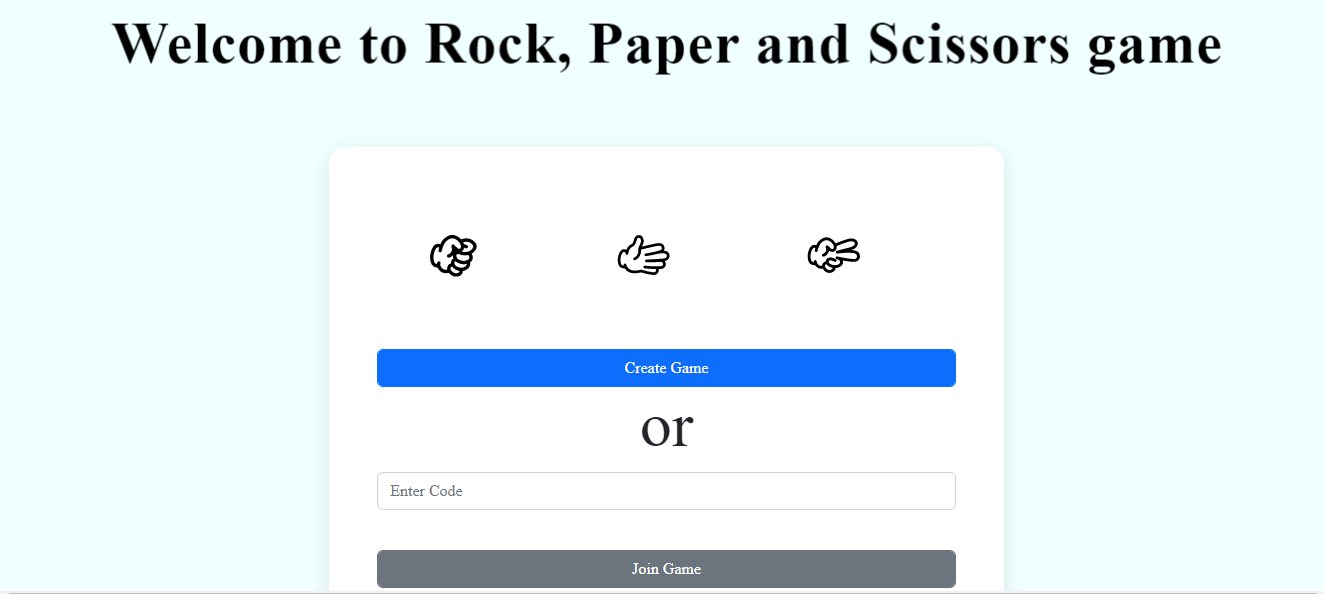
**CN Project Report**

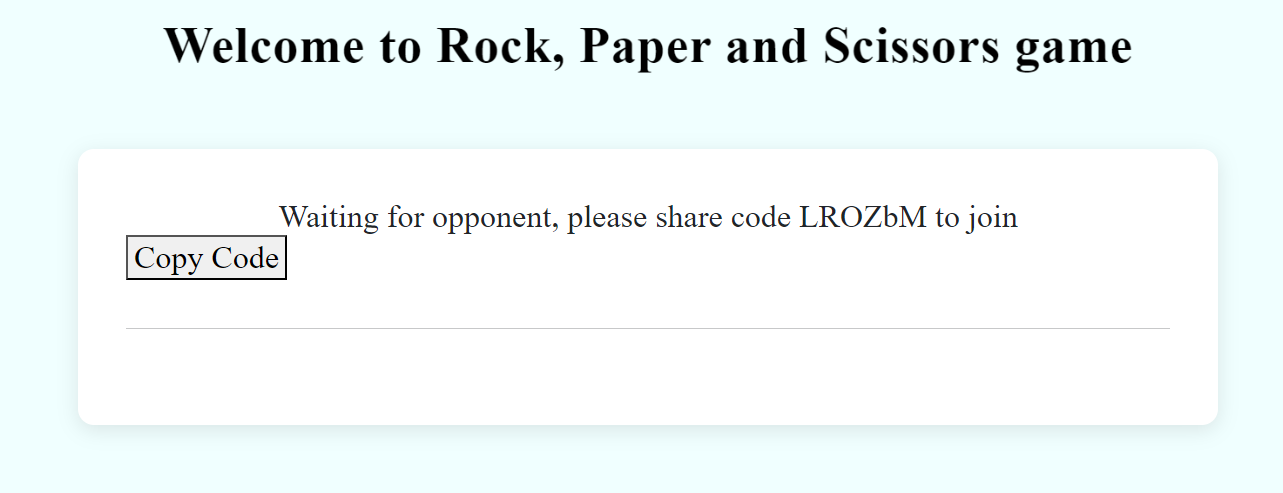
