FACULTAD UISEK BUSINESS & DIGITAL SCHOOL

Trabajo de Titulación:

“Diseño e Implementación de un Serious Game 2D

para la Enseñanza del Subneteo de Redes”

Realizado por:

PABLO ANDRÉS JUSTICIA RODRÍGUEZ

Director del proyecto:

MSc. DAVID PATRICIO CANDO GARZÓN

Como requisito para la obtención del título de:

INGENIERO EN SISTEMAS DE INFORMACIÓN CON ÉNFASIS EN CONTENIDOS INTERACTIVOS

Quito, julio 2025

DECLARATORIA

El presente Trabajo de Titulación titulado :

“Diseño e Implementación de una plataforma educativa 3D para la Concientización sobre Protección de Datos Personales en Redes Sociales”

Realizado por:

PABLO ANDRÉS JUSTICIA RODRÍGUEZ

Ha sido dirigido por el profesor :

MSc. DAVID PATRICIO CANDO GARZÓN

Quien considera que constituye un trabajo original de su autor

Como requisito para la obtención del título de:

INGENIERO EN SISTEMAS DE INFORMACIÓN CON ÉNFASIS EN CONTENIDOS INTERACTIVOS

MSc. DAVID PATRICIO CANDO GARZÓN

PROFESOR

DECLARACION JURAMENTADA

Yo , PABLO ANDRÉS JUSTICIA RODRÍGUEZ , con cedula de identidad No. 1724448996, declaro bajo juramento que el trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado a calificación profesional; y que ha consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración, cedo mis derechos de propiedad intelectual correspondiente a este trabajo, a la UNIVERSIDAD INTERNACIONAL SEK, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

Pablo Andrés Justicia Rodríguez

C.C: 1724448996

LOS PROFESORES INFORMANTES:

PhD. JOE LUIS CARRION JUMBO

PhD. VIVIANA ELIZABETH CAJAS CAJAS

Después de revisar el trabajo presentado lo han calificado como apto para su defensa oral ante el tribunal examinador.

PhD. Joe Carrión

PhD. Viviana Cajas

INDICE

Tabla de contenido

Imágenes

INDICE DE TABLAS

DEDICATORIA

A mis padres, que me han visto pasar por buenos y malos momentos siempre apoyándome a terminar mi carrera, siendo mi pilar de apoyo y fortaleza para lograr este tan importante título. También dedicar este trabajo a una de las personas más importantes de mi vida, mi abuelita Juana Calvache la cual desde pequeño, con su amor incondicional, me enseñó a ser perseverante y luchador ante lo que se me presente.

A mí increíble hermano que ha pasado estos años acompañándome con sus ocurrencias fugaces y es mi principal crítico. Gracias a todos los previamente mencionados he llegado a tener la capacidad de estar posicionado donde estoy con la cabeza en alto, presentando esta tesis.

A mis amigos, que a lo largo de los años nos hemos seguido frecuentando y que me han apoyado en momentos difíciles, donde he podido sentirme en un segundo hogar y he vuelto a ganar fuerzas para continuar en este largo camino.

Con esto quiero mostrar el orgullo que tengo de haber tenido gente que tuviera confianza en mí y que espero estén orgullosos del fruto de todos estos años

AGRADECIMIENTO

En primer lugar, a Dios quien me ha permitido estar aquí presente, y me ha cuidado en mis múltiples momentos de debilidad, siempre me ha iluminado para poder presentar este trabajo, me considero afortunado y bendecido, al tener la confianza de sentir su compañía.

A mi padre el cual me ha guiado en diferentes aspectos de mi vida, ha estado presente en momentos difíciles, siempre buscando que me convierta en un hombre de bien.

A mi madre que ha sido una figura amorosa y rigurosa en mi vida, la cual se ha encargado de impartir una actitud perseverante y determinada. Ella ha velado por mi bienestar y ha buscado siempre darme las herramientas para poder progresar en mis habilidades, con ellos me he vuelto más capaz de seguir adelante, siendo guiado por ambos en cada paso de este largo camino.

A mi hermano que ha sido capaz de mostrarme nuevos enfoques de la vida y ha desarrollado en mí una persona un tanto más paciente.

A mi abuelita, que desde pequeño me cuidó y me enseñó conceptos como la valentía de enfrentar el día a día y la responsabilidad de mis acciones.

También quiero agradecer a mi director de tesis, quien me ha guiado y con sus conocimientos y dirección me ha permitido llegar a concluir este proceso tan importante

A la Universidad Internacional SEK, a todo su personal, y especialmente a mis profesores, por haberme acogido durante estos años contribuyendo a mi formación para convertirme en un profesional que busca ayudar a la sociedad, tratando de innovar y de ser posible algún día ayudar a otros con encontrar su camino dentro de esta profesión.

RESUMEN

El presente trabajo de titulación tiene como objetivo el diseño y desarrollo de un videojuego educativo 2D, también conocido como serious game, enfocado en facilitar la enseñanza del subneteo de redes IP. Esta herramienta interactiva responde a la necesidad de encontrar métodos didácticos alternativos frente a las tradicionales estrategias teóricas, que muchas veces resultan abstractas, poco motivadoras y difíciles de comprender para estudiantes de carreras técnicas o tecnológicas.

A través de la utilización del motor de desarrollo Unity, se implementó una experiencia de aprendizaje dinámica que combina teoría, práctica, preguntas interactivas y minijuegos diseñados para evaluar y reforzar el conocimiento adquirido. El juego se estructura en niveles progresivos que abordan desde la introducción conceptual hasta la aplicación práctica, mediante ejercicios que simulan mecánicas como la conexión de direcciones IP mediante cables, selección de máscaras correctas y validación de respuestas con retroalimentación inmediata

Como resultado, el proyecto no solo constituye una propuesta funcional y pedagógicamente sólida, sino que también demuestra la viabilidad de integrar videojuegos en el proceso formativo como un recurso complementario o un autoaprendizaje, especialmente en áreas técnicas donde la comprensión conceptual suele ser un obstáculo inicial. Se concluye que el uso de juegos serios puede transformar el aprendizaje tradicional en experiencias más significativas, interactivas y accesibles para las generaciones de estudiantes.

Palabras clave

Aprendizaje interactivo, Enseñanza de redes, Gamificación educativa, Serious game, Subneteo de redes, Unity, Videojuegos educativos

ABSTRACT

This degree project presents the design and implementation of a 2D educational video game, also known as a serious game, aimed at facilitating the learning of IP subnetting. The interactive tool emerges as a pedagogical alternative to traditional theoretical methods, which often proves to be abstract, demotivating, and difficult to grasp for students in technological careers.

Using the game engine, a dynamic learning experience was developed that integrates theoretical content, practical exercises, interactive quizzes and minigames specifically designed to evaluate and reinforce knowledge acquisition. The game is structured into progressive levels that guide the players from basic concepts to applied subnetting scenarios Its features include tasks as cable-matching IP addresses, identifying correct subnetting masks, and receiving immediate feedback based on player responses

As a result, the project stands as a solid educational proposal, demonstrating the feasibility and benefits of integrating serious games into technical education. It offers a compelling model for improving student motivation, understanding, and knowledge retention, highlighting the potential of video games as powerful learning tools for new generations.

Keywords

Educational gamification, Educational video games, Interactive learning, Network subnetting, Network teaching, Serious game, Unity

1. CAPÍTULO 1

INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

En la actualidad, el aprendizaje de fundamentos en redes informáticas[1] representa un reto considerable para estudiantes de nivel técnico o universitario, especialmente en lo que respecta a conceptos complejos como la implementación de redes mediante el uso de subneteo[2]. A pesar de su relevancia en el campo de las telecomunicaciones, la administración de sistemas y la configuración de redes a nivel empresarial, muchos alumnos presentan dificultades al interiorizar estos conocimientos debido a la naturaleza abstracta y tradicional que dependen en exceso de contenidos teóricos, lecturas densas o ejercicios mecánicos sin un contexto práctico claro ni una retroalimentación inmediata. En consecuencia, se puede observar fácilmente una desmotivación creciente entre los estudiantes, una falta de compresión real de los procedimientos técnicos y un aprendizaje superficial que limita su preparación profesional

La situación se vuelve aún más compleja al considerar que los jóvenes actuales, pertenecientes a generaciones nativas digitales[3], muestran una marcada preferencia por entornos interactivos, visuales y lúdicos para el aprendizaje. Sin embargo, en muchos centros educativos no se aprovechan estas características, lo que genera una desconexión entre la forma de enseñanza y las necesidades de los reales de los estudiantes. Ante esta

1. Redes informáticas: Son conjuntos de sistemas informáticos interconectados que permiten compartir información entre los equipos
2. Subneteo: Es el proceso de dividir un conjunto de direcciones IP en dos o más rangos, denominados subredes
3. Generaciones nativas digitales: Se refiere a las personas nacidas en la era digital o que crecieron familiarizadas con la tecnología desde la infancia

brecha pedagógica[1], surge la necesidad de explorar herramientas innovadoras que integren tecnología e interactividad en un solo entorno mientras se ofrece nuevas vías para el aprendizaje

1.2 JUSTIFICACIÓN

El aprendizaje del subneteo de redes IP, representa un desafío importante para estudiantes de carreras técnicas y de ingeniería, ya que involucra conceptos abstractos relacionados con clases de direcciones IP, máscaras de subred y lógica detrás del subneteo. Estos temas al ser técnicos y teóricos suelen generar confusión y desinterés entre los estudiantes, especialmente cuando se presentan de forma tradicional, mediante clases magistrales o ejercicios que no promueven la participación en estos procesos

En este contexto, surge la necesidad de implementar nuevas estrategias pedagógicas que faciliten la comprensión de estos contenidos y fomenten un aprendizaje significativo. Los juegos serios han demostrado ser herramientas eficaces para apoyar procesos educativos, ya que combinan el entretenimiento con objetivos didácticos concretos. Al integrar dinámicas interactivas y elementos lúdicos, los videojuegos permiten a los estudiantes aprender mediante la exploración, el ensayo, el error, y la resolución de problemas en entornos controlados y motivadores

Desarrollar un videojuego en 2D que busque enseñar conceptos de subneteo de manera práctica y entretenida responde a la necesidad mencionada. Este tipo de herramienta no solo promueve la comprensión conceptual, sino que también estimula la

1. Pedagógico/a: Adjetivo que significa “perteneciente o relativo a la ciencia de la enseñanza y aprendizaje

participación y la retención de conocimiento y la autonomía del estudiante, Además, al ser una solución tecnológica accesible y escalable se puede usar en diferentes contextos educativos, tanto como herramienta para poder enseñar de estos conceptos a nuevos alumnos, así como para aprenderlos de manera independiente.

1.3 METODOLOGÍA

I. Investigación documental

a) Se llevó a cabo una revisión exhaustiva de bibliografía en redes de computadoras, haciendo énfasis en los temas de direccionamiento IP, cálculo de subredes, máscaras, y prácticas de subneteo por medio del uso de libros de Redes.

b) Se consultaron diferentes fuentes de información como artículos científicos de google scholar u otros relacionados al tema de subneteo

c) Se investigaron diferentes conceptos dentro de juegos de varios géneros con el fin de estructurar mecánicas simples que promuevan o estimulen al estudiante dentro del proceso de aprendizaje.

II. Identificación de herramientas

Para el desarrollo del videojuego educativo se adoptó una metodología de desarrollo de software de tipo iterativa e incremental, la cual permitió construir el producto en diferentes fases. Cada nivel y minijuego fue diseñado, implementado y probado de manera independiente, lo que facilitó la detección temprana de errores y el ajuste progresivo de las mecánicas de juego y de la interfaz. Este enfoque, cercano a los principios

de metodologías ágiles, fue el más adecuado considerando que el proyecto requería constante experimentación con mecánicas y su funcionalidad

a) Para el desarrollo del juego se seleccionaron herramientas que brindarán una funcionalidad flexible y adecuada a los requerimientos del proyecto. Se optó por Unity[1] como motor de desarrollo, debido a su versatilidad al momento de crear videojuegos 2D, su compatibilidad multiplataforma y su gran comunidad de soporte, este funciona como el motor para el desarrollo del juego.

b) Visual Studio[2] fue seleccionada como el entorno de desarrollo integrado para la programación en C# lenguaje compatible con Unity. Esta herramienta permitió organizar, Esta herramienta permitió organizar, depurar y compilar los scripts encargados del movimiento del jugador, La activación de eventos, la validación de respuestas, el bloqueo del movimiento durante minijuegos y la gestión de la interfaz.

c) Para la creación de assets gráficos simples, como botones, plugs, etc, se utilizó Paint y herramientas básicas del programa para poder usarse. Estos recursos visuales fueron exportados en formato PNG y Optimizados para su integración directa en Unity, contribuyendo al diseño visual del juego sin necesidad de herramientas de diseño avanzadas.

1. Unity: Motor de videojuegos multiplataforma desarrollado por Unity Technologies
2. Visual Studio: Entorno de desarrollo integrado que permite crear apps para múltiples plataformas

III. Implementación y pruebas

La implementación del videojuego educativo se llevó a cabo siguiendo la metodología ágil previamente descrita, mediante iteraciones progresivas que permitieron integrar y validar cada funcionalidad de manera independiente

a) Se comenzó por el diseño de la escena principal, integrando un personaje jugable controlado con teclado, una cámara que lo sigue y zonas de interacción marcada en zonas entre otros.

b) Se implementan paneles informativos que muestran contenido teórico sobre subneteo, esto realizado por medio de objetos UI como TextMeshProGUI[1] e Image. El jugador puede dar click sobre el cartel respectivo y con ello muestra la información asignada al cartel con el fin de mostrar la teoría necesaria.

c) Se desarrolló un minijuego de conexiones, en el cual se debe unir los huecos de colores de un lado al otro dependiendo de las respuestas en base a la pregunta, el sistema verifica la respuesta para saber si la respuesta es correcta.

d) Se implementó un minijuego de Palanca donde el jugador deberá arrastrar una palanca a la posición correcta y luego presionar un botón para validar. Si la respuesta es correcta entonces se cierra la actividad.

e) Se implementó un sistema de preguntas interactivas reutilizables en diferentes escenas, donde se presentan preguntas de opción múltiple sobre subneteo. Igualmente habrá retroalimentación clara e inmediata siendo que si se falla se enviará al jugador al inicio del nivel mientras que si acierta la actividad desaparecerá.

1. TextMeshProUGUI: Componente del paquete TextMesh Pro de Unity usado para renderizar texto de alta calidad en la interfaz de usuario

IV. Evaluación y recomendaciones

a) Se realizaron pruebas internas durante el proceso de desarrollo para verificar el funcionamiento de cada una de las mecánicas del juego, como los paneles informativos, el minijuego de conexión de direcciones IP, el sistema de preguntas de opción múltiple y el minijuego de la palanca.

b) Se validó que las interfaces se activan correctamente al entrar en las zonas correspondientes del escenario, y que el sistema reaccionara de forma adecuada a las respuestas correctas e incorrectas.

