Use Case 1: View Leaderboard

- Iteration: 1
- Primary actor: User
- Goal in context: User wants to view the leaderboard of a game
- Preconditions:
    - The user is logged in to the application
    - A game has been selected
- Trigger: user opens the leaderboard section and selects a game
- Scenario:
    1. User selects a game
    2. System gets the leaderboard data from the game that user has selected
    3. System sorts the players by points/wins in descending order.
    4. System displays the sorted list of players • Postconditions: ○ The leaderboard is displayed with the ranking of players.
- Exceptions:
    - If there is no data to display
    - If user is not logged in
    - If app is not running
- Priority:
    - High
- When available: Iteration 2
- Frequency of Use:
    - High (user checks ranking often)
- Channel to actor: ○ Application GUI
- Secondary Actors:
    - None
- Channels to Secondary Actors:
    - N/A
- Open Issues:
    - How often should leaderboard update?
    - Should leaderboard constantly update while app is running?

Use Case 2: Interacting with the Leaderboard System Basic (ILSB)

- Iteration: 1, started March 4
- Primary Actor: Player, Alex

- Goal in context: To interact with the leaderboard system, including viewing rankings, updating score.
- Preconditions:
    - The leaderboard system is operational and accessible. ○ Players have valid accounts with scores stored in the system.
    - The system can process updates and display ranking correctly.
- Trigger: ○ Alex completes a game and attempts to interact with the leaderboard,
- Scenario:
    1. Alex finishes a game session and earns a new score.
    2. The system automatically updates Alex's score in the database.
    3. Alex navigates to the leaderboard screen.
    4. The system retrieves and displays the latest rankings.
    5. Alex sees their updated ranking, verifies their score is correct, and is happy that he is higher than his rival now.
    6. Alex logs out
- Post conditions: ○ The leaderboard accurately reflects the updated score.
- Exceptions:
    ○ Delay in output/new updates to leaderboard ranking/score ○ Network issue
- Priority:
    ○ Medium, as leaderboard enhance player engagement but do not affect core gameplay.
- When available: ○ Iteration 4
- Frequency of Use: ○ Regular, depending on player activity.
- Channel to actor: ○ App Interface (in-game leaderboard menu)
- Secondary actors:
    ○ Game server (processes score updates and retrieves rankings)
- Channel to secondary actors:
    - Game database (scores and retrieves player scores)
    - LeaderBoardUpdater
- Open issues:
    - How long until the system will update the scores of the player?
    - How would the ranking system handle players with the same score/ranking?

Use Case 3: Automatically Update Leaderboard when opening app

- Iteration: 1
- Primary actor: System
- Goal in Context: System automatically updates the leaderboard with the new results when on the leaderboard page
- Preconditions:
  - ○ User is on the leaderboard page
- Trigger: The leaderboard page is opened by the player
- Scenario:
  1. If the leaderboard is already up to date, no changes are made.
  2. If The system detects that the leaderboard needs an update, seeing the difference between the database kept and the displayed rankings
  3. System processes and automatically updates and sorts the new ranking
  4. The Interface then receives the new list and displays that to the user
- Postconditions: ○ The leaderboard displays the most recent rankings at all times.
- Exceptions:
  - ○ Network Error/ no Internet Connection ○ Server timeout ○ Corrupt data
- Priority:
  - ○ High, will allow for the system to show the most recent ranking of players
- When available: ○ Iteration 3
- Frequency of use: ○ High
- Channel to actor: ○ NA
- Secondary actors:
  - ○ NA
- Channels to secondary actors: ○ NA
- Open issues:
  - ○ Should the System be automatically updating (everytime a game finish), is this an efficient system?
    - How many times should the system update the leaderboard automatically?

Use Case 4: Update leaderboard when update button is pressed in leaderboard page

- Iteration: 1, started March 5
- Primary actor: Player, Max
- Goal in Context: User can manually update the leaderboard page to show new results.
- Preconditions:

- The leaderboard page is already open by the player.

