# Christopher Columbus Adventure Game –Requirements

## Front End

The front end is built using React and TypeScript. It features a dynamic grid-based user interface with the following functionalities:  
  
- Displays a 20x20 ocean grid where each cell can represent water, island, pirate, ship, monster, treasure, or whirlpool.  
- User can control the Columbus Ship using arrow keys (keyboard events).  
- Game objects like Chase Pirate, Patrol Pirate, Sea Monster, Island, and Whirlpool can be added via on-screen buttons.  
- Users can switch pirate strategies (Chase/Patrol) via a popup overlay on each pirate.  
- Real-time game state is fetched from the backend (`/state`) and mapped visually using images.  
- UI displays freeze ability status and allows its activation via the "Use Freeze" button.  
- Game start screen, game-over overlays (victory/defeat), and animated ship/waves are included for a visually rich experience.  
- The game supports reset and exit controls for seamless interaction.

## Back End

The backend is implemented in Java using a nanoHTTP-based server architecture. Key components and design patterns used include:  
  
- OceanMap(Singleton): Manages a shared 2D grid for game state.  
- Ship (Observable): Represents Columbus' ship; pirate ships observe its location.  
- PirateShip (Observer, Strategy, Decorator): Can have strategies (Chase/Patrol) and be frozen via FreezeShipDecorator.  
- SeaMonsterGroup, SeaMonster (Composite): Grouped monsters move within bounds and participate in collision checks.  
- Whirlpool: Transports ships to random positions when entered.  
- GameServer: Handles API requests such as movement (`/move`), element addition (`/add-element`), strategy switch (`/change-strategy`), freeze usage (`/use-freeze`), and state retrieval (`/state`).  
- Pirate movement is reactive to Columbus' ship updates; collisions are handled by checking map state and game rules.  
- Game ends when the Columbus ship either reaches the treasure or is caught by a pirate/monster.  
- JSON response structure is used to return game state to the front end for rendering.