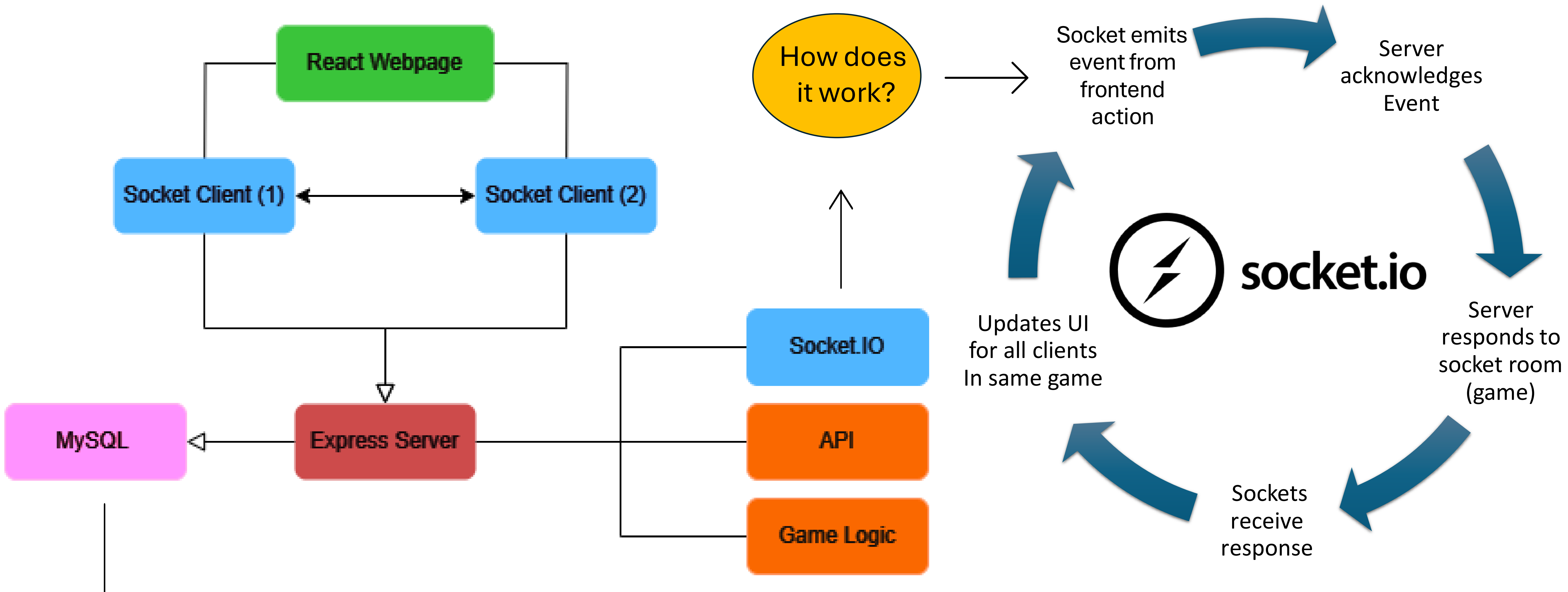
RamBluff

Sean Berlin ● Kirtan Chavda ● Jared Colletti ● Zachary Leopold

RamBluff is an exciting online multiplayer blackjack game where strategy and luck collide. Dive into easy-to-play but hard-to-master blackjack action that's perfect for both beginners and seasoned players. With RamBluff, you can challenge friends or players from around the world in real-time. Get ready for fast-paced matches, clever hits, and the thrill of the win. Join RamBluff now and prove you have what it takes to be a blackjack legend!



React Webpage

Socket Client (1)

Socket Client (2)

Express Server

MySQL

Socket.IO

API

Game Logic

How does it work?

Socket emits event from frontend action

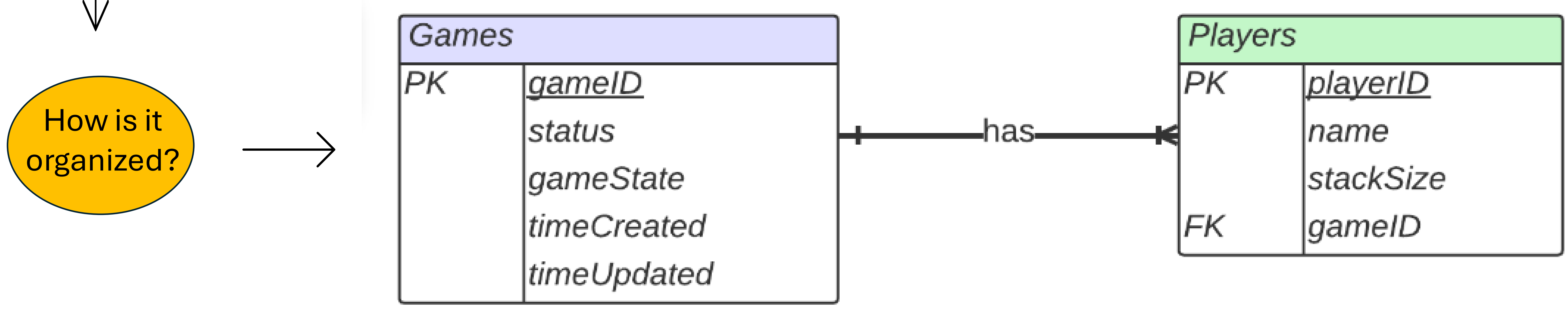
Server acknowledges Event

Server responds to socket room (game)

Sockets receive response

Updates UI for all clients In same game

How is it organized?



