A. Implementation Details

1. *Server Program:*

- ***Job:***
  - Server program sets up the communication environment and protocols that clients need to follow.
  - Performing the gameplay.
  - Keeping the server alive all the time even if connections are lost.
  - Keeps the current situation of the game and informs clients.

- ***Methods:***
  - ***askPlayerPlayAgain(index, player_info):***
    - This method is invoked by an individual thread to ask the specified player if she/he wants to play again.
    - If player says no, player's reference in player_identifiers list is made None.
    - If player says yes, a message telling the player to wait for the next game is sent to the player.
  - ***collectDYWPAinfo(player_identifiers):***
    - This method is invoked by main thread.
    - Main thread creates as many threads as the player count and assigns each thread to ask a specific player and calls askPlayerPlayAgain() method for each of the players.
    - Each thread is joined using .join( ) method to make the Main thread to wait for each thread to collect the 'do you want to play again' information.
  - ***acceptConnections():***
    - This method is invoked by the Seconder thread that always runs concurrently with the Main thread to continuously accept connections while Main thread is running for the gameplay.
    - Seconder thread creates as many threads as the accepted connections and assigns each thread to a connection to be able to listen each guest client at the same time.
  - ***registerTheGuest(connectionSocket, addr):***
    - This method is invoked by the thread that is assigned to listen this guest client.
    - The guest client is prompt to specify a 'username' and a 'password.
    - If 'username' received is not found in the players dictionary, 'username' and 'password' is added to the dictionary.
    - This method can be actively used even if a game is active at the time. Registrations are welcome all the time.
  - ***loginTheGuest(guest_info):***
    - This method is invoked by the thread that is assigned to listen this guest client.
    - The guest client is prompt to specify a 'username' and a 'password.

- If player count for the game is reached, login is not accepted, guest client connection closes.
- If 'username' received is not found in the players dictionary, guest client is informed with an error message.
- If 'username' received is found in the players dictionary, 'password' received is compared with the 'password' value of the 'username' key in the dictionary. If it is not a match, guest client is informed with an error message.

- ***showMenuToTheGuest(connectionSocket, addr):***
  - This method is invoked by the thread that is assigned to listen this guest client.
  - A simple menu is sent to the guest client that informs the client with the actions that he/she can take.
  - Selection 1: invokes registerTheGuest() method.
  - Selection 2: invokes oginTheGuest() method.
  - Selection 3: closes both clientSocket of client and connectionSocket of server.
- ***loadPlayerInfo():***
  - This method is invoked by the Main thread.
  - At the beginning of the program, all registration information of players are loaded from a .json file to a python dictionary as {'username':'password'} pairs.
- ***savePlayerInfo(players):***
  - This method is invoked by the Main thread.
  - At the end of a gameplay, players dictionary is dumped(written) into the .json file by overwriting the file to save all newly registered and previously registered players.
- ***waitUntilClientReadsTheMessage(player_info):***
  - This method is invoked by all the threads that sends a message to a client.
  - This method acts as an 'handshake' between the client and the server.
  - A message sent to a client by the server may not be received by client before the server sends another message to the same client. In such a case, client receives the concatenation of all messages sent up to that time. This causes confusion between server and client programs. In order to make sure each message received by the client separately, server waits for a 'message received' signal from the client before moving on. That 'message received' signal is an arbitrary success code number '700'.
- ***initializePlaceholder(word, placeholder):***
  - This method is invoked by the Main thread.
  - Placeholder is the current situation of the game phrase represented as dashes(-) in a list.
  - This method appends as many dashes as the character count in the game phrase to the placeholder list.
- ***updatePlaceholder(word, placeholder, guess=None):***
  - This method is invoked by the Main thread.
  - Placeholder is updated according to the guess made.

- If the guess is a single character, It is checked if the guess is found in the phrase. If found, the index of the placeholder which is the index of the character in the phrase is updated from dash(-) to character itself.
- If the guess is more than a single character, guess is directly compared to the game phrase.

- ***wordIsFullyPredicted(placeholder):***
  - This method is invoked by the Main thread.
  - Checks if the game phrase is fully guessed by checking the dash(-) count left in the placeholder.
  - If no dashes left in placeholder, returns True.
- ***sendSameMessageToAllPlayers(player_identifiers, message, skip_player_index=(-1),inform_also_server=True):***
  - This method is used to send a specific message to all the clients currently playing the game.
  - If a player should not receive the message, its index is specified in the 'skip_player_index' parameter and that player does not receive the message.
  - If the message sent should not be printed on the server console, 'inform_also_server' parameter should be specified False.
- ***sendMessageOnlyIndexDifferToAllPlayers(player_identifiers, message):***
  - This method is used to send a specific message to all the clients currently playing the, but including their specific indexes to the message.
- ***sendMessageToSpecifiedPlayer(player_identifiers, message, index):***
  - This method is used to send a specific message to a specific client among the clients that are currently playing the game.
- ***main(global scope):***
  - This method is the scope that the Main thread runs.
  - In this scope the gameplay happens.
  - Main thread waits for sufficient amount of connections and starts the gameplay.
  - **Gameplay:**
    - Players are assigned a player number regarding their connection order to the server.
    - At each round, players are informed with the current situation of the placeholder of the game phrase, wrong guesses and whose turn it is. The player that has the turn makes the guess.
    - The guess is checked and results are sent to players.
    - If the game phrase is fully predicted, game is ended and players are informed with who the winner is.
    - If the phrase is not fully predicted yet, next round starts with the next player.
    - After the game ends, all the players are asked if they want to play again.
    - All process starts over again.