1.4 OBJETIVO DE LA IMPLEMENTACIÓN

El objetivo de la implementación del videojuego educativo es desarrollar una herramienta interactiva que facilite el aprendizaje del subneteo de redes IP, mediante mecánicas simples y un entorno accesible, con el propósito de apoyar tanto la enseñanza formal por parte de docentes como el aprendizaje autónomo de los estudiantes.

Para complementar este objetivo, se plantearon los siguientes objetivos específicos:

a. Diseñar un entorno fácil de entender que permita a principiantes adquirir nociones básicas de subneteo de redes.

b. Incorporar mecánicas de juego sencillas que refuercen el aprendizaje sin generar distracciones.

c. Crear una experiencia educativa que pueda ser utilizada tanto en el aula como de manera autodidacta.

2 INTRODUCCIÓN A LA TEORÍA

2.1 Introducción a las redes

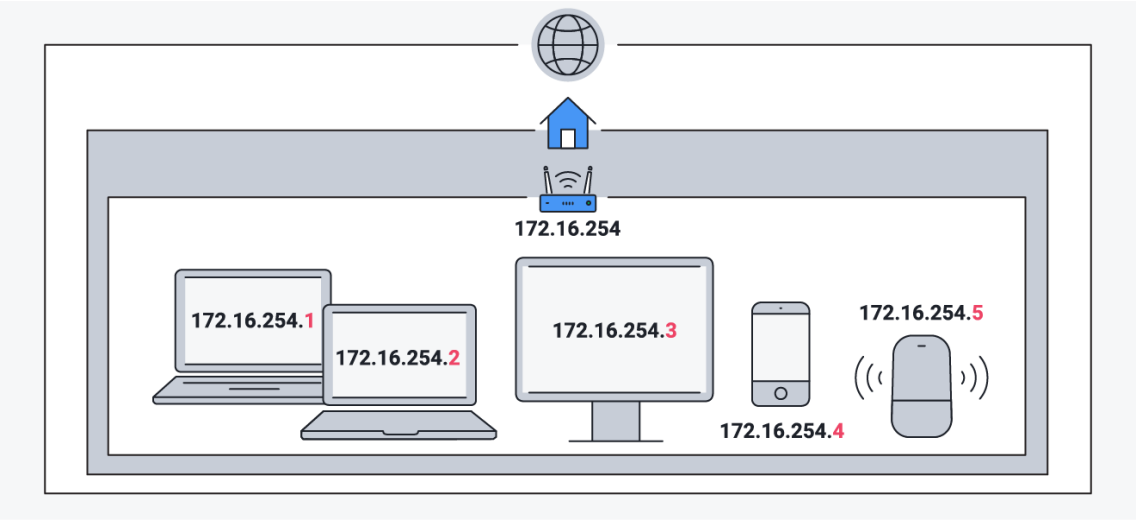
Los conceptos teóricos para el subneteo son las bases fundamentales estructuradas con el fin de dar un orden y entendimiento mucho mejor acerca de qué procesos se llevan a cabo, qué información es la que se busca obtener y otros relacionados. con eso dicho se puede proceder a entender los conceptos básicos teóricos como por ejemplo las direcciones IP

2.1.1 Direcciones IP

Una dirección IPv4 (IP Versión 4) es un identificador de 32 bits único asignado a cada interfaz de red en internet (Oviedo et al., 2018)(*¿Qué Es CIDR? - Explicación De Los Bloques Y La Notación CIDR - AWS*, n.d.). En la práctica se representa como cuatro octetos separados por puntos donde cada octeto que representa 8 bits es valorado entre 0 y 255. Estas direcciones son jerárquicas: parte de sus bits identifican la red y a la parte del host (dispositivos) dentro de la red. Sin estas direcciones IPv4 no sería posible enrutar paquetes de internet, por lo que son esenciales para la comunicación en redes informáticas. Aunque también existe el concepto de redes IPv6 que tienen las características de tener 128 bits, el IPv4 es el más comúnmente usado.

Figura 1

Representación de asignación de IPs en los dispositivos de un espacio definido



Nota: Tomado de Burdová, C. (2022), AVG. https://www.avg.com/es/signal/what-is-an-ip-address

2.1.2 Subredes y Subneteo

Una subred es un segmento o rango de direcciones lógicas que forma parte de una red mayor (*¿Qué Es Una Subred? | Cómo Funciona Una Subred | Cloudflare*, n.d.).En la práctica, subnetear significa dividir una red grande en varias redes más pequeñas para mejorar la eficiencia y gestión. Por ejemplo, en una empresa con miles de equipos conviene subdividir su red en subredes internas para reducir el tamaño de cada dominio de difusión (broadcast), limitar el tráfico local y delegar administración por departamentos. El subneteo consiste entonces en tomar un bloque de direcciones originalmente contiguas y crear dentro de él múltiples bloques (subredes), de modo que las IPs de una subred sean "locales" entre sí y "remotas" respecto a otras subredes (Oviedo et al., 2018). Al crear subredes se podría decir que se toman prestados bits adicionales de la porción de host de la dirección original para formar los identificadores de subred, lo que genera más subredes de menor tamaño, cada uno con su propio identificador de red.

Esta técnica permite, por ejemplo, ajustar el espacio de direcciones a las necesidades reales, crear una estructura jerárquica de red y controlar el flujo de paquetes entre subredes mediante routers.

2.1.3 Máscara de subred

La máscara de subred consiste en una secuencia de 32 bits que, aplicada a una dirección IPv4, determina qué parte corresponde al identificador de red (incluida la subred) y qué parte al identificador de host. En la notación binaria, todos los bits marcados con 1 definen la porción de red y los bits 0 la porción de host. Por ejemplo, una máscara de 24 bits consecutivos ("/24") en binario es 11111111.11111111.11111111.00000000, que en decimal resulta 255.255.255.0. En general, para calcular una máscara de subred se decide cuántos bits de red se necesitan, se pone ese número de unos binarios seguidos, y se completa con ceros para llegar a 32 bits. El resultado binario se convierte a decimal punto por punto. Por ejemplo, para 26 bits de red se tendría 11111111.11111111.11111111.11000000. El cálculo puede entenderse como "llenar con unos bits de red y con ceros los de hosts" La máscara también se expresa mediante el sufijo "/n" en la notación CIDR, que indica directamente el número de bits de red.

2.1.4 Dirección de Broadcast

La dirección de broadcast o de difusión de una subred es aquella en la que todos los bits de la porción de host valen 1 (Equipo editorial de IONOS, 2022).Corresponde al valor más alto de la subred. En ella, la máscara de subred puesta sobre la dirección convierte todos los bits de host a 1; esta dirección no se asigna a ningún dispositivo, sino que representa a todos los hosts de segmento. El propósito de la dirección de broadcast es permitir la transmisión multipunto: al enviar un paquete a esta dirección, el mensaje llega a todos los equipos (hosts) de esa red local. En otras palabras, es el identificador que facilita el envío simultáneo de datos a todos los nodos de la subred por ejemplo para

buscar servicios localmente. Ahí tanto la dirección de red como la de broadcast quedan reservadas y ya no pueden asignarse a los hosts.

2.1.5 Rango de direcciones de hosts

En cada subred, las direcciones asignables a dispositivos (hosts) son aquellas que quedan entre la dirección de red y la dirección de broadcast. Es decir, el primer host utilizable es la dirección de red más uno y el último es la de broadcast menos uno. Matemáticamente, si una subred tiene N bits para hosts, el total de direcciones posibles es 2 elevado a la N. Sin embargo, al restar 2 direcciones que están siempre reservadas. Por ejemplo, con 8 bits de host planteados, se elevaría 2 a la octava potencia dando un total de 256 direcciones, más a eso hay que restar 2 direcciones, quedando utilizables 254 direcciones útiles para hosts, Este rango de direcciones disponibles es clave para dimensionar el tamaño de la subred según el número de dispositivos necesarios.

2.1.6 Clases de IP

Ya habiendo visto estos conceptos altamente utilizados dentro del área del subneteo, Se puede comenzar a hablar de las diferentes clasificaciones que han existido para los diferentes rangos de IPs. Antes de la adopción CIDR, las direcciones IPV4 se organizaban en diferentes clases, un esquema que definía bloques de direcciones con tamaños fijos. Este método fue ampliamente utilizado durante las primeras etapas de desarrollo de internet y sirvió para el desarrollo de un método mejorado para el desarrollo de subneteo. Las clases eran:

Clase A: Comprende direcciones desde 0.0.0.0 hasta 127.255.255.255. Está pensada para redes extremadamente grandes, con 8 bits para red y 24 bits para host, lo que permite más de 16 millones de hosts por red.

Clase B: Abarca desde 128.0.0.0 hasta 191.255.255.255. Utiliza 16 bits para host y 16 para red permitiendo alrededor de 65,000 hosts por red.

Clase C: Va desde 192.0.0.0 hasta 223.255.255.255. Tiene 24 bits para la red y 8 para host, dejando la capacidad de 254 hosts por red. Este era el más común en oficinas o redes menores.

Clase D: Cubre desde 224.0.0.0 a 239.255.255.255 y está destinada exclusivamente para multicast o, en otras palabras, el envío de datos a múltiples equipos dentro de una misma red.

Clase E: Comprende desde 240.0.0.0 hasta 255.255.255.255. Estas direcciones están reservadas para investigaciones futuras y no se utilizan en redes públicas.

Figura 2

Tabla indicativa de clases de IP

Nota: Tomado de C. Ortiz (s.f.), Cecomart. https://cecomart.com/tipos-ip-que-son-ventajas-clases/

2.1.7 CIDR

El CIDR (Classless Inter-Domain Routing)es un esquema de direccionamiento introducido en 1993 con el propósito de solucionar las limitaciones del antiguo sistema de clases IP. Bajo el modelo clásico de redes tenían tamaños fijos y predefinidos según

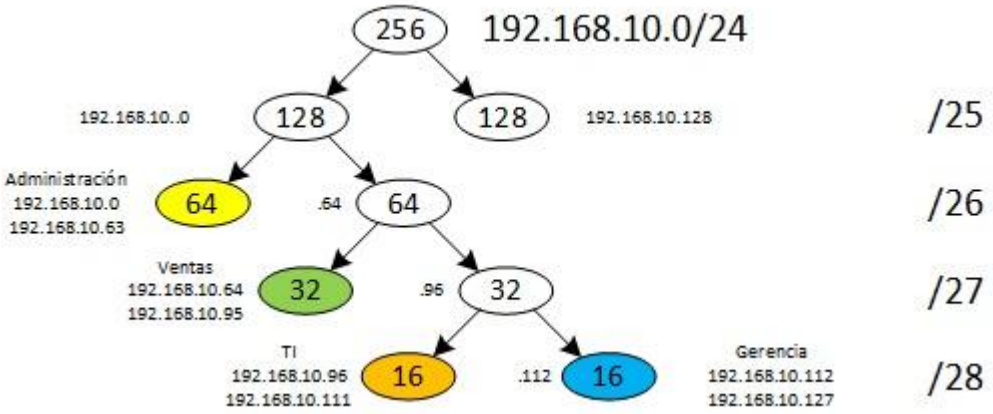
clase, lo que provocaba un gran desperdicio de direcciones. CIDR elimina la rigidez del modelo previamente establecido permitiendo una división flexible del espacio de direcciones IP. En lugar de depender de clases, CIDR emplea máscaras de subred de longitud variable, representadas en notación “/n” donde n indica el número de bits asignados a la parte de red. Por ejemplo, la red 192.168.1.0/28 indica que los primeros 28 bits son de red y los 4 restantes para hosts, permitiendo así subredes del tamaño exacto que una organización necesita.

2.1.8 Mapeo

El mapeo de redes hace referencia a la representación gráfica o esquemática del proceso de subdivisión de hosts dentro de una red principal, conforme se incrementa la cantidad de bits asignados a la parte de red en una dirección IP. Este proceso se basa en lógica binaria de base 2, donde cada bit adicional utilizado para identificar la red permite duplicar la cantidad de subredes posibles, pero a la vez reduce a la mitad el número de hosts disponibles por subred. Por ejemplo, como se puede observar en la figura 3 una máscara de subred /24 ofrece un bloque de 256 direcciones IP con 254 pueden ser asignadas a hosts. Si esta se divide usando una máscara /25, el resultado son dos subredes de 128 direcciones. Este proceso continúa sucesivamente con /26, /27 generando más subredes de menor tamaño, lo cual permite un uso más eficiente de distribución de redes según la necesidad

Figura 3

Imagen de referencia sobre mapeo



Nota. Elaboración propia.

3. DESARROLLO DEL ÁREA PRÁCTICA

La presente sección ejemplifica, desde una perspectiva académica y técnica, cómo resolver paso a paso un ejercicio práctico de subneteo. Este proceso es fundamental en la administración y diseño de redes IP., ya que permite dividir una red mayor en subredes más pequeñas, optimizando el uso de direcciones y asegurando el aislamiento y control de tráfico. Se desarrollará a profundidad la explicación del proceso de resolución de un ejercicio completo basado en un escenario realista, en donde diferentes departamentos dentro de una empresa requieren de un número de IPs.

3.1 Análisis Iniciales de Requerimientos

El escenario parte de la idea de que existen 4 departamentos en una empresa que necesitan asignación de IPs para sus trabajadores. Se tiene esta tabla como referencia.

Tabla 1

Planteamiento de necesidad de hosts por área

Área de la empresa	Cantidad Requerida
Administración	50
Ventas	25
TI	12
Gerencia	8

Según Forouzan(2007), el subneteo consiste en tomar una red mayor y dividirla en segmentos lógicos más pequeños, asignando rangos de direcciones IP que satisfagan las necesidades específicas de cada subred. Esto por medio de obtener los diferentes campos que nos ayudan a entender de mejor manera la separación y explicación de la separación de rangos de IP. Por ello el objetivo es obtener:

- Dirección de red
- Máscara CIDR
- Máscara decimal
- Rango de IPs válidas
- Dirección de broadcast
- Número de hosts utilizables

3.2 Cálculo de la máscara CIDR

Primero se deberá obtener la máscara CIDR por medio del uso de la fórmula:

$$Hosts\ vylidos = 2^h - 2$$

Donde h es la cantidad de bits asignados a la porción de host. Se resta 2 porque la primera dirección representa la red y la última es para el broadcast (Tanenbaum, 2011).

