ModeX Arcade

INTRODUCCIÓN

El presente documento expone de manera integral el desarrollo de un sistema arcade compuesto por cuatro videojuegos implementados mediante el lenguaje de programación Python, utilizando específicamente las librerías especializadas Arcade y Pygame. La industria del entretenimiento digital ha experimentado una evolución significativa en las últimas décadas, transformándose desde experiencias simples hasta complejos sistemas interactivos que demandan altos estándares de calidad técnica y diseño. En este contexto dinámico, surge la necesidad de herramientas de desarrollo que permitan crear experiencias interactivas sofisticadas sin comprometer la accesibilidad para desarrolladores en formación ni la calidad del producto final.

El proyecto se fundamenta en la identificación de una oportunidad clara: explorar y demostrar el potencial de los módulos especializados de Python para el desarrollo ágil y eficiente de videojuegos bidimensionales. A diferencia de los motores de desarrollo comerciales tradicionales que frecuentemente presentan curvas de aprendizaje pronunciadas y costos asociados considerables, Python ofrece un ecosistema de desarrollo accesible, bien documentado y con una comunidad activa que facilita la resolución de problemas técnicos.

La metodología RAD (Rapid Application Development) se adoptó como marco de trabajo fundamental para la gestión y ejecución del proyecto. Esta decisión metodológica responde directamente a las características específicas del desarrollo de videojuegos, donde la validación temprana de mecánicas, el prototipado rápido y la iteración constante son factores críticos para el éxito. La metodología RAD permitió establecer ciclos de desarrollo cortos con entregables funcionales en cada iteración, facilitando la validación continua de mecánicas de juego, la detección temprana de problemas técnicos y el ajuste ágil de características según los resultados de las pruebas. Este enfoque iterativo resultó fundamental para gestionar efectivamente la complejidad inherente al desarrollo simultáneo de múltiples títulos dentro de una plataforma unificada, manteniendo estándares consistentes de calidad mientras se optimizaban los tiempos de implementación y se reducían los errores asociados a la etapa de desarrollo.

PLANTEAMIENTO DEL PROBLEMA

El desarrollo de videojuegos se ha consolidado como una disciplina técnica compleja que requiere la convergencia de múltiples áreas de conocimiento especializadas. Esta complejidad inherente genera barreras de entrada significativas para desarrolladores en formación, estudiantes de programación y profesionales que desean incursionar en esta industria. La problemática se manifiesta en diversas dimensiones que afectan tanto el proceso de aprendizaje como la viabilidad de proyectos de desarrollo independiente.

En primer lugar, el desarrollo de videojuegos tradicionalmente demanda conocimientos avanzados en programación de sistemas, requiriendo comprensión profunda de conceptos como gestión de memoria, optimización de algoritmos, programación orientada a objetos, patrones de diseño y arquitectura de software. A esto se suma la necesidad de dominar técnicas de programación gráfica, incluyendo renderizado bidimensional y tridimensional, sistemas de animación, efectos visuales y optimización de rendimiento gráfico. El componente de física computacional añade otra capa de complejidad, requiriendo conocimientos de mecánica, detección y resolución de colisiones, simulación de movimiento y respuesta física. Finalmente, el diseño de interacción y experiencia de usuario demanda habilidades en diseño de interfaces, sistemas de retroalimentación y curvas de dificultad.

La problemática se agrava cuando se considera el panorama actual de herramientas disponibles para el desarrollo de videojuegos. Los motores de desarrollo profesionales, aunque extremadamente poderosos y utilizados ampliamente en la industria, presentan curvas de aprendizaje pronunciadas que pueden resultar intimidantes para desarrolladores novatos. Estos sistemas requieren la comprensión de ecosistemas complejos con múltiples herramientas integradas, editores especializados, sistemas de scripting propietarios y flujos de trabajo específicos.

El presente proyecto surge como una respuesta directa a esta problemática multifacética. Se plantea la exploración y validación de Python, específicamente mediante las librerías Arcade y Pygame, como una alternativa viable que puede ocupar el espacio intermedio entre herramientas excesivamente simples y sistemas profesionales complejos

JUSTIFICACIÓN

La justificación del presente proyecto se fundamenta en la convergencia de múltiples dimensiones que validan su pertinencia y relevancia. Desde la perspectiva técnica, Python se posiciona como uno de los lenguajes de programación más accesibles y simultáneamente más demandados en la industria tecnológica actual.

El desarrollo de videojuegos constituye un ejercicio integral que fortalece competencias fundamentales en programación orientada a objetos, pensamiento lógico y capacidades de resolución de problemas complejos. Los desafíos inherentes al desarrollo de videojuegos requieren análisis crítico, descomposición de problemas en componentes manejables, diseño de algoritmos eficientes y depuración sistemática, generando un producto comercialmente viable y técnicamente robusto.

El dominio de Python y las metodologías de desarrollo de videojuegos proporciona una base técnica sólida que permite escalar posteriormente a motores más complejos, ya que los conceptos fundamentales de bucles de juego, gestión de estados, sistemas de entidades y arquitectura de software son universales y transferibles. Las habilidades y arquitecturas implementadas son aplicables en contextos que trascienden el desarrollo de videojuegos de entretenimiento, incluyendo software empresarial, aplicaciones de visualización de datos, herramientas de simulación y soluciones interactivas personalizadas.

La adopción de la metodología RAD (Rapid Application Development) se justifica por las características específicas del desarrollo de videojuegos, donde la validación temprana y frecuente de mecánicas es crucial. La naturaleza altamente iterativa del diseño de juegos requiere ciclos rápidos de prototipado, prueba y refinamiento, facilitados mediante entregables funcionales frecuentes y flexibilidad para ajustar el enfoque según los objetivos del proyecto y las necesidades del mercado.

En síntesis, la justificación se sustenta en la convergencia de relevancia técnica actual, valor profesional integral, aplicabilidad práctica amplia y metodología apropiada para el dominio específico del desarrollo de videojuegos.

