Computer Networks
Kim Finical
Socket Project

Design Document: Full Project

(A) Message Format

1. Register

Message Format: register <player> <IPv4-address> <m-port> <r-port> <p-port>

Description:

- register = allows for player registration with the server
- player = name of player registering to the server
- IPv4-address = players machine IPv4-address
- m-port = manager port used with a range of 3500-3999
- r-port = logical network port
- p-port = peer-to-peer port

Response:

- If the registration to the game server was successful and not a duplicate it will respond with “SUCCESS. Welcome to Go Fish”
- If the registration to the game server was not successful or the player is already registered it will respond with “Failed to register player”

2. Query Players

Message Format: query-players

Description:

- query-players = gives player access to a list of players that have registered to the game server

Response:

- If the query was successful it will list the player information and provide a success message “SUCCESS. Printing a list of all registered players”
- It will then print out all the players info which includes: name, IPv4-address, m-port, r-port, and p-port

- If the query was not successful it will set to zero and the list will be empty and provide a success message “SUCCESS. There are No players registered”

3. **Start-Game**

   Message Format: start-game <player> <k>

   Description:

   - start-game = Starts the game with 1 <= k <= 4 players with <player> as the dealer
   - <k> is the number of additional players in the game

   Response:

   - If a player is not registered and if <k> is not in the proper range both will print out a “FAILURE” message
   - If all conditions for the start game are met correctly it will print out a “SUCCESS” message

4. **Query Games**

   Message Format: query-games

   Description:

   - query-games = gives player access to a list of ongoing games happening on the game server

   Response:

   - If the query was successful it will respond with the number of ongoing games that are happening.
   - If the query was not successful it will respond with “There are no other games.”

5. **End-Game**

   Message Format: end-game <game-identifier> <dealer-player>

   Description:

   - end-game = the whole game running successfully and when the game is completed stops once the <game-identifier> <dealer-player> is entered by the dealer

   Response:

   - If the <game-identifier> <dealer-player> do not match then it will output a “FAILURE” message

- On th either hand if the inputs do match then a “SUCCESS” message with me outputted

6. **De-Register**

Message Format: de-register <player>

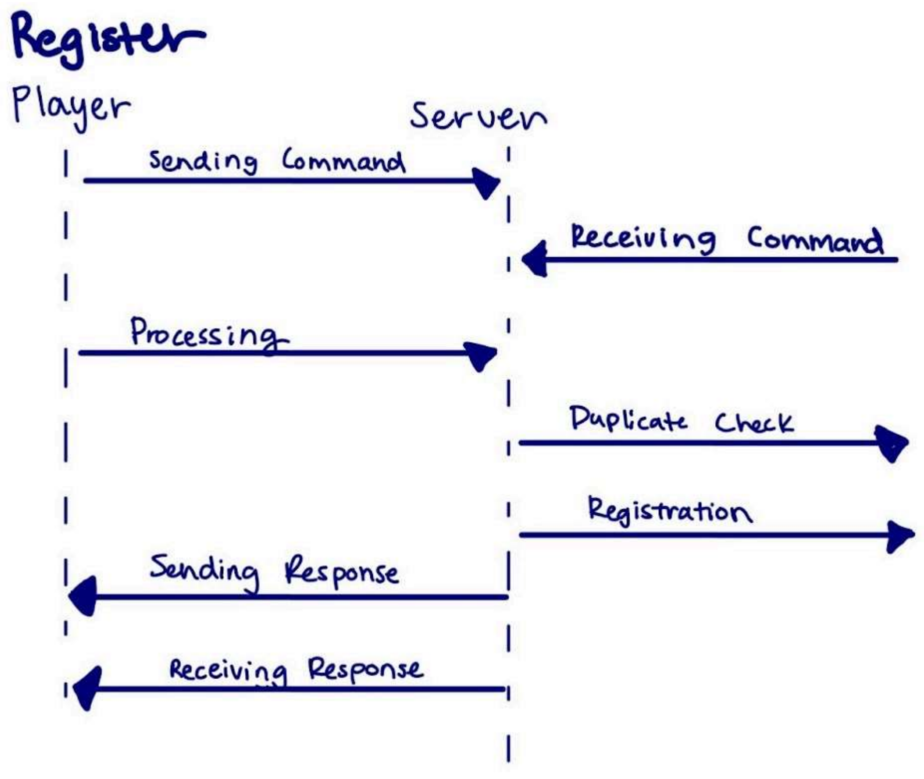
Description:

- de-register = allows for player to de-register and leave the game
- player = the name of player that will be de-registerd

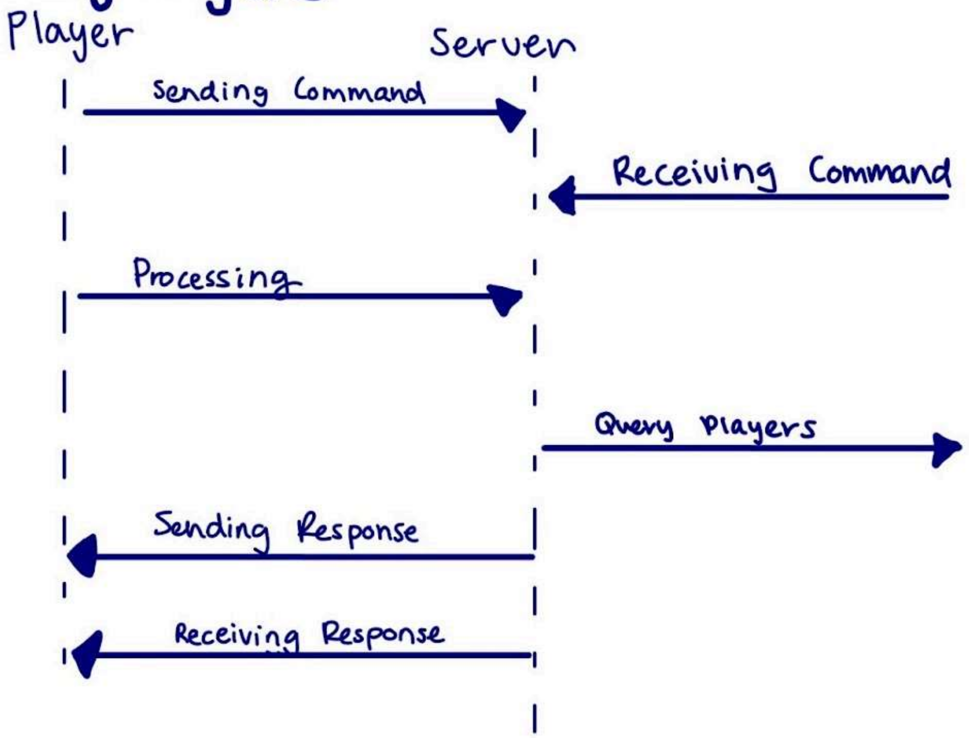
Response:

- If the de-registration to the game server was successful it will respond with “<player> + was de-registered from the game.”
- If the de-registration to the game server was not successful it will respond with “Failed to de-register player, player does not exist”

