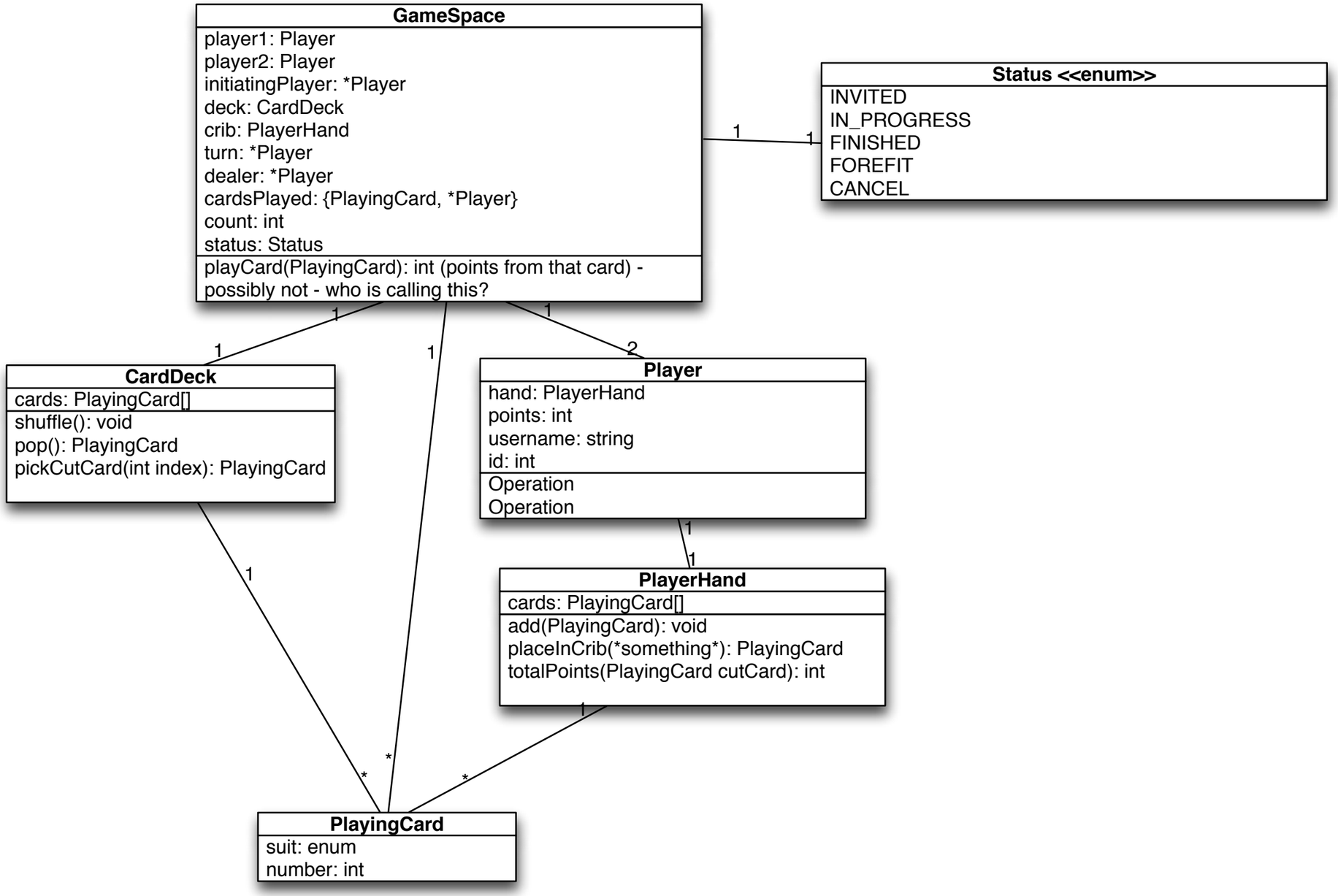
