Test report

This report is divided into two sections: automated test cases and manual test cases

Automated test cases for Server Responses:

To run the automated test cases:

1. Open two terminal windows in the submission folder
2. In one terminal, run the server using: *python3 server.py ./config.json*
3. In the other terminal, run: *./testing.sh*
4. Once the teets are complete, stop the server using *CTRL + C*

The results will display whether each test case passed or failed, along with the expected output and, if failed, the received output.

Notes on test coverage:

The automated tests cover all possible server protocols with a single client. As a result the excluded protocols are:

- Successful REGISTER (to avoid creating multiple accounts),
- JOIN an existing room (requires multiple clients),
- Successful CREATE, JOIN, and in-room commands (PLACE, FORFEIT, BEGIN, INPROGRESS, GAMEEND).

Manual test cases:

Manual testing was conducted to handle the remaining scenarios that were excluded in the automated testing. Below are some examples.
For the few server response screenshots, received data and current rooms are shown while out of room commands are received to avoid redundancy.
For the third test case onwards, screenshots will omit repetitive protocols, such as consecutive placement actions leading to a draw, and not include server response screenshot, to avoid redundancy.

Test 1 - Registering new user

Verify if a new user can register and log in successfully.

Steps:

- Input the REGISTER command
- Enter a valid username and password
- Input the LOGIN command with same credentials

Client Input:

```
xurge@xurge-VM-linuxmint:~/Desktop/uni/2024/sem-2/info-1112/ass-2/submission$ python3 client.py lo
calhost 8002
REGISTER
Enter username: yet another user
Enter password: password
Successfully created user account yet another user
QUIT
Exiting
```

Server Response:

```
xurge@xurge-VM-linuxmint:~/Desktop/uni/2024/sem-2/info-1112/ass-2/submission$ python3 server.py ./conf
ig.json
Client connected: ('127.0.0.1', 47162)
Received data: ['REGISTER', 'yet another user', 'password']
{}
Received data: ['']
Client disconnected: ('127.0.0.1', 47162)
^C
Closing server...
```

Test 2 - Game Ending in a Win with Viewer Joining Mid-Game

Check if a game can conclude with a winner, with a third client joining as a viewer mid-game.

Steps:

- Client 1:
    - Login, and create room
    - Wait for *Client 2* to join, begin placing
    - Wait for *Client 3* to join
    - Win the game
- Client 2:
    - Login, join the room (player), and place
    - Wait for *Client 3* to join
    - Allow *Client 1* to win
- Client 3:
    - Login, join the room (viewer), and spectate

Client 1 Input:

```
xurge@xurge-VM-linuxmint:~/Desktop/uni/2024/sem-2/info-1112/ass-2/submission$ python3 client.py lo
calhost 8002
LOGIN
Enter username: user
Enter password: password
Welcome user
CREATE
Enter room name you want to create: room
Successfully created room room
Waiting for second player
Match between user and another user will commence, it is currently user's turn

-------------
|   |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Your turn:
PLACE
Column: 0
Row: 0

-------------
| X |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Wait for your turn

-------------
| X | O |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------
```

```
Your turn:
PLACE
Column: 0
Row: 1

-------------
| X | O |   |
-------------
| X |   |   |
-------------
|   |   |   |
-------------

Wait for your turn

-------------
| X | O |   |
-------------
| X | O |   |
-------------
|   |   |   |
-------------

Your turn:
PLACE
Column: 0
Row: 2

-------------
| X | O |   |
-------------
| X | O |   |
-------------
| X |   |   |
-------------

Congratulations, you won!
QUIT
Exiting
```

Client 2 Input:

```
xurge@xurge-VM-linuxmint:~/Desktop/uni/2024/sem-2/info-1112/ass-2/submission$ python3 client.py loc
alhost 8002
LOGIN
Enter username: another user
Enter password: password
Welcome another user
JOIN
Enter room name you want to join: room
Do you want to have a room list as player or viewer? (Player/Viewer) player
Successfully joined room room as a PLAYER
Match between user and another user will commence, it is currently user's turn

-------------
|   |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Wait for your turn

-------------
| X |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Your turn:
PLACE
Column: 1
Row: 0

-------------
| X | O |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

Wait for your turn
```

```
Wait for your turn

-------------
| X | O |   |
-------------
| X |   |   |
-------------
|   |   |   |
-------------

Your turn:
PLACE
Column: 1
Row: 1

-------------
| X | O |   |
-------------
| X | O |   |
-------------
|   |   |   |
-------------

Wait for your turn

-------------
| X | O |   |
-------------
| X | O |   |
-------------
| X |   |   |
-------------

Sorry you lost. Good luck next time.
QUIT
Exiting
```

Client 3 Input:

```
xurge@xurge-VM-linuxmint:~/Desktop/uni/2024/sem-2/info-1112/ass-2/submission$ python3 client.py localho
st 8002
LOGIN
Enter username: yet another user
Enter password: password
Welcome yet another user
JOIN
Enter room name you want to join: room
Do you want to have a room list as player or viewer? (Player/Viewer) viewer
Successfully joined room room as a VIEWER
Match between user and another user is currently in progress, it is user's turn
user's turn

-------------
| X | O |   |
-------------
| X |   |   |
-------------
|   |   |   |
-------------

another user's turn

-------------
| X | O |   |
-------------
| X | O |   |
-------------
|   |   |   |
-------------

user's turn

-------------
| X | O |   |
-------------
| X | O |   |
-------------
| X |   |   |
-------------

user has won this game
QUIT
Exiting
```

