New Fully Dressed Use Case Description:
LOAD GAME

Primary Actor: User

Stakeholders and Interests:

- Users: To be able to load a saved game. Users want to be able to pick up where they left a previous game along. All game assets should be identical to the time of saving, and play resumes at the current player's turn.

Preconditions:

- There should be a menu interface which allows the user to load a game.
- There needs to be a sub-system which can take load the previously saved assets and use them to recreate the player's last game.

Postconditions:

- A successfully loaded game should provide four player objects, 81 game tile objects, and an indication for the next player.
- If there are no saved games available, the player will be provided with a method to move back into the main menu interface.

Main Success Scenario:

1. User selects a load game option from the main menu interface.
2. The system checks if there are any previously saved games.[Alt 1]
3. The system provides the user with a list of previously saved games.
4. The user selects a saved game for loading.
5. The system asks the user to confirm to load the game or to go back to the main menu.
6. The user confirms they want to load the game.
7. The system unpacks four players, 81 game tiles, and an indication for next player from the loaded game data, and uses it to rebuild a game board for play. [Alt 2]
8. The system displays the game board with the correct positions of the pawns, fences and turn order from the saved game.
9. The system hands control to the user, or an appropriate AI player for play to resume on next turn. [Use case ends]

Alternate Flows:

- Alt 1: The system finds no file
    - The system displays file not found
- Alt 2: The file is corrupt and the data cannot be read.
    - The system displays a corrupted data message, and presents the user with the main menu.

Open Issues:

- Can a user change any initially configured settings (ie: color, players) for a resumed game?