How the Red Sea Game Works?

1. Game Initialization

- On "New Game", the backend (GameState.java) spawns the player ship, pirates, sea monsters, whirlpools, and islands based on difficulty level.
- Ship is wrapped with ShieldedShip to enable Boost and Shield functionality.

2. Player Movement

- Each key press or button click triggers the move(String direction) method in GameState.
- Movement logic checks:
    - If the move is within bounds.
    - If boost is activated (move 3 steps instead of 1).
    - If the ship hits a whirlpool or island.

3. Boost & Shield Logic

- BoostManager tracks turn count and makes Boost available every few moves.
- When Boost is used, ShieldedShip.activateBoost() is triggered.
- Shields are reduced if a pirate collides with the ship. If shields = 0 → game ends.

4. Enemy Behavior

- PirateManager handles pirate movement based on Strategy (e.g., ChaseStrategy or PatrolStrategy).
- Pirates freeze for 3 turns if the ship enters an island.
- SeaMonsterManager periodically spawns new monsters.

5. Whirlpool Interaction

- If the ship enters a whirlpool, WhirlpoolManager.teleport(ship) moves it to a random grid position.

6. Win & Loss Conditions

- Game ends with a **"Victory"** if the ship lands on a treasure.
- Game ends with **"Game Over"** if all shields are lost due to pirate/monster collisions.

7. Frontend Interaction

- React frontend sends HTTP requests (/move, /newgame, etc.) to backend.Server responds with updated JSON state → parsed and rendered on the UI grid.