- ***Exceptions Handled:***
  - I tried to cover many of the exceptions in the server program.
  - Connection losses are handled during all the phases of the gameplay.
  - Connection losses are handled during the Registration and Login.
  - Unexpected user inputs are handled.

- ***Imported Modules:***
  - ***socket module:***
    - This module forms the basis of all network communications
    - By including this module, It is possible to create sockets within our program.
  - ***threading module:***
    - This module is used to perform the tasks simultaneously.
    - While the Main thread is performing the gameplay, another thread is used to wait for connections and send menus, and many other threads are used to prompt users and collect answers in parallel.
  - ***json module:***
    - This module is used to make the writing and reading operation of a python dictionary to/from a file very simple.
    - Since the python dictionary format and JavaScript object notation format is similar, it is very useful to save and load dictionaries to/from '.json' files.

2. *Client Program:*

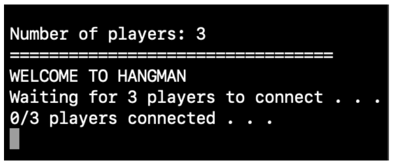
- ***Job:***
    - Following the protocol that the server set.

- ***Methods:***
  - ***waitUntilServerReadsTheMessage():***
    - This method acts as an 'handshake' between the client and the server.
    - A message sent to the server by the client may not be received by server before the same client sends another message to the server. In such a case, server receives the concatenation of all messages sent up to that time. This causes confusion between server and client programs. In order to make sure each message received by the server separately, client waits for a 'message received' signal from the server before moving on. That 'message received' signal is an arbitrary success code number '700'.
  - ***main(global scope):***
    - The client continuously waits for a message from the server and takes an action regarding the message received.

- ***Imported Modules:***
  - ***socket module***

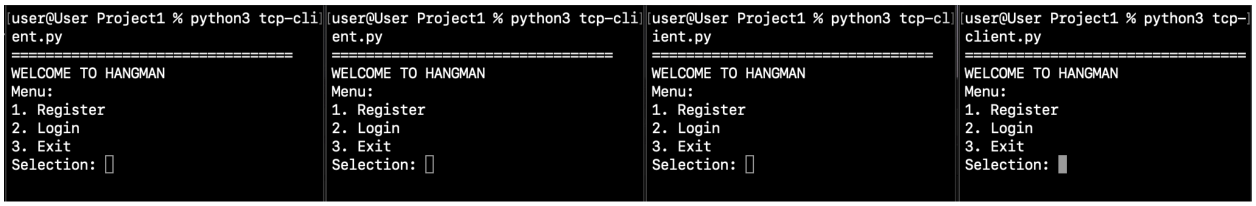
Screenshots from the Gameplay:

- ***Figure 1:***
  - Server program is executed.
  - Number of players is specified as 3.
  - Server Greets us and waits for the connections.



```
Number of players: 3
================================
WELCOME TO HANGMAN
Waiting for 3 players to connect . . .
0/3 players connected . . .
```

- ***Figure 2:***
  - 4 clients are connected to the server.
  - Menu is sent each of them by 4 different threads.



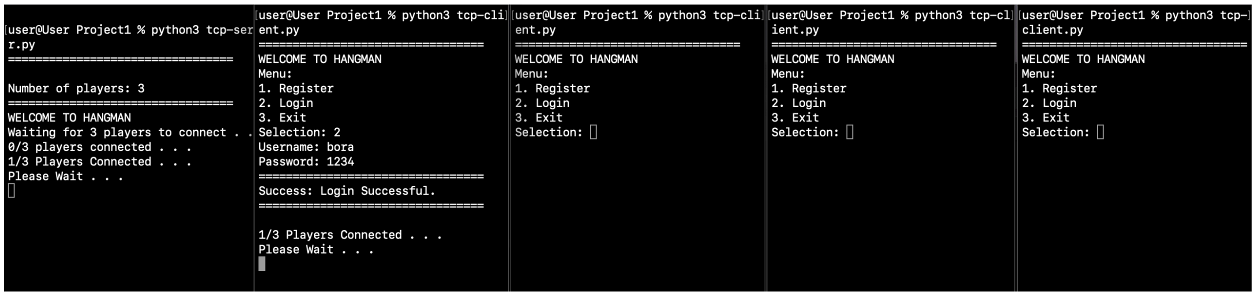
```
user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

```
user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

```
user@User Project1 % python3 tcp-cl
ient.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

```
user@User Project1 % python3 tcp-
client.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

- ***Figure 3:***
  - Leftmost client Selects 2 to Login to play the game.
  - Enters username:bora, password:1234.
  - Logins and starts to wait for other logins.



```
user@User Project1 % python3 tcp-ser
r.py
================================

Number of players: 3
================================
WELCOME TO HANGMAN
Waiting for 3 players to connect . .
0/3 players connected . . .
1/3 Players Connected . . .
Please Wait . . .
```

