

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería



Desarrollo de un videojuego con contenido personalizado mediante la integración de *Large Language Models* y técnicas de generación procedural

Trabajo de graduación en modalidad de Trabajo Profesional presentado por

Cristian Aguirre

Para optar al grado académico de Licenciado en Ingeniería en Ciencias de la Computación y Tecnologías de la Información

Guatemala, enero del 2025

Vo.Bo.:

(f) _____
Ing. Alhvi Barcarcel

Tribunal Examinador:

(f) _____
Ing. Alhvi Barcarcel

(f) _____
MSc. Eddy Omar Castro Jauregi

(f) _____
MSc. Marcos Antonio Gutierrez Suárez

Fecha de aprobación: Guatemala, 02 de diciembre de 2024.

Agradecimientos

Quisiera comenzar expresando mi más sincero agradecimiento a mi asesora de proyecto, la Ing. Alhvi Romancina Balcarcel Rodas, quien me guió y orientó a lo largo de este proyecto con su experiencia, paciencia y dedicación. Su apoyo constante fue fundamental para superar los desafíos que surgieron en el proceso y su confianza en mí me impulsó a seguir adelante y superar los desafíos.

A mi familia, especialmente a mis padres, les agradezco de todo corazón el amor incondicional y el apoyo constante que me han brindado desde el primer día. Su confianza en mí ha sido el impulso que me permitió llegar hasta aquí. A mis hermanos, gracias por sus palabras de aliento y por ser un ejemplo a seguir en este camino. A mi novia, gracias por tu compañía, ánimo y comprensión a lo largo de este proceso. Sin ustedes, este logro no habría sido posible.

A la Universidad del Valle de Guatemala, mi agradecimiento por ofrecerme un espacio para desarrollarme tanto académica como profesionalmente. También extiendo mi gratitud al Departamento de Ingeniería en Ciencias de la Computación y Tecnologías de la Información, por la confianza depositada en mi proyecto y el ambiente de aprendizaje que me brindaron.

Finalmente, a mis amigos y compañeros, gracias por estar a mi lado en los momentos de estrés y en los de alegría. Su apoyo y amistad fueron un soporte y me ayudaron a seguir adelante en los momentos más difíciles. Cada uno de ustedes hizo que este proceso fuera más especial e inolvidable.

Índice

Agradecimientos	III
Lista de Figuras	VIII
Lista de Cuadros	IX
Resumen	X
1. Introducción	1
2. Objetivos	2
2.1. Objetivo General	2
2.2. Objetivos Específicos	2
3. Justificación	3
4. Antecedentes	4
5. Alcance	5
6. Marco Teórico	6
6.1. Videojuegos	6
6.1.1. NPCs	6
6.1.2. Mecánica de juego	6
6.1.3. Pruebas de jugabilidad	6
6.2. Herramientas de desarrollo	7
6.2.1. Unity	7
6.2.2. C#	7
6.2.3. Unity Asset Store	7
6.3. Generación Procedural de Contenido	7
6.3.1. Aleatoriedad Controlada	8
6.3.2. Función de Distribución	8
6.3.3. Algoritmo Random Walk	8
6.3.4. Algoritmo Poisson Disk Sampling	9
6.4. Inteligencia Artificial	9
6.4.1. <i>Large Language Models (LLM)</i>	9
6.4.2. Machine Learning (ML)	9
6.4.3. Redes Neuronales	10

6.4.4. Procesamiento de Lenguaje Natural (NLP)	10
6.4.5. OpenAI	10
6.4.6. API de OpenAI	10
6.4.7. Inworld	11
6.4.8. Plataforma Hugging Face	11
6.4.9. Azure AI	11
7. Metodología	12
7.1. Proceso de Diseño del Videojuego	12
7.1.1. Conceptualización	12
7.1.2. Género	13
7.1.3. Elección de plataforma de Desarrollo	13
7.2. Desarrollo del Videojuego Base	13
7.2.1. Recolección de Assets	13
7.2.2. Configuración de Unity	13
7.2.3. Construcción del Mapa Inicial	14
7.3. Definición e integración de mecánicas de juego	15
7.3.1. Mecánicas de Movimiento	15
7.3.2. Mecánicas de interacción	16
7.3.3. Mecánicas de recolección	16
7.3.4. Condición de victoria	16
7.4. Investigación de Técnicas de Generación Procedural de Contenido	17
7.4.1. Estudio de Algoritmos	17
7.5. Integración de Técnicas de Generación Procedural de Contenido	17
7.5.1. Selección aleatoria de objetos	17
7.5.2. Generación con características aleatorias en NPCs	18
7.5.3. Aplicación de materiales y elementos adicionales	18
7.5.4. Generación de laberinto	19
7.5.5. Generación de sector orgánico del mapa	19
7.5.6. Colocación aleatoria de objetos en el sector generado	20
7.6. Integración de <i>Large Language Models</i>	21
7.6.1. Configuración inicial de API de OpenAI	21
7.6.2. Desarrollo de contenido contextual	22
7.6.3. Integración de NPCs Dinámicos	23
7.7. Pruebas de Validación	24
7.7.1. Validación de interacción con NPCs	25
7.7.2. Validación del atractivo Visual	26
7.7.3. Validación de mecánicas	26
7.8. Costos y consumo de API	27
8. Resultados	28
8.1. Escenario base para fase inicial del videojuego	28
8.2. Gameplay inicial	29
8.3. Generación procedural de contenido	30
8.3.1. Generación de área orgánica	30
8.3.2. Generación de laberinto	30
8.3.3. Posicionamiento de objetos ornamentales	31
8.3.4. Generación de NPCs con características distintas	31
8.3.5. Distribución de elementos objetivo	32
8.4. Generación de contenido con LLMs	33
8.5. Interacción con NPCs alimentados por LLMs	35
8.6. UI	36
8.7. Pruebas de jugabilidad	37
8.7.1. Validación de atractivo visual	37

8.7.2. Validación de aceptación en mecánicas de juego	38
8.7.3. Validación de interacción con NPCs	40
8.8. Análisis de Costos y Consumo de API	41
8.8.1. Consumo de API en una Sesión de Juego	41
8.8.2. Gastos Asociados a una Sesión de Juego	42
8.8.3. Consumo de API en un Mes de Pruebas	43
8.8.4. Costos Acumulados en un Mes de Pruebas	44
9. Análisis de resultados	45
10. Conclusiones	47
11. Recomendaciones	48
Glosario	49
Bibliografía	51
Anexos	52
1. Ejemplo básico de integración de API de OpenAI en C# y Unity	52
2. Dashboard de Costos y Actividad de la API de OpenAI	53
.2.1. Dashboard de Costos	53
.2.2. Dashboard de Actividad	53

Lista de Figuras

6.1. Ejemplo de ejecución de algoritmo random walk	8
6.2. Cuadro comparativo Poisson Disk Sampling vs Distribución aleatoria	9
7.1. Grid establecido en eje Z para crear un plano horizontal.	14
7.2. Colocación de tiles en grid configurado por layers	15
7.3. Concepto de chat a desplegar al momento de interactuar con un personaje	16
7.4. Bloque de terreno diseñado para formar secciones orgánicas en el mapa, utilizado como base en la generación procedural.	20
7.5. Sesión de juego	25
8.1. Escenario base utilizado como punto inicial del videojuego.	29
8.2. Vista principal del juego en ejecución.	29
8.3. Área generada en primera ejecución.	30
8.4. Área generada en segunda ejecución.	30
8.5. Configuración inicial de laberinto	30
8.6. Generación de laberinto terminada	30
8.7. Ejemplo de posicionamiento procedural de objetos ornamentales, como árboles, rocas y vegetación, utilizando algoritmo de Poisson Disk Sampling.	31
8.8. Generación de NPCs con características visuales distintas	32
8.9. Ejemplo de objeto requerido posicionado aleatoriamente en un área del mapa	32
8.10. Ejemplo de interacción con un NPC para la narrativa contextual del mundo.	35
8.11. Ejemplo de interacción con un NPC para solicitar indicaciones.	36
8.12. Ejemplo de generación procedural de la receta necesaria para obtener la victoria en el videojuego.	36
8.13. Grafico de barras que muestra la percepción de los jugadores sobre el atractivo visual del juego.	38
8.14. Gráfico de barras que muestra la influencia del diseño visual del juego en el interés de los jugadores.	38
8.15. Gráfico de barras que muestra el nivel de involucramiento de los jugadores	39
8.16. Gráfico de barras que muestra la percepción de los jugadores sobre la intuición de las mecánicas del juego.	39
8.17. Gráfico de barras que muestra la percepción de los jugadores sobre la coherencia de las respuestas de los NPCs con el contexto del juego.	40
8.18. Gráfico circular que muestra si los jugadores notaron diferencias en las interacciones con los NPCs después de jugar dos veces.	41
8.19. Gráfico de barras que muestra la percepción de los jugadores sobre la coherencia de las respuestas de los NPCs con el contexto del juego.	41

8.20. Número de solicitudes a la API realizadas en el mes de noviembre.	42
8.21. Cantidad de tokens consumidos en el mes de noviembre.	42
8.22. Gastos mensuales acumulados en el uso de la API de OpenAI al inicio del mes de noviembre.	43
8.23. Número total de solicitudes a la API durante el mes de octubre.	43
8.24. Cantidad total de tokens consumidos durante el mes de octubre.	44
8.25. Gastos mensuales acumulados en el uso de la API de OpenAI durante el mes de octubre.	44
1. Desglose de costos mensuales acumulados en el uso de la API de OpenAI en octubre.	53
2. Monitoreo de actividad en la API de OpenAI, mostrando el uso de tokens y las solicitudes en octubre.	53

Lista de Cuadros

7.1. Preguntas de validación para la interacción con NPCs	25
7.2. Preguntas de validación para el atractivo visual del juego.	26
7.3. Preguntas de validación para las mecánicas del juego.	26
8.1. Validación de la variación en las historias generadas sobre el origen del pueblo.	33
8.2. Leyendas generadas alrededor del pueblo de Valle Sereno.	34
8.3. Personajes en la historia de Valle Sereno.	34
8.4. Enemigos en la historia de Valle Sereno.	35
8.5. Comentarios recibidos en las sesiones de prueba.	37
8.6. Momentos de inmersión en el juego según los jugadores	40

Resumen

Este trabajo explora la integración de *Large Language Models* y técnicas de generación procedural de contenido en el desarrollo de videojuegos, con el objetivo de ofrecer experiencias de juego altamente inmersivas, personalizadas y estéticamente atractivas. A través de un enfoque interdisciplinario que combina el desarrollo de software, la inteligencia artificial, la narratología y el arte digital, se investiga cómo estas tecnologías emergentes pueden transformar el diseño de videojuegos, permitiendo la creación de narrativas únicas y mundos generativos que ofrecen interacciones distintas a cada jugador, todo ello enmarcado en un estilo visual atractivo y de alta calidad.

Este estudio combina teorías algorítmicas consolidadas con modelos de inteligencia artificial avanzados para facilitar interacciones dinámicas y en tiempo real. Esta integración busca innovar en el diseño de videojuegos, combinando prácticas computacionales establecidas con avances tecnológicos recientes. El enfoque resultante se centra en expandir las fronteras creativas y tecnológicas, estableciendo nuevos estándares para la producción de experiencias lúdicas que son visualmente impactantes y altamente interactivas.

Los resultados obtenidos incluyen el desarrollo de un prototipo de videojuego que no solo ejemplifique estas técnicas innovadoras, sino que también cumpla con altos estándares estéticos, proporcionando un marco conceptual y práctico que contribuya al avance de la industria del entretenimiento interactivo. Este estudio pretende no solo aportar al conocimiento existente en el campo, sino también abrir nuevas vías para la creatividad, la innovación y la expresión artística en el diseño de videojuegos.

CAPÍTULO 1

Introducción

Desde su concepción en la década de 1960, la industria de los videojuegos ha experimentado una transformación dramática, pasando de simples líneas y formas a complejos y envolventes mundos virtuales. Impulsado por avances tecnológicos y una demanda creciente de experiencias interactivas más inmersivas y personalizadas, este crecimiento ha redefinido las expectativas de una base de jugadores diversa y en expansión [8]. Cada innovación ha expandido el alcance e influencia cultural de los videojuegos, consolidándolos como una forma dominante de entretenimiento que ahora compite con industrias tradicionales como el cine y la televisión, demostrando ser un verdadero rey del entretenimiento del siglo XXI [15].

La adopción de modelos de lenguaje grande, potenciados por los últimos avances en inteligencia artificial y aprendizaje automático, ha revolucionado la forma en que los desarrolladores construyen personajes y diálogos. Estos modelos analizan y generan texto que emula el estilo y la estructura del lenguaje humano, permitiendo a los creadores elaborar tramas más complejas y emocionantes. Esta capacidad de generar contenido narrativo de forma dinámica ha abierto nuevas vías para personalizar y enriquecer las interacciones en los videojuegos, ofreciendo experiencias más profundas y envolventes.

En paralelo, la generación procedural de contenido ha transformado el diseño de mundos de juego. En lugar de diseñar manualmente cada elemento, los desarrolladores pueden emplear algoritmos para crear paisajes y estructuras automáticamente y en tiempo real. Esta técnica no solo ahorra recursos sino que también enriquece la experiencia de juego con variaciones únicas en cada sesión, asegurando que no haya dos partidas iguales y manteniendo la frescura y el factor sorpresa en cada juego.

Finalmente, aunque la integración de modelos de lenguaje grande y técnicas de generación procedural muestra cómo la inteligencia artificial y los métodos tradicionales pueden profundizar la inmersión y complejidad de los juegos, es crucial mantener un equilibrio en cada aspecto del desarrollo de videojuegos. Los jugadores modernos valoran tanto la innovación tecnológica como la calidad visual. Por ello, el desafío para los desarrolladores radica en encontrar una armonía que no solo impulse la mecánica y las tecnologías de juego, sino que también satisfaga las altas expectativas estéticas actuales así como el énfasis en la calidad de jugabilidad. Así, la innovación debe ser tanto técnica como visualmente atractiva, asegurando que los videojuegos continúen siendo experiencias envolventes y visualmente impresionantes.

CAPÍTULO 2

Objetivos

2.1. Objetivo General

Desarrollar un videojuego que integre narrativas personalizadas y mundos generativos mediante la aplicación de modelos de lenguaje grande y técnicas de generación procedural de contenido, con el fin de ofrecer a los jugadores experiencias de juego altamente inmersivas y personalizadas.

2.2. Objetivos Específicos

- Implementar técnicas de generación procedural de contenido para ofrecer experiencias de juego únicas y personalizadas a cada jugador.
- Integrar modelos de lenguaje grande para potenciar la interacción con personajes no jugables (NPCs), agregando variabilidad a la narrativa del juego de manera dinámica y realista.
- Desarrollar un diseño visual atractivo para el videojuego, evaluando su aceptación mediante pruebas con usuarios.
- Implementar mecánicas de juego que mantengan a los jugadores comprometidos y ofrezcan una experiencia envolvente.

CAPÍTULO 3

Justificación

La industria del entretenimiento interactivo ha experimentado una evolución sin precedentes en las últimas décadas, impulsada por avances tecnológicos que permiten ofrecer gráficos cada vez más realistas y experiencias de juego más inmersivas. La importancia de mantener y mejorar la calidad de los productos desarrollados se refleja en movimientos estratégicos del mercado, como la adquisición de Activision Blizzard por Microsoft en 2022 por aproximadamente 69,000 millones de dólares, destacando la lucratividad y el potencial de crecimiento continuo del sector. En 2023, la industria generó más de 180,000 millones de dólares, con un notable aumento de contribuciones provenientes de plataformas móviles [26].

En este contexto dinámico, el campo de la investigación en videojuegos se presenta como un terreno fértil para la experimentación y la creación de nuevas formas de entretenimiento digital. Explorar cómo las tecnologías emergentes pueden mejorar la calidad e innovación en el desarrollo de videojuegos es esencial para mantenerse a la vanguardia en una industria en constante cambio. Este proyecto busca contribuir al cuerpo de conocimiento existente, proporcionando nuevas perspectivas sobre cómo la integración de modelos de lenguaje grande y técnicas de generación procedural puede enriquecer la experiencia de juego.

A través de un enfoque interdisciplinario que combina la informática, la inteligencia artificial y la narratología, se pretende explorar cómo la narrativa puede ser personalizada y adaptativa, ofreciendo a los jugadores una experiencia inmersiva y entretenida. Al avanzar en la comprensión de estos conceptos, se espera allanar el camino para futuras innovaciones en el diseño de videojuegos y abrir nuevas posibilidades para la creatividad y la expresión artística en el mundo del entretenimiento interactivo.

CAPÍTULO 4

Antecedentes

En la vanguardia del desarrollo de videojuegos, proyectos como 'The Endless Mission' y 'No Man's Sky' destacan por su uso innovador de tecnologías emergentes. Sean Murray de Hello Games describe 'No Man's Sky' como un esfuerzo por crear un universo expansivo donde cada estrella, planeta, flora y fauna son generados proceduralmente, ofreciendo una experiencia única a cada jugador"^[16]. Esta cita subraya la centralidad de la generación procedural en la visión del juego, que busca ofrecer una inmersión sin precedentes en un universo continuamente expansivo. Dicho juego ha revolucionado el uso de la generación procedural para crear un universo expansivo, donde cada estrella, planeta, flora y fauna son generados proceduralmente, ofreciendo una experiencia única a cada jugador.

Por otra parte, 'The Endless Mission', desarrollado por E-Line Media, se sumerge en un ámbito educativo dentro de la industria de los videojuegos. Este proyecto único permite a los jugadores no solo disfrutar del juego, sino también modificarlo y entender su estructura interna. Según Brenden Sewell, el Director Creativo, 'The Endless Mission' es una plataforma que introduce a los jugadores en la creación de videojuegos hasta el nivel de código. 'Vas a explorar diferentes géneros de juegos y usar herramientas poderosas para crear tus propias experiencias, lo que te permite entender cómo el software configura la experiencia del juego' [6]. Este enfoque busca democratizar el desarrollo de videojuegos, proporcionando a los jugadores herramientas para que se conviertan en creadores, y al mismo tiempo, vivan experiencias inmersivas y entretenidas.