**Project Members:**

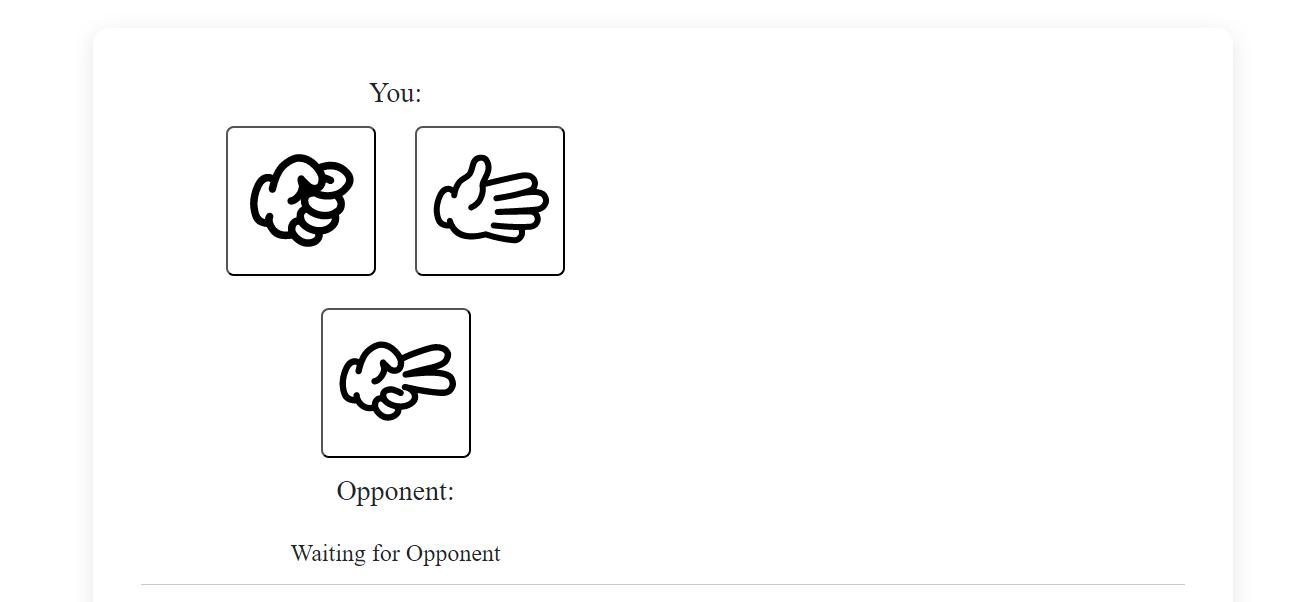
**Nashit Budhwani (20K-0274)  
Ali Hassan Mughal (20K-0264)**

