Group 2: BlackJack
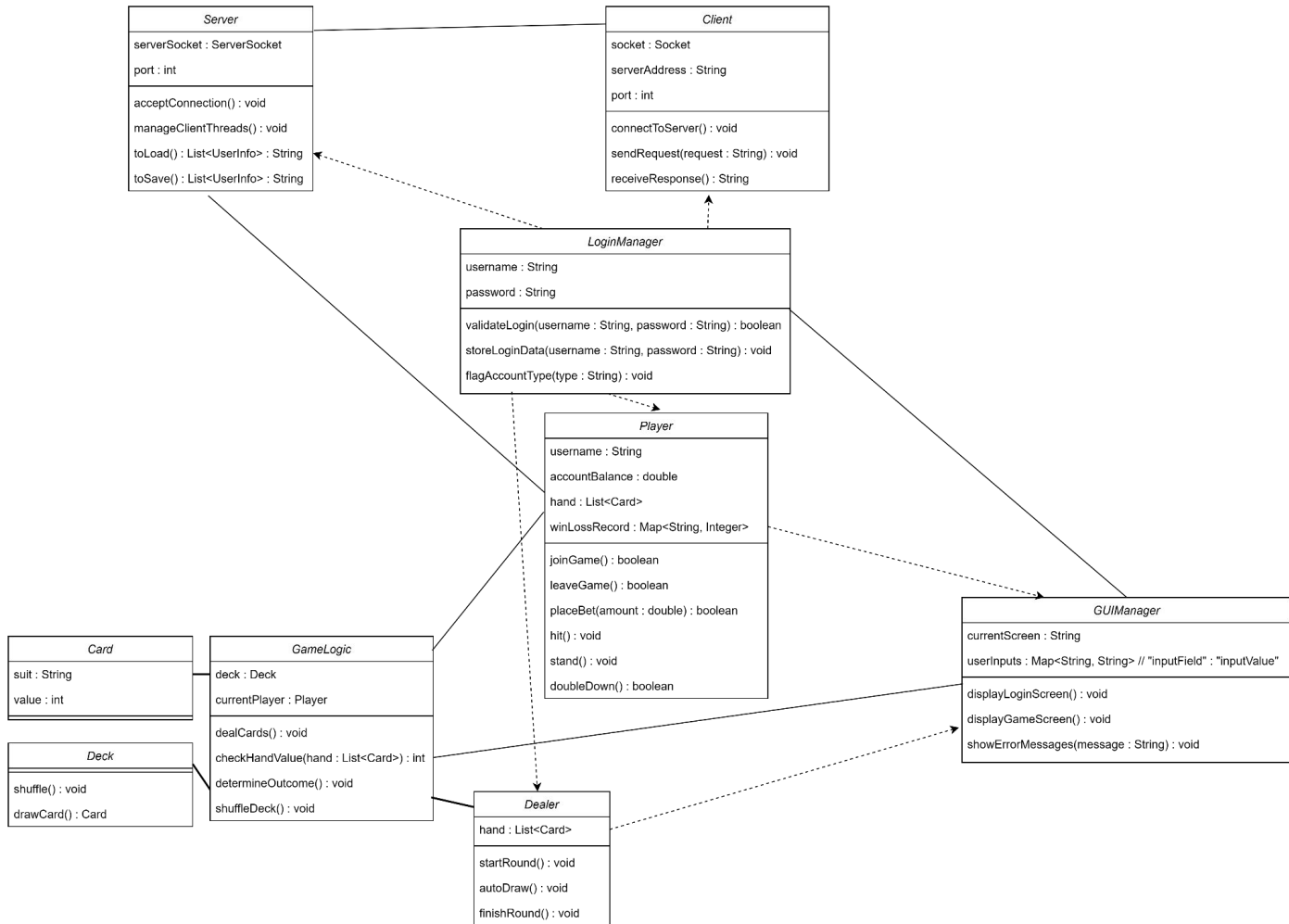
Ayoub Mekkaoui
Andrew Nguyen
Sandeep Deoja
Ishwdeep Singh

Top 5 Class Candidates

- Server/Client
  - Server and Client will have a connection to communicate.
- LoginManager
  - This class will initiate user login (w/ password) as well as flag users as Player or Dealer.
- Player
  - The Player class will handle all of the actions a player can take.
  - The Player class will also be responsible to send updated data to Server's "database" (file).
- Dealer
  - The Dealer Class will control the flow of the round.
- GameLogic (Card/Deck "subclasses")
  - Include methods to enforce the rules of the game.
  - Will also be in control of shuffling and dealing cards (not the dealer), requesting actions from players and dealers once certain conditions are met, determine the outcome of rounds, in addition to other maintenance.

Server

serverSocket : ServerSocket

port : int

---

acceptConnection() : void

manageClientThreads() : void

toLoad() : List<UserInfo> : String

toSave() : List<UserInfo> : String

Client

socket : Socket

serverAddress : String

port : int

---

connectToServer() : void

sendRequest(request : String) : void

receiveResponse() : String

LoginManager

username : String

password : String

---

validateLogin(username : String, password : String) : boolean

storeLoginData(username : String, password : String) : void

flagAccountType(type : String) : void

Player

username : String

accountBalance : double

hand : List<Card>

winLossRecord : Map<String, Integer>

---

joinGame() : boolean

leaveGame() : boolean

placeBet(amount : double) : boolean

hit() : void

stand() : void

doubleDown() : boolean

GUIManager

currentScreen : String

userInputs : Map<String, String> // "inputField" : "inputValue"

---

displayLoginScreen() : void

displayGameScreen() : void

showErrorMessages(message : String) : void

Card

suit : String

value : int

Deck

shuffle() : void

drawCard() : Card

GameLogic

deck : Deck

currentPlayer : Player

---

dealCards() : void

checkHandValue(hand : List<Card>) : int

determineOutcome() : void

shuffleDeck() : void

Dealer

hand : List<Card>

---

startRound() : void

autoDraw() : void

finishRound() : void

Server Requirements

- The server will designate a port for the Client to connect
- A single server will fork threads for single or multiple Clients to connect
  - This feature uses an inbuilt interface which allows multithreaded operation

Client Requirements

- Client will connect to the Server in a designated port
- One or many Clients can connect to the Server
  - Multithreading option helps accomplish this feature

Login Requirements

- Username and password will be validated with contents of the text file containing user data.
- The system will flag the user as a dealer or a player.

Player Requirements

- Since we want to maximize fair play, the player will only be able to bet at the start of the round.
- The player will have the options to hit, stand, or double down during the round.
- The player will be able to deposit funds to their bankroll.
- The player will not be able to bet more than what they have in their bankroll.

Dealer Requirements

- The Dealer will control the flow of the round
- The Dealer will start and end the round

*****we presented Sequence Diagram that we couldn't paste over to the slides*****