# ASSIGNMENT NO-4

## Client.py

```
from timeit import default_timer as timer
from dateutil import parser
import threading
import datetime
import socket
import time

# client thread function used to send time at client side
def startSendingTime(slave_client):
    while True:
        # provide server with clock time at the client
        slave_client.send(str(datetime.datetime.now()).encode())
        print("Recent time sent successfully")
        time.sleep(5)

# client thread function used to receive synchronized time
def startReceivingTime(slave_client):
    while True:
        # receive data from the server
        synchronized_time = parser.parse(slave_client.recv(1024).decode())
        print("Synchronized time at the client is: " + str(synchronized_time))

# function used to Synchronize client process time
def initiateSlaveClient(port=8080):
    slave_client = socket.socket()
    # connect to the clock server on local computer
    slave_client.connect(('127.0.0.1', port))
    # start sending time to server
    print("Starting to receive time from server\n")
    send_time_thread = threading.Thread(
        target=startSendingTime,
        args=(slave_client,)
    )
    send_time_thread.start()
    # start receiving synchronized time from server
    print("Starting to receive synchronized time from server\n")
    receive_time_thread = threading.Thread(
        target=startReceivingTime,
        args=(slave_client,)
    )
    receive_time_thread.start()

# Driver function
if __name__ == '__main__':
    # initialize the Slave / Client
    initiateSlaveClient(port=8080)
```

**Server.py**

```
# Python3 program imitating a clock server
from functools import reduce
from dateutil import parser
import threading
import socket
import time
import datetime
# datastructure used to store client address and clock data
client_data = {}
```

```
''' nested thread function used to receive
clock time from a connected client '''
def startReceivingClockTime(connector, address):
    while True:
        # receive clock time
        clock_time_string = connector.recv(1024).decode()
        clock_time = parser.parse(clock_time_string)
        clock_time_diff = datetime.datetime.now() - clock_time
        client_data[address] = {
            "clock_time": clock_time,
            "time_difference": clock_time_diff,
            "connector": connector
        }
        print("Client Data updated with: " + str(address))
        time.sleep(5)

''' master thread function used to open portal for
accepting clients over given port '''
def startConnecting(master_server):
    # fetch clock time at slaves / clients
    while True:
        # accepting a client / slave clock client
        master_slave_connector, addr = master_server.accept()
        slave_address = str(addr[0]) + ":" + str(addr[1])
        print(slave_address + " got connected successfully")
        current_thread = threading.Thread(
            target=startReceivingClockTime,
            args=(master_slave_connector, slave_address)
        )
        current_thread.start()
```

