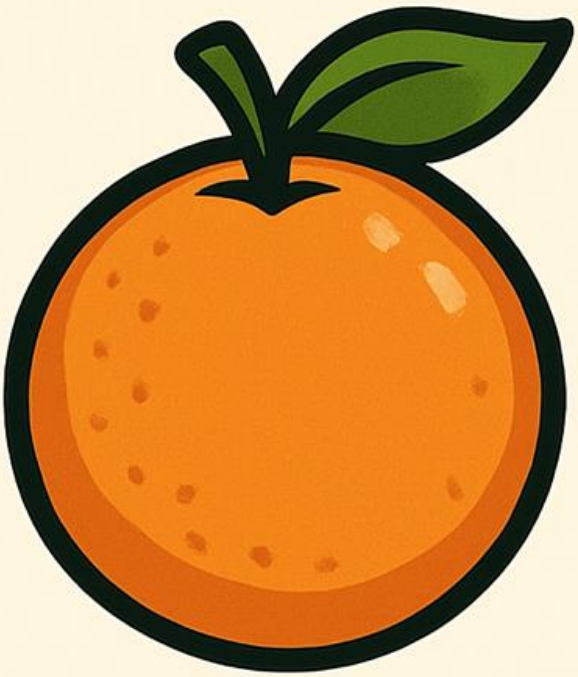

ORANGE GAMES

The Orange Ninja

Request for Proposal
Version 1.0

Document History

Version	When	Who	What
1.0	9/18/25	Aiden Weaver - 1.0,2.0 Syed Masum 6.0, Ro - 3.0, 8.0 Jaskaran – 4.0 Gurneet – 5.0, 10.0 Safal – 7.0, 9.0	Initial Drafting
2.0	9/19/25		Final Draft

Table of Contents

1.0 Problem description / opportunity / expression of need

The Rouge-lite genre is filled with serious, grungy games that are dark in tone. For a rouge-lite game to be a rouge-lite, it has to be difficult with a tone to match the relentless grind for progression. However, a game in this genre doesn't have to be that way. There is a place for the dark and serious games within the genre, but it should not dominate it. More rogue-lite games should be brighter and lighthearted.

2.0 Project Objectives

We aim to take the genre in a different direction by adding a very colorful and humorous game with an ironically serious tone. Instead of having the main interest of the game fighting through progression, we will instead focus on finding interesting and fun combinations that allow the player to feel like they are breaking the game with upgrades and weapon combinations.

You will progress through increasingly difficult levels with bosses along the way as you upgrade your character, the Orange Ninja, in fun and diverse ways.

The game will include:
Player (The Orange Ninja)
- keeps a record of current stats
- ability to move through levels
- can attack enemies and interact with objects
- can be upgraded and wield different weapons

- Interactables
  - objects inside the map the player can interact with
  - items dropped that the player can pick up
- Enemies
  - increasingly difficult enemies in each level
  - able to attack the player if inside battle radius
  - drops items and despawns when killed by player
- Menu/UI
  - when game is initiated a new/load game menu is generated
  - a settings menu can be accessed in game to include a help screen
  - health and equipped items HUD
  - can toggle audio on/off
- Audio
  - background music associated with the level theme
  - sound effects of player, enemy and interactables
  - compressed to decrease load times
- Levels
  - able to generate dungeons/floors randomly
  - scale difficulty to increase as the player progresses
  - populates maps with objects including player/enemies

3.0 Current system(s) – if any / similar systems

There are multiple systems in the market today, one example being *Dan the Man*. What we are planning is similar in structure, but with a playful twist. Instead of relying on a heavy, serious tone, our game will adopt a lighter, more humorous style, creating a unique experience within the rogue-lite genre.

While many existing rogue-lite games have achieved commercial success, they often miss an important element: variety in tone. That is where our project stands out. By blending the challenging rogue-lite environment with the lighthearted style of the Orange Ninja, we aim to deliver a fresh and engaging mix. Even the bosses will reflect this lighter tone to align with the overall theme and atmosphere of the game.

4.0 Intended users and their basic interaction with the system

Users:

- Casual players who just want quick sessions.
- Rogue-lite players who like testing upgrades and builds.
- Anyone who prefers a light, colorful style instead of a heavy tone.

How they interact with the system:

- **Gameplay:** Control the Orange Ninja, move through levels, fight enemies, and collect items.
- **Progression:** Pick up upgrades and weapons to make stronger combinations as they move through harder levels.
- **Menus/UI:** Use start menus, in-game HUD (health, items), and settings (sound, help).
- **Feedback:** Get rewards, drops, and progress through enemies and bosses, encouraging replay and new strategies.

