Assignment 1

B00152842

# Code

## s.py

Imports

from jsonrpcserver import Success, method, serve

from jsonrpcclient import request, parse

import requests

import base64

import sys

import os

import subprocess

Start server

if len(sys.argv) > 1:

    serverNumber = sys.argv[1]

else:

    print("sorry you forgot to add a server number... shutting down")

    quit()

myFriends = list()

Server request functions

***To minimise code repetition I wrote methods call functions with/without methods, so each request in code is only one line***

def getFunctionWithoutParam(portToCall, function):

    try:

        response = requests.post(f"http://localhost:{portToCall}/", json=request(function))

        parsed = parse(response.json())

        print(parsed.result)

        return parsed.result

    except:

        print(" ----- Are you sure that port and function exist?" + serverNumber + function)

        return False

def getFunctionWithParam(portToCall, function, params):

    try:

        response = requests.post(f"http://localhost:{portToCall}/", json=request(function, params=params))

        parsed = parse(response.json())

        print(parsed.result)

        return parsed.result

    except:

        print(" ----- Are you sure that port and function exist?" + serverNumber + function)

        return False

Ping

@method

def ping():

    return Success("pong")

@method

def whoareyou():

    return Success(f"Server Number: {serverNumber} Port Number: 500{serverNumber}")

Make folder

@method

def make\_folder(folder\_name):

    try:

        os.mkdir(folder\_name)

        return Success(f"Folder {folder\_name} created")

    except:

        return Success(f"Folder {folder\_name} already exists")

Delete folder

@method

def delete\_folder(folder\_name):

    try:

        os.rmdir(folder\_name)

        return Success(f"Folder {folder\_name} deleted")

    except:

        return Success(f"Folder {folder\_name} does not exist")

Get version

@method

def get\_version():

    result = subprocess.run(["py", "--version"], capture\_output=True, text=True)

    finalResultOutout = result.stdout

    return Success(f"Python version: {finalResultOutout}")

Search

@method

def search(file\_name):

    if os.path.isfile(file\_name):

        return Success(f"File {file\_name} exists")

    else:

        return Success(f"File {file\_name} does not exist")

Add friend (assists online method)

@method

def add\_friend(friend):

    myFriends.append(friend)

    return Success(f"Friend {friend} added")

Online

@method

def online(friends):

    myFriends.extend(friends)

    sent\_from = friends[0]

    for friend in myFriends:

        if friend != sent\_from:

            getFunctionWithParam("500" + friend, "add\_friend", {"friend": serverNumber})

    return Success(f"Online signal sent")

Offline

@method

def offline():

    for friend in myFriends:

        getFunctionWithParam("500" + friend, "remove\_friend", {"friend": serverNumber})

    return Success(f"Offline signal sent")

Start up

@method

def startup(server\_num):

    if server\_num in myFriends:

        return Success(f"Server {server\_num} already running")

    os.system(f'start /b py s.py {server\_num} &')

    getFunctionWithParam("500" + server\_num, "online", {"friends": [serverNumber] + myFriends})

    myFriends.append(server\_num)

    return Success(f"Server {server\_num} started")

Shut down

@method

def shutdown():

    offline()

    quit() # try catch caught this as an error, so it didnt run, meaning server kept running

    return Success(f"Server {serverNumber} shutdown")

Get version

@method

def get\_friends():

    return Success(f"Server {serverNumber} has friends: {myFriends}")

Remove friend (assists offline method)

@method

def remove\_friend(friend):

    print(friend + "  " + serverNumber)

    myFriends.remove(friend)

    return Success(f"Friend {friend} removed")

Heart beat

@method

def heart\_beat():

    responded = list()

    for friend in myFriends:

        response = getFunctionWithoutParam("500" + friend, "ping")

        if response:

            responded.append(friend)

    return Success(f"Sent from {serverNumber} Response from: {responded}")

