server.py

```python
import socket
import time
import random
import json


SERVER_IP = "127.0.0.1"
PORT = 5000




def get_local_time():
   return random.randint(int(time.time() - 1e5), int(time.time() + 1e5))




def main():
   ## Create server socket
   server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   server_socket.bind((SERVER_IP, PORT))
   server_socket.listen(1)


   ## Get local time
   server_local_time = get_local_time()


   print(f"Time server listening on {SERVER_IP}:{PORT}")
   print(f"Server time: {server_local_time}")


   is_client_enough = False


   clients = []


   while not is_client_enough:
       ## Accept client connection
       client_socket, client_address = server_socket.accept()
       print(f"Connection established with {client_address}")


       clients.append(client_socket)


```

```python
        option = input("Do you want to add more clients? (y/n) ")
        if option == "n" or option == "N":
            is_client_enough = True
        else:
            print("Waiting for more clients..." + "\n")


    client_local_times = []


    ## Get local time from all clients
    for client_socket in clients:
        time_req_body = json.dumps({"operation": "time_req"})
        client_socket.send(time_req_body.encode())


        client_local_time_response = json.loads(client_socket.recv(1024).decode())


        client_local_times.append(float(client_local_time_response["client_time"]))


    ## Calculate adjusted time
    average_offset = sum(client_local_times) / len(client_local_times)
    adjusted_time_offset = (server_local_time + average_offset) / 2


    ## Send adjusted time to all clients
    for i, client_socket in enumerate(clients):
        print(
            f"Client {client_socket.getpeername()} LocalTime : {client_local_times[i]}"
        )
        adjusted_time = json.dumps(
            {
                "adjusted_time": client_local_times[i] - adjusted_time_offset,
                "operation": "time_adj",
            }
        )


        client_socket.send(str(adjusted_time).encode())
        print(f"Adjusted time sent to {client_socket.getpeername()}")


    server_socket.close()




if __name__ == "__main__":
    main()



```

client.py

```python
import socket
import time
import json
import random


SERVER_IP = "127.0.0.1"
PORT = 5000




def get_local_time():
    return random.randint(int(time.time() - 1e5), int(time.time() + 1e5))




def main():
    ## Connect to server
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((SERVER_IP, PORT))
    print(f"Connected to {SERVER_IP}:{PORT}")


    ## Get local time
    client_local_time = get_local_time()


    time_adjusted = False


    while not time_adjusted:
        server_res = json.loads(client_socket.recv(1024).decode())


        if server_res["operation"] == "time_req":
            ## Send local time to server
            print(f"Local time: {client_local_time}")
            client_socket.send(json.dumps({"client_time": client_local_time}).encode())


        if server_res["operation"] == "time_adj":
            ## Adjust local time
            print(f"Time adjustment: {server_res['adjusted_time']}")
            client_local_time += float(server_res["adjusted_time"])


            print(f"Adjusted time: {client_local_time}")


            time_adjusted = True


    client_socket.close()




if __name__ == "__main__":
    main()



```