Games	
PK	gameID
	status
	gameState
	timeCreated
	timeUpdated

has

Players	
PK	playerID
	name
	stackSize
FK	gameID

React

- Page / Component Rendering
- Game Initialization
- Socket Initialization
- API Communication
- Player / Game Actions
- Event Handling

node

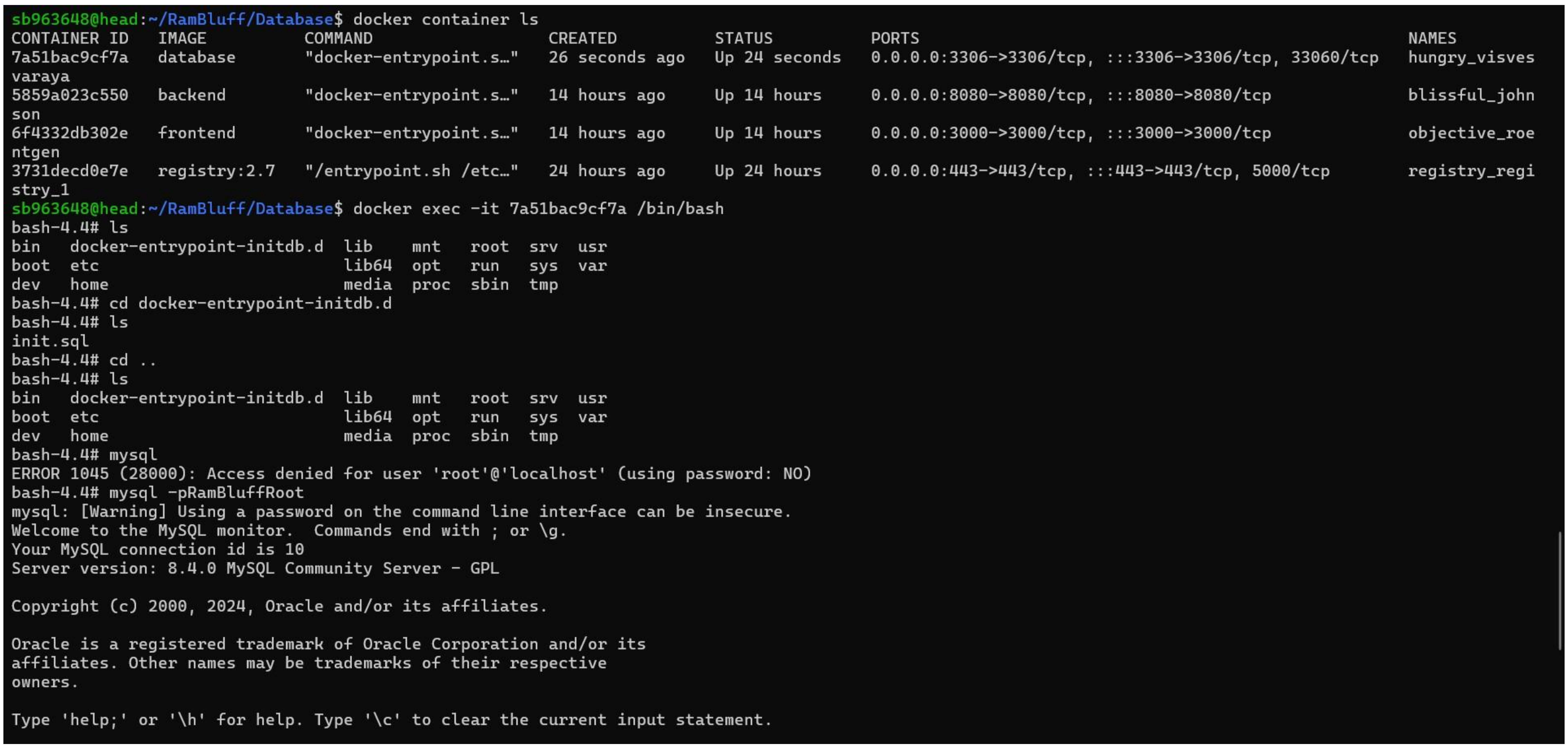
- API Integration
- Database Management / Communication
- Socket IO Implementation
- Game State Evaluation
- Server Initialization

- Game State / Player Definitions
- Round Progression
- Winning Condition Evaluation
- Ledger Calculation
- Error Handling / Hand Validation

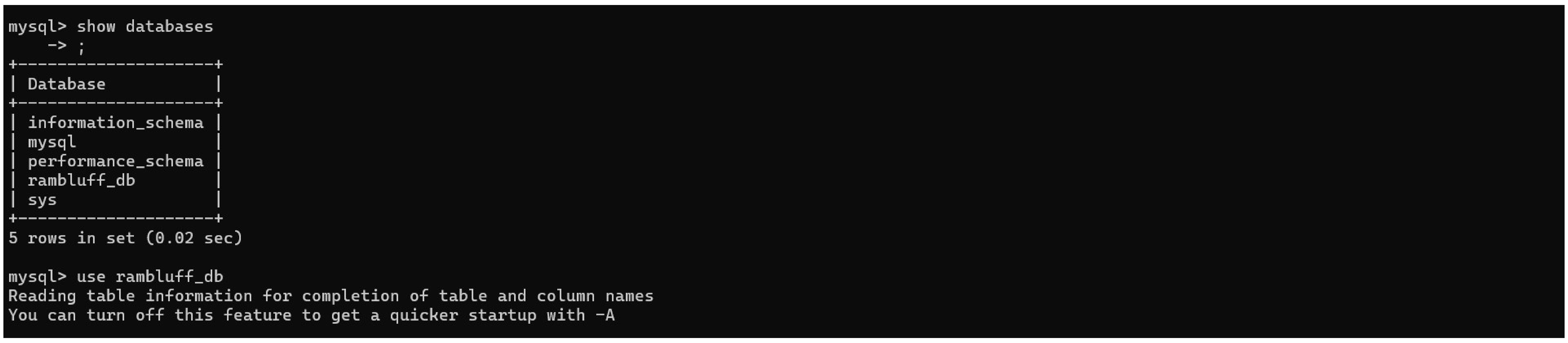
MySQL

- Game / Player Storage
- Game State Mutations
- Game Model
- Player Model
- Data Models are Predefined in Backend