- Trigger: User presses the update button in leaderboard page
- Scenario:
  1. Max notices that the leaderboard Is not up to date, so he navigates to and presses the Update button.
  2. This action sends a request to the server to update the leader board data and a loading graphic is shown to the user to show the request is in progress
  3. The System returns the updated leaderboard data and replaces the old leaderboard
  4. The system ends the loading graphic and displays the recent data and a notification is given to the player (e.g., “The leaderboard has been updated”)
- Postconditions: ○ The leaderboard reflects the latest rankings after the update request
- Exceptions:
  ○ Network Error/ no Internet Connection ○
  Server timeout ○ Corrupt data
- Priority:
  ○ Medium / high
- When available: ○ Iteration 3
- Frequency of use: ○ Occasionally, whenever the player notices a discrepancy
- Channel to actor: ○ Leaderboard interface, Button
- Secondary actors:
  ○ Game Server
- Channels to secondary actors:
  ○ Game Database
- Open issues:
  ○ Should there be a cooldown to prevent excessive requests to the server?

Use Case 5: Search Leaderboard for player names

- Iteration: 1, started March 5
- Primary actor: Player, Jordan
- Goal in Context: User can search for a specific player's ranking and wins on the leaderboard.
- Preconditions:
  ○ The system allows text-based search functionality within the leader board interface.
- Trigger: Jordan wants to find another player’s ranking and enters a name in the search bar.
- Scenario:
  1. Jordan enters a player's username (e.g., "LakerLebron23") into the search bar.

2. The system validates the input (e.g., checks for invalid characters).
3. The system searches the database for matching usernames/players.
4. If the exact username/player is found, search bar would show that result and player, Jordan can pick that result, the system then displaying the player's (e.g., "LakerLeborn23”) ranking, wins, and relevant stats.
5. If multiple partial matches are found (e.g., "LakeBorn23”), the system presents a list of possible matches, which the player can pick from to display.
6. If no results are found, the system informs Jordan that the player does not exist in the leaderboard and should retry.

- Postconditions:
    - Jordan successfully finds and views the target player's leaderboard data.
    - If the player does not exist, Jordan receives a clear notification (e.g., "Player Searched does not exist, retry with another username.”)
- Exceptions:
    - ○ Invalid Input: If Jordan enters special characters or an empty string, the system rejects the input and prompts for a valid search term. ○ Database Delay: If the search takes too long, the system provides a loading indicator and a retry option.
- Priority:
    - ○ Medium
- When available: ○ Iteration 3
- Frequency of use: ○ High/ Daily, players will want to see their results compared with others
- Channel to actor:
    - ○ Leaderboard Interface, Search bar
- Secondary actors:
    - ○ Game server
- Channels to secondary actors:
    - ○ Game database
- Open issues:
    - Is there a quick and efficient way of sorting/ searching through all the database of players to find the Searched player? ○ Are there any rules to the search bar, specific keys, case-sensitive, and so forth?
    - Is there any other form of searching players, such as ID number?

Use Case 6: Filter Leatherboard

- Iteration: 1, started March 5
- Primary actor: Player, Lebron
- Goal in Context: User wants to filter the leaderboard using a specific criterion
- Preconditions:
    - The system supports filtering options (e.g., by region, friends, game mode).
- Trigger: Lebron applies a filter onto the leaderboard (e.g., Tic-Tac-toe and Regional)
- Scenario:
    1. Lebron selects the filter option on the leaderboard interface.
    2. The system presents available filters (e.g., global vs. regional rankings, friendsonly, Win-Ratio, Game, Etc.).
    3. Lebron chooses that he wants to see the highest regional players playing the game TIC-TAC-TOE
        - Selects Regional
        - Selects tic-tac-toe as game
        - Leaves other selections as unselected
    4. The system applies the filter, retrieves, and displays the updated leaderboard based on the selected filter.
    5. If no results match the selected filter, the system informs Lebron and suggests removing or adjusting filters.
    6. Lebron interacts with the filtered leaderboard, checking rankings and stats. Lebron can select any player and see in greater detail any other stats of the player.
    7. Lebron changes or removes the filter to return to the default view of leaderboard
- Postconditions:
    - The leaderboard updates according to the applied filter. ○ System notifies user of no result with given filter ○ After Removing Filters, the leaderboard will go to default
- Exceptions:
    - Network Failure ○ No Results Found ○ Slow Database Response
- Priority:
    - Medium
- When available: ○ Iteration 3
- Frequency of use:
    - Medium
- Channel to actor: ○ Leaderboard Interface, Filter Options

- Secondary actors:
    - Game Server
- Channels to secondary actors:
    - Game Database
- Open issues: ○ How should the system handle large datasets?