Pass message

@method

def pass\_msg(target, servers):

    servers.remove(serverNumber)

    if serverNumber == target:

        return Success("Message received")

    elif len(servers) != 0:

        print("Message passed to " + servers[0])

        response = getFunctionWithParam("500" + servers[0], "pass\_msg", {"target": target, "servers": servers})

        if response == "Message received":

            return Success(response)

    return Success("Server not found")

if \_\_name\_\_ == "\_\_main\_\_":

    print(f"server number {serverNumber} running.....")

    portNumber = '500' + serverNumber

    serve(port=int(portNumber))

## c.py

Imports

from jsonrpcclient import request, parse, Ok

import requests

import base64

Methods to get user input and server number (minimise code repetition)

def getInput(message):

    print(message)

    return input()

def getServer():

    server = getInput(f"Please enter the server number {serverList}: ")

    try:

        if server not in serverList:

           print(" ----- Server not found")

           return getServer()

        return int(server) + 5000

    except:

        print(" ----- Please enter a valid number")

        return getServer()

Server request functions

***To minimise code repetition I wrote methods call functions with/without methods, so each request in code is only one line***

def getFunctionWithoutParam(portToCall, function):

    try:

        response = requests.post(f"http://localhost:{portToCall}/", json=request(function))

        parsed = parse(response.json())

        print(parsed.result)

        return parsed.result

    except:

        print(" ----- Are you sure that port and function exist?")

        return False

def getFunctionWithParam(portToCall, function, params):

    try:

        response = requests.post(f"http://localhost:{portToCall}/", json=request(function, params=params))

        parsed = parse(response.json())

        print(parsed.result)

        return parsed.result

    except:

        print(" ----- Are you sure that port and function exist?")

        return False

serverList = list()

Functions get needed parameters from user and get request using above methods

def ping():

    portToCall = getServer()

    getFunctionWithoutParam(portToCall, "ping")

def make\_folder():

    portToCall = getServer()

    folder\_name = getInput("Please enter the folder name: ")

    getFunctionWithParam(portToCall, "make\_folder", {"folder\_name": folder\_name})

def delete\_folder():

    portToCall = getServer()

    folder\_name = getInput("Please enter the folder name: ")

    getFunctionWithParam(portToCall, "delete\_folder", {"folder\_name": folder\_name})

def whoareyou():

    portToCall = getServer()

    getFunctionWithoutParam(portToCall, "whoareyou")

def get\_version():

    portToCall = getServer()

    getFunctionWithoutParam(portToCall, "get\_version")

def search():

    portToCall = getServer()

    file\_name = getInput("Please enter the file name: ")

    getFunctionWithParam(portToCall, "search", {"file\_name": file\_name})

def startup():

    server\_num = getInput("Enter the sever number you want to start")

    started = getFunctionWithParam("500" + serverList[0], "startup", {"server\_num": server\_num})

    if started == "Server " + server\_num + " started":

        serverList.append(server\_num)

def shutdown():

    server\_num = getServer()

    getFunctionWithoutParam(server\_num, "shutdown")

    serverList.remove(str(server\_num - 5000))

def get\_friends():

    portToCall = getServer()

    getFunctionWithoutParam(portToCall, "get\_friends")

def heart\_beat():

    getFunctionWithoutParam("500" + serverList[0], "heart\_beat")

def pass\_msg():

    target = getInput("Please enter the server you want to pass the message too: ")

    getFunctionWithParam("500" + serverList[0], "pass\_msg", {"target": target, "servers": serverList})

Main method to process user inputs and call relevant methods

if \_\_name\_\_ == "\_\_main\_\_":

    print("Welcome!")

    serverList.append("1")

    while True:

        print("please type a menu option")

        print("1. make folder")

        print("2. delete folder")

        print("3. who are you")

        print("4. get version")

        print("5. search")

        print("6. startup")

        print("7. shutdown")

        print("8. get friends")

        print("9. heart beat")

        print("10. pass message")

        print("11. remove server")

        print("12. ping")

        option = int(input())

        if option == 1:

            make\_folder()

        elif option == 2:

            delete\_folder()

        elif option == 3:

            whoareyou()

        elif option == 4:

            get\_version()

        elif option == 5:

            search()

        elif option == 6:

            startup()

        elif option == 7:

            shutdown()

        elif option == 8:

            get\_friends()

        elif option == 9:

            heart\_beat()

        elif option == 10:

            pass\_msg()

        elif option == 11:

            server\_num = getServer()

            serverList.remove(server\_num)

        elif option == 12:

            ping()

        else:

            print(" ----- Please enter a valid option")

