Desarrollo minijuegos para la mejora de procesos multitasking

David Andrés Torres Betancour

Informe de práctica profesional, como requisito para optar por el título de:

Ingeniería de Sistemas.

Asesores:

Astrid Duque Ramos, Doctora en Ingeniería informática (Interna)

Jorge Andrés Gómez, Desarrollador de videojuegos (Externo)

Universidad de Antioquia

Facultad de Ingeniería

Departamento de Ingeniería de Sistemas

Medellín, Colombia

2022

Cita	(Torres Betancour, 2022)
Referencia **Estilo APA 7 (2020)**	Torres Betancour, D. A. (2022). *Desarrollo minijuegos para la mejora de procesos multitasking,* Semestre de industria, Pregrado,Universidad de Antioquia, Medellín

Dedicatoria

Dedicado al profesor Sebastian Isaza, quien me introdujo en el mundo de la programación y me motivó a iniciar esta grandiosa carrera.

Agradecimientos

Al semillero de videojuegos de la Universidad, quienes me dieron el camino para iniciar en el Desarrollo de videojuegos.

A la empresa Indie Level Studio por todo el acompañamiento y confianza depositada.

A los asesores Astrid y Andrés, por todo el apoyo y atención prestada durante todo el desarrollo de las prácticas.

Tabla de contenido

Lista de figuras

Siglas, acrónimos y abreviaturas

GDD	Game Design Document.
SOLID	Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion.

Resumen

Indie Level Studio es una compañía de tecnología que se dedica al desarrollo de videojuegos. Donde su objetivo está en la formación, capacitación y mercadeo en contenido digital gamificado por medio de plataformas interactivas. En este caso, la empresa Teleperformance, requiere un método ágil y eficaz que capacite y mejore el desempeño de sus empleados, especialmente en los procesos multitasking a través de una plataforma interactiva, que se pueda acceder desde un navegador. La empresa Indie Level Studio, propone el desarrollo de una serie de videojuegos, al practicante se le asigna el videojuego IceCream , que consiste en atender los clientes en una heladería, memorizando y preparando los pedidos que se soliciten en el menor tiempo posible.

En este proyecto, se desarrollaron tres versiones del videojuego (alfa, beta y versión candidata) y se desplegó en la plataforma del cliente, en este proceso se pudo poner en práctica el conocimiento adquirido en el pregrado, de una forma más profesional, asegurando una alta calidad del producto a través de pruebas manuales y automatizadas.

Palabras clave: videojuegos , WebGL, gamificación.

Abstract

Indie Level Studio is a technology company dedicated to video game development. Where its objective is the education, training and marketing of gamified digital content through interactive platforms. In this case, the company Teleperformance requires an agile and effective method that trains and improves the performance of its employees, especially in multitasking processes through an interactive platform that can be accessed from a browser. The company Indie Level Studio, proposes the development of a series of video games, the student is assigned the video game IceCream , which consists of serving customers in an ice cream parlor, memorizing and preparing the orders that are requested in the shortest possible time.

In this project, three versions of the video game were developed (alpha, beta and candidate version) and it was deployed on the client's platform. In this process, the knowledge acquired in the undergraduate course could be put into practice more professionally, ensuring a high quality of the product through manual and automated tests.

Keywords: video games, WebGL, gamification.

Introducción

Indie Level Studio S.A.S es una compañía de tecnología que se dedica al desarrollo de videojuegos, ya sea para terceros o construyendo ideas propias. Sus proyectos se enfocan en la formación, capacitación y mercadeo en contenido digital gamificado por medio de plataformas interactivas. Actualmente, la empresa ha participado en más de 200 proyectos y seis videojuegos propios, donde destacan Jeanne,Terra y Logicubes, posicionándose de una forma más firme en el área de tecnología.

La empresa Teleperformance, requiere un método ágil y eficaz que capacite y mejore el desempeño de sus empleados, especialmente en los procesos multitasking. Es por esto que busca a la empresa Indie Level Studio, para diseñar un conjunto de minijuegos para que ataquen el problema de una forma directa y se ajuste a las temáticas propias de la empresa.

