

# Testing Document: Assignment 1 COIS 3040H

By Sidak Singh Sra

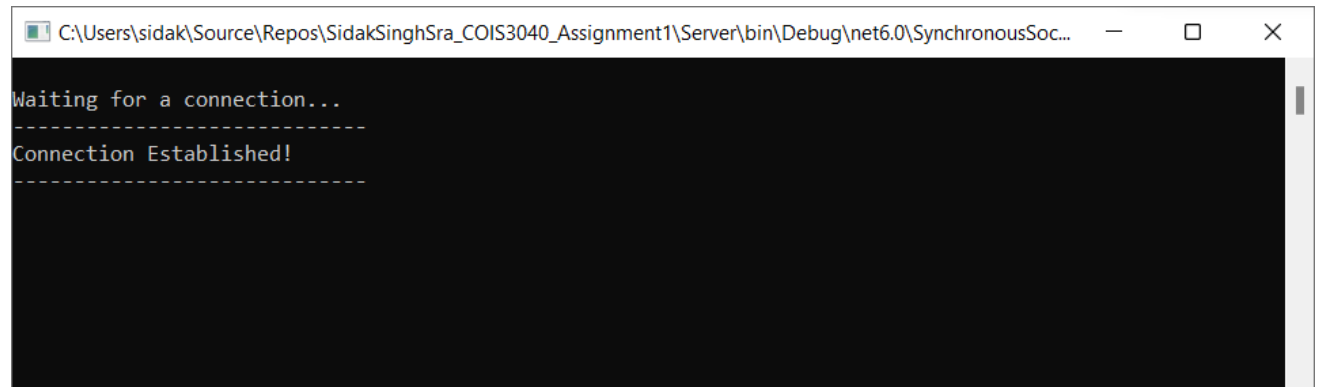
Student ID: 0689168

## Test 1

**Description:** In this test, the server was started first, then the client and Socket were immediately connected, and the server also transmitted a Knock Knock! message to the client application. In addition, after the socket is established, the server application answers with Connection Established!

**Purpose:** To see if a connection has been established between the client and the server, as well as to see if a Knock-Knock message has been received to the client application.

### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
```

### Client Application:



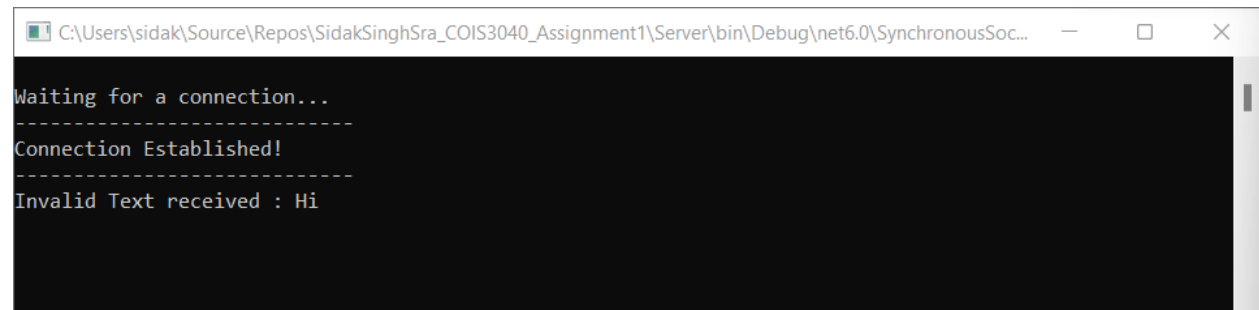
```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client:
```

## Test 2

**Description:** In this test, the client provided an invalid input, to which the server replied, "Please enter the proper response, it may be Who's there? As seen in the screenshots below. And the server waits for the client to re-enter the Valid string, prompting the client to enter a valid input as stated.

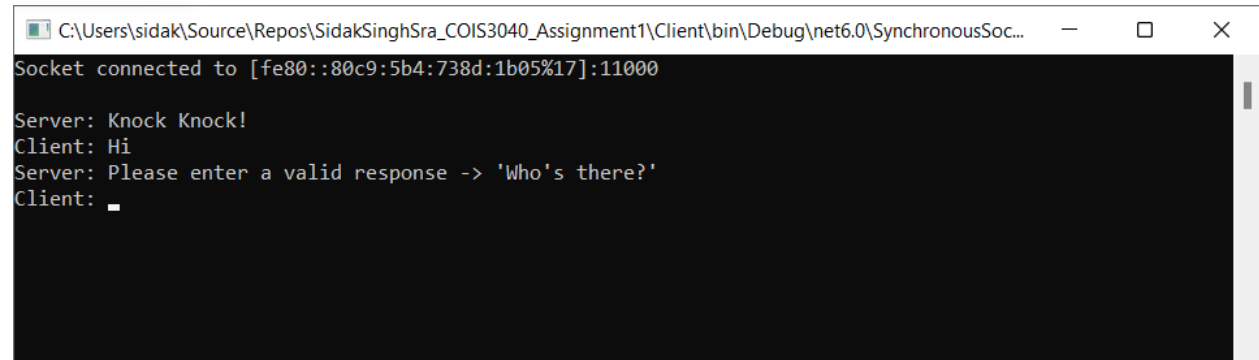
**Purpose:** To see what happens when a client sends the server an invalid message.

### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Invalid Text received : Hi
```