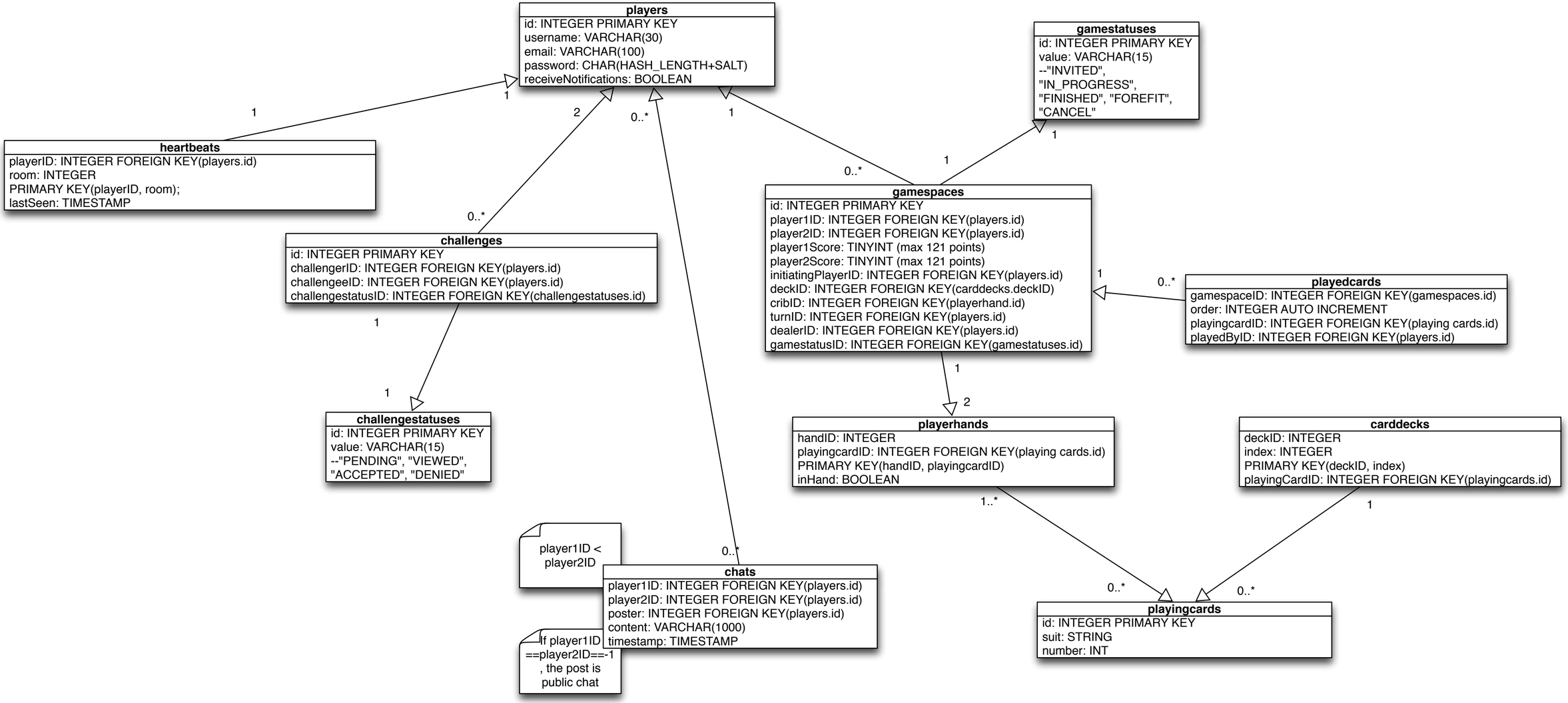