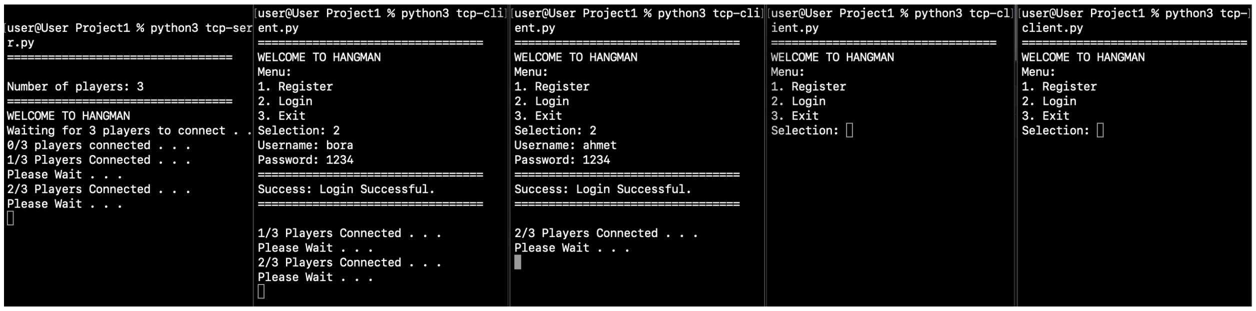
```
user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: bora
Password: 1234
================================
Success: Login Successful.
================================

1/3 Players Connected . . .
Please Wait . . .
```

```
user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

```
user@User Project1 % python3 tcp-cl
ient.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

```
user@User Project1 % python3 tcp-
client.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

- ***Figure 4:***
  - Second client Selects 2 to Login to play the game.
  - Enters username:ahmet, password:1234.
  - Logins and starts to wait for other logins.
  - Also other already logged-in player is informed.



```
user@User Project1 % python3 tcp-ser
r.py
================================

Number of players: 3
================================
WELCOME TO HANGMAN
Waiting for 3 players to connect . .
0/3 players connected . . .
1/3 Players Connected . . .
Please Wait . . .
2/3 Players Connected . . .
Please Wait . . .
```

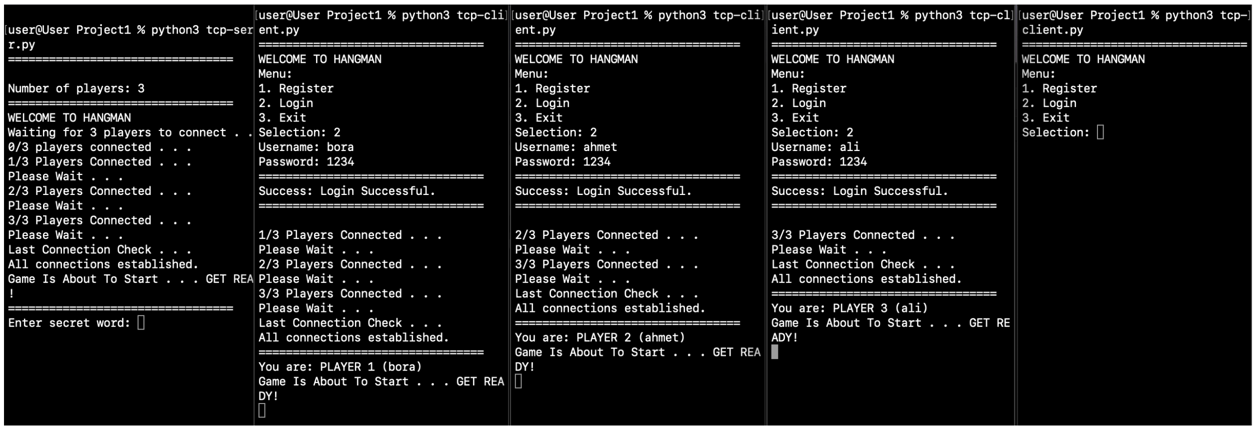
```
user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: bora
Password: 1234
================================
Success: Login Successful.
================================

1/3 Players Connected . . .
Please Wait . . .
2/3 Players Connected . . .
Please Wait . . .
```

```
user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: ahmet
Password: 1234
================================
Success: Login Successful.
================================

2/3 Players Connected . . .
Please Wait . . .
```

```
user@User Project1 % python3 tcp-cl
ient.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

```
user@User Project1 % python3 tcp-
client.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

- ***Figure 5:***
  - Third client Selects 2 to Login to play the game.
  - Enters username:ahmet, password:1234.
  - Logins and starts to wait for other logins.
  - Also other already logged-in players are informed.
  - Server is prompt to enter a secret word and a hint to start the game.