### Client Application:



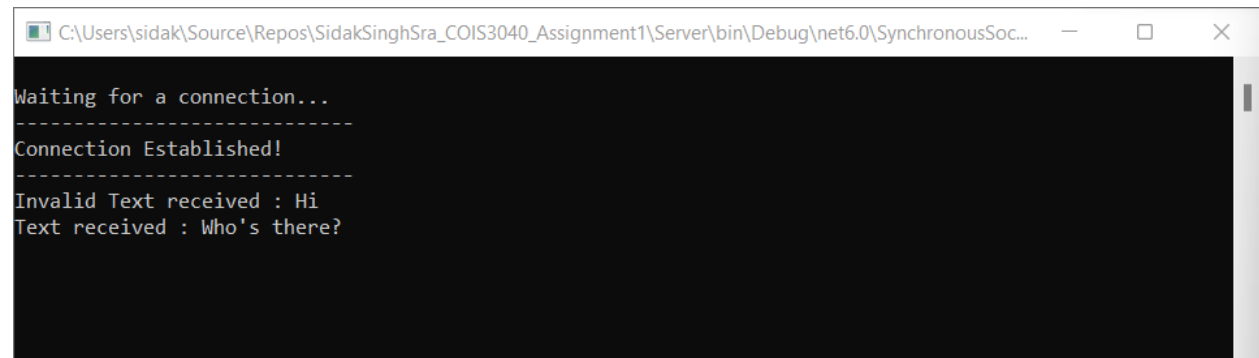
```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client: Hi
Server: Please enter a valid response -> 'Who's there?'
Client: _
```

### Test 3

**Description:** In this test, the client enters a valid string, and the server sends the client a random message name from an array.

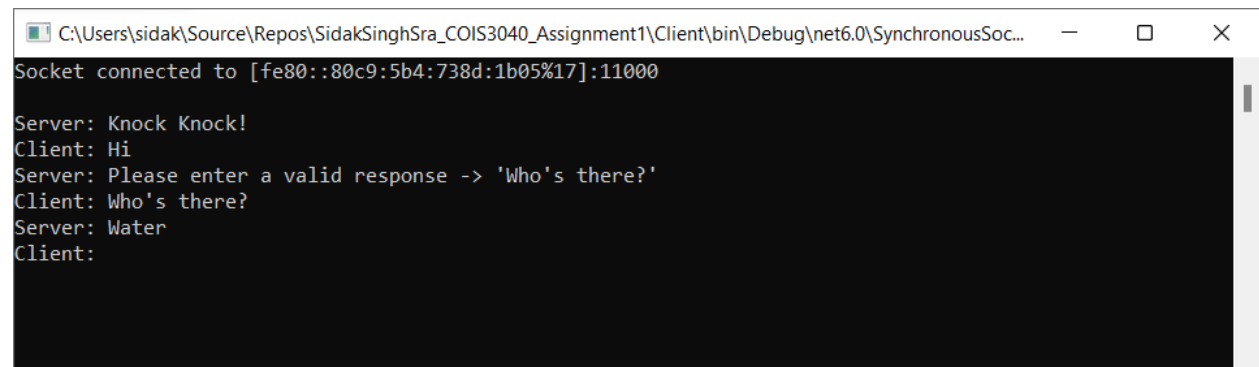
**Purpose:** To see what happens if a valid string is entered by the user. Will the Server be able to correctly respond to the input message?

#### Server Application:

A screenshot of a Windows console window titled "C:\Users\sidak\Source\Repos\SidakSinghSra\_COIS3040\_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...". The console output shows the server's state: it starts by waiting for a connection, then reports "Connection Established!". It then receives the input "Hi" and responds with "Invalid Text received : Hi". Finally, it receives the input "Who's there?" and responds with "Text received : Who's there?".

```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Invalid Text received : Hi
Text received : Who's there?
```

#### Client Application:

A screenshot of a Windows console window titled "C:\Users\sidak\Source\Repos\SidakSinghSra\_COIS3040\_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...". The console output shows the client's interaction with the server: it connects to the server at [fe80::80c9:5b4:738d:1b05%17]:11000. The server sends the message "Knock Knock!". The client responds with "Hi". The server then prompts the client with "Please enter a valid response -> 'Who's there?'". The client responds with "Who's there?". The server responds with "Water". Finally, the client sends an empty message.

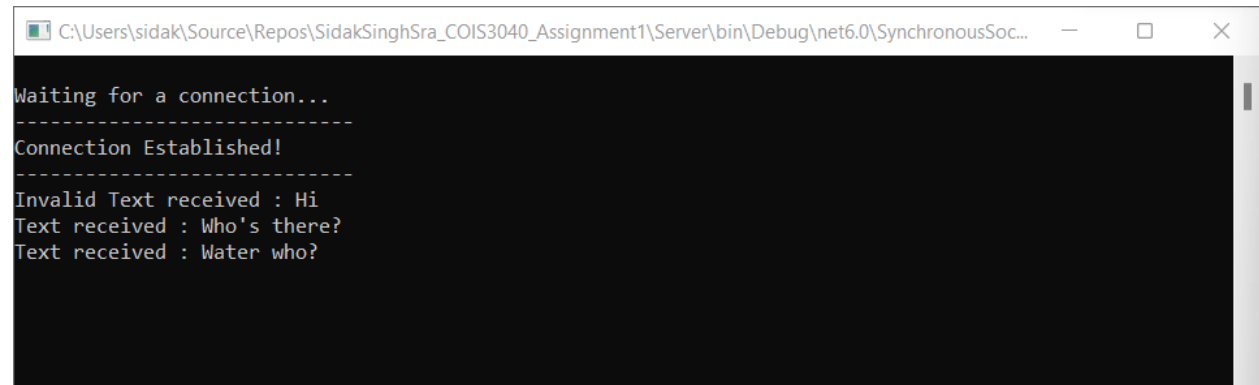
```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client: Hi
Server: Please enter a valid response -> 'Who's there?'
Client: Who's there?
Server: Water
Client:
```

#### Test 4

**Description:** In this test, the Client responds to the Server's name. In addition, Server delivered the client a Knock-Knock joke. Furthermore, the Server transmitted a Knock-Knock message to the client, requesting that the user input again.

