Game Design Document (GDD)

(Game Design Document (GDD) for ASCII Roguelike Game)

Game Title: "Edge of Oblivion"

Genre: ASCII Roguelike

Target Platform: PC(Unix, Windows)

Target Audience: Fans of roguelike games, strategy, and retro gamers.

1. Game Overview

In Edge of Oblivion, players explore a dystopian cyberpunk future on the dying planet Xalora, where cities are crumbling and resources are scarce. The planet faces destruction from an interdimensional union force called The Legion, who seek to replace it with a new universe. The only hope lies in the Singularity Core, a powerful weapon hidden deep within Xalora's core that can either save or destroy the planet.The player's main objective is to find this weapon and prevent the extermination of the planet. The maps take place in vast cyberpunk canalizations tunnels that stretch deep into the planet's core.

2. Core Gameplay Mechanics

2.1 Procedural Dungeon Generation

Map Layout: Dungeons are generated procedurally at the start of each new game, creating unique layouts with rooms, corridors, traps, and secret passages. Map is a tile-based grid.

Level Structure: Each dungeon level is represented in ASCII(emoji), with symbols representing various environmental features.

ENEMY = " 👽 "	WATER = " 🟪 "	SHARD = " 🔮 "
ENEMY_SHOOT = " 🤖 "	PLANT = " 🌿 "	FIRE = " 🔥 "
ENEMY_MAGE = " 🧙‍♀️ "	WEAPON = " 🔫 "	SLOW = " ⏳ "
ENEMY_SPRINT = " 👻 ”	FLOOR = " ⚫ "	PILL = " 💊 "
DOOR = " 🚪 "	KEY = " 🔑 "	
TRAP = " 💀 "	POTION_SP = " 🍷 "	
POTION_HP = " 🧪 "	COIN = " 🪙 "	
SCROLL = " 📜 "	REAGEN = " 🧬 "	
EXIT = " 🔴 "	ARMOR = " 🛡 "	

2.2 Turn-Based Combat

Each turn, players can attack only after that it's enemy turn to attack. Enemies take their turns after the player or during collisions. Depending on enemy type there are different fighting types.Fighting loop is running till the moment when either enemy or player is dead. There is a possibility of using weapons and armour what player can buy during the map traversal. Enemies can use special abilities depending on type of enemy.

Melee attacks(close range physical combat) happen when the player and an enemy are next to each other. Ranged attacks and spells are launched from a distance.

Actions include managing health (HP), stamina(SP) and perks(abilities)

2.3 Character Classes and Progression

Stats and classes:

Stats: SP(stamina), HP(Health points), EXP(experience)

Classes:

- Outcast - Balanced in everything. Decent with a gun. Has great luck of finding reagens.
  STR: Average HLF: Average INT: Average
- Byte - Smart but weak. Great with tech. Can detonate dynamites.
  STR: Low HLF: Average INT: High
- Venom - Strong and tough. Can stop time and make enemies slower.
  STR: High HLF: Average INT: Low

Leveling System: Experience points are gained by defeating enemies so that you can gain new skills or money. During the game you can find Reagent Flasks and buy them with money you gained which can boost your health or strength. Also you can craft shards with reagens what you can use to have good ending.

2.4 Items and Equipment

- Weapons: Armour, Riffle, Special abilities(bomb or timer(depending on player type))
- Potions: Reagent Flasks (two types: one - boost lives(health) second - strength(experience))
- Scrolls: Small tips or lore drops
- Keys and doors: behind every door there are a shard, you need a key to unlock it.
- Reagen: use them to upgrade stats or craft shards.
- Gold: currency ($) used to buy all items in the game.
- Plant: You can hide behind it or find new reagen.
- Cost of all items depends on player level!

2.5 Environmental Hazards

- Traps: Hidden throughout the dungeon, such as spike pits poison gas etc. Traps can be avoided
- through careful navigation.
- Locked Doors: Require keys.
- Teleporters: Randomly transport the player to another part of the map

2.6 Enemies

Each room(tunnel) is populated with monsters of varying difficulty.

- Galactic Warriors: (common enemy direct attack, weak, swarm in numbers). *(Difficulty:1)*
- Quick Spirit: (can go anywhere on the map, movement is not limited). *(Difficulty:3)*
- Shooting ninja: Can attack from distance to defeat you need a weapon. *(Difficulty: 5)*
- Bio Mage: There are 3 types of mages: *(Difficulty: 7)*
  - one has ability to teleport,
  - second has ability to heal
  - third can attack twice

Collisions and Combat: Collisions can result in death or health loss. Advanced bosses may place traps, follow the player, or attack remotely.

2.7 Permadeath

If the player dies, the game is over. No saving or reloading. Each playthrough is unique due to procedural. generation.