Desde la empresa Indie Level Studio se designó al equipo Gameplay Developer, para crear los minijuegos, de la mano del equipo de arte; cada programador estará encargado de solo un minijuego, en el caso del practicante fue el de IceCream, que consiste en atender los clientes en una heladería, memorizando y preparando los pedidos que se soliciten en el menor tiempo posible. En el videojuego se tendrán que desarrollar sistemas de puntaje, rounds, interacción con los objetos, generadores de contenido aleatorio, entre otros. Por otra parte, también se estará desarrollando sistemas más generales que necesiten los otros minijuegos, como PathFinder, implementación de mockups, apoyar procesos de testing, entre otras actividades que lleguen a hacer falta para culminar el proyecto con éxito.

1 Objetivos

- **1.1 Objetivo general**

Desarrollar minijuegos para capacitar y mejorar los procesos de multitasking y toma de decisiones rápida, por parte de los empleados de la empresa Teleperformance.

1.2 Objetivos específicos

- Diseñar GDD (Game Design Document) de cada videojuego.
- Diseñar diagramas UML de cada sistema a desarrollar
- Realizar implementación de los Mockups.
- Implementar plugins propios de la empresa.
- Desarrollar las mecánicas de juego de cada proyecto.
- Desarrollar los efectos visuales requeridos en cada minijuego.
- Documentar los sistemas de control realizados.
- Depurar cada entrega del videojuego.

2 Marco teórico

En los videojuegos es de crucial importancia la constante adopción de nuevas tecnologías que mejoren el proceso de desarrollo, desde herramientas para la comunicación del equipo, hasta librerías que automatizan o agilicen la construcción del producto que se solicita. Las herramientas y conceptos, para este proyecto se mencionan a continuación:

- **GDD (Game Design Document):** este documento contiene la forma en que se van a implementar los requerimientos del videojuego, permitiendo observar de forma general aspectos como la historia, el flujo, las mecánicas, referencias visuales, entre otros. Principalmente es creado y editado por el equipo de desarrollo .

- **Ciclo de vida del lanzamiento de software**: es el conjunto de etapas que tienen los proyectos al momento de ser lanzados al mercado . Se componente de las siguientes versiones:
    - **Alpha:** presenta las características esenciales que tendrá el software, pero el producto aún se encuentra inestable.
    - **Beta:** representa la primera versión completa del producto, aún puede ser inestable pero a partir de aquí ya no se podrán agregar más características. Solo pequeñas ediciones y corrección de errores.
    - **Estable:** etapa donde se han corregido todos los errores y los parámetros para el correcto funcionamiento del software. El producto se encuentra listo para lanzamiento.

- **Unity:** es una herramienta de desarrollo de videojuegos creada por la empresa Unity Technologies. Proporcionando motores de audio, animación, física, renderizado . Entre los paquetes adicionales que brinda este motor, se encuentra Unity Test Framework, el cual contiene un conjunto de herramientas de testing para los desarrolladores, donde cada

una de las características del proyecto se le pueden realizar pruebas unitarias, ayudando a corregir y verificar los componentes desarrollados dentro de la plataforma.