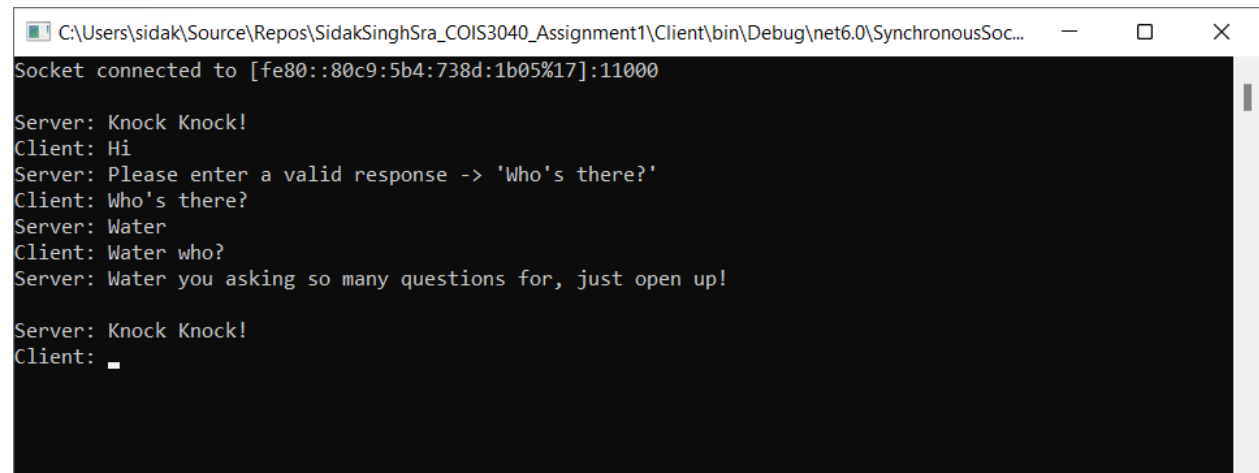
**Purpose:** To test what happens if a user enters a legitimate string, as well as whether the server will send another Knock-Knock message to the client.

#### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Invalid Text received : Hi
Text received : Who's there?
Text received : Water who?
```

#### Client Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client: Hi
Server: Please enter a valid response -> 'Who's there?'
Client: Who's there?
Server: Water
Client: Water who?
Server: Water you asking so many questions for, just open up!

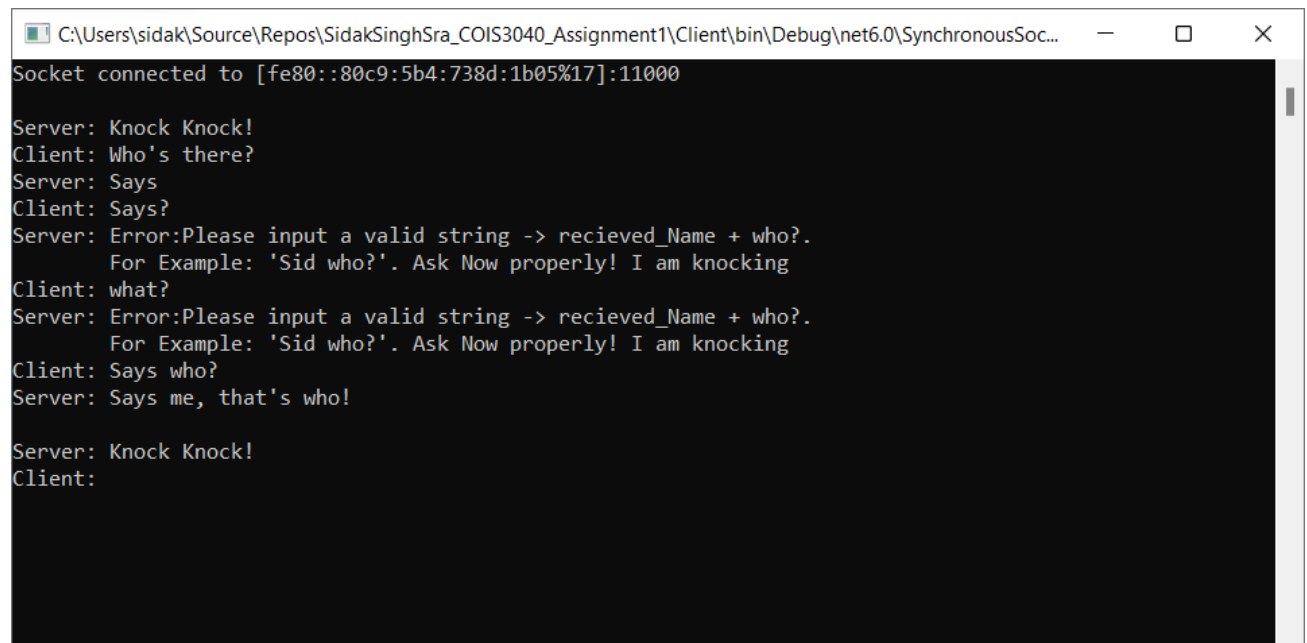
Server: Knock Knock!
Client: _
```

## Test 5

**Description:** After receiving a name message from the Server, the Client enters an incorrect input in this test. When a client enters an incorrect string, the server tells him what he should have stated and prompts him to enter a valid string again. Until the client enters a legitimate input, the server will repeatedly request that he do so.