```
[user@User Project1 % python3 tcp-ser
r.py
================================

Number of players: 3
================================
WELCOME TO HANGMAN
Waiting for 3 players to connect . .
0/3 players connected . . .
1/3 Players Connected . . .
Please Wait . . .
2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
Game Is About To Start . . . GET REA
!
================================
Enter secret word:
```

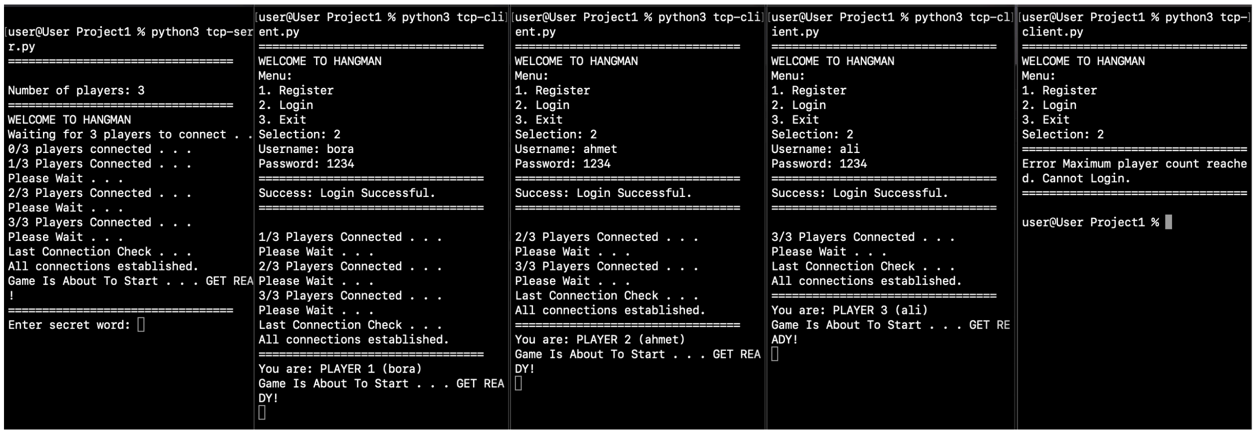
```
[user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: bora
Password: 1234
================================
Success: Login Successful.
================================

1/3 Players Connected . . .
Please Wait . . .
2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
================================
You are: PLAYER 1 (bora)
Game Is About To Start . . . GET REA
DY!
```

```
[user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: ahmet
Password: 1234
================================
Success: Login Successful.
================================

2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
================================
You are: PLAYER 2 (ahmet)
Game Is About To Start . . . GET REA
DY!
```

```
[user@User Project1 % python3 tcp-cl
ient.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: ali
Password: 1234
================================
Success: Login Successful.
================================

3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
================================
You are: PLAYER 3 (ali)
Game Is About To Start . . . GET RE
ADY!
```

```
[user@User Project1 % python3 tcp-
client.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection:
```

- ***Figure 6:***
  - Fourth client Selects 2 to Login to play the game.
  - Immediatelly informed by its connection thread that the maximum player count is reached and connection of the guest is closed.
  - However, the guest would be able to Select 1 to Register just fine.



```
[user@User Project1 % python3 tcp-ser
r.py
================================

Number of players: 3
================================
WELCOME TO HANGMAN
Waiting for 3 players to connect . .
0/3 players connected . . .
1/3 Players Connected . . .
Please Wait . . .
2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
Game Is About To Start . . . GET REA
!
================================
Enter secret word:
```

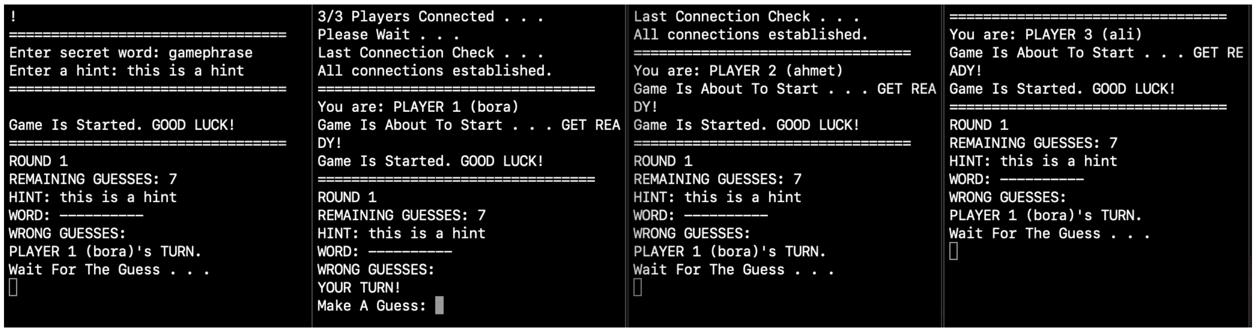
```
[user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: bora
Password: 1234
================================
Success: Login Successful.
================================

1/3 Players Connected . . .
Please Wait . . .
2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
================================
You are: PLAYER 1 (bora)
Game Is About To Start . . . GET REA
DY!
```

```
[user@User Project1 % python3 tcp-cli
ent.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: ahmet
Password: 1234
================================
Success: Login Successful.
================================

2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
================================
You are: PLAYER 2 (ahmet)
Game Is About To Start . . . GET REA
DY!
```

```
[user@User Project1 % python3 tcp-cl
ient.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: ali
Password: 1234
================================
Success: Login Successful.
================================

3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
================================
You are: PLAYER 3 (ali)
Game Is About To Start . . . GET RE
ADY!
```