- **GitLab:** es un servicio web para la gestión de repositorios, donde se puede llevar un control de versiones basado en Git.
- **ClickUp**: con el fin de llevar un seguimiento de los proyectos, observando las personas que están a cargo y las tareas que están realizando, se optó por esta herramienta, la cual es una plataforma que ofrece todo lo necesario para una gestión minuciosa de los proyectos, herramientas como listas de tareas, notas, recordatorios, estimación de tiempos, entre otros. Permitiendo a todo el equipo de trabajo usar la misma herramienta para la planeación y colaboración en todas las fases del proyecto.
- **FileZilla**: para subir el compilado del videojuego al servidor Azure de la plataforma del cliente, se hace por medio de esta herramienta la cual podemos conectarnos a cualquier servidor, indicando la correspondiente ruta y credenciales, logrando consultar, adquirir y manipular el contenido del mismo.
- **Google Drive:** con el objetivo de compartir y almacenar la documentación del proyecto con los integrantes del equipo , se realiza a través de Google Drive, el cual es el servicio de almacenamiento de datos en internet que provee Google.
- **Discord:** en cuanto a la comunicación, se realiza por la plataforma Discord, la cual consiste en un conjunto de canales de voz y chat, donde cada miembro tiene un rol asignado, logrando que la difusión de la información entre el equipo de trabajo sea efectiva.
- **GitKraken**: es una herramienta de escritorio para controlar Git de una forma gráfica, permitiendo enlazar directamente con varios repositorios, facilitando el flujo de trabajo al momento de crear múltiples ramas, logrando tener una visualización más clara de todo el proyecto.
- **Rider JetBrains:** es un editor de código C# para Unity con autocompletado inteligente que funciona en Windows, Mac y Linux.

3 Metodología

La metodología de desarrollo de la empresa Indie Level Studio, ver Figura 1, cuenta con una primera etapa en la que se hace el levantamiento de requisitos del software, luego se pasa a la planeación donde se asignan las tareas y se estiman tiempos, después se inicia la etapa de implementación donde de forma periódica se estarán entregando avances del videojuego. Finalmente, si los requisitos fueron completados y el cliente a futuro no hará actualizaciones, el proyecto termina, en caso contrario, se vuelve a la etapa de implementación para añadir los cambios solicitados o funcionalidades que estén pendientes.

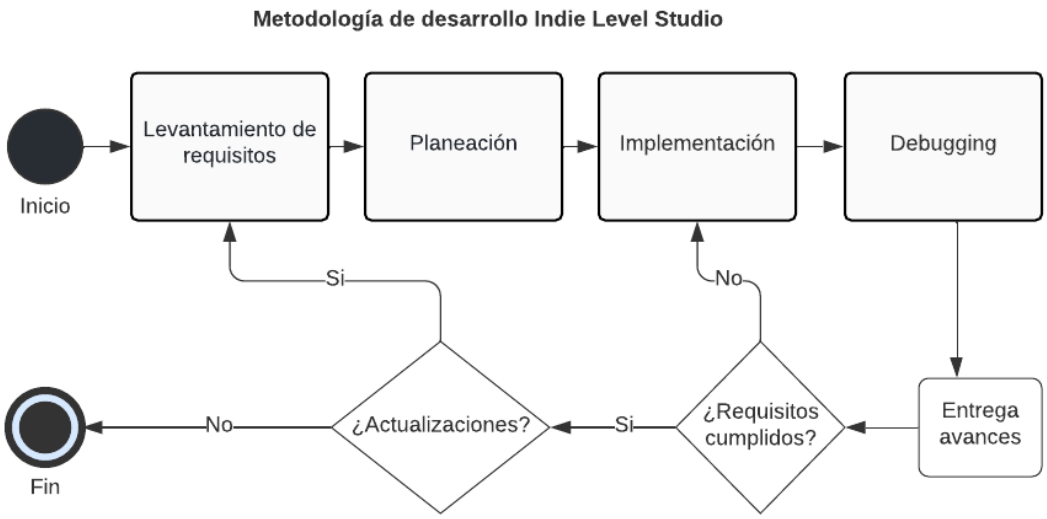

Figura 1. Metodología de desarrollo Indie Level Studio

- **Levantamiento de requisitos:** es la etapa donde la productora, una vez se haya reunido con el cliente, entregará al equipo de desarrollo cada uno de los requisitos del producto. Con esta información se procede a crear un GDD , el cual será la propuesta inicial que se les hará a los clientes sobre el producto que van a adquirir.
- **Planeación:** con el fin de asignar las tareas a realizar y la estimación de los tiempos del proyecto, se diseña la arquitectura del videojuego con el uso de un diagrama de clases, describiendo cada una de las clases y sus responsabilidades. Terminada esta arquitectura,

se procede a definir cada una de las actividades a realizar, indicando las personas responsables y sus fechas estimadas, toda esta información se consolida en la plataforma ClickUp .