Estos ejemplos ilustran cómo la generación procedural está enriqueciendo las experiencias de juego actuales, ofreciendo mundos únicos y expansivos. Esta innovación subraya la importancia de desarrollar técnicas que permitan experiencias de juego más personalizadas. Adicionalmente, el uso emergente de Modelos de Lenguaje Grande (LLMs) se perfila como un complemento prometedor, potencializando aún más la personalización y la interacción en los videojuegos. Aunque aún en etapas iniciales, proyectos como AI Dungeon [13] exploran estas tecnologías para crear interacciones dinámicas y adaptativas, marcando el camino hacia futuras innovaciones que podrían transformar significativamente la industria del videojuego.

CAPÍTULO 5

Alcance

Este proyecto se centra en el desarrollo de un videojuego 3D, que contará con un mapa principal dividido en tres secciones diferenciadas: una villa, un laberinto y un bosque, cada una con sus propios desafíos y características. Incluirá un personaje principal con controles de movimiento y varios personajes no jugables (NPCs) con los que los usuarios podrán interactuar. Además, se integrarán enemigos que añadirán complejidad a la jugabilidad.

Para garantizar que cada sesión ofrezca una experiencia única, el diseño incorporará la generación procedural de contenido y la interacción dinámica con los personajes no jugables (NPCs). Este diseño no solo facilitará la evaluación de la jugabilidad sino que también servirá como un indicador fundamental de que el proyecto ha alcanzado su completitud y está listo para pasar a las fases de prueba y validación.

Dado que el enfoque de este proyecto es experimental y de investigación, no se desarrollarán mecánicas o narrativas excesivamente complejas. El objetivo es explorar la integración de técnicas de generación procedural, inteligencia artificial y desarrollo de videojuegos sin enfocarse en la post-producción ni en actualizaciones continuas del juego. Asimismo, este proyecto no incluirá fases de despliegue que requieran análisis y logística adicionales.

Se presupone la disponibilidad de herramientas de desarrollo necesarias, particularmente la API de OpenAI, cuyo uso podría incurrir en costos adicionales que serán considerados en la evaluación del proyecto. El período de desarrollo está estimado en aproximadamente seis meses, lo que impone limitaciones temporales y presupuestarias al alcance del trabajo.

CAPÍTULO 6

Marco Teórico

6.1. Videojuegos

6.1.1. NPCs

Un NPC (Non-Player Character) es un personaje que no está bajo el control del jugador en ciertos entornos; los cuales simulan el comportamiento y la racionalidad de los jugadores reales, con poca o nula interacción humana. Los NPCs son controlados por inteligencia artificial, la cual se basa en un conjunto de reglas ya establecidas que buscan replicar el comportamiento más adecuado según los datos del entorno. Sin embargo, entre las limitaciones de dicha inteligencia artificial, se destaca la falta de comportamiento natural, lo que a menudo es percibido por el jugador como ilógico o predecible [21].

6.1.2. Mecánica de juego

Una mecánica de juego se refiere a las reglas y sistemas que rigen las interacciones y el comportamiento dentro de un videojuego. Son los componentes fundamentales que definen cómo se juega y cómo los jugadores pueden interactuar con el mundo del juego y sus elementos. Las mecánicas de juego pueden incluir aspectos como el movimiento del personaje, la gestión de recursos, la resolución de *puzzles*, el combate, entre otros. La correcta implementación y balanceo de estas mecánicas es crucial para crear una experiencia de juego atractiva y entretenida [1].

6.1.3. Pruebas de jugabilidad

Las pruebas de jugabilidad son una metodología crucial en el desarrollo de videojuegos, ya que permiten evaluar cómo los jugadores interactúan con el juego y qué tan intuitiva y satisfactoria es su experiencia. Estas pruebas se llevan a cabo mediante la observación directa de jugadores mientras interactúan con el juego, así como a través de encuestas y entrevistas para recopilar retroalimentación sobre aspectos específicos del diseño y la jugabilidad. El objetivo es identificar y corregir problemas de usabilidad, como la dificultad de navegación, la comprensión de las mecánicas del juego, y la

accesibilidad, para asegurar que el juego sea agradable y fácil de jugar para una amplia audiencia [11].

6.2. Herramientas de desarrollo

6.2.1. Unity

Unity es un motor de videojuegos creado por la empresa del mismo nombre en el cual se pueden desarrollar aplicaciones en 2D, 3D, realidad aumentada (AR) y realidad virtual (VR). A lo largo de los últimos años, esta plataforma se ha vuelto extremadamente popular debido a su flexibilidad y facilidad de uso [5]. Unity es una herramienta *multiplataforma*, lo que permite desarrollar juegos para diversos sistemas operativos, incluidos macOS y Linux, así como para dispositivos móviles como iOS y Android. Además, es compatible con consolas de videojuegos como PlayStation y Xbox [23]. La plataforma cuenta con un entorno de desarrollo integrado (IDE) que incluye un editor visual intuitivo, permitiendo a los desarrolladores mover objetos, realizar cambios y ajustar propiedades en tiempo real. Esto facilita considerablemente la construcción y el diseño de videojuegos [22].

6.2.2. C#

C# es un lenguaje de programación desarrollado por Microsoft que se utiliza ampliamente en el desarrollo de aplicaciones y videojuegos en Unity. Su sintaxis es similar a otros lenguajes de programación como Java y C++, lo que facilita su aprendizaje para los desarrolladores con experiencia en estos lenguajes. C# es un lenguaje orientado a objetos, lo que permite una estructura de código modular y reutilizable, facilitando la gestión de proyectos grandes y complejos [10]. Unity utiliza C# como su principal lenguaje de scripting, proporcionando a los desarrolladores una poderosa herramienta para crear lógica de juego, gestionar la física, controlar animaciones y mucho más. La integración de C# en Unity permite la creación de scripts personalizados que pueden interactuar directamente con el motor de juego, ofreciendo flexibilidad y control detallado sobre el comportamiento de los elementos en el juego.

6.2.3. Unity Asset Store

La Unity Asset Store es un mercado en línea donde los desarrolladores pueden comprar y vender activos que se utilizan en el desarrollo de videojuegos. Estos activos incluyen modelos 3D, texturas, sonidos, herramientas de script, paquetes de animación, y otros recursos que pueden ahorrar tiempo y esfuerzo durante el desarrollo del juego. La Asset Store facilita la colaboración y el intercambio de recursos entre desarrolladores, permitiendo que incluso pequeños equipos de desarrollo o desarrolladores independientes puedan acceder a activos de alta calidad sin necesidad de crearlos desde cero. Esta plataforma contribuye a acelerar el desarrollo y mejorar la calidad de los videojuegos producidos con Unity [22].

6.3. Generación Procedural de Contenido

La generación procedural de contenido es un campo en el desarrollo de videojuegos que se centra en la creación automática de contenido dentro de los juegos utilizando algoritmos sofisticados. Esta técnica permite generar elementos del juego como niveles, mapas, texturas, y narrativas sin intervención directa del usuario, lo que la hace crucial para ampliar la variedad y la profundidad de

los juegos. Al emplear PCG, los desarrolladores pueden producir un número infinito de variaciones en el contenido del juego, asegurando que cada sesión de juego sea única y fresca. Este enfoque no solo mejora la rejugabilidad de un juego sino que también reduce significativamente el tiempo y los recursos necesarios para desarrollar grandes volúmenes de contenido. En el contexto de juegos, PCG es especialmente valorado por su capacidad para adaptar y optimizar el contenido generado para que sea jugable y desafiante, manteniendo la coherencia con el diseño y las restricciones del juego [20].

6.3.1. Aleatoriedad Controlada

La Aleatoriedad Controlada se refiere a la utilización de algoritmos que generan contenido de manera aleatoria, pero con restricciones específicas para asegurar coherencia y equilibrio en el juego. Esta técnica es esencial en la generación procedural de contenido para evitar resultados completamente impredecibles que podrían desbalancear el juego o romper la inmersión del jugador [20].

6.3.2. Función de Distribución

Una Función de Distribución es una función matemática que describe todas las probabilidades de los posibles valores de una variable aleatoria. En los videojuegos, las funciones de distribución son utilizadas para modelar la aparición de eventos y objetos de manera que se garantice una experiencia de juego balanceada y emocionante. Estas funciones ayudan a controlar la aleatoriedad en la generación procedural de contenido, asegurando que el juego sea justo y desafiante [9].

6.3.3. Algoritmo Random Walk

El algoritmo Random Walk es un método estocástico que simula un proceso de toma de decisiones aleatorias, permitiendo la creación de caminos o patrones sin una dirección predefinida. Esta técnica es ampliamente utilizada en la generación procedural de entornos en videojuegos, donde cada paso tomado es determinado al azar, pero influenciado por el paso anterior, creando así un entorno dinámico y único en cada iteración. Este método es eficaz para simular la exploración natural en espacios virtuales, ofreciendo una base robusta para desarrollos que requieren de variabilidad y sorpresa en la navegación o la estructura del terreno [24].

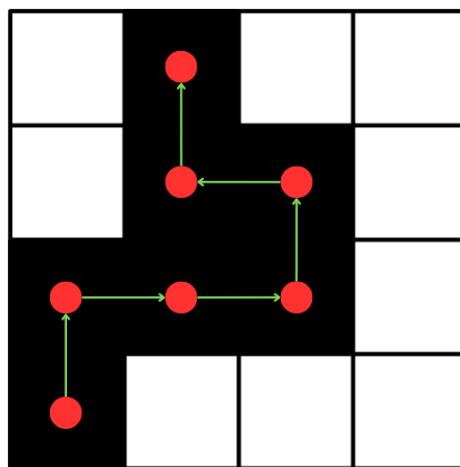


Figura 6.1: Ejemplo de ejecución de algoritmo random walk

6.3.4. Algoritmo Poisson Disk Sampling

El algoritmo de Poisson Disk Sampling se utiliza para el posicionamiento espaciado de objetos dentro de un área definida, asegurando que no haya una congestión visual o funcional en la generación procedural de entornos. Esta técnica es crucial para la simulación de distribuciones naturales, como la disposición de árboles en un bosque o rocas en un paisaje, permitiendo que cada elemento mantenga una distancia mínima con sus vecinos, lo que resulta en una representación más realista y visualmente atractiva del entorno. Este método ha demostrado ser efectivo no solo en videojuegos, sino también en aplicaciones de visualización y modelado de datos [17].

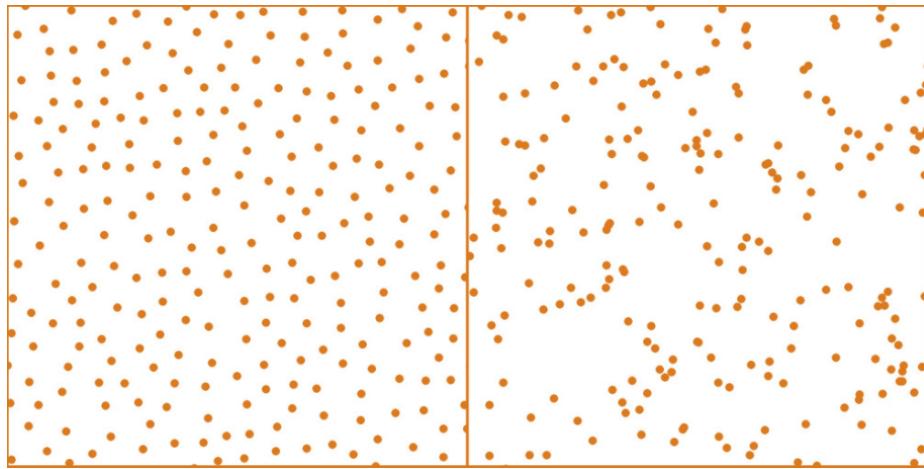


Figura 6.2: Cuadro comparativo Poisson Disk Sampling vs Distribución aleatoria

6.4. Inteligencia Artificial

La Inteligencia Artificial (IA) se refiere a la simulación de procesos de inteligencia humana por parte de sistemas informáticos. Estos procesos incluyen el aprendizaje (la adquisición de información y reglas para usar la información), el razonamiento (usar reglas para llegar a conclusiones aproximadas o definidas) y la autocorrección. En el desarrollo de videojuegos, la IA se utiliza para crear personajes no jugables (NPCs), generar contenido dinámico y adaptar la dificultad del juego en función del rendimiento del jugador [19].

6.4.1. Large Language Models (LLM)

Los *Large Language Models* (Modelos de Lenguaje Grande) son modelos de aprendizaje profundo que se han entrenado con grandes cantidades de datos textuales para comprender y generar lenguaje humano. Estos modelos, como GPT-3 de OpenAI, utilizan redes neuronales profundas con miles de millones de parámetros para procesar y producir texto coherente y contextualmente relevante [4].

6.4.2. Machine Learning (ML)

El aprendizaje automático, conocido en inglés como 'machine learning', se define como un subcampo de la ciencia computacional que se dedica al desarrollo de algoritmos capaces de aprender de los datos y realizar predicciones. Inspirado por la capacidad humana de aprender de la experiencia,

este campo permite a las máquinas mejorar su desempeño en tareas específicas mediante el procesamiento extensivo de datos. Los algoritmos de aprendizaje automático construyen modelos que pueden hacer predicciones o tomar decisiones inteligentes basadas en datos nuevos, ajustando sus parámetros automáticamente con cada nuevo conjunto de datos para optimizar continuamente su rendimiento [25].

6.4.3. Redes Neuronales

Las Redes Neuronales son un tipo de algoritmo de aprendizaje profundo que están inspirados en la estructura y función del cerebro humano. Estas redes se componen de capas de nodos (neuronas) que procesan datos de entrada para detectar patrones complejos. En el contexto de los videojuegos, las redes neuronales pueden utilizarse para el reconocimiento de patrones, la generación de contenido y la toma de decisiones en tiempo real [14].

6.4.4. Procesamiento de Lenguaje Natural (NLP)

El Procesamiento de Lenguaje Natural (NLP) es una rama de la IA que se ocupa de la interacción entre los ordenadores y el lenguaje humano. NLP permite a las máquinas entender, interpretar y responder al lenguaje humano de manera significativa. En los videojuegos, NLP puede ser utilizado para crear interacciones más naturales entre los jugadores y los NPCs, así como para analizar y generar diálogos en el juego [4].

6.4.5. OpenAI

OpenAI es una organización de investigación en inteligencia artificial (IA) que tiene como objetivo promover y desarrollar IA amigable de manera que beneficie a toda la humanidad. Fundada en diciembre de 2015 por Elon Musk, Sam Altman y otros, OpenAI se estableció con la visión de investigar y democratizar el acceso a tecnologías de inteligencia artificial general (AGI) que puedan exhibir comportamientos equivalentes o superiores a los humanos en todas las tareas. A lo largo de los años, OpenAI ha desarrollado una serie de tecnologías de IA avanzadas, y es conocida por sus contribuciones significativas en el campo del aprendizaje profundo, procesamiento de lenguaje natural y robótica [18].

6.4.6. API de OpenAI

La API de OpenAI es una herramienta que permite el acceso a avanzados modelos de lenguaje basados en inteligencia artificial. Desarrollada por OpenAI, esta API facilita la integración de capacidades avanzadas de procesamiento del lenguaje natural en diversas aplicaciones y servicios. Diseñada para ser escalable y accesible, la API permite a los desarrolladores de todos los niveles implementar funcionalidades complejas de IA, como la generación de texto, traducción, resumen y mucho más, con relativa facilidad y sin la necesidad de entrenar modelos desde cero. Este enfoque no solo acelera el desarrollo de aplicaciones innovadoras, sino que también abre nuevas posibilidades en campos como la educación, la atención médica y el entretenimiento, donde la interacción natural y contextual con la tecnología puede mejorar significativamente la experiencia del usuario [18].

La API de OpenAI opera mediante un sistema basado en tokens, en el cual los costos de uso dependen de la cantidad de tokens procesados en cada solicitud. Los tokens representan fragmentos de palabras, y cada interacción —tanto en la entrada del usuario como en la respuesta generada por el modelo— consume un número determinado de tokens. Este sistema permite una estructura de

costos flexibles, en la cual los usuarios pagan únicamente por el uso real de la API, sin cuotas fijas. La cantidad de tokens consumidos puede variar según la complejidad y extensión de la solicitud, y los modelos de lenguaje más avanzados, como gpt-4, suelen tener un costo por token más alto debido a su mayor capacidad y precisión. Este esquema de precios basado en tokens permite a los desarrolladores ajustar y optimizar el uso de la API de acuerdo con las necesidades específicas de cada proyecto, controlando los costos a medida que se incrementa el volumen de interacciones.

6.4.7. Inworld

El framework Inworld está diseñado para la creación de personajes virtuales interactivos impulsados por inteligencia artificial. Este sistema permite desarrollar NPCs (Non-Player Characters) capaces de responder de manera natural y dinámica a las interacciones del jugador, empleando modelos de lenguaje avanzados para generar diálogos contextuales y comportamientos realistas. Su integración con Unity facilita el desarrollo de juegos narrativos e interactivos al ofrecer un SDK optimizado que permite definir personalidades, intenciones y estilos de conversación únicos para cada personaje. Además, Inworld incorpora un sistema de memoria que permite a los NPCs recordar interacciones previas, mejorando la continuidad y profundidad de las conversaciones. Su arquitectura escalable y compatible con múltiples plataformas lo convierte en una opción ideal para proyectos que buscan ofrecer experiencias inmersivas basadas en inteligencia artificial en Unity [2].

6.4.8. Plataforma Hugging Face

Hugging Face es una plataforma que ofrece modelos preentrenados para tareas como generación de texto, traducción, clasificación y análisis de sentimientos, brindando flexibilidad para adaptarse a diversos casos de uso. La integración de Hugging Face en Unity se puede realizar mediante su API, que permite incorporar funcionalidades de inteligencia artificial directamente en los juegos. Su sistema modular facilita la implementación de modelos personalizados, mientras que el soporte para llamadas asíncronas optimiza el rendimiento en dispositivos con recursos limitados. [7].

6.4.9. Azure AI

Azure AI es una suite de servicios de inteligencia artificial desarrollada por Microsoft que proporciona herramientas avanzadas para el procesamiento de lenguaje natural, reconocimiento de voz e imágenes y análisis de datos. Su integración con Unity permite aprovechar servicios escalables y modulares que optimizan la interacción y personalización dentro de los juegos. Con Azure AI, es posible implementar características como reconocimiento de voz, traducción automática, análisis de sentimientos y toma de decisiones basadas en el comportamiento del usuario. Su arquitectura en la nube garantiza escalabilidad y rendimiento para aplicaciones de alto procesamiento, mientras que sus SDKs simplifican la conexión con Unity. [3].