Se debe de intentar distintas potencias de 2 hasta tener una que contenga un número que contenga los hosts necesarios por parte de la sección que se esté realizando. En este caso siendo esta la sexta potencia pues quedarían como hosts válidos 62 IPs disponibles, es decir, una máscara de /26. Según la documentación de Cisco(2013), el prefijo CIDR define cuántos bits iniciales están fijos para la red, y por tanto cuántos están disponibles para hosts.

3.3 Conversión a máscara decimal.

Ya con la máscara CIDR el siguiente paso lógico es transformarlo a la notación decimal que es reconocida como el estándar para dividir una red /24 en 4 subredes iguales. Es por ello que se debe separar los octetos en base a la máscara anterior para poder traducirlo. Quedando de la siguiente manera: 11111111.11111111.11111111.11000000.

Como tal estos son los números de octetos que deben ser tomados en cuenta, cada número representa un número siendo el primero 128 y el último 1, cada que se avanza un número se debe dividir el valor que representa a la mitad y ese valor se suma para la interpretación, al ser /26 se

pone 2 números como 1, que serán los que se deben tomar en cuenta, en este caso para interpretar se debe sumar 128 más su mitad siendo 64 que daría 192. Esto quedaría como 255.255.255.192.

3.4 Cálculo de dirección broadcast

La dirección de broadcast es aquella en la que todos los bits de hosts se ponen en 1. Dado que la máscara es /26, quedan 6 bits para host, por lo tanto, el total de direcciones en la subred es de dos a la sexta potencia que da 64.

Si la IP base era 192.168.10.0 y se resta 64-1 = 63 se obtiene la dirección de broadcast que es 192.168.10.63.

La dirección de broadcast es siempre la última dentro del rango de la subred y sirve para enviar mensajes a todos los hosts de dicha red (Forouzan, 2007, p. 107).

3.5 Rango de direcciones IP válidas

El rango de IP válidas de direcciones IP es aquel que excluye la dirección de red y de broadcast. Por tanto:

- Primer host: 192.168.10.1
- Último host: 192.168.10.62

Esto ofrece exactamente 62 direcciones válidas, lo que satisface el requerimiento original. Según GeeksforGeeks (2021), el rango de IPs válidas se obtiene comenzando con la dirección de red +1 y terminando en broadcast -1.

“El rango de direcciones IP válidas en una subred excluye la dirección base y la dirección de difusión” (Forouzan, 2007, p. 150).

3.6 comprobación y reinicio del ciclo

Para culminar con esta sección de explicación se recurre a verificar que sean correctos los campos anotados, siendo que de haberse resuelto correctamente se deberá tener una tabla con los resultados de la imagen de la parte inferior.

Figura 4

Tabla semi resuelta de explicación de la primera fila

Departamento	Hosts Reales	Red	Máscara CIDR	Máscara Decimal	Rango IPs válidas	Broadcast
Administración	50	192.168.10.0	/26	255.255.255.192	192.168.10.1 - 192.168.10.62	192.168.10.63
Ventas	25	192.168.10.64	/27			
TI	12					
Gerencia	8					

Nota. Elaboración propia.

Una vez que se tiene dichos resultados se repite el proceso de resolución para cada uno de los campos de los siguientes departamentos, tomando en cuenta su número de IPs requeridas, máscaras CIDR, Máscaras decimales, Broadcasts e IPs válidas.

Figura 5

Tabla completamente resuelta de la explicación

Departamento	Hosts Reales	Red	Máscara CIDR	Máscara Decimal	Rango IPs válidas	Broadcast
Administración	50	192.168.10.0	/26	255.255.255.192	192.168.10.1 - 192.168.10.62	192.168.10.63
Ventas	25	192.168.10.64	/27	255.255.255.224	192.168.10.65 - 192.168.10.94	192.168.10.95
TI	12	192.168.10.96	/28	255.255.255.240	192.168.10.97 - 192.168.10.110	192.168.10.111
Gerencia	8	192.168.10.112	/28	255.255.255.240	192.168.10.113 - 192.168.10.126	192.168.10.127

Nota. Elaboración propia.

4. DESARROLLO DEL PRODUCTO

Como parte del trabajo descriptivo también es importante destacar el proceso de construcción del videojuego de educativo, pues muchos pueden pensar que es tan simple como comenzar a pensar en un nivel y empezar a desarrollar, sin embargo hay muchos factores que juegan un rol importante dentro del desarrollo, comenzando por las decisiones previas a comenzar con el desarrollo en sí, por ejemplo está el verificar el alcance de la idea, planear los niveles requeridos, investigar la profundidad del tema, buscar recursos, entre otros que se verán aquí.

4.1 Definición de Motor

El primer paso dentro del proceso de desarrollo del juego serio fue la fase de selección del motor de desarrollo que permitiera llevar a cabo de manera eficiente la propuesta tanto en su dimensión y complejidad técnica, así como educativa. Como tal esta sección tuvo especial relevancia gracias a que esta decisión movería todo el proyecto en una dirección u otra, definiría cuáles assets eran mejores de implementar, los recursos necesarios para la realización del proyecto, el tiempo estimado de desarrollo, etc. Tras una larga etapa de investigación y análisis comparativo de diversas herramientas de desarrollo disponibles en el mercado se optó por Unity como el entorno principal para construir el videojuego. Esta decisión respondió a múltiples factores que posicionan a Unity como una de las plataformas más robustas, accesibles y versátiles para la creación y diseño de juegos tanto en 2D como en 3D. Especialmente dentro de contextos educativos y de desarrollo independiente.

Unity se conoce por ser principalmente un motor de desarrollo de alto nivel multiplataforma ampliamente utilizado dentro de la industria de los videojuegos y en aplicaciones interactivas. Su entorno de trabajo integra herramientas de diseño visual, físicas, programación, control de interfaces y simulación, lo que permite construir desde prototipos simples, hasta productos comerciales de alto nivel desarrollados por equipos enteros de profesionales. Uno de los principales motivos para seleccionar Unity fue su amplia documentación, comunidad relativamente activa, un servicio de tienda de assets integrado que permite realizar mucho más fácilmente el trabajo, entre otras ventajas que ayudan a facilitar el aprendizaje autodidacta y la resolución de problemas técnicos a través de foros, tutoriales, guías oficiales. En proyectos de carácter académico, donde no siempre se cuenta con equipos multidisciplinarios extensos, este apoyo se transforma en un recurso mucho mejor.

Comparado con otros motores disponibles en el mercado, Unity presenta una serie de ventajas las cuales nos brindan mejores avances respecto al proyecto, en comparación a otros motores que no resultarían tan eficientes.

- Versatilidad en el desarrollo de 2D y 3D: A diferencia de otros motores como Unreal Engine, Unity permite un desarrollo fluido en juegos 2D, lo que es más adecuado para el enfoque educativo del presente proyecto. Su sistema de cámaras ortográficas, sprites y físicas adaptadas a entornos bidimensionales lo hacen ideal para experiencias visuales simples sin la necesidad de renderizado avanzado
- Lenguaje de programación C#: Unity emplea C# como lenguaje de scripting, un lenguaje moderno, estructurado y ampliamente utilizado en el ámbito profesional y académico. Esto facilitó la implementación de scripts personalizados para controlar la lógica del juego, como el movimiento del jugador, la activación de

interfaces o la validación de respuestas. En comparación, motores como Godot o Gamemaker utilizan un lenguaje menos estandarizado que limitan su alcance.

- Sistema UI altamente personalizable : La implementación de interfaces de usuario fue un eje central en el diseño del juego, y Unity ofrece herramientas específicas como el sistema Canvas, textos implementados por medio del TextMeshPro y un sistema de gestión de eventos para construir paneles interactivos, menús, botones, ventanas interactivas, etc. Otros motores como Godot también permiten la creación de interfaces, pero sus herramientas en general cuentan con más dificultad al momento de empezar con el desarrollo dentro del mismo.
- Integración con herramientas externas: Unity se complementa eficazmente con entornos de desarrollo como Visual Studio, lo cual facilita de gran manera la programación, la depuración y organización del código. Así mismo, permite importar recursos gráficos desde herramientas básicas como Paint, hasta programas para assets más complejos como blender, esto otorga flexibilidad a desarrolladores con todo tipo de niveles de experiencia en la creación de estos complementos.
- Licencia gratuita para proyectos pequeños: Para fines académicos y de investigación, Unity ofrece una versión gratuita para el público con acceso gratuito con todas sus funcionalidades, siempre que el proyecto no genere ingresos superiores a cierto límite que lo declara como un juego de categoría superior. Esta condición permitiría el desarrollo del juego sin limitaciones técnicas o de funcionalidad y sin la necesidad de un coste económico directo. Esto representa una ventaja frente a motores gráficos como Gamemaker que requiere licencias de pago para funcionalidad completa.

- Extensa biblioteca de Assets: Unity posee también una extensa biblioteca de assets llamada “Unity Asset Store” que ofrece una gran cantidad de recursos gráficos, sonidos, scripts, plantillas que aceleran el desarrollo. Esto resultó útil durante las primeras etapas de desarrollo donde se necesitaron extensamente assets visuales representativos de los conceptos u otros recursos necesarios para el desarrollo de la idea original.

4.2 Búsqueda de assets

Uno de los primeros pasos dentro del proceso del desarrollo del juego fue la identificación y selección de los recursos visuales o assets necesarios para la construcción del entorno gráfico del juego interactivo. Existieron múltiples factores que fueron claves al momento de pensar en cuál debería ser el estilo visual del juego, al tratarse de un proyecto educativo relacionado con redes de la informática, será fundamental tomar en cuenta que los elementos visuales que permitieran representar de forma comprensible componentes tecnológicos relacionados al tema o que busquen representar de forma visual los elementos usados.

Igualmente, al tratarse de un juego interactivo que tiene que si o si tener movimiento y ser entretenido de cierta forma, buscando atraer al jugador mientras se muestra entornos muy relacionados al mundo videojuego. Por ello es que se recurrió a diferentes librerías de recursos públicos, como Unity Asset Store, Itchi,o y OpenGameArt en busca de un estilo que pueda ser icónico.

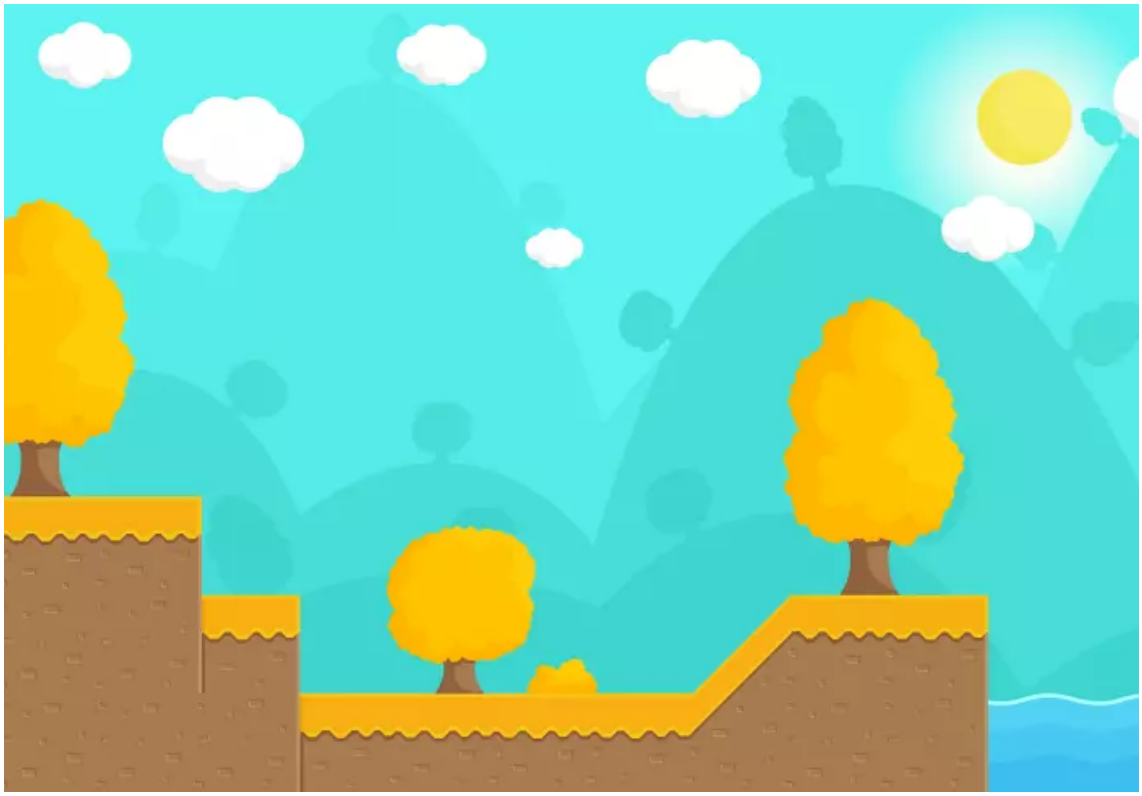
4.2.1 Estilo visual

Aquí se presentan algunos de los estilos explorados en el área conceptual y explicando mejor el posible uso que podrían tener.

4.2.1.1 Estilo Soft Drawings

Figura 6

Referencia de estilo visual soft drawing



Nota: Tomado de Bayat Games (s.f.), Unity Asset Store.

https://assetstore.unity.com/packages/2d/environments/free-platform-game-assets-85838

Este es un estilo visual claramente muy atractivo visualmente, se podría describir como un estilo altamente colorido, atractivo por su diseño familiar, cálido, semi curvilíneo y definido, sin embargo, este tiene múltiples carencias que lo terminaron descartando del primer boceto.

Para comenzar, si bien sus características previamente mencionadas han sido altamente atrayentes, este estilo puede resultar problemático por lo específico que es, si bien la imagen muestra en general lo atrayente del estilo, se nota que no tiene la más mínima coincidencia con la temática del proyecto, incluso aunque se han podido encontrar ciertos assets con este estilo

atrayente y que si son tecnológicos, estos suelen ser complicados de encontrar, aplicar y sobre todo de replicar, al buscar desarrollar un proyecto que no se base únicamente en uno o dos niveles, las opciones muy cerradas por la falta de recursos que mantengan un estilo en particular, esto más que nada se busca por no mezclar múltiples estilos que podrían llenar a confundir al usuario o en el mejor de los casos arruinaría la inmersión al proyecto al notarse de muy pobre calidad si no puede mantener siquiera un mismo estilo definido.

La falta de assets y dificultad de replicar en caso de buscar fabricar nuevos elementos no es el único inconveniente, sino también el enfoque para el que se puede sugerir, cómo se podrá ver este estilo si bien es bonito puede verse muy infantil si no se maneja claramente, algo que es una gran brecha pues el público al que busca atraer son personas jóvenes o adultas que busquen aprender conceptos que ya son un tanto más complejos, teniendo bases en la tecnología, por ello un estilo que puede verse demasiado colorido e infantil puede llegar a repeler al verdadero público objetivo al pensar que este proyecto no tiene un enfoque en ellos o que no va a presentar un enfoque teórico serio como una herramienta de introducción a estos conceptos.