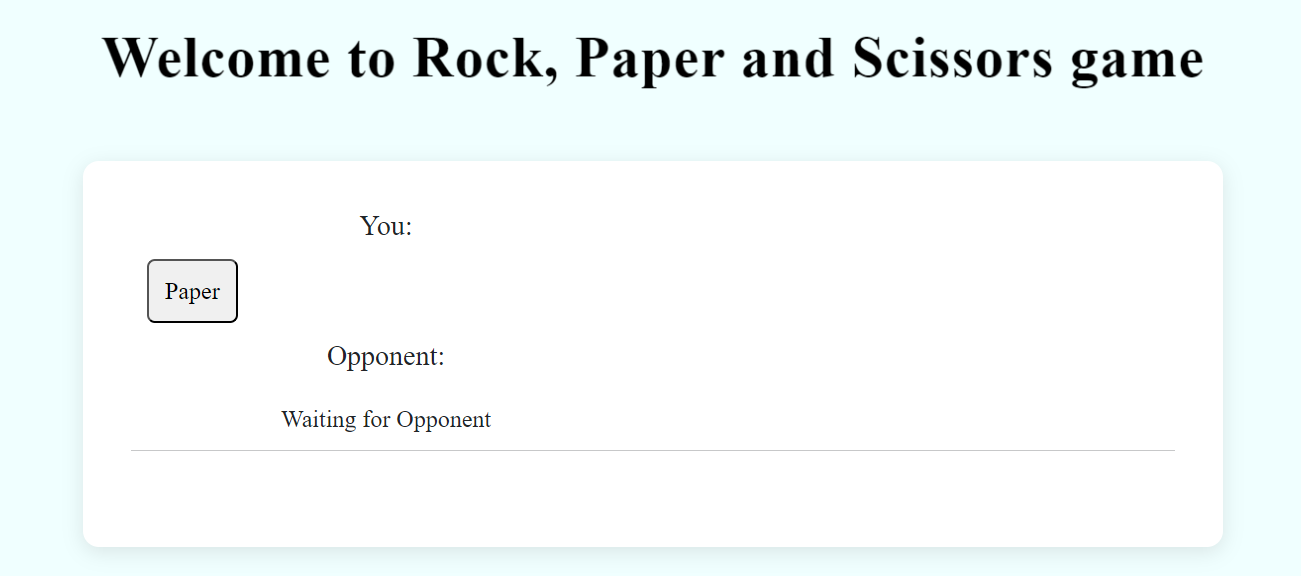
**Project Snippets:**

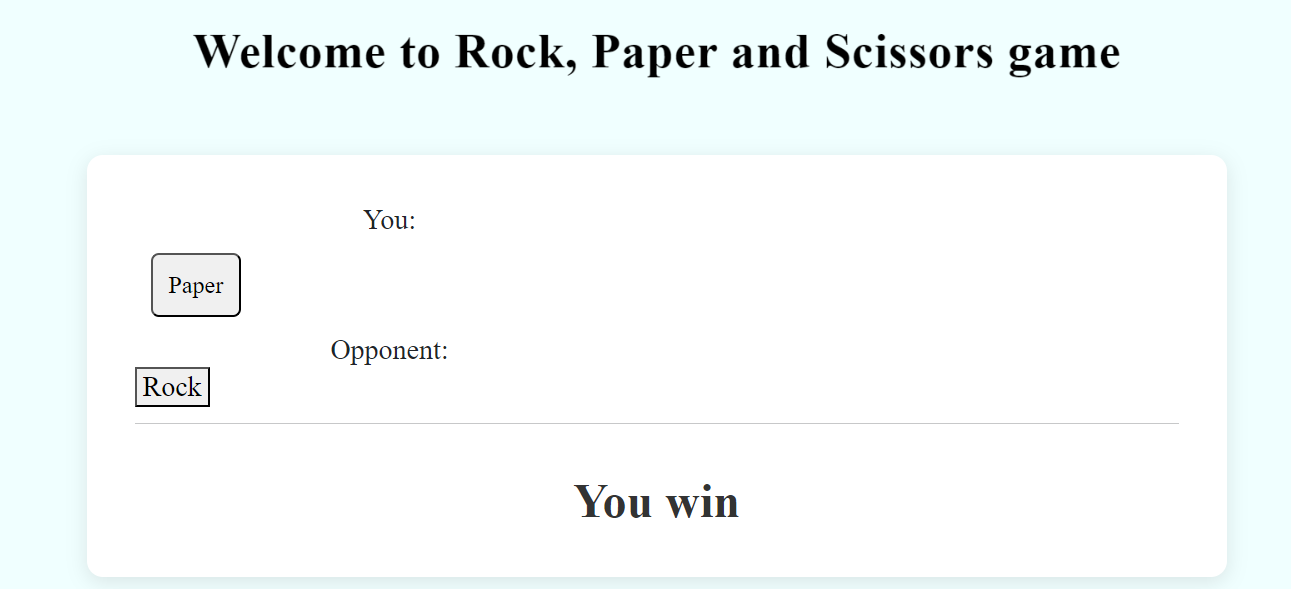
**Player 1:**

****

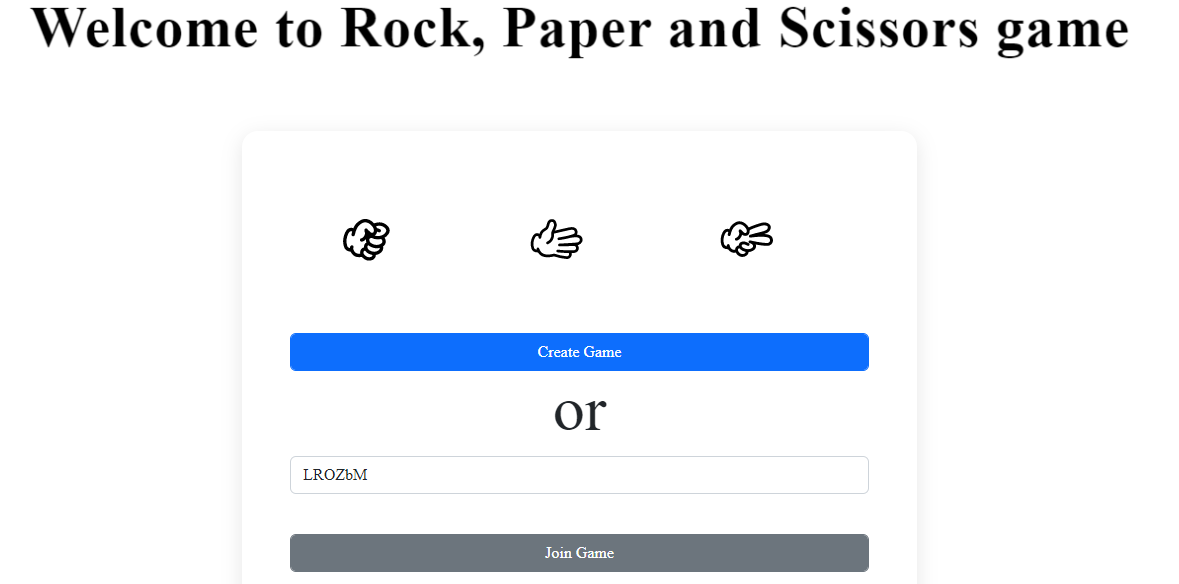
****

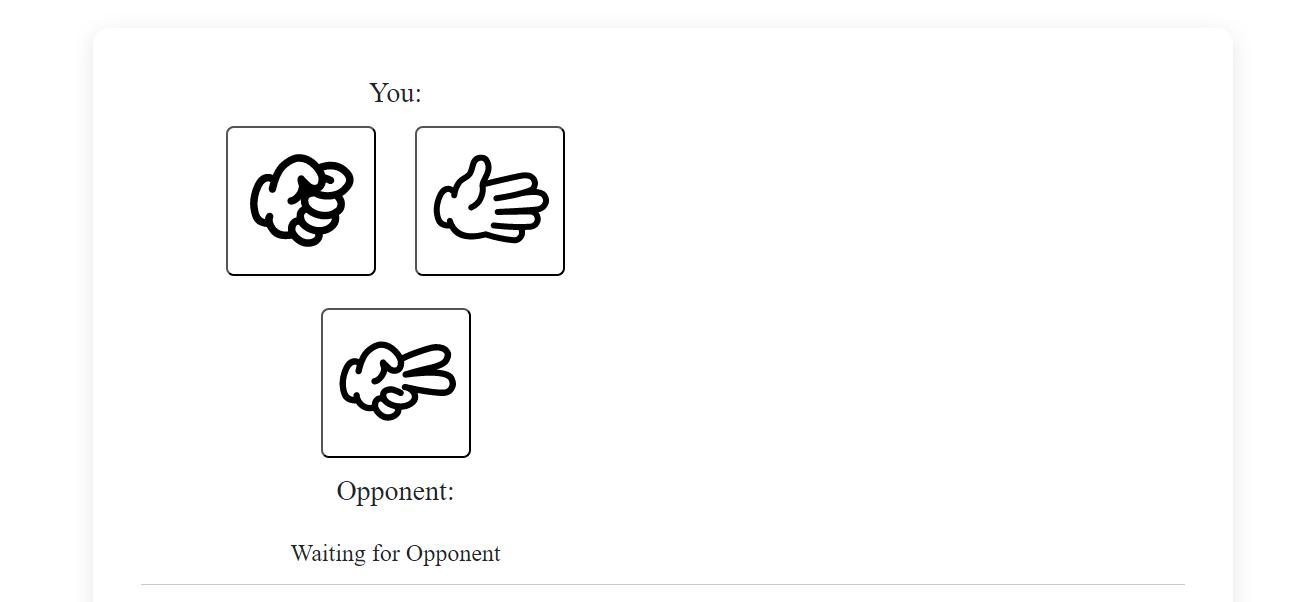
****

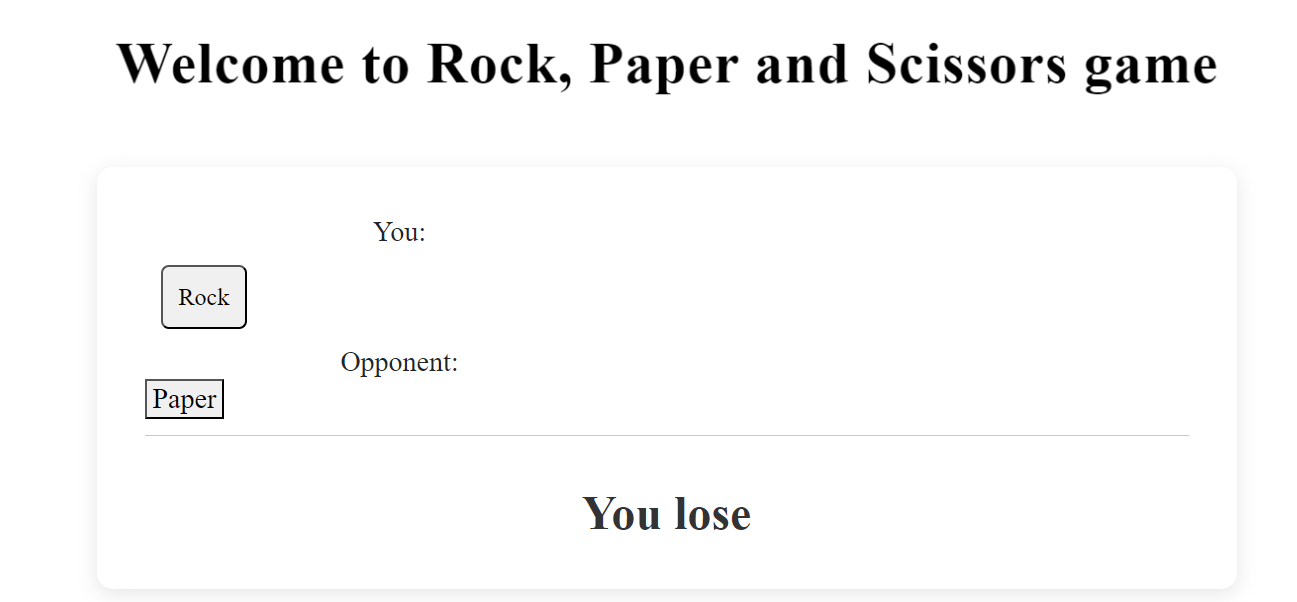
****

****

**Player 2:**

****





**Introduction:**

The rock-paper-scissors multiplayer game using UDP is a simple application that allows two players to play rock-paper-scissors against each other in real-time. The game is built on top of a client-server architecture where the clients communicate with the server via UDP. The game server is implemented using Node.js and the Socket.io library. The game client is implemented using HTML, CSS, and JavaScript, and communicates with the server using the Socket.io client library.