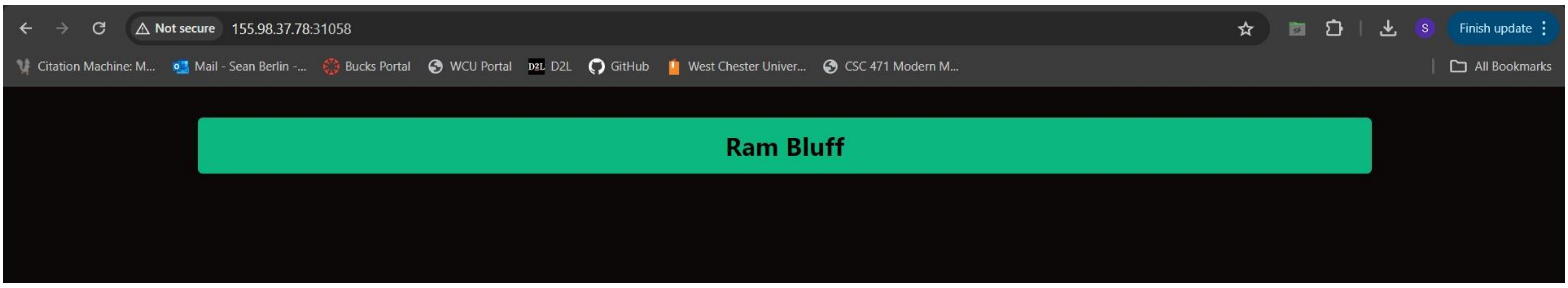
CI/CD



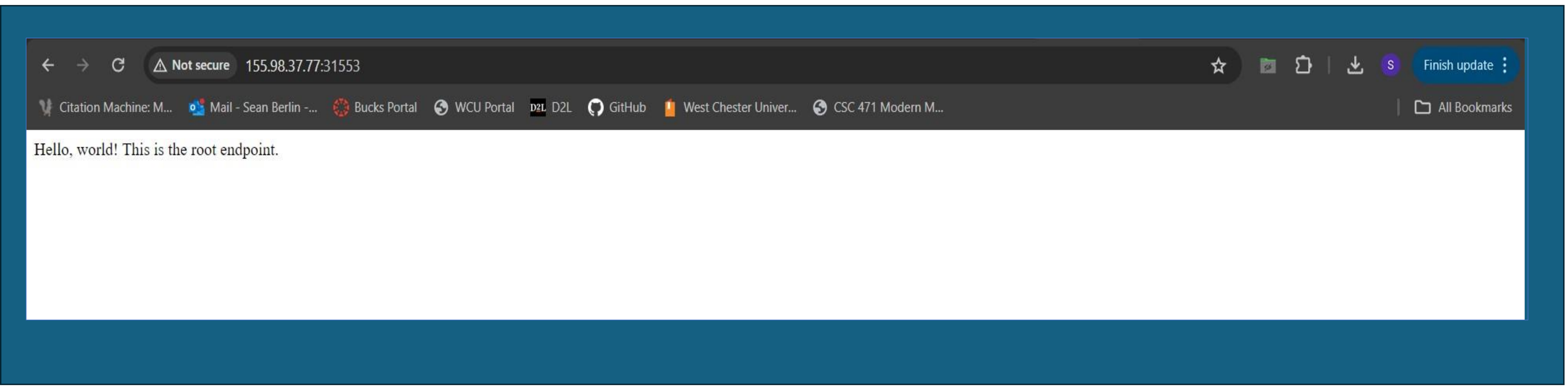
```
sb963648@head:~/RamBluff/Database$ docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                                                  NAMES
7a51bac9cf7a   database       "docker-entrypoint.s…"   26 seconds ago   Up 24 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   hungry_visves
varaya
5859a023c550   backend        "docker-entrypoint.s…"   14 hours ago     Up 14 hours     0.0.0.0:8080->8080/tcp, :::8080->8080/tcp              blissful_john
son
6f4332db302e   frontend       "docker-entrypoint.s…"   14 hours ago     Up 14 hours     0.0.0.0:3000->3000/tcp, :::3000->3000/tcp              objective_roe
ntgen
3731decd0e7e   registry:2.7   "/entrypoint.sh /etc…"   24 hours ago     Up 24 hours     0.0.0.0:443->443/tcp, :::443->443/tcp, 5000/tcp        registry_regi
stry_1
sb963648@head:~/RamBluff/Database$ docker exec -it 7a51bac9cf7a /bin/bash
bash-4.4# ls
bin   docker-entrypoint-initdb.d  lib    mnt   root  srv  usr
boot  etc                         lib64  opt   run   sys  var
dev   home                        media  proc  sbin  tmp
bash-4.4# cd docker-entrypoint-initdb.d
bash-4.4# ls
init.sql
bash-4.4# cd ..
bash-4.4# ls
bin   docker-entrypoint-initdb.d  lib    mnt   root  srv  usr
boot  etc                         lib64  opt   run   sys  var
dev   home                        media  proc  sbin  tmp
bash-4.4# mysql
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)
bash-4.4# mysql -pRamBluffRoot
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.4.0 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```



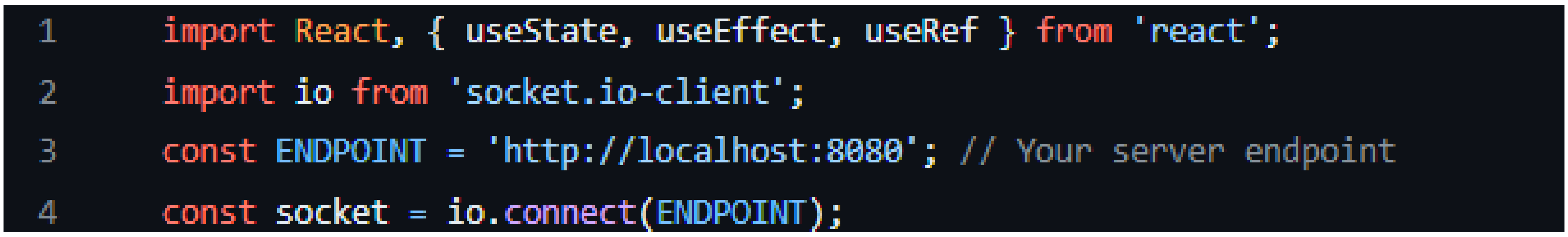
```
mysql> show databases
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| rambluff_db        |
| sys                |
+--------------------+
5 rows in set (0.02 sec)