CAPÍTULO 7

Metodología

7.1. Proceso de Diseño del Videojuego

En el desarrollo de un videojuego, es fundamental realizar un énfasis significativo en una fase de diseño bien definida y estructurada. Este proceso abarca desde la concepción de la idea más básica hasta su desarrollo e integración con elementos más complejos. Un enfoque detallado en esta fase no solo garantiza la funcionalidad del videojuego, sino que también asegura que sea visualmente atractivo y capaz de generar un compromiso positivo (engagement) por parte de los jugadores.

7.1.1. Conceptualización

El videojuego se basa en un mundo medieval antiguo y misterioso, diseñado para ofrecer una experiencia de exploración y desafío a los jugadores. Desde el inicio, el enfoque estuvo en crear un entorno que permitiera integrar de manera efectiva mecánicas de juego impulsadas por inteligencia artificial y generación procedural de contenido. Esto asegura que cada partida sea única, ofreciendo nuevas narrativas y desafíos, permitiendo una experiencia personalizada y dinámica.

Estética

En cuanto a la estética, se decidió utilizar un estilo low poly. Este enfoque visual minimalista no solo simplifica el desarrollo gráfico, sino que también mantiene un diseño amigable y coherente con el mundo medieval. La simplicidad de las formas y la optimización del rendimiento permiten que los jugadores se concentren en la exploración y la narrativa, sin comprometer la calidad visual ni la inmersión.

Temática

La temática del juego, centrada en un entorno medieval y misterioso, fue elegida estratégicamente para complementar las capacidades de la inteligencia artificial. Este ambiente no solo favorece la creación de historias dinámicas a través de la generación procedural, sino que también refuerza el

tono aventurero del juego. A medida que los jugadores exploran este mundo cambiante, interactúan con personajes no jugables (NPCs) y resuelven misterios, todo mientras navegan por un entorno en constante evolución.

Estilo

Así mismo para garantizar una experiencia visual y jugable coherente, se optó por un estilo isométrico con vista top-down. Esta perspectiva proporciona una visión clara del entorno, facilitando la navegación y la interacción. Los movimientos simples e intuitivos del personaje, junto con la perspectiva elegida, mejoran la inmersión del jugador y optimizan la experiencia de exploración.

7.1.2. Género

La decisión de definir el juego como un RPG (Role-Playing Game) se tomó tras evaluar las posibilidades que este género ofrece en términos de exploración, historia y desarrollo de personajes. El enfoque en un mundo medieval y misterioso, junto con la integración de inteligencia artificial para generar narrativas y mecánicas procedurales, encajaba de manera natural con las características de un RPG.

7.1.3. Elección de plataforma de Desarrollo

La selección de la plataforma de desarrollo fue un proceso deliberado en el que se consideraron múltiples motores, como Unity, Unreal y Godot. Se realizó un análisis comparativo basado en características técnicas y la experiencia previa del desarrollador, con el objetivo de elegir la mejor opción para las necesidades del proyecto.

Finalmente, se decidió utilizar Unity como motor de desarrollo. Esta elección se basó en la familiaridad del desarrollador con esta herramienta, lo cual se consideró como una oportunidad de desarrollo más eficiente. Además, Unity ofrece una extensa documentación, una comunidad activa y un soporte robusto para la integración de inteligencia artificial y generación procedural, características clave para este proyecto.

7.2. Desarrollo del Videojuego Base

7.2.1. Recolección de Assets

Se realizó una búsqueda general de assets que pudieran utilizarse en el proyecto, con el objetivo de evitar dedicar tiempo a la creación de assets propios. Finalmente, se eligieron los assets de Kenney [12], una fuente reconocida que ofrece una amplia colección de assets gratuitos con licencia abierta. Estos assets se alinean perfectamente con el estilo low poly definido previamente, y al ser todos de una misma fuente, garantizan una coherencia visual en el juego. Además, la licencia abierta permitió integrarlos sin restricciones, optimizando el desarrollo.

7.2.2. Configuración de Unity

En la configuración inicial del proyecto de Unity, se partió de la creación de un nuevo proyecto de tipo 3D, lo que asegura que se cuenten con todas las herramientas y configuraciones adecuadas

para trabajar en un entorno tridimensional. La interfaz inicial de Unity proporciona los elementos básicos para empezar, como una cámara, un objeto de luz y un plano vacío, que sirven de referencia para comenzar a construir el entorno del juego.

Una vez creado el proyecto, es recomendable seguir buenas prácticas de organización del proyecto desde el principio, creando una estructura de carpetas que permita gestionar los diferentes recursos del juego de manera ordenada. En este caso, se crearon las siguientes carpetas básicas:

- AssetKits: En esta carpeta se almacenan los conjuntos de recursos utilizados en el juego, como texturas, materiales, o modelos descargados.
- Models: Aquí se guardan los modelos 3D que se utilizan en el desarrollo del juego. Esto ayuda a separar los modelos de otros recursos y facilita su gestión.
- Scripts: Esta carpeta contiene los archivos de código necesarios para implementar la lógica del juego. Separar los scripts del resto de los recursos ayuda a mantener el proyecto más legible y organizado.

7.2.3. Construcción del Mapa Inicial

En este proyecto, se optó por una perspectiva de juego top-down RPG (Role-Playing Game) con un estilo de cámara isométrico en 3D. Esta elección estética y funcional se fundamenta en la intención de ofrecer a los jugadores una visión clara y estratégica del entorno de juego, facilitando la navegación y la interacción con el mundo virtual.

Para llevar a cabo la construcción del mapa, se utilizó la tecnología Tilemap proporcionada por Unity. Aunque esta herramienta está diseñada principalmente para la creación de mapas 2D, se decidió emplearla en un entorno 3D debido a su flexibilidad y eficiencia en la gestión de grandes superficies de terreno. La integración de Tilemap en un videojuego 3D permite la construcción de escenarios complejos y detallados de manera modular, lo que facilita la personalización y la iteración rápida durante el proceso de desarrollo.

Configuración de Tilemap

La configuración de Tilemap para un entorno 3D implicó varios pasos clave. Primero, se realizó una adaptación de las cuadrículas (grids) tradicionales de 2D a 3D, permitiendo que los tiles se proyectaran en un espacio tridimensional. Esto se logró ajustando las coordenadas y la orientación de los tiles para que se alinearan con la perspectiva isométrica deseada (ver Figura 7.1).

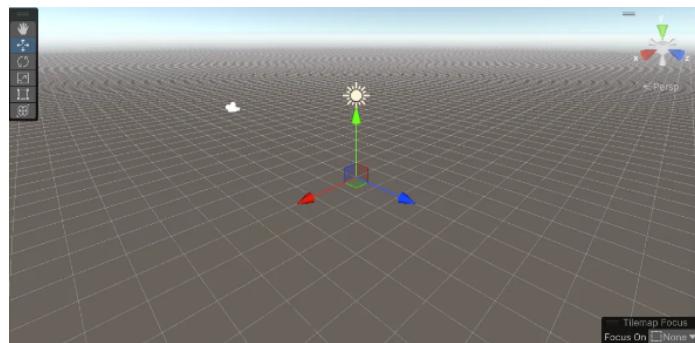


Figura 7.1: Grid establecido en eje Z para crear un plano horizontal.

Posteriormente, se crearon y configuraron varios sets de tiles personalizados que incluían texturas y modelos específicos diseñados para el entorno del juego. Estos tiles se organizaron en capas (layers), lo que permitió un control preciso sobre la disposición de elementos del mapa, como el terreno, los obstáculos, y las estructuras arquitectónicas. La utilización de capas facilitó la superposición de diferentes elementos y la creación de efectos de profundidad y altura, esenciales para mantener la coherencia visual en un entorno 3D.

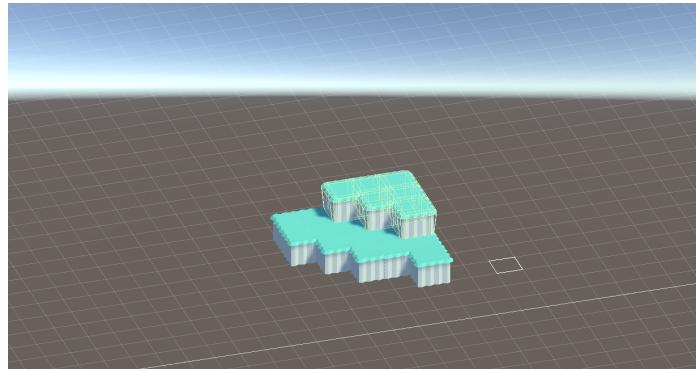


Figura 7.2: Colocación de tiles en grid configurado por layers

7.3. Definición e integración de mecánicas de juego

Dada la naturaleza explorativa y experimental de este proyecto, se tomó la decisión de implementar una mecánica de juego simple, fácil de entender, pero que a su vez fuera atractiva y capaz de capturar la atención de los jugadores. Esta elección estratégica se fundamenta en la necesidad de crear un sistema de juego accesible, que permita a los jugadores familiarizarse rápidamente con las dinámicas del juego, mientras se exploran y experimentan con las tecnologías de inteligencia artificial (IA) que forman parte esencial del proyecto.

Se realizó un análisis exhaustivo de las tendencias actuales en la industria de los videojuegos para identificar mecánicas populares y adecuadas para la integración de IA. Se observó un aumento de popularidad en los RPG, lo que influyó en la selección de mecánicas que combinan simplicidad en el diseño con el potencial de implementar tecnologías avanzadas, facilitando una experimentación más significativa.

7.3.1. Mecánicas de Movimiento

Para la implementación del movimiento del jugador, se utilizó un Character Controller, un componente comúnmente empleado en Unity para gestionar el movimiento de personajes en juegos en 3D. Este componente proporciona funcionalidades esenciales como el manejo de colisiones, la detección del terreno, y la capacidad de mover al personaje en diversas direcciones sin necesidad de utilizar un motor de física completo.

Dado que el estilo del videojuego es isométrico 3D, se realizaron ajustes específicos para adaptarse a esta perspectiva. En particular, se prestó especial atención a la rotación de la cámara, que influye directamente en la orientación y el movimiento del jugador. Para asegurar que el movimiento se sintiera natural y coherente con la vista isométrica, se implementó un sistema de control que alinea los movimientos del jugador con la rotación actual de la cámara. Esto permitió que las entradas de dirección (como avanzar o retroceder) se interpretaran correctamente, independientemente de la rotación de la cámara, ofreciendo una experiencia de juego intuitiva y fluida.

7.3.2. Mecánicas de interacción

El proceso de integración de los LLMs en el juego implicó la implementación de mecánicas de interacción sencillas y funcionales. En primer lugar, se programaron los NPCs con rutinas de acciones predeterminadas que se activan durante su comportamiento normal en el juego. Cuando el jugador se encuentra a una distancia cercana, los NPCs interrumpen su rutina, giran hacia el jugador y activan la posibilidad de interacción. La interacción se activa cuando el jugador presiona la tecla E en el teclado.

Para gestionar esta interacción, se desarrolló un sistema de chat en pantalla que funcionaría como el medio de comunicación principal entre el jugador y los NPCs. El diseño del sistema incluyó un campo de entrada donde el jugador podía escribir mensajes, que luego serían procesados por los LLMs para generar respuestas en tiempo real desde los NPCs. Este sistema de chat, como se ilustra en la Figura 7.3, no solo facilitó la interacción fluida entre el jugador y los personajes del juego, sino que también fue crucial durante el proceso de integración y en las fases de prueba del juego, permitiendo ajustar la respuesta de los NPCs de manera eficiente.



Figura 7.3: Concepto de chat a desplegar al momento de interactuar con un personaje

7.3.3. Mecánicas de recolección

Con el objetivo de crear una mecánica de juego clara y funcional, se diseñó un flujo que permitiera al jugador utilizar las mecánicas de movimiento e interacción previamente implementadas para recolectar objetos distribuidos a lo largo del mapa. Estos objetos fueron definidos como interactuables y se implementaron con la lógica de recolección para proporcionar un reto al jugador, incentivando la exploración de cada rincón del mapa. Además, la recolección se integró con la interacción de los NPCs, quienes ofrecían pistas o información adicional sobre los objetos, lo que añadió una capa de complejidad y motivación a la mecánica de recolección.

7.3.4. Condición de victoria

Finalmente, se definió una condición de victoria estándar que complementara las mecánicas previamente implementadas. Esta consistió en permitir al jugador depositar los objetos recolectados en un recipiente designado. La victoria se alcanzaba al recolectar una cantidad específica de objetos. Es importante mencionar que esta mecánica se estableció como una base, con la intención de expan-

dirla en fases posteriores del desarrollo, donde la explicación y el propósito de la recolección serían proporcionados por los LLMs, añadiendo mayor complejidad y profundidad a la narrativa.

7.4. Investigación de Técnicas de Generación Procedural de Contenido

7.4.1. Estudio de Algoritmos

Durante esta fase, el enfoque estuvo en identificar los algoritmos más adecuados para las necesidades del proyecto, particularmente para la generación procedural de contenido y la interacción con NPCs impulsada por LLMs. Se llevó a cabo una investigación exhaustiva que incluyó fuentes académicas y trabajos de la comunidad de desarrolladores. Se exploraron soluciones utilizadas en proyectos similares, lo que permitió evaluar opciones en función de su rendimiento y compatibilidad con la plataforma seleccionada.

Entre los algoritmos seleccionados se encuentran Random Walk, Poisson Disk Sampling, y técnicas de aleatoriedad. Estos algoritmos fueron elegidos debido a su capacidad para generar estructuras de manera eficiente en un entorno isométrico y proporcionar la variabilidad necesaria en la distribución de objetos y elementos del mundo. Se priorizó su simplicidad y su facilidad de implementación, así como su capacidad para cumplir con los requerimientos de escalabilidad y flexibilidad del proyecto.

7.5. Integración de Técnicas de Generación Procedural de Contenido

Basado en uno de los principales objetivos del proyecto, que consiste en ofrecer una experiencia altamente personalizable para el jugador, se llevó a cabo un análisis sobre la integración de técnicas de generación procedural en el desarrollo del juego. El propósito de esta investigación fue encontrar formas de implementar estas técnicas de manera que se ajustaran a la narrativa y mecánicas del proyecto, buscando no solo crear escenarios dinámicos, sino también asegurar que estos escenarios reforzaran la inmersión y coherencia del mundo de juego.

En esta etapa se buscó identificar momentos específicos dentro de la historia del videojuego donde la generación procedural pudiera no solo aumentar la rejugabilidad, sino también aportar variedad y frescura en cada sesión de juego. Esto implicó no solo crear escenarios aleatorios, sino que dichos entornos generados proceduralmente estuvieran alineados con el tono y los eventos narrativos preestablecidos. De esta manera, los jugadores pueden experimentar entornos únicos que siguen siendo coherentes con la historia central, aportando tanto al diseño visual como a la progresión narrativa.

7.5.1. Selección aleatoria de objetos

Para iniciar, se implementaron técnicas simples de aleatoriedad, esenciales para el desarrollo del videojuego. En esta etapa, se integraron algoritmos sencillos de selección aleatoria con el objetivo de definir un conjunto específico de objetos a partir de un grupo mayor. Estos algoritmos permitieron generar la configuración inicial del juego de manera dinámica, asegurando que cada partida presentara diferentes combinaciones de elementos en el mapa.

Como parte de la mecánica central, se estableció un conjunto global de objetos, del cual se selecciona una cantidad específica de manera aleatoria, dentro de un rango definido en el código. Este rango permite elegir un subconjunto de objetos entre dos límites predefinidos. El resultado es un subconjunto más pequeño de objetos seleccionados, que luego se utiliza para modificar la narrativa del juego, adaptando la experiencia del jugador según las selecciones realizadas y generando variabilidad en cada partida.

7.5.2. Generación con características aleatorias en NPCs

Antes de integrar los sistemas de inteligencia artificial en los NPCs, se prepararon los elementos que definirían su aspecto visual. El objetivo fue asegurar que cada NPC fuera visualmente único y distinto. Para lograr esta variabilidad, se dividieron los personajes en distintas partes, como la cabeza, el cuerpo, las extremidades y otros accesorios. Cada una de estas partes contaba con múltiples variaciones, lo que permitió que el sistema seleccionara aleatoriamente entre las opciones disponibles para generar personajes visualmente distintos.

Con esta estructura definida, se implementó un algoritmo de generación aleatoria que seleccionaba una variación de cada parte del cuerpo. Al combinar diferentes cabezas, cuerpos, brazos y piernas, se garantizaba que los NPCs creados en cada ejecución fueran únicos. Este enfoque permitió enriquecer el entorno visual del juego, haciendo que los NPCs fueran fácilmente distinguibles y contribuyeran a la inmersión del jugador en el mundo del juego.

El código desarrollado utilizaba un catálogo de partes del cuerpo, almacenado en el ScriptableObject, para asignar de forma aleatoria un modelo de cabeza, cuerpo, brazos y piernas a cada NPC al ser instanciado. Para garantizar la coherencia visual, las partes seleccionadas se asignaban a los objetos del modelo 3D del NPC, utilizando el componente SkinnedMeshRenderer de Unity, que permitía aplicar las diferentes mallas a las partes correspondientes del cuerpo. Este proceso no solo permitió generar personajes con apariencias variadas, sino que también mantenía la flexibilidad para modificar las combinaciones fácilmente si se requería una mayor personalización o expansión en el futuro.

7.5.3. Aplicación de materiales y elementos adicionales

Además de las partes del cuerpo, se integró la posibilidad de aplicar un material aleatorio a todas las partes del personaje. Este material, seleccionado de una lista predefinida, era asignado a cada una de las mallas del NPC, asegurando que tanto el color como la textura de la piel, ropa o accesorios fueran consistentes en todo el modelo. De esta manera, no solo se lograba una variabilidad en la estructura física del NPC, sino también en su aspecto superficial, aportando aún más diversidad a los personajes que poblaban el mundo del juego.

Finalmente, el sistema incluyó un proceso para manejar la asignación de elementos adicionales, como capas o accesorios, que también recibían materiales aleatorios para complementar la apariencia del NPC. Con esta metodología, se creó una amplia gama de personajes únicos, sin necesidad de modelar manualmente cada uno de ellos, lo que permitió ahorrar tiempo y recursos durante el desarrollo, mientras que se mantenía un alto nivel de personalización en el juego.

