Team: Nels Anderson
Ehsan Karimi
Justin Visher
Cary Sullivan

Title: OO Poker
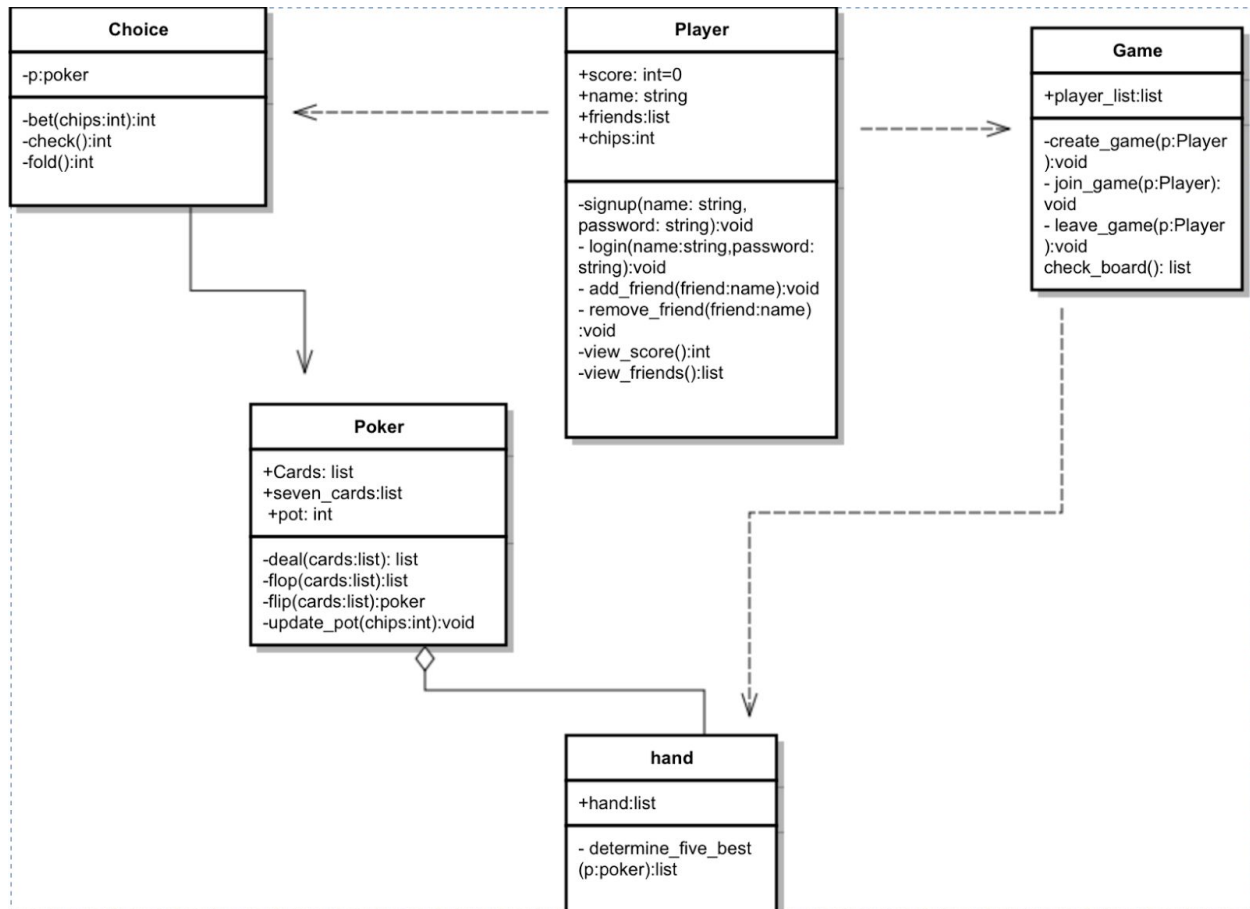
Refactoring and Design Pattern Choice:

Our team decided to employ the Observer design pattern as we revisited and refactored our system. This pattern focuses on setting up your system such that when an object's state is altered, it's dependents are also updated automatically.

The new implementation adds the observer feature to rate each player's hand in order to choose a winner. Additionally, mid-game when a player at the table folds, they will be able to see the current calculation of who is most likely to win depending on what they cards they have.

As can be seen in "New Class Diagram" we have added the observer, linking the hand and poker classes. Additionally, we have included two new classes, Hand Rate & Cards, to better modularize our system.

Old Class Diagram:

Choice

-p:poker

-bet(chips:int):int
-check():int
-fold():int

Player

+score: int=0
+name: string
+friends:list
+chips:int

-signup(name: string,
password: string):void
- login(name:string,password:
string):void
- add_friend(friend:name):void
- remove_friend(friend:name)
:void
-view_score():int
-view_friends():list

Game

+player_list:list

-create_game(p:Player
):void
- join_game(p:Player):
void
- leave_game(p:Player
):void
check_board(): list

Poker

+Cards: list
+seven_cards:list
 +pot: int

-deal(cards:list): list
-flop(cards:list):list
-flip(cards:list):poker
-update_pot(chips:int):void

hand

+hand:list

- determine_five_best
(p:poker):list

New Class Diagram:

Choice

-p:poker

-bet(chips:int):int
-check():int
-fold():int

Player

+score: int=0
+name: string
+friends:string[]
+chips:int

-signup(name: string,
password: string):void
- login(name:string,password:
string):void
- add_friend(friend:name):void
- remove_friend(friend:name)
:void
-view_score():int
-view_friends():string[]

Game

+player_list:list

-create_game(p:Player
):void
- join_game(p:Player):
void
- leave_game(p:Player
):void
check_board(): int[]

Cards

+type: string
+value: int

Poker

+Cards: cards[]
+seven_cards: cards[]
+pot: int
+observers:Observer[]

-deal(cards:Cards[],
ob:Observer): Cards[]
-flop(cards:Cards[],ob:
Observer):Cards[]
-flip(cards:Cards[],
ob:Observer):Cards[]
-update_pot(chips:int):void
+attach(in::observer)
+detach(in::observer)
+notify()

<<Interface>>
Observer

Attribute

+update():void

Hand

+hand: cards[]
+five_card: cards[]

- determine_five_best
(p:poker):Cards[]
- determine_hand_rate
():int
-update():void

Hand rate

-rate_hand(hand:Hand
):int