TicTacToe Unity Game - Technical Documentation

Developer: Samuel Adeosun | **Date:** February 8, 2026 **Engine:** Unity 6 (6000.3.6f1) | **Language:** C#

1. Mini-Game Choice: TicTacToe

TicTacToe was selected for its **simple core mechanics** allowing focus on sophisticated features, **clear win conditions** ideal for achievement systems, **universal appeal** requiring no tutorial, and **scalable complexity** providing opportunities for creative enhancements (AI difficulty, achievements, animations).

2. Creative Feature: Achievement Milestone System

Comprehensive achievement system with five unique achievements that increase player engagement, provide progression goals, and add depth to the game:

Achievement	Unlock Condition
First Victory	Win any game mode
Hat Trick	Win 3 games consecutively
AI Conqueror	Beat Hard AI difficulty
Perfect Victory	Win without opponent getting 2-in-a-row
Friendly Rivalry	Complete 10 games in PvP mode

3. Technical Architecture

Design Patterns

Singleton - Global GameManager access
ScriptableObject - Designer-friendly achievement data
MVP - GamePresenter/GameView separation
Observer - Event-driven achievement notifications

Core Systems

AchievementManager - Progress tracking and unlocking
GameBoard - 2D array board state
WinChecker - Win detection algorithms
AIGameController - AI with difficulty levels

Persistence: PlayerPrefs stores achievement state and statistics across sessions.

4. Bonus Feature: Achievement Notification & Showcase

Notification System

- Smooth slide-in/out animations (Coroutines + Lerp)
- Configurable 3-second display duration
- Audio feedback on unlock
- Queue-based sequential display

Showcase Panel

- Full achievement gallery with progress tracking
- Dynamic UI from ScriptableObject data

5. Generative AI Tool Usage

Tools Used: GitHub Copilot and Dearify.ai

GitHub Copilot

- Real-time code completion and suggestions
- Boilerplate code generation for achievement tracking

Dearify.ai

- **UI/Screen Generation:** Visualized achievement notification and showcase layouts

- Helped design notification panel structure
- Assisted with showcase gallery mockups
- Provided UI/UX design guidance

Impact: 30-40% faster development, reduced debugging time, improved code quality and UI design clarity.

6. Challenges & Solutions

1. Achievement Persistence: Implemented PlayerPrefs storage with unique key prefixes for reliable save/load functionality

2. Perfect Victory Detection: Real-time board monitoring to track when opponent achieves 2-in-a-row states

3. Notification Queue Management: Queue-based system prevents UI overlap when multiple achievements unlock simultaneously

7. Future Improvements

Achievement Expansion: 10+ new achievements, achievement tiers (Bronze/Silver/Gold), secret achievements, statistics dashboard

Polish & Animation: Cell placement particles, victory celebrations, character themes, background music, dynamic sound effects

New Features: Online multiplayer, replay system, daily challenges, customizable skins, tournament brackets, leaderboards

Technical: Unit testing, JSON/Binary save system, analytics integration, localization support

Conclusion

This TicTacToe implementation successfully combines classic gameplay with modern game design through a comprehensive achievement system that adds replay value while maintaining simplicity. The architecture demonstrates solid engineering principles with MVP pattern, ScriptableObjects, and event-driven design. AI tools (Copilot for coding, Dearify for UI visualization) accelerated development by 30-40% while maintaining high code quality.

Development Time: 1 Week | **Lines of Code:** ~2000+ C#