Este enfoque automatizado en la creación de NPCs visualmente diversos, combinando variaciones de partes del cuerpo y materiales, proporcionó una base sólida para la posterior integración de sistemas de inteligencia artificial que permitieran interacciones complejas entre los jugadores y los NPCs. La diversidad visual de los personajes añadía realismo y variedad al entorno, enriqueciendo la experiencia del jugador en cada sesión de juego.

7.5.4. Generación de laberinto

Uno de los escenarios clave definidos en el concepto inicial del videojuego es un laberinto, diseñado para introducir elementos de puzzle en las mecánicas de juego. Para lograr que este laberinto fuera dinámico y variado, se decidió utilizar técnicas de generación procedural, de modo que el escenario se generara de forma diferente en cada partida, aumentando así la rejugabilidad.

El primer paso fue definir un objeto prefabricado (prefab) que funcionaría como un bloque del laberinto dentro de una cuadrícula (grid). Cada bloque tiene un ancho y profundidad predefinidos, y se utiliza para construir la estructura general del laberinto. Para este caso específico, en el que se busca generar un laberinto, el objeto prefabricado fue diseñado con cuatro paredes, donde cada pared es un objeto hijo del bloque principal, representando las barreras del laberinto.

A continuación, se desarrolló un script encargado de generar la estructura completa del laberinto. El script crea una cuadrícula en la que cada punto de la misma representa uno de los bloques prefabricados. Este sistema permite instanciar un número determinado de bloques en los ejes horizontal y vertical, formando la base del laberinto.

Una vez que la cuadrícula está inicializada, comienza la segunda etapa del proceso, en la que se aplica un algoritmo conocido como random walk. Este algoritmo tiene como objetivo recorrer la cuadrícula de forma secuencial, visitando cada bloque de manera aleatoria. Al llegar a un bloque, se registra como "visitado", lo que indica que ya forma parte del camino del laberinto. A medida que el algoritmo avanza hacia un nuevo bloque, se elimina una de las paredes que lo separa del bloque previamente visitado, conectando los bloques entre sí.

Este proceso de eliminación de paredes crea pasillos dentro del laberinto, dando lugar a rutas que no estaban presentes en la cuadrícula inicial. A lo largo de la ejecución del algoritmo, se van generando caminos que pueden ramificarse o incluso conectarse entre sí, formando un diseño intrincado y variable. El resultado final es un laberinto con rutas únicas y caminos inesperados en cada partida, lo que no solo aumenta la dificultad, sino que también asegura que cada experiencia sea diferente, añadiendo un elemento de exploración y sorpresa para el jugador. Al no seguir un patrón fijo, el algoritmo logra que los jugadores se enfrenten a un reto nuevo cada vez que se genera un nuevo laberinto.

7.5.5. Generación de sector orgánico del mapa

Siguiendo con la implementación de generación procedural, se decidió aplicar una técnica similar en un sector específico del mapa, cuyo objetivo era construir una ambientación más natural. Para recrear un entorno boscoso, se utilizó el mismo algoritmo Random Walk previamente implementado en la generación del laberinto, aunque con ciertos ajustes para lograr el resultado deseado en este caso.

Este algoritmo resultó ser útil, en términos generales, para la creación de caminos que dirigían al jugador hacia una única dirección, generando una estructura variable en cada ejecución, lo que añadía un componente de imprevisibilidad al juego. Sin embargo, para este sector específico, se realizó una modificación del algoritmo con el fin de obtener un espacio más orgánico, donde no necesariamente hubiera caminos definidos, sino áreas irregulares por las cuales el jugador pudiera moverse libremente y explorar, simulando de esta forma un entorno natural más auténtico.

El ajuste consistió en modificar la lógica del algoritmo, de manera que, tras la generación de la cuadrícula base en la escena, los bloques del grid se eliminaban sin la necesidad de mantener muros delimitadores entre celdas. Esto permitió la creación de rooms.^º áreas abiertas de forma irregular, imitando más de cerca la topografía de un espacio natural. Al eliminar la estructura de caminos delimitados por paredes, se obtuvo una distribución de terreno que ofrecía una sensación más fluida

y libre, evocando la exploración de un bosque o área sin intervención humana.

Para mantener la coherencia visual del juego, las celdas de este sector fueron diseñadas como bloques de tierra, en línea con los utilizados en otras áreas del mapa (ver Figura 7.4). Esto aseguró que el estilo estético permaneciera consistente en todo el entorno del juego, pero con la particularidad de que este sector presentara un carácter más salvaje y sin explorar, lleno de vegetación.

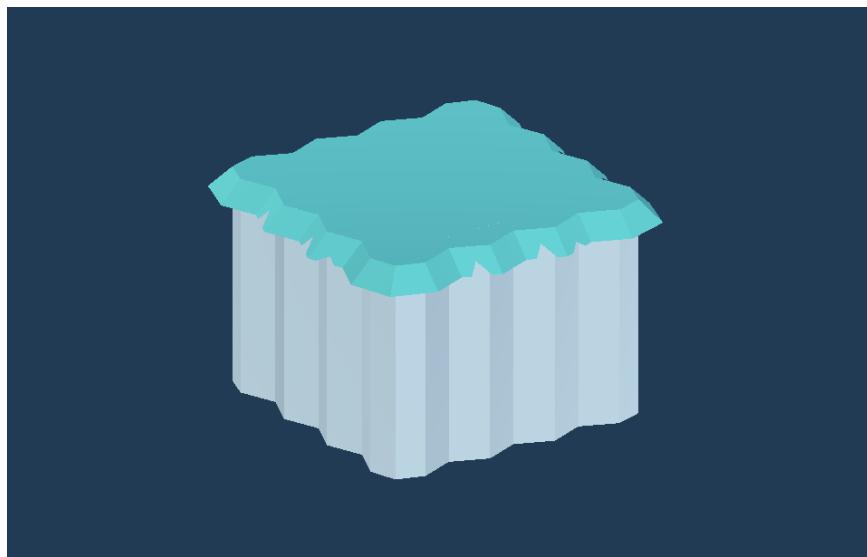


Figura 7.4: Bloque de terreno diseñado para formar secciones orgánicas en el mapa, utilizado como base en la generación procedural.

Un aspecto clave en esta fase del proceso fue la implementación de un mecanismo para almacenar las celdas que habían sido visitadas. Este procedimiento consistió en guardar, en un arreglo temporal, las posiciones de los bloques del grid que se habían recorrido durante la generación. Esta información es fundamental para las etapas posteriores, ya que permite identificar las áreas disponibles para la colocación de elementos contextuales, como árboles o vegetación. Así, se garantiza que los objetos se coloquen en ubicaciones coherentes dentro del espacio jugable, enriqueciendo tanto el aspecto visual como la funcionalidad del entorno.

7.5.6. Colocación aleatoria de objetos en el sector generado

Tras la creación de las áreas irregulares en el sector orgánico del mapa, el siguiente paso fue la colocación de elementos contextuales como árboles, vegetación y otros objetos grandes que reforzaran la ambientación natural del entorno. Este proceso no solo debía garantizar la coherencia visual, sino también respetar una distribución lógica y realista de los objetos dentro del mapa generado.

El primer paso para la colocación de objetos grandes, como árboles o piedras, fue procesar nuevamente las posiciones almacenadas durante la generación del mapa. Estas posiciones correspondían a bloques visitados o eliminados, pero se realizó un filtrado adicional: solo se seleccionaron aquellas posiciones que no tuvieran vecinos adyacentes, es decir, áreas completamente aisladas dentro del grid. De este modo, se evitaba que los árboles u otros elementos grandes se colocaran demasiado cerca de otras áreas no eliminadas, garantizando que estos objetos se distribuyeran en espacios despejados y sin superposición con bloques cercanos.

Una vez identificadas estas posiciones filtradas, el algoritmo procedió a colocar los objetos grandes. Se respetó una distancia mínima entre ellos para asegurar una distribución natural y equilibrada.

Además, se implementó una regla que alternaba entre la colocación de árboles y piedras: después de colocar un número determinado de árboles, se insertaba una piedra en la siguiente posición filtrada, lo que introducía variedad y evitaba la repetición excesiva de un solo tipo de objeto.

Tras esta fase, se continuó con la colocación de objetos más pequeños, como arbustos u otras formas de vegetación. Para estos, se utilizó el conjunto de posiciones donde se habían eliminado bloques, respetando una distancia mínima entre los objetos pequeños para mantener la coherencia visual y evitar superposiciones. Además, se añadieron objetos especiales en ciertas posiciones clave, para dar mayor detalle al entorno y hacer que el sector generado resultara más inmersivo.

Este enfoque permitió que el sector orgánico mantuviera su aspecto natural y dinámico, con una distribución de elementos que no solo reforzaba la ambientación, sino que también añadía desafíos visuales y funcionales a la exploración del jugador. La combinación de un filtrado de posiciones y la generación procedural aseguraba que cada nueva creación del mapa fuera única, proporcionando una experiencia renovada en cada partida.

7.6. Integración de *Large Language Models*

Como bien se ha mencionado uno de los objetivos planteados para este trabajo fue implementar modelos de lenguaje largo (*Large Language Models*) con el objetivo de enriquecer las interacciones con los NPCs del videojuego. La finalidad de esta integración fue dotar a los NPCs de una inteligencia artificial más avanzada, permitiendo que el jugador interactuara con ellos en tiempo real y de una manera más fluida y natural. Esto no solo añade una capa de complejidad a las mecánicas de juego, sino que también permite que las interacciones tengan un impacto directo en el progreso del jugador dentro del juego.

Mediante el uso de estos modelos, los NPCs se convierten en elementos clave para avanzar en la narrativa. El jugador es responsable de llevar a cabo interacciones útiles para obtener información necesaria y relevante para completar misiones o desbloquear nuevas áreas. El objetivo es hacer que la experiencia sea más inmersiva y dinámica, permitiendo que cada conversación con los NPCs pueda proporcionar pistas o detalles que dependerán de la habilidad del jugador para interactuar de forma adecuada.

7.6.1. Configuración inicial de API de OpenAI

Para la integración de los modelos de lenguaje, se utilizó la API de OpenAI debido a que ofrece documentación extensa y actualizaciones frecuentes, lo cual facilitó su implementación directa en el sistema de NPCs. El proceso de configuración comenzó con la creación de un nuevo proyecto en la plataforma de OpenAI, donde se generaron las claves API necesarias para acceder a los servicios del modelo. Dado que el acceso a la API es un servicio de pago, se requirió una cuota mínima de 5 dólares para habilitar su uso. Una vez obtenidas las claves y activado el acceso, se procedió a realizar una integración básica con el motor del juego, empleando el modelo gpt-3.5-turbo, el cual ofrece un equilibrio entre calidad de respuestas y eficiencia, lo cual resulta de gran utilidad para manejar las interacciones en tiempo real entre los jugadores y los NPCs.

La configuración inicial consistió en crear un sistema que permitiera enviar mensajes desde el juego hacia el modelo de lenguaje y recibir respuestas coherentes con la narrativa del videojuego. Para lograr esto, se desarrolló un script en Unity que gestionaba las peticiones y respuestas generadas por el modelo de manera dinámica. El sistema fue diseñado para que los jugadores pudieran interactuar con los NPCs mediante preguntas o diálogos, obteniendo respuestas generadas en tiempo real que se adaptaban a la historia y el contexto del juego.

Este enfoque inicial sentó las bases para la interacción entre los NPCs y los jugadores, permitiendo una experiencia más fluida e inmersiva. A partir de esta integración básica, se estableció un punto de partida que más adelante se refinó y amplió, añadiendo mayor complejidad a las interacciones, como se detallará en las siguientes secciones del proyecto.

7.6.2. Desarrollo de contenido contextual

La integración de *Large Language Models* (LLMs) no solo se limitó a las interacciones con los NPCs, sino que también desempeñó un papel crucial en la generación de contenido contextual para enriquecer la narrativa del juego. A través del uso de la API de OpenAI, se logró crear y adaptar historias, elementos narrativos y objetivos dinámicos, basados en resultados aleatorios generados durante la partida. A continuación, se explican las diferentes integraciones que se realizaron para dar forma a este contenido.

Contexto inicial del mundo

Uno de los primeros usos de los LLMs fue la generación del contexto narrativo que definía el mundo en el que se desarrolla el juego. Se utilizó un prompt que solicitaba una descripción detallada del mundo, incluyendo su historia, mitología y características generales. A partir de esta información, el modelo generó un relato con el fin de ofrecer a los jugadores una inmersión inmediata en el entorno del juego.

Este contexto inicial no solo sirvió para ambientar el juego, sino que también proporcionó un marco narrativo coherente en el que se desarrollan todas las interacciones con los NPCs y otros elementos del juego. La historia generada incluía detalles sobre la geografía del mundo, sus habitantes, y los misterios que el jugador podría descubrir a lo largo de su aventura. Esta narrativa, si bien fue generada por el modelo, podía ser adaptada o ampliada en futuras sesiones de juego, permitiendo flexibilidad y variabilidad según las decisiones tomadas por el jugador.

Historia del enemigo principal

Otra de las integraciones destacadas de los LLMs fue la generación de la historia del enemigo principal del juego. A partir de un prompt específico, se solicitaron fragmentos de una historia con características oscuras que detallaran el trasfondo y la naturaleza de este enemigo. Esta técnica no solo ofreció al jugador una historia intrigante que descubrir, sino que también permitió tener flexibilidad en la forma en que esta historia se desplegaría a lo largo del juego, haciendo que el enemigo se sintiera más dinámico y misterioso.

Contextualización de objetivos generados con generación procedural

Uno de los aspectos clave en el uso de los LLMs fue la generación de recetas de poción, que jugaron un papel central en las mecánicas de victoria o derrota del juego. Este proceso comenzó con la generación procedural de ingredientes, sistema el cual fue explicado anteriormente en esta metodología. Estos ingredientes formaron la base para las recetas de las poción, las cuales los jugadores deben reunir para avanzar en el juego.

Una vez determinados los ingredientes, se generó una receta completa utilizando el modelo de lenguaje, que ofrecía un contexto narrativo coherente con el mundo del juego. Estas recetas no solo describen los ingredientes, sino que también agregan detalles que ayudaban al jugador a entender

cómo y dónde encontrar los elementos necesarios, integrando así la exploración del entorno con la recolección de los ingredientes.

7.6.3. Integración de NPCs Dinámicos

En esta sección se describe el proceso llevado a cabo para integrar modelos de lenguaje a gran escala (LLMs) en los NPCs del videojuego, con el fin de hacer que sus interacciones sean más dinámicas e inmersivas. Esta integración permitió dotar a los NPCs de la capacidad de responder de manera contextual y adaptativa, basándose en los diálogos del jugador y en las interacciones dentro del juego. Utilizando la API de OpenAI, se logró crear NPCs que reaccionan de forma coherente y en tiempo real, lo que proporcionó una experiencia de juego más enriquecedora y flexible.

Preparación y generación de atributos

El primer paso para la integración de los NPCs dinámicos fue la generación de sus atributos principales, tales como el nombre, el rol y la personalidad. Para esto, se utilizó un sistema que generaba estos atributos mediante prompts enviados a la API de OpenAI. A partir de estos prompts, se solicitaban respuestas que incluían los valores para los campos mencionados, los cuales se asignaban de forma automática a cada NPC. Este proceso fue fundamental para crear personajes únicos y diferenciados, con los que el jugador pudiera interactuar de maneras que fueran coherentes con la historia y el ambiente del juego.

Cada NPC fue instanciado con atributos personalizados, lo que permitía una diferenciación no solo visual, sino también en términos de comportamiento. Esta personalización no solo enriqueció la narrativa, sino que también hizo que los NPCs fueran más interesantes y relevantes dentro del mundo del juego, ya que cada uno tenía su propio trasfondo y personalidad, lo que afectaba la forma en que respondían a las preguntas del jugador.

Inicialización del cerebro de los NPCs

Una vez que los NPCs tenían sus atributos básicos asignados, se procedió a inicializar lo que se llamó como su "cerebro". Este cerebro virtual era responsable de gestionar las respuestas que los NPCs daban al jugador. A través de la integración con la API de OpenAI, se enviaban instrucciones a cada NPC con la información relevante del mundo del juego, como detalles sobre el lugar en el que vivían o los personajes que conocían. Estas instrucciones iniciales les permitían responder de manera adecuada a las preguntas del jugador y desempeñar el rol que se les había asignado.

El proceso de inicialización del cerebro incluía la entrega de un prompt inicial que contenía toda la información relevante sobre el entorno y los otros personajes con los que el NPC había interactuado. De esta manera, cada NPC recibía una visión general del mundo del juego y de su propio papel dentro de él, lo que les permitía responder de forma coherente a cualquier interacción que surgiera. Este enfoque dinámico fue esencial para darles a los NPCs una sensación de autonomía dentro del juego.

Interacción dinámica en tiempo real

La integración de los modelos de lenguaje permitió que las interacciones con los NPCs ocurrieran en tiempo real, lo que significa que el jugador podía entablar conversaciones con ellos en cualquier momento, y las respuestas se generaran al instante. Para lograr esto, se diseñó un sistema que permitía que los NPCs recibieran preguntas o comentarios del jugador y enviaran dichas preguntas

a la API de OpenAI. A partir de ahí, la respuesta del modelo era procesada y mostrada en el juego como si el NPC hubiera formulado una respuesta por sí mismo.

Este proceso de interacción dinámica se llevó a cabo mediante una interfaz de chat sencilla, que permitía al jugador escribir preguntas o comentarios. Las respuestas de los NPCs se basaban en la información proporcionada inicialmente y en el contexto de la conversación. Así, los NPCs podían ofrecer consejos, pistas o simplemente interactuar de forma casual, enriqueciendo la experiencia del jugador sin la necesidad de predefinir todos los diálogos.

Adaptación a la narrativa del juego

Uno de los aspectos clave en la integración de los NPCs dinámicos fue asegurar que las respuestas generadas por el modelo se adaptaran a la narrativa del juego. Para esto, se desarrollaron una serie de reglas y limitaciones que guiaban el comportamiento de los NPCs, evitando respuestas que fueran incoherentes con la historia o el mundo en el que se desenvolvían. Por ejemplo, si el jugador pregunta algo que no está alineado con el rol del NPC, este puede evitar la pregunta o responder que no sabe, lo que pretende mantener la consistencia dentro del juego.