- **Implementación:** se ingresa al software de Unity y al editor de código Riden Jetbrains para iniciar a implementar las actividades definidas durante la planeación. Al final de cada actividad se tendrá una reunión con el director de desarrollo para validar la arquitectura implementada y subir los cambios al repositorio en GitLab.
- **Debugging:** con la herramienta Unity Test Framework, se realiza una serie de test unitarios a cada uno de los sistemas desarrollados para identificar posibles problemas y pasar a su correspondiente corrección.
- **Entrega avances:** para asegurar que los requisitos solicitados por el cliente se están cumpliendo, de forma periódica se realiza una entrega de avances del videojuego, en el servidor del cliente, usando la herramienta FileZilla.
- **Actualizaciones :** en caso de que al terminar el proyecto, el cliente desee añadir funcionalidades adicionales a las que inicialmente se plantearon, se pasa nuevamente al levantamiento de requisitos, para iniciar todo el proceso de desarrollo.

Como metodología de seguimiento, cada día se realiza una reunión para hablar sobre los avances o problemas presentados, al terminar la jornada, se reportan las actividades desarrolladas en la herramienta ClickUp. Ver **Figura 2**.

Figura 2. Tareas reportadas en ClickUp

4 Resultados

Como resultado final, se logró exitosamente la entrega y el despliegue del videojuego IceCream, en la plataforma del cliente. A continuación se presentan los resultados obtenidos en cada una de las etapas de desarrollo del videojuego:

A. **Creación del GDD:** En la primera semana de desarrollo se creó el documento GDD siguiendo las rúbricas de la empresa, donde se definió :
   - Título: Ice Cream
   - Aspectos técnicos: desarrollado en unity para plataforma Web HTML5 con envío de estadísticas a servicios REST.
   - Duración: 10 minutos aproximadamente.
   - Aspecto formativo: evaluar memoria y combinación adecuada de elementos.
   - Jugadores objetivo: 15 a 60 años.
   - Resumen historia: un joven en una búsqueda desesperada de trabajo es contratado en la heladería y debe mantener su reputación sino será despedido.
   - Referencias: Diner Dash, Cooking Diary, Great Pizza
   - Referencias Gráficas:Cooking Joy (**Figura 3**), Cooking Family ,StarChef.

Figura 3. Videojuego Cooking Joy

B. **Diseño de la arquitectura:** entre los principales componentes de la arquitectura, se encuentra el dispositivo del usuario para ingresar al navegador, ya sea por móvil o escritorio, el servidor Azure del cliente y los videojuegos en Unity que estarán almacenados en él. Al momento del cliente ingresar al navegador, el servidor azure, por medio de internet, entregará el videojuego solicitado con la información del usuario. Ver **Figura 4**.