mysql> use rambluff_db
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```



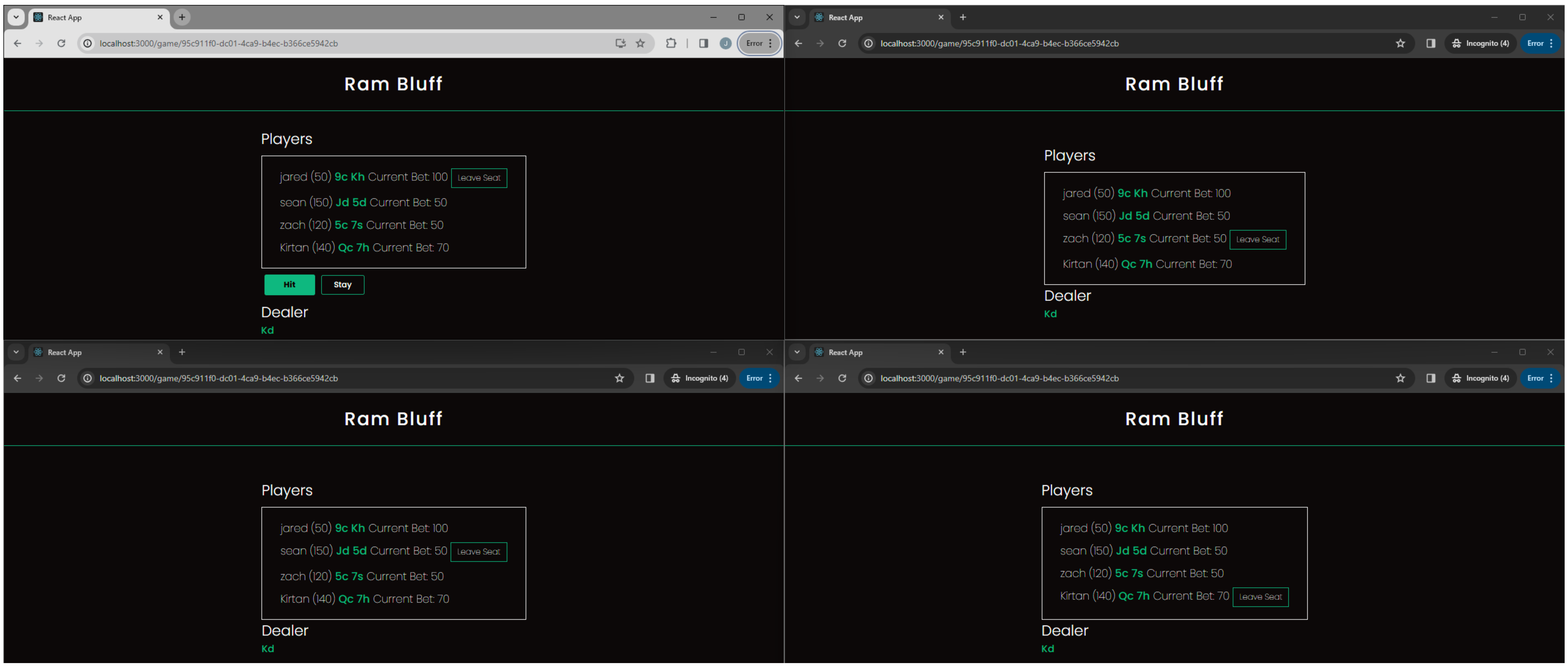
Ram Bluff



Hello, world! This is the root endpoint.



```
import React, { useState, useEffect, useRef } from 'react';
import io from 'socket.io-client';
const ENDPOINT = 'http://localhost:8080'; // Your server endpoint
const socket = io.connect(ENDPOINT);
```



Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100 Leave Seat
sean (150) **Jd 5d** Current Bet: 50
zach (120) **5c 7s** Current Bet: 50
Kirtan (140) **Qc 7h** Current Bet: 70