Asimismo, se utilizaron prompts específicos para garantizar que los NPCs brindaran pistas o sugerencias relevantes para avanzar en el juego, sin hacer que las respuestas fueran demasiado explícitas. Esto permitió que el jugador tuviera un papel activo en la resolución de problemas y descubrimiento de secretos, basándose en su habilidad para interactuar de manera efectiva con los

7.7. Pruebas de Validación

Para evaluar el prototipo desarrollado, se llevó a cabo una sesión de playtesting en la que participaron usuarios externos al proyecto, quienes probaron las distintas mecánicas y el funcionamiento general de un primer prototipo del videojuego. Este proceso resultó invaluable, ya que permitió obtener retroalimentación directa de usuarios que interactuaban por primera vez con el juego.

Con base en los comentarios y recomendaciones recibidos, se identificaron diversas deficiencias y errores en el desarrollo del videojuego. Posteriormente, se realizó un análisis detallado de cada aspecto señalado, descartando o priorizando aquellos que resultaban más relevantes para alcanzar los objetivos del proyecto.



Figura 7.5: Sesión de juego

Finalmente, se regresó a una fase de desarrollo para implementar correcciones basadas en la retroalimentación recibida de los usuarios. Se puso especial énfasis en abordar las problemáticas identificadas, asegurando que las mejoras estuvieran alineadas con los objetivos del proyecto y respondieran de manera efectiva a las necesidades de los jugadores.

7.7.1. Validación de interacción con NPCs

Para validar la interacción con los NPCs, se destinaron momentos específicos durante las sesiones de juego en los que los usuarios, tras haber probado el prototipo y experimentado las interacciones con los personajes, completaron encuestas para registrar sus impresiones. Este proceso permitió obtener una resultados cuantitativos para evaluar si los objetivos relacionados con la integración de los modelos de lenguaje (LLMs) en el videojuego fueron alcanzados. A través de estas encuestas, se recogieron opiniones sobre la coherencia, naturalidad y utilidad de las respuestas generadas por los NPCs, permitiendo analizar de manera estructurada la calidad de la interacción.

Preguntas
<ul style="list-style-type: none"> ■ En una escala del 1 al 5, ¿cómo consideras que las respuestas de los NPCs fueron coherentes con el contexto del juego? ■ En una escala del 1 al 5, ¿cómo crees que las respuestas de los NPCs enriquecieron la historia del juego? ■ En una escala del 1 al 5, ¿cómo calificarías la naturalidad de las interacciones con los NPCs? ■ Después de jugar dos veces, ¿notaste diferencias en las interacciones con los NPCs? ■ ¿Las respuestas generadas por los NPCs te ayudaron a avanzar en el juego o resolver problemas?

Tabla 7.1: Preguntas de validación para la interacción con NPCs

7.7.2. Validación del atractivo Visual

Otro aspecto fundamental planteado en el desarrollo del proyecto fue la creación de un producto visualmente impactante y atractivo para los jugadores. Al igual que en la validación de las interacciones con los NPCs, se utilizó una encuesta para recolectar las impresiones de los usuarios, pero en esta ocasión, centrada en el prototipo visual del videojuego. Este proceso permitió medir de manera estructurada si el estilo gráfico y la calidad visual del juego alcanzaban los objetivos propuestos, proporcionando un punto de referencia clave para evaluar la aprobación y el impacto visual del juego entre los jugadores.

Preguntas
<ul style="list-style-type: none"> ■ En una escala del 1 al 5, ¿cómo calificarías la calidad visual del juego, donde 1 es muy baja y 5 es excelente? ■ En una escala del 1 al 5, ¿cómo evaluarías el atractivo del estilo visual del juego, siendo 1 no atractivo y 5 muy atractivo? ■ ¿Consideras que los gráficos y el arte del juego cumplen con los estándares de calidad de la industria? ■ En una escala del 1 al 5, ¿en qué medida te ayudó el diseño visual del juego a mantener tu interés en la experiencia de juego, donde 1 es nada y 5 es mucho?

Tabla 7.2: Preguntas de validación para el atractivo visual del juego.

7.7.3. Validación de mecánicas

Finalmente, para cumplir con todos los objetivos del proyecto, se aplicó una última encuesta a los jugadores, enfocada en las mecánicas del juego. El propósito de esta encuesta fue recopilar impresiones sobre si las mecánicas ofrecieron una experiencia de juego completa, manteniendo a los usuarios comprometidos e inmersos en la partida. De esta manera, se buscó validar si el proyecto logró proporcionar una jugabilidad atractiva y coherente con las expectativas iniciales.

Preguntas
<ul style="list-style-type: none"> ■ En una escala del 1 al 5, ¿qué tan involucrado te sentiste mientras jugabas, donde 1 es nada comprometido y 5 es muy comprometido? ■ En una escala del 1 al 5, ¿cómo evaluarías la variedad de las mecánicas de juego en términos de mantener tu interés durante la sesión, donde 1 es nada variadas y 5 es muy variadas? ■ En una escala del 1 al 5, ¿qué tan intuitivas encontraste las mecánicas del juego, donde 1 es nada intuitivas y 5 es muy intuitivas? ¿Sentiste que fue fácil aprender a jugar? ■ ¿Hubo momentos en el juego donde las mecánicas te hicieron sentir realmente inmerso en la experiencia? ¿Puedes describir esos momentos? ■ ¿Qué aspecto del juego (puzzles, combate, exploración, etc.) te pareció el más entretenido y por qué?

Tabla 7.3: Preguntas de validación para las mecánicas del juego.

7.8. Costos y consumo de API

Como paso adicional de validación, se destinó un espacio para recopilar los costos asociados al desarrollo de este proyecto, centrándose específicamente en el uso de los servicios de OpenAI a través de su API. Para ello, se utilizó el panel de control proporcionado por la plataforma de OpenAI, el cual permite monitorear las estadísticas de uso y los costos asociados en tiempo real. Esto permitió obtener una visión más clara de las implicaciones financieras del proyecto y evaluar su viabilidad para futuras implementaciones.

Para llevar a cabo un análisis detallado del consumo en diferentes contextos, se seleccionaron dos períodos de observación. En primer lugar, se utilizó el mes de octubre, que fue el período de mayor uso debido a que en ese mes se realizaron todas las pruebas intensivas con usuarios. Esto permitió capturar el comportamiento de consumo y los costos asociados bajo condiciones de uso intensivo, simulando un escenario más cercano a un entorno de producción.

Por otro lado, se aprovechó el mes de noviembre para registrar el consumo de una única sesión de juego. Durante ese mes, no se realizaron otras pruebas ni se utilizó el servicio de OpenAI, salvo para una sesión controlada. Esto permitió que el dashboard y las analíticas de OpenAI reflejaran exclusivamente los datos de esta sesión, proporcionando una visión precisa del consumo de tokens y los costos asociados a una interacción típica en tiempo real entre el jugador y los NPCs. Este enfoque permitió contrastar los costos y el consumo entre un período de uso intensivo y una sesión individual, facilitando una evaluación más completa de los recursos necesarios para el proyecto.

CAPÍTULO 8

Resultados

En esta sección, se presentan los resultados concretos obtenidos tras la culminación del proyecto, mostrando que se alcanzó el objetivo de desarrollar un videojuego 3D que integra modelos de lenguaje grande y técnicas de generación procedural de contenido para brindar una experiencia personalizada e inmersiva. Los datos recopilados reflejan cómo se materializaron los objetivos propuestos en el diseño y funcionamiento del juego, evidenciando la incorporación efectiva de tecnologías avanzadas para enriquecer la interacción del usuario. Los resultados destacan el logro de un entorno de juego dinámico y estéticamente atractivo, que respalda la promesa de ofrecer experiencias de juego únicas y personalizadas a cada jugador, cumpliendo así con todos los objetivos planteados.

8.1. Escenario base para fase inicial del videojuego

El desarrollo del escenario base cumplió con el propósito de proporcionar un punto de partida sólido para la narrativa y las mecánicas de juego. Como se puede observar en la Figura 8.1, el entorno construido incluye elementos característicos de una villa, con edificaciones, vegetación y objetos que ofrecen un ambiente coherente con la temática del juego. Este escenario neutral no solo establece el contexto visual, sino que también sirve como base para las mecánicas de exploración iniciales, permitiendo al jugador familiarizarse con el mundo.



Figura 8.1: Escenario base utilizado como punto inicial del videojuego.

8.2. Gameplay inicial

La implementación del gameplay inicial se centró en desarrollar las mecánicas básicas que sustentan la experiencia de juego, tal como se muestra en la Figura 8.2. La creación de un flujo completo de juego, desde la exploración hasta la validación de las condiciones de victoria, permitió una experiencia clara y funcional para el jugador. Estas mecánicas, aunque sencillas en su primera iteración, forman la estructura principal que permitirá expandir el juego en futuras fases.



Figura 8.2: Vista principal del juego en ejecución.

8.3. Generación procedural de contenido

La integración de técnicas de generación procedural en el proyecto contribuyó a garantizar una experiencia de juego dinámica y única para cada usuario. El uso de la generación procedural, tanto en los entornos como en los NPCs, cumplió con el objetivo de ofrecer una experiencia personalizada, lo que añade valor a la rejugabilidad del videojuego.

8.3.1. Generación de área orgánica

Para la creación de un entorno más natural, se adaptó el algoritmo random walk para generar un área orgánica que simula un bosque. La Figura 8.3 y 8.4 muestran la variabilidad obtenida en dos sesiones diferentes, lo que subraya el éxito de este enfoque para generar áreas no uniformes y con características naturales.



Figura 8.3: Área generada en primera ejecución.



Figura 8.4: Área generada en segunda ejecución.

8.3.2. Generación de laberinto

Otra de las aplicaciones tempranas de la generación procedural en el proyecto involucró la creación de un laberinto utilizando nuevamente el algoritmo de random walk. Este enfoque aseguró la producción de un recorrido único en cada sesión de juego. Las Figuras 8.5 y 8.6 ilustran claramente este proceso: la Figura 8.5 muestra la configuración inicial del laberinto, compuesta por una cuadrícula vacía preparada para la generación de caminos. A medida que avanza el algoritmo, se van delineando los caminos hasta completar el laberinto, como se observa en la Figura 8.6. Esta secuencia destaca cómo, desde una estructura inicial simple, el sistema procede a desarrollar un laberinto completo y complejo.

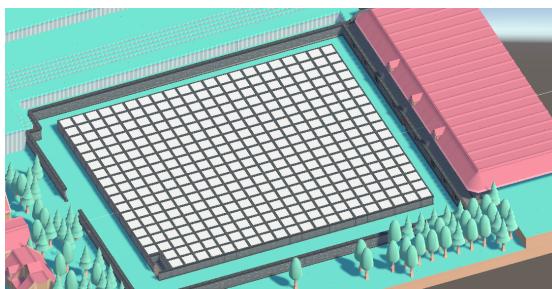


Figura 8.5: Configuración inicial de laberinto

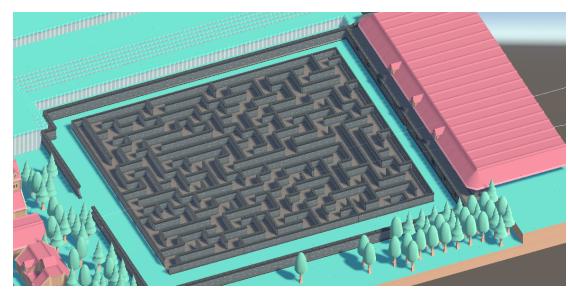


Figura 8.6: Generación de laberinto terminada

8.3.3. Posicionamiento de objetos ornamentales

La Figura 8.7 ilustra la aplicación efectiva del algoritmo de Poisson Disk Sampling en la distribución espacial de elementos ornamentales, como árboles, rocas y vegetación, dentro del entorno de juego. Este algoritmo facilitó la disposición de estos objetos de manera coherente y estéticamente agradable, manteniendo una adecuada separación entre ellos para evitar superposiciones y preservar la naturalidad del paisaje. El resultado es un ambiente visualmente armonioso que refuerza la inmersión del jugador en el mundo del juego.



Figura 8.7: Ejemplo de posicionamiento procedural de objetos ornamentales, como árboles, rocas y vegetación, utilizando algoritmo de Poisson Disk Sampling.

8.3.4. Generación de NPCs con características distintas

La generación procedural no solo se limitó a los entornos, sino que también se aplicó a los NPCs del videojuego. Como se muestra en la Figura 8.8, los NPCs fueron creados con variaciones visuales, lo que permitió que cada personaje no jugable tuviera características únicas.



Figura 8.8: Generación de NPCs con características visuales distintas

8.3.5. Distribución de elementos objetivo

El posicionamiento de los objetos objetivo, necesarios para completar las misiones del juego, se logró mediante la reutilización de los algoritmos de posicionamiento descritos anteriormente. Como se observa en la Figura 8.9, los elementos fueron distribuidos de manera coherente con el entorno, asegurando que cada uno tuviera un propósito claro en las mecánicas del juego.



Figura 8.9: Ejemplo de objeto requerido posicionado aleatoriamente en un área del mapa

8.4. Generación de contenido con LLMs

La integración de Inteligencia Artificial se utilizó para generar historias y descripciones que contextualizan el mundo del juego. Como se muestra en el Cuadro 8.1 ,8.2 y 8.3, los resultados incluyeron detalles sobre la historia y mitología del mundo, así como acontecimientos relevantes que impactan la narrativa del juego los cuales posteriormente fueron consumidos por los distintos NPCs del mapa y asegurar así una mejor interacción.

Prueba	Texto Generado
1	<p>En la época medieval, en medio de un oscuro y tormentoso invierno, un poderoso hechicero llamado Eldrion realizó un conjuro en lo más profundo del bosque cercano a las colinas de Valle Sereno. Su intento por controlar la fuerza de la naturaleza y garantizar la prosperidad de sus descendientes resultó en la aparición de un mágico laberinto subterráneo que se extendía por debajo de las colinas, conectando misteriosamente el corazón del bosque con el pueblo recién creado. Las criaturas míticas y elementales, convocadas por la magia desatada, se unieron a los primeros colonos que llegaron al valle, contribuyendo con sus habilidades sobrenaturales en la construcción de las primeras estructuras. Los guardianes de la naturaleza, como los centauros y los espíritus de los árboles, protegieron el asentamiento de cualquier amenaza externa. A lo largo de las generaciones, los habitantes de Valle Sereno aprendieron a convivir en armonía con la magia que impregnaba el lugar, celebrando festivales de luz y sombra para honrar la unión entre lo mundano y lo sobrenatural. Cada ciclo de vida en el valle se caracterizaba por la manifestación de dones mágicos únicos en sus habitantes, legados de Eldrion y su hechicería ancestral.</p>
2	<p>Hace siglos, en la época medieval, un poderoso mago conocido como Astorius convocó a las criaturas mágicas de la tierra y el aire para crear Valle Sereno. Con su bastón mágico, hizo surgir colinas y bosques encantados que rodeaban la tierra elegida, donde la vida florecía en armonía con la magia. Las criaturas míticas poblaron los bosques, desde elfos hasta unicornios, creando un equilibrio único entre lo natural y lo sobrenatural. A lo largo de los años, los habitantes humanos se unieron a este lugar, atraídos por la paz y la belleza que solo Valle Sereno podía ofrecer. Juntos construyeron casas de colores brillantes y formas inusuales, influenciadas por la magia que fluía a través de la tierra. Las murallas protectoras eran tan imponentes que parecían conjurar una fuerza invisible que garantizaba la seguridad de los lugareños. Más allá de los bosques encantados, se decía que existía un laberinto construido por los mismos dioses, donde los valientes podían encontrar tesoros incomparables o perderse para siempre en sus pasillos retorcidos. Así, Valle Sereno se convirtió en un refugio para aquellos que buscaban un mundo donde la magia y lo extraordinario se fusionaban con la vida cotidiana, creando un lugar de leyenda que trascendía las barreras de la realidad.</p>

Tabla 8.1: Validación de la variación en las historias generadas sobre el origen del pueblo.

Prueba	Texto Generado
1	Se dice que en las noches de luna llena, un ave gigante de plumaje dorado y ojos brillantes sobrevuela el bosque de Valle Sereno, cantando melodías que llenan de paz a quienes la escuchan. Se cree que este ser mágico, conocido como el Fénix de la Luna, es el guardián de los bosques y trae consigo buena fortuna a aquellos que lo ven. También se rumorea que en lo más profundo del laberinto que se encuentra en las colinas cercanas, existe una puerta secreta que conduce a un mundo desconocido lleno de tesoros antiguos y criaturas de proporciones colosales. Solo aquellos valientes y sabios que logren resolver los enigmas del laberinto podrán acceder a este reino misterioso y descubrir sus secretos ocultos.
2	En Valle Sereno se cuenta la leyenda del Dragón de Cristal, una criatura majestuosa con escamas transparentes que reflejan los colores del arcoíris. Se dice que el dragón habitaba en las profundidades del bosque, protegiendo un tesoro de conocimiento antiguo y poderes mágicos. Los habitantes de Valle Sereno, temerosos pero fascinados por la presencia del dragón, le llevaban ofrendas de cristales y gemas en un intento por aplacar su furia. Sin embargo, solo aquellos dignos de su confianza podrían acceder al dragón y descubrir los secretos que guardaba en su guarida en lo más profundo del bosque.

Tabla 8.2: Leyendas generadas alrededor del pueblo de Valle Sereno.

Prueba	Texto Generado
1	En los albores de Valle Sereno, se cuenta la historia de la maga Elenara, quien se dice que tenía el poder de controlar los elementos con solo un gesto de su mano. Condujo al pueblo en una época de prosperidad y armonía con la naturaleza, creando la fuente central que aún adorna la plaza. Su legado se conserva en los susurros del viento y en los arco iris que aparecen después de las tormentas. Otro personaje emblemático es el enigmático caballero Sir Galahad, quien defendió a Valle Sereno de una invasión de sombras malévolas que emergieron del laberinto en tiempos antiguos. Su armadura reluce como estrellas nocturnas en la memoria de los habitantes, y se dice que vaga eternamente por los bosques en busca de justicia y redención. Al frente de los asuntos comerciales estaba la astuta mercader Celia Acuña, quien transformó la economía del pueblo al descubrir una ruta comercial secreta a través del laberinto que le permitió comerciar con reinos desconocidos y productos exóticos. Su legado se refleja en la vitalidad económica de Valle Sereno y en el recuerdo de una audacia que desafía los límites del laberinto.
2	En Valle Sereno existe una leyenda sobre la figura de Elionar, un mago errante que recorría los caminos del pueblo hace siglos. Se dice que Elionar era capaz de controlar los elementos con solo un pensamiento, y que su sombra tenía vida propia, protegiéndolo de cualquier peligro. Se cuenta que Elionar desapareció misteriosamente tras enfrentarse a una criatura de pesadillas que habitaba en lo más profundo del bosque. Su legado sigue vivo en Valle Sereno, donde los aldeanos sostienen que su espíritu vaga por los árboles durante las noches de luna llena, protegiendo a los habitantes del mal y guiando a los viajeros perdidos de regreso a sus hogares.