# Scripts

## run\_script.bat

@echo off

start /B py s.py 1

py c.py

pause

## test\_script.bat

@echo off

start /B py s.py 1

py test.py > test.log 2>&1

pause

# Unit tests

## Test code

import unittest, requests

from jsonrpcclient import request, parse, Ok

from c import getFunctionWithoutParam, getFunctionWithParam

import os

"""

-------------------------- RUN USING test\_script.bat --------------------------------------------------------

"""

class MyTests(unittest.TestCase):

    def test\_01\_startup(self):

        self.assertEqual(getFunctionWithParam(5001, "startup", {"server\_num": "2"}), "Server 2 started")

        self.assertEqual(getFunctionWithParam(5001, "startup", {"server\_num": "3"}), "Server 3 started")

        self.assertEqual(getFunctionWithParam(5001, "startup", {"server\_num": "2"}), "Server 2 already running")

        self.assertEqual(getFunctionWithParam(5001, "startup", {"server\_num": "3"}), "Server 3 already running")

    def test\_02\_ping(self):

        # should return pong (work)

        self.assertEqual(getFunctionWithoutParam(5001, "ping"), "pong")

        self.assertEqual(getFunctionWithoutParam(5002, "ping"), "pong")

        self.assertEqual(getFunctionWithoutParam(5003, "ping"), "pong")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithoutParam(5005, "ping"))

        self.assertFalse(getFunctionWithoutParam(3000, "ping"))

    def test\_03\_make\_folder(self):

        # make folder

        # should return folder created (work)

        self.assertEqual(getFunctionWithParam(5001, "make\_folder", {"folder\_name": "test\_folder1"}), "Folder test\_folder1 created")

        self.assertEqual(getFunctionWithParam(5002, "make\_folder", {"folder\_name": "test\_folder2"}), "Folder test\_folder2 created")

        self.assertEqual(getFunctionWithParam(5003, "make\_folder", {"folder\_name": "test\_folder3"}), "Folder test\_folder3 created")

        # should return folder already exists (work)

        self.assertEqual(getFunctionWithParam(5001, "make\_folder", {"folder\_name": "test\_folder1"}), "Folder test\_folder1 already exists")

        self.assertEqual(getFunctionWithParam(5002, "make\_folder", {"folder\_name": "test\_folder2"}), "Folder test\_folder2 already exists")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithParam(5005, "make\_folder", {"folder\_name": "test\_folder"}))

        self.assertFalse(getFunctionWithParam(3000, "make\_folder", {"folder\_name": "test\_folder"}))

    def test\_04\_delete\_folder(self):

        # delete folder

        # should return folder deleted (work)

        self.assertEqual(getFunctionWithParam(5001, "delete\_folder", {"folder\_name": "test\_folder1"}), "Folder test\_folder1 deleted")

        self.assertEqual(getFunctionWithParam(5002, "delete\_folder", {"folder\_name": "test\_folder2"}), "Folder test\_folder2 deleted")

        self.assertEqual(getFunctionWithParam(5003, "delete\_folder", {"folder\_name": "test\_folder3"}), "Folder test\_folder3 deleted")

        # should return folder does not exist (work)

        self.assertEqual(getFunctionWithParam(5001, "delete\_folder", {"folder\_name": "test\_folder1"}), "Folder test\_folder1 does not exist")

        self.assertEqual(getFunctionWithParam(5002, "delete\_folder", {"folder\_name": "test\_folder2"}), "Folder test\_folder2 does not exist")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithParam(5005, "delete\_folder", {"folder\_name": "test\_folder"}))

        self.assertFalse(getFunctionWithParam(3000, "delete\_folder", {"folder\_name": "test\_folder"}))

    def test\_05\_whoareyou(self):