Hit Stay

Dealer
Kd

Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100
sean (150) **Jd 5d** Current Bet: 50
zach (120) **5c 7s** Current Bet: 50 Leave Seat
Kirtan (140) **Qc 7h** Current Bet: 70

Dealer
Kd

Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100
sean (150) **Jd 5d** Current Bet: 50 Leave Seat
zach (120) **5c 7s** Current Bet: 50
Kirtan (140) **Qc 7h** Current Bet: 70

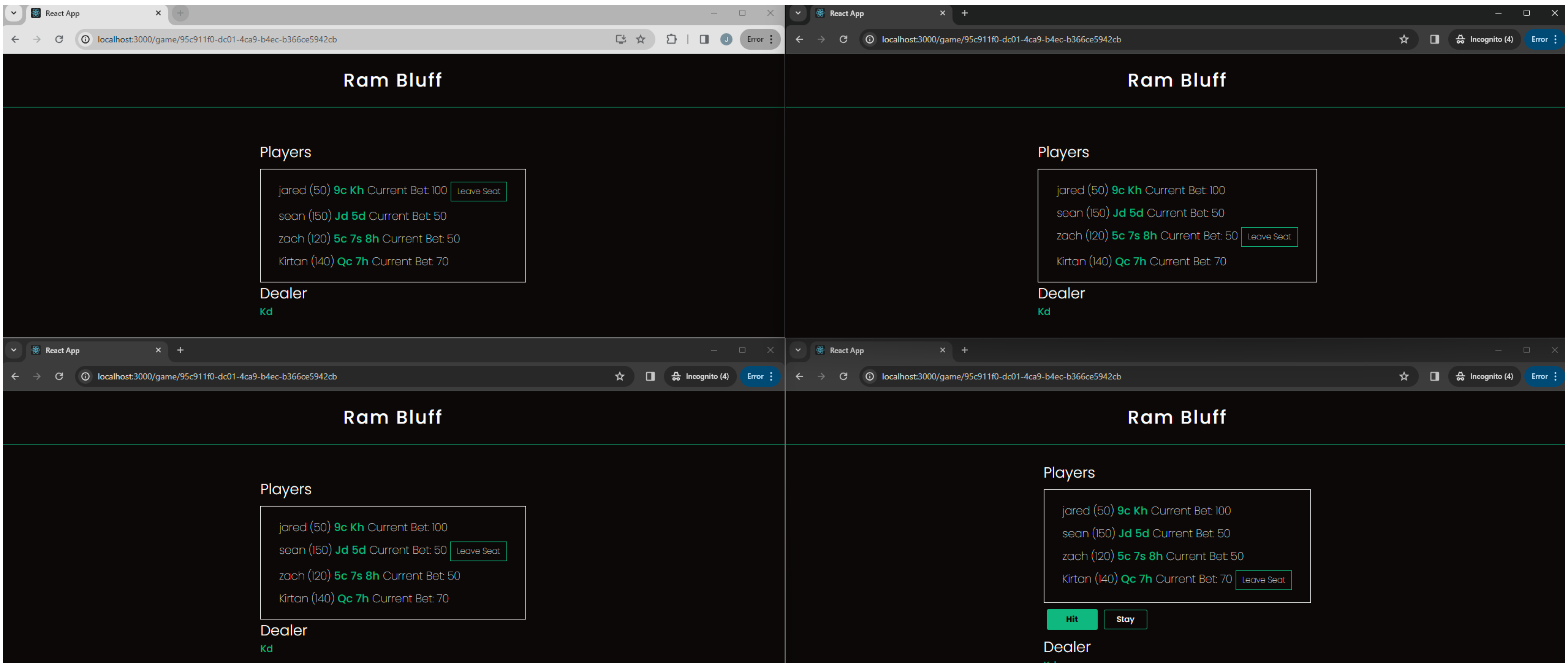
Dealer
Kd

Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100
sean (150) **Jd 5d** Current Bet: 50
zach (120) **5c 7s** Current Bet: 50
Kirtan (140) **Qc 7h** Current Bet: 70 Leave Seat

Dealer
Kd



Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100 Leave Seat
sean (150) **Jd 5d** Current Bet: 50
zach (120) **5c 7s 8h** Current Bet: 50
Kirtan (140) **Qc 7h** Current Bet: 70

Dealer
Kd

Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100
sean (150) **Jd 5d** Current Bet: 50
zach (120) **5c 7s 8h** Current Bet: 50 Leave Seat
Kirtan (140) **Qc 7h** Current Bet: 70

Dealer
Kd

Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100
sean (150) **Jd 5d** Current Bet: 50 Leave Seat
zach (120) **5c 7s 8h** Current Bet: 50
Kirtan (140) **Qc 7h** Current Bet: 70

