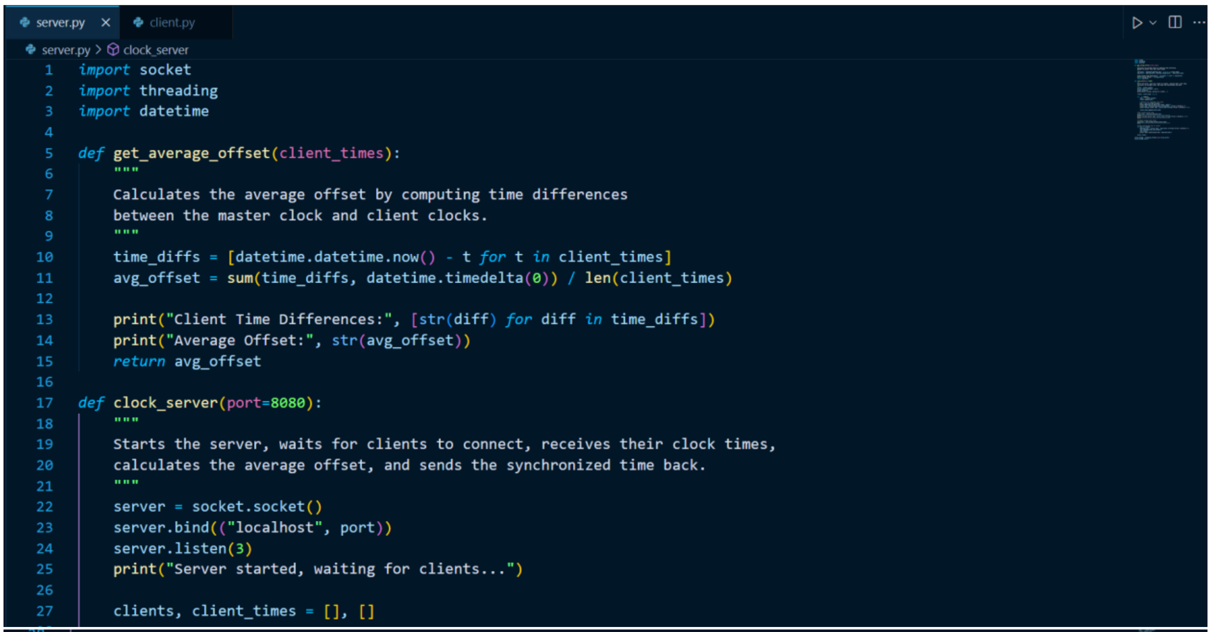
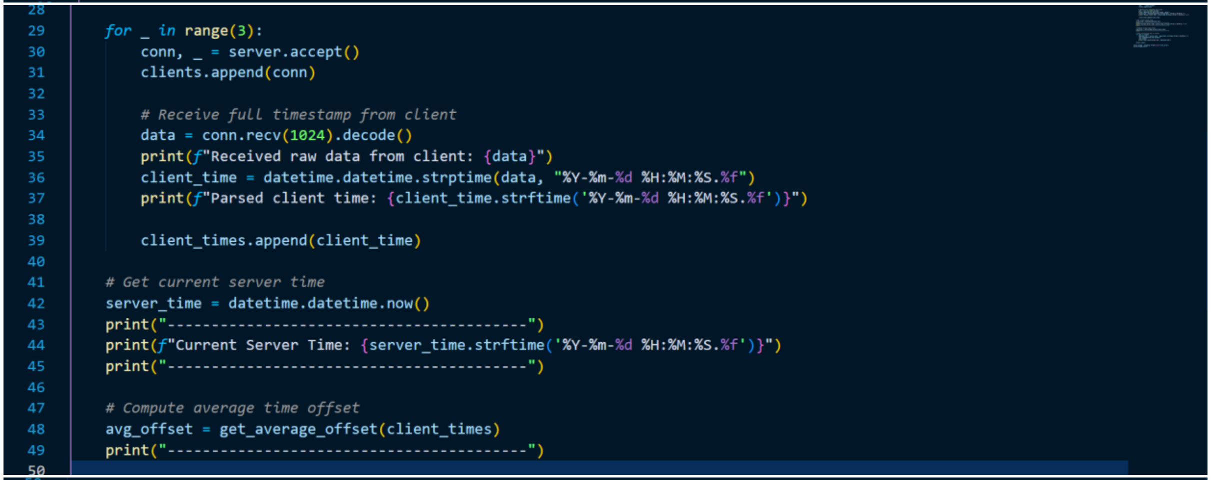
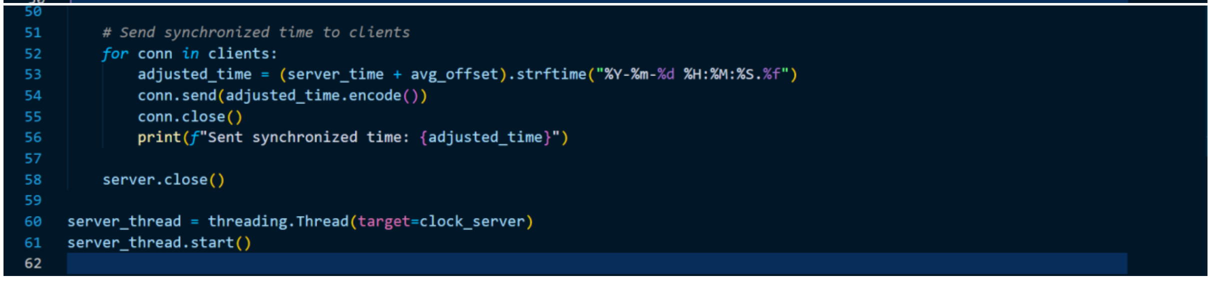
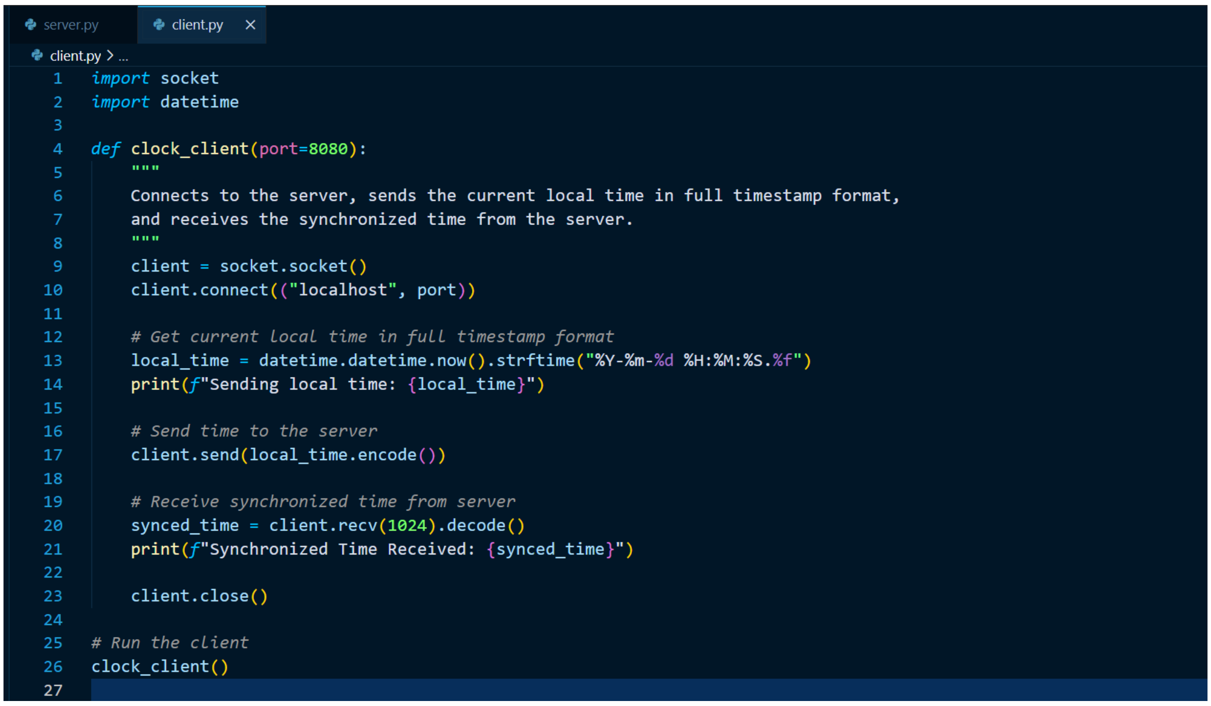






```python
import socket
import threading
import datetime

def get_average_offset(client_times):
    """
    Calculates the average offset by computing time differences
    between the master clock and client clocks.
    """
    time_diffs = [datetime.datetime.now() - t for t in client_times]
    avg_offset = sum(time_diffs, datetime.timedelta(0)) / len(client_times)

    print("Client Time Differences:", [str(diff) for diff in time_diffs])
    print("Average Offset:", str(avg_offset))
    return avg_offset

def clock_server(port=8080):
    """
    Starts the server, waits for clients to connect, receives their clock times,
    calculates the average offset, and sends the synchronized time back.
    """
    server = socket.socket()
    server.bind(("localhost", port))
    server.listen(3)
    print("Server started, waiting for clients...")

    clients, client_times = [], []

    for _ in range(3):
        conn, _ = server.accept()
        clients.append(conn)

        # Receive full timestamp from client
        data = conn.recv(1024).decode()
        print(f"Received raw data from client: {data}")
        client_time = datetime.datetime.strptime(data, "%Y-%m-%d %H:%M:%S.%f")
        print(f"Parsed client time: {client_time.strftime('%Y-%m-%d %H:%M:%S.%f')}")

        client_times.append(client_time)