**Purpose:** To see what happens if the user repeatedly enters an incorrect string and then enters a valid string, would the server deliver a joke and then do Knock Knock?

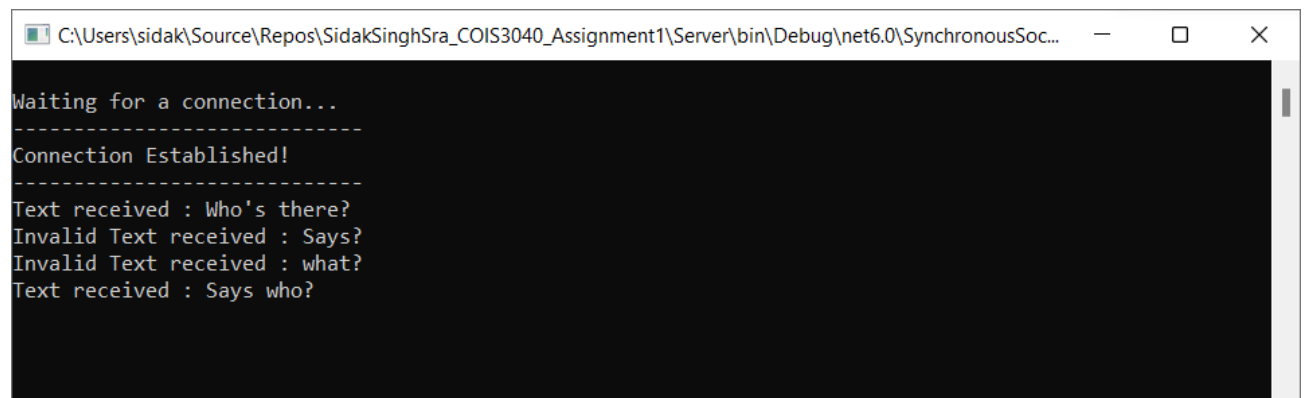
### Client Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client: Who's there?
Server: Says
Client: Says?
Server: Error:Please input a valid string -> recieved_Name + who?.
        For Example: 'Sid who?'. Ask Now properly! I am knocking
Client: what?
Server: Error:Please input a valid string -> recieved_Name + who?.
        For Example: 'Sid who?'. Ask Now properly! I am knocking
Client: Says who?
Server: Says me, that's who!

Server: Knock Knock!
Client:
```

### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Invalid Text received : Says?
Invalid Text received : what?
Text received : Says who?
```

## Test 6

**Description: Availability** - In this test, the Client inputs the 'quit' message, and the Client terminal application is terminated once the 'quit' message is entered. This message is forwarded to the server, which publishes it on its Server Application and then waits for a new connection to be created. This is a non-functional availability criterion that allows the client to connect to the server anytime they want without running the server application again.

**Purpose:** To see what happens if the client types 'quit' and then sends that message to the server. Will the client application be terminated?

### Client Application:

```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000

Server: Knock Knock!
Client: Who's there?
Server: Says
Client: Says?
Server: Error:Please input a valid string -> recieved_Name + who?.
        For Example: 'Sid who?'. Ask Now properly! I am knocking
Client: what?
Server: Error:Please input a valid string -> recieved_Name + who?.
        For Example: 'Sid who?'. Ask Now properly! I am knocking
Client: Says who?
Server: Says me, that's who!

Server: Knock Knock!
Client: Who's there?
Server: To
Client: quit
```

### Server Application:

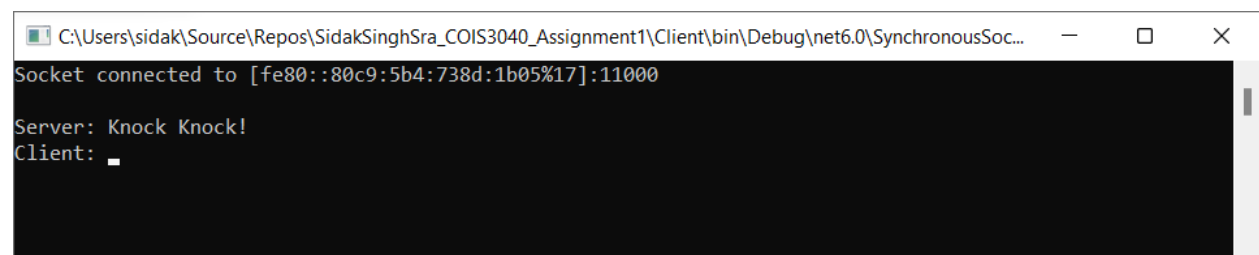
```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Invalid Text received : Says?
Invalid Text received : what?
Text received : Says who?
Text received : Who's there?
Text received : quit

Waiting for a connection...
-
```