Dealer
Kd

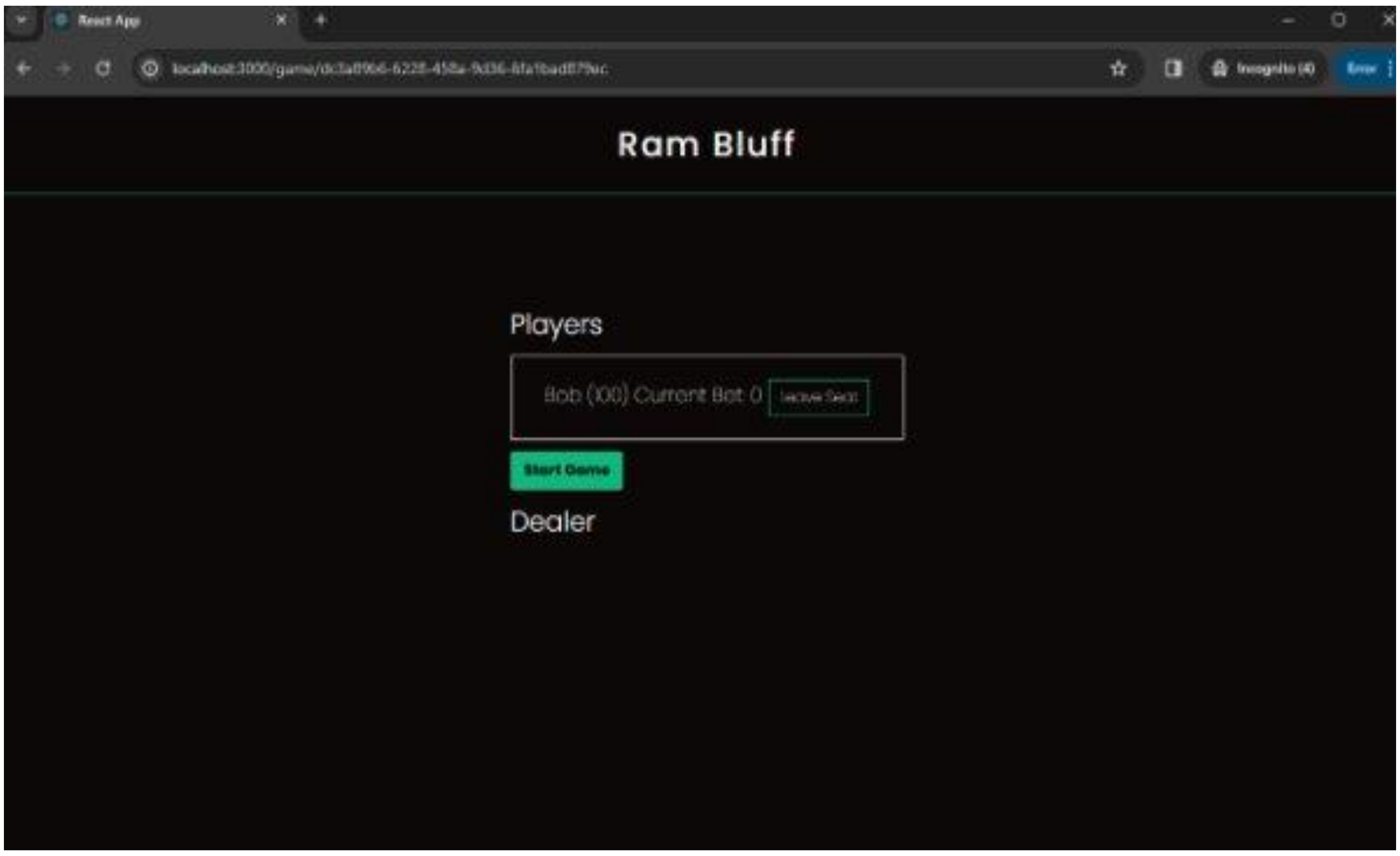
Ram Bluff

Players

jared (50) **9c Kh** Current Bet: 100
sean (150) **Jd 5d** Current Bet: 50
zach (120) **5c 7s 8h** Current Bet: 50
Kirtan (140) **Qc 7h** Current Bet: 70 Leave Seat