**Server Implementation**:

The server-side of the application is implemented using Node.js and the Socket.io library. The server-side code is responsible for creating a unique room for the players to connect to and also for handling the game logic. When a player creates a new game, the server generates a unique room ID using the makeid function, stores the room ID in a rooms object, and makes the player the first player of the game. When a player joins a game, the server checks if the room ID is valid and if it is, adds the player to the room and marks the player as the second player of the game.

Once both players have joined the room, the game begins. The players can choose rock, paper, or scissors by clicking the corresponding button on their screen. When a player makes a choice, the client sends the choice to the server, and the server stores the choice in the rooms object. If both players have made a choice, the server determines the winner and sends the result back to the clients.

**Code:**

const express = require('express');

const app = express();

const http = require('http');

const path = require('path')

const server = http.createServer(app);

const { Server } = require("socket.io");

const io= new Server(server);

const rooms = {};

app.use(express.static(path.join(\_\_dirname,'client')));

app.get('/healthcheck',(req,res)=>{

res.send('<h1>RPS App running...</h1>');

});

app.get('/',(req,res)=>{

res.sendFile(\_\_dirname + '/client/index.html');

});

function makeid(length) {

let result = '';

const characters = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';

const charactersLength = characters.length;

let counter = 0;

while (counter < length) {

result += characters.charAt(Math.floor(Math.random() \* charactersLength));

counter += 1;

}

return result;

}

io.on('connection',(socket)=>{

console.log('A user in connected');

socket.on('disconnect',()=>{

console.log('A user disconnected');

});

socket.on('createGame',()=>{

const roomUniqueId=makeid(6)

rooms[roomUniqueId]={};

socket.join(roomUniqueId);

socket.emit("newGame",{roomUniqueId: roomUniqueId})

});

socket.on('joinGame',(data)=>{

if(rooms[data.roomUniqueId]!=null){

socket.join(data.roomUniqueId);

socket.to(data.roomUniqueId).emit("playersConnected",{});

socket.emit("playersConnected");

}

});

socket.on("p1Choice",(data) => {

let rpsValue= data.rpsValue;

rooms[data.roomUniqueId].p1Choice = rpsValue;

socket.to(data.roomUniqueId).emit("p1Choice",{rpsValue : data.rpsValue});

if(rooms[data.roomUniqueId].p2Choice!=null){

delcareWinner(data.roomUniqueId);

}

});

socket.on("p2Choice",(data) => {

let rpsValue= data.rpsValue;

rooms[data.roomUniqueId].p2Choice = rpsValue;

socket.to(data.roomUniqueId).emit("p2Choice",{rpsValue : data.rpsValue});

if(rooms[data.roomUniqueId].p1Choice!=null){

delcareWinner(data.roomUniqueId);

}

});

});

function delcareWinner(roomUniqueId){

let p1Choice = rooms[roomUniqueId].p1Choice;

let p2Choice = rooms[roomUniqueId].p2Choice;

let winner = null;

if(p1Choice == p2Choice){

winner = "d";

} else if (p1Choice=="Paper"){

if(p2Choice == "Scissor"){

winner= "p2";

} else {

winner = "p1"

}

} else if (p1Choice=="Rock"){

if(p2Choice == "Paper"){

winner= "p2";

} else {

winner = "p1"

}

} else if (p1Choice=="Scissor"){

if(p2Choice == "Rock"){

winner= "p2";

} else {

winner = "p1"

}

}

io.sockets.to(roomUniqueId).emit("result",{

winner : winner

});

rooms[roomUniqueId],p1Choice=null;

rooms[roomUniqueId],p2Choice=null;

}

server.listen(3000,()=>{

console.log('listening on \*:3000');

});

**Client Implementation:**

The client-side of the application is implemented using HTML, CSS, and JavaScript. When a player opens the game in their web browser, they are presented with the option to create a new game or join an existing game by entering a room ID. When a player creates a new game, the client sends a createGame message to the server, and when a player joins a game, the client sends a joinGame message to the server along with the room ID. Once the player has joined a game, they are presented with the game screen.