4.2.1.2 Estilo 2.5 y 3D

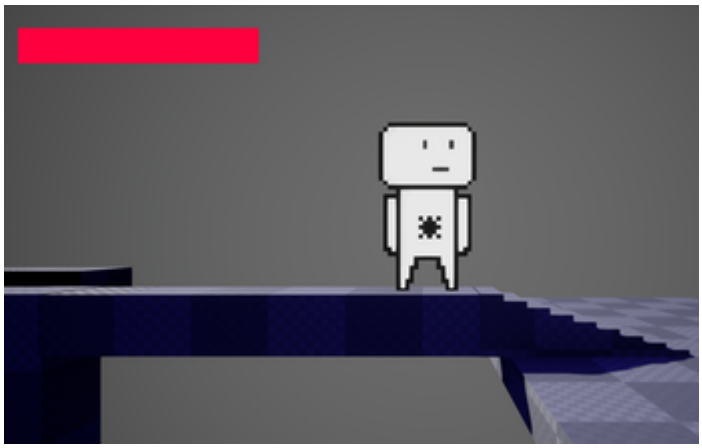
Figura 7

Referencia de estilo visual 3D

Nota: Tomado de LaFritz, J. (s.f.), Itch.io. https://ktmarine1999.itch.io/25d-platformer

Figura 8

Referencia de un estilo visual 2.5D



Nota: Tomado de Tibordanyi27 (s.f.), Itch.io. https://tibordanyi27.itch.io/collegeprototypemockup

Este es un estilo muy marcado y apreciado en la industria de videojuegos, como se puede observar en las imágenes, los juegos de tipo plataformero no siempre tienen que manejar assets en esta dimensión, siendo que también pueden manejarse con elementos del 3D, si se usa tanto assets en 2D como 3D en una escena, se considera un 2.5D, caso contrario es 3D. Este como se puede notar resulta relativamente curioso al mezclar conceptos que incluso llegan a ser vistosos si se usan correctamente. Aparte que el uso de los mismo no es tan complejo de replicar al poder utilizar herramientas como Blender para creación de assets en 3D modelados.

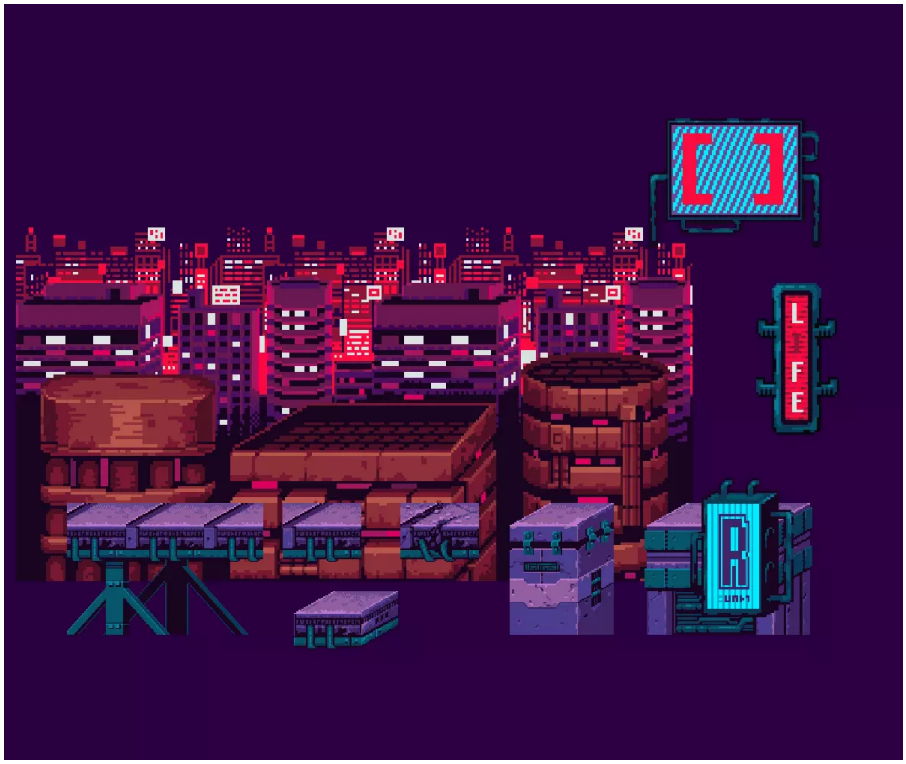
El problema que se encontró con el manejo de estos estilos es la experimentación que conlleva dicho estilo, con ello cabe aclarar que ninguno de los 2 es totalmente viable por el tema del tiempo manejado. Si usa el 3D puede ser un tanto complejo encontrar assets a bajo costo, y el fabricar elementos propios requiere cierto nivel de habilidad y tiempo para poder diseñarlos, si bien es posible, el tiempo es un factor que juega en contra por lo que el diseñar demasiados assets no sería posible, por ende, había que manejar un estilo marcado que tuviera suficientes assets usables para poder solo ciertos elementos como parte del trabajo artesanal a realizar por parte del desarrollador. Sin embargo, este estilo es altamente recomendable si se cuenta con equipos que se puedan distribuir mejor la carga o más tiempo de desarrollo.

El mismo argumento se puede utilizar para el estilo de diseño 2.5D, el manejo de elementos en 2d aumentaba un tanto la complejidad del proyecto, siendo que ahora no solo habría que dedicar cierto lapso de tiempo a buscar assets en 3D, sino también en 2D, algo que si bien daría un estilo marcado y bonito, no es recomendable para diseñadores primerizos que aún les cueste un tanto manejar el motor gráfico y las dificultades que conlleva la implementación de elementos de 2 dimensiones en entornos de 3 dimensiones y viceversa.

4.2.1.3 Estilo Píxel Art (Seleccionado):

Figura 9

Referencia de estilo píxel art



Nota: Tomado de LaFritz, J. (s.f.), Itch.io. https://ktmarine1999.itch.io/25d-platformer

Nota. Elaboración propia

El estilo visual píxel art es uno de los mejores estilos para múltiples enfoques y resulta particularmente útil para este proyecto de desarrollo por múltiples motivos que se alinean con todo lo previamente mencionado. Para comenzar , se debe mencionar el hecho de que el estilo píxel art es un gran referente dentro del mundo de los videojuegos, pues recuerda y homenajea a los primeros videojuegos diseñados en 8 y 16 bits. Con ello también se obtiene un factor importante que atrae al público objetivo, al mostrar un estilo tan claro que da a entender esa neutralidad en la edad que se busca. De igual manera las ventajas que se tiene son varias, por ejemplo, está la facilidad de encontrar assets de uso público de esta temática, al ser un estilo de arte tan común y expresivo, hay muchas personas que se dedican activamente a realizar esta clase

de assets con su talento, e incluso lo que no hay no es tan difícil de crear, siendo que puede ser un estilo altamente replicable pues incluso existen algunas herramientas especializadas en el arte píxel art. Por último, también resultaba fácil de manejar. Como tal, por ser un estilo marcado en 2D lo maneja todo dentro de un mismo formato que no es complicado siendo entendible con simples tutoriales.

4.2.2 Definición de personaje

El definir el estilo visual primero fue más que vital para poder comenzar a buscar lo necesario para el proyecto, como tal uno de los puntos vitales para el juego es el personaje que el jugador va a poder controlar, este es el medio que tiene el usuario de interactuar con el entorno que se haya diseñado, un pedazo del juego con especial relevancia pues también suele ser el personaje que guiará la aventura en cuestión. Sin embargo, en un serious game como el mío, este es un punto que queda muy reducido pues el enfoque no será conocer más de nuestro protagonista, sino poder aprender más respecto a un tema en específico, que en este caso son las subredes. Por ello el enfoque usado para buscar a un personaje principal se basa en la idea que el personaje pueda ser simple ya que no existe la necesidad de detalles exagerados u otros parecidos, de hecho, bajo esta filosofía se pensó que incluso los usuarios pueden sentirse más identificados de cierta forma, al ser un diseño simple, este no presenta género, edad, complexión, u otro rasgo, permitiendo sentir que están usando un avatar de sí mismos recorriendo los niveles.

Figura 10

Personaje Principal del juego

Nota: Tomado de Goldmetal (s.f.), Unity Asset Store.

https://assetstore.unity.com/packages/2d/characters/simple-2d-platformer-assets-pack-188518

4.2.3 Fondos y plataformas

Aunque muchos no tomen esto en cuenta la inmersión dentro de un juego depende de muchos factores, varios de estos son pequeños detalles los cuales muchos no toman en cuenta, sin embargo si estos no estuvieran sería muy evidente la falta de planeación y producción dentro del mismo, este es muchas veces el caso de los backgrounds y las plataformas, con esto en mente era importante el buscar assets adecuados para los entornos que buscaba retratar, siendo que en su mayoría se buscó assets relacionados a ciudades, pues estas son lo suficientemente tecnológicas para relacionarse al concepto de redes y modernidad que busco dar.

Figura 11

Referencia de assets en estilo píxel art

Nota: Tomado de Ansimuz (s.f.), Unity Asset Store.

https://assetstore.unity.com/packages/2d/environments/warped-city-assets-pack-138128

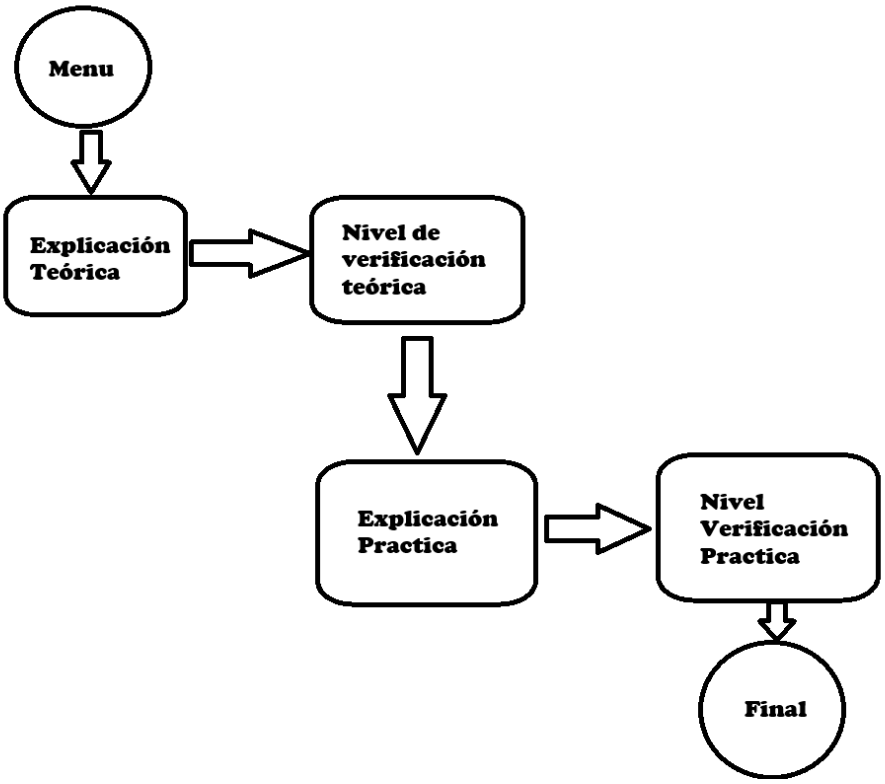
4.3 Diseño esquema de niveles

Una vez establecido los recursos visuales, se procedió al diseño del esquema de niveles, el cual dependiendo del nivel en el que se estuviera trabajando, se debía cumplir la función de ser una herramienta de aprendizaje o información, y en otros casos de niveles prácticos, debían de funcionar como método de testeo del aprendizaje del usuario y a la vez ser un tanto desafiantes

para mantener centrado al usuario. Para comenzar con este proceso se debió comenzar planeando el orden de los niveles.

Figura 12

Figura de referencia de estructura general del juego



Nota: Elaboración propia.

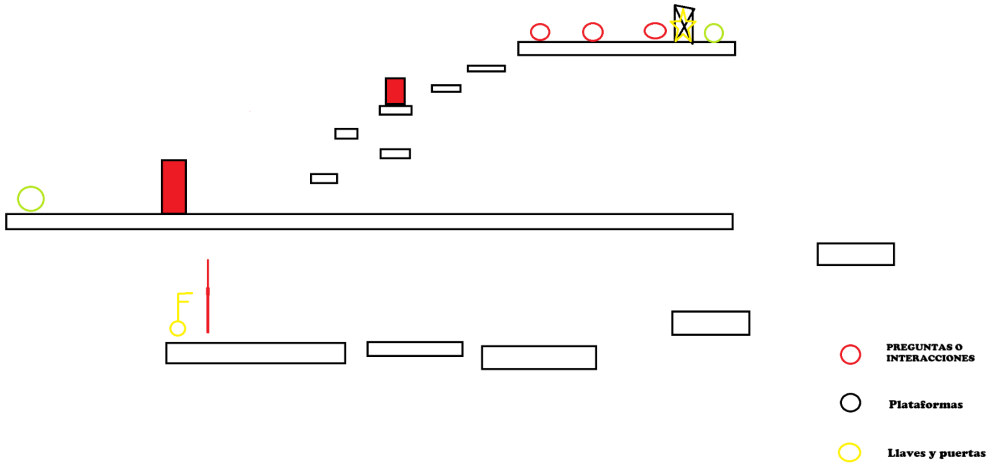
Como se puede observar en la imagen el enfoque estructurado que se decidió manejar para la creación del proyecto se sigue de una estructura simplificada, se comienza por un menú donde el jugador es recibido para poder comenzar el juego interactivo, entrando a un nivel entero donde se habla de la explicación teórica del subneteo de redes y sus conceptos, todos explorados a profundidad en el primer capítulo de este trabajo teórico, de ahí se pasa a un nivel de verificación práctica donde se observa por medio de preguntas a lo largo del nivel si los conceptos aprendidos en el anterior nivel están claros. Después este se deriva a un nuevo nivel donde el enfoque será la explicación de la parte práctica buscando explicar de mejor manera como se realiza la tabla manual de subneteo de redes, ayudando a mejorar el entendimiento de dichos temas, con ello se

va explicando de la mejor manera este tema, se procede después a un nuevo nivel que de la misma manera ayuda a poner a prueba el conocimiento adquirido, teniendo diferentes pruebas por pasar relacionadas a este contenido para llegar al final del juego donde se le felicita al usuario por haber completado exitosamente este pequeño curso introductorio al mundo de las redes.