### Test 7

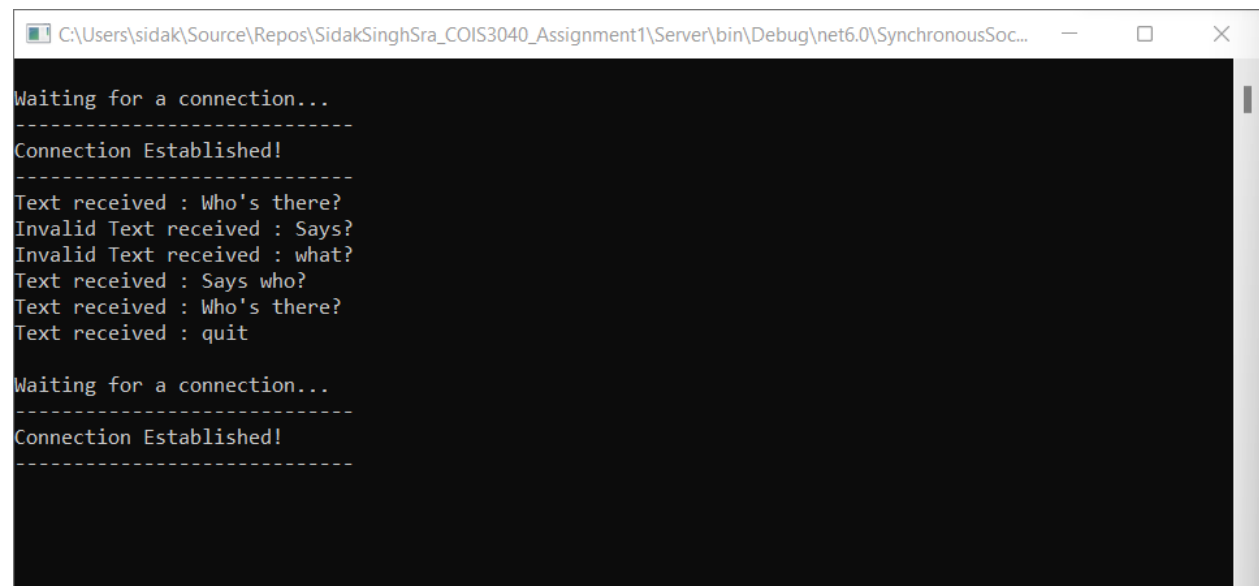
**Description:** In this test, we see if the connection is restored after re-running the client executable file after the client exited the application by entering 'quit' (Continue from test 6). In this test, we can observe that after restarting the client program the server instantly connects to the socket and sends a message to the client application - Knock Knock!! This makes the availability technique more effective.

#### Client Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client: _
```

#### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Invalid Text received : Says?
Invalid Text received : what?
Text received : Says who?
Text received : Who's there?
Text received : quit

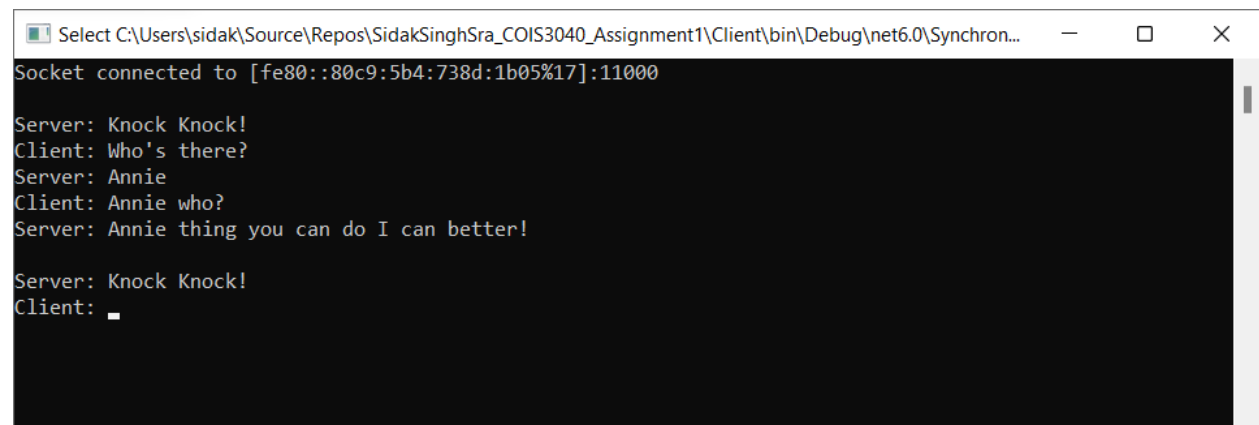
Waiting for a connection...
-----
Connection Established!
-----
```