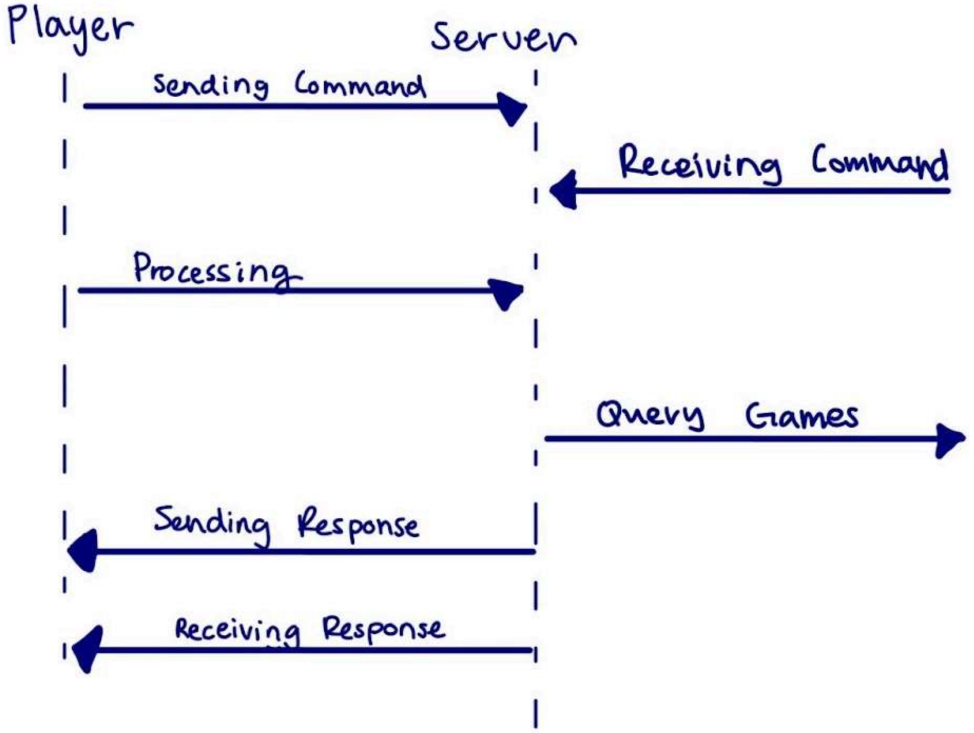
(B) Time-Space Diagram



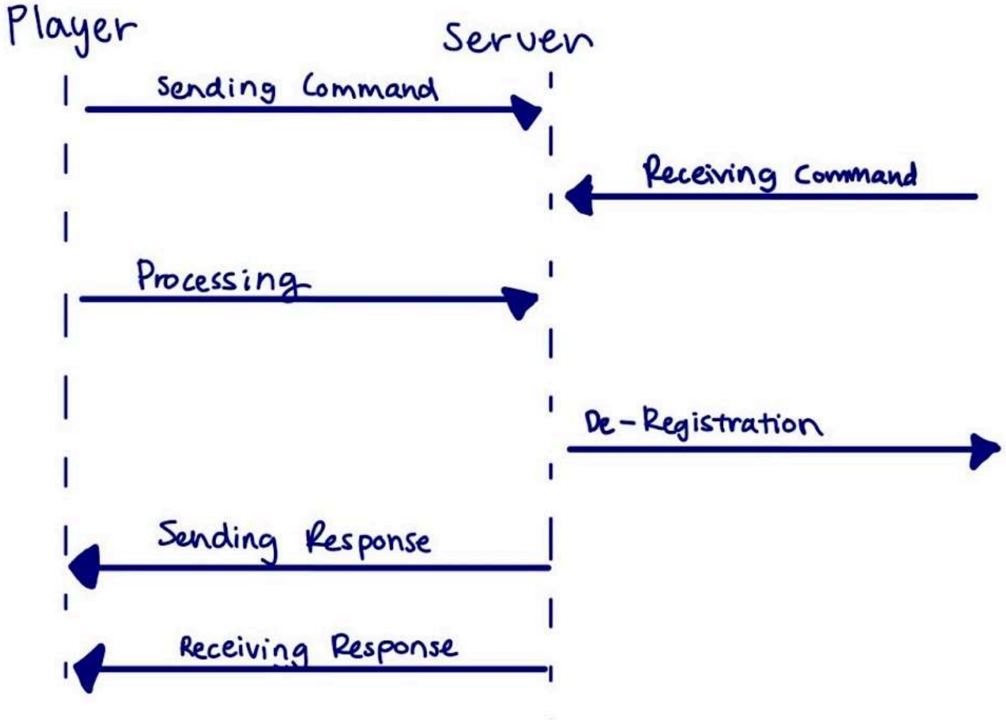
Query Players



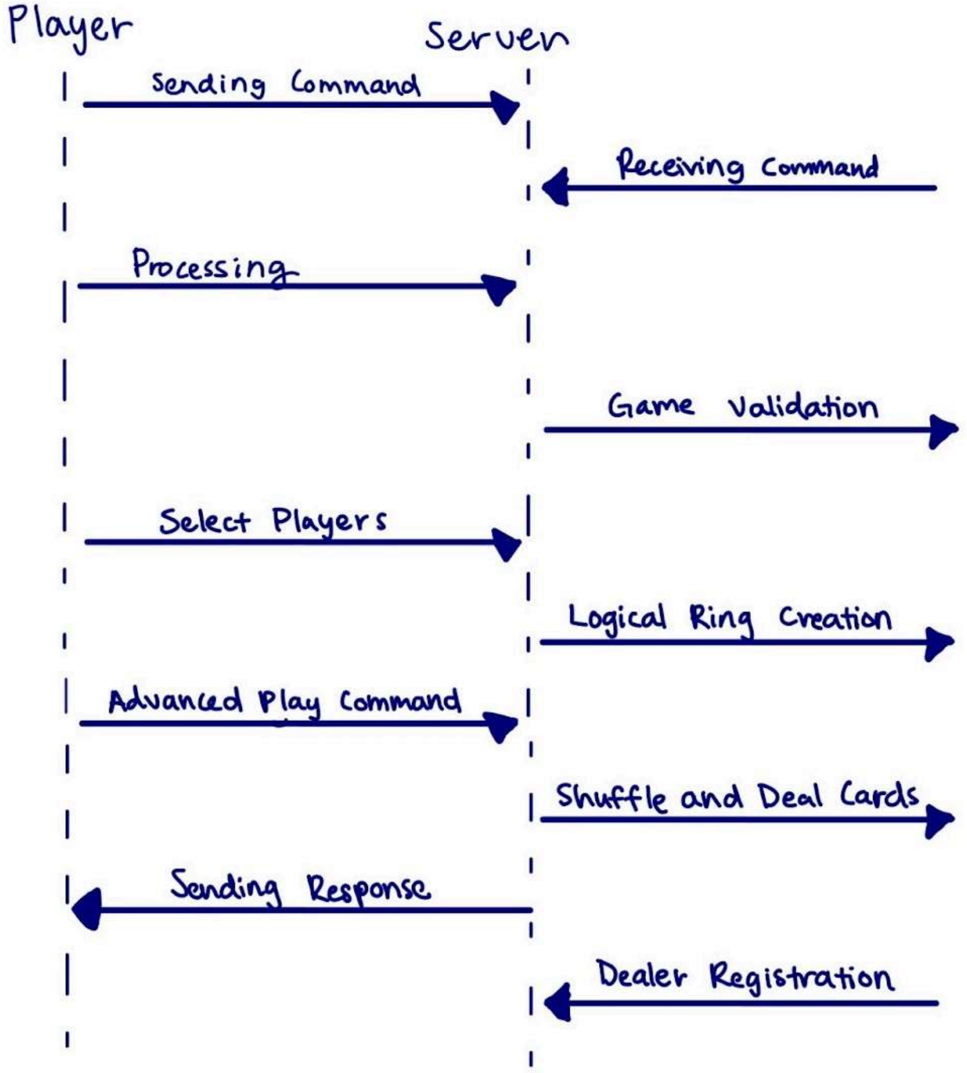
Query Games



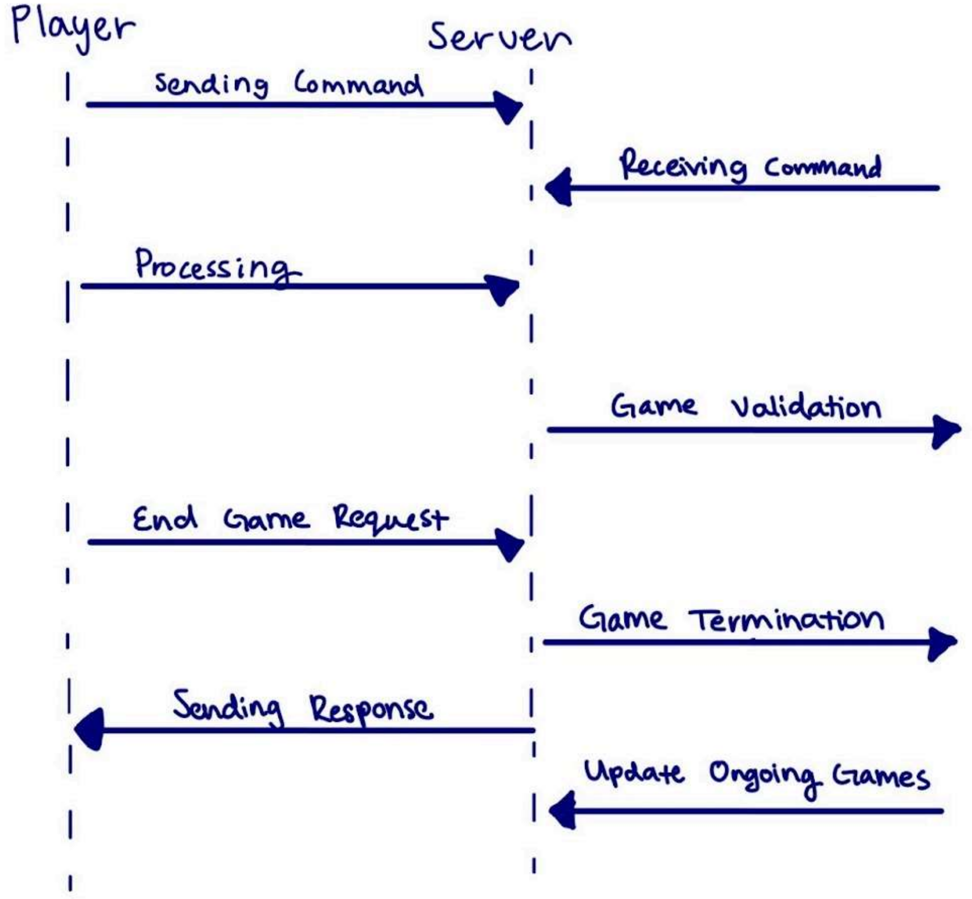
De-Register



Start-Game


Player
Server
Sending Command
Receiving Command
Processing
Game Validation
Select Players
Logical Ring Creation
Advanced Play Command
Shuffle and Deal Cards
Sending Response
Dealer Registration

End-Game


Player
Serven
Sending Command
Receiving Command
Processing
Game Validation
End Game Request
Game Termination
Sending Response
Update Ongoing Games

(C) Design Decisions

Data Structures

- playerDatatbase: this is where all the players that have been registered information is stored. It contains the players name, IPv4-address, m-port, r-port, and p-port.
- gameDatabase: this is where all the running games information is stored. It contains the games dealer and selectedPlayers.
- onGoingGames: this is where all active games progress are being followed. It contains a list of information including the gameIdentifier, dealer and selectedPlayers.
- portR and portP: this is where the logical network communication and P2P communication ports for players.
- dealer: this is where the dealer information and game identifiers are stored.

Algorithms

- Shuffle the deck: Uses the “shuffle” and “random” functions to shuffle the deck of cards at random.
- Random Player Selection: Uses the “random” library to select a specific number of players to play a game.