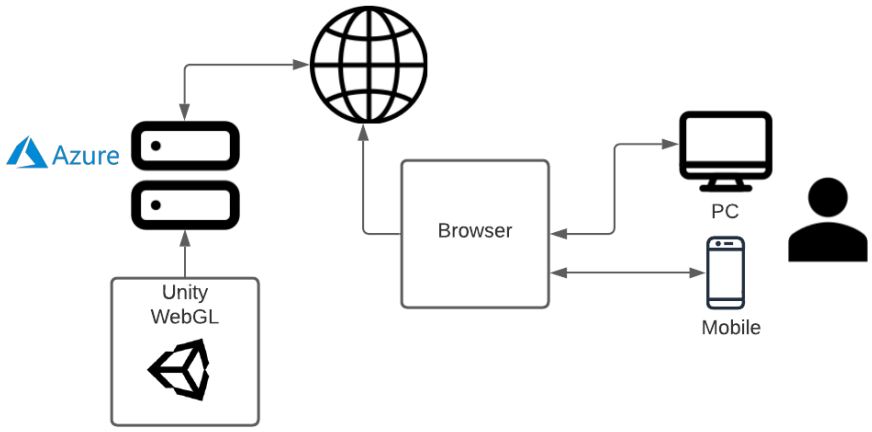

Figura 4. Arquitectura Videojuegos Teleperformance

Teniendo en cuenta lo anterior y con la ayuda de la herramienta FileZilla, el equipo de desarrollo sólo tendrá que subir un compilado del videojuego compatible para WebGL en la plataforma Azure. El mantenimiento de los servidores es responsabilidad propia de los clientes.

- **Diagrama de Clases:** en este diagrama se presentan las principales clases a implementar, en la **Figura 5** se puede observar el diagrama que consistirá de un controlador principal y un conjunto de sistemas que una vez ejecutadas las responsabilidades designadas, notifica al controlador principal para continuar el flujo del videojuego. Con base en el diagrama anterior, se presentan las principales clases a implementar:

