PRD - Snake Evolution Game v2.0

Product Requirements Document - Versione Riveduta e Migliorata

Documento redatto da: Senior Software Architect
Data: Novembre 2025
Versione: 2.0 (Aggiornato post-revisione critica)
Status: Approved for Development

Executive Summary

Il presente documento descrive i requisiti per lo sviluppo di **Snake Evolution v2.0**, un gioco arcade single-player ispirato al classico Snake con l'aggiunta di un **sistema di evoluzione progressiva** come meccanica innovativa. Questa versione incorpora migliorie significative rispetto a v1.0, affrontando edge cases critici, scalabilità, e robustezza del sistema.

1. Descrizione del Prodotto

1.1 Che cos'è Snake Evolution?

Snake Evolution è un gioco arcade single-player che mantiene la semplicità del classico Snake arricchendolo con un **sistema di trasformazione evolutiva del serpente** che si attiva al raggiungimento di milestone di lunghezza. Il gioco introduce nuove dimensioni di gameplay tramite potenziamenti visivi, meccaniche progressive, e un sistema di progression tangibile.

Differenziatori Chiave:

- Sistema evoluzione dinamica in 5 stadi progressivi
- Velocità incrementale che aumenta la difficoltà nel tempo
- Effetti visivi e sonori per feedback immediato
- Salvataggio persistente dei progressi
- Leaderboard locale con top 10 score

1.2 Pubblico di Destinazione

- Gamer casual 7-60 anni
- Appassionati di giochi retrò con innovazioni
- Giocatori mobile che cercano sessioni brevi (5-10 min)
- Studenti/professionisti durante pause lavorative

1.3 Positioning Strategico

Snake Evolution si posiziona come **"Classic Snake Reimagined"** - una reinterpretazione moderna di un gioco iconico, mantenendo semplicità e accessibility mentre introduce meccaniche progressive che aumentano engagement e replayability.

2. Obiettivi Strategici

2.1 Obiettivi di Business

1. Acquisire 10,000+ download nel primo mese di lancio
2. Mantenere retention Day-7 ≥ 30%
3. Raggiungere rating ≥ 4.0 stelle su 100+ recensioni
4. Posizionarsi come alternativa innovativa ai classici Snake

2.2 Obiettivi Tecnici

1. Garantire stabilità con **99.9% uptime** durante gameplay
2. Mantenere **60 FPS costante** su device target
3. Load time < 2 secondi da tutti gli stati
4. **Zero data loss** su localStorage corruption (recovery mechanism)

2.3 Obiettivi UX/Engagement

1. Tutorial < 30 secondi (auto-learning preferito)
2. Sessione media 5-10 minuti
3. Feedback visivo immediato (< 50ms) su tutte le azioni
4. Percezione di progression chiara (evoluzione visibile)

2.4 KPI SMART - Revisione v2.0

KPI	Target	Metodo Misurazione	Criticità
Downloads	10k+ in 30 giorni	Store analytics	Alta
Retention D1	50%	User tracking	Alta
Retention D7	30%	User tracking	Media
Rating	4.0+ stelle	App store rating	Media
Avg Session	5-10 min	Gameplay time tracking	Media
Crash Rate	< 0.5%	Telemetry tracking	Alta
Avg FPS	55+ (target 60)	Performance profiler	Alta
Data Integrity	100%	Checksum validation	Alta

3. Vincoli, Ipotesi e Risk Mitigation

3.1 Vincoli Tecnici

Hardware/Browser:

- Compatibilità: Chrome 60+, Firefox 55+, Safari 11+, Edge 79+
- Performance minima: 30 FPS su device mobile 2015+
- Memory: Max 50MB runtime
- Storage: 500KB localStorage disponibile

Implementazione:

- Vanilla JavaScript (zero external runtime dependencies)
- Canvas 2D API (non WebGL per compatibilità)
- Web Audio API per audio

3.2 Vincoli Aziendali

- Budget: $0 (indie development)
- Team: 1-2 sviluppatori
- Timeline: 8 settimane
- Post-launch support: 1 developer FT

3.3 Ipotesi Critiche

- **Ipotesi 1**: Utenti gradiscono progressione visibile (Evolution system)
  - Risk: Potrebbe percepirsi come incrementale
  - Mitigation: Testing con focus group prima di lancio
- **Ipotesi 2**: Vanilla JS performance è sufficiente
  - Risk: Framerate cala su device vecchi
  - Mitigation: Profiling early, optimization continuous
- **Ipotesi 3**: localStorage è stabile per salvataggio (no cloud)
  - Risk: Data loss se localStorage corrotto
  - Mitigation: Implementare checksum + recovery mechanism

3.4 Risk Mitigation Dettagliata

Risk	Probabilità	Impatto	Mitigation	Owner
Race condition durante state change	Media	Alto	State machine con locking + unit tests	Dev