```
[user@User Project1 % python3 tcp-
client.py
================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
================================
Error Maximum player count reache
d. Cannot Login.
================================

user@User Project1 %
```

- ***Figure 7:***
  - After server enters the secret word and a hint, server starts the first round.
  - Raund 1 begins. It is Player 1 (bora)'s turn, since he was the fist to login.



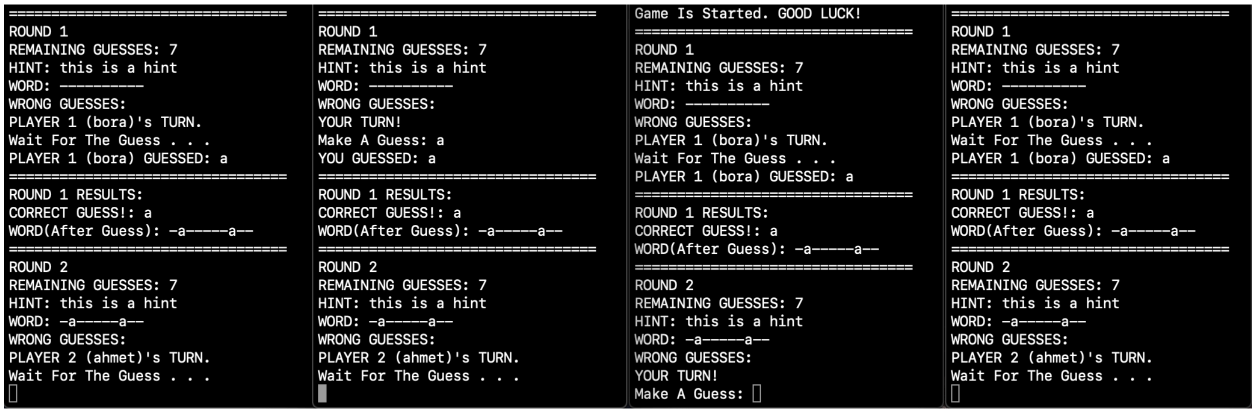
```
!
=================================
Enter secret word: gamephrase
Enter a hint: this is a hint
=================================

Game Is Started. GOOD LUCK!
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
```

```
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
All connections established.
=================================
You are: PLAYER 1 (bora)
Game Is About To Start . . . GET REA
DY!
Game Is Started. GOOD LUCK!
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
YOUR TURN!
Make A Guess:
```

```
Last Connection Check . . .
All connections established.
=================================
You are: PLAYER 2 (ahmet)
Game Is About To Start . . . GET REA
DY!
Game Is Started. GOOD LUCK!
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
```

```
=================================
You are: PLAYER 3 (ali)
Game Is About To Start . . . GET RE
ADY!
Game Is Started. GOOD LUCK!
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
```

- ***Figure 8:***
  - Player 1 (bora) makes a guess: 'a'
  - Round 1 results are sent to the players.
  - Raund 2 begins. Player 2 (ahmet)'s turn.



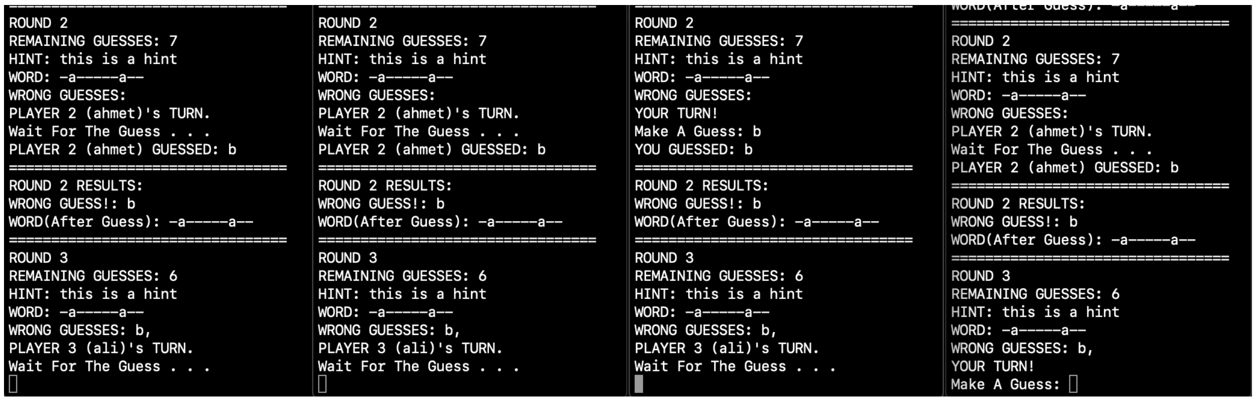
```
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: a
=================================
ROUND 1 RESULTS:
CORRECT GUESS!: a
WORD(After Guess): -a-----a--
=================================
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
PLAYER 2 (ahmet)'s TURN.
Wait For The Guess . . .
```

```
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
YOUR TURN!
Make A Guess: a
YOU GUESSED: a
=================================
ROUND 1 RESULTS:
CORRECT GUESS!: a
WORD(After Guess): -a-----a--
=================================
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
PLAYER 2 (ahmet)'s TURN.
Wait For The Guess . . .
```

```
Game Is Started. GOOD LUCK!
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: a
=================================
ROUND 1 RESULTS:
CORRECT GUESS!: a
WORD(After Guess): -a-----a--
=================================
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
YOUR TURN!
Make A Guess:
```