    # Get current server time
    server_time = datetime.datetime.now()
    print("----------------------------------------")
    print(f"Current Server Time: {server_time.strftime('%Y-%m-%d %H:%M:%S.%f')}")
    print("----------------------------------------")

    # Compute average time offset
    avg_offset = get_average_offset(client_times)
    print("----------------------------------------")

    # Send synchronized time to clients
    for conn in clients:
        adjusted_time = (server_time + avg_offset).strftime("%Y-%m-%d %H:%M:%S.%f")
        conn.send(adjusted_time.encode())
        conn.close()
        print(f"Sent synchronized time: {adjusted_time}")

    server.close()

server_thread = threading.Thread(target=clock_server)
server_thread.start()
```



```python
import socket
import datetime

def clock_client(port=8080):
    """
    Connects to the server, sends the current local time in full timestamp format,
    and receives the synchronized time from the server.
    """
    client = socket.socket()
    client.connect(("localhost", port))

    # Get current local time in full timestamp format
    local_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S.%f")
    print(f"Sending local time: {local_time}")

    # Send time to the server
    client.send(local_time.encode())

    # Receive synchronized time from server
    synced_time = client.recv(1024).decode()
    print(f"Synchronized Time Received: {synced_time}")

    client.close()

# Run the client
clock_client()

```