Hit Stay

Dealer

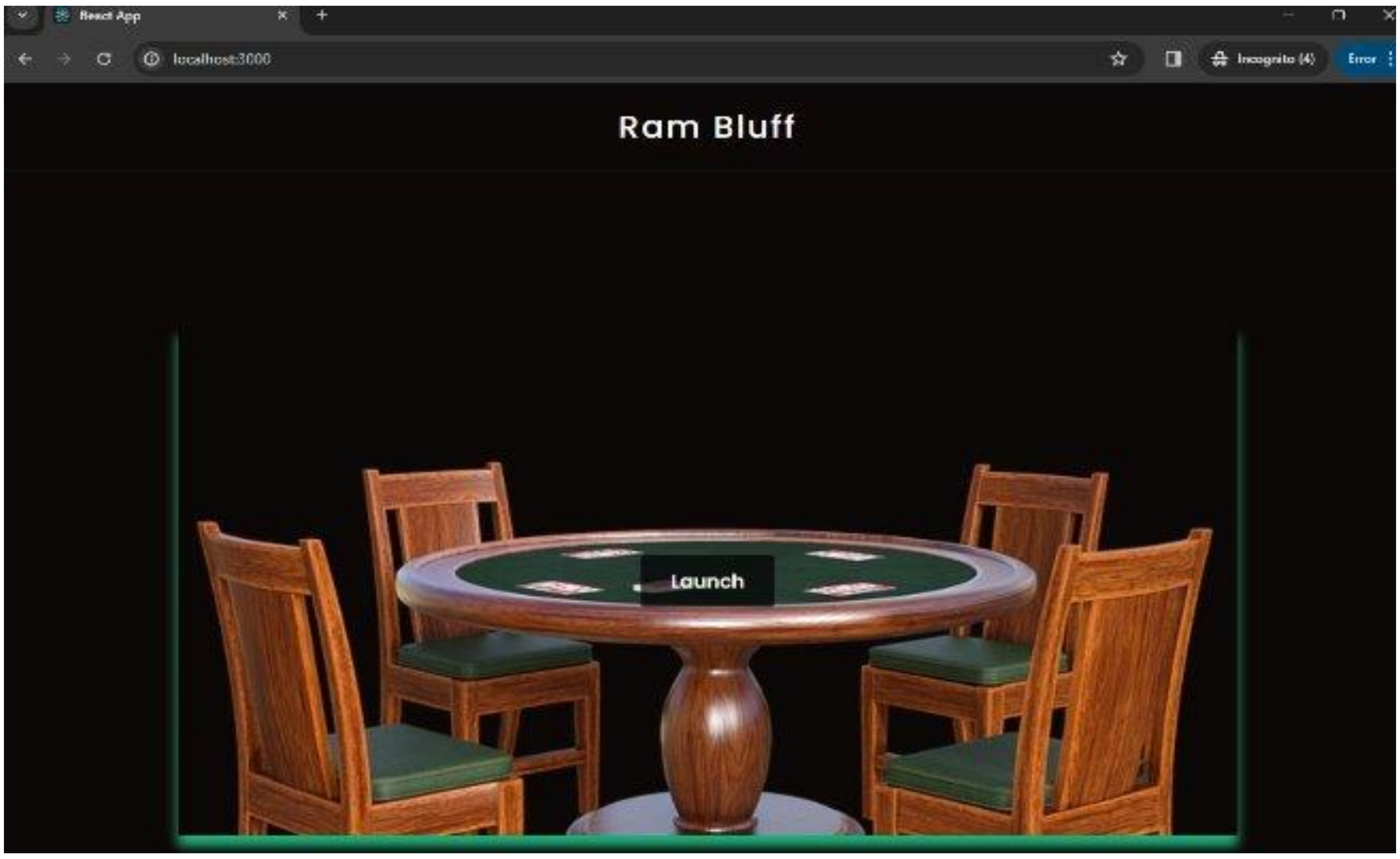


Ram Bluff

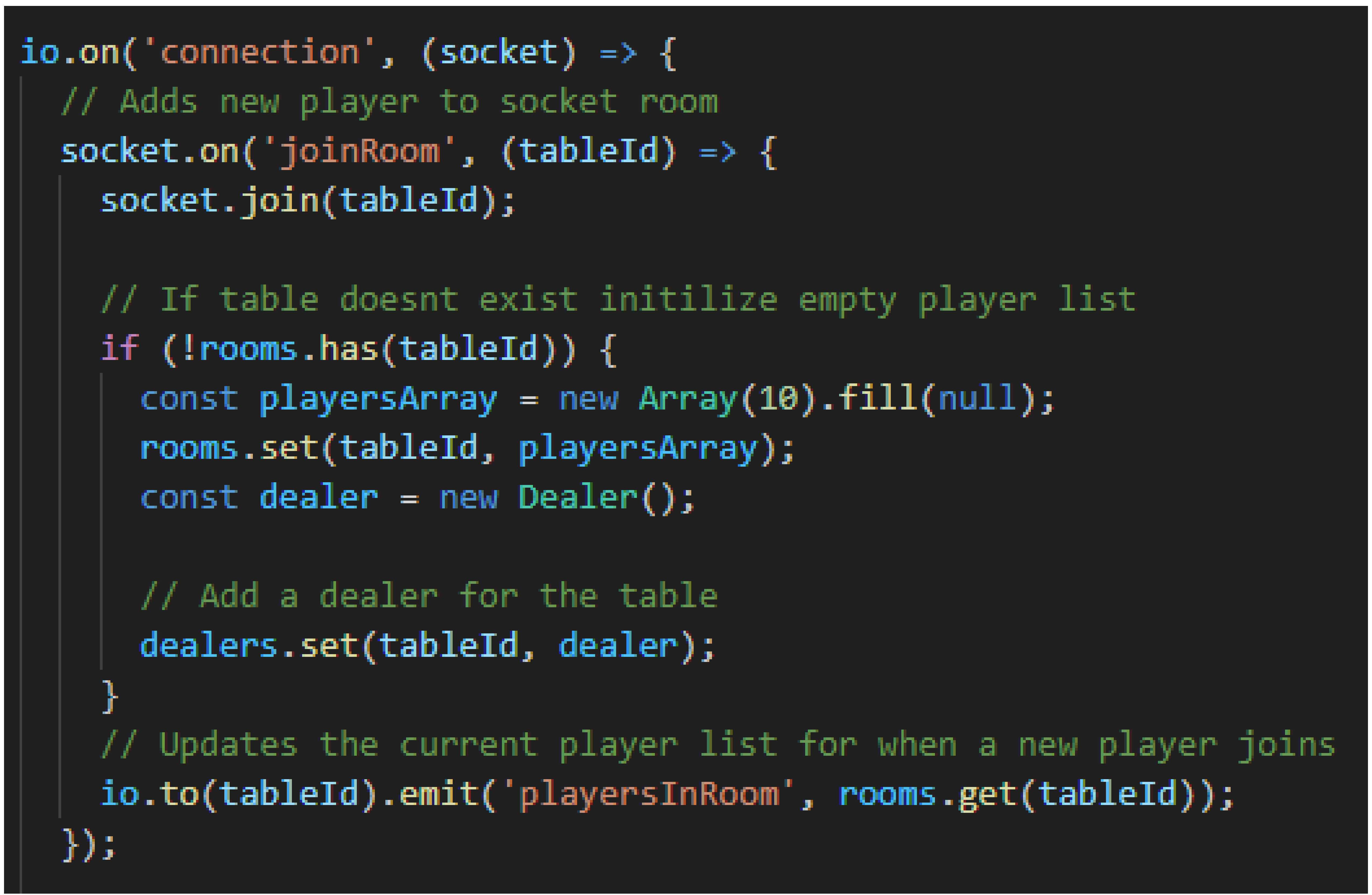
Players

Bob (100) Current Bet: 0

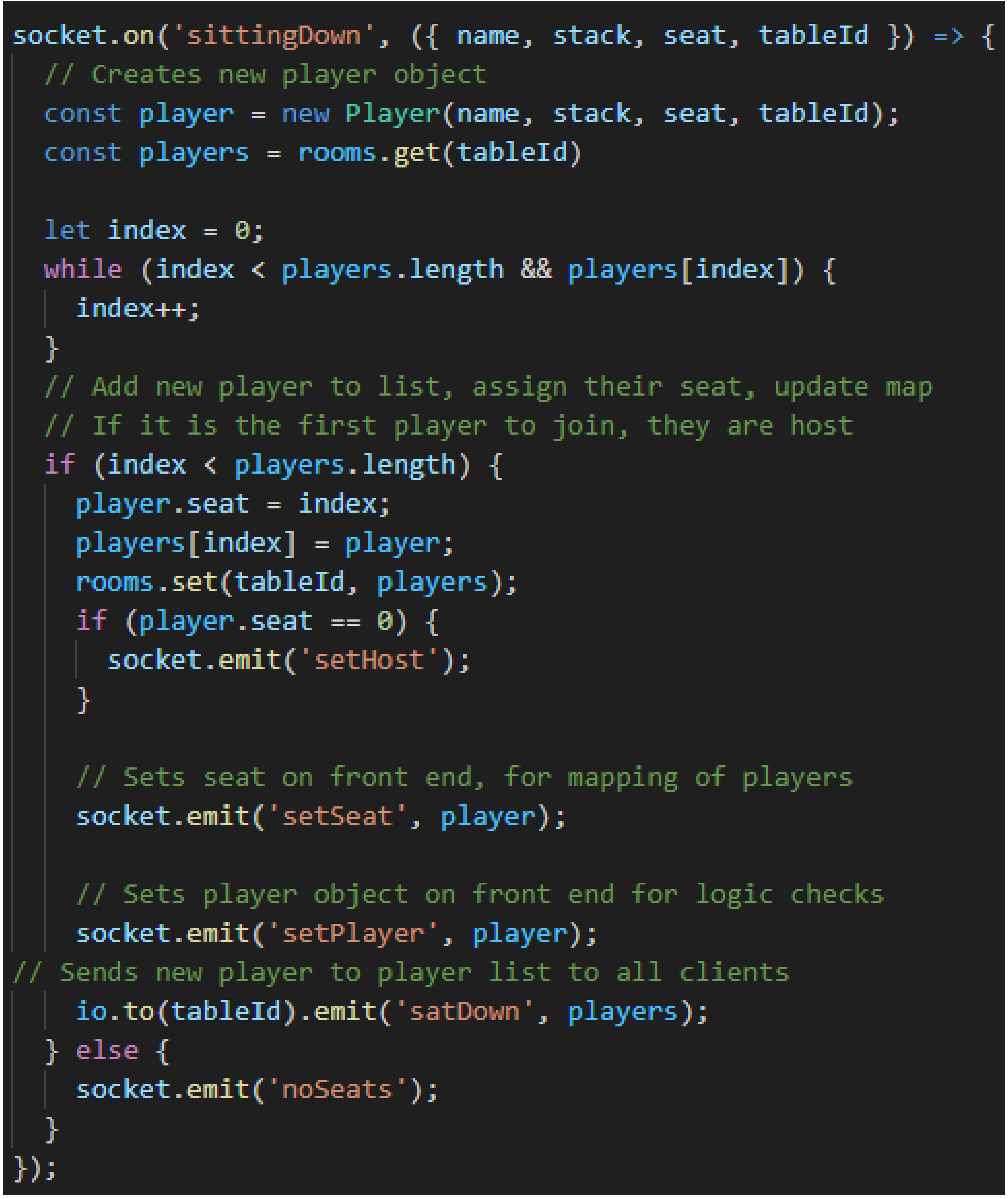
Dealer



Ram Bluff



```
io.on('connection', (socket) => {
  // Adds new player to socket room
  socket.on('joinRoom', (tableId) => {
    socket.join(tableId);

    // If table doesnt exist initilize empty player list
    if (!rooms.has(tableId)) {
      const playersArray = new Array(10).fill(null);
      rooms.set(tableId, playersArray);
      const dealer = new Dealer();

      // Add a dealer for the table
      dealers.set(tableId, dealer);
    }
    // Updates the current player list for when a new player joins
    io.to(tableId).emit('playersInRoom', rooms.get(tableId));
  });
```



```
socket.on('sittingDown', ({ name, stack, seat, tableId }) => {
  // Creates new player object
  const player = new Player(name, stack, seat, tableId);
  const players = rooms.get(tableId)

  let index = 0;
  while (index < players.length && players[index]) {
    index++;
  }
  // Add new player to list, assign their seat, update map
  // If it is the first player to join, they are host
  if (index < players.length) {
    player.seat = index;
    players[index] = player;
    rooms.set(tableId, players);
    if (player.seat == 0) {
      socket.emit('setHost');
    }

    // Sets seat on front end, for mapping of players
    socket.emit('setSeat', player);

    // Sets player object on front end for logic checks
    socket.emit('setPlayer', player);
// Sends new player to player list to all clients
    io.to(tableId).emit('satDown', players);
  } else {
    socket.emit('noSeats');
  }
});
```