HIPÓTESIS

La hipótesis central del presente proyecto sostiene que es posible desarrollar un sistema arcade compuesto por múltiples videojuegos funcionales y técnicamente sólidos utilizando exclusivamente Python con las librerías Arcade y Pygame, constituyendo una plataforma viable para crear experiencias de calidad profesional en el dominio bidimensional sin recurrir a motores comerciales complejos.

Se hipotetiza que las librerías Arcade y Pygame proporcionan las capacidades técnicas necesarias para implementar mecánicas de juego complejas, incluyendo sistemas de física básica, detección de colisiones precisas, renderizado eficiente de gráficos bidimensionales y gestión fluida de entrada de usuario, permitiendo alcanzar un rendimiento satisfactorio en hardware de especificaciones medias.

La metodología RAD dará como resultado ciclos de desarrollo significativamente más cortos, permitiendo completar cada videojuego individual en períodos de 2-3 semanas. Su naturaleza iterativa facilitará la detección temprana de problemas de diseño o implementación, reduciendo costos de corrección en fases avanzadas. Los entregables funcionales frecuentes permitirán validación continua de mecánicas, resultando en productos finales más alineados con expectativas de jugabilidad y calidad técnica.

Finalmente, se postula que el proyecto es viable dentro de las restricciones de tiempo, recursos humanos y capacidades técnicas disponibles, permitiendo a un desarrollador individual o equipo pequeño completar el sistema arcade en 3-6 meses sin inversión financiera significativa, estableciendo un precedente replicable para futuros proyectos en contextos de recursos limitados.

OBJETIVO GENERAL

Desarrollar, implementar y validar un sistema arcade funcional compuesto por cuatro videojuegos bidimensionales integrados, utilizando Python como lenguaje de programación principal en combinación con las librerías especializadas Arcade y Pygame, aplicando principios fundamentales de diseño de videojuegos, programación orientada a objetos y técnicas de optimización de rendimiento, con el fin de crear experiencias interactivas entretenidas y de calidad que demuestren la viabilidad técnica de estas herramientas para el desarrollo de videojuegos de entretenimiento profesional.

OBJETIVOS SECUNDARIOS

- Crear cuatro videojuegos funcionales con mecánicas de juego diferenciadas que proporcionen experiencias de entretenimiento distintas y complementarias dentro de un sistema arcade unificado.

- Diseñar e implementar una arquitectura de software modular, escalable y mantenible que facilite la extensión futura del sistema, aplicando patrones de diseño apropiados que promuevan la reutilización de código.

- Optimizar el rendimiento del sistema para lograr ejecución fluida en hardware de especificaciones medias mediante técnicas eficientes de gestión de recursos.

- Desarrollar una interfaz de usuario cohesiva e intuitiva que integre los cuatro juegos con sistemas de navegación, feedback visual y auditivo que mejoren la experiencia del jugador.

En conclusión, el proyecto busca desarrollar un sistema arcade de cuatro videojuegos que demuestre la viabilidad de Python para el desarrollo profesional. Los objetivos establecen un marco integral que abarca arquitectura modular, optimización de rendimiento y validación metodológica, convergiendo en un producto funcional que consolida a Python como un lenguaje útil para videojuegos comerciales.

.

PROPUESTA DE VALOR

La definición de la propuesta de valor resulta esencial para articular claramente qué beneficios específicos aporta el proyecto y cómo se diferencia de alternativas existentes en el ámbito del desarrollo de videojuegos. Esta sección identifica las ventajas competitivas, los beneficios únicos y las contribuciones distintivas que el proyecto ofrece, estableciendo el fundamento estratégico que justifica su desarrollo y ejecución.

Desde la perspectiva del desarrollo de videojuegos, el proyecto demuestra que es posible crear experiencias de entretenimiento de calidad utilizando herramientas accesibles y de código abierto, eliminando barreras técnicas y económicas que tradicionalmente han limitado el acceso al desarrollo de juegos. Esta demostración práctica valida que no es necesario recurrir a motores comerciales complejos o costosos para producir videojuegos funcionales, entretenidos y técnicamente sólidos, abriendo posibilidades para desarrolladores independientes, pequeños estudios, y entusiastas que desean crear sus propios juegos sin inversión financiera significativa.

Desde el punto de vista del entretenimiento, el proyecto ofrece a los usuarios finales diversión accesible mediante cuatro juegos con mecánicas diferenciadas y experiencias de juego variadas, presentados en una interfaz cohesiva que evoca la nostalgia de arcades clásicos mientras incorpora estándares modernos de jugabilidad y usabilidad.

Los requisitos técnicos mínimos del sistema permiten que jugadores con hardware de gama media o inferior puedan disfrutar de los juegos sin necesidad de equipos especializados, democratizando el acceso al entretenimiento digital. La variedad de géneros integrados en un solo sistema proporciona opciones para diferentes preferencias de juego, desde acción rápida hasta desafíos de lógica, maximizando el valor de entretenimiento para audiencias diversas.

En conclusión, con este modelo de desarrollo de bajo costo demostrado valida aproximaciones sostenibles que pueden replicarse en diversos contextos, particularmente relevante para desarrolladores independientes que buscan crear juegos sin comprometer recursos financieros significativos antes de validar viabilidad comercial de sus ideas.

IMPACTO SOCIAL, TECNOLÓGICO Y/O DESARROLLO SUSTENTABLE

El proyecto genera impacto en múltiples dimensiones que trascienden el producto técnico inmediato, contribuyendo al ecosistema tecnológico y social de manera significativa.

Desde la perspectiva social, el proyecto democratiza el acceso al desarrollo de videojuegos al demostrar que es posible crear experiencias de entretenimiento de calidad sin barreras económicas significativas. Esta democratización beneficia especialmente a desarrolladores en países en desarrollo, estudiantes y entusiastas que no tienen acceso a herramientas comerciales costosas. Los videojuegos resultantes son accesibles para usuarios con hardware modesto, ampliando el acceso al entretenimiento digital.