## Test 8

**Description:** In this test, we enter valid values and interact with the Server again successfully.

**Purpose:** To see if the client program is still running successfully when the client reconnects to the server socket.

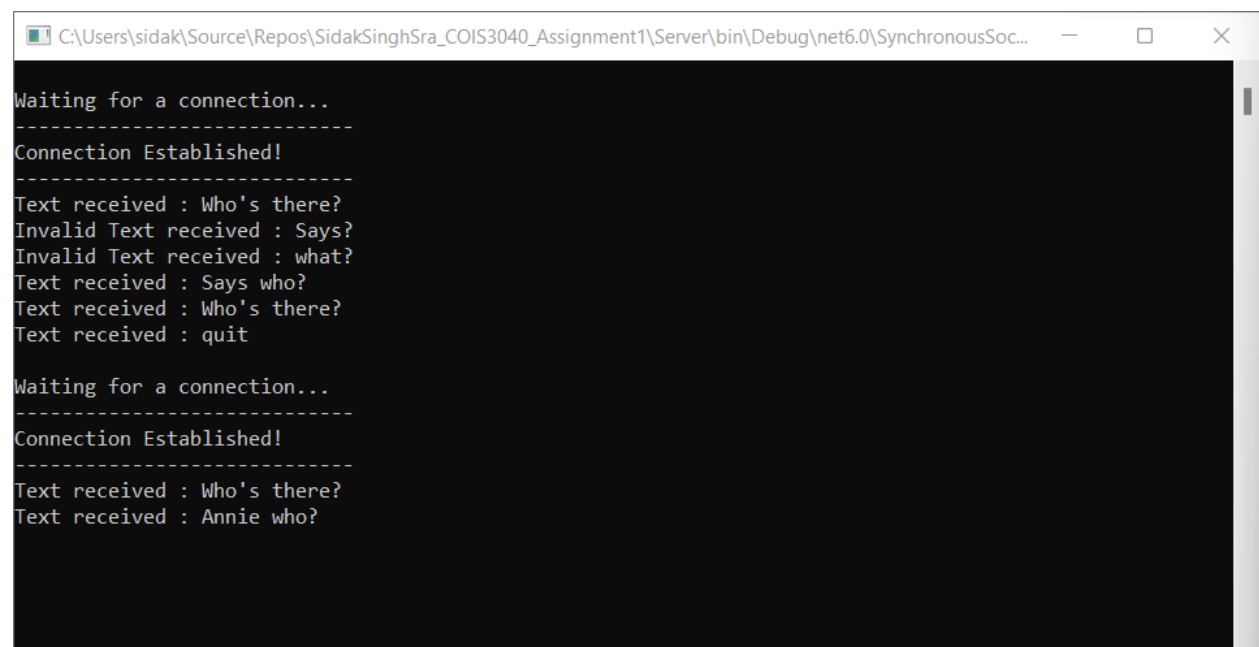
### Client Application:



```
Select C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\Synchron...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000
Server: Knock Knock!
Client: Who's there?
Server: Annie
Client: Annie who?
Server: Annie thing you can do I can better!

Server: Knock Knock!
Client: _
```

### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Invalid Text received : Says?
Invalid Text received : what?
Text received : Says who?
Text received : Who's there?
Text received : quit

