Functional Requirements Document (FRD)

Project Title: Tic Tac Toe Game
Date: August 21, 2025
Prepared By: Abhisek Swain

1. Purpose

The purpose of this document is to define the functional and non-functional requirements for the Tic Tac Toe Game application.
This system provides an interactive platform for two players (human vs human) or a player versus computer to play on a 3x3 grid.
The application includes a clear user interface, real-time move validation, and automatic detection of win and draw states.

This FRD serves as the foundation for development, testing, and deployment of the Tic Tac Toe Game, ensuring all stakeholders have a shared understanding of its functionalities and objectives.

2. Project Overview

2.1 Objective

To develop a lightweight web-based application that provides an enjoyable Tic Tac Toe experience with options to play against another person or a basic rule-based AI, while offering simple score tracking and intuitive controls.

2.2 Scope

- Responsive web UI with a 3x3 game board.
- Modes: Player vs Player (local), Player vs Computer (basic AI).
- Win/draw detection and result announcement.
- Scoreboard with wins/losses/draws and option to reset.
- Player name customization.
- Lightweight sound and visual feedback (optional).

2.3 Out of Scope

- Online multiplayer over the internet.
- Ranked matchmaking or global leaderboards.
- Advanced AI (minimax with pruning, ML-based opponents).
- Mobile native apps (Android/iOS).

2.4 Assumptions

- Game uses a standard 3x3 grid with alternating turns.
- Game runs in modern browsers with JavaScript enabled.
- No dependency on external enterprise systems.
- No user account is required unless leaderboard is enabled.

3. Functional Modules

3.1 Menu Structure

Menu Item	Submenus/Features
Home	Start Game, Choose Mode (Single Player/Multiplayer), Instructions
Dashboard	Recent games, Quick start, Last results
Game Board	3x3 grid, move validation, turn indicator, move history
Scoreboard	Wins, Losses, Draws; Reset Scores
Settings	Player Names, AI Difficulty, Sound On/Off, Theme (Light/Dark)
Help	Rules, How to Play, About

3.2 Design Patterns

- Collapsible sidebar with icons and labels (optional).
- Highlight active menu items and current player's turn.
- Role-based visibility (e.g., only Admin sees Settings for leaderboards).
- Accessible UI with keyboard support and ARIA labels.
- Mobile-friendly tap targets for board cells.

3.3 Home Page Features

- **Start Game:** Begin a new session with selected mode and player names.
- **Choose Mode:** Single Player (vs Computer) or Two Players (local).

- **Instructions:** Rules of Tic Tac Toe and how turns work.
- **Recent Results:** Last 5 game outcomes (optional).

3.4 Game Detail Page

Section	Description
Game Information	Mode, Player names, AI level (if applicable)
Game Rules	Standard 3x3 grid, alternate turns, X goes first by default
Current Board State	Visual grid with X/O placements
Turn Indicator	Highlights whose turn it is
Move History	Sequence of moves (cell positions)
Result	Win (with winning line highlighted) or Draw
Scoreboard	Cumulative wins/losses/draws across rounds
Timer (Optional)	Per-move or per-game timer settings
Accessibility	Keyboard navigation and screen-reader text for cells

1. Non-Functional Requirements

Requirement	Details
Performance	UI should respond within 200 ms for cell clicks; game logic resolves instantly.
Scalability	Supports multiple concurrent users on a shared hosting environment without degradation.
Security	No PII; if profiles/leaderboards enabled, use secure storage and role-based access.
Availability	99.5% uptime for hosted web version.
Logging and Monitoring	Client-side error logging and basic analytics (optional).
Data Persistence	Local storage for scores and preferences; server DB only if online profiles are enabled.
Browser Compatibility	Latest versions of Chrome and Edge; graceful fallback for others.

2. User Roles and Permissions

Role	Permissions

Administrator	Full access to settings, theme, AI levels, and (if enabled) leaderboard management.
Player	Play games, set names, select mode, view scores.
Guest/Read-Only	View-only access to game board demo and help.
Observer	Spectate ongoing local games (if screen-share or kiosk mode).

3. Dependencies

Dependency	Owner
Web hosting / static site (e.g., Netlify/Vercel)	Project Team
CI/CD pipeline setup	Project Team / DevOps
Authentication provider (optional; e.g., Okta/OAuth)	Project Team / IAM
Analytics (optional)	Project Team
GitHub repository access	Project Team

4. Pre-Requisites

- Modern web browser (Chrome/Edge).
- Access to hosting platform account.
- Git access for source code.
- Basic UI assets (icons, sounds) if used.

5. Reporting Requirements

- Simple score summary and session statistics within the app.
- Optional: export results as CSV or integrate with a lightweight analytics dashboard.

6. Governance & Risk Management

- Weekly check-ins for scope and UI review.
- Track risks (e.g., scope creep, browser quirks, accessibility gaps) and maintain mitigation plans.
- Adopt agile ceremonies for iteration demos and feedback.

7. Okta Integration Analysis

Note: This section applies only if authentication and user profiles/leaderboards are enabled.

Option - 1

- **Front End (React/Vue):** Configure Client ID and Issuer URL.
- Add dependencies for Okta SDK; open Okta login via Issuer URL.
- Send login request with credentials and Client ID; store access token and user details client-side.
- Control page content based on role; include access token in request headers.

Back End (if used):

- Secure REST APIs; validate access token.
- Provide user info endpoint if needed for profile/leaderboard.

Option - 2

- **Front End:** Minimal responsibility—UI relies on session-based auth handled by the backend.
- **Back End:** Open Okta login, set redirect URLs, authenticate and manage session (JSESSIONID).
- Protect routes using server-side security; provide user info and token endpoints if needed.
- On logout, clear the session and remove cookies.

Okta Setup

- Create a tenant and obtain Issuer URI and Client ID.
- Set roles and user info inside claims as required for role-based access.