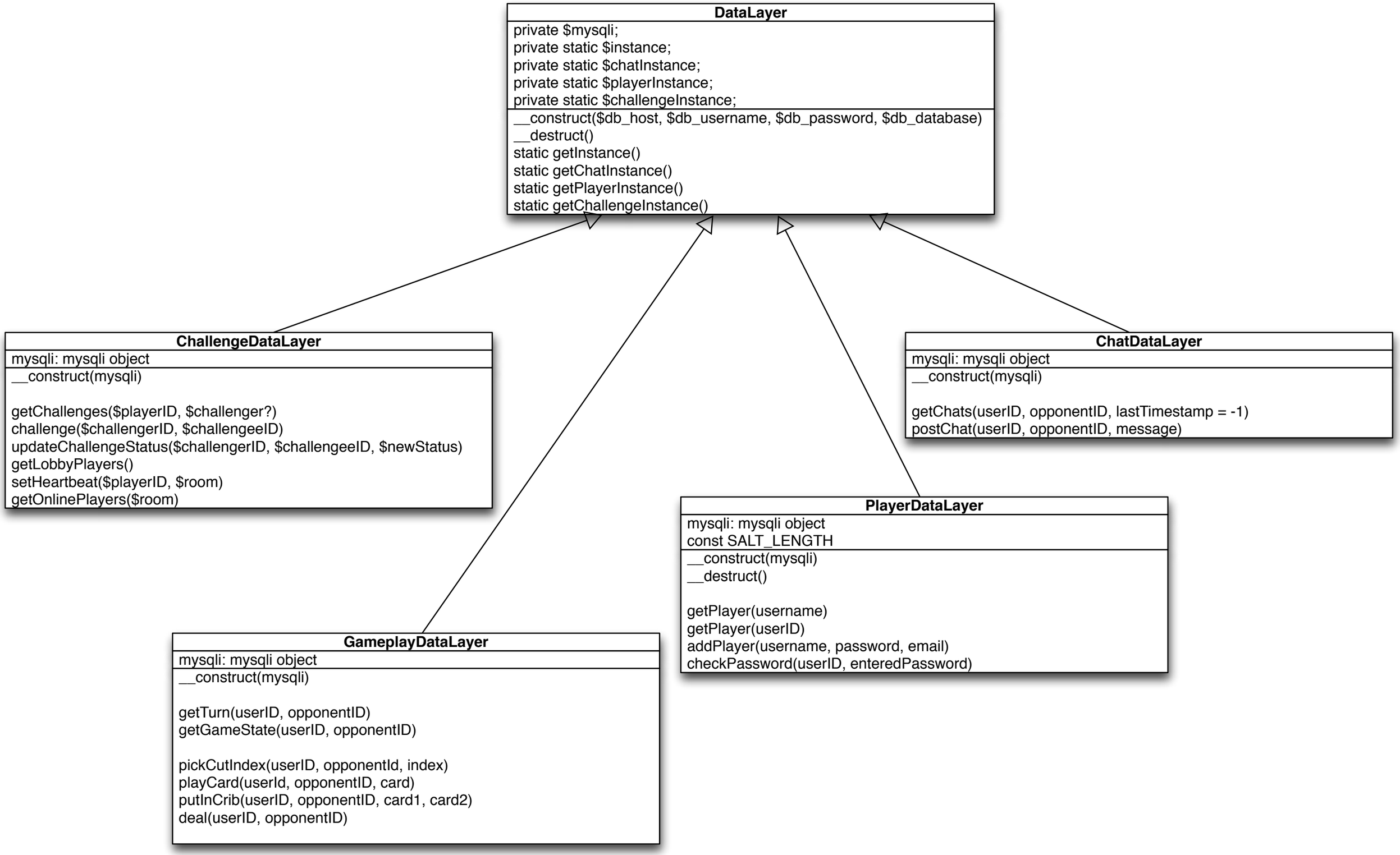