Risk	Probabilità	Impatto	Mitigation	Owner
Self-collision bug (serpente lungo)	Bassa	Alto	Spatial hashing + property-based testing	Dev
Performance calo su device lenti	Media	Medio	Early profiling + optimization checklist	Dev
Audio context suspension (mobile)	Alta	Basso	AudioContext lifecycle management	Dev
localStorage quota exceeded	Bassa	Medio	Quota checking + graceful fallback	Dev
Data loss localStorage	Molto Bassa	Alto	Checksum validation + recovery layer	Dev

4. Ambito del Prodotto (Scope Management)

4.1 Included Features (MVP)

✅ Meccanica Snake classica (movimento, crescita, collisioni)
✅ Sistema evoluzione 5 stadi con velocità incrementale
✅ Salvataggio automatico high score con checksum
✅ Leaderboard locale (top 10 + current session)
✅ Audio (BGM + SFX con toggle volume)
✅ Menu principale, game HUD, game over screen
✅ Input handling (keyboard + touch)
✅ Performance optimization (dirty rectangle rendering)
✅ Error handling robusto (state recovery)
✅ Property-based testing per edge cases

4.2 Excluded Features (Out of Scope)

❌ Multiplayer / Online leaderboard
❌ In-app purchases / Monetization
❌ Cloud sync / Cross-device progression
❌ Social features / Sharing
❌ Skin customization / Themes
❌ Multiple mappe / Level design
❌ Difficulty modes
❌ Achievements / Badge system
❌ Offline mode (richiede Service Worker v2)
❌ Analytics telemetry (roadmap v2)

4.3 Scope Freeze Policy

Lo scope è **frozen fino a completamento MVP**. Richieste di feature aggiuntive entrano in backlog per v1.1+.

5. Requisiti Funzionali Dettagliati

5.1 Requisiti Core Gameplay

RF-001: Movimento Serpente

- Serpente si muove continuamente in direzione corrente
- Input controls: Frecce tastiera / WASD / Touch swipe
- **Criticità**: Implementare input buffering per smooth gameplay
- **Acceptance Criteria**:
  - Movimento inizia al centro griglia (10,10)
  - Velocità: 100ms/step (base), accelerato da evoluzione
  - No 180° turns (previene auto-collision accidentale)

RF-002: Raccolta Cibo

- Cibo spawna randomicamente sulla griglia
- Serpente cresce quando tocca cibo
- Score incrementa di 10 punti per cibo raccolto
- **Acceptance Criteria**:
  - Spawn chance: 5% delle celle libere ogni 3-5 secondi
  - Growth: 1 segmento added alla coda
  - Visual/audio feedback immediato (< 50ms)

RF-003: Collision Detection (CRITICO)

- Game over se serpente tocca bordo griglia
- Game over se serpente si tocca il corpo (self-collision)
- **Acceptance Criteria**:
  - Boundary check: x < 0 || x >= 20 || y < 0 || y >= 20
  - Self-collision: Controlla da 4° segmento in poi
  - **Scalability**: Spatial hashing per serpenti > 50 segmenti
  - No false positives in edge cases

RF-004: Evolution System (INNOVAZIONE UNICA)

- Serpente evolve in 5 stadi basati su lunghezza
- Ogni stadio: colore, velocità, effetti visivi, audio
- **Acceptance Criteria**:
  - Stadio Base (0-10): Verde, velocità 100%, nessun effetto

- Stadio Crescente (11-25): Blu, +5% velocità, particle trail
- Stadio Consapevole (26-50): Verde scuro, +10%, trail glow
- Stadio Potenziato (51-75): Rosso, +15%, particle explosion
- Stadio Leggendario (76+): Oro, +20%, aura glitch
- Transizione animata 1 secondo con audio cue

6. Requisiti Non-Funzionali

6.1 Performance Requirements

Metrica	Target	Threshold
FPS	60	30 min (mobile)
Input Latency	16ms	50ms max
Load Time	1.5s	2s max
Memory	30MB	50MB max
CPU Usage	10%	25% max

6.2 Reliability Requirements

- **Availability**: 99.9% uptime durante gameplay
- **Data Integrity**: 100% accuracy su high scores
- **Error Recovery**: Game deve recoveryare da state corruption
- **Crash Rate**: < 0.5% delle sessioni

6.3 Security Requirements

- XSS Prevention: Sanitizzare input giocatore (playerName)
- GDPR Compliance: Allow data export/deletion
- Input Validation: Rate limiting + debouncing
- Storage: Checksum validation per localStorage

6.4 Compatibility Requirements

Browser	Min Version	Mobile
Chrome	60+	Yes
Firefox	55+	Yes
Safari	11+	Yes
Edge	79+	Yes

7. User Stories & Acceptance Criteria

Story 1: Giocare una partita completa

Come giocatore casual
Voglio giocare una sessione di Snake
Affinché possa rilassarmi e divertirmi

