En este capítulo se detalla la aportación de cada uno de los alumnos en el desarrollo del proyecto.

Nicolás Rosa Caballero

Existen 3 aportaciones principales de Nicolás al proyecto:

- Simulador en C++
  - Maquetación inicial del proyecto, implementación de OpenGL, GLFW y IMGUI.
  - Implementación de la lógica inicial de la simulación de arena y la interacción con el usuario.
  - Refactorización de las partículas básicas para este modelo: Arena, Agua, Ácido, Roca, Aire y Gas para generalizar su comportamiento y facilitar el desarrollo de nuevas partículas.
  - Investigación e implementación de un sistema de interacciones entre partículas.
  - Optimizaciones y revisiones del código pertinentes para asegurar que la implementación sea un buen referente comparativo para el resto de implementaciones.
- Simulador en Lua con Love2D
  - Diseño de la arquitectura del proyecto y la implementación de toda la lógica de la simulación de arena.
  - Implementación de un sistema de partículas programables en Lua.
  - Diseño e implementación del API para la creación de partículas y sus interacciones.
  - Sistema de eventos implementado con Beholder para disminuir la dependencias entre componentes.
  - Implementación de un sistema de multithreading para mejorar el rendimiento del simulador.
  - Implementación de un algoritmo para encontrar el mayor número de hilos y tamaño de chunk en función del tamaño del canvas y la cantidad de núcleos de la CPU que permita aprovechar la mayor cantidad de hilos simultáneos sin incurrir en condiciones de carrera.
  - Sincronización del trabajo entre los hilos aplicando una implementación propia de la técnica “work stealing” en base al uso de canales.
  - Implementación de un sistema de doble buffer para evitar condiciones de carrera y asegurar la consistencia de los datos.
  - Elaboración de la documentación del API del sistema de partículas.
- Simulador en Rust con Macroquad
  - Creación del proyecto y configuración de las dependencias.
  - Configuración del proyecto para soportar WebAssembly, aplicando características distintas según el destino de compilación mediante flags de compilación condicional.
  - Diseño de la arquitectura del proyecto y la implementación de la lógica de la simulación de arena.
  - Implementación de un sistema de comunicación con WebAssembly mediante una cola de comandos global.
  - Implementación de un sistema de plugins para facilitar la creación de nuevas partículas.
  - Extensión del sistema de plugins mediante la creación de un tipo de plugin que toma como datos un fichero JSON y genera una función a ejecutar para la partícula.

    - Diseño e implementación del formato de JSON para definir partículas y sus interacciones.
- Web Vue3 envoltorio del ejecutable de Rust
    - Creación de la web usando Vue, Vite y TailwindCSS.
    - Diseño en Canva y posterior implementación de la interfaz gráfica de la web.
    - Implementación del código JavaScript pegamento que permite comunicar la web y el ejecutable WebAssembly.
    - Implementación de un sistema de guardado y cargado de plugins en formato JSON.
    - Implementación de un menú de ayuda y un sistema de gestos para facilitar la interacción con la web.
    - Implementación de integración continua en GitHub mediante GitHub Actions para automatizar la generación de la web.
    - Interactividad de los botones, gestos y otros elementos de la web mediante JavaScript y Vue (variables reactivas, watchers).
    - Implementación de Pinia para gestionar un estado global reactivo y minimizar la interdependencia entre componentes.
    - Colaboración con Jonathan para integrar Blockly en la web.
    - Edición de algunos generadores y definiciones de bloques creados por Jonathan para adaptarlos a las necesidades del proyecto.
    - Diseño del logo de la web.
- Otros
    - Elaboración del plan de pruebas con usuario.
    - Realización de parte de las pruebas de usabilidad con usuario.
    - Elaboración de pruebas de rendimiento entre distintos simuladores.
    - Elaboración de figuras para la memoria mediante scripting en Typst y Canva.

Jonathan Andrade Gordillo

- Simulador en C++
  - Configuración incial del proyecto así como configurado de solución, proyecto y biblicotecas
  - Implementación de partículas iniciales como agua, roca y gas
  - Asistido en la interaccion con el usuario añadiendo pincel ajustable
  - Añadido propiedades físicas a las particulas como la densidad
  - Movimiento que se ajuste a estos parámetros físicos
  - Investigación de sistema alternativo de interaccion entre partículas mediante funciones anónimas
  - Solución de bugs a lo largo del desarrollo relacionados con rendimiento e interacciones
- Simulador en Rust con Vulkan haciendo uso de GPU
  - Investigación de posibles formas de hacer uso de la GPU para el cálculo de la lógica, entre ella añadir OpenMP o SYCL al proyecto principal
  - Desarrollo de pipeline gráfico básico haciendo uso de Vulkan
  - Desarrollo de sistema de interacción básico para colocar partículas
  - Implementacion de interfaz mediante ImGUI
  - Implementación de partícula de arena
  - Investigación y desarrollo de compute shaders que permitan delegar el movimiento a la GPU
  - Exploración de diferentes tamaños de work group que den lugar a un mayor rendimiento de ejecución
- Blockly para simulador de Rust
  - Investigación sobre las necesidades del proyecto y los requisitos del módulo de Blockly.
  - Creación de todos los bloques presentes en el proyecto, así como de los posibles mutadores que necesiten a excepcion de uno
  - Ajuste del toolbox para incluir los bloques desarrollados
  - Implementación de los generadores para cada bloque creado, aunque algunos de ellos tuvieron que ser corregidos más tarde junto a Nicolás de
  - Colaboración con mi compañero para incluir Blockly en la página web
- Otros
  - Realización de parte de las pruebas de usabilidad con usuario.
  - Elaboración de pruebas de rendimiento entre distintos simuladores.