Tabla 8.3: Personajes en la historia de Valle Sereno.

Prueba	Texto Generado
1	En las sombras del bosque de Valle Sereno, acecha El Devorador de Sueños, un ser antiguo y malévolos que se alimenta de los sueños de los habitantes del pueblo. Su presencia ha sumido a la comunidad en un constante estado de pesadilla y desasosiego, provocando insomnio, ansiedad y paranoia entre la población. Los aldeanos, temerosos de perder sus sueños y su cordura, han visto disminuir su productividad y vitalidad diaria, afectando gravemente la economía y la cohesión social de Valle Sereno. El Devorador de Sueños, con su aspecto de sombra oscura y ojos brillantes como brasas, es descrito en las leyendas como un ser sin rostro que se desplaza silenciosamente entre los árboles, susurra temores en las noches y se alimenta de las esperanzas de los inocentes. A medida que su influencia se extiende, los habitantes de Valle Sereno deben unirse para enfrentar esta oscura amenaza y encontrar una forma de liberarse de la opresión del Devorador de Sueños.
2	Valle Sereno se enfrenta al terrorífico Lord de las Sombras, un ser maligno cuya presencia oscurece el cielo y hace que las noches sean eternas en el pueblo. Se dice que su mirada paraliza a quienes se atreven a cruzar su camino y que su aliento gélido despierta el miedo en los corazones de los habitantes. Lord de las Sombras ha causado que muchos aldeanos desaparezcan misteriosamente en la oscuridad de la noche, sembrando el terror y la desconfianza entre la población. Su malvada influencia ha debilitado la economía local al hacer huir a los comerciantes y ha dividido a los líderes del pueblo en diferentes facciones que buscan formas de enfrentar esta oscura amenaza.

Tabla 8.4: Enemigos en la historia de Valle Sereno.

8.5. Interacción con NPCs alimentados por LLMs

La integración de LLMs para alimentar las interacciones con los NPCs se definió como uno de los objetivos más importantes del proyecto. Como se aprecia en las Figuras 8.10 y 8.11, los NPCs fueron capaces de interactuar en tiempo real con el jugador, ofreciendo respuestas coherentes y contextuales que se alineaban con la narrativa del juego.

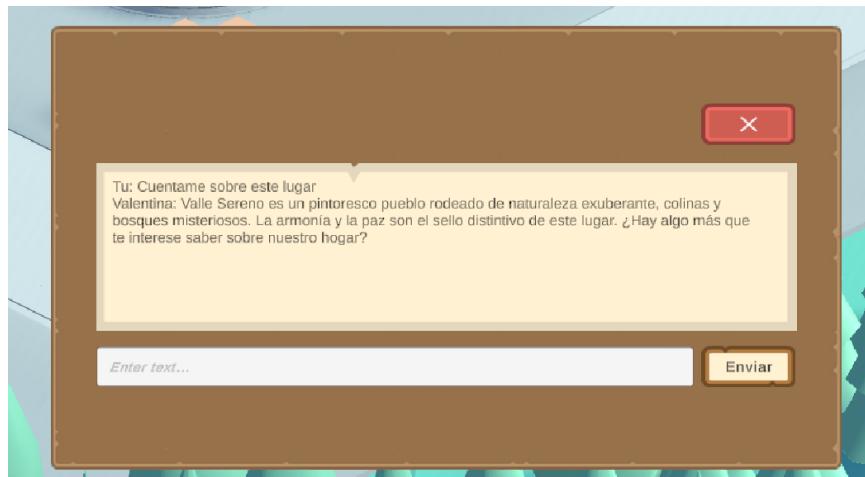


Figura 8.10: Ejemplo de interacción con un NPC para la narrativa contextual del mundo.

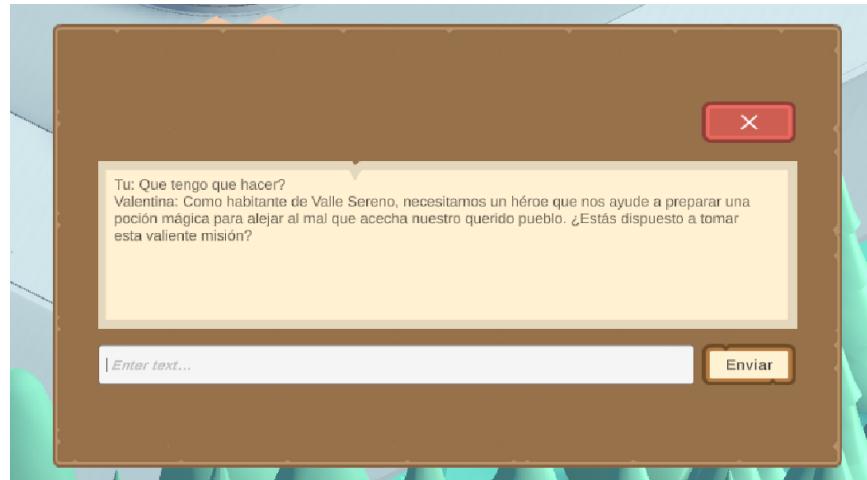


Figura 8.11: Ejemplo de interacción con un NPC para solicitar indicaciones.

8.6. UI

La interfaz de usuario (UI) desempeñó un papel fundamental en la creación de una experiencia de juego amigable y atractiva, asegurando que los jugadores pudieran interactuar de manera intuitiva con el entorno y recibir información clave de manera clara. Como se muestra en la Figura 8.12, se buscó mantener una coherencia visual entre los elementos de la UI y el mundo en el que se desarrolla la narrativa, logrando una integración fluida que refuerza la inmersión del jugador.

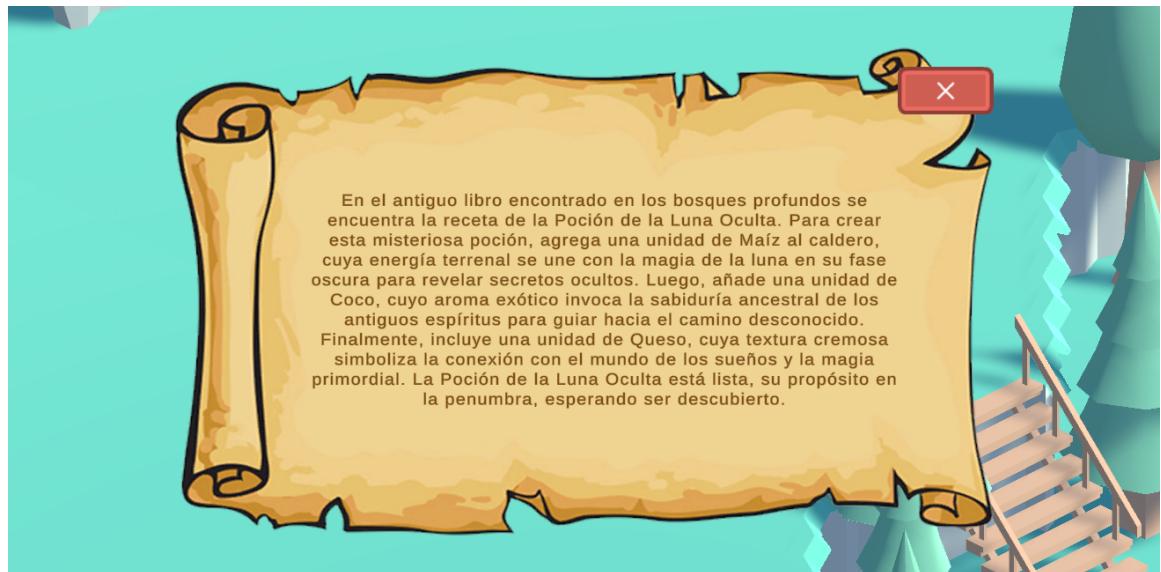


Figura 8.12: Ejemplo de generación procedural de la receta necesaria para obtener la victoria en el videojuego.

8.7. Pruebas de jugabilidad

Las pruebas de juego fueron uno de los principales métodos para validar los resultados y verificar el cumplimiento de los objetivos propuestos. La Tabla 8.5 muestra un resumen de las primeras impresiones de los usuarios, capturando sus reacciones tras una sesión de juego en la que se evaluaron las etapas iniciales de integración de los Modelos de Lenguaje Grande (LLMs) en los NPCs y las mecánicas de juego.

Participante	Retroalimentación
Usuario 1	El diálogo con los personajes y el mapa se siente bastante nutrido. Me gusta que hayas utilizado AI para las conversaciones. La animación de la espada y el escudo se sienten algo fuera de tiempo.
Usuario 2	Todo se siente bien, aunque la interacción con personajes o ítems se siente brusca ya que no se registra desde todos los ángulos. También hay un par de bugs al hablar con los personajes, como el mensaje “adelante con la pregunta del jugador”.
Usuario 3	Me gusta el juego. Sin embargo, habría que solucionar que no aparezca el historial de movimientos en la caja de diálogo. Además, es difícil interactuar con personajes u objetos, ya que cuesta que aparezca la “E” al acercarse. También recomendaría hacer coincidir la velocidad de movimiento con el movimiento de las piernas para evitar que parezca que el personaje flota.

Tabla 8.5: Comentarios recibidos en las sesiones de prueba.

Luego de completar el proceso de corrección y desarrollo basado en los comentarios de los usuarios, se llevó a cabo una segunda ronda de pruebas para evaluar la versión final del proyecto. Estos resultados tienen como objetivo validar los avances alcanzados, enfocándose en las integraciones definidas: atractivo visual, interacción con NPCs y generación procedural. Para obtener una métrica cuantitativa, después de las sesiones individuales, se utilizó la encuesta previamente diseñada con preguntas específicas que permitieron medir de manera porcentual la aceptación por parte de los usuarios.

Esta encuesta se aplicó a un total de nueve personas, una muestra seleccionada que permitió obtener resultados representativos para validar los cambios implementados. La elección de esta cantidad se debió a que las sesiones eran presenciales, extensas y personalizadas, pues cada jugador debía realizar dos pruebas de juego, lo que facilitó un análisis más profundo de sus impresiones y una evaluación detallada de la experiencia de usuario.

8.7.1. Validación de atractivo visual

El atractivo visual del videojuego recibió una evaluación positiva por parte de los usuarios. Como se muestra en la Figura 8.13, el 47.8 % de los jugadores calificó el diseño visual con la puntuación más alta (5), el 34.8 % lo evaluó con una puntuación de 4 y finalmente un 17.4 % con una puntuación media de 3. Ningún jugador asignó una calificación por debajo de 3, lo que refleja una percepción mayoritariamente favorable del atractivo visual del juego.

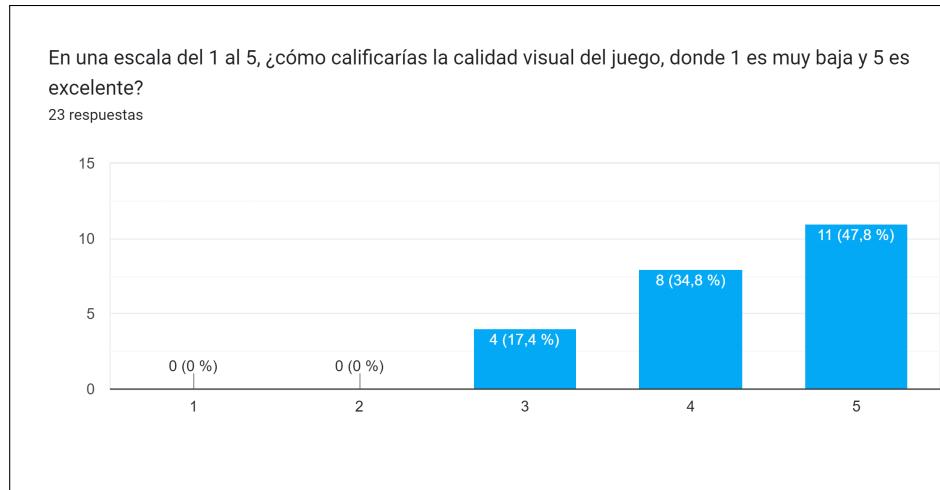


Figura 8.13: Gráfico de barras que muestra la percepción de los jugadores sobre el atractivo visual del juego.

Por otro lado, una validación clave fue la relación entre la calidad visual y el nivel de involucramiento de los jugadores, como se observa en la Figura 8.14. En este gráfico, se destaca que el 52.2% de los participantes calificó la calidad visual del juego con la máxima puntuación (5), el 30.4% otorgó una calificación de 4 y un 17.4% una puntuación media de 3 puntos. No se registraron puntuaciones por debajo de 3, lo que refuerza la alta aceptación del diseño visual entre los jugadores.

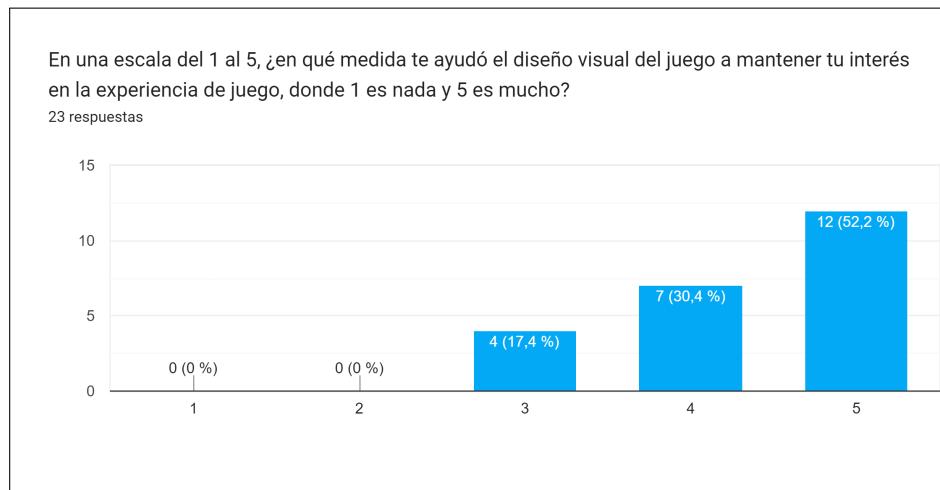


Figura 8.14: Gráfico de barras que muestra la influencia del diseño visual del juego en el interés de los jugadores.

8.7.2. Validación de aceptación en mecánicas de juego

Para validar el objetivo específico de implementar mecánicas de juego que mantengan a los jugadores comprometidos y ofrezcan una experiencia envolvente, se realizaron una serie de preguntas a los usuarios durante las pruebas.

En la Figura 8.15, se observa que el 30.4% de los jugadores otorgó la calificación más alta (5), indicando un alto nivel de compromiso con el juego. Además, el 60.9% calificó su experiencia con

un 4, mientras que solo el 8.7 % le asignó una puntuación de 3. Aunque la mayoría de los jugadores eligió una calificación de 4 en lugar de 5, no se registraron puntuaciones por debajo de la puntuación media, lo que evidencia un elevado grado de involucramiento por parte de los usuarios durante la experiencia de juego.

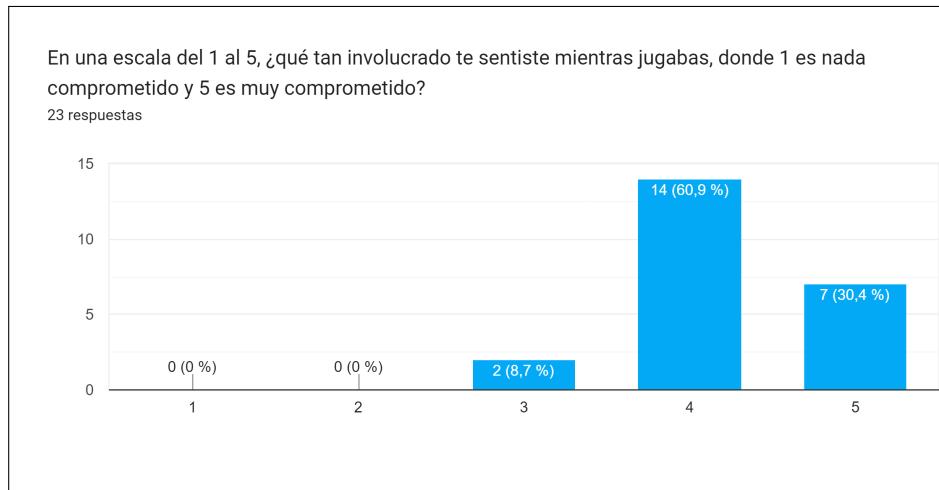


Figura 8.15: Gráfico de barras que muestra el nivel de involucramiento de los jugadores

Por otro lado, la Figura 8.16 presenta los resultados sobre la percepción de la intuición de las mecánicas del juego. El 60.9 % de los participantes consideró que las mecánicas eran muy intuitivas, con una puntuación de 5, mientras que el 34.8 % las calificó con 4. Solo el 4.3 % de los jugadores dio una puntuación de 3, lo que indica que, en general, las mecánicas del juego fueron percibidas como fáciles de entender y aprender.

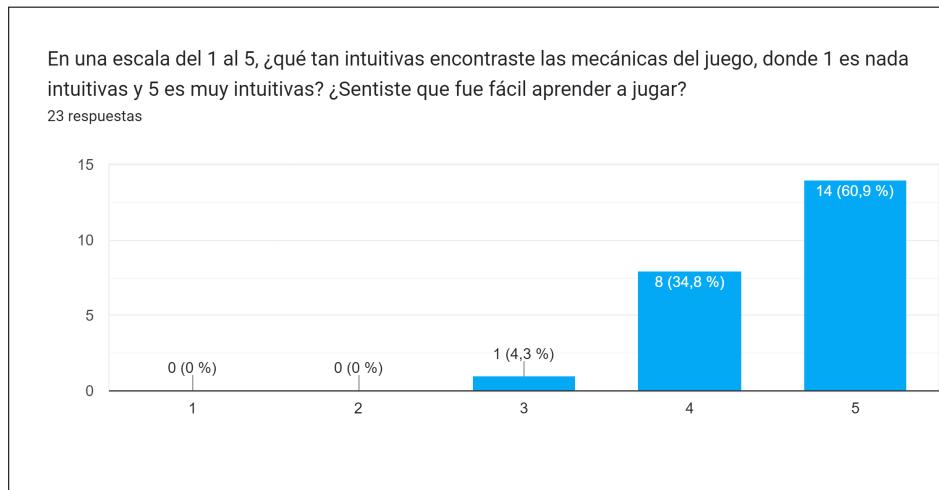


Figura 8.16: Gráfico de barras que muestra la percepción de los jugadores sobre la intuición de las mecánicas del juego.