        # should return server number and port number (work)

        self.assertEqual(getFunctionWithoutParam(5001, "whoareyou"), "Server Number: 1 Port Number: 5001")

        self.assertEqual(getFunctionWithoutParam(5002, "whoareyou"), "Server Number: 2 Port Number: 5002")

        self.assertEqual(getFunctionWithoutParam(5003, "whoareyou"), "Server Number: 3 Port Number: 5003")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithoutParam(5005, "whoareyou"))

        self.assertFalse(getFunctionWithoutParam(3000, "whoareyou"))

    def test\_06\_get\_version(self):

        # should return server version (work)

        self.assertTrue(getFunctionWithoutParam(5001, "get\_version"))

        self.assertTrue(getFunctionWithoutParam(5002, "get\_version"))

        self.assertTrue(getFunctionWithoutParam(5003, "get\_version"))

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithoutParam(5005, "get\_version"))

        self.assertFalse(getFunctionWithoutParam(3000, "get\_version"))

    def test\_07\_search(self):

        # should return file found (work)

        self.assertEqual(getFunctionWithParam(5001, "search", {"file\_name": "c.py"}), "File c.py exists")

        self.assertEqual(getFunctionWithParam(5002, "search", {"file\_name": "s.py"}), "File s.py exists")

        self.assertEqual(getFunctionWithParam(5003, "search", {"file\_name": "run\_script.bat"}), "File run\_script.bat exists")

        # should return file not found (work)

        self.assertEqual(getFunctionWithParam(5001, "search", {"file\_name": "test\_file1"}), "File test\_file1 does not exist")

        self.assertEqual(getFunctionWithParam(5002, "search", {"file\_name": "test\_file2"}), "File test\_file2 does not exist")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithParam(5005, "search", {"file\_name": "test\_file"}))

        self.assertFalse(getFunctionWithParam(3000, "search", {"file\_name": "test\_file"}))

    def test\_08\_getFriends(self):

        # should return server number and friends (work)

        self.assertEqual(getFunctionWithoutParam(5001, "get\_friends"), "Server 1 has friends: ['2', '3']")

        self.assertEqual(getFunctionWithoutParam(5002, "get\_friends"), "Server 2 has friends: ['1', '3']")

        self.assertEqual(getFunctionWithoutParam(5003, "get\_friends"), "Server 3 has friends: ['1', '2']")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithoutParam(5005, "get\_friends"))

        self.assertFalse(getFunctionWithoutParam(3000, "get\_friends"))

    def test\_09\_heartbeat(self):

        # should return up servers (work)

        self.assertEqual(getFunctionWithoutParam(5001, "heart\_beat"), "Sent from 1 Response from: ['2', '3']")

        self.assertEqual(getFunctionWithoutParam(5002, "heart\_beat"), "Sent from 2 Response from: ['1', '3']")

        self.assertEqual(getFunctionWithoutParam(5003, "heart\_beat"), "Sent from 3 Response from: ['1', '2']")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithoutParam(5005, "heart\_beat"))

        self.assertFalse(getFunctionWithoutParam(3000, "heart\_beat"))

    def test\_10\_pass\_msg(self):

        # should return message received (work)

        self.assertEqual(getFunctionWithParam(5001, "pass\_msg", {"target": "1", "servers": ["1", "2", "3"]}), "Message received")

        self.assertEqual(getFunctionWithParam(5001, "pass\_msg", {"target": "2", "servers": ["1", "2", "3"]}), "Message received")

        self.assertEqual(getFunctionWithParam(5001, "pass\_msg", {"target": "3", "servers": ["1", "2", "3"]}), "Message received")

        # should return server not found (work)

        self.assertEqual(getFunctionWithParam(5001, "pass\_msg", {"target": "4", "servers": ["1", "2", "3"]}), "Server not found")

        self.assertEqual(getFunctionWithParam(5001, "pass\_msg", {"target": "5", "servers": ["1", "2", "3"]}), "Server not found")

        self.assertEqual(getFunctionWithParam(5001, "pass\_msg", {"target": "3000", "servers": ["1", "2", "3"]}), "Server not found")

