Fully Dressed Use Case 4: Placing the Stone

Primary Actor : Player (Human or Computer)

Stakeholders and Interests:

Stakeholders	Interests
Player	Wants to place stones on high-scoring squares, Wants clear feedback on valid/invalid placements, Wants to get accurate score updates, Wants to understand the consequences of placement.
Other Player(Human and/or Computer)	Wants accurate enforcement of rules, Wants to see updated board state after placement, Want to get accurate score updates.
Developers	Wants the stone placement to be interactive and error free, Wants to maintain consistent game state, Accurately apply placement rules, and update the score and board.
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 player has rolled the dice.
- At least one valid combination is available from the dice roll.

Success Guarantee (Postconditions):

- The stone is placed on the board.
- The game state is updated to reflect the new position of the stone.
- The game board is updated.

- The player is informed of the successful placement.
- The player's score is updated.
- The next player's turn begins.

Main Success Scenario:

1. The system displays the valid combinations to the player.
2. The player selects a valid combination from the dice roll.
3. The system displays the available squares on the board that match the selected combination and prompts the player to choose one.
4. The player selects a square to place the stone.
5. The system verifies that the square is available.[Alt1: *Invalid Square Selection*]
6. The system verifies the square is not occupied and asks the user to place the stone on the selected square.[Alt2: *Square is Occupied*]
7. The player places the stone on the selected square.
8. The system places the stone on the square.[Alt3: *Five of a Kind or 6 Straight*, Alt4: *Six of a kind*]
9. The system updates the game state and informs the player of the successful placement.
10. The system updates the player's score based on the square's color (white=1, black=2, pink=3).
11. The system displays the updated board and score to all players.[Alt5: *Player chooses to save*]
12. The system checks if the game has reached a win condition.
13. If the game is not over, the system proceeds to the next player's turn. [Use Case Ends]

Alternative Flows:

Alt1: *Invalid Square Selection*

1. The system alerts the player.
2. The system prompts reselection.
3. Flow resumes at step 5.

Alt2: *Square is Occupied*

1. The system informs the player that the selected square is occupied.
2. The player selects a new square.
3. Flow resumes at step 6.

Alt3: *Five of a Kind or 6 Straight*

1. The system informs the player they have rolled a special combination allowing placement on any free square.
2. The system highlights every single free square on board.
3. The player places the stone on any free square.
4. The system places the player's stone on the selected square
5. Flow resumes at Step 9.

Alt4: *Six of a kind*

1. The system informs the player they have rolled a special combination allowing placement on any square, including occupied ones.
2. The system highlights all squares on the board.
3. The player selects a square.
4. If the player rolls a six-of-a-kind and selects an occupied square,
    - The system offers the option to move the existing stone to another free square
    - The system prompts for a new location.
5. Flow resumes at Step 9.

Alt5: *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 12.

Exceptions:

- If the system fails to place the stone or update the scores, it informs the player, logs the error details, and halts the current move so the player can retry. The use case ends.
- If the board is updated, but the game crashes, the board cannot be updated, the game saves and exits.

Special Requirements:

- The system must provide clear visual feedback when a stone is placed.
- The system must ensure that the stone placement is accurate and matches the dice combination.
- The stone placement process should not exceed 5 seconds.
- The system must cater to users with color vision deficiency by providing alternative visual cues.
- The game should allow saving and resuming sessions.

Open Issues:

- What happens if no valid squares remain for a displaced stone?
- How should we handle ties when multiple players have the same score at game end?
- How should displaced stones be visualized when using the six of a kind special combination?