```
=================================
ROUND 1
REMAINING GUESSES: 7
HINT: this is a hint
WORD: ----------
WRONG GUESSES:
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: a
=================================
ROUND 1 RESULTS:
CORRECT GUESS!: a
WORD(After Guess): -a-----a--
=================================
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
PLAYER 2 (ahmet)'s TURN.
Wait For The Guess . . .
```

- ***Figure 9:***
  - Player 2 (ahmet) makes a guess: 'b'
  - Round 2 results are sent to the players.
  - Raund 3 begins. Player 3 (ali)'s turn.



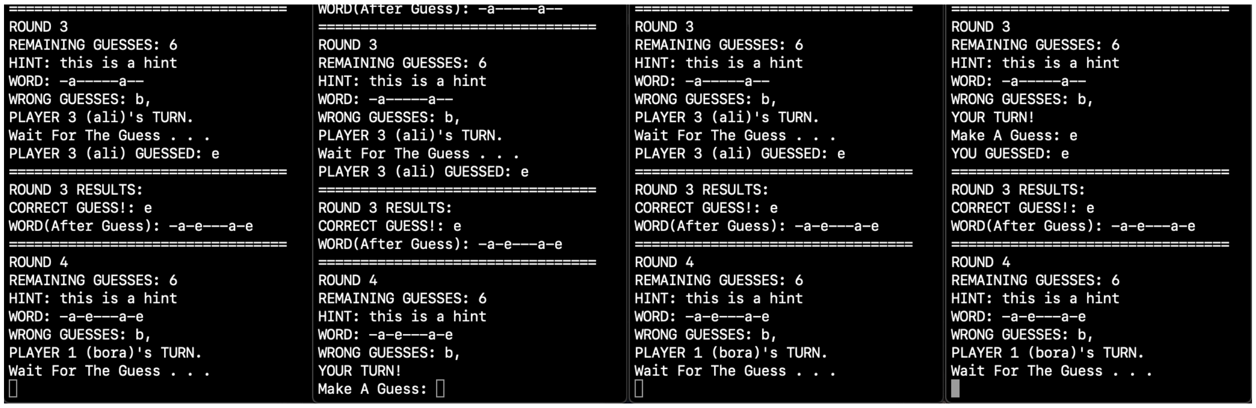
```
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
PLAYER 2 (ahmet)'s TURN.
Wait For The Guess . . .
PLAYER 2 (ahmet) GUESSED: b
=================================
ROUND 2 RESULTS:
WRONG GUESS!: b
WORD(After Guess): -a-----a--
=================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
```

```
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
PLAYER 2 (ahmet)'s TURN.
Wait For The Guess . . .
PLAYER 2 (ahmet) GUESSED: b
=================================
ROUND 2 RESULTS:
WRONG GUESS!: b
WORD(After Guess): -a-----a--
=================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
```

```
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
YOUR TURN!
Make A Guess: b
YOU GUESSED: b
=================================
ROUND 2 RESULTS:
WRONG GUESS!: b
WORD(After Guess): -a-----a--
=================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
```

```
WORD(After Guess): -a-----a--
=================================
ROUND 2
REMAINING GUESSES: 7
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES:
PLAYER 2 (ahmet)'s TURN.
Wait For The Guess . . .
PLAYER 2 (ahmet) GUESSED: b
=================================
ROUND 2 RESULTS:
WRONG GUESS!: b
WORD(After Guess): -a-----a--
=================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
YOUR TURN!
Make A Guess:
```

- ***Figure 10:***
    - Player 3 (ali) makes a guess: 'e'
    - Round 3 results are sent to the players.
    - Raund 4 begins. Player 1 (bora)'s turn.



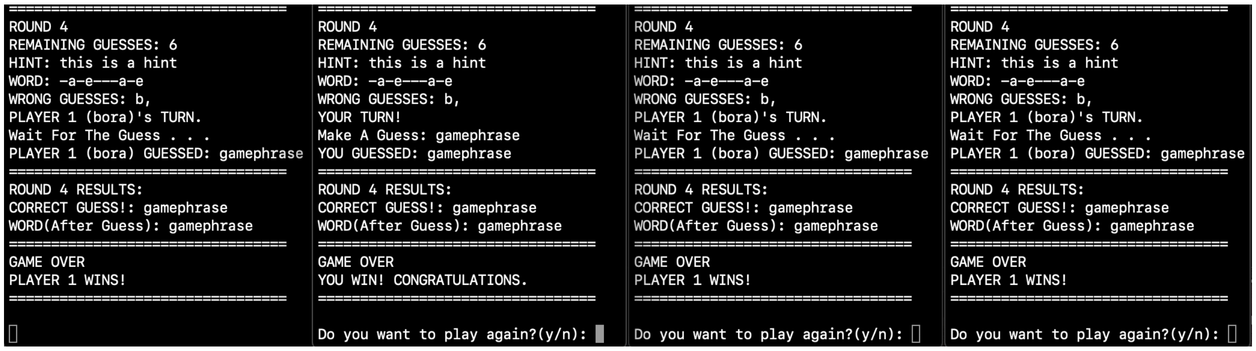
```
================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
PLAYER 3 (ali) GUESSED: e
================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
```

```
WORD(After Guess): -a-----a--
================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
PLAYER 3 (ali) GUESSED: e
================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
YOUR TURN!
Make A Guess:
```

```
================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
PLAYER 3 (ali) GUESSED: e
================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
```

```
================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
YOUR TURN!
Make A Guess: e
YOU GUESSED: e
================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
```

- ***Figure 11:***
    - Player 1 (bora) makes a guess: 'gamephrase'
    - Round 3 results are sent to the players.
    - Game ends. Each player is asked if they want to play again by their connection threads simultaneously.



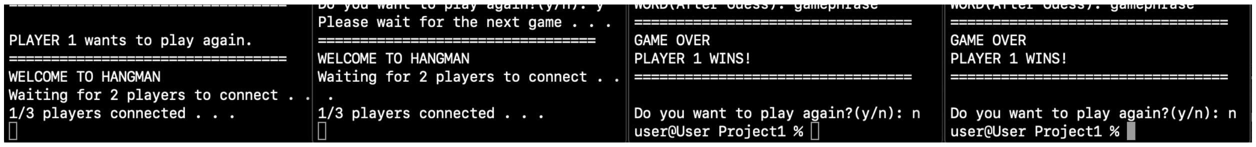
```
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
================================
GAME OVER
PLAYER 1 WINS!
================================
```

