Requirements Document: Poker Tool Suite

Project Title: Modern Nightlife Poker Engine & Simulator

Version: 1.1.0

Author: Jenna James

Date: January 11, 2026

Status: Finalized / For Review

1. Project Overview

The Poker Tool Suite is a comprehensive software package designed to provide high-performance hand evaluation, statistical simulation, and strategic education. It is tailored for developers and players who require mathematically accurate poker logic paired with a modern, responsive user interface.

2. Functional Requirements

2.1 Core Poker Engine (TexasHoldemEngine)

- **Combination Generation:** The system must generate all 21 possible 5-card combinations from any 7-card pool (2 hole cards + 5 community cards).
- **Hand Ranking:** The system must accurately identify the 10 standard poker hand ranks, from High Card to Royal Flush.
- **Tie-Breaking:** The system must utilize a frequency-based `valueOrder` array to resolve ties using kickers when primary hand strengths are identical.
- **Winner Determination:** The system must identify and return a list of winners, correctly handling split-pot scenarios where multiple players share the best hand.

2.2 Statistical Simulator

- **Monte Carlo Simulation:** The system must run a user-defined number of random trials (up to 10,000+) to calculate rank probabilities.
- **Deck Management:** The system must generate a standard 52-card deck and utilize the Fisher-Yates algorithm for unbiased shuffling.
- **Non-Blocking UI:** The system must use `requestAnimationFrame` to chunk large computations, ensuring the browser UI remains responsive during simulations.

2.3 Educational Trainer (HandAnalyzer)

- **Strategic Mapping:** The system must map specific hand strengths to human-readable advice strings (e.g., "be cautious" for Straights or "unbeatable monster" for Royal Flushes).
- **Visual Data Normalization:** The system must identify and return the specific cards used in a winning hand to facilitate UI highlighting.

3. Use Cases

Use Case 1: Evaluate a Texas Hold'em Winner

- **Actor:** User (Player/Dealer)
- **Description:** Determining the winner among multiple players after the "River" (all 5 community cards) is dealt.
- **Preconditions:** 5 community cards and at least 1 player with 2 hole cards are provided to the `TexasHoldemEngine`.
- **Flow:**
    1. User inputs card strings for the board and players.
    2. The engine generates 21 combinations per player.
    3. The engine evaluates each combo via the `PokerHand` class.
    4. The engine compares the "Best 5" for all players and returns the winner(s).

Use Case 2: Run Probability Analysis

- **Actor:** User (Analyst/Student)
- **Description:** Calculating the likelihood of being dealt a specific hand rank.
- **Preconditions:** The `StatisticalSimulator` is initialized.
- **Flow:**
    1. User selects the number of trials.
    2. The simulator shuffles a virtual deck for each trial.
    3. The top 5 cards are evaluated and the results are tallied.
    4. The system renders an HTML table showing the count and percentage for each rank.

Use Case 3: Receive Strategic Advice

- **Actor:** User (Student/Novice)
- **Description:** Obtaining an educational breakdown of a specific 5-card hand.
- **Preconditions:** A 5-card string is passed to the `HandAnalyzer`.
- **Flow:**
    1. The analyzer identifies the rank.
    2. The analyzer retrieves corresponding advice from its logic mapping.

3. The system identifies which cards in the hand are active (e.g., the two cards forming a Pair).
4. The system displays the advice and highlights relevant cards.

4. Technical & Reliability Requirements

- **Edge Case Handling:** The system must handle "Wheel" straights (A-2-3-4-5), 3-character "10" card inputs, and unordered card strings.
- **Performance:** Simulations must execute without freezing the browser's main thread.
- **Cross-Browser Compatibility:** Logic must be bundled via Browserify into a `bundle.js` file for universal browser support.
- **Test Coverage:** All core logic must pass unit tests in the `/test` directory, including edge-case verification.