5.0 Known interactions with other systems within or outside of the client organization.

1. GitHub Repository (External Collaboration System):
The project will be stored and managed using GitHub. This allows all team members to collaborate, track changes, and maintain version control of both code and documentation.

2. Unity Game Engine (External Development Platform):
The system depends on Unity for building, testing, and running the game. Unity integrates with code, assets, and third-party tools, making it essential for development and deployment.

3. Canvas Submission Portal (Client Requirement System):
All finalized proposals and deliverables will be submitted through Canvas as required by the instructor. This ensures proper documentation and feedback loops between the team and the client (instructor).

6.0 Known constraints to development

- **Upgrade**: We must make frequent upgrades which may require reworking existing code, which can slow progress.

- **Balance**: Balancing gameplay elements is time-consuming, as overpowered or weak features can reduce fairness and player engagement. Rigorous testing and implementation are necessary.

- **Bug Fixing**: Identifying and resolving bugs can delay project and make it hard for us to meet the deadline.

- **Unity**: While Unity provides flexibility, we are all still very new to the software and it is quite challenging to learn and make a working project in such a short time.

- **Devices**: A lot of the devices we have does not meet the recommended system requirements need to work on unity and make a product that delivers a high-quality experience.

- **Time Management**: With all the group members having a very busy schedule it is hard for us to find a perfect time to sit down and work together on the project.

7.0 Project Schedule

Date	Time	Objective
09/03	2 hours	Team setup: assign roles (programming, art, testing, documentation), create & initialize GitHub repository, set up Unity project
09/17	3 hours	Game design document (GDD) draft: story, gameplay mechanics, feature list, tech stack; review and finalize scope
09/25	30 min	Present initial concept & RPF to class/instructor
10/07	20 hours	Project skeleton: Unity scenes setup, player movement, camera system; Git branching & coding guidelines
10/30	50 hours	Core mechanics implemented (e.g., scoring system, win/lose conditions); basic UI (menus) Playable prototype: add enemies/NPCs, animations, Levels Preferable: Conduct first round of internal testing
11/10	20 hours	Mid-project milestone: game looks close to final; polish graphics, add audio; document known missing features

11/15	10 hours	Finalize gameplay loop: add special features (power-ups, extra levels), debug, optimize performance
11/25	10 hours	Comprehensive testing & QA: bug fixes, balance difficulty, user feedback
12/05	1 hour	Complete Project

8.0 How To Submit Proposals

All project proposals must be submitted electronically by specified deadlines. To ensure consistency and accessibility, the following submission process will be followed:

1. **File Format:** All proposals should be submitted as PDF documents.
2. **Repository Upload:** Each version will be stored in the team's GitHub repository under a dedicated "Documentation" folder for version control and collaboration.
3. **Submission:** A finalized proposal submitted through the canvas app.
4. **Version Tracking:** Each submission should include a version number, date, and list of contributors, clearly documented in the "Document History" section.
5. **Deadlines:** Proposals must be submitted before 11:59 PM on the assigned due date. Late submissions will only be accepted with prior instructor approval.

9.0 Glossary of terms

- Rogue-lite: A subgenre of video games with procedurally generated levels, permadeath, and repeat playthroughs. Unlike roguelikes, rogue-lite games often allow permanent upgrades between runs.
- Procedural Generation: A method of creating game content (levels, enemies, items) using algorithms instead of manual design, giving each playthrough unique variety.
- UI (User Interface): The on-screen system that allows the player to interact with the game, such as menus, buttons, and HUD.
- HUD (Heads-Up Display): In-game display of player stats such as health, weapons, or inventory.
- PC (Non-Playable Character): Game characters not controlled by the player, including enemies, bosses, or supporting roles.
- Boss: A challenging, high-level enemy typically appearing at the end of a level or section, requiring advanced strategy to defeat.

- Upgrade: Any item, skill, or power that improves the player's abilities or weapons, often used to progress through harder levels.
- GitHub Repository: A cloud-based platform used for version control and collaboration, storing code, assets, and documentation.
- Unity: A game development engine used to design, test, and deploy the Orange Ninja game across multiple platforms.
- Build: A compiled and playable version of the game created for testing or release.
- QA (Quality Assurance): The process of testing and debugging the game to identify issues, balance mechanics, and ensure stability.
- Prototype: An early, testable version of the game used to experiment with mechanics and features before full development.

*Note: Remember that "system" means product, service, and/or system your group would like to see created, built, upgraded, and/or changed. It is a broad term.