Además de estas gráficas, en la Tabla 8.6 se muestran algunos momentos de inmersión descritos por los jugadores durante las pruebas. Para obtener estas respuestas, se les preguntó a los participantes: '¿Hubo momentos en el juego donde las mecánicas te hicieron sentir realmente inmerso en la experiencia? ¿Puedes describir esos momentos?'. Esta pregunta fue planteada con el fin de

identificar instancias específicas en las que las mecánicas del juego lograron generar un mayor nivel de inmersión en la narrativa y el entorno del juego.

Respuesta	Descripción de la inmersión
1	La comunicación constante con los NPC's para descubrir mi rol en el juego y la historia es lo que más me hizo sentir inmerso.
2	Sí, al hablar con los NPCs sentía que estaba hablando con una persona real.
3	El poder recoger un ítem del mapa y preguntarle a un NPC cómo usarlo.

Tabla 8.6: Momentos de inmersión en el juego según los jugadores

8.7.3. Validación de interacción con NPCs

La interacción con los NPCs fue otro aspecto clave validado en las pruebas. En la Figura 8.17, se observa que el 47.8 % de los jugadores calificó con la máxima puntuación (5) la coherencia de las respuestas de los NPCs con el contexto del juego, mientras que el 39.1 % otorgó una calificación de 4. El 13 % de los participantes evaluó con un 3, lo cual muestra que las respuestas de los NPCs fueron coherentes y alineadas con el entorno narrativo del juego.

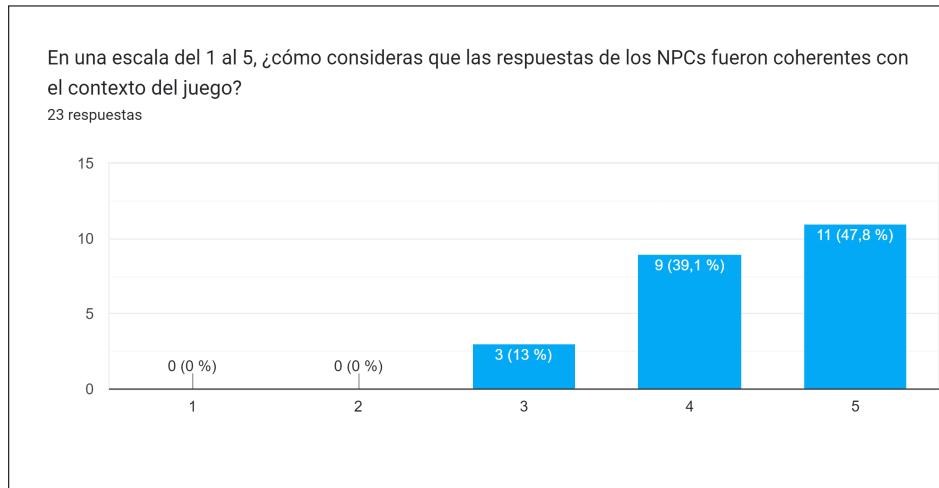


Figura 8.17: Gráfico de barras que muestra la percepción de los jugadores sobre la coherencia de las respuestas de los NPCs con el contexto del juego.

En la Figura 8.18, se muestra un gráfico circular que revela si los jugadores notaron diferencias en las interacciones con los NPCs tras haber jugado dos veces. El 87 % de los jugadores indicó que sí notó diferencias, mientras que el 13 % respondió que no. Estos resultados sugieren que las interacciones con los NPCs se percibieron como dinámicas y variadas.

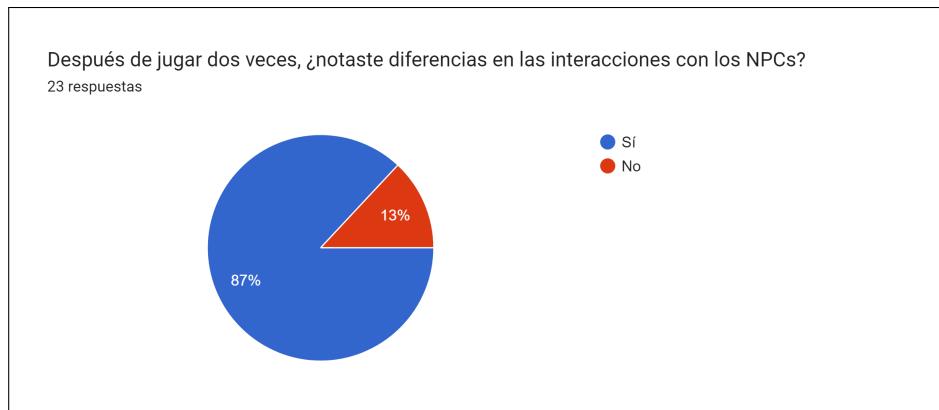


Figura 8.18: Gráfico circular que muestra si los jugadores notaron diferencias en las interacciones con los NPCs después de jugar dos veces.

Finalmente, la Figura 8.19 presenta los resultados sobre la capacidad de las respuestas de los NPCs para ayudar a los jugadores a avanzar en el juego o resolver problemas. El 82.6% de los participantes afirmó que las respuestas generadas por los NPCs fueron útiles, mientras que el 17.4% indicó que no lo fueron. Esto refuerza la importancia de la interacción con los NPCs como una herramienta que enriquece la experiencia de juego y facilita la progresión.

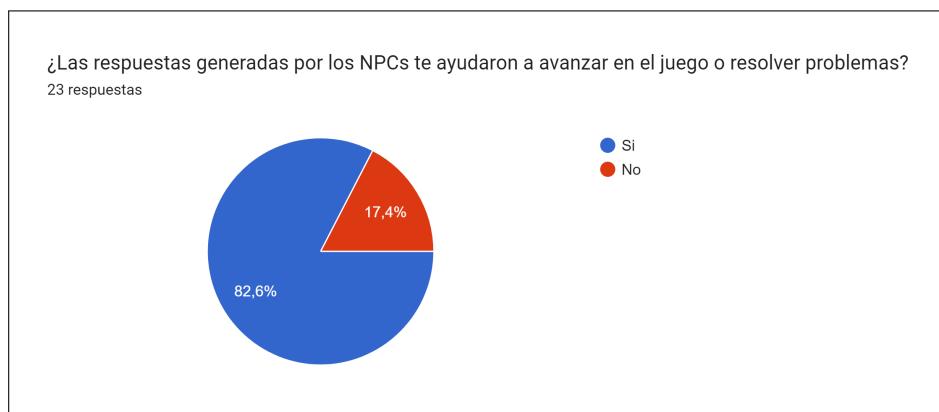


Figura 8.19: Gráfico de barras que muestra la percepción de los jugadores sobre la coherencia de las respuestas de los NPCs con el contexto del juego.

8.8. Análisis de Costos y Consumo de API

A continuación, se presentan los resultados del análisis de costos y consumo de la API de OpenAI en distintos escenarios de uso. Estos resultados permiten visualizar el impacto financiero del proyecto en contextos de uso intensivo y en sesiones individuales, proporcionando información clave para futuras decisiones de implementación.

8.8.1. Consumo de API en una Sesión de Juego

Este apartado describe el uso de la API de OpenAI en una única sesión de juego, incluyendo la cantidad de tokens consumidos y el número de solicitudes realizadas a la API durante la interacción

en tiempo real de un jugador. Como se observa en la figura 8.20, en una sesión de aproximadamente cinco minutos con interacciones básicas y breves con los NPCs, se realizaron un total de 36 solicitudes a la API.



Figura 8.20: Número de solicitudes a la API realizadas en el mes de noviembre.

Asimismo, en la figura 8.21 se muestra que estas 36 solicitudes generaron un consumo de 93,108 tokens durante las interacciones con el modelo de lenguaje (LLM).

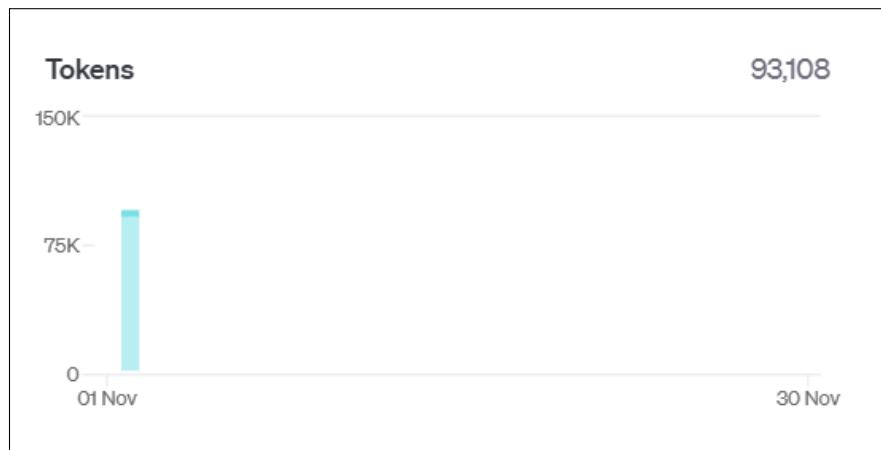


Figura 8.21: Cantidad de tokens consumidos en el mes de noviembre.

8.8.2. Gastos Asociados a una Sesión de Juego

Finalmente, se presentan los costos generados en una sesión típica de juego, los cuales ascendieron a \$0.05 USD, como se muestra en la figura 8.22. Cabe recordar que esta sesión, descrita previamente en términos de consumo de tokens y solicitudes, consistió en una prueba sencilla donde los LLMs se emplearon para generar información básica del entorno y los personajes, así como para facilitar interacciones básicas del jugador con los NPCs.

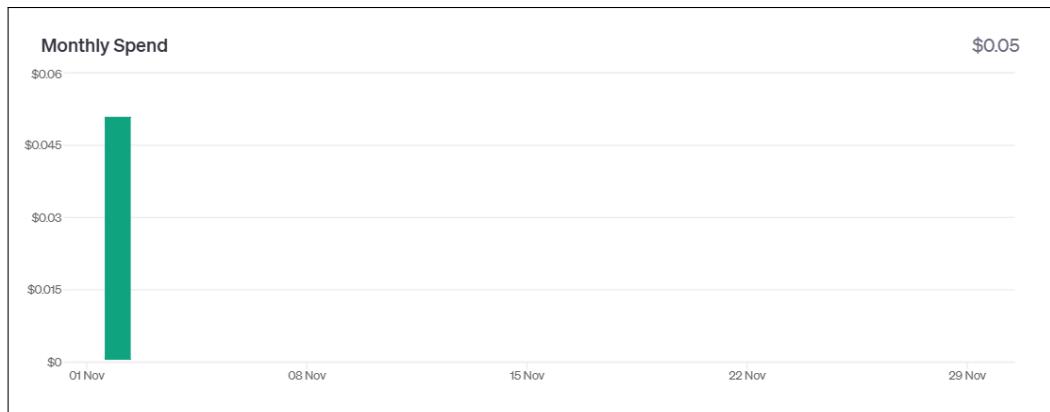


Figura 8.22: Gastos mensuales acumulados en el uso de la API de OpenAI al inicio del mes de noviembre.

8.8.3. Consumo de API en un Mes de Pruebas

Después de analizar el consumo de API en una sesión individual, se procedió a evaluar el uso acumulado durante un mes completo de pruebas intensivas. Según el dashboard de OpenAI, octubre fue el mes de mayor consumo, lo cual es congruente dado que fue el período en el que se realizaron la mayor cantidad de pruebas. Como se muestra en la figura 8.23, en octubre se realizaron un total de 4,024 solicitudes a la API de OpenAI.

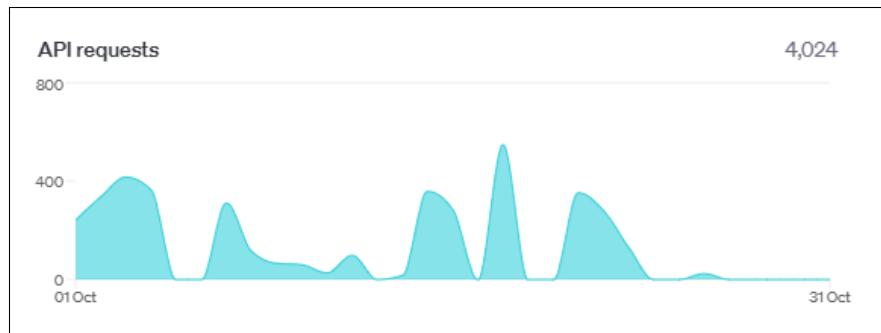


Figura 8.23: Número total de solicitudes a la API durante el mes de octubre.

Además, en la figura 8.24 se observa el total de tokens consumidos durante el mismo período, alcanzando la cifra de 8,235,853 tokens. Este nivel de consumo reafirma la intensidad de las interacciones y la cantidad de contenido generado dinámicamente por los LLMs, lo que fue fundamental para enriquecer la experiencia del jugador y mantener la variabilidad en cada sesión.

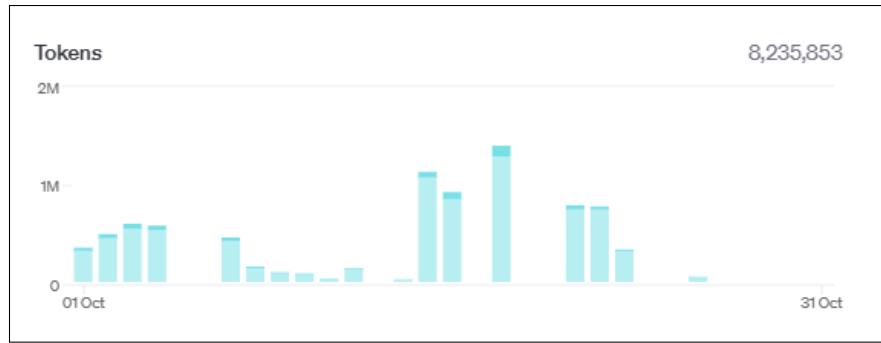


Figura 8.24: Cantidad total de tokens consumidos durante el mes de octubre.

8.8.4. Costos Acumulados en un Mes de Pruebas

En relación con los costos acumulados durante el mes completo de pruebas, la figura 8.25 muestra un gasto mensual total de \$4.53 USD en el uso de la API de OpenAI. Este valor proporciona una estimación de la inversión necesaria para sostener un flujo continuo de interacciones y generación de contenido mediante los LLMs. Además, sirve como una referencia útil para proyectar los costos en caso de que se considere implementar esta tecnología en etapas de producción y llevar el proyecto a una escala mayor.

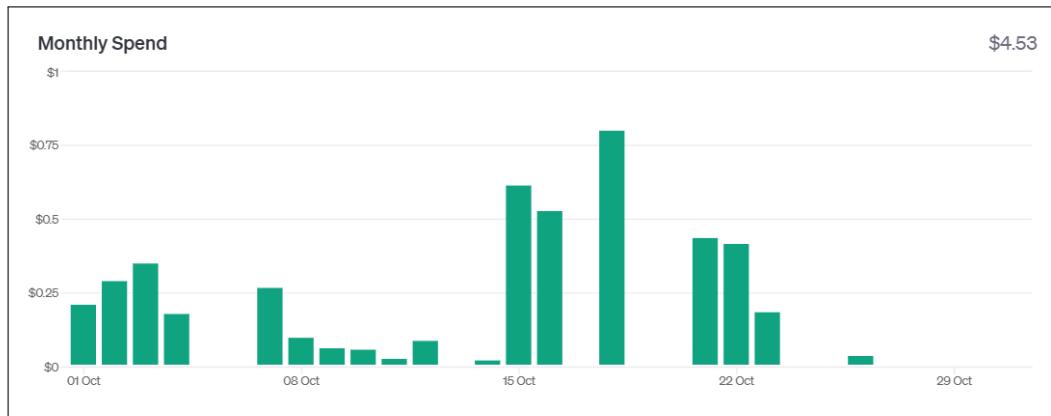


Figura 8.25: Gastos mensuales acumulados en el uso de la API de OpenAI durante el mes de octubre.

CAPÍTULO 9

Análisis de resultados

En el desarrollo de este proyecto, se lograron cumplir los objetivos planteados, integrando modelos de lenguaje grande (LLMs) y técnicas de generación procedural de contenido para crear una experiencia de juego única y personalizada. El videojuego desarrollado no solo permitió que los jugadores interactuaran de manera dinámica con los personajes no jugables (NPCs) gracias a las capacidades avanzadas de los LLMs, sino que también ofreció mundos generados proceduralmente que garantizaron variabilidad y riqueza en cada partida. Esto demostró la efectividad de las técnicas implementadas para ofrecer experiencias diversas, asegurando que cada sesión de juego fuera distinta.

A medida que los jugadores avanzaban, la generación procedural resultó fundamental, permitiendo la creación de sectores únicos que variaban en cada sesión de juego. Aunque este proceso no es evidente a simple vista para el jugador, funciona en segundo plano, contribuyendo significativamente a una experiencia de juego única. La generación de ambientes orgánicos de forma procedural potenció la emoción de aventura y exploración, tal como se definió desde un inicio en la conceptualización de la idea, impulsando a los jugadores a descubrir lugares desconocidos que ningún otro jugador ha explorado previamente.

Otro claro ejemplo fue la integración de los NPCs, donde el uso de técnicas avanzadas de generación procedural permitió crear personajes con características visuales únicas que enriquecieron las interacciones en el juego. Cada NPC no solo aportó variedad al mundo, sino que también ofreció oportunidades para diálogos y misiones que reflejaron la diversidad y profundidad del entorno. Estos detalles, aunque parezcan pequeños, resultaron esenciales para construir un mundo de juego convincente y creíble, en el que los jugadores sintieron que cada interacción tenía un toque único.

Además, la implementación de modelos de lenguaje grande (LLMs) fue fundamental para mejorar la interacción entre los NPCs y los jugadores. Estos modelos no solo facilitaron una comunicación directa y fluida, sino que hicieron que los diálogos fueran más naturales y contextuales. Esta capacidad de adaptar las interacciones en tiempo real no solo potenció la inmersión, sino que también incrementó la relevancia de cada NPC dentro de la narrativa general del juego. Con la integración de LLMs, el proyecto no solo añadió una mecánica de interacción, sino que también estableció un nuevo estándar para el desarrollo narrativo en los videojuegos.

