

Service-oriented Architecture INT 3505 1

Homework 1: Socket programming

1. Team member

	Student name	ID
1	Nguyễn Việt Minh Nghĩa	15021358

2. How to run

Run the server first,

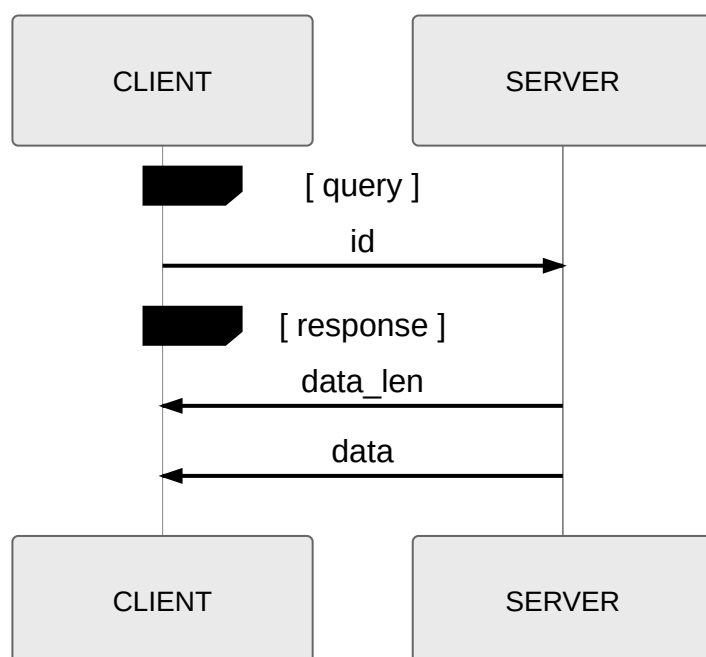
```
python3 server.py
```

then the client

```
python3 client.py
```

Enter the ID to get the information.

3. Description



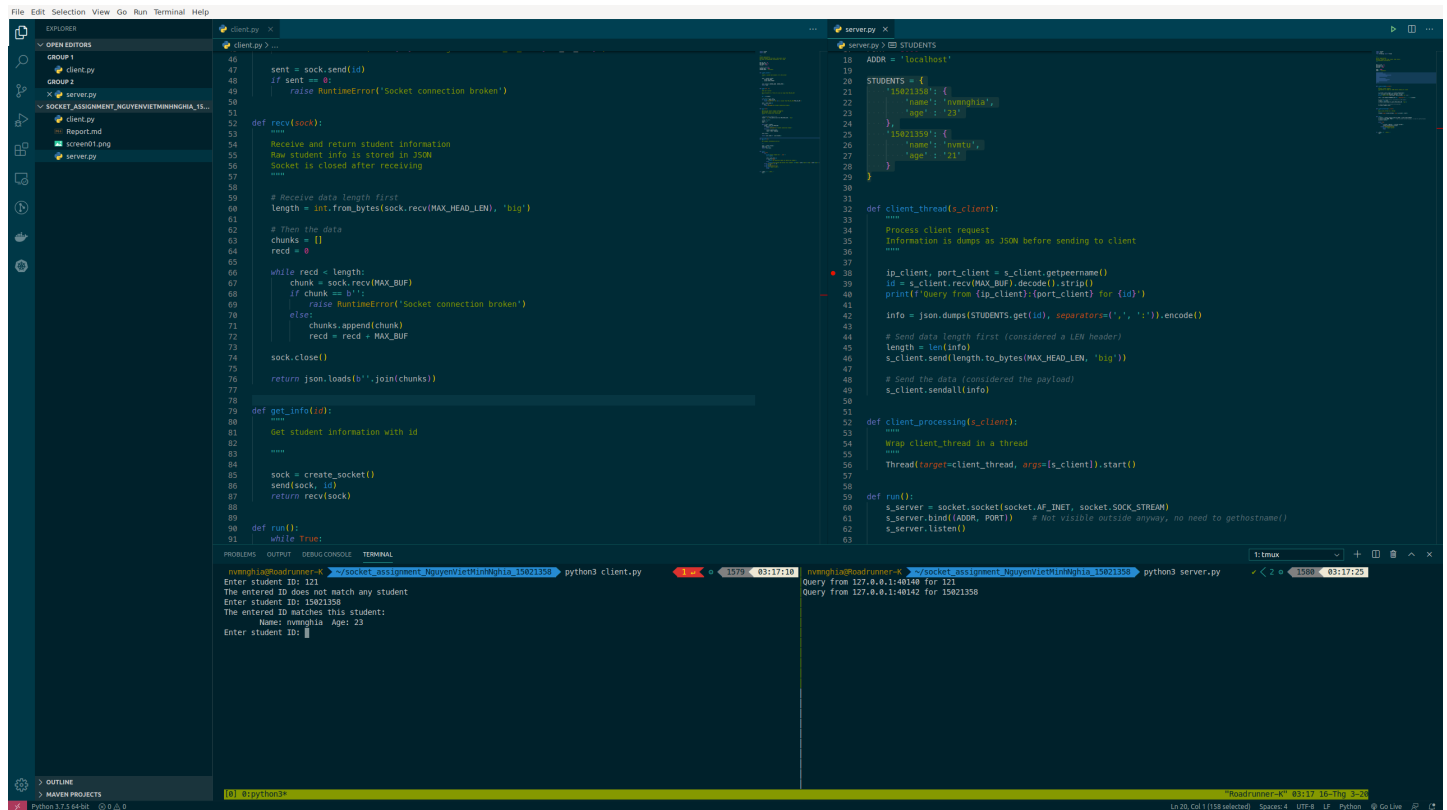
As TCP transmission doesn't have EOT , the size of the data must be known beforehand.

