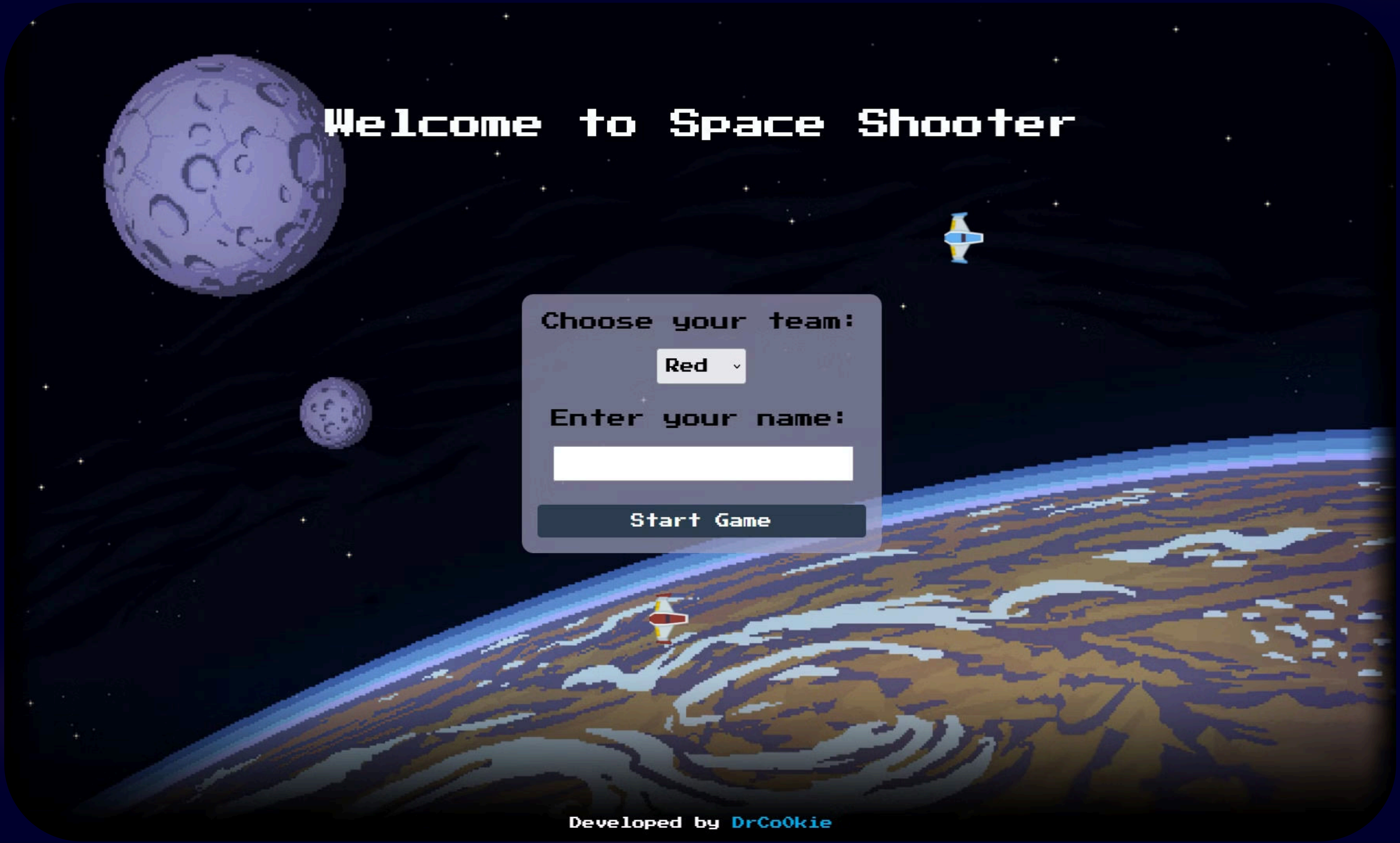

Welcome to Space Shooter

Choose your team: Red

Enter your name:

Developed by DrCo0kie

Space Shooter

Multiplayer

Team-based

Space-themed

Pew-Pew!

Awesome soundtracks!

Architecture

1 Front-end

You know, the part users see and go, "Wow, that's cool!"

Tech Stack: React, CSS, HTML

- Displaying spaceships and black holes
- Handling user input (pew-pew! 🚀)
- Keeping the UI snazzy and responsive

2 Back-end

Tech Stack: Node.js, Socket.io

- Handling game logic (who wins, who loses, who respawns where)
- Managing player data and scores
- Keeping track of all the laser shots and explosions

3 Stuff in between

Tech equivalent of a messenger pigeon, but faster than the speed of light, it keeps everything in sync.

Tech Stack: WebSockets via Socket.io

- Ensuring real-time communication
- Making sure the player's actions are instantly reflected in the game
- Keeping everyone in sync, like a well-oiled machine

Used Technologies

1

React

Pros: Modular, and component-based.

Cons: New Framework to learn, it takes time

2

Node.js

Pros: Standard and known technology.

Cons: Limited for CPU-intensive operations, bottleneck for handling multi-user calculations

3

Socket.IO

Pros: Simplifies real-time communication, easy to use.

Cons: I don't see any

Challenges Encountered

1

Synchronization

Movements, projectiles, respawn

2

State Management

Managing the game's state across components especially unwanted re-rendering

3

Browser Compatibility

Edge vs Firefox requestAnimationFrame()

What I Learned

Problem-Solving and maths

Painfully solving the synchronization issues, remembering the math formulas for player and projectiles movements at 3 am

Real-Time Communication

Understanding WebSockets and implementing real-time features, like managing player interactions.

State Management in React

Managing the game's state across components was a challenge that improved my understanding of React's state management.

Time for a Demo!

Future Improvements

Enhanced Graphics

More animations, explosions and Pew-Pew to provide a more visually stunning experience for players.

Additional Features

Power-ups, rooms, more levels with cool gravity effects. Maybe a single player mode?

Improved Scalability

Optimizing the server-side code by using frameworks like **Phaser** and **Colyseus** to make the game more robust

Questions and Answers

Thank you for your attention!