TIC TAC TOE

Use Case: Player joins matchmaking queue

Iteration: 1

Primary Actor: Player

Goal in Context: The player wants to quickly find an opponent by entering the Tic Tac Toe matchmaking system.

Preconditions:

- The player is logged in.
- The player has a valid account and an assigned MMR/rank for Tic Tac Toe.

Trigger: The player selects the “Join Queue” option from the Tic Tac Toe game interface.

Scenario:

1. The player clicks the “Join Queue” button.
2. The system verifies the player’s current Tic Tac Toe rank and MMR.
3. The system randomly assigns the player to one of the two queue pairs for their rank.
4. The player is added to the selected Tic Tac Toe matchmaking queue.

Postconditions: The player is now waiting in a Tic Tac Toe matchmaking queue for pairing.

Exceptions:

- Player is already in an active queue.
- Network/server issues prevent the queue join.

Priority: High

When Available: Always

Frequency of Use: High (each game session)

Channel to Actor: Tic Tac Toe client interface

Secondary Actors: Tic Tac Toe matchmaking service

Channel to Secondary Actors: N/A

Open Issues:

- Determining how to handle simultaneous queue join requests.

Use Case: MMR changes after a game completes

Iteration: 1

Primary Actor: Game/Player

Goal in Context: Update the player's MMR based on the outcome of a completed Tic Tac Toe game.

Preconditions:

- The Tic Tac Toe game has just finished.
- The game outcome (win, loss, or tie) is recorded.

Trigger: The Tic Tac Toe game has ended.

Scenario:

The game computes the result of the Tic Tac Toe game.

The system invokes the corresponding method on the player's Tic Tac Toe stats object.

The player's MMR is recalculated.

Updated MMR is stored and reflected in the player's profile and leaderboard.

Postconditions: The player's Tic Tac Toe MMR accurately reflects their recent game performance.

Exceptions:

- Errors in MMR calculation.

Priority: High

When Available: Always

Frequency of Use: Every Tic Tac Toe game completion.

Channel to Actor: Internal game processing.

Secondary Actors: Stats and leaderboard

Channel to Secondary Actors: N/A

Open Issues:

- Refine the formula used to update player ratings in Tic Tac Toe so that changes are smooth and fairly reflect performance differences.

Use Case: Display Leaderboard

Iteration: 1

Primary Actor: Game/Player

Goal in Context: The player wants to view a ranked list of Tic Tac Toe players based on metrics such as MMR or wins.

Preconditions:

- The Tic Tac Toe leaderboard data is available.
- The game is responsive.

Trigger: The player selects the “View Leaderboard” option from the Tic Tac Toe game menu.

Scenario:

1. The player clicks on the “Leaderboard” tab.
2. The system retrieves Tic Tac Toe leaderboard data.
3. The sorted leaderboard is displayed on the player's screen.

Postconditions: The player is presented with an updated and ranked leaderboard for Tic Tac Toe.

Exceptions:

- Missing leaderboard stats.

Priority: High

When Available: Always

Frequency of Use: Moderate

Channel to Actor: Tic Tac Toe game client interface

Secondary Actors: Leaderboard system

Channel to Secondary Actors: N/A

Open Issues:

- How will we ensure the stats are calculated effeciently and that the display of the leaderboard is consistent even with large data sets?

Use Case: Player Leaves Matchmaking Queue

Iteration: 1

Primary Actor: Player

Goal in Context: The player wants to exit the matchmaking queue before being matched with an opponent.

Preconditions:

- The player is currently in the matchmaking queue.
- The matchmaking system is responsive.

Trigger: The player selects the "Leave Queue" option from the matchmaking screen.
Scenario:

1. The player clicks on the “Leave Queue” button.
2. The system removes the player from the matchmaking queue.
3. The system confirms the player has successfully left the queue.

Postconditions: The player is no longer in the matchmaking queue and can take other actions.

Exceptions:

- The player is matched with an opponent at the same time they attempt to leave, making the action invalid.
- Server issues prevent immediate removal from the queue.

Priority: Medium

When Available: Always, as long as the player is in the queue.

Frequency of Use: Occasional

Channel to Actor: Tic Tac Toe game client interface

Secondary Actors: Matchmaking system

Channel to Secondary Actors: N/A

Open Issues:

- Should there be a cooldown or penalty for frequently leaving the queue?
- How will we handle edge cases where a match is found at the same time the player tries to leave?

Use Case: Updating Player Stats

Iteration: 1

Primary Actor: Leaderboard System

Goal in Context: Update player statistics (wins, losses, ties, and MMR) into the leaderboard system after each game concludes.

Preconditions:

- The player can have an existing profile in the system
- The player can be a new player
- The match must have concluded with a win, loss, or tie
- CSV file which stores leaderboard data must be accessible

Trigger: The match ends, and the client updates the player statistics into the CSV file for the leaderboard to read

Scenario:

1. The player completes a game
2. The tic Tac Toe client records the match result
3. The result of a game is sent to the LeaderboardManager
4. The leaderboardManger reads the csv file and calls update_player

Postconditions: After the result of a game the player's stats are updated accurately into the leaderboard

Exceptions:

- Corrupt or missing csv file
- Duplicate player entries in the file
- Inconsistent match results (One player reports a win and the other reports a tie or loss)

Priority: High

When Available: Always

Frequency of Use: High (each game session)

Channel to Actor: Tic Tac Toe client interface

Secondary Actors: Tic Tac Toe matchmaking service

Channel to Secondary Actors: N/A

Open Issues:

- How will a result mismatch (One player reports win and other reports tie or loss) be resolved
- Will the leaderboard be updated after every game concludes or after a set interval?

Use Case: **Handling MMR tie on leaderboard**

Iteration: 1

Primary Actor: Leaderboard System

Goal in Context: A fair method for ranking players when multiple players have the exact same MMR on the leaderboard

Preconditions:

- At least two players have the same MMR in the database
- The system must relay on a predetermined method for ranking tied players
- A player's stats are properly recorded

Trigger: A match ends which updates the leaderboard, causing two or more players to have the same MMR

Scenario:

1. The player completes a game
2. The System detects that two or more players have the same MMR
3. The leaderboard sorting mechanism applies the tie breaker rule
4. The ranking order is determined by: Higher number of wins is ranked higher, If wins are also tied, fewer losses ranks higher, If still tied, most recent match result is considered otherwise system uses player ID as a last resort

Postconditions: Players with the same MMR are ranked fairly based on secondary criteria and updates the leaderboard accordingly.

Exceptions:

- Leaderboard sorting failure
- If two players have exactly identical stats the system sorts them by player ID

Priority: Medium

When Available: Always

Frequency of Use: Low (happens when multiple players have the same MMR)

Channel to Actor: Tic Tac Toe client interface

Secondary Actors: N/A

Channel to Secondary Actors: N/A

Open Issues:

- Should win streaks be a factor into ranking