On the game screen, the player can choose rock, paper, or scissors by clicking the corresponding button. When the player makes a choice, the client sends the choice to the server along with the room ID. The client also displays the player's choice on the screen. When the server determines the winner, the client displays the result on the screen.

**Code:**

console.log('Client.js executing');

var socket = io();

let roomUniqueId = null;

let player1=false;

function createGame(){

player1=true;

socket.emit('createGame');

}

function joinGame(){

roomUniqueId=document.getElementById('roomUniqueId').value;

socket.emit('joinGame',{roomUniqueId: roomUniqueId});

}

socket.on("newGame", (data)=>{

roomUniqueId = data.roomUniqueId;

document.getElementById('initial').style.display = 'none';

document.getElementById('gamePlay').style.display = 'block';

let copyButton = document.createElement('button');

copyButton.style.display= 'block';

copyButton.innerText= 'Copy Code';

copyButton.addEventListener('click',()=>{

navigator.clipboard.writeText(roomUniqueId).then(function() {

console.log('Async: Copying to clipboard was successful!');

}, function(err) {

console.error('Asyn: Could not copy text: ',err);

});

});

document.getElementById('waitingArea').innerHTML=`Waiting for opponent, please share code ${roomUniqueId} to join`;

document.getElementById('waitingArea').appendChild(copyButton);

});

socket.on("playersConnected", ()=>{

document.getElementById('initial').style.display = 'none';

document.getElementById('waitingArea').style.display = 'none';

document.getElementById('gameArea').style.display = 'block';

});

socket.on("p1Choice",(data)=>{

if(!player1){

createOpponentChoiceButton(data);

}

});

socket.on("p2Choice",(data)=>{

if(player1){

createOpponentChoiceButton(data);

}

});

socket.on("result",(data)=>{

let winnerText = '';

if(data.winner != 'd'){

if (data.winner == 'p1' && player1){

winnerText = 'You win';

} else if (data.winner == 'p1'){

winnerText = 'You lose';

} else if (data.winner == 'p2' && !player1){

winnerText = 'You Win';

} else if (data.winner == 'p2'){

winnerText = 'You lose';

}

} else {

winnerText = `It's a Draw`;

}

document.getElementById('opponentState').style.display = 'none';

document.getElementById('opponentButton').style.display = 'block';

document.getElementById('winnerArea').innerHTML = winnerText;

});

function createOpponentChoiceButton(data){

document.getElementById('opponentState').innerHTML="Opponent made a choice";

let opponentButton = document.createElement('button');

opponentButton.id = 'opponentButton';

opponentButton.style.display='none';

opponentButton.innerText = data.rpsValue;

document.getElementById('player2Choice').appendChild(opponentButton);

}

function sendChoice(rpsValue){

const choiceEvent=player1 ? "p1Choice" : "p2Choice";

socket.emit(choiceEvent, {

rpsValue: rpsValue,

roomUniqueId: roomUniqueId

});

let playerChoiceButton = document.createElement('button');

playerChoiceButton.style.display = 'block';

playerChoiceButton.innerText = rpsValue;

document.getElementById('player1Choice').innerHTML = "";

document.getElementById('player1Choice').appendChild(playerChoiceButton);

}

**Conclusion:**

The rock-paper-scissors multiplayer game using UDP is a simple application that demonstrates how to build a real-time multiplayer game using Node.js and the Socket.io library. The game is built on top of a client-server architecture and uses UDP for communication between the clients and the server. The application is a great starting point for anyone interested in building real-time multiplayer games using web technologies.