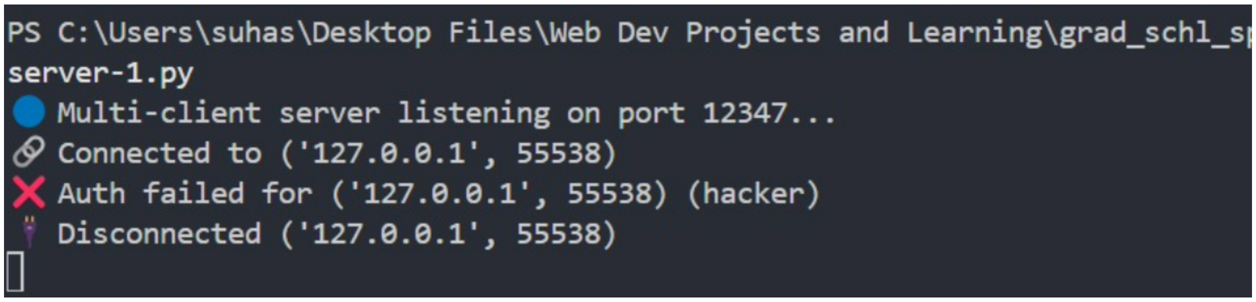
```
chaomeng@MacBookPro activity-3 % python3 client-1.py
🟢 Connected to server.
Username: suhas
Password: secure123
✅ Authentication successful.
Enter message to send: Hello, Bluetooth 1!
 Echoed from server: Hello, Bluetooth 1!
chaomeng@MacBookPro activity-3 % 
```



```
PS C:\Users\suhas\Desktop Files\Web Dev Projects and Learning\grad_schl_spring_summer_2025\perva
server-1.py
🔵 Multi-client server listening on port 12347...
🔗 Connected to ('127.0.0.1', 49976)
❌ Auth failed for ('127.0.0.1', 49976) ()
🔌 Disconnected ('127.0.0.1', 49976)
🔗 Connected to ('127.0.0.1', 50231)
✅ suhas authenticated from ('127.0.0.1', 50231)
📨 Received from suhas: Hello, Bluetooth 1!
🔌 Disconnected ('127.0.0.1', 50231)
```



```
● chaomeng@MacBookPro activity-3 % python3 client-1.py
  🟢 Connected to server.
  Username: hacker
  Password: wrong
  ❌ Authentication failed.
○ chaomeng@MacBookPro activity-3 % ▋
```



```
PS C:\Users\suhas\Desktop Files\Web Dev Projects and Learning\grad_schl_s
server-1.py
🔵 Multi-client server listening on port 12347...
🔗 Connected to ('127.0.0.1', 55538)
❌ Auth failed for ('127.0.0.1', 55538) (hacker)
🔌 Disconnected ('127.0.0.1', 55538)
▯
```

Brief Report

1 How service discovery works

When the server starts it calls

```
bluetooth.advertise_service(
   server_sock,
   name="EchoServer",
   service_id="00001101-0000-1000-8000-00805F9B34FB",
   service_classes=["00001101-0000-1000-8000-00805F9B34FB",
                  bluetooth.SERIAL_PORT_CLASS],
   profiles=[bluetooth.SERIAL_PORT_PROFILE],
)
```

This registers a Service Discovery Protocol (SDP) record on the local adapter.
The record stores three key items: (1) the UUID, (2) the RFCOMM channel (we fixed it to 5), and (3) a human-readable name.
A remote client broadcasts an SDP ServiceSearch request that contains the same UUID. BlueZ replies with the matching record; the client extracts the `host` (MAC address) and `port` (channel 5) fields and then opens an RFCOMM socket:

```
svc = bluetooth.find_service(uuid=UUID)[0]
sock.connect((svc["host"], svc["port"]))
```

Thus, discovery is connection-less and occurs before the RFCOMM link is created.

2 Challenges faced & solutions

Challenge	Impact	Solution
PyBluez could not be built on macOS (Xcode issues)	No `bluetooth` module on one device	Moved both server & client to Linux; alternatively used pre-built pybluez2 wheel.
“Address already in use” when re-running server	Previous test still held channel 5	Added graceful `sock.close()` in `finally` block or waited 30 s for RFCOMM release.
Permission denied on RFCOMM bind	Non-root users can’t open Bluetooth sockets	Launched scripts with `sudo`, or added user to the `bluetooth` group.
Single-threaded server blocked on slow client	Other clients had to wait	Wrapped each accepted socket in `threading.Thread`, allowing concurrent echoes.
Security: any nearby device could connect	Potential data sniffing	Enabled built-in authentication via `sock.set_security_level("medium")`; only paired, authenticated devices are accepted.

Challenges faced

The instructions posted did not work, so we had to find an alternative solution. Clearly, based on the above screenshots, it worked.