El impacto tecnológico se manifiesta en la validación práctica de herramientas de código abierto para desarrollo profesional de videojuegos. El proyecto demuestra que Python puede competir efectivamente con lenguajes tradicionalmente asociados al desarrollo de juegos como C++ o C#, al menos en el dominio bidimensional. Las soluciones técnicas implementadas y documentadas contribuyen al conocimiento colectivo de la comunidad de desarrolladores, facilitando futuros proyectos similares.

Desde la perspectiva de desarrollo sustentable, el proyecto promueve un modelo de desarrollo de software sostenible basado en herramientas de código abierto que no generan dependencias comerciales ni costos recurrentes de licenciamiento. La arquitectura modular y bien documentada facilita mantenimiento y extensión a largo plazo sin obsolescencia programada. El uso eficiente de recursos computacionales permite ejecución en hardware existente sin promover ciclos innecesarios de actualización de equipos.

En conclusión, el impacto integral del proyecto se extiende más allá de sus objetivos técnicos inmediatos, contribuyendo a la democratización del desarrollo de software, el fortalecimiento del ecosistema de código abierto y la promoción de modelos de desarrollo sustentables que no dependen de recursos económicos significativos ni generan dependencias tecnológicas limitantes.

MANUAL DE USUARIO

ÍNDICE

1. INTRODUCCIÓN

1.1 Propósito del manual

El presente manual tiene como finalidad proporcionar a los usuarios una guía completa y detallada que permita comprender el funcionamiento integral de la aplicación y de los cuatro juegos que la conforman. Asimismo, busca facilitar la instalación, configuración, uso, solución de problemas y aprovechamiento máximo de la plataforma de entretenimiento desarrollada en Pygame.

Este documento está diseñado tanto para usuarios finales como para evaluadores, docentes y desarrolladores que deseen conocer el funcionamiento interno y externo de la aplicación.

1.2 Público objetivo

Este manual está dirigido a:

- Usuarios que desean interactuar con la aplicación recreativa.
- Estudiantes que requieren comprender su funcionamiento.
- Personal docente o evaluador que analiza el proyecto.
- Desarrolladores que deseen mantener o actualizar el software.

No se requiere experiencia técnica avanzada para operar la aplicación.

1.3 Descripción General de la Aplicación

La aplicación es una plataforma interactiva que reúne cuatro minijuegos independientes en un solo entorno gráfico. Todos los juegos han sido desarrollados con la librería Pygame de Python e integrados mediante un menú principal que permite iniciar cada uno de ellos de manera simple y rápida.

La interfaz ofrece navegación intuitiva, elementos visuales claros, controles accesibles y un diseño adaptable a cualquier usuario, independientemente de su experiencia en videojuegos.

1.4 Características Principales

- Cuatro juegos completamente funcionales dentro de un mismo entorno.
- Interfaz general sencilla y organizada.
- Gráficos optimizados y animaciones fluidas.
- Controles accesibles por teclado o mouse, según el diseño de cada juego.
- Código modular que facilita futuras expansiones.
- Sistema de puntuación independiente por juego.
- Posibilidad de ejecución desde Visual Studio Code o mediante un archivo ejecutable.

1.5 Tecnologías Utilizadas

- Pygame: biblioteca para gráficos, eventos, audio y renderizado.
- Visual Studio: entorno de desarrollo para edición, pruebas y ejecución.
- Paquetes adicionales: sys, os, random y librerías propias.

---

2. REQUISITOS DEL SISTEMA

2.1 Requisitos mínimos de hardware

- Procesador Dual Core 2.0 GHz o superior.
- 4 GB de memoria RAM.
- Tarjeta gráfica con soporte para aceleración 2D.
- 500 MB de espacio libre.

2.2 Requisitos de software (Python, librerías, versión de Pygame)

- Python 3.10 o superior.
- Pygame versión 2.1 o mayor.
- Windows 10 o superior.

2.3 Dependencias incluidas o necesarias

- Librerías estándar de Python.
- Archivos multimedia (imágenes, sonidos y sprites).
- Pygame (instalación obligatoria).

---

3. INSTALACIÓN Y EJECUCIÓN

3.1 Instalación de Python

1. Visitar https://python.org.
2. Descargar la versión más reciente.
3. Marcar Add to PATH antes de instalar.
4. Completar la instalación.

3.2 Instalación de Pygame

Usar el siguiente comando en la terminal: pip install pygame

3.3 Cómo abrir el Proyecto en Visual Studio

1. Abrir Visual Studio.
2. Seleccionar Open Folder.
3. Ubicar la carpeta del proyecto.
4. Cargar archivos .py del juego.

3.4 Ejecución desde Visual Studio

- Presionar Ctrl + F5 para ejecutar sin depuración.
- Revisar la consola para errores.

3.6 Solución de errores comunes al iniciar

- Error: Pygame no encontrado: reinstalar pygame.
- Pantalla negra: revisar rutas de imágenes.
- Cierre inmediato: ejecutar desde consola para ver errores.

---

4. INTERFAZ GENERAL DE LA APLICACIÓN

4.1 Pantalla de inicio

Muestra el nombre de la aplicación y una representación gráfica de los videojuegos para avanzar y poder elegir un juego se tiene que presionar “Enter”

4.2 Menú principal

Incluye botones para acceder a cada uno de los cuatro juegos.

4.3 Navegación entre juegos

- Seleccionar el juego deseado mediante el número del juego
- Para regresar al menú principal del juego primero tienes que pausar el juego en el que te encuentres y presionar el número de:”Salir al menú principal del juego”

4.4 Botones y elementos interactivos

El sistema presenta botones con funciones como:

- **Enter:** Iniciar un juego (en la pantalla inicial del juego).
- **Enter:** Pausar los juegos.
    - Navegar con ↑ y ↓ para elegir la opción presionar “Enter”

4.5 Cierre de la aplicación

- Cerrar la ventana desde la “X”.
- Presionando la Tecla Esc (Escape)

---

5. JUEGO 1 – NEON RUSH

5.1 Descripción del Juego

Neon Rush es un emocionante juego de carreras de vista cenital (top-down) con estética retro-futurista. Controlas un vehículo de alto rendimiento en circuitos cerrados llenos de curvas peligrosas, rectas vertiginosas y obstáculos que pondrán a prueba tus habilidades al volante. Cada pista presenta un diseño único con paredes luminosas de neón que delimitan el circuito, creando una experiencia visual vibrante y dinámica.