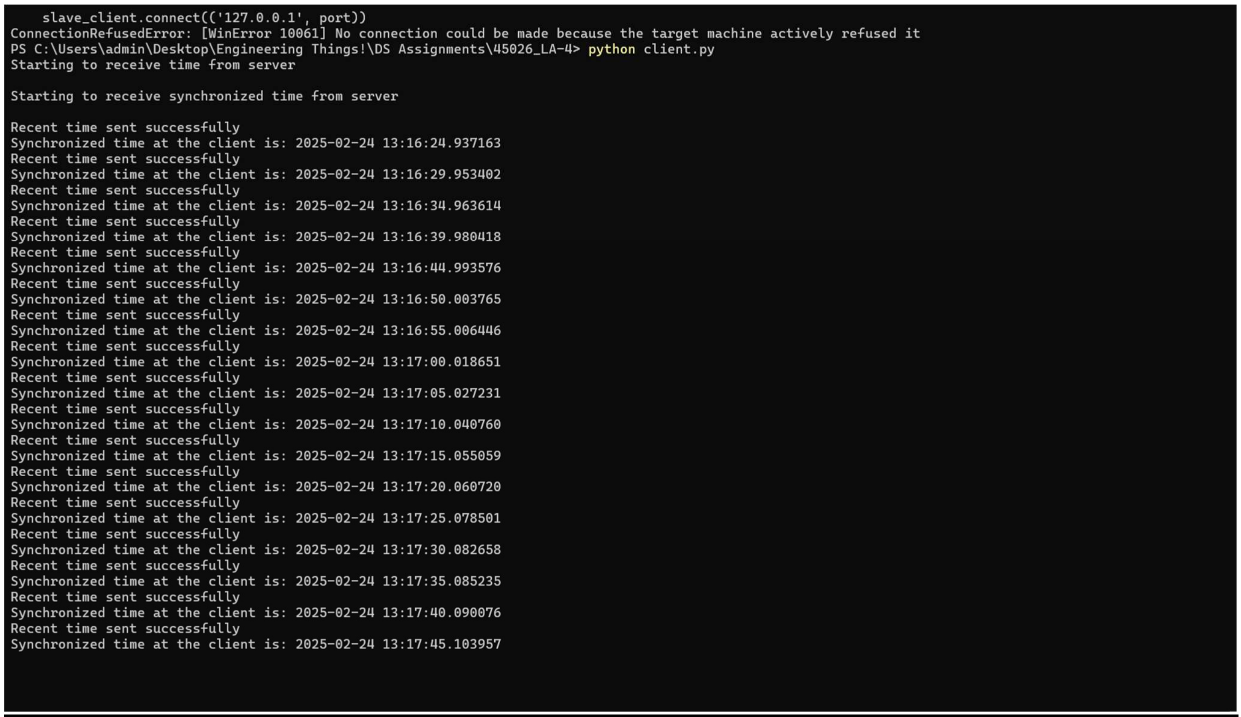
```
# subroutine function used to fetch average clock difference
def getAverageClockDiff():
    current_client_data = client_data.copy()
    time_difference_list = [
        client['time_difference'] for client_addr, client in client_data.items()
    ]
    sum_of_clock_difference = reduce(
        lambda x, y: x + y, time_difference_list, datetime.timedelta(0, 0)
    )
    average_clock_difference = sum_of_clock_difference / len(client_data)
    return average_clock_difference


''' master sync thread function used to generate
cycles of clock synchronization in the network '''
def synchronizeAllClocks():
    while True:
        print("New synchronization cycle started.")
        print("Number of clients to be synchronized: " + str(len(client_data)))
        if len(client_data) > 0:
            average_clock_difference = getAverageClockDiff()
            for client_addr, client in client_data.items():
                print("printing" ,client_addr, client)
                try:
                    synchronized_time = datetime.datetime.now() + average_clock_difference
                    client['connector'].send(str(synchronized_time).encode())
                except Exception as e:
                    print("Something went wrong while sending synchronized time through " +
str(client_addr))
            else:
                print("No client data. Synchronization not applicable.")
            print("\n\n")
            time.sleep(5)

# function used to initiate the Clock Server / Master Node
def initiateClockServer(port=8080):
    master_server = socket.socket()
    master_server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    print("Socket at master node created successfully\n")
    master_server.bind(('', port))
    # Start listening to requests
    master_server.listen(10)
    print("Clock server started...\n")
    # start making connections
    print("Starting to make connections...\n")
    master_thread = threading.Thread(
        target=startConnecting,
        args=(master_server,)
    )
    master_thread.start()
    # start synchronization1
    print("Starting synchronization parallelly...\n")
    sync_thread = threading.Thread(
        target=synchronizeAllClocks,
        args=()
    )
    sync_thread.start()

# Driver function
if __name__ == '__main__':
    # Trigger the Clock Server
    initiateClockServer(port=8080)
```