        # should return false (not work as servers arent up)

        self.assertFalse(getFunctionWithParam(5005, "pass\_msg", {"target": "1", "servers": ["1", "2", "3"]}))

        self.assertFalse(getFunctionWithParam(3000, "pass\_msg", {"target": "1", "servers": ["1", "2", "3"]}))

    def test\_11\_shutdown(self):

        # should return server shutdown (work)

        self.assertFalse(getFunctionWithoutParam(5001, "shutdown"))

        self.assertFalse(getFunctionWithoutParam(5002, "shutdown"))

        self.assertFalse(getFunctionWithoutParam(5003, "shutdown"))

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main()

## Failed output

..........F

======================================================================

**FAIL**: test\_11\_shutdown (\_\_main\_\_.MyTests.test\_11\_shutdown)

----------------------------------------------------------------------

Traceback (most recent call last):

  File "C:\Users\davyt\OneDrive - Technological University Dublin\College\Sem6\Web Dist\Assignment\test.py", line 134, in test\_11\_shutdown

    self.assertEqual(getFunctionWithoutParam(5001, "shutdown"), "Server 1 shutdown")

AssertionError: False != 'Server 1 shutdown'

----------------------------------------------------------------------

Ran 11 tests in 199.106s

FAILED (failures=1)

Server 2 started

Server 3 started

Server 2 already running

Server 3 already running

pong

pong

pong

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Folder test\_folder1 created

Folder test\_folder2 created

Folder test\_folder3 created

Folder test\_folder1 already exists

Folder test\_folder2 already exists

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Folder test\_folder1 deleted

Folder test\_folder2 deleted

Folder test\_folder3 deleted

Folder test\_folder1 does not exist

Folder test\_folder2 does not exist

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Server Number: 1 Port Number: 5001

Server Number: 2 Port Number: 5002

Server Number: 3 Port Number: 5003

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Python version: Python 3.11.9

Python version: Python 3.11.9

Python version: Python 3.11.9

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

File c.py exists

File s.py exists

File run\_script.bat exists

File test\_file1 does not exist

File test\_file2 does not exist

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Server 1 has friends: ['2', '3']

Server 2 has friends: ['1', '3']

Server 3 has friends: ['1', '2']

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Sent from 1 Response from: ['2', '3']

Sent from 2 Response from: ['1', '3']

Sent from 3 Response from: ['1', '2']

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Message received

Message received

Message received

Server not found

Server not found

Server not found

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

## Successfully output

...........

----------------------------------------------------------------------

Ran 11 tests in 205.155s

OK

Server 2 started

Server 3 started

Server 2 already running

Server 3 already running

pong

pong

pong

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Folder test\_folder1 created

Folder test\_folder2 created

Folder test\_folder3 created

Folder test\_folder1 already exists

Folder test\_folder2 already exists

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Folder test\_folder1 deleted

Folder test\_folder2 deleted

Folder test\_folder3 deleted

Folder test\_folder1 does not exist

Folder test\_folder2 does not exist

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Server Number: 1 Port Number: 5001

Server Number: 2 Port Number: 5002

Server Number: 3 Port Number: 5003

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Python version: Python 3.11.9

Python version: Python 3.11.9

Python version: Python 3.11.9

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

File c.py exists

File s.py exists

File run\_script.bat exists

File test\_file1 does not exist

File test\_file2 does not exist

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Server 1 has friends: ['2', '3']

Server 2 has friends: ['1', '3']

Server 3 has friends: ['1', '2']

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Sent from 1 Response from: ['2', '3']

Sent from 2 Response from: ['1', '3']

Sent from 3 Response from: ['1', '2']

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

Message received

Message received

Message received

Server not found

Server not found

Server not found

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

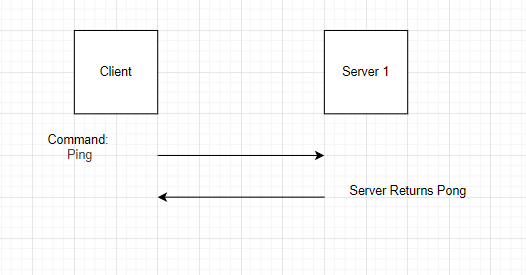
 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

 ----- Are you sure that port and function exist?