3. Game Controls and AI

Move character(Arrow keys)

- I Inventory
- E Attack (melee)
- Q Attack(weapon)

3.1 Graphical User Interface

ASCII-based graphics. Emoji style. Curses python library

3.2 AI

Player has 4 states: HAPPY, DEPRESSED, IDLE and ANGRY.

1. ANGRY – when player's HP is almost 0, player start attacking with more power and has more damage points.
2. DEPRESSED – when player doesn't have enough medicaments(pills) the player is depressed and inflict less damage.
3. IDLE – basic state
4. HAPPY – when player has a lot of medicaments, it start attacking with more power and has more damage points.

Enemy AI: Enemy moves in any direction. Decision are made depending on position of different objects, if player is nearby enemy initiate attack and shooting enemy can inflict damage from the distance and also enemies can steal valuable items like money or keys.

4.Procedural Map Generation

Each map is generated using one of three distinct techniques tailored to the thematic and gameplay needs of the levels. Below is an overview of the methods employed:

4.1 Cellular Automata (Middle Levels: Forest)

Cellular automata are used to generate the forest levels, creating organic, natural-looking terrain with a blend of open spaces and dense areas. This method simulates natural processes by iteratively refining the map through a series of rules:

- Initial Seed: A random distribution of tiles representing walls and open spaces based on the noise grid. (Flood fill required)
- Rules: Based on the number of adjacent wall tiles, cells transition to walls or open spaces.
- Result: Dense, maze-like environments with occasional clearings, enhancing exploration and survival mechanics.

4.2 Drunkard's Walk (First Levels: City)

The drunkard's walk algorithm is employed for the first levels set in the city. This method creates winding, labyrinthine paths that feel like urban alleyways and streets. (Flood fill required)

- Algorithm: A "walker" moves randomly, carving paths through a grid-based map. The process continues until a desired amount of floor space is generated.
- Features: Narrow corridors and unexpected turns simulate a chaotic, sprawling cityscape.
- Gameplay Impact: Promotes tension and exploration as players navigate tight, unpredictable spaces.

4.3 BSP Tree (Final Levels: Canalization)

The final levels, set in the canalization, utilize Binary Space Partitioning (BSP) to generate structured, dungeon-like maps. This method ensures logical, interconnected layouts suitable for the climactic moments of the game.

- Algorithm: The map is recursively divided into smaller rectangular sections.
- Room Placement: Rooms are placed within partitions, with corridors connecting them.
- Result: A structured but non-linear design, allowing strategic gameplay and preparation for final challenges.

4.4 Dynamic Variety and Replayability

By combining these three methods, each level feels unique while maintaining thematic cohesion:

- Early city levels is exploration of chaotic streets to get to know the planet environment.
- Middle forest levels shift focus to survival and adaptability in more organic terrain.
- Final canalization levels test player skill and strategy with structured, dungeon-like challenges. Canalization is a final level where you will get access to the final safe where will be Singularity Core.

5. Story and Lore

5.1 Plot Overview

The main events take place in a dystopian cyberpunk future on the planet Xalora, which is on the brink of destruction. The planet is nearing total ruin—resources are scarce, cities are crumbling, and an overwhelming sense of decay permeates society. The primary enemies in this game are a union of other universes called The Legion, who seek to rearrange the cosmic order and create a new universe in its place. The only hope lies in a hidden weapon buried deep within Xalora's core, known as the Singularity Core—a device with the potential to either bring prosperity or cause destruction. This weapon holds the power to stop the Legion, and the player must find it to save the universe.

5.2 Setting

Edge of Oblivion:(Roguelike/Cyberpunk/Sci-Fi): A dark, roguelike adventure on a dying planet, where players navigate decaying cities, tech ruins, and battle interdimensional threats

6. Technical Specifications

Graphics: ASCII-based rendering using the terminal or text-based display.

Engine: Simple custom-built or existing roguelike python library.

Performance: Low system requirements due to minimal graphics and text-based nature.

Minimal System Requirements: OS: Windows 7/10 or Linux; CPU: 1.2 GHz Dual-Core Processor; RAM: 2 GB; Storage: 300 MB available space;

Graphics: Integrated graphics or terminal-based rendering; Input: Keyboard (no mouse required)

Programming Language: Python

7. Development and gameplay parts

Procedural Dungeon Generation: Creating random dungeon layouts with rooms, corridors, and increasing difficulty.
Player Movement, Combat, and Inventory: Implementing basic tile-based movement, turn-based combat, and an system of items to gain during the game, which add aditional combat features like abilities.
Enemy Combat: Developing enemies and bosses with basic attack patterns, stats, and behaviors. Adjusting combat difficulty, damage, and defense with every level.
Content Creation: Adding character classes, unique abilities, weapons, and items. Designing dungeon environments with traps and hazards.