## Output-Client

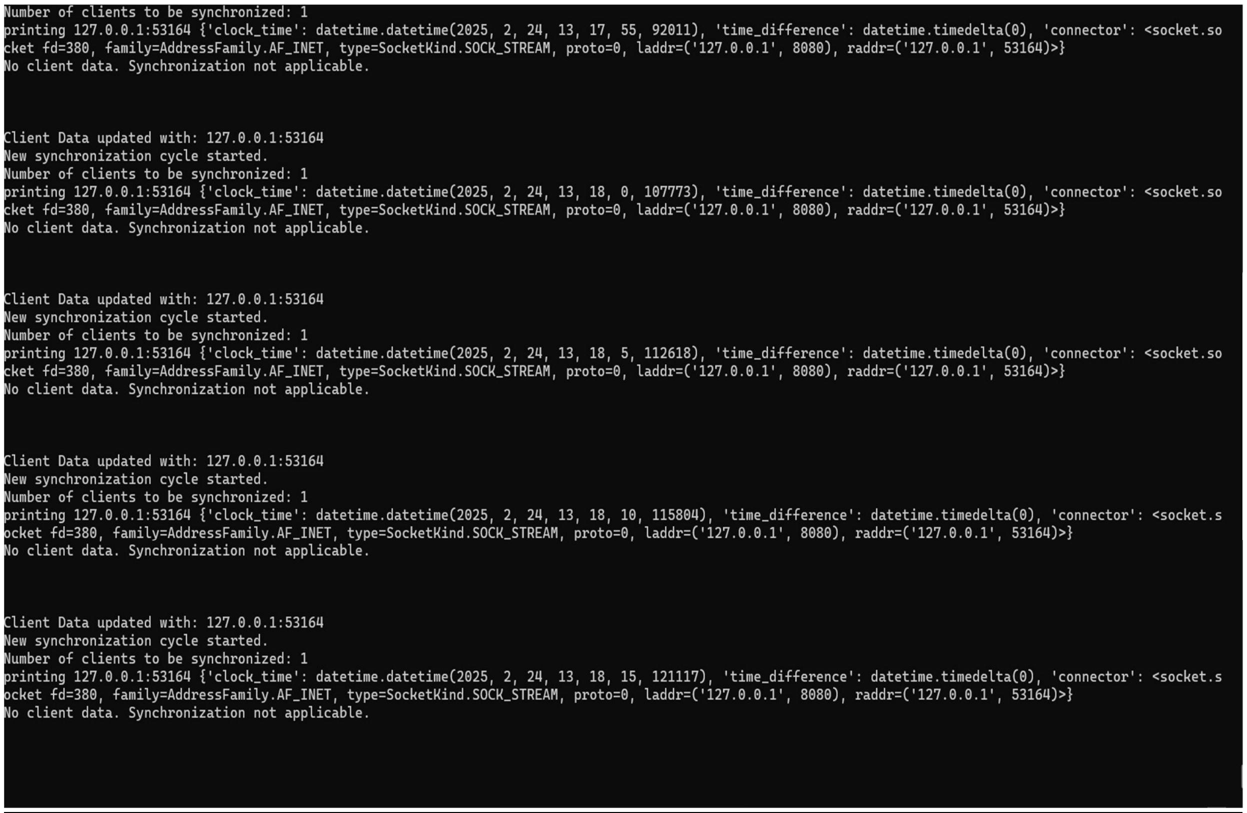


```
    slave_client.connect(('127.0.0.1', port))
ConnectionRefusedError: [WinError 10061] No connection could be made because the target machine actively refused it
PS C:\Users\admin\Desktop\Engineering Things!\DS Assignments\45026_LA-4> python client.py
Starting to receive time from server

Starting to receive synchronized time from server

Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:24.937163
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:29.953402
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:34.963614
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:39.980418
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:44.993576
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:50.003765
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:16:55.006446
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:00.018651
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:05.027231
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:10.040760
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:15.055059
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:20.060720
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:25.078501
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:30.082658
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:35.085235
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:40.090076
Recent time sent successfully
Synchronized time at the client is: 2025-02-24 13:17:45.103957
```

## Output-Server



```
Number of clients to be synchronized: 1
printing 127.0.0.1:53164 {'clock_time': datetime.datetime(2025, 2, 24, 13, 17, 55, 92011), 'time_difference': datetime.timedelta(0), 'connector': <socket.so
cket fd=380, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8080), raddr=('127.0.0.1', 53164)>}
No client data. Synchronization not applicable.



Client Data updated with: 127.0.0.1:53164
New synchronization cycle started.
Number of clients to be synchronized: 1
printing 127.0.0.1:53164 {'clock_time': datetime.datetime(2025, 2, 24, 13, 18, 0, 107773), 'time_difference': datetime.timedelta(0), 'connector': <socket.so
cket fd=380, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8080), raddr=('127.0.0.1', 53164)>}
No client data. Synchronization not applicable.



Client Data updated with: 127.0.0.1:53164
New synchronization cycle started.
Number of clients to be synchronized: 1
printing 127.0.0.1:53164 {'clock_time': datetime.datetime(2025, 2, 24, 13, 18, 5, 112618), 'time_difference': datetime.timedelta(0), 'connector': <socket.so
cket fd=380, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8080), raddr=('127.0.0.1', 53164)>}
No client data. Synchronization not applicable.



Client Data updated with: 127.0.0.1:53164
New synchronization cycle started.
Number of clients to be synchronized: 1
printing 127.0.0.1:53164 {'clock_time': datetime.datetime(2025, 2, 24, 13, 18, 10, 115804), 'time_difference': datetime.timedelta(0), 'connector': <socket.s
ocket fd=380, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8080), raddr=('127.0.0.1', 53164)>}
No client data. Synchronization not applicable.



Client Data updated with: 127.0.0.1:53164
New synchronization cycle started.
Number of clients to be synchronized: 1
printing 127.0.0.1:53164 {'clock_time': datetime.datetime(2025, 2, 24, 13, 18, 15, 121117), 'time_difference': datetime.timedelta(0), 'connector': <socket.s
ocket fd=380, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8080), raddr=('127.0.0.1', 53164)>}
No client data. Synchronization not applicable.
```