```
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
YOUR TURN!
Make A Guess: gamephrase
YOU GUESSED: gamephrase
================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
================================
GAME OVER
YOU WIN! CONGRATULATIONS.
================================

Do you want to play again?(y/n):
```

```
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
================================
GAME OVER
PLAYER 1 WINS!
================================

Do you want to play again?(y/n):
```

```
================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
================================
GAME OVER
PLAYER 1 WINS!
================================

Do you want to play again?(y/n):
```

- ***Figure 12:***
    - Player 1 (bora) wants to play again.
    - Player 2 (ahmet) does not want to play again.
    - Player 3 (ali) does not want to play again.
    - Server waits for the remaining 2 connections.



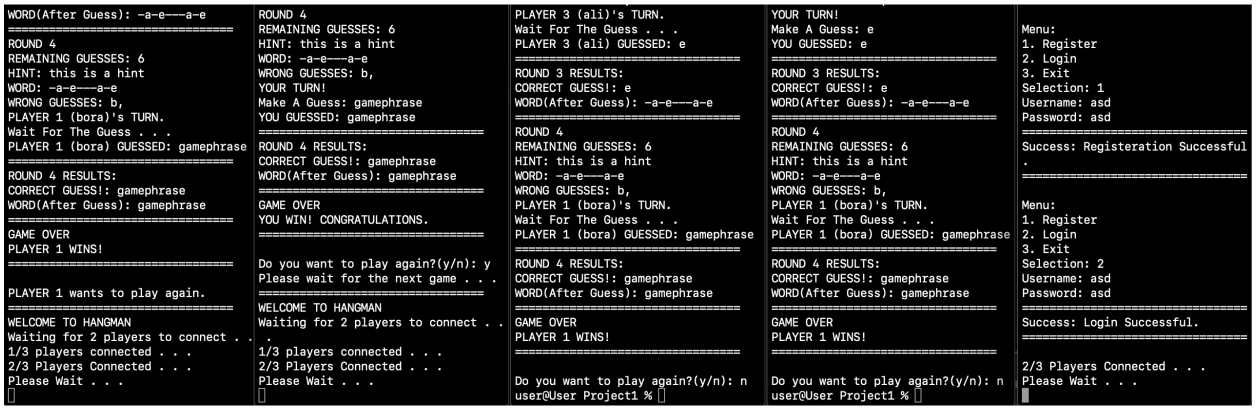
```
================================

PLAYER 1 wants to play again.
================================
WELCOME TO HANGMAN
Waiting for 2 players to connect . .
1/3 players connected . . .
```

```
Do you want to play again?(y/n): y
Please wait for the next game . . .
================================
WELCOME TO HANGMAN
Waiting for 2 players to connect . .
.
1/3 players connected . . .
```

```
WORD(After Guess): gamephrase
================================
GAME OVER
PLAYER 1 WINS!
================================

Do you want to play again?(y/n): n
user@User Project1 %
```

```
WORD(After Guess): gamephrase
================================
GAME OVER
PLAYER 1 WINS!
================================

Do you want to play again?(y/n): n
user@User Project1 %
```

- ***Figure 13:***
  - Rightmost client Registers. Username:asd, password:asd
  - Rightmost client Logins.



```
WORD(After Guess): -a-e---a-e
=================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
PLAYER 1 WINS!
=================================

PLAYER 1 wants to play again.
=================================
WELCOME TO HANGMAN
Waiting for 2 players to connect . .
1/3 players connected . . .
2/3 Players Connected . . .
Please Wait . . .
```

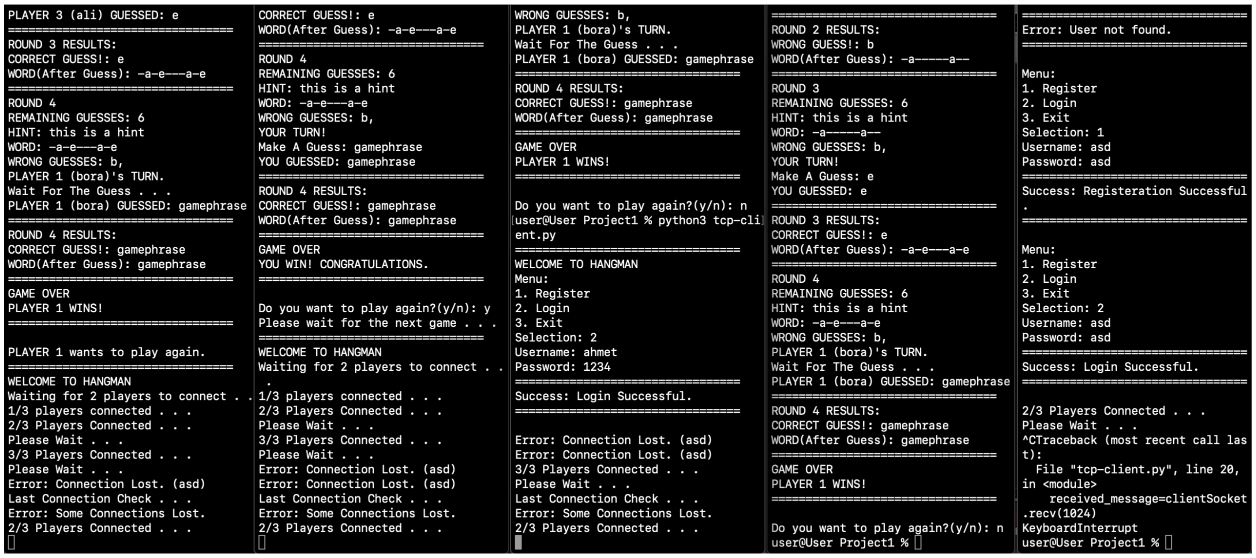
```
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
YOUR TURN!
Make A Guess: gamephrase
YOU GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
YOU WIN! CONGRATULATIONS.
=================================