El juego combina elementos de carreras arcade clásicas con mecánicas modernas, ofreciendo un sistema de checkpoints que registra tu progreso, vueltas cronometradas y un sistema de turbo que añade velocidad explosiva cuando más lo necesitas.

5.2 Objetivo

Tu misión es completar el número requerido de vueltas al circuito dentro del límite de tiempo establecido. Para lograrlo debes:

- Cruzar la línea de meta después de pasar todos los checkpoints para completar cada vuelta
- Completar todas las vueltas requeridas antes de que el tiempo se agote
- Evitar colisiones con paredes y obstáculos que reducirán tu velocidad
- Usar el turbo estratégicamente para ganar velocidad en momentos clave

Condiciones de victoria:

- Completar todas las vueltas requeridas dentro del tiempo límite

Condiciones de derrota:

- El tiempo se agota antes de completar todas las vueltas

5.3 Controles (Teclado)

Controles de Movimiento:

Tecla	Acción	Descripción
↑ (Flecha Arriba)	Acelerar	Aumenta gradualmente la velocidad del vehículo. Mantén presionada para acelerar continuamente
↓ (Flecha Abajo)	Frenar/Reversa	Reduce la velocidad o permite retroceder si estás detenido
← (Flecha Izquierda)	Girar a la Izquierda	Rota el vehículo hacia la izquierda. El radio de giro depende de tu velocidad actual
→ (Flecha Derecha)	Girar a la Derecha	Rota el vehículo hacia la derecha. El radio de giro depende de tu velocidad actual

Controles Especiales:

Tecla	Acción	Descripción
Z	Activar Turbo	Activa un impulso de velocidad temporal. Solo disponible cuando está cargado
ENTER	Pausar	Abre el menú de pausa donde puedes reanudar, reiniciar o salir
ESC	Salir	Cierra inmediatamente el juego

Consejos de Control:

- No aceleres constantemente en curvas cerradas: Suelta el acelerador momentáneamente para tomar curvas más cerradas con mayor control
- Anticipa los giros: El vehículo tiene inercia, así que comienza a girar antes de llegar a la curva
- Combina frenado y giro: En curvas muy cerradas, frenar ligeramente mientras giras te dará mayor precisión
- El turbo es tu mejor aliado en rectas: Guarda el turbo para las secciones rectas más largas para maximizar su efectividad

5.4 Interfaz y HUD (Heads-Up Display)

El HUD muestra información crucial en tiempo real durante la carrera:

Elementos del HUD**:**

Superior Izquierda:

- **"VUELTA X/Y"**
  - **X:** Número de vuelta actual
  - **Y:** Total de vueltas requeridas
  - **Color:** Violeta eléctrico
  - **Ejemplo:** "VUELTA 1/3" significa que estás en la primera vuelta de tres

Superior Centro:

- **"PROGRESO: X%"**
  - Porcentaje de avance en la vuelta actual
  - Se calcula según los checkpoints pasados
  - 0%: No has pasado ningún checkpoint en esta vuelta
  - 100%: Has pasado todos los checkpoints y estás listo para cruzar la meta
  - Color: Violeta eléctrico

Superior Derecha:

- **"TIEMPO: XX.XXS"**
  - Tiempo restante en formato de segundos con dos decimales
  - **Color dinámico:**
    - **Verde:** Tiempo abundante (>30 segundos)
    - **Amarillo:** Tiempo moderado (10-30 segundos)
    - **Rojo:** Tiempo crítico (<10 segundos)

Inferior Derecha:

- **"VELOCIDAD: XXX PX/S"**
  - Velocidad actual en píxeles por segundo
  - Actualización en tiempo real
  - Color: Violeta eléctrico
- **Estado del Turbo (justo debajo de velocidad):**
  - **"TURBO LISTO":** El turbo está disponible para usar
  - **"TURBO: X.Xs":** Tiempo restante de turbo activo
  - **"Recargando: X.Xs":** Tiempo hasta que el turbo esté disponible de nuevo

Elementos Visuales en Pista:

- **Paredes:** Bordes luminosos de neón que delimitan la pista

5.5 Mecánicas Principales

5.5.1 Sistema de Conducción

Aceleración:

- La velocidad aumenta gradualmente mientras mantienes presionada la flecha arriba
- Existe una velocidad máxima normal que no puedes superar sin turbo
- La aceleración es más efectiva a bajas velocidades

Frenado:

- Mantén presionada la flecha abajo para reducir velocidad
- El frenado es más efectivo a altas velocidades
- Si frenas completamente, puedes retroceder lentamente

Giro:

- El radio de giro depende de tu velocidad actual
- A mayor velocidad: Giros más amplios y menos precisos
- A menor velocidad: Giros más cerrados y controlados
- El vehículo tiene inercia realista que afecta el manejo

Fricción Automática:

- Si no aceleras, el vehículo desacelera gradualmente por fricción
- Útil para tomar curvas sin frenar activamente

5.5.2 Sistema de Turbo

El turbo es tu herramienta clave para ganar velocidad explosiva en momentos estratégicos**.**

Características:

- Activación: Presiona la tecla Z cuando esté disponible
- Duración: El turbo dura varios segundos (tiempo fijo)
- Efecto: Aumenta temporalmente tu velocidad máxima y da un impulso inmediato
- Cooldown: Después de usarlo, debes esperar a que se recargue completamente
- Indicador visual: El HUD muestra claramente el estado (Listo/Activo/Recargando)

Estrategias de Uso:

- Úsalo en rectas largas para maximizar el beneficio
- Evita usarlo en curvas cerradas: Perderás control y chocarás
- Guárdalo para momentos críticos cuando el tiempo se esté agotando
- Combínalo con líneas de carrera óptimas para aprovechar todo su potencial

5.5.3 Sistema de Colisiones

Colisión con Paredes:

- Al chocar con una pared, tu vehículo se detiene instantáneamente
- Tu velocidad se reduce drásticamente (aproximadamente al 30%)
- El vehículo es empujado ligeramente alejándose de la pared
- Penalización: Pérdida de tiempo y velocidad valiosos

Prevención de Colisiones:

- Anticipa las curvas: Comienza a girar antes de llegar a la pared
- Reduce velocidad en curvas cerradas: Suelta el acelerador o frena ligeramente
- Aprende la pista: Memoriza las curvas más peligrosas
- Mantén distancia de las paredes: Conduce por el centro cuando sea possible

5.5.4 Sistema de Vueltas

Compleción de Vuelta:

1. El contador de vueltas se incrementa en 1

Victoria:

- Completar todas las vueltas requeridas (ejemplo: 3/3)
- Transición automática a pantalla de victoria
- Posibilidad de avanzar al siguiente nivel

5.7 Niveles o Dificultad

Estructura de Niveles:

Neon Rush cuenta por el momento con 1 nivel, cada nivel cuenta con características únicas**:**

Parámetros Configurables por Nivel:

- Número de vueltas requeridas: Puede variar entre niveles (1, 3, 5 vueltas, etc.)
- Tiempo límite: Cada nivel tiene un tiempo máximo diferente para completar las vueltas
- Diseño de pista: Cada nivel presenta un circuito único con curvas, rectas y obstáculos diferentes
- Música de fondo: Cada nivel tiene su propia banda sonora

Nivel Actual: Level 1

Características:

- Vueltas requeridas: 3
- Tiempo límite: Suficiente para aprender las mecánicas
- Dificultad: Introductoria - perfecta para familiarizarse con los controles

5.8 Consejos para Jugar

1. **Aprende la pista primero:**
    - En tu primera vuelta, conduce lento y memoriza el trazado
    - Identifica las curvas más peligrosas
2. **No aceleres siempre al máximo:**
    - En curvas cerradas, suelta el acelerador
    - Es mejor llegar lento que chocar y perder velocidad
3. **Usa el turbo en rectas:**
    - Nunca uses turbo en curvas cerradas
    - Guárdalo para las rectas más largas
    - Planifica cuándo lo necesitarás más
4. **Mantén distancia de las paredes:**
    - Conduce por el centro de la pista cuando sea posible
    - Deja margen de error para reaccionar

---

6. JUEGO 2 – PROJECT Z

6.1 Descripción del juego

El juego es un shooter arcade en pixel art, ambientado en un mundo post-apocalíptico devastado por una infección zombi. El jugador controla a un sobreviviente que debe abrirse paso a través de ciudades destruidas, ranchos en llamas, zonas industriales abandonadas y escenarios infestados de muertos vivientes. Cada partida es intensa, rápida y llena de acción, combinando reflejos, estrategia y manejo eficiente de recursos.

6.2 Objetivo

El objetivo del juego es sobrevivir y eliminar la mayor cantidad records de zombis, avanzando a través de zonas infestadas o resistiendo oleadas cada vez más peligrosas. El jugador debe demostrar habilidad para afrontar la mayor cantidad de hordas en el modo Defensa, mientras acumula kills, puntos y registra su desempeño

6.3 Controles (teclado, mouse)

- **Controles jugador 1:**
  - **Movimiento Izquierda:** A
  - **Movimiento Derecha:** D
  - **Disparo:** W
  - **Recarga:** R
  - **Golpe cuerpo a cuerpo;** S
- **Controles jugador 2:**
  - **Movimiento Izquierda:** ←
  - **Movimiento Derecha:** →
  - **Disparo:** ↑
  - **Recargar:** P
  - **Golpe cuerpo a cuerpo:** ↓
- **Controles generales:**
  - **Pausar:** ESC
  - **Navegar hacia arriba en menús:** ↑ Arriba
  - **Navegar hacia abajo en menús:** ↓ Abajo

6.4 Interfaz y HUD

Incluye:

- Barra de vida.
- Contador de munición.
- Ronda actual.
- Número de zombis restantes.
- Pack a punch: Máquina central de mejora de armas

6.5 Niveles o dificultad

El juego no cuenta con una dificultad fija la dificultad va aumentando progresivamente a la hora de ir avanzando con las oleadas de zombies de los cuales progresivamente iran aumentando vida y daño de los mismos

6.6 Consejos para jugar

- No mal gastar balas
- Mejorar el arma lo antes posible
- Usar el ataque cuerpo a cuerpo al inicio y rematar con balas
- Agarrar todos los power up posibles

---

7. JUEGO 2 – COSMIC ODDISEY

7.1 Descripción del juego

Cosmic Oddisey es un juego de plataformas 2D de acción y aventura con estética pixel art retro. El jugador debe navegar a través de niveles complejos llenos de obstáculos, enemigos patrulleros y jefes finales desafiantes. El juego combina mecánicas clásicas de plataformas con elementos modernos como sistema de disparos, escudo temporal y física avanzada. Cada nivel presenta desafíos únicos con plataformas de diferentes tipos y culmina en enfrentamientos épicos contra jefes con múltiples fases de combate.