Server Response:

```
xurge@xurge-VM-linuxmint:~/Desktop/uni/2024/sem-2/info-1112/ass-2/submission$ python3 server.py ./conf
ig.json
Client connected: ('127.0.0.1', 46450)
Received data: ['LOGIN', 'user', 'password']
{}
Received data: ['CREATE', 'room']
Client connected: ('127.0.0.1', 38778)
Received data: ['LOGIN', 'another user', 'password']
{'room': {'players': ['user'], 'p_sockets': [<socket.socket fd=4, family=AddressFamily.AF_INET, type=S
ocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8002), raddr=('127.0.0.1', 46450)>], 'viewers': []
, 'v_sockets': [], 'game_state': [[' ', ' ', ' '], [' ', ' ', ' '], [' ', ' ', ' ']], 'board_status':
'000000000', 'turn': 0, 'game_begun': False}}
Received data: ['JOIN', 'room', 'PLAYER']
Client connected: ('127.0.0.1', 33030)
Received data: ['LOGIN', 'yet another user', 'password']
{'room': {'players': ['user', 'another user'], 'p_sockets': [<socket.socket fd=4, family=AddressFamily
.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 8002), raddr=('127.0.0.1', 46450)>
, <socket.socket fd=5, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127
.0.0.1', 8002), raddr=('127.0.0.1', 38778)>], 'viewers': [], 'v_sockets': [], 'game_state': [['X', 'O'
, ' '], [' ', ' ', ' '], [' ', ' ', ' ']], 'board_status': '120000000', 'turn': 2, 'game_begun': True}
}
Received data: ['JOIN', 'room', 'VIEWER']
{}
Received data: ['']
Client disconnected: ('127.0.0.1', 46450)
{}
Received data: ['']
Client disconnected: ('127.0.0.1', 38778)
{}
Received data: ['']
Client disconnected: ('127.0.0.1', 33030)
^C
Closing server...
```

Test 3 - Game Ending in a Draw with Viewer Joining Before Game Begins

Validate that a game can end in a draw with a third client joining as a viewer before the game starts.

Steps:

- Client 1:
    - Login, and create room
    - Wait for *Client 3* to join
    - Now allow *Client 2* to join
    - Begin placing until draw
- Client 2:
    - Login, and wait for *Client 3* to join
    - Join the room (player)
    - Begin placing moves until the game concludes in a draw
- Client 3:
    - Login, join the room (viewer), and spectate entire game

Client 1 Input:

```
Your turn:
PLACE
Column: 0
Row: 1

-------------
| X | O | X |
-------------
| X | O | X |
-------------
| O | X | O |
-------------

Game ended in a draw
QUIT
Exiting
```

Client 2 Input:

```
Wait for your turn

-------------
| X | O | X |
-------------
| X | O | X |
-------------
| O | X | O |
-------------

Game ended in a draw
QUIT
Exiting
```

Client 3 Input:

```
user's turn

-------------
| X | O | X |
-------------
| X | O | X |
-------------
| O | X | O |
-------------

Game ended in a draw
QUIT
Exiting
```

Test 4 - Explicit Forfeit

Ensure a game can end when one client explicitly forfeits.

Steps:

- Client 1:
    - Login, and create room
    - Wait for *Client 2* to join
    - Place then forfeit
- Client 2:
    - Login, and join the room (player)
    - Keep placing until *Client 1* forfeits

Client 1 Input:

```
Your turn:
FORFEIT

-------------
| X | O |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

another user won due to the opposing player forfeiting
QUIT
Exiting
```

Client 2 Input:

```
Wait for your turn

-------------
| X | O |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

another user won due to the opposing player forfeiting
QUIT
Exiting
```

Test 5 - Implicit Forfeit (Disconnect)

Verify if a game ends successfully when one client disconnects mid-game.

Steps:

- Client 1:
    - Login, and create room
    - Wait for *Client 2* to join
    - Place then disconnect
- Client 2:
    - Login, and join the room (player)
    - Keep placing until *Client 1* disconnects

Client 1 Input:

```
Your turn:
^CTraceback (most recent call last):
  File "/home/xurge/Desktop/uni/2024/sem-2/info-1112/ass-2/submission/client.py", line 393, in <mo
dule>
    main(sys.argv[1:])
  File "/home/xurge/Desktop/uni/2024/sem-2/info-1112/ass-2/submission/client.py", line 377, in mai
n
    msg_to_func[user_msg](client_socket, user_msg)
  File "/home/xurge/Desktop/uni/2024/sem-2/info-1112/ass-2/submission/client.py", line 101, in cre
ate
    begin(client_socket, '')
  File "/home/xurge/Desktop/uni/2024/sem-2/info-1112/ass-2/submission/client.py", line 289, in beg
in
    user_msg = input("Your turn:\n")
KeyboardInterrupt
```

Client 2 Input:

```
Wait for your turn

-------------
| X | O |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------

another user won due to the opposing player forfeiting
QUIT
Exiting
```