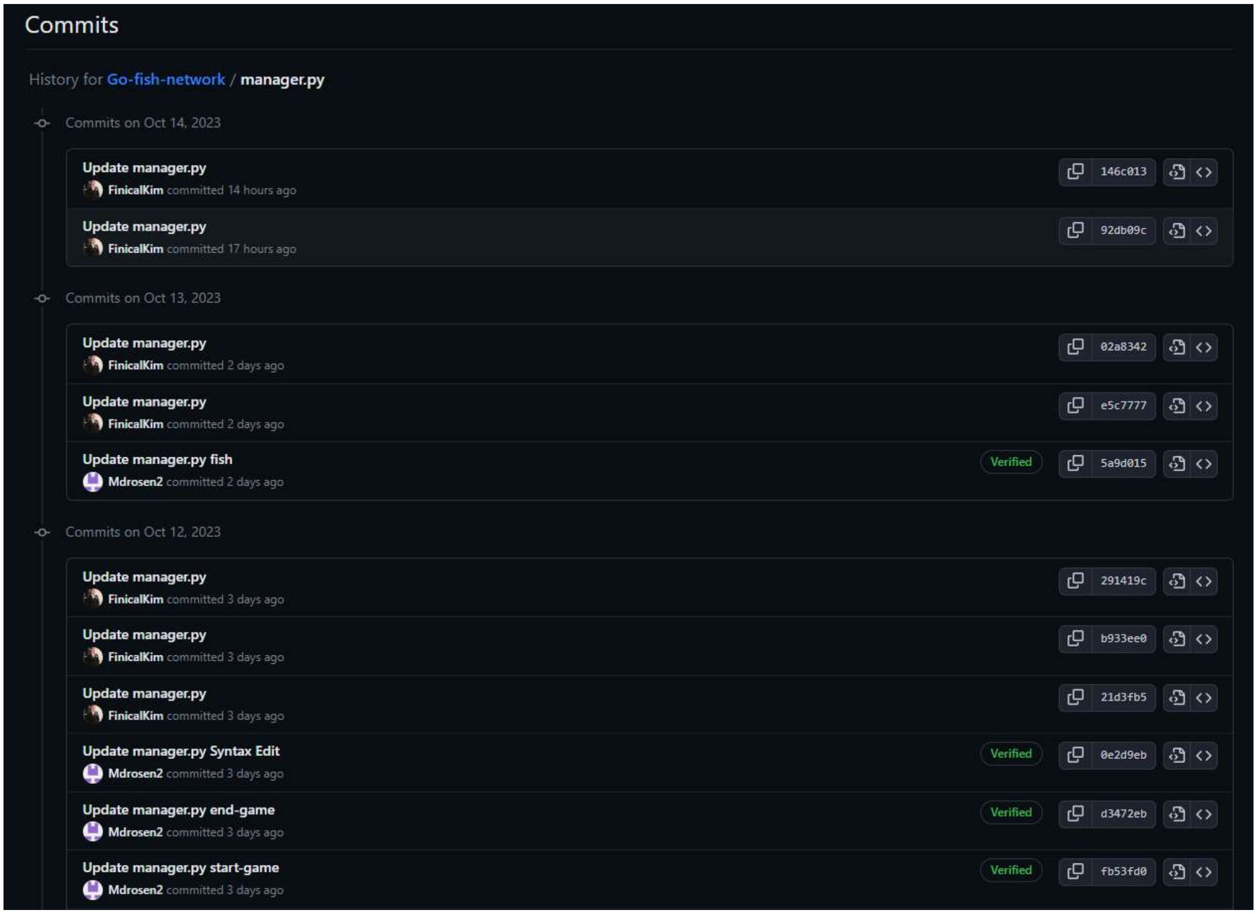
Implementation Considerations

- Port-Range: Makes sure the m-port that is provided falls between the desired range of 3500-3999.
- Player Registration: There is an extra check before adding a player. It checks if the players name has already been registered. If not it adds the players information to the database.

Other Design Decisions

- Error Handling: This was included to respond appropriately to unforeseen situations within the game.

(D) Version Control Snap Shot



Commits

History for Go-fish-network / manager.py

Commits on Oct 14, 2023

Update manager.py
FinicalKim committed 14 hours ago — 146c013

Update manager.py
FinicalKim committed 17 hours ago — 92db09c

Commits on Oct 13, 2023

Update manager.py
FinicalKim committed 2 days ago — 02a8342

Update manager.py
FinicalKim committed 2 days ago — e5c7777

Update manager.py fish
Mdrosen2 committed 2 days ago — Verified — 5a9d015

Commits on Oct 12, 2023

Update manager.py
FinicalKim committed 3 days ago — 291419c

Update manager.py
FinicalKim committed 3 days ago — b933ee0

Update manager.py
FinicalKim committed 3 days ago — 21d3fb5

Update manager.py Syntax Edit
Mdrosen2 committed 3 days ago — Verified — 0e2d9eb

Update manager.py end-game
Mdrosen2 committed 3 days ago — Verified — d3472eb

Update manager.py start-game
Mdrosen2 committed 3 days ago — Verified — fb53fd0

(E) Video Demo Link

https://youtu.be/t8hmQ1QSt1w

Time Stamps:

(a) 0:00 Introduction
(b) 0:24 Compile programs
(c) 0:50 Register players
1:35 Start game
(d) 1:50 Query player
2:07 Query games
(e) 2:20 Play Go Fish
18:40 Winner declared
(f) 19:00 Deregister players