GameSpace
player1: Player
player2: Player
initiatingPlayer: *Player
deck: CardDeck
crib: PlayerHand
turn: *Player
dealer: *Player
cardsPlayed: {PlayingCard, *Player}
count: int
status: Status
playCard(PlayingCard): int (points from that card) -
possibly not - who is calling this?
Status <<enum>>
INVITED
IN_PROGRESS
FINISHED
FOREFIT
CANCEL
CardDeck
cards: PlayingCard[]
shuffle(): void
pop(): PlayingCard
pickCutCard(int index): PlayingCard
Player
hand: PlayerHand
points: int
username: string
id: int
Operation
Operation
PlayerHand
cards: PlayingCard[]
add(PlayingCard): void
placeInCrib(*something*): PlayingCard
totalPoints(PlayingCard cutCard): int
PlayingCard
suit: enum
number: int
1
1
1
1
1
2
1
1
1
1
1
*
*
*

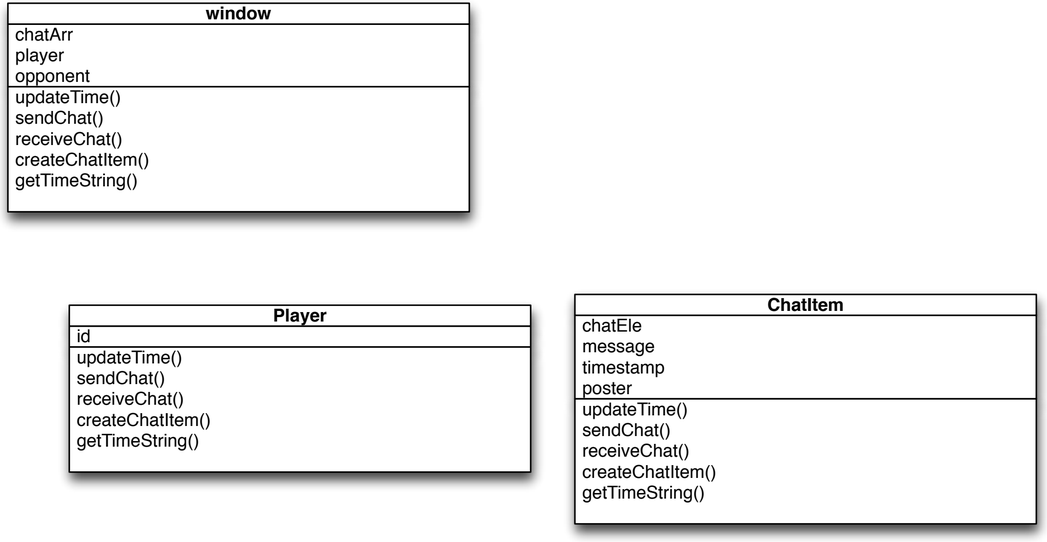

players
id: INTEGER PRIMARY KEY
username: VARCHAR(30)
email: VARCHAR(100)
password: CHAR(HASH_LENGTH+SALT)
receiveNotifications: BOOLEAN
gamestatuses
id: INTEGER PRIMARY KEY
value: VARCHAR(15)
--"INVITED",
"IN_PROGRESS",
"FINISHED", "FOREFIT",
"CANCEL"
heartbeats
playerID: INTEGER FOREIGN KEY(players.id)
room: INTEGER
PRIMARY KEY(playerID, room);
lastSeen: TIMESTAMP
challenges
id: INTEGER PRIMARY KEY
challengerID: INTEGER FOREIGN KEY(players.id)
challengeeID: INTEGER FOREIGN KEY(players.id)
challengestatusID: INTEGER FOREIGN KEY(challengestatuses.id)
gamespaces
id: INTEGER PRIMARY KEY
player1ID: INTEGER FOREIGN KEY(players.id)
player2ID: INTEGER FOREIGN KEY(players.id)
player1Score: TINYINT (max 121 points)
player2Score: TINYINT (max 121 points)
initiatingPlayerID: INTEGER FOREIGN KEY(players.id)
deckID: INTEGER FOREIGN KEY(carddecks.deckID)
cribID: INTEGER FOREIGN KEY(playerhand.id)
turnID: INTEGER FOREIGN KEY(players.id)
dealerID: INTEGER FOREIGN KEY(players.id)
gamestatusID: INTEGER FOREIGN KEY(gamestatuses.id)
playedcards
gamespaceID: INTEGER FOREIGN KEY(gamespaces.id)
order: INTEGER AUTO INCREMENT
playingcardID: INTEGER FOREIGN KEY(playing cards.id)
playedByID: INTEGER FOREIGN KEY(players.id)
challengestatuses
id: INTEGER PRIMARY KEY
value: VARCHAR(15)
--"PENDING", "VIEWED",
"ACCEPTED", "DENIED"
playerhands
handID: INTEGER
playingcardID: INTEGER FOREIGN KEY(playing cards.id)
PRIMARY KEY(handID, playingcardID)
inHand: BOOLEAN
carddecks
deckID: INTEGER
index: INTEGER
PRIMARY KEY(deckID, index)
playingCardID: INTEGER FOREIGN KEY(playingcards.id)
player1ID < player2ID
chats
player1ID: INTEGER FOREIGN KEY(players.id)
player2ID: INTEGER FOREIGN KEY(players.id)
poster: INTEGER FOREIGN KEY(players.id)
content: VARCHAR(1000)
timestamp: TIMESTAMP
If player1ID ==player2ID==-1 , the post is public chat
playingcards
id: INTEGER PRIMARY KEY
suit: STRING
number: INT
1
1
2
0..*
1
0..*
1
1
0..*
1
0..*
1
1
2
1..*
0..*
0..*
1
0..*