Acceptance Criteria:

- [ ] Posso avviare una nuova partita dal menu
- [ ] Il serpente inizia al centro con 3 segmenti
- [ ] Posso controllare il serpente con frecce/swipe
- [ ] Cibo appare sulla griglia
- [ ] Il mio score incrementa quando raccolgo cibo
- [ ] Partita termina su collisione
- [ ] Vedo il mio score finale e high score

Story 2: Osservare l'evoluzione del serpente

Come giocatore che cerca progression
Voglio vedere il mio serpente evolversi visivamente
Affinché possa sentire una sensazione di achievement

Acceptance Criteria:

- [ ] Al raggiungere 11 segmenti, serpente cambia colore a blu
- [ ] Al raggiungere 26 segmenti, serpente diventa verde scuro
- [ ] La velocità aumenta ad ogni evoluzione
- [ ] Un'animazione 1 secondo segnala la trasformazione
- [ ] Sento un suono di "upgrade" all'evoluzione
- [ ] L'HUD mostra lo stadio evolutivo corrente

Story 3: Competere coi miei record

Come giocatore competitivo
Voglio salvare e visualizzare i miei migliori score
Affinché possa battermi nel tempo

Acceptance Criteria:

- [ ] Il mio high score viene salvato automaticamente
- [ ] Lo score persiste dopo chiusura del browser
- [ ] Posso visualizzare la leaderboard dal menu

- [ ] Vedo i top 10 score con nomi e data
- [ ] Il mio nome viene salvato con lo score

8. Metriche di Successo & Monitoring

8.1 Metriche Funzionali

Metrica	Target	Metodo	Frequenza
Completion Rate (game completato)	40%+	Telemetry	Daily
Avg Score	150+ punti	Telemetry	Daily
Max Evolution Stage Reached	Stage 3+	Telemetry	Daily
Session Duration	5-10 min media	Telemetry	Daily

8.2 Metriche Tecniche

- **FPS Stability**: 95% sessioni ≥ 55 FPS
- **Crash Rate**: Track < 0.5%
- **Load Time**: P95 < 2s
- **Error Frequency**: < 1 error per 100 sessioni

8.3 Business Metrics

- **DAU (Daily Active Users)**: Target 100+
- **Retention D1/D7**: 50%/30%
- **Rating**: 4.0+ stelle
- **Download Velocity**: 200+/giorno prime 2 settimane

9. Roadmap & Timeline

Phase 1: Setup & Core (Settimane 1-3)

- Week 1: Architettura, setup build tools, game loop
- Week 2: Snake movement, collision detection, grid system
- Week 3: Food system, score tracking

Phase 2: Evolution & Polish (Settimane 4-5)

- Week 4: Evolution system, visual effects, audio
- Week 5: UI menus, game over screen, HUD

Phase 3: Persistence & Optimization (Settimana 6)

- Storage manager, leaderboard, localStorage checksum
- Performance profiling e optimizations
- Input handling refinements

Phase 4: Testing & Launch (Settimane 7-8)

- Unit testing (70%+ coverage)
- Property-based testing per edge cases
- Playtesting with external users
- Bug fixes e final optimization
- Deploy to web + app stores

10. Criteri di Accettazione Tecnici

CAT-001: Stabilità State Machine

- [x] Nessuna transizione di stato invalida
- [x] Race conditions identificate e risolte
- [x] State locking implementato
- [x] Recovery da corruption testato

CAT-002: Performance Garantiti

- [x] 60 FPS ≥ 95% sessioni
- [x] Load time < 2s con throttling 3G
- [x] Memory < 50MB runtime
- [x] Profiling report allegato

CAT-003: Data Integrity

- [x] Checksum validation per localStorage
- [x] Recovery mechanism testato
- [x] Zero data loss in stress testing
- [x] Quota exceeded handling implementato

CAT-004: Testing Coverage

- [x] Unit tests 70%+ coverage (core logic)
- [x] Property-based tests per collision + evolution
- [x] Integration tests per save/load
- [x] Chaos testing per edge cases

11. Appendici

11.1 Glossario

- **Evolution Stage**: Livello progressivo del serpente basato su lunghezza
- **Self-Collision**: Serpente che si tocca il proprio corpo
- **State Machine**: Gestione degli stati di gioco (MENU, PLAYING, GAMEOVER)
- **Dirty Rectangle**: Rendering solo aree della schermata cambiate
- **Spatial Hashing**: Tecnica di ottimizzazione per collision detection su larga scala

11.2 Riferimenti & Best Practices

- Segue principi di "Fail-Fast, Recover-Gracefully"
- Event-driven architecture per loose coupling
- Immutable state per predictability
- SOLID principles applicati dove possibile

Documento Versione 2.0 - Approvato per Sviluppo
Incorpora tutte le mitigazioni critiche identificate in revisione
Data: Novembre 2025