# Diagrams

## Ping



## Who are you

A diagram of a server number

AI-generated content may be incorrect.

## Make folder

A screenshot of a computer screen

AI-generated content may be incorrect.

## Delete folder

A diagram of a server

AI-generated content may be incorrect.

## Get version

A diagram of a computer program

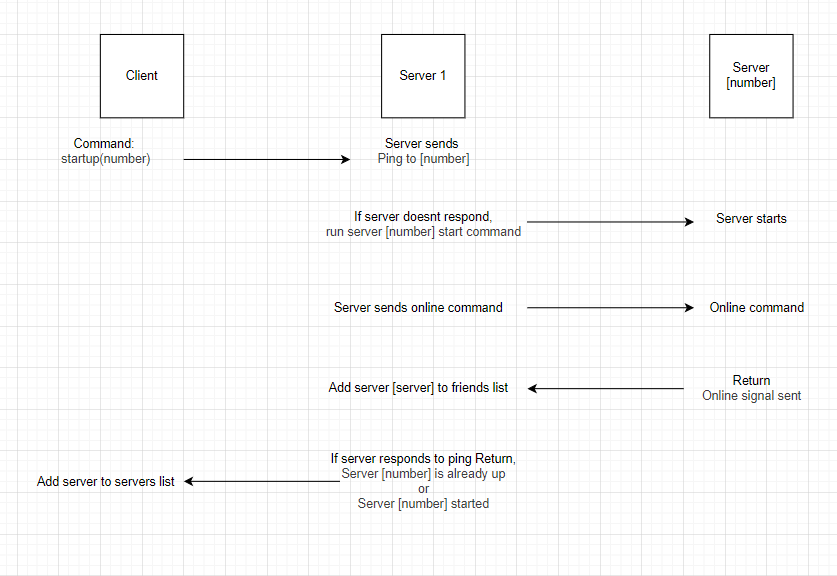
AI-generated content may be incorrect.

## Search

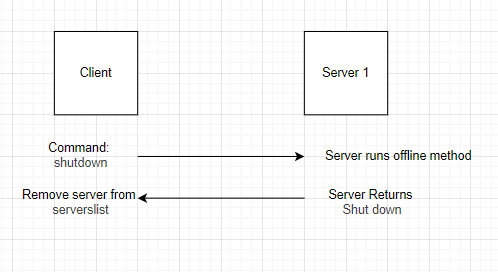
A diagram of a server

AI-generated content may be incorrect.

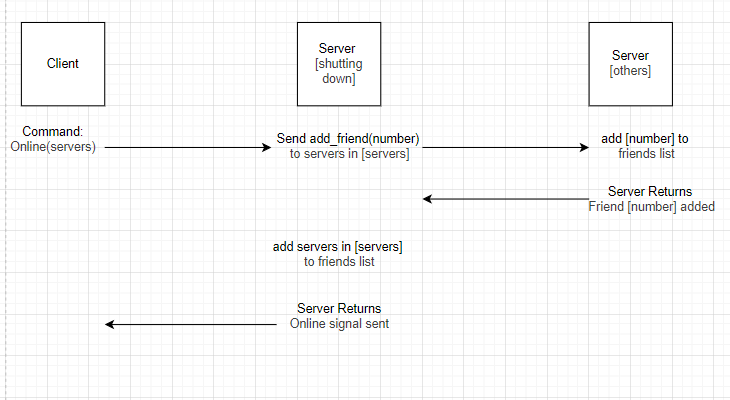
## Start up



## Shut down



## Online



## Offline

A diagram of a server

AI-generated content may be incorrect.

## Heartbeat

A diagram of a server

AI-generated content may be incorrect.

## Pass message

A diagram with text and arrows

AI-generated content may be incorrect.