Fully Dressed Use Case 2: Take a Turn

Primary Actor : Game players and/or Computer player

Stakeholders and Interests:

Stakeholders	Interests
Players	Wants to complete their turn efficiently, place/move stones strategically, winning the game by maximizing their score.
Computer Player	Plays fairly with the rest of the players according to the difficulty level, Adheres to rules, and Proceeds with speed so that the player can determine the move of the computer.
Developers	Updating game state in real time, Ensuring rules are followed, Record scores, Maintaining the game.
Distributors	Reaching the target gaming audience, ensuring the availability of the product and its updates.
Customer Support	Handling game failures if any, Assisting with any game queries, Handling complaints, Providing tailored solutions keeping the company’s reputation in mind

Preconditions:

- The game has been set up.
- The system is ready to give a turn to one of the players or computer randomly.

Success Guarantee (Postconditions):

- The player’s turn is completed.
- Dice are rolled, and a valid combination is selected.
- A stone is placed on a valid square (or an existing stone is moved for six-of-a-kind).
- The game board is updated and saved (if exited).
- The player’s score is updated.

- The turn passes to the next player.

Main Success Scenario:

1. The system prompts the current player to roll all the six dice.[Alt1: *Computer Player Turn*]
2. The player clicks "Roll Dice" (or the computer automatically rolls).
3. The system generates six random dice values (1-6) and displays them.
4. The system shows all valid combinations possible with current dice roll.
5. The player selects a valid combination from the list.[Alt2: *Invalid combination selected*]
6. The system highlights squares on the board matching the combination's color/type.
7. The player selects an available square. [Alt3: *Occupied square selected (non-special case)*]
    - If a special combination (five-of-a-kind, 1–6 straight, or six-of-a-kind):
        - ➢ For five-of-a-kind or 1–6 straight: Player selects any free square.[Alt4: *Five of a Kind or 6 Straight*]
        - ➢ For six-of-a-kind: Player selects any square (free or occupied). [Alt5: *Six of a kind*]
8. The system places the player's stone on the square (or moves an existing stone for six-of-a-kind).
9. The system awards points based on square color (pink=3, black=2, white=1)
10. The system updates the player's score based on the square's point value.
11. The system informs players of the updated game state.
12. The system prompts the player to save the game. [Alt6: *Player chooses to save*]
13. The system passes the turn to the next player. [Use Case Ends]

Alternative Flows:

Alt1: *Computer Player Turn*

1. System identifies computer player turn
2. System calculates optimal move based on difficulty level
    - Easy Mode: The system randomly selects a valid square from available options.
    - Hard Mode: The system selects the highest-scoring valid square (prioritizing pink > black > white).

3. System executes move automatically
4. Flow resumes at step 10.

Alt2: *Invalid combination selected*

1. System informs the player the combination is invalid and redisplays valid options.
2. Flow resumes at Step 4.

Alt3: *Occupied square selected (non-special case)*

1. System informs the player the square is occupied and prompts reselection.
2. Flow resumes at Step 7.

Alt4: *Five of a Kind or 6 Straight*

1. The player may place a stone on any free square.
2. Flow resumes at Step 8.

Alt5: *Six of a kind*

1. Players can place the stone on any free square.
2. System moves the existing stone to a new location chosen by the player, if the selected square is occupied.
3. Flow resumes at Step 8.

Alt6: *Player chooses to save*

1. Before making a move, the player requests to save the game.
2. System saves the current game state.
3. Flow resumes at step 1.

Exceptions:

- If an error occurs during auto-save, the system logs the error, informs the player, and continues the game without saving.
- If playing online, a connection issue prevents the move from being registered. The system attempts to reconnect or inform the player of the problem.

Special Requirements:

- Dice results and board updates must render within 2 seconds.
- Move validation must occur within 1 second

- Computer player decisions take 1-3 seconds, to showcase thinking.
- Visual and text-based feedback for moves.
- The game should allow saving and resuming sessions.

Open Issues:

- How does the computer player's hard mode prioritizes moves?
- How are conflicting moves resolved for six-of-a-kind (moving an opponent's stone)?
- Should there be a time limit for moves?