Ajax Interface

.

Application:
- chat
  - GET: getChat(userID, opponentID, lastTimestamp)
  - POST: postChat(userID, opponentID, message)
- game
  - GET: getTurn(userID, opponentID)
    - //Gets cards on table, cut card, hand, points
    - getGameState(userID, opponentID)
  - POST: pickCutIndex(userID, opponentID, index)
    - playCard(userID, opponentID, card)
    - putInCrib(userID, opponentID, card1, card2)
    - deal(userID, opponentID, numberOfShuffles)
- challenge
  - GET: getChallenges($playerID, $challenger = null)
    - getLobbyPlayers()
  - POST: challenge($challengerID, $challengeeID)
    - updateChallengeStatus($challengerID, $challengeeID, $newStatus)


DataLayer
private $mysqli;
private static $instance;
private static $chatInstance;
private static $playerInstance;
private static $challengeInstance;
__construct($db_host, $db_username, $db_password, $db_database)
__destruct()
static getInstance()
static getChatInstance()
static getPlayerInstance()
static getChallengeInstance()
ChallengeDataLayer
mysqli: mysqli object
__construct(mysqli)
getChallenges($playerID, $challenger?)
challenge($challengerID, $challengeeID)
updateChallengeStatus($challengerID, $challengeeID, $newStatus)
getLobbyPlayers()
setHeartbeat($playerID, $room)
getOnlinePlayers($room)
ChatDataLayer
mysqli: mysqli object
__construct(mysqli)
getChats(userID, opponentID, lastTimestamp = -1)
postChat(userID, opponentID, message)
PlayerDataLayer
mysqli: mysqli object
const SALT_LENGTH
__construct(mysqli)
__destruct()
getPlayer(username)
getPlayer(userID)
addPlayer(username, password, email)
checkPassword(userID, enteredPassword)
GameplayDataLayer
mysqli: mysqli object
__construct(mysqli)
getTurn(userID, opponentID)
getGameState(userID, opponentID)
pickCutIndex(userID, opponentId, index)
playCard(userId, opponentID, card)
putInCrib(userID, opponentID, card1, card2)
deal(userID, opponentID)


window
chatArr
player
opponent
updateTime()
sendChat()
receiveChat()
createChatItem()
getTimeString()
Player
id
updateTime()
sendChat()
receiveChat()
createChatItem()
getTimeString()
ChatItem
chatEle
message
timestamp
poster
updateTime()
sendChat()
receiveChat()
createChatItem()
getTimeString()

Deck

e

selectCutCard();
dealCards()

Scoreboard

pins[player1] = [{ele, int}, {ele, int}]
pins[player1] = [{ele, int}, {ele, int}]

constructor(player1Info, player2Info);

```
playerInfo = {
        "id" : playerID,
        "score": score,
        "backPinPosition": position
}
```

changeScore(player, newScore)
addPoints(player, points) - call ChangeScore()
private movePiece
private pieceCoordinates

PlayerHand

isCrib
cardArray

sendToCrib(PlayingCard)
play(PlayingCard)

PlayingCard

ele
x
y
suit
number
played

constructor(is)
isVisible()