Con ello se empezó el planteamiento de la idea para los niveles que terminarían:

Figura 13

Imagen referencial del diseño utilizado en niveles

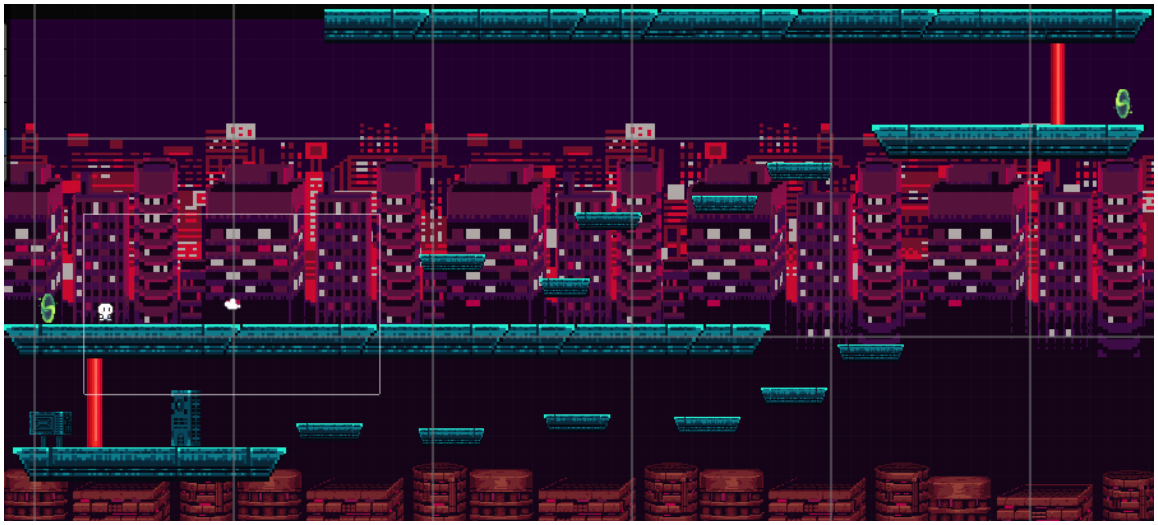


Nota: Elaboración propia.

El esquema visto en la imagen es uno de los modelos beta diseñados como referencia para la idea del diseño de niveles siendo este uno de los que sí llegó a la edición final del juego funcionando como la base de otro nivel.

Figura 14

Figura mostrando resultado de diseño de nivel referenciando bocetos



Nota: Elaboración propia.

Como tal este es un ejemplo de que la planificación previa llega a ser una herramienta clave al momento de optimizar el tiempo, enfocar las ideas y organizar bien el proyecto. Como tal se diría que, al poder tener estos bocetos, se llega a tener una mejor idea de lo que se desea, pudiendo llegar a mejorar estos bocetos al momento de ya estar creando el nivel, igualmente nos permite identificar errores, o cambios necesarios en el diseño.

4.4 Investigación de uso de unity y referencias de UIs

Al momento de comenzar con la implementación del videojuego serio en Unity, fue necesario adquirir conocimientos técnicos que permitieran utilizar correctamente el motor de desarrollo. Esta fase implicó una investigación autodidacta inmersiva en temas relacionados con la creación de paneles, posibles usos de UIs dentro del proyecto para el desarrollo de las interacciones con el entorno. Los principales temas abordados temas abordados incluyeron:

- Creación y manejo de escenas 2D
- Implementación de sistemas UI en escenas Interactivas
- Programación de interacciones mediante el uso de scripts y herramientas del motor

- Gestión de colisiones o detección de zonas activación
- Validación de respuestas y comprobación de tareas

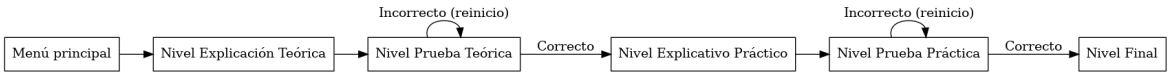
Se puede tomar la fase de investigación como una de las más vitales ya que permitió transformar múltiples ideas y elementos en conceptos útiles dentro del mismo entorno de Unity, generando componentes interactivos funcionales.

4.5 Diagramas relacionados a el producto

Con respecto al esquema general de los niveles y la estructura de navegación del videojuego educativo, se consideró pertinente aplicar herramientas de la ingeniería de software que permitan representar de manera clara y ordenada la secuencia de interacción. En este sentido, el modelo OOHDM (Object Oriented Hypermedia Design Method) facilita visualizar cómo se relacionan las diferentes escenas o niveles del juego —menú principal, nivel de explicación teórica, prueba teórica, explicación práctica, prueba práctica y nivel final—, ofreciendo un esquema general que refleja tanto la progresión lineal como las bifurcaciones en caso de respuestas correctas o incorrectas.

Figura 15

Figura de diagrama simplificado estilo OOHDM



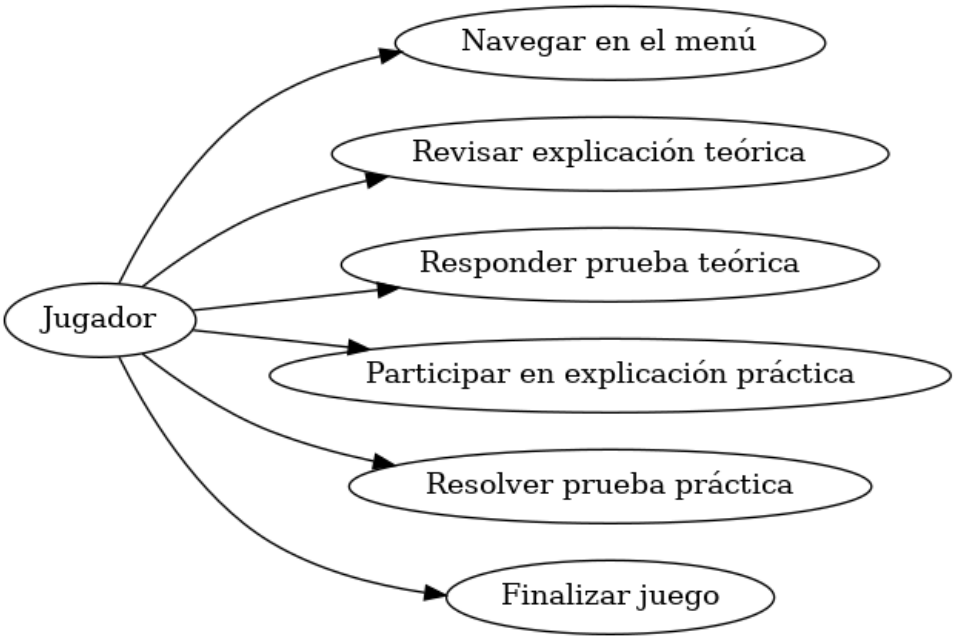
Nota: Elaboración propia.

De igual manera, el diagrama de casos de uso resulta fundamental para identificar las interacciones que el jugador puede realizar dentro del sistema, especificando las acciones principales que permiten avanzar en la experiencia, tales como navegar en el menú, revisar explicaciones, responder pruebas y completar el juego. Estos diagramas no solo ordenan la lógica interna del desarrollo, sino que también evidencian la correspondencia entre los objetivos pedagógicos y las mecánicas implementadas en el videojuego.

Si bien la parte de la investigación acerca del uso del motor y sus funciones resultó muy importante, lo mismo fue para la búsqueda de referencias e ideas potenciales para las interfaces dentro del juego. Como tal resultará obvio el pensar en la idea de que un juego tiene que tener al menos zonas didácticas y entretenidas que puedan llegar a justificar su existencia bajo la premisa de ser un juego y no una experiencia, por ello resultaba difícil el pensar en referencias para poder tomar al momento de diseñar mini juegos que busquen más que nada entretener al usuario mientras siguen siendo usadas para servir como herramientas de aprendizaje, mezclar ambos conceptos resultaba desafiante por lo que se buscó referencias en juegos modernos que pudieran acoplarse dentro de estas nuevas ideas.

Figura 16

Figura de diagrama de casos de uso



Nota: Elaboración propia.

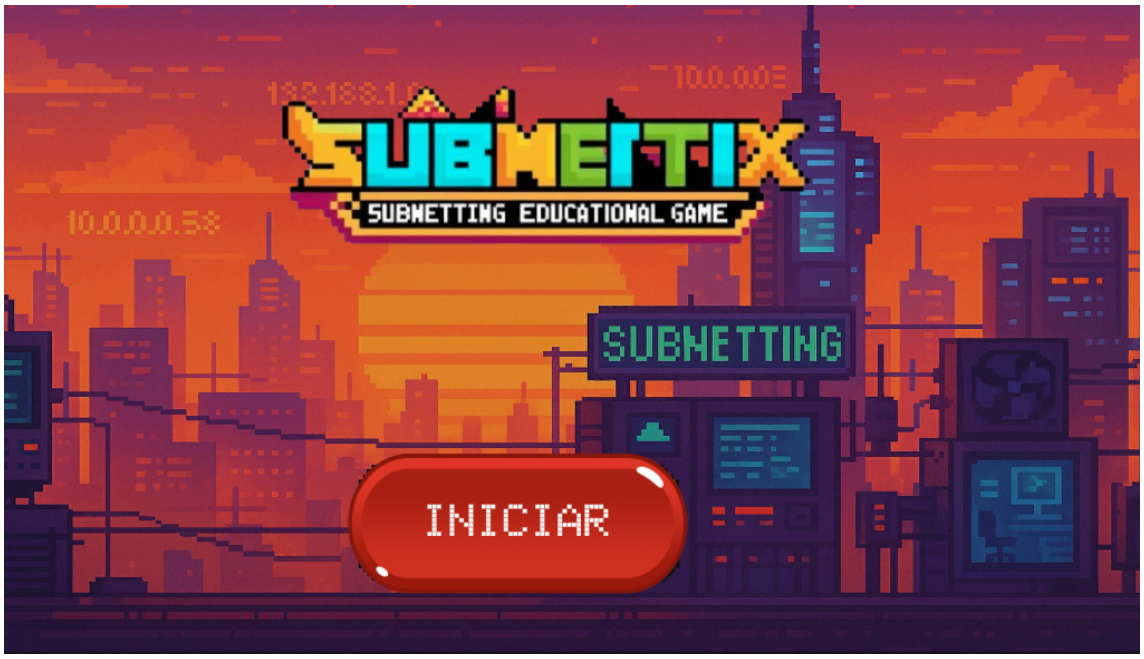
5. EXPLORACIÓN DEL PRODUCTO TERMINADO

5.1 Menú

El primer punto de interacción del jugador con el serious game es el menú principal el cual tiene la función de facilitar la introducción al usuario.

Figura 17

Interfaz del Menú

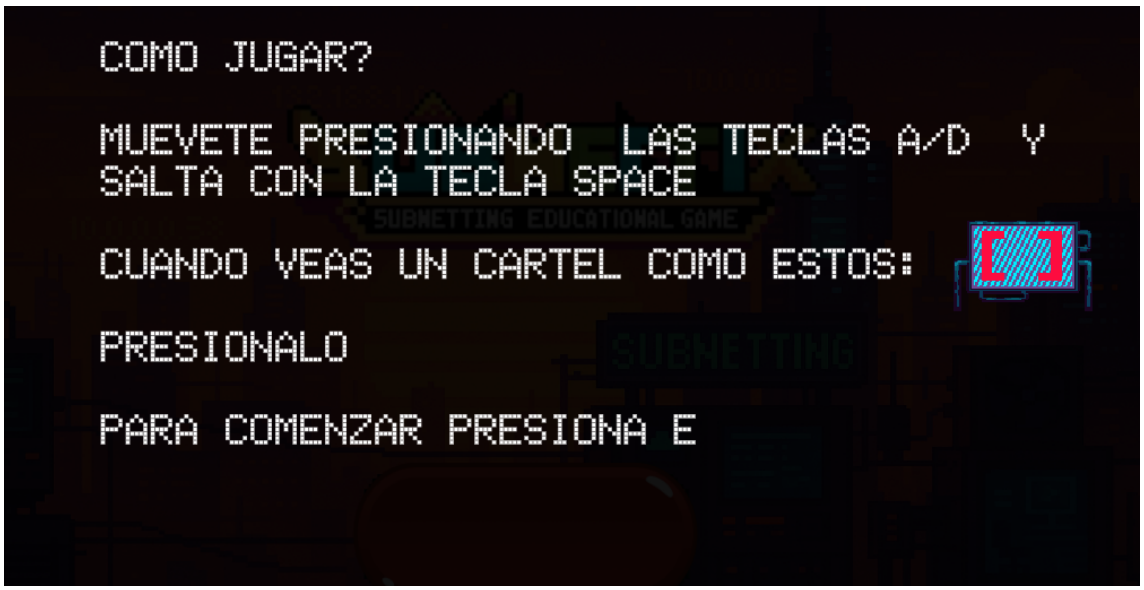


Nota: Elaboración propia.

Este consiste en un fondo estático con temática tecnológica, coherente con el contenido educativo del proyecto. En el centro de la pantalla se encuentra un único botón que permite al jugador iniciar el juego, pasando a una pantalla en negro la cual indica el funcionamiento básico del juego.

Figura 18

Pantalla de indicativo dentro del juego de cómo jugar



Nota: Elaboración propia.

Esta estructura simple facilita el acceso inmediato al contenido y evita distracciones, permitiendo un inicio rápido a la experiencia a jugadores de cualquier edad.

