use_case_descriptions copy

Use Case Descriptions

1. Player Connection Use Cases

UC1: Connect to Game Server

Primary Actor: Player
Preconditions: Server is running, player has game client
Main Success Scenario:

1. Player launches game client
2. Client attempts connection to server (port 30000)
3. Server accepts connection
4. Server assigns player ID (1 or 2)
5. Server initializes game state
6. Client receives confirmation and game starts

UC2: Player Matchmaking

Primary Actor: Player
Preconditions: Player is connected to server
Main Success Scenario:

1. Player enters matchmaking queue
2. Server pairs players based on availability
3. Server creates game session

4. Players receive game start notification

2. Gameplay Use Cases

UC3: Make Game Move

Primary Actor: Active Player
Preconditions: Game in progress, player's turn active
Main Success Scenario:

1. Player selects board position (1-9)
2. Client sends move to server
3. Server validates move
4. Server updates game state
5. Server notifies opponent
6. Opponent's board updates
   Alternative Flow:

- If invalid move:
  1. Server rejects move
  2. Player receives error message
  3. Player must select different position

UC4: Turn Management

Primary Actor: Game Server
Preconditions: Game in progress
Main Success Scenario:

1. Server tracks current player turn
2. Server enables active player's moves
3. Server disables inactive player's moves
4. Server switches turns after valid move

3. Game State Use Cases

UC5: Synchronize Game State

Primary Actor: Game Server
Preconditions: Game in progress
Main Success Scenario:

1. Server maintains game board state
2. Server broadcasts updates to both players
3. Clients update local board display
4. Players see consistent game state

UC6: End Game Session

Primary Actor: Game Server
Preconditions: Game in progress
Main Success Scenario:

1. Server detects win condition or max turns
2. Server calculates final result
3. Server notifies both players
4. Server closes connections
5. Clients display game result

4. Connection Management Use Cases

UC7: Handle Player Disconnection

Primary Actor: Game Server
Preconditions: At least one player is connected
Main Success Scenario:

1. Server detects player disconnection
2. Server notifies remaining player
3. Server updates game state
4. Server terminates game session
5. Remaining client displays disconnect message

UC8: Process Game Logic

Primary Actor: Game Server
Preconditions: Valid move received
Main Success Scenario:

1. Server receives player move
2. Server validates move against game rules
3. Server updates game board state
4. Server checks for win condition
5. Server broadcasts updated state
   Alternative Flow:

- If invalid move:
    1. Server rejects move

   2. Server sends error to player
   3. Turn remains with current player

UC9: Receive Opponent Move

Primary Actor: Inactive Player
Preconditions: Opponent has made move
Main Success Scenario:

1. Client receives move from server
2. Client validates move locally
3. Client updates game board
4. Client enables player controls
   Alternative Flow:

- If connection lost:
   1. Client detects timeout
   2. Client attempts reconnection
   3. Client displays error if failed

Use Case Diagram

```
Error parsing Mermaid diagram!

No diagram type detected matching given configuration
for text: Player1[Player 1]
    Player2[Player 2]
    Server[Game Server]
    UC1[Connect to Server]
    UC2[Make Move]
    UC3[Receive Move]
    UC4[Handle Disconnect]
    UC5[Process Logic]
    UC6[End Game]
    Player1 --> UC1
    Player1 --> UC2
    Player1 --> UC3
    Player2 --> UC1
    Player2 --> UC2
    Player2 --> UC3
    Server --> UC4
    Server --> UC5
    Server --> UC6
    UC1 --> Server
    UC2 --> Server
```