




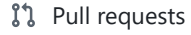




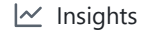












 keerthanamurugan06 / 2a_Stop_and_Wait_Protocol 





 Code  Pull requests  Actions  Projects  Wiki  Security  Insights  Settings

 Watch 0   

 0 stars  2.1k forks  0 watching  Branches  Activity  Tags

 Public repository · Forked from [NaliniG007/2a_Stop_and_Wait_Protocol](#)

  1 Branch  0 Tags  

 Go to file 

Go to file

Add file 

Code



This branch is [2 commits ahead of](#) NaliniG007/2a_Stop_and_Wait_Protocol:main .

[Contribute](#)[Sync fork](#)

keerthanamurugan06 Success

f8c5cf2 · 3 minutes ago



README.md

Success

3 minutes ago



Screenshot 2026-02-02 144...

updated files

5 minutes ago



Screenshot 2026-02-02 144...

updated files

5 minutes ago

 README

2a_Stop_and_Wait_Protocol

AIM

To write a python program to perform stop and wait protocol

ALGORITHM

1. Start the program.
2. Get the frame size from the user
3. To create the frame based on the user request.
4. To send frames to server from the client side.
5. If your frames reach the server it will send ACK signal to client
6. Stop the Program

PROGRAM

client.py



```
import socket
import time

client = socket.socket()
client.connect(('localhost', 8000))
client.settimeout(5)

while True:
    msg = input("Enter a message (or type 'exit' to quit): ")

    client.send(msg.encode())

    if msg.lower() == 'exit':
        print("Connection closed by client")
        client.close()
        break

    try:
        ack = client.recv(1024).decode()
        if ack == "ACK":
            print(f"Server acknowledged: {ack}")
    except socket.timeout:
        print("No ACK received, retransmitting...")
        continue
```

server.py

```
import socket
server = socket.socket()
server.bind(('localhost', 8000))
server.listen(1)
print("Server is listening...")
conn, addr = server.accept()
print(f"Connected with {addr}")

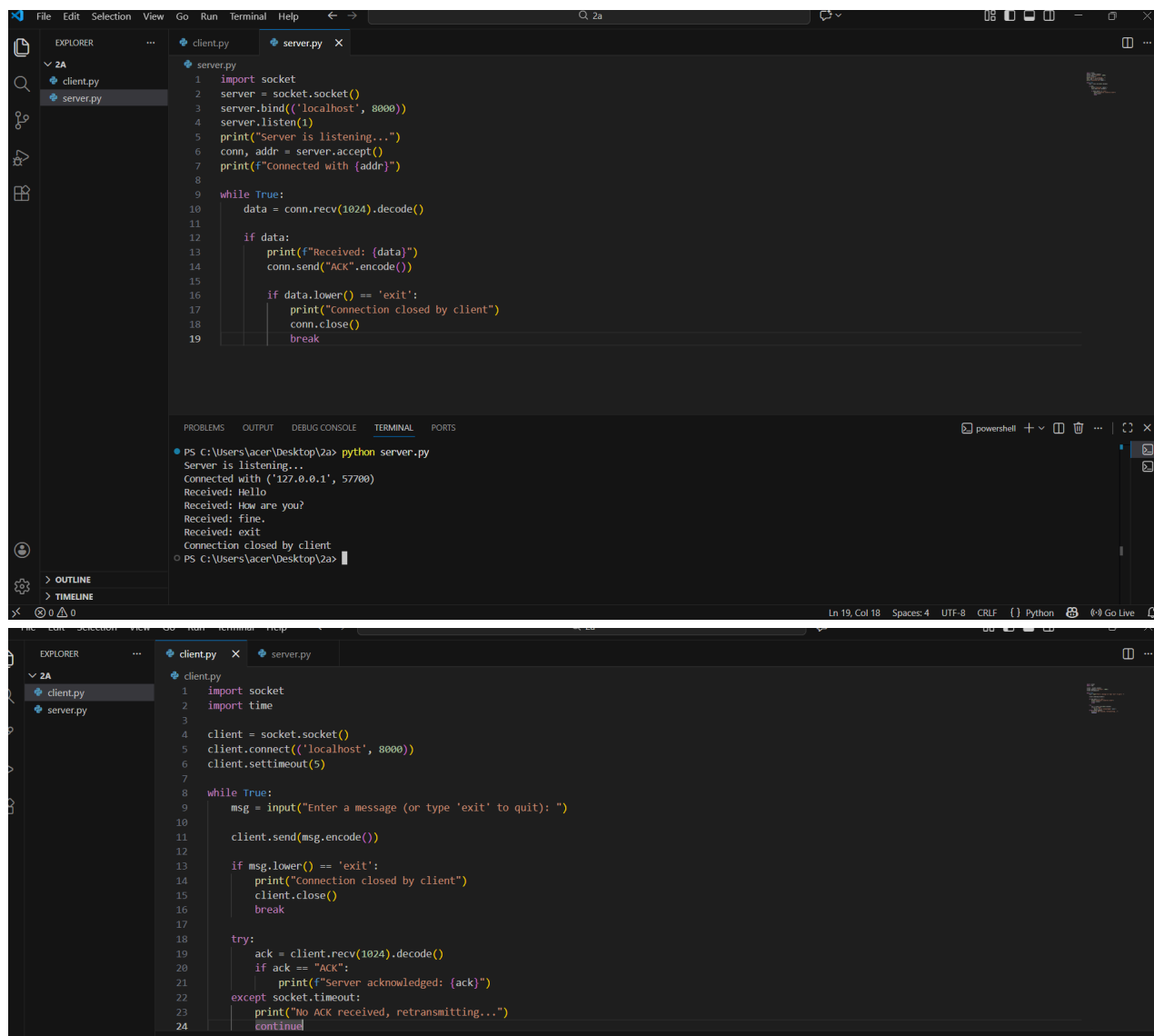
while True:
    data = conn.recv(1024).decode()

    if data:
        print(f"Received: {data}")
        conn.send("ACK".encode())

        if data.lower() == 'exit':
            print("Connection closed by client")
            conn.close()
            break
```



OUTPUT



```
server.py
1 import socket
2 server = socket.socket()
3 server.bind(('localhost', 8000))
4 server.listen(1)
5 print("Server is listening...")
6 conn, addr = server.accept()
7 print(f"Connected with {addr}")
8
9 while True:
10     data = conn.recv(1024).decode()
11
12     if data:
13         print(f"Received: {data}")
14         conn.send("ACK".encode())
15
16         if data.lower() == 'exit':
17             print("connection closed by client")
18             conn.close()
19             break
```

```
PS C:\Users\acer\Desktop\2a> python server.py
Server is listening...
Connected with ('127.0.0.1', 57700)
Received: Hello
Received: How are you?
Received: fine.
Received: exit
Connection closed by client
PS C:\Users\acer\Desktop\2a>
```

```
client.py
1 import socket
2 import time
3
4 client = socket.socket()
5 client.connect(('localhost', 8000))
6 client.settimeout(5)
7
8 while True:
9     msg = input("Enter a message (or type 'exit' to quit): ")
10
11     client.send(msg.encode())
12
13     if msg.lower() == 'exit':
14         print("Connection closed by client")
15         client.close()
16         break
17
18     try:
19         ack = client.recv(1024).decode()
20         if ack == "ACK":
21             print(f"Server acknowledged: {ack}")
22     except socket.timeout:
23         print("No ACK received, retransmitting...")
24         continue
```

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)