5.2 Nivel explicación teórica

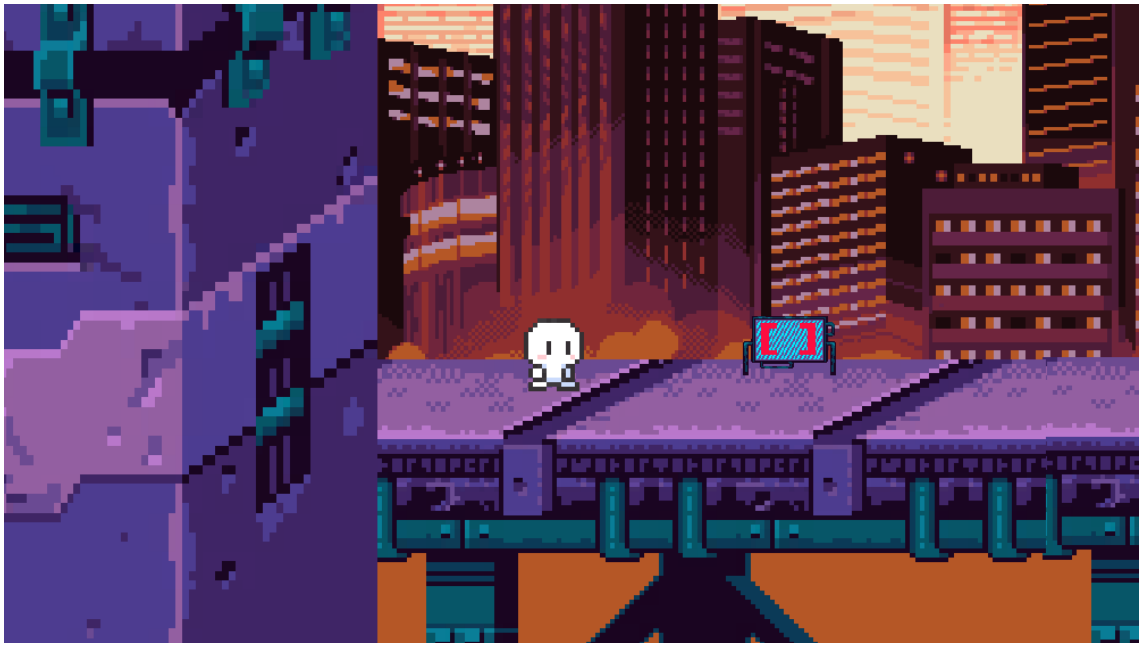
El segundo nivel del juego corresponde a la fase de explicación teórica, está diseñado específicamente para introducir al jugador en los fundamentos necesarios del subneteo de redes IP. En esta etapa no se presenta desafíos prácticos ni preguntas, pues se prioriza la exposición guiada del contenido teórico en un entorno controlado, sencillo y libre de distracciones.

La disposición del nivel sigue un camino recto y lineal que conduce al jugador de manera progresiva a través de una serie de carteles informativos interactivos. Esta estructura fue pensada para mantener el enfoque del estudiante evitando bifurcaciones o interrupciones

que generen confusión. El entorno visual es limpio y simple, con un fondo neutral y elementos estáticos que refuerzan la sensación de avance sin sobre estimular al usuario.

Figura 19

Imagen de referencia de nivel de explicación Teórica

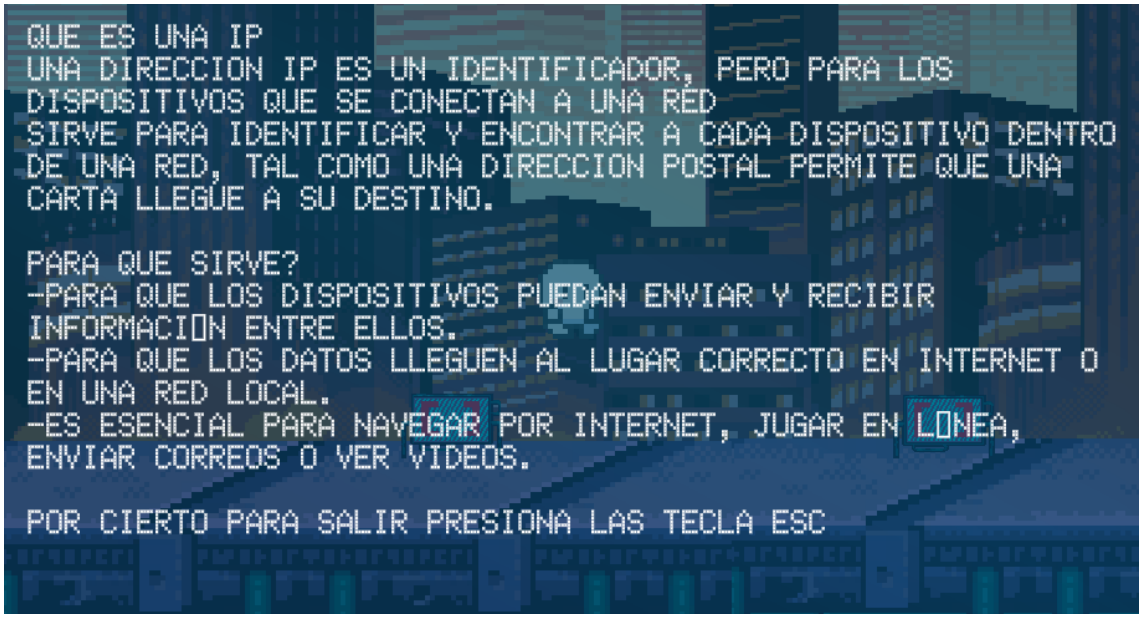


Nota: Elaboración propia.

Los carteles informativos cumplen una función central dentro de mí experiencia del serious game, pues cada uno de ellos presenta contenidos fundamentales relacionados al subneteo, como la dirección IP, las estructuras de las máscaras de red, el cálculo de rangos válidos, las direcciones de red y broadcast, entre otros. Esta información corresponde directamente a la teoría desarrollada previamente en el capítulo 2 de la tesis, que sirve como base conceptual para todo el juego. Al acercarse a un cartel, el jugador puede presionarlo para desplegar en la pantalla la información correspondiente al cartel.

Figura 20

Imagen de referencia de UIs dentro del área de explicación teórica



Nota: Elaboración propia.

El nivel termina cuando el jugador consigue alcanzar el portal de salida, ubicado al final del camino. Este portal actuará en todos los niveles como transición al siguiente nivel. Este busca reforzar la idea del jugador de progreso dentro del aprendizaje.

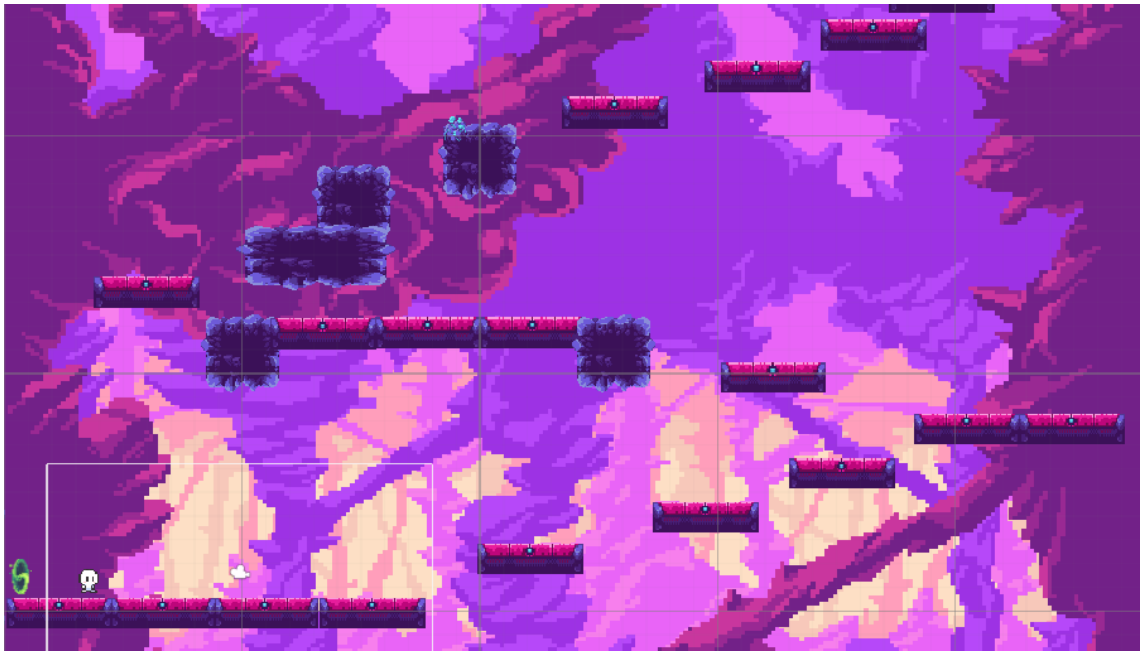
5.3 Nivel puesta a prueba teórica

Luego de completar el nivel de explicación teórica, el jugador accede a un entorno diseñado para evaluar la comprensión de los conceptos discutidos previamente. El objetivo de este nivel es permitir que el estudiante compruebe su nivel de entendimiento del tema en base a lo planteado por medio de preguntas de opción múltiple.

El escenario ahora sí es variado al ahora buscar resultar desafiante al jugador, en este caso se decidió tratarlo como un nivel ascendente , buscando un equilibrio entre los desafíos de plataformas y las respectivas preguntas que pueden realizarse a lo largo del nivel.

Figura 21

Imagen de referencia de nivel de puesta a prueba teórica



Nota: Elaboración propia.

Cada pregunta se desarrolla dentro de un panel visual donde el jugador puede elegir entre varias respuestas. Al seleccionar una respuesta el sistema da una retroalimentación inmediata, siendo que si acierta se le permite continuar, si es incorrecta el jugador es llevado atrás en el nivel, repitiendo el proceso como refuerzo a dicho conocimiento.

Este diseño promueve un modelo de ensayo y error guiado, donde el fallo no es penalizado de forma negativa, sino que se convierte en parte del proceso de aprendizaje activo. Además, se refuerzan los conceptos clave antes de avanzar a la siguiente etapa del juego.

5.4 Nivel explicación práctica

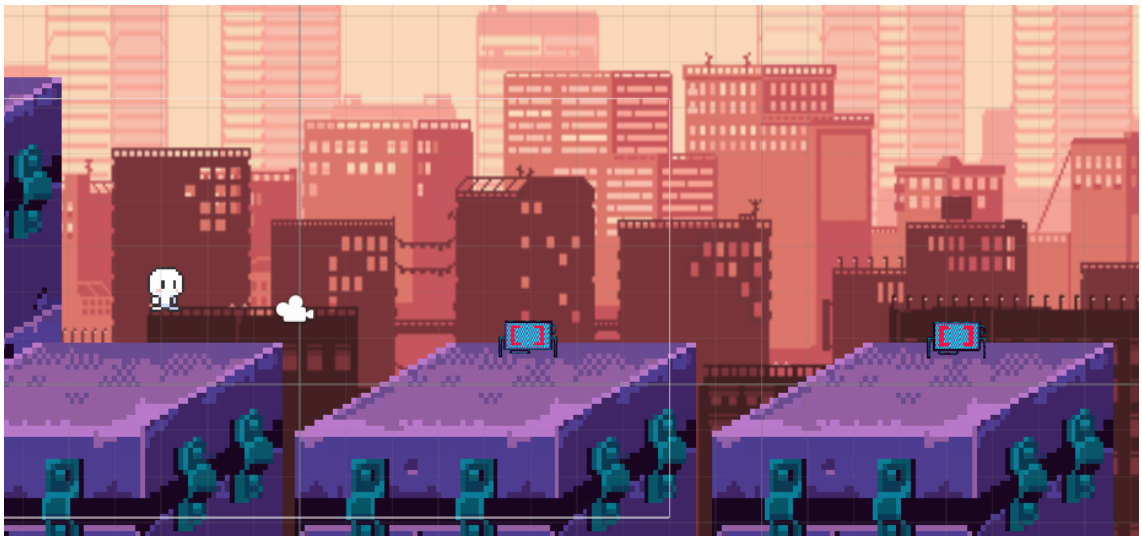
En esta fase del videojuego, el jugador accede al nivel de explicación práctica, donde los conceptos teóricos previamente estudiados comienzan a aplicarse de manera visual e interactiva. Este nivel representa la transición entre el conocimiento abstracto y su aplicación concreta,

brindando al estudiante un entorno donde puede observarse el proceso de resolución de ejercicios aplicados a entornos más reales dentro de un contexto dinámico.

A diferencia de los niveles anteriores, este escenario introduce pequeños desafíos de plataformas, que tienen como objetivo mantener la atención activa del jugador sin llegar a desviar el foco principal: el aprendizaje. Estos elementos lúdicos no interfieren con el contenido educativo, sino que funcionan como pausas breves en segmentos de explicación, permitiendo que el jugador se mantenga estimulado y comprometido.

Figura 22

Imagen de referencia de nivel de explicación práctica

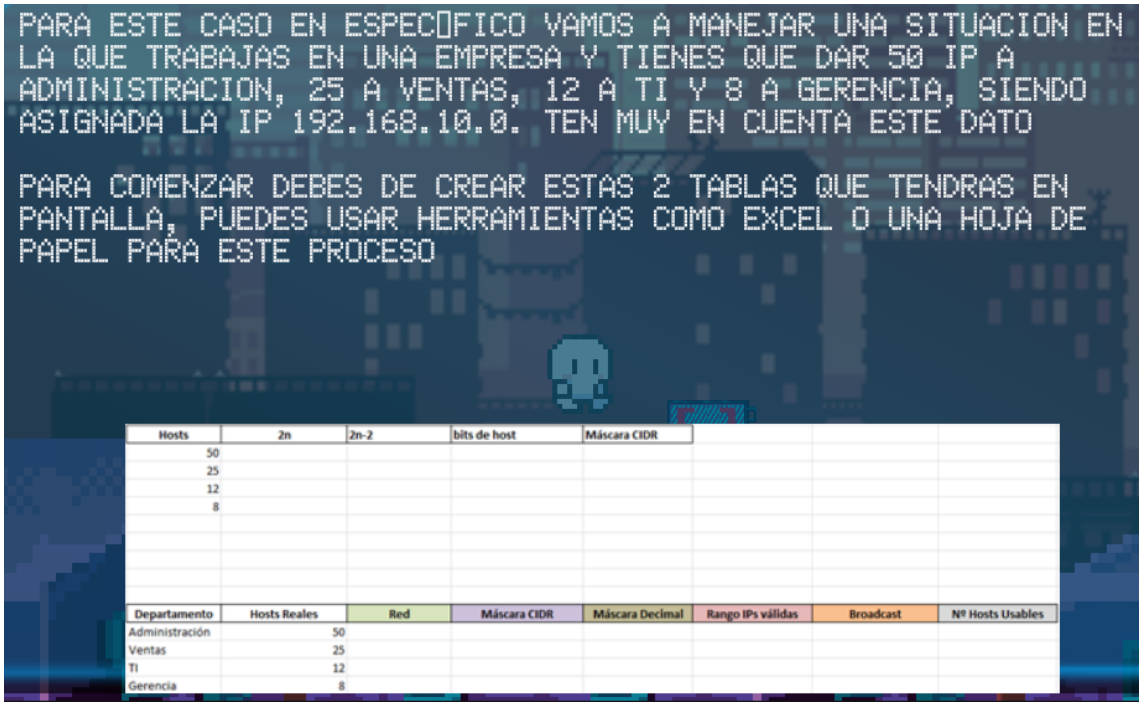


Nota: Elaboración propia.

Cada etapa del ejercicio es explicada visualmente mediante paneles de UI y acompañada de ilustraciones representativas, para facilitar la comprensión de los procesos involucrados. El jugador no resuelve directamente el ejercicio en esta etapa, sino que lo observa como una demostración guiada donde se invita al jugador a realizar un paso a paso buscando reforzar el aprendizaje y aclarar dudas antes de enfrentarse a un desafío más complejo.