7.2 Objetivo

El objetivo principal es completar todos los niveles disponibles derrotando enemigos y venciendo al jefe final de cada nivel. El jugador debe:

- Derrotar al Boss del nivel para completar el nivel
- Completar cada nivel en el tiempo disponible de cada nivel

7.3 Controles (teclado)

Movimiento básico:

- **Flecha Izquierda (←):** Mover al jugador hacia la izquierda
- **Flecha Derecha (→):** Mover al jugador hacia la derecha
- **Flecha Arriba (↑):** Saltar (altura fija determinada por física del nivel)

Acciones de combate:

- **Barra Espaciadora (SPACE):** Disparar proyectiles hacia adelante (dirección de movimiento)
- **Tecla B:** Golpear cuerpo a cuerpo a un enemigo
- **Tecla V:** Activar escudo temporal (protección contra daño)

Controles del sistema:

- **Tecla ESC (Escape):** Salir del juego inmediatamente
- **Tecla Enter:** Pausar el juego (funcionalidad de pausa)/Seleccionar opción en Menú de Pausa
- **Flecha Arriba/Flecha Abajo:** Navegación entre opciones en Menú de Pausa

Notas importantes:

- El movimiento es continuo mientras se mantiene presionada la tecla
- El salto no puede ejecutarse en el aire (no hay doble salto)
- Los disparos tienen cadencia automática sin cooldown
- El escudo tiene duración limitada y tiempo de recarga

7.4 Interfaz y HUD

Elementos del HUD (Head-Up Display):

Esquina superior izquierda:

- **Tiempo transcurrido: [número]s:** Tiempo transcurrido desde el comienzo del nivel actual
    - Muestra el tiempo transcurrido desde el comienzo del nivel actual
    - Color Neon Cyan

Esquina derecha izquierda:

- **Información de tiempo de uso de escudo:[número]s:** Información de uso de Escudo (tiempos)
    - Según el uso del escudo, se muestra información de tiempos en la pantalla
    - Color Neon Cyan

Centro inferior de la pantalla:

- **Nombre del Boss del Nivel**
    - Colores distintivos del Boss
    - Nombre del Boss del Nivel Actual

Elementos visuales en juego:

- **Barra de vida del Boss**
    - Muestra salud restante del jefe
    - Escudo: Imagen de escudo si está activo
- **Barras de vida de enemigos:** Sobre cada enemigo vivo
    - Indicador rojo de salud individual
    - Desaparece cuando el enemigo muere
- **Efectos visuales del jugador:**
    - Escudo: Imagen de escudo cuando está activo
    - Animaciones de estado (idle, run, jump)
    - Proyectiles visibles

Fuentes utilizadas:

- Press Start 2P (retro pixel)
- Pixel Emulator (números y texto principal)

Información adicional:

- El HUD es fijo (GUI Camera) y no se mueve con la cámara del mundo
- Los colores contrastan con el fondo para legibilidad

7.5 Mecánicas principales

Sistema de movimiento y física:

- Gravedad configurable: Cada nivel puede tener gravedad diferente
- Movimiento lateral: Velocidad horizontal constante mientras se presiona tecla
- Salto básico: Impulso vertical contra la gravedad
- Sincronización con plataformas móviles: El jugador se mueve junto con la plataforma
- Detección de suelo: Sistema de física detecta si está sobre superficie sólida
- Caída al vacío: Si Y < -200, el jugador muere totalmente

Sistema de combate:

- **Disparos del jugador:**
  - Proyectiles rectilíneos en dirección de movimiento
  - Sin límite de munición
  - Colisión con enemigos y jefe
  - Destrucción al impactar o salir de límites del nivel
- **Escudo temporal:**
  - Protección completa contra daño por tiempo limitado
  - Representación visual (imagen)
  - Tiempo de recarga tras desactivarse
  - No previene caída al vacío
- **Daño por contacto:**
  - Colisión con enemigos reduce vida
  - Sistema de invulnerabilidad temporal tras recibir daño
  - Efecto visual al perder vida (opacidad del sprite)

Tipos de plataformas:

1. **Estáticas:** No se mueven, base sólida
2. **Móviles:** Se mueven horizontal o verticalmente entre límites

Sistema de enemigos:

- **Patrullaje:** Movimiento entre límites izquierdo/derecho
- **Daño por contacto:** Reducen vida del jugador al tocarlo
- **Sistema de respawn:** Pueden reaparecer tras muerte (configurable)
- **Barras de vida individuales:** Feedback visual de salud
- **Limpieza automática:** Enemigos sin respawn se eliminan permanentemente

Sistema de Boss:

- **Activación por proximidad:** Se activa cuando el jugador entra en radio
- **Múltiples fases de combate:** Comportamiento cambia según salud
- **Habilidades especiales:**
    - Invocación de enemigos adicionales
    - Lanzamiento de proyectiles dirigidos
    - Escudo temporal propio
    - Patrones de ataque complejos
- **Física independiente:** Motor de física separado del jugador
- **Sincronización con plataformas:** Se mueve con plataformas móviles
- **Derrota requerida: D**ebe ser eliminado para completar nivel

Sistema de cámara inteligente:

- **Dead Zone:** Área central donde el jugador se mueve sin mover cámara
    - Izquierda: 4800px, Derecha: 400px
    - Inferior: 100px, Superior: 500px
- **Límites automáticos:** Se restringe a dimensiones del nivel

7.6 Niveles o dificultad

Estructura de niveles:

El juego utiliza un sistema de niveles modulares cargados dinámicamente desde archivos JSON. Cada nivel es completamente personalizable:

Características por nivel**:**

- **Dimensiones únicas:** Ancho y alto configurables (ej: 5000x800 píxeles)
- **Gravedad personalizada:** Puede variar entre niveles para diferentes sensaciones
- **Layout único:** Distribución específica de muros, plataformas y enemigos
- **Fondos temáticos:** Cada nivel puede tener múltiples capas de fondo
- **Spawn point específico:** Posición inicial del jugador

Progresión de dificultad:

Nivel 1 (Introducción):

- Layout simple con plataformas estáticas y móviles
- Múltiples enemigos patrulleros básicos
- Boss con patrones predecibles (Nexus)
- Gravedad estándar (1.0)

Sistema de registro:

- Cada nivel tiene ID único (entero positivo)
- Registro automático en GameState
- Tracking de muerte por nivel

Condiciones de progreso:

- Derrotar al Boss
- Sobrevivir con porcentaje de vida

Transiciones:

