Use Case #2 Play Scrabble

Goal in Context:

A player must be able not just to play the game, but also enjoy it.
Different approaches:

- Fascinating gameplay. It may include awesome and unique game mechanics, that make the game special compared to others.
- Amazing visual style. A pleasant game interface and smooth animations etc.

Level:

Summary

Preconditions:

- The player is already logged in.
- The player has already chosen a game mode (Versus AI only, Versus real players or mixed).
- The player (and his match-mates if network game was chosen) already knows the game rules or completed the tutorial mode.

Success End Condition:

The game shows match stats and updates the players' profiles in the database. (The JavaFX Scene is changed from "game interface" to "after-match lobby". This scene includes match stats, chat and few actions like "quit", "restart" etc.).

Failed End Condition:

The host quits the game or closes the application.

Primary Actor:

Player(s) / User(s)

Trigger:

A match has been started ("Start" button in match lobby was clicked).

Main Success Scenario:

1. Every player (humans or AI) has successfully joined the game.
2. Every player gets his own start pack of letter tiles. Each pack should contain at least 1 or 2 vowels. Points that a specific letter tile has, should be balanced (i. e. rare letters have more points than the ones that are used more frequently in words).
3. The game establishes the game turn sequence.
4. Game Loop, while match is not over:
   4.1. Depending on:
      a) If it IS 1st round: The player on the top of the game turn sequence must make a word (horizontal or vertical) that passes through the center cell of the game grid.
      b) If it IS NOT 1st round: The player on the top of the game turn sequence must make a word (horizontal or vertical) that passes through one of the tile letters.
   4.2. The game analyzes the word and verifies if it is valid (the word gets highlighted with green if it is valid and with red otherwise). Total points of the word, considering the special grid cells (point multipliers), are also displayed.
   4.3. The player chooses the best option, and then presses the "OK" button, which will end the player's turn and attribute him the word points (players' scores are displayed near to their nicknames).
   4.4. The leaderboard is restructured taking into account the player's new score.

4.5. The game checks if the match is over. Multiple cases:
   a) The letter tile bag is empty, and the player is out of letter tiles.
   b) Taking into account remaining letter tiles, no more valid letter can be placed.
   c) Etc.
4.6. Depending on:
   a) If the match IS over: Quit the game loop.
   b) If the match IS NOT over: The game turn sequence is changed, putting the current player on the end and the next player on the top.

5. The game shows match stats and updates the players' profiles in the database. (The JavaFX Scene is changed from "game interface" to "after-match lobby". This scene includes match stats, chat and few actions like "quit", "restart" etc.).

Special Requirements:

1. During each round:
   1.1. Every player should be able to view a word's definition by pressing on the word.
   1.2. Only the current player should be able to interact with letter tiles and the grid.

Extensions (Alternative Flows):

4) Somewhere in the game loop:
   4a) The host quits the game:
      1) Every player (human or AI) is getting kicked from the game.
      2) The game finishes.
   4b) Someone else quits the game:
      1) The game continues without that player.
   4c) Every player quits the game, except one player:
      1) The remaining player wins the game.