Figura 23

Imagen de referencia de UIs dentro del nivel de explicación práctica.



Nota: Elaboración propia.

La estructura del nivel está diseñada para que el jugador avance dentro del mismo, leyendo los carteles en orden, permitiendo un mejor entendimiento. Este nivel cumple con el objetivo pedagógico de brindar un puente entre la teoría y la práctica, ayudando a entender el uso práctico de los conceptos en ejercicios y preparándolo adecuadamente para afrontar el reto de ejecutar dichos procedimientos por su cuenta.

5.5 Nivel puesta a prueba práctica

Este nivel representa el punto culminante del proceso de aprendizaje dentro del videojuego, en el cual el estudiante aplica de forma autónoma y activa los conocimientos adquiridos a través de ejercicios prácticos diseñados como minijuegos interactivos. Aquí se consolida la experiencia educativa mediante desafíos que simulan situaciones reales por medio de un ejercicio planteado.

El nivel inicia con una interfaz de usuario contextual (UI), que funciona como introducción a la tarea. En esta interfaz se describe brevemente un ejercicio, incluyendo una IP dada y un objetivo claro. Este contexto se brinda con el fin de que el jugador pueda resolver el ejercicio basándose en el proceso previamente revisado.

Una vez comprendido el contexto, el jugador avanza por el nivel y se encuentra con una serie de desafíos interactivos que representan diferentes formas de validar lo aprendido. Si bien se tiene también las clásicas preguntas de opción múltiple, se tiene también otras como:

- Minijuego de palanca: Una interfaz visual en la cual el jugador debe mover una palanca hacia una de las tres posiciones posibles, cada una mostrando una opción de respuesta en base a su respectiva pregunta. Tras posicionar la palanca, debe presionar un botón de validación. Sí la respuesta es correcta, el sistema la acepta y desbloquea el acceso al final del nivel, si no se solicita que lo intente nuevamente.
- Minijuego de conexión de cables: En esta mecánica el jugador debe emparejar visualmente componentes como direcciones IP con sus máscaras correctas o relacionar subredes con sus rangos válidos, arrastrando cables desde un panel de origen a uno de destino. Esta dinámica ayuda a reforzar la comprensión de la relación entre conceptos, al mismo tiempo que apoya el pensamiento visual y lógico.

Cada una de estas tareas ayuda al ser completadas a eliminar un bloqueo progresivo que impide acceder al portal del final. De esta forma el jugador entiende que solo resolviendo correctamente los desafíos puede avanzar, incentivando el repaso, la revisión de errores y el aprendizaje autónomo.

El nivel ha sido diseñado con una dificultad más elevada que los anteriores, no sólo en términos del contenido que se evalúa, sino también en la forma de interactuar con los elementos del juego. La combinación de distintas modalidades de interacción permite medir de forma más integral la comprensión del jugador, además de mantener su atención activa durante el proceso.

Una vez todas las actividades han sido superadas correctamente, el portal final se desbloquea, marcando el cierre del proceso de evaluación práctica y permitiendo al jugador avanzar hacia el nivel final. Este diseño refuerza la sensación de logro y la motivación intrínseca del estudiante, quien percibe claramente cómo cada acción correcta contribuye a su progreso dentro del juego.

5.6 Final

La última escena del videojuego cumple una función simbólica: cerrar la experiencia educativa de manera positiva, reforzando en el jugador la sensación de haber alcanzado un logro tras superar los distintos retos teóricos y prácticos.

Se presenta como una plataforma elevada, sencilla y sin desafíos, donde el jugador es recibido con una breve animación de fuegos artificiales que celebran la culminación del recorrido. No existen obstáculos ni elementos interactivos complejos, ya que el propósito de esta sección no es evaluar, sino reconocer el esfuerzo y el progreso realizado por el estudiante a lo largo del juego.

Figura 24

Celebración final por terminar la experiencia interactiva

Nota: Elaboración propia.

Tras unos segundos de celebración visual, aparece en la pantalla un mensaje final que felicita al jugador por haber completado con éxito la experiencia. Este texto destaca que ahora posee un conocimiento general sobre el subneteo de redes IP, gracias a su participación activa, la resolución de ejercicios y la superación de los minijuegos incluidos en el juego.

Una vez leído el mensaje, el jugador es redirigido automáticamente al menú principal del juego, pero con una diferencia visual importante: aparece una medalla o insignia junto al botón de inicio, indicando que el juego ha sido completado. Este cambio visual funciona como un sello de progreso que permanece visible en futuras sesiones y refuerza el sentido de logro personal. La medalla no solo representa una recompensa simbólica, sino también puede funcionar como un mecanismo de motivación para otros usuarios que inicien el juego por primera vez.

Figura 25

Referencia general del menú una vez se termina el juego serio

Nota: Elaboración propia.

Con esto se busca generar una sensación de satisfacción en el jugador, conclusión clara del proceso y un estímulo que refuerce la utilidad del aprendizaje autónomo mediante videojuegos educativos.

6. EXPLICACIÓN CÓDIGOS

Dentro del mismo juego contamos con múltiples códigos los cuales resultan muy importantes para el correcto funcionamiento del mismo, es por ello que se intentará explicar los realmente vitales para el programa.

6.1 Códigos generales

Dentro de esta sección se explicará aquellos códigos que sirven para el funcionamiento general y que están mayormente presentes en casi todo el producto.

6.1.1 Movimiento Jugador

Figura 26

Captura de script MovimientoJugador

```
public class MovimientoJugador : MonoBehaviour
{
    private Rigidbody2D rb2D;

    [Header("Movimiento")]
    private float movimientoHorizontal = 0f;
    [SerializeField] private float velocidadDeMovimiento;
    [Range(0, 0.3f)][SerializeField] private float suavizadoDeMovimiento;
    private Vector3 velocidad = Vector3.zero;
    private bool mirandoDerecha = true;

    [Header("Salto")]
    [SerializeField] private float fuerzaDeSalto;
    [SerializeField] private LayerMask queEsSuelo;
    [SerializeField] private Transform controladorSuelo;
    [SerializeField] private Vector3 dimensionesCaja;
    [SerializeField] private bool enSuelo;
    private bool salto = false;

    [Header("Animacion")]
    private Animator animator;

    private void Start()
    {
        rb2D = GetComponent<Rigidbody2D>();
        animator = GetComponent<Animator>();
    }

    private void Update()
    {
        movimientoHorizontal = Input.GetAxisRaw("Horizontal") * velocidadDeMovimiento;

        animator.SetFloat("Horizontal",Mathf.Abs(movimientoHorizontal));

        if (Input.GetButtonDown("Jump"))
        {
            salto = true;
        }
    }

    private void FixedUpdate()
```

Nota: Elaboración propia.

El script “Movimiento Jugador” controla el desplazamiento básico del personaje dentro del juego en un entorno 2D. Este componente gestiona tanto el movimiento horizontal como la capacidad de salto, utilizando un Rigidbody2D para aplicar física de manera semi-realista. Este incluye también detección de contacto con el suelo para validar cuando es posible saltar, y vincula estos comportamientos a los sprites del personaje del jugador.

6.1.2 Seguirjugador

El script está diseñado para controlar el comportamiento de la cámara principal del juego, haciendo que esta siga suavemente al jugador. Este mantiene una distancia definida por un offset que asegura una perspectiva adecuada en 2D. Además, incluye una función pública que permite activar o desactivar dinámicamente el seguimiento.

Figura 27

Captura de script Seguirjugador

```
public class Seguirjugador : MonoBehaviour
{
    public Transform player;
    public Vector3 offset = new Vector3(0, 0, -10);
    public float smoothSpeed = 0.125f;
    private bool followPlayer = true;

    void FixedUpdate()
    {
        if (followPlayer)
        {
            Vector3 desiredPosition = player.position + offset;
            Vector3 smoothedPosition = Vector3.Lerp(transform.position, desiredPosition, smoothSpeed);
            transform.position = smoothedPosition;
        }
    }
```

Nota: **Elaboración propia.**

6.1.3 Portal

El script permite la transición entre escenas dentro del juego, funcionando como un punto de entrada o salida que transporta al jugador a otra ubicación predefinida. Cuando el personaje entra en contacto con el área del portal, facilitando la navegación estructurada entre niveles mientras se usa una forma creativa para realizar esta función.

6.1.4 SpawnPoint

Se encarga de posicionar al jugador correctamente al cargar una nueva escena, verifica que coincida el identificador del punto de aparición con el destino guardado previamente en game objects. Utiliza PlayerPrefs para recuperar el identificador almacenado y de coincidir reposiciona al jugador en el punto designado.

6.2 Menú

6.2.1 Inicio

Gestiona la secuencia de entrada al juego, controlando la transición desde el menú principal hacia el primer nivel jugable. Al iniciar la escena, muestra un panel con botones e instrucciones básicas, activando una nueva pantalla informativa tras pulsar el botón. El avance hacia el juego se completa cuando el jugador presiona la tecla E. Este sistema asegura que el jugador reciba las instrucciones necesarias antes de comenzar, mejorando así la comprensión y accesibilidad a la experiencia.

Figura 28

Captura de script de Inicio

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

public class Inicio : MonoBehaviour
{
    [SerializeField] private string nombreEscena = "Teoria";
    [SerializeField] private GameObject panelMenuPrincipal;
    [SerializeField] private GameObject panelInstruccion;

    private bool puedeIniciar = false;

    private void Start()
    {
        panelMenuPrincipal.SetActive(true);
        panelInstruccion.SetActive(false);

        Button boton = GetComponent<Button>();
        if (boton != null)
        {
            boton.onClick.AddListener(() =>
            {
                panelMenuPrincipal.SetActive(false);
                panelInstruccion.SetActive(true);
                puedeIniciar = true;
            });
        }
    }

    private void Update()
    {
        if (puedeIniciar && Input.GetKeyDown(KeyCode.E))
        {
            SceneManager.LoadScene(nombreEscena);
        }
    }
}
```

Nota: Elaboración propia.

6.2.2 Mostrar Insignia

Controla la visibilidad de una insignia especial que simboliza la finalización del juego. Utiliza PlayerPrefs para verificar si el jugador ha desbloqueado dicha insignia y si proviene directamente de la escena final. En caso afirmativo, el objeto visual insignia se activa, otorgando una retroalimentación positiva al jugador por haber completado la experiencia.

6.3 Nivel Explicación Teórica

Desde este punto en adelante se va a hablar de los scripts incorporados en cada escena, evitando repetir los mencionados previamente por estar implícitamente en cada escenario del serious game.

6.3.1 ControladorCartel

Encargado de gestionar la visualización de mensajes informativos dentro del juego, utilizando la función “Mostrar Mensaje”, el texto es cargado dinámicamente en una interfaz mientras se activan los visuales necesarios. Del mismo modo la función “Ocultar Mensaje” permite cerrar el cartel y su fondo. Además, el script incluye una funcionalidad para cerrar el mensaje al presionar la tecla Escape.

Figura 29

Captura de script ControladorCartel.

```
[TextArea(4, 10)]
public string mensajeDelCartel;

private ControladorCartel cartelManager;
public GameObject indicadorVisual; // Asigna desde el Inspector
private bool jugadorCerca = false;

void Start()
{
    cartelManager = FindObjectOfType<ControladorCartel>();
    if (indicadorVisual != null)
        indicadorVisual.SetActive(false);
}

void Update()
{
    // Esto es opcional: permite mostrar el mensaje también al presionar E si estás cerca
    if (jugadorCerca && Input.GetKeyDown(KeyCode.E))
    {
        cartelManager.MostrarMensaje(mensajeDelCartel);
    }
}

void OnTriggerEnter2D(Collider2D other)
{
    if (other.CompareTag("Player"))
    {
        jugadorCerca = true;
        if (indicadorVisual != null)
            indicadorVisual.SetActive(true);
    }
```

Nota: Elaboración propia.

6.3.2 Cartel Interactivo

Permite que ciertos objetos dentro del juego funcionen como carteles informativos que el jugador puede activar. Al detectar la proximidad del jugador mediante colisiones, igualmente sirve para activar un indicador visual que sugiere la posibilidad de interacción. El mensaje asociado al cartel puede mostrarse al hacer click o presionar la tecla E. Este script funciona esencialmente junto al "Controlador Cartel" para el funcionamiento correcto de los carteles a lo largo del nivel.

6.4 Nivel Teórico

6.4.1 Controlador Decisiones

Este es el encargado de la gestión del sistema de preguntas interactivas dentro del juego, permitiendo presentar cuestionarios de opción múltiple. Al activarse, bloquea el

movimiento del jugador y muestra una interfaz con la pregunta y sus posibles respuestas. Si el jugador selecciona la opción correcta, se reactivan todas las funciones de movimiento, permitiendo al jugador continuar con el nivel. En caso de equivocarse, se activa se activa una secuencia visual con un panel fade y un mensaje que dice que el jugador debe volver a intentarlo mientras reposiciona al jugador en el punto de reinicio.

6.4.2 Zona Pregunta

Activa dinámicamente una pregunta interactiva cuando el jugador entra en una zona específica del escenario. Utiliza detección de colisión para identificar al jugador y, en ese momento, invoca el método “Mostrar Pregunta”, desplegando así la interfaz de evaluación.

Figura 30

Captura de script ZonaPregunta.

```
public class ZonaPregunta : MonoBehaviour
{
    public Controladordecisiones controlador;
    public PreguntaInteractiva pregunta;

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Player"))
        {
            controlador.MostrarPregunta(pregunta);
            gameObject.SetActive(false); // Evita repetir
        }
```

Nota: Elaboración propia.

6.4.3 Zona Caída

El script se encarga de reposicionar al jugador cuando este cae en una zona delimitada como fuera del límite del escenario o una zona de error. Al detectar una colisión con el jugador, su posición se restablece a un punto de reaparición. Esto ayuda al progreso y a la mejora de la experiencia, evitando que el jugador pueda llegar a trabarse en alguna parte. Como dato extra este código está presente en todas las escenas donde el jugador puede caerse hacia el vacío, sin embargo, no está presente en todos los niveles por lo que no puede ponerse como parte del área general.

Figura 31

Captura de script ZonaPregunta.