- Pantalla de transición de 3 segundos
- Carga automática del siguiente nivel
- Reset de tiempo de nivel

Fin del juego:

- Completar todos los niveles registrados
- Pantalla de victoria (PlatformGameEnd)

7.7 Consejos para jugar

Estrategias de movimiento:

1. **Domina el timing de los saltos**
    - Practica la distancia exacta que cubre cada salto
    - Memoriza la altura máxima alcanzable
    - En plataformas móviles, salta en dirección del movimiento para mayor alcance
2. **Usa el momentum a tu favor**
    - Corre antes de saltar para cubrir distancias largas
    - Las plataformas móviles te dan velocidad adicional
    - Las plataformas de rebote pueden alcanzar áreas inaccesibles normalmente

3. **Gestión del escudo**
    - Activa el escudo ANTES de entrar en zonas peligrosas
    - No desperdicies el escudo en áreas seguras
    - Úsalo estratégicamente en combates contra el Boss
    - Recuerda su tiempo de recarga

Estrategias de combate:

1. **Prioriza objetivos**
    - Elimina enemigos básicos antes de enfrentarte al Boss
    - Los enemigos con respawn son menos prioritarios
    - Enfócate en enemigos que bloquean tu camino
2. **Posicionamiento estratégico**
    - Mantén distancia de enemigos mientras disparas
    - Usa plataformas elevadas como ventaja táctica
    - El Boss tiene patrones predecibles, aprende a esquivarlos
3. **Conserva tu salud**
    - Evita daño innecesario (mejor esquivar que tanquear)

Mecánicas avanzadas:

1. **Sincronización con plataformas móviles**
    - Anticipa su movimiento antes de saltar
    - Muévete con ellas cuando estés encima
    - Úsalas como transporte en secciones largas
2. **Cadena de plataformas desvanecientes**
    - Observa el patrón antes de comprometerte
    - Planea tu ruta completa, no solo el primer salto
    - Ten siempre un plan de respaldo
3. **Combate contra Boss**
    - Aprende sus patrones en la primera fase
    - El Boss cambia comportamiento según su salud
    - Escudo + disparos agresivos en ventanas de oportunidad
    - Los enemigos invocados pueden ser ignorados si te enfocas

---

8. JUEGO 4 – GHOST OF TSUSHIMA

8.1 Descripción del juego

Juego de peleas 2D multijugador local que enfrenta a dos personajes en combate directo. Inspirado en el estilo de combate de juegos clásicos de lucha, con mecánicas de movimiento, salto y ataques múltiples.

Personajes Jugables

1. Martial Hero (Jugador 1) - Guerrero samurái con técnicas de espada
2. Skeleton (Jugador 2) - Guerrero esqueleto con ataques especiales

Modos de Juego

- **Combate Local 1v1:** Dos jugadores en la misma pantalla
- **Sistema de Vida:** Cada personaje tiene 100 puntos de vida
- **Victoria por Eliminación:** El combate termina cuando un personaje llega a 0 HP

8.2 Objetivo

¡Derrota a tu oponente! Cada personaje comienza con 100 puntos de vida. El primero en reducir la vida del oponente a 0 gana la batalla.

8.3 Controles (teclado, mouse)

- **Controles jugador 1:**
- **Mover a la izquierda:** ← Flecha Izquierda
- **Mover a la derecha:** → Flecha Derecha
- **Saltar:** ↑ Flecha Arriba
- **Ataque básico:** 0 (cero)
- **Ataque básico:** Click Izquierdo
- **Ataque especial:** Click Derecho
- **Controles jugador 2:**
- **Mover a la izquierda:** A
- **Mover a la derecha:** D
- **Saltar:** W
- **Ataque básico:** G
- **Ataque medio:** H
- **Ataque fuerte:** J

- **Controles generales:**
  - **Iniciar juego / Pausar / Seleccionar:** ENTER
  - **Salir del juego:** ESC
  - **Mostrar información de depuración:** F3
  - **Navegar hacia arriba en menús:** ↑ Arriba
  - **Navegar hacia abajo en menús:** ↓ Abajo
  - **Seleccionar mapas:** 1 - 6

8.4 Interfaz y HUD

Durante el Combate Verás:

- Barra de Vida del Jugador 1 (Arriba a la izquierda)
- Barra de Vida del Jugador 2 (Arriba a la derecha)
- Indicadores de Daño
- Personajes

8.5 Consejos para jugar

Fundamentos de combate:

- Mantén distancia media: ni muy cerca ni muy lejos del oponente
- Observa las animaciones: cada ataque tiene frames de inicio predecibles
- El frame medio conecta el golpe: anticipa cuándo llegará el impacto
- Usa movimiento lateral entre ataques para reposicionarte

Estrategias por personaje:

Martial Hero (Jugador 1):

- Aprovecha tu movilidad: muévete constantemente con las flechas
- Ataque básico (0/Click izquierdo): rápido y confiable
- Ataque especial (Click derecho): mayor daño, úsalo cuando conectes
- Combina salto + ataque para presionar desde arriba

Skeleton (Jugador 2):

- Tienes 3 tipos de ataque: usa el apropiado según distancia
- Ataque básico (G): alcance estándar, bajo daño
- Ataque medio (H): mayor alcance, daño moderado
- Ataque fuerte (J): máximo alcance y daño, pero más lento
- Tu ventaja es el rango: mantén distancia óptima

Errores comunes a evitar:

- No spamees ataques: son predecibles y te dejan vulnerable
- No te quedes quieto: eres un blanco fácil
- No ataques desde muy lejos: desperdicias animaciones
- No ignores la vida: vigila ambas barras constantemente

Gestión del combate:

- Cuando tengas ventaja de vida: juega defensivo y seguro
- Cuando estés perdiendo: toma riesgos calculados
- Lee los patrones: cada jugador tiene hábitos repetitivos
- Paciencia sobre agresión: espera oportunidades claras

Controles críticos:

- Practica la sincronización salto + ataque
- Domina el movimiento rápido: cambios de dirección súbitos
- Memoriza las teclas sin mirar: reflejos sobre pensamiento
- Usa ENTER para pausar y respirar cuando necesites

---

9. PREGUNTAS FRECUENTES (FAQ)

9.1 Preguntas Generales de la Aplicación

P1: ¿Puedo ejecutar la aplicación sin instalar Python?