First, the client sends the id (MAX_ID_LEN=64 bytes max) to the server.

Then the server replies with the length of the data (itself MAX_HEAD_LEN=4 bytes max), simulating a LEN header.

Finally, after obtaining the data size to read, the client receives data from the server, in MAX_BUF=4096 -byte chunk.

4. Screenshot



```
client.py
46 sent = sock.send(id)
47 if sent == 0:
48     raise RuntimeError('Socket connection broken')
49
50
51
52 def recv(sock):
53     """
54     Receive and return student information
55     Raw student info is stored in JSON
56     Socket is closed after receiving
57     """
58
59     # Receive data length first
60     length = int.from_bytes(sock.recv(MAX_HEAD_LEN), 'big')
61
62     # Then the data
63     chunks = []
64     recd = 0
65
66     while recd < length:
67         chunk = sock.recv(MAX_BUF)
68         if chunk == b'':
69             raise RuntimeError('Socket connection broken')
70         else:
71             chunks.append(chunk)
72         recd = recd + MAX_BUF
73
74     sock.close()
75
76     return json.loads(b''.join(chunks))
77
78
79 def get_info(id):
80     """
81     Get student information with id
82     """
83
84
85     sock = create_socket()
86     send(sock, id)
87     return recv(sock)
88
89
90 def run():
91     while True:
```

```
server.py
18 ADDR = 'localhost'
19
20 STUDENTS = {
21     '15021358': {
22         'name': 'nmvngia18',
23         'age': '23'
24     },
25     '15021359': {
26         'name': 'nmvngia19',
27         'age': '21'
28     }
29 }
30
31
32 def client_thread(s_client):
33     """
34     Process client request
35     Information is dumps as JSON before sending to client
36     """
37
38     ip_client, port_client = s_client.getpeername()
39     id = s_client.recv(MAX_BUF).decode().strip()
40     print(f'Query from {ip_client}:{port_client} for {id}')
41
42     info = json.dumps(STUDENTS.get(id), separators=(',', ':')).encode()
43
44     # Send data length first (considered a LEN header)
45     length = len(info)
46     s_client.send(length.to_bytes(MAX_HEAD_LEN, 'big'))
47
48     # Send the data (considered the payload)
49     s_client.sendall(info)
50
51
52 def client_processing(s_client):
53     """
54     Wrap client_thread in a thread
55     """
56     Thread(target=client_thread, args=[s_client]).start()
57
58
59 def run():
60     s_server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
61     s_server.bind((ADDR, PORT)) # Not visible outside anyway, no need to gethostname()
62     s_server.listen()
63
```

```
python3 client.py
Enter student ID: 121
The entered ID does not match any student
Enter student ID: 15021358
The entered ID matches this student:
Name: nmvngia Age: 23
Enter student ID:

python3 server.py
Query from 127.0.0.1:4042 for 121
Query from 127.0.0.1:4042 for 15021358
```