Cada una de estas mejoras contribuyó a una experiencia más rica y variada, lo que demostró el potencial de las tecnologías emergentes para transformar el desarrollo de videojuegos. A medida que el proyecto avanzaba, la integración de estas técnicas continuó abriendo nuevas posibilidades para

la personalización y la innovación en el diseño de juegos, prometiendo una evolución constante en la forma en que los jugadores experimentan y interactúan con los mundos virtuales.

Al finalizar el proceso de desarrollo, se observó cómo la integración de la interfaz de usuario (UI) en el videojuego contribuyó significativamente a mejorar la experiencia del usuario, facilitando una interacción intuitiva y presentando información esencial de forma clara y eficaz. Este resultado destacó la eficacia de una UI bien diseñada para reforzar no solo la funcionalidad del juego, sino también para enriquecer la experiencia inmersiva al reducir distracciones y maximizar la comprensión y facilidad de uso.

Las pruebas de jugabilidad fueron fundamentales para validar los componentes clave del proyecto. A partir de las primeras impresiones y la retroalimentación obtenida, se realizaron ajustes que mejoraron la calidad del producto final, especialmente en lo referente a las mecánicas de juego y las interacciones con los NPCs. Este proceso iterativo permitió afinar las áreas que necesitaban atención, garantizando que el juego brindara una experiencia coherente y atractiva para los jugadores.

Después de aplicar las correcciones y completar una versión final del producto, se sometió a una nueva prueba con el fin de recopilar las impresiones de los jugadores. En el aspecto visual, los jugadores expresaron una valoración positiva del entorno gráfico del juego, lo que sugirió que el diseño estético cumplió su objetivo de crear un mundo envolvente y visualmente atractivo. El equilibrio entre los elementos gráficos y la funcionalidad fue clave para mantener a los jugadores inmersos, destacando la importancia del diseño visual en la experiencia de juego.

En cuanto a las mecánicas de juego, los jugadores se sintieron comprometidos con la propuesta, lo que indicó que las mecánicas implementadas fueron intuitivas y mantuvieron el interés a lo largo de las sesiones de prueba. Las interacciones con objetos y personajes dentro del juego, junto con la fluidez de las acciones, lograron que los usuarios se sintieran integrados en la narrativa, lo cual fue un objetivo fundamental del proyecto. Estos momentos de inmersión, en los que los jugadores interactuaban con el entorno y los NPCs, reflejaron el éxito de las mecánicas en generar una experiencia envolvente.

Por otro lado, la integración de los NPCs mediante modelos de lenguaje grande (LLMs) fue bien recibida. La capacidad de los NPCs para generar respuestas coherentes y contextualizadas dentro del mundo del juego añadió un nivel de profundidad a las interacciones que aumentó la inmersión. Los jugadores percibieron estas interacciones como naturales y dinámicas, lo cual demostró la eficacia de la implementación de los LLMs en el desarrollo narrativo.

Finalmente, el análisis del consumo de la API de OpenAI reveló patrones significativos de uso y permite comprender mejor el impacto de su integración en el desarrollo de un videojuego. A lo largo de las pruebas, el consumo de tokens y solicitudes mostró una demanda constante de recursos, especialmente durante períodos de evaluación intensiva. Esto es importante para futuros desarrolladores que buscan replicar esta metodología, ya que permite anticipar el consumo de recursos al implementar modelos de lenguaje en un entorno de juego interactivo.

Aunque el análisis de costos no fue un objetivo central del proyecto, se decidió incluirlo para proporcionar información práctica y relevante. Estos datos pueden ser de gran utilidad para desarrolladores que deseen explorar la viabilidad de integrar tecnologías similares en proyectos propios, ayudándoles a evaluar y planificar la inversión necesaria para garantizar una experiencia de juego fluida y sostenible en el tiempo.

En conjunto, estos resultados evidenciaron que los objetivos planteados se lograron de manera efectiva, creando una experiencia de juego equilibrada entre lo visual, lo narrativo y lo interactivo. El uso de tecnologías emergentes como la generación procedural y los modelos de lenguaje no solo enriqueció la jugabilidad, sino que también introdujo un elemento de personalización y variabilidad en cada partida, lo que prometió una experiencia fresca y dinámica en cada sesión.

CAPÍTULO 10

Conclusiones

- La integración de *Large Language Models* (LLMs) en los videojuegos demostró ser una herramienta efectiva para mejorar la narrativa dinámica y personalizar la experiencia de juego de los usuarios, proporcionando interacciones más inmersivas con los NPCs.
- La implementación de técnicas de generación procedural de contenido logró aumentar la rejubabilidad del videojuego, generando entornos únicos en cada sesión sin comprometer la coherencia narrativa.
- El enfoque en la estética visual, unido a las técnicas de generación procedural y la integración de inteligencia artificial, permitió desarrollar un videojuego que equilibra la calidad visual con la experiencia de juego, asegurando una inmersión total para los jugadores.
- La combinación de LLMs con generación procedural permitió que el juego no solo ofrezca variaciones visuales, sino también una adaptación de la historia, lo que amplía la profundidad de la experiencia de juego.

CAPÍTULO 11

Recomendaciones

- Para futuros desarrollos que busquen llevar técnicas similares a la producción, es crucial investigar y establecer métodos para el uso eficiente de APIs de Modelos de Lenguaje Grande. Se recomienda especialmente planificar un enfoque que controle los costos asociados y mantenga el consumo dentro de un presupuesto predefinido, garantizando así la sostenibilidad del proyecto a largo plazo.
- Para futuras investigaciones en la integración de Modelos de Lenguaje Grande (LLMs) en videojuegos con fines experimentales, se recomienda considerar el desarrollo de un proyecto en 2D en lugar de 3D. Esto podría simplificar el proceso de desarrollo y permitir un mayor enfoque en la integración de la inteligencia artificial, maximizando el aprendizaje y los resultados sin distraer recursos en aspectos gráficos más complejos.
- Para futuras implementaciones de la API de OpenAI en proyectos desarrollados en Unity, se recomienda explorar el uso de modelos más avanzados para lograr interacciones más naturales y de mayor calidad con los personajes, optimizando así la experiencia de usuario y enriqueciendo la narrativa dentro del juego.

Glosario

- **Script:** Un conjunto de instrucciones escritas en un lenguaje de programación que automatiza tareas dentro de un software o juego.
- **Scriptable Object:** Un tipo especial de objeto en Unity que permite almacenar datos y compartirlos entre diferentes instancias y escenas sin necesidad de duplicarlos, utilizado comúnmente para manejar configuraciones y recursos.
- **API:** Interfaz de programación de aplicaciones, un conjunto de definiciones y protocolos que permite a diferentes programas comunicarse entre sí, facilitando la integración de funcionalidades externas en un proyecto.
- **Material:** En Unity, un Material define cómo un objeto se verá al ser renderizado. Asigna texturas y propiedades visuales como el color, brillo o transparencia a los objetos tridimensionales en un juego.
- **Prompts:** Instrucciones o entradas que se le proporcionan a un modelo de lenguaje (como GPT) para generar una respuesta o contenido, basado en el contexto o la solicitud del usuario.
- **Isométrico:** Un tipo de perspectiva visual en diseño y videojuegos donde los objetos se muestran en una vista tridimensional con un ángulo de 30 grados, sin distorsión de perspectiva, creando una representación 3D en un plano 2D.
- **Vista Top-Down:** Una perspectiva visual en videojuegos en la que la cámara se posiciona directamente encima del personaje o escenario, ofreciendo una vista aérea completa.
- **RPG:** Acrónimo de Role-Playing Game.^o juego de rol, un género de videojuegos donde los jugadores asumen el rol de un personaje en un mundo virtual, desarrollando habilidades y tomando decisiones que afectan la narrativa y el entorno.
- **Assets:** Recursos digitales utilizados en videojuegos, como modelos 3D, texturas, sonidos y animaciones, que se integran para construir el contenido visual y auditivo del juego.
- **Instanciar:** El proceso de crear una instancia o copia de un objeto en programación, particularmente en videojuegos, permitiendo que múltiples copias de un objeto existan en el entorno del juego.
- **Tokens:** Unidades de conteo en procesamiento de lenguaje natural que representan fragmentos de texto. En la API de OpenAI, los tokens son el recurso consumido al generar respuestas; incluyen palabras o partes de palabras, y su cantidad determina el costo de la solicitud al modelo.

Bibliografía

- [1] Adams, Ernest y Andrew Rollings: *Fundamentals of Game Design*. Pearson Prentice Hall, ilustrada edición, 2007.
- [2] AI, Inworld: *Inworld: Framework para la creación de NPCs interactivos impulsados por IA*. Inworld Documentation, 2023. Disponible en: <https://inworld.ai>.
- [3] Azure, Microsoft: *Azure AI: Servicios escalables de inteligencia artificial para desarrolladores*. Microsoft Azure Documentation, 2023. Disponible en: <https://azure.microsoft.com/en-us/products/ai-services>.
- [4] Brown, T. B. y cols.: *Language Models are Few-Shot Learners*. arXiv preprint arXiv:2005.14165, 2020.
- [5] Calle, M. De La: *Desarrollo de un Videojuego en Unity*, 2019. https://oa.upm.es/58120/1/TFG_MARIO_DE_LA_CALLE_PEMAU.pdf, Universidad Politécnica de Madrid.
- [6] Clark, Peter: ‘*The Endless Mission’ Wants to Teach You How to Program Video Games*’. Variety, Junio 2018. <https://variety.com/2018/gaming/features/the-endless-mission-interview-1202846005/>, Available at: <https://variety.com/2018/gaming/features/the-endless-mission-wants-to-teach-you-how-to-program-video-games-1202845916/>.
- [7] Face, Hugging: *Hugging Face: Transformando la IA con modelos de lenguaje avanzados*. Hugging Face Documentation, 2023. Disponible en: <https://huggingface.co/docs/transformers/index>.
- [8] García, F. y M. Gértrudix: *Aplicaciones de los videojuegos de contenido histórico en el aula*. ICONO14 Revista científica de Comunicación y Tecnologías emergentes, 4(1):217–230, 2006.
- [9] Grinstead, C. M. y J. L. Snell: *Introduction to Probability*. American Mathematical Society, 1997.
- [10] Healy, J.: *Learn C# in One Day and Learn It Well*. LCF Publishing, 2018.
- [11] Ibarra Monteagudo, Yohandy: *Proceso de pruebas en el desarrollo de videojuegos*. Tesis de Licenciatura, Universidad de las Ciencias Informáticas, 2018.
- [12] Kenney: *Kenney game assets*. <https://kenney.nl/>, n.d. Accessed: May, 2024.
- [13] Latitude, Inc.: *AI Dungeon*. <https://aidungeon.com/>, 2021. Accessed: 2021-10-10.
- [14] LeCun, Y., Y. Bengio y G. Hinton: *Deep Learning*. Nature, 521(7553):436–444, 2015.

- [15] López, J.: *Videojuegos, un rey Midas del entretenimiento y de los negocios.* Crónica, Diciembre 2021. Disponible en: [<https://www.cronica.com.mx/escenario/videojuegos-rey-midas-entretenimiento-negocios.html>].
- [16] MacDonald, Keza: *No Man's Sky developer Sean Murray: 'It was as bad as things can get'.* The Guardian, Julio 2018. <https://www.theguardian.com/games/2018/jul/20/no-mans-sky-next-hello-games-sean-murray-harassment-interview>, Available at: <https://www.theguardian.com/games/2018/jul/20/no-mans-sky-developer-sean-murray-it-was-as-bad-as-things-can-get>.
- [17] Nuić, Hrvoje y Žarko Mihajlović: *Algorithms for procedural generation and display of trees.* En *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, páginas 230–235. IEEE, 2019.
- [18] OpenAI: *OpenAI.* <https://www.openai.com>, 2023.
- [19] Russell, S. y P. Norvig: *Artificial Intelligence: A Modern Approach.* Pearson, 2020.
- [20] Shaker, Noor, Julian Togelius y Mark J. Nelson: *Procedural Content Generation in Games.* Springer, 2016.
- [21] Silva, J. y P. Souza: *Game AI Pro: Collected Wisdom of Game AI Professionals.* CRC Press, 2021.
- [22] Technologies, Unity: *Unity User Manual*, 2019. <https://docs.unity3d.com/es/530/Manual/UnityManual.html>.
- [23] Trenzano, M.: *Desarrollo de un videojuego de plataformas 2D en Unity*, 2017. <https://riunet.upv.es/bitstream/handle/10251/91746/TRENZANO%20-%20Desarrollo%20de%20un%20videojuego%20de%20plataformas%202D%20con%20Unity.pdf>, Universidad Politécnica de Valencia.
- [24] Xia, Feng, Jiaying Liu, Hansong Nie, Yonghao Fu, Liangtian Wan y Xiangjie Kong: *Random Walks: A Review of Algorithms and Applications.* IEEE Transactions on Emerging Topics in Computational Intelligence, 0(0), May 2019.
- [25] Zhou, Zhi Hua: *Machine Learning.* Springer Nature, 2021.
- [26] Órus, A.: *Industria mundial del videojuego - Datos estadísticos.* Statista, Mayo 2024. Disponible en: [<https://es.statista.com/temas/9150/industria-mundial-del-videojuego/topFacts>].

Anexos

.1. Ejemplo básico de integración de API de OpenAI en C# y Unity

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using OpenAI;
5
6  public class ChatGPTManager : MonoBehaviour
7  {
8      private OpenAIApi openAI = new OpenAIApi();
9      private List<ChatMessage> messages = new List<ChatMessage>();
10
11     public async void AskGPT(string newText)
12     {
13         ChatMessage message = new ChatMessage();
14         message.Content = newText;
15         message.Role = "user";
16
17         messages.Add(message);
18
19         CreateChatCompletionRequest request = new CreateChatCompletionRequest();
20         request.Messages = messages;
21         request.Model = "gpt-3.5-turbo";
22
23         var response = await openAI.CreateChatCompletion(request);
24
25         if (response.Choices != null && response.Choices.Count > 0)
26         {
27             var ChatResponse = response.Choices[0].Message;
28             messages.Add(ChatResponse);
29
30             Debug.Log(ChatResponse.Content);
31         }
32     }
33
34     void Start()
35     {
36         AskGPT("Hola como estas el dia de hoy?, para responder a este prompt, lo
37             unico que tienes que contestar es '1' para estoy bien y '2' para indicar
38             que estas mal, tu respuesta no debe ser mas que un numero, ya sea 1 o
39             2");
40     }
41 }
```

2. Dashboard de Costos y Actividad de la API de OpenAI

2.1. Dashboard de Costos

La figura 1 muestra el desglose de costos acumulados en el uso de la API de OpenAI durante el mes de octubre, brindando una perspectiva clara sobre los gastos generados en este período.

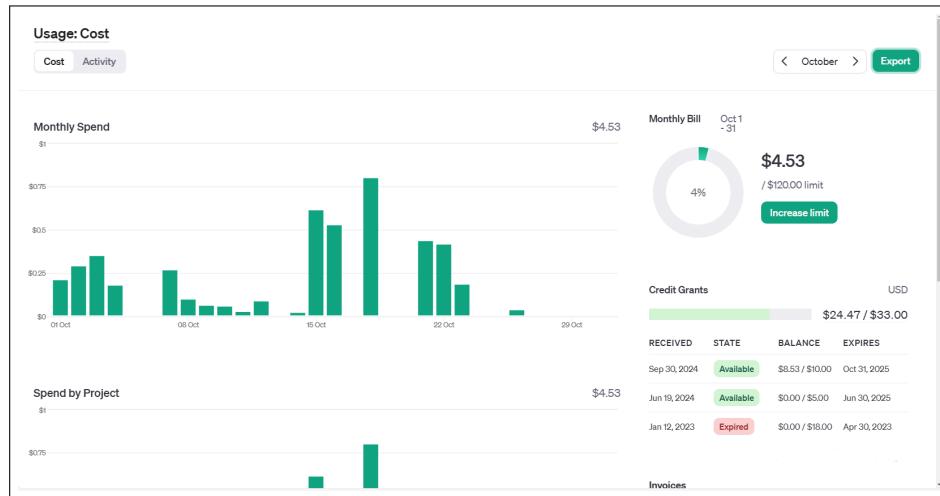


Figura 1: Desglose de costos mensuales acumulados en el uso de la API de OpenAI en octubre.

2.2. Dashboard de Actividad

La figura 2 presenta el monitoreo de la actividad de la API de OpenAI, incluyendo el uso de tokens y las solicitudes realizadas, lo cual es fundamental para analizar el comportamiento y la carga de trabajo en el proyecto.

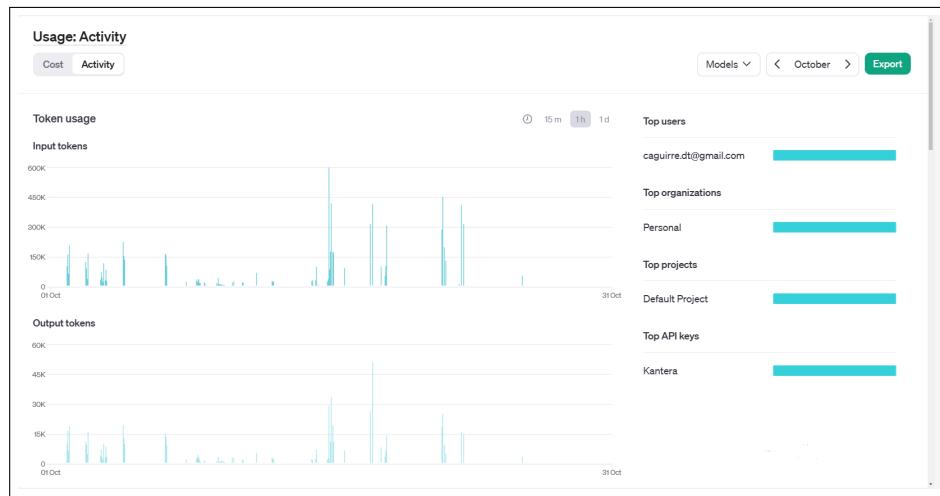


Figura 2: Monitoreo de actividad en la API de OpenAI, mostrando el uso de tokens y las solicitudes en octubre.