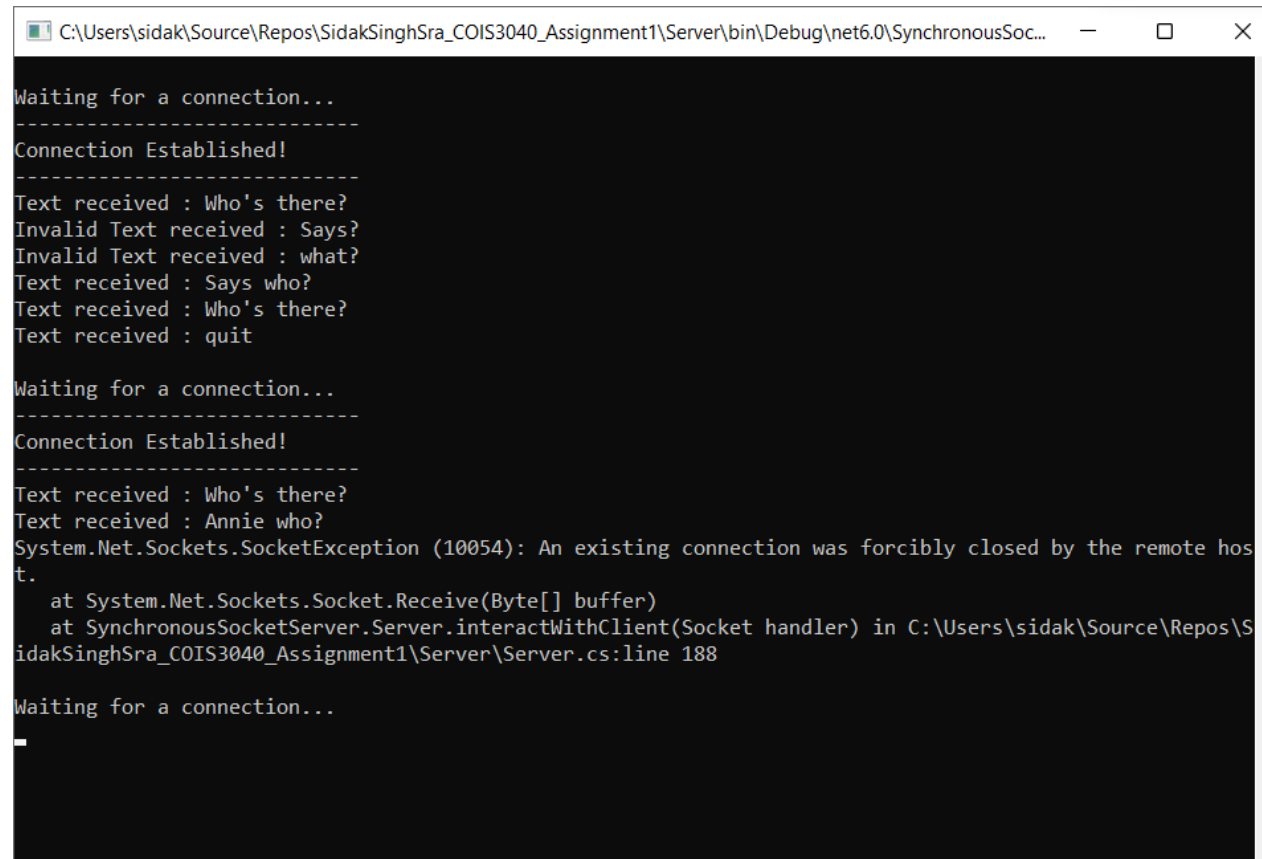
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Text received : Annie who?
```



## Test 9

**Description:** In this case, we shut the client program directly rather than sending the message 'quit'. We discovered that Server would catch a Socket Exception and print the message to its terminal. It then waits for the connection to be established once more. This means that no matter what the client does, our server will not crash.

### Server Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Server\bin\Debug\net6.0\SynchronousSoc...
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Invalid Text received : Says?
Invalid Text received : what?
Text received : Says who?
Text received : Who's there?
Text received : quit

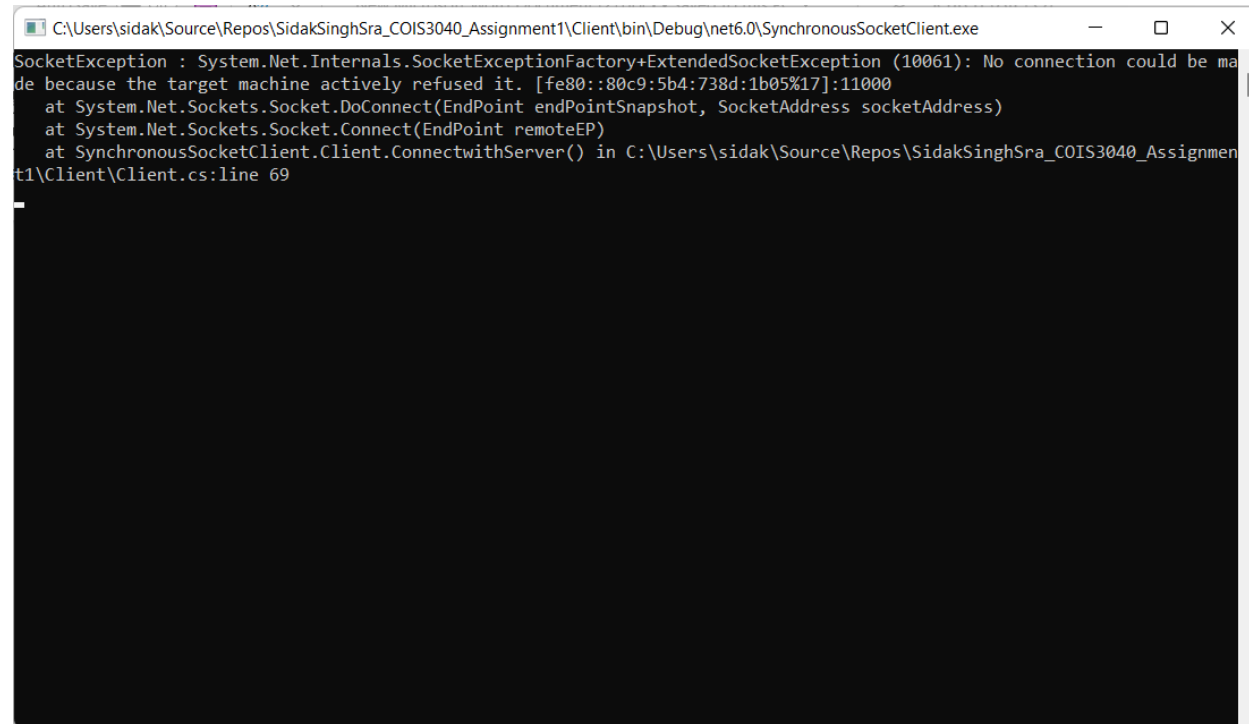
Waiting for a connection...
-----
Connection Established!
-----
Text received : Who's there?
Text received : Annie who?
System.Net.Sockets.SocketException (10054): An existing connection was forcibly closed by the remote hos
t.
   at System.Net.Sockets.Socket.Receive(Byte[] buffer)
   at SynchronousSocketServer.Server.interactWithClient(Socket handler) in C:\Users\sidak\Source\Repos\S
idakSinghSra_COIS3040_Assignment1\Server\Server.cs:line 188

Waiting for a connection...
-
```

## Test 10

**Description:** In this test, we start our client program without running the server, and the client application catches a Socket Exception and produces this message.