Do you want to play again?(y/n): y
Please wait for the next game . . .
=================================
WELCOME TO HANGMAN
Waiting for 2 players to connect . .
 .
1/3 players connected . . .
2/3 Players Connected . . .
Please Wait . . .
```

```
PLAYER 3 (ali)'s TURN.
Wait For The Guess . . .
PLAYER 3 (ali) GUESSED: e
=================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
=================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
PLAYER 1 WINS!
=================================


Do you want to play again?(y/n): n
user@User Project1 %
```

```
YOUR TURN!
Make A Guess: e
YOU GUESSED: e
=================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
=================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
PLAYER 1 WINS!
=================================


Do you want to play again?(y/n): n
user@User Project1 %
```

```

Menu:
1. Register
2. Login
3. Exit
Selection: 1
Username: asd
Password: asd
=================================
Success: Registeration Successful
.
=================================


Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: asd
Password: asd
=================================
Success: Login Successful.
=================================

2/3 Players Connected . . .
Please Wait . . .
```

- ***Figure 14:***
  - Client in the middle logins, meanwhile Rightmost client decides to close the game.
  - Server first sees 3 logins and tends to start the game, but checks for the login connections last time before starting the game.
  - Sees that the (asd) left.
  - Server and logged-in players waits again.



```
PLAYER 3 (ali) GUESSED: e
=================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
=================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
PLAYER 1 WINS!
=================================

PLAYER 1 wants to play again.
=================================
WELCOME TO HANGMAN
Waiting for 2 players to connect . .
1/3 players connected . . .
2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Error: Connection Lost. (asd)
Last Connection Check . . .
Error: Some Connections Lost.
2/3 Players Connected . . .
```

```
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
=================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
YOUR TURN!
Make A Guess: gamephrase
YOU GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
YOU WIN! CONGRATULATIONS.
=================================

Do you want to play again?(y/n): y
Please wait for the next game . . .
=================================
WELCOME TO HANGMAN
Waiting for 2 players to connect . .
 .
1/3 players connected . . .
2/3 Players Connected . . .
Please Wait . . .
3/3 Players Connected . . .
Please Wait . . .
Error: Connection Lost. (asd)
Error: Connection Lost. (asd)
Last Connection Check . . .
Error: Some Connections Lost.
2/3 Players Connected . . .
```

```
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
PLAYER 1 WINS!
=================================

Do you want to play again?(y/n): n
user@User Project1 % python3 tcp-cli
ent.py
=================================
WELCOME TO HANGMAN
Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: ahmet
Password: 1234
=================================
Success: Login Successful.
=================================

Error: Connection Lost. (asd)
Error: Connection Lost. (asd)
3/3 Players Connected . . .
Please Wait . . .
Last Connection Check . . .
Error: Some Connections Lost.
2/3 Players Connected . . .
```

```
=================================
ROUND 2 RESULTS:
WRONG GUESS!: b
WORD(After Guess): -a-----a--
=================================
ROUND 3
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-----a--
WRONG GUESSES: b,
YOUR TURN!
Make A Guess: e
YOU GUESSED: e
=================================
ROUND 3 RESULTS:
CORRECT GUESS!: e
WORD(After Guess): -a-e---a-e
=================================
ROUND 4
REMAINING GUESSES: 6
HINT: this is a hint
WORD: -a-e---a-e
WRONG GUESSES: b,
PLAYER 1 (bora)'s TURN.
Wait For The Guess . . .
PLAYER 1 (bora) GUESSED: gamephrase
=================================
ROUND 4 RESULTS:
CORRECT GUESS!: gamephrase
WORD(After Guess): gamephrase
=================================
GAME OVER
PLAYER 1 WINS!
=================================

Do you want to play again?(y/n): n
user@User Project1 %
```

```
=================================
Error: User not found.
=================================

Menu:
1. Register
2. Login
3. Exit
Selection: 1
Username: asd
Password: asd
=================================
Success: Registeration Successful
.
=================================

Menu:
1. Register
2. Login
3. Exit
Selection: 2
Username: asd
Password: asd
=================================
Success: Login Successful.
=================================

2/3 Players Connected . . .
Please Wait . . .
^CTraceback (most recent call las
t):
  File "tcp-client.py", line 20,
in <module>
    received_message=clientSocket
.recv(1024)
KeyboardInterrupt
user@User Project1 %
```