```
public class ZonaCaida : MonoBehaviour
{
    public Transform puntoRespawn;

    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Player"))
        {
            collision.transform.position = puntoRespawn.position;
        }
    }
}
```

Nota: Elaboración propia.

6.5 Nivel Explicación Práctica

6.5.1 Cartel Interactivo Avanzado

Para este nivel se ha tenido que modificar uno de los scripts anteriores, para poder ir alternando entre carteles interactivos, básicamente esta es una versión superior usada en este nivel para dar la información de una mejor manera al tener múltiples diapositivas rápidas. Funciona parecido al otro script en el sentido de que aún para acceder el jugador

necesita dar click en el cartel para desplegar la primera diapositiva y después para avanzar se hace uso de la tecla E.

6.5.2 Controlador Cartel Avanzado

Gestiona la visualización de carteles de múltiples diapositivas compuestas por texto e imágenes. Una vez que el jugador interactúa con un cartel, este controlador toma la lista de diapositivas asociadas y las presenta secuencialmente mediante una interfaz futurista. La navegación se realiza presionando la tecla E para avanzar o ESC para salir.

6.6 Nivel Práctico

6.6.1 Barrera Bloqueo

El script controla una barrera física dentro del juego que impide que el jugador avance ya sea hacia una tarea en específico o el final. Si el jugador intenta cruzar la barrera sin haber cumplido con la tarea asociada, se desplegará un panel informativo con un mensaje personalizado que explica el requisito pendiente. El sistema verifica constantemente si la condición ha sido cumplida, y en tal caso, destruye automáticamente la barrera permitiendo al jugador avanzar.

6.6.2 Estado Tareas

Actúa como un gestor global del estado de tareas clave dentro del juego. Este busca que haya múltiples instancias activas durante la ejecución que estén disponibles en cualquier momento. En su interior se guardan variables booleanas que indican si el jugador ha completado los desafíos planteados en el nivel. Este sistema centralizado permite condicionar el avance del jugador en distintas áreas.

6.6.3 Pantalla Nivel

Se encarga de mostrar una pantalla de introducción al nivel. Al comenzar la escena, despliega un panel con un mensaje definido previamente que se muestra con indicaciones hacia el nivel. La pantalla permanece activa hasta que el usuario presiona la tecla E, momento en el que se oculta el panel y se reanuda el tiempo normal del juego.

6.6.4 Aparición Palanca

El script se encarga de la activación de la interfaz de usuario del minijuego palanca cuando el jugador entra en una zona específica del mapa. Utiliza la función para detectar la colisión con el jugador y verificar si la tarea correspondiente aún no ha sido completada.

6.6.5 Gestor Palanca

Es el encargado de gestionar todo el flujo del minijuego relacionado con la palanca, desde su activación hasta su finalización o cancelación. Este componente controla la visibilidad de la interfaz del desafío, bloquea el movimiento del jugador y de la cámara mientras se realiza la actividad, y permite registrar si la tarea ha sido completada exitosamente.

6.6.6 Palanca Interactiva

Gestiona la funcionalidad de una palanca virtual arrastrable en la interfaz del juego, permitiendo al juego ubicarla en una de tres posiciones: arriba, medio, abajo.

Utiliza interfaces de eventos para detectar y responder ante el arrastre del mouse. Durante la interacción, compara la distancia vertical del puntero con zonas de referencia predefinidas para determinar en qué posición debe colocarse la palanca.

6.6.7 Palanca Visual

Se encarga de representar visualmente el estado actual de la palanca dentro del juego alternando entre sus tres sprites predefinidos. El script actualiza el componente Image en base al estado o posición actual de la palanca.

6.6.8 Verificador Palanca

Es responsable de evaluar si el jugador ha colocado correctamente una palanca interactiva en la posición adecuada, establecida mediante la variable posición correcta. Al presionar el botón de verificación, el sistema compara el estado actual de la palanca y la respuesta correcta. Si es correcto, se guarda el estado de la tarea como completada en Playerprefs y se reactiva el movimiento del jugador y la cámara. En caso de error, se muestra un mensaje de retroalimentación temporal y se reinicia al jugador en un punto de control predeterminado.

6.6.9 Aparición

Se encarga de activar la interfaz del minijuego de conexiones IP cuando el jugador entra en una zona determinada del escenario. Utilizando el evento OntriggerEnter2D, verifica si el jugador colisiona con el área y en caso de que no se haya completado antes, se despliega la UI correspondiente.

6.6.10 Activador Conexión

Permite detectar cuando el jugador hace click sobre un punto de conexión en el minijuego de cables IP. Al activarse mediante un evento, este script informa a otro componente que se ha seleccionado un punto de origen de la conexión.

6.6.11 Gestor Conexiones

El script Gestor Conexiones controla la lógica del minijuego de conexiones IP dentro del juego serio. Su principal función es registrar cuántas conexiones correctas ha realizado el jugador y, al alcanzar el número requerido, finalizar exitosamente la actividad. También lleva un control de las líneas creadas para poder eliminarlas si el jugador cancela la actividad, asegurando una experiencia limpia y reiniciable. Al finalizar el minijuego correctamente, se actualiza el estado de la tarea como completada y se restablece el movimiento y el flujo del juego, garantizando una progresión fluida y estructurada.

6.6.12 Línea Conexión

Gestiona la lógica del sistema de arrastre para crear conexiones visuales entre puntos de red en el minijuego de cables. Utiliza un Line Renderer para dibujar líneas en tiempo real desde un punto de origen hacia el cursor del jugador mientras se mantiene presionado el botón del mouse. Al soltar el click, verifica si el jugador ha conectado con el punto de destino correcto comparando los identificadores. Si la conexión es válida se registra como correcta y se mantiene visible la línea, de lo contrario, la conexión se elimina.

6.7 Final

6.7.1 Final

El script controla la secuencia de finalización del juego, activando una interfaz gráfica que felicita al jugador por completar la experiencia. Al detectar que el jugador

entra en la zona final, se desactiva el movimiento y se muestra un panel de cierre. Posteriormente al presionar la tecla E se marca en PlayerPrefs que se desbloquee una insignia en el menú. Finalmente se redirige al jugador a la escena principal, cerrando así el ciclo del juego interactivo.

7. CONCLUSIONES

El desarrollo e implementación de un videojuego serio en 2D enfocado en la enseñanza subneteo de redes IP permitió demostrar cómo las tecnologías interactivas pueden convertirse en una herramienta altamente eficaz para reforzar el aprendizaje de contenidos técnicos complejos. A lo largo del presente proyecto, se validó que los enfoques tradicionales de la enseñanza en áreas como redes informáticas suelen resultar poco motivadores para estudiantes, especialmente cuando el tutor busca implementar sus propios métodos, o los temas son presentados de forma abstracta o ejercicios poco guiados sin ninguna forma de saber si se está entendiendo correctamente. Frente a esta problemática, la presente propuesta se constituyó como una alternativa innovadora, práctica, accesible y lúdica que responde a características cognitivas y cultural de los estudiantes.

Se logró diseñar un entorno fácil de entender que permitió a los principiantes adquirir nociones básicas de subneteo de redes. Los carteles informativos dentro del escenario funcionaron como un puente entre la teoría y la práctica, facilitando conceptos abstractos como máscaras de subred, direcciones válidas y broadcast fueron interiorizados de manera progresiva. Esa estructura visual y guiada permite entender mejor los conceptos de subneteo, asegurando que el entorno no solo cumpliera una función lúdica, sino también formativa. En lo referente a la etapa de pruebas, el desarrollo se sustentó principalmente en un proceso iterativo de prueba y error, donde cada módulo del juego fue validado en términos de funcionalidad antes de integrarse al producto final. Estas pruebas consistieron en verificar el correcto comportamiento de los scripts de movimiento, la activación de eventos, la validación de respuestas y la interacción con los distintos elementos de la interfaz permitió ajustar de manera progresiva los errores detectados durante la implementación, consolidando un producto estable. Asimismo, el

proyecto demostró que es factible el desarrollo de soluciones educativas funcionales y de calidad utilizando software gratuito o de código abierto. Unity, en su versión gratuita, se consolidó como el eje central del desarrollo, permitiendo crear entornos interactivos con paneles informativos. A su vez, Visual Studio facilitó la programación de Scripts en C#, y editores gráficos básicos como Paint sirvieron para diseñar los elementos visuales esenciales del juego. Todo ello evidencia que las tecnologías interactivas pueden ser implementadas de manera eficaz, incluso con recursos limitados, siempre que exista una planificación pedagógica clara y un enfoque centrado en la experiencia del usuario.

Asimismo, se concluye que el uso de Unity como motor de desarrollo fue una decisión acertada, ya que permitió implementar sistemas personalizados de interacción, estructurar el flujo del juego con niveles jerárquicos y modularizados, y construir interfaces funcionales. Esta herramienta facilitó la programación de scripts reutilizables, el control de movimiento del jugador, la gestión y creación de las diferentes UIs y la creación de escenas que responden directamente a los objetivos pedagógicos del proyecto. En ese sentido, Unity demostró ser no solo una plataforma de desarrollo de videojuegos, sino también una solución eficaz para crear experiencias interactivas efectivas.

En términos del público objetivo, el juego responde adecuadamente a las necesidades formativas de carreras técnicas o tecnológicas, quienes requieren recursos educativos más prácticos, dinámicos y visuales para interiorizar conceptos como el cálculo de subredes, el uso de máscaras de red, la identificación de direcciones válidas o broadcast, entre otros. La estructura progresiva simple y la inclusión de carteles informativos dentro del entorno facilitan una transición gradual entre la teoría y la práctica, promoviendo el aprendizaje constructivo.

El uso de videojuegos en la educación técnica no debe considerarse una tendencia pasajera, sino una oportunidad de transformación pedagógica. Este proyecto ha

evidenciado que sí es posible realizar experiencias de aprendizaje útiles, motivadoras, baratas y centradas en temáticas que resultan fundamentales para los estudiantes, usando la lógica de los videojuegos como vehículos para enseñar competencias específicas, creando experiencias didácticas más ricas, participativas y acorde a las necesidades de las generaciones actuales.

8. RECOMENDACIONES

Acompañar el uso del juego con guías didácticas p materiales de apoyo que permitan al estudiante reforzar la comprensión del subneteo. Aunque el serious game fue diseñado para ser una herramienta autodidacta y autoexplicativa, su impacto puede ser aún mayor si se complementa con la orientación de un docente o la inclusión de manuales que ofrezcan ejemplos alternativos, aclaraciones teóricas o estrategias distintas para resolver los ejercicios. De esta manera, el entorno de aprendizaje mantiene su carácter accesible para principiantes, pero se enriquece con la posibilidad de recibir apoyo adicional que refuerce los contenidos.

Promover el uso de herramientas accesibles y gratuitas: Esté proyecto demostró como Unity, Visual Studio y editores básicos de gráficos, es posible desarrollar productos funcionales y pedagógicos. Esto permite desarrollar productos funcionales y pedagógicos. Esto permite que incluso instituciones con recursos limitados puedan adoptar tecnologías interactivas en sus procesos educativos.

Para una teórica versión 2.0 se podría implementar un conjunto de mejoras que eleven tanto la funcionalidad como la profundidad pedagógica del videojuego. Entre las principales propuestas se encuentra la inclusión de un sistema de retroalimentación inmediata que explique el porqué de los errores cometidos, una base de ejercicios dinámicos con más niveles de dificultad, así mismo poder generar ejercicios con IPs aleatorias, así mismo se podría mejorar para que en caso de ser utilizado por un docente como herramienta de soporte para enseñanza, se pueda personalizar el contenido, como paneles interactivos personalizados o ejercicios que el profesor plantee.

9. BIBLIOGRAFÍA

¿Qué es CIDR? - Explicación de los bloques y la notación CIDR - AWS. (n.d.). Amazon Web Services, Inc. https://aws.amazon.com/es/what-is/cidr/

Oviedo, Byron & Samaniego Mena, Eduardo & Oviedo, Jorge. (2018). Libro de Fundamentos de REDES. https://www.researchgate.net/publication/358105030_Libro_de_Fundamentos_de_REDES

Broadcast: definición y funcionamiento de la conexión multipunto. (2022, May 11). IONOS Digital Guide. https://www.ionos.com/es-us/digitalguide/servidores/know-how/broadcast/

Configure IP addresses and unique subnets for new users. (2025, January 24). Cisco. https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13788-3.html

Forouzan, B. A. (2007). Network layer: delivery, forwarding, and routing. McGraw-Hill Education - Access Engineering. https://www.accessengineeringlibrary.com/content/book/9780073250328/chapter/chapter22

GeeksforGeeks. (2025, January 16). How to calculate number of host in a subnet? GeeksforGeeks. https://www.geeksforgeeks.org/computer-networks/how-to-calculate-number-of-host-in-a-subnet/

Tanenbaum, A., & Wetherall, D. (2011). Redes de computadoras (5th ed.). Pearson Educación.

https://gc.scalahed.com/recursos/files/r161r/w25733w/redes_de_computadoras-freelibros-org.pdf

Bayat Games. (s.f.). Free Platform Game Assets [Recurso digital]. Unity Asset Store. https://assetstore.unity.com/packages/2d/environments/free-platform-game-assets-85838

Ortiz, C. (s.f.). ¿Qué son las direcciones IP, sus tipos, ventajas y clases? Cecomart. https://cecomart.com/tipos-ip-que-son-ventajas-clases/

LaFritz, J. (s.f.). 25D Platformer. Itch.io. https://ktmarine1999.itch.io/25d-platformer

Tibordanyi27. (s.f.). College Prototype Mockup. Itch.io. https://tibordanyi27.itch.io/college-prototypemockup

Goldmetal. (s.f.). Simple 2D Platformer Assets Pack. Unity Asset Store. https://assetstore.unity.com/packages/2d/characters/simple-2d-platformer-assets-pack-188518

Ansimuz. (s.f.). Warped City Assets Pack. Unity Asset Store. https://assetstore.unity.com/packages/2d/environments/warped-city-assets-pack-138128

Equipo editorial, Etecé. (2025, February 20). *Redes informáticas - Concepto, elementos y tipos*. Concepto. https://concepto.de/redes-informaticas/#:~:text=Las%20redes%20inform%C3%A1ticas%2C%20redes%20de,el%C3%A9ctricos%20o%20las%20ondas%20electromagn%C3%A9ticas

Cervantes, M. H. (n.d.). *Lic. en Informática*. https://repositorio-uapa.cuaed.unam.mx/repositorio/moodle/pluginfile.php/823/mod_resource/content/7/contenido/index.html#:~:text=Subneteo%20o%20subnet

Technologies, U. (n.d.). TextMesh Pro - Unity Manual. https://docs.unity3d.com/es/2019.4/Manual/com.unity.textmeshpro.html#:~:text=TextMesh%20Pro%20is%20the%20ultimate,and%20the%20legacy%20Text%20Mesh

Seo. (2025, 22 julio). ¿Qué es Unity y para qué sirve? Ejemplos | The Core School. *The Core*. https://www.thecoreschool.com/blog/que-es-unity/

7. ANEXO A

Capturas de imagen del producto terminado