### Client Application:



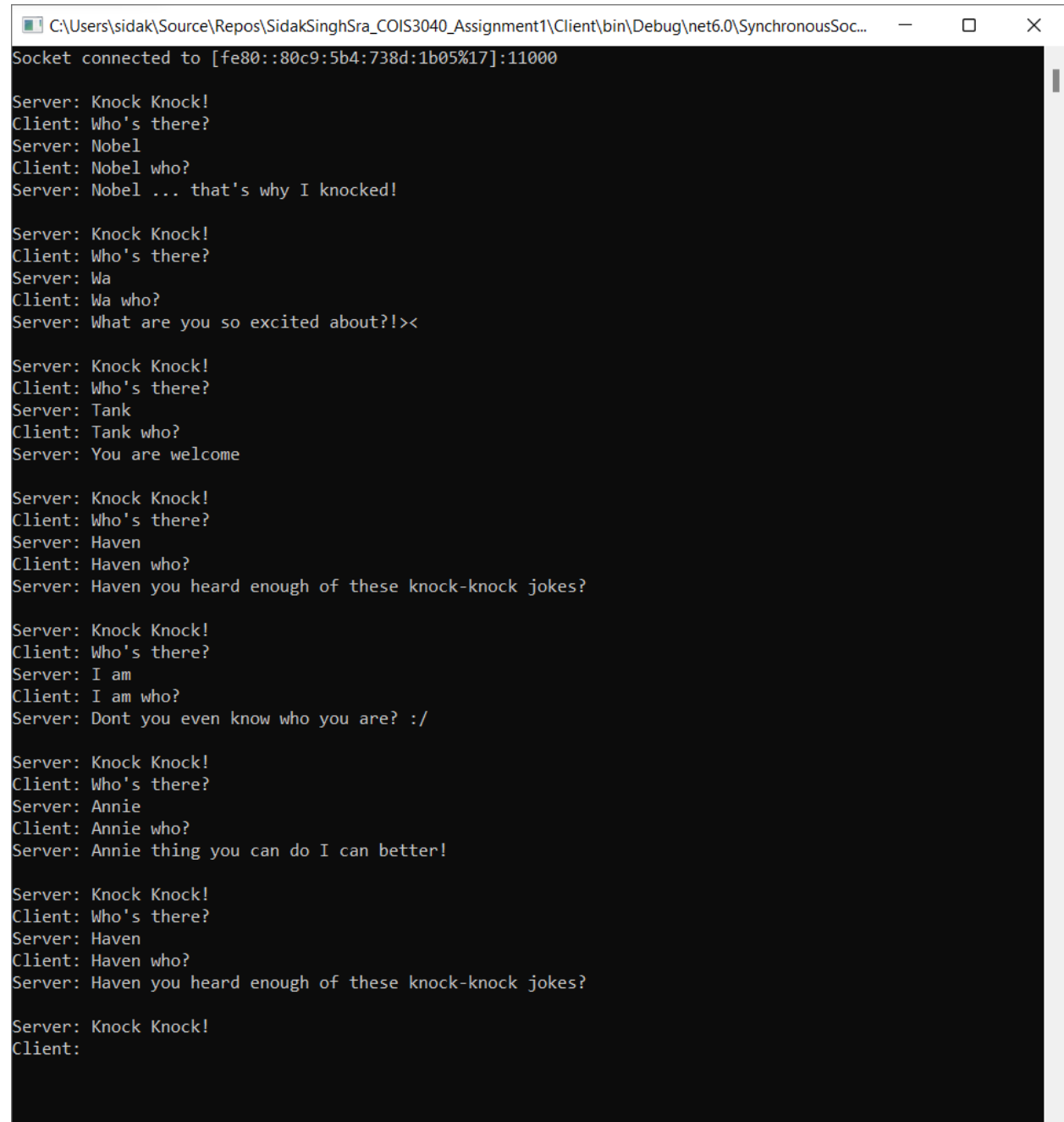
```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSocketClient.exe
SocketException : System.Net.Internals.SocketExceptionFactory+ExtendedSocketException (10061): No connection could be made because the target machine actively refused it. [fe80::80c9:5b4:738d:1b05%17]:11000
    at System.Net.Sockets.Socket.DoConnect(EndPoint endPointSnapshot, SocketAddress socketAddress)
    at System.Net.Sockets.Socket.Connect(EndPoint remoteEP)
    at SynchronousSocketClient.Client.ConnectwithServer() in C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\Client.cs:line 69
```

## Test 11

**Description:** In this case, the client supplied a correct message for all of the input, and Server repeatedly told the client Knock-Knock Jokes and sent the knock-knock message.

**Purpose:** To see if the Server tells at least five distinct knock knock jokes.

### Client Application:



```
C:\Users\sidak\Source\Repos\SidakSinghSra_COIS3040_Assignment1\Client\bin\Debug\net6.0\SynchronousSoc...
Socket connected to [fe80::80c9:5b4:738d:1b05%17]:11000

Server: Knock Knock!
Client: Who's there?
Server: Nobel
Client: Nobel who?
Server: Nobel ... that's why I knocked!

Server: Knock Knock!
Client: Who's there?
Server: Wa
Client: Wa who?
Server: What are you so excited about?!><

Server: Knock Knock!
Client: Who's there?
Server: Tank
Client: Tank who?
Server: You are welcome

Server: Knock Knock!
Client: Who's there?
Server: Haven
Client: Haven who?
Server: Haven you heard enough of these knock-knock jokes?

Server: Knock Knock!
Client: Who's there?
Server: I am
Client: I am who?
Server: Dont you even know who you are? :/

Server: Knock Knock!
Client: Who's there?
Server: Annie
Client: Annie who?
Server: Annie thing you can do I can better!

Server: Knock Knock!
Client: Who's there?
Server: Haven
Client: Haven who?
Server: Haven you heard enough of these knock-knock jokes?

Server: Knock Knock!
Client:
```

## Server Application:

 C:\Users\sidak\Source\Repos\SidakSinghSra\_COIS30

Waiting for a connection...

-----  
Connection Established!

-----  
Text received : Who's there?

Text received : Nobel who?

Text received : Who's there?

Text received : Wa who?

Text received : Who's there?

Text received : Tank who?

Text received : Who's there?

Text received : Haven who?

Text received : Who's there?

Text received : I am who?

Text received : Who's there?

Text received : Annie who?

Text received : Who's there?

Text received : Haven who?