R: No, la aplicación requiere tener Python 3.10 o superior instalado en el sistema, junto con la biblioteca Pygame versión 2.1 o mayor. La aplicación se distribuye como código fuente que debe ejecutarse desde Visual Studio o mediante la línea de comandos. Es fundamental seguir las instrucciones de instalación detalladas en la sección 3 de este manual para configurar correctamente el entorno de desarrollo antes de ejecutar la aplicación.

P2: ¿Cómo puedo cambiar entre los diferentes juegos disponibles?

R: Desde el menú principal, utilice las teclas numéricas (1-4) para seleccionar el juego deseado. Para regresar al menú principal desde cualquier juego, primero pause la partida presionando ENTER y luego seleccione la opción "Salir al menú principal" utilizando las flechas de navegación.

P3: ¿La aplicación guarda automáticamente mi progreso o puntuaciones?

R: Actualmente, el sistema de guardado varía según el juego. Los récords de puntuación y estadísticas se mantienen durante la sesión activa de la aplicación. PROJECT Z cuenta con un sistema de tabla de récords que registra los mejores desempeños. Se recomienda consultar las características específicas de cada juego en sus respectivas secciones del manual para información detallada sobre la persistencia de datos.

9.2 Preguntas Técnicas y Solución de Problemas

P4: ¿Qué debo hacer si la aplicación se cierra inmediatamente al ejecutarla?

R: Si la aplicación presenta un cierre abrupto, ejecute el programa desde la terminal o consola de comandos para visualizar los mensajes de error. Los problemas más comunes incluyen: (1) Pygame no instalado correctamente - solución: reinstalar mediante pip install pygame, (2) Rutas de archivos multimedia incorrectas - verificar la integridad de las carpetas de assets, (3) Versión incompatible de Python - asegurarse de utilizar Python 3.10 o superior, (4) Dependencias faltantes - verificar que todas las librerías del archivo requirements.txt estén instaladas.

P5: ¿Es posible jugar con un controlador o gamepad en lugar del teclado?

R: Actualmente, la aplicación está diseñada exclusivamente para controles de teclado y mouse (según el juego). El soporte para controladores externos no está implementado en la versión actual. Todos los juegos han sido optimizados para proporcionar una experiencia fluida utilizando los controles de teclado especificados en las secciones individuales de cada juego.

P6: ¿Qué requisitos de pantalla son necesarios para una visualización óptima?

R: Se recomienda una resolución mínima de pantalla de 1280x720 píxeles para una experiencia visual óptima. La aplicación está diseñada para adaptarse a diferentes resoluciones, pero resoluciones inferiores pueden resultar en elementos de interfaz recortados o dificultades para visualizar información del HUD. Para mejor rendimiento, configure su sistema con aceleración gráfica 2D habilitada.

9.3 Preguntas sobre Juegos Específicos

P7: En COSMIC ODYSSEY, ¿qué sucede si caigo al vacío durante un nivel?

R: En COSMIC ODYSSEY, caer al vacío (cuando Y < -200) resulta en la muerte instantánea del personaje. Tendrás que volver al menú de selección de personajes y comenzar la aventura de nuevo.

P8: ¿Cómo funciona el sistema de turbo en NEON RUSH y cuándo debo utilizarlo?

R: El sistema de turbo en NEON RUSH proporciona un impulso temporal de velocidad con las siguientes características: (1) Activación mediante la tecla Z cuando el indicador muestre "TURBO LISTO", (2) Duración limitada de varios segundos, (3) Período de recarga obligatorio tras su uso. La estrategia óptima consiste en reservar el turbo para secciones rectas del circuito, evitando su uso en curvas cerradas donde la velocidad adicional compromete el control del vehículo.

P9: En PROJECT Z, ¿existe alguna estrategia para maximizar la supervivencia en oleadas avanzadas?

R: Para optimizar la supervivencia en oleadas progresivamente difíciles de PROJECT Z, se recomienda: (1) Mejorar el arma mediante la máquina Pack-a-Punch en la etapa más temprana posible, (2) Utilizar el ataque cuerpo a cuerpo para debilitar enemigos en rondas iniciales, conservando munición para oleadas críticas, (3) Recolectar sistemáticamente todos los power-ups disponibles, (4) Gestionar eficientemente la munición mediante recargas estratégicas. La dificultad escala incrementando progresivamente la vida y daño de los enemigos.

P10: ¿GHOST OF TSUSHIMA permite jugar contra la computadora o solo modo multijugador local?

R: GHOST OF TSUSHIMA está diseñado exclusivamente como experiencia multijugador local para dos jugadores en la misma computadora. No incluye modo contra inteligencia artificial ni opción para juego individual. Ambos jugadores deben estar presentes físicamente para participar en el combate. El Jugador 1 controla al personaje Martial Hero utilizando las flechas direccionales, mientras que el Jugador 2 controla a Skeleton mediante las teclas WASD.

10 GLOSARIO TÉCNICO

Términos Clave

- **Actor:** Clase base para todos los personajes jugables en el juego de peleas
- **Hitbox:** Rectángulo de colisión para detectar impactos
- **Frame:** Imagen individual de una animación
- **Surface:** Objeto de Pygame que contiene una imagen
- **Blit:** Operación de dibujar una imagen sobre otra
- **dt (Delta Time):** Tiempo transcurrido entre frames
- **Sprite Sheet:** Imagen única que contiene múltiples frames
- **Offset:** Desplazamiento de posición para ajustes visuales
- **Fallback:** Valor o comportamiento por defecto en caso de error
- **Cache:** Almacenamiento temporal para evitar recargas