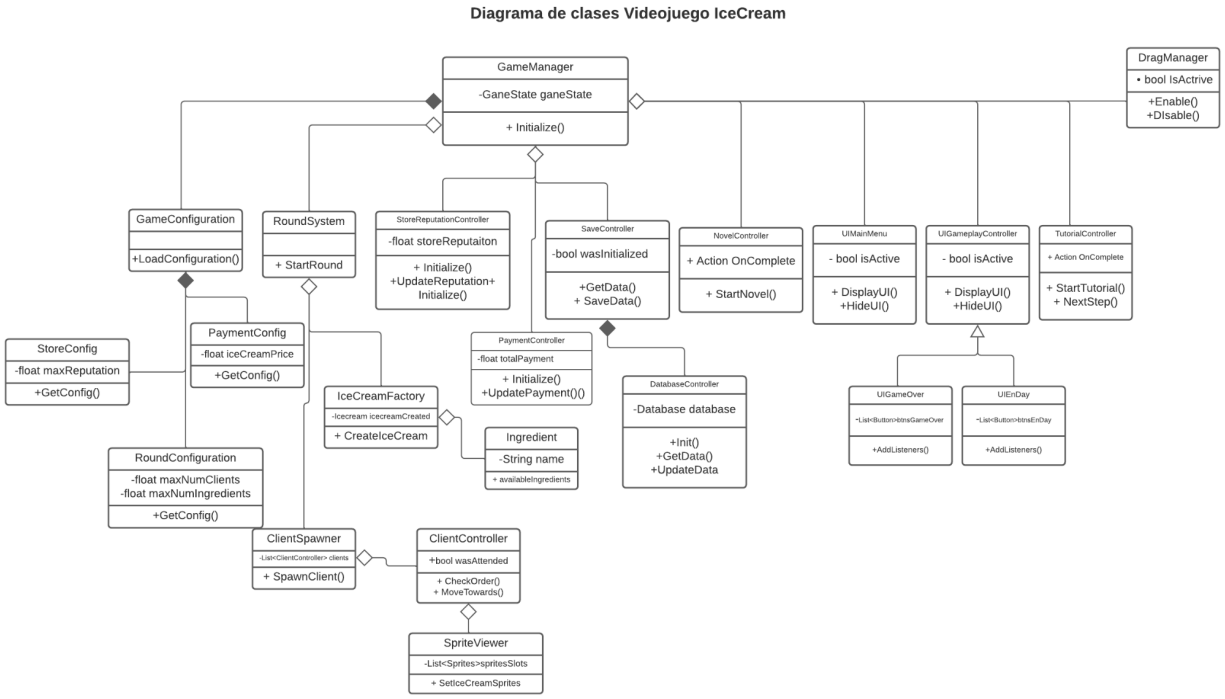

Figura 5. Diagrama de clases videojuego Ice Cream

- **GameManager:** se encarga de inicializar los demás sistemas y controlar el flujo principal del videojuego.
- **StoreReputationController:** maneja la reputación de la tienda a lo largo de los niveles
- **PaymentController:** controla el pago que realicen los clientes en la tienda lo largo de los niveles
- **GameConfiguration:** scriptableObject que contiene la configuración principal del videojuego
- **RoundConfiguration:** scriptableObject que contiene la configuración de un round
- **DragSystem:** sistema Drag, él cual tiene la mecánica principal del videojuego para ir armando el helado
- **Ingredient:** scriptableObject que contiene la descripción general de cada ingrediente
- **ClientSpawner:** generador de Clientes en escena.
- **ClientController:** cliente de la heladería, se encarga de realizar el pedido y de validar la orden.
- **IceCreamFactory:** fabrica de helados, genera helados de forma aleatoria
- **SpriteViewer:** coloca en escena los sprites correspondientes al helado que el cliente solicita
- **IceCream:** scriptableObject que contiene los ingredientes del helado.

- **TutorialController:** sistema para enseñar al jugador a interactuar con el videojuego.
- **SaveManager:** es el puente entre la base de datos y los datos que se generan en el juego, donde se encarga de guardar y cargar los datos en el juego.
- **DatabaseController:** Se comunica con la base de datos del cliente.

C. **Implementación Arquitectura:**

Utilizando la herramienta de Unity y siguiendo las rúbricas de la empresa, se crea la estructura base del proyecto, como se observa en la **Figura 6**, la cual indica cada uno de los elementos principales del videojuego, como Cámaras, Personajes, Luces, Sonidos, Controladores, Interfaz de Usuario, entre otros.

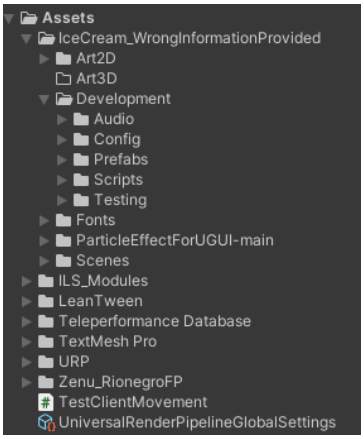

Figura 6: Estructura del proyecto

- **Desarrollo de Componentes:** para el desarrollo de la mayoría de los componentes mencionados en la **Figura 5** , se siguió el flujo que se puede observar en la **Figura 7**, que consta de los siguientes pasos:

Figura 7. Pasos para crear un componente

1. **Desarrollo Scripts:** se desarrollan los scripts para implementar el componente y se añaden en escena los objetos necesarios para su funcionamiento, un ejemplo de esta última parte se puede observar en la **Figura 7.1**

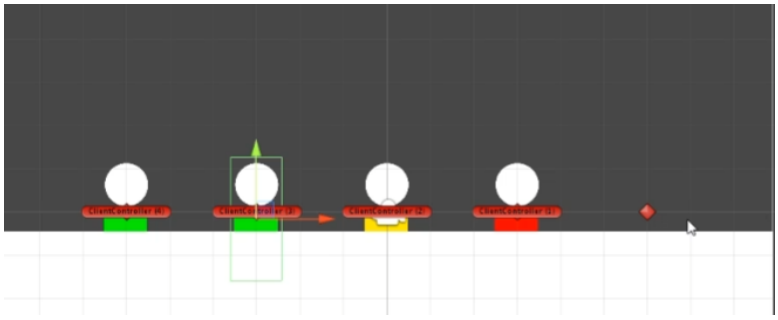

Figura 7.1 Clientes en escena pertenecientes al ClientSpanwer

2. **Creación Prefab** : se crea un objeto con la funcionalidad desarrollada, actuando como una plantilla a partir de la cual se pueden crear nuevas instancias del objeto en la escena, por ejemplo el Prefab que se muestra en la **Figura 7.2**, representa un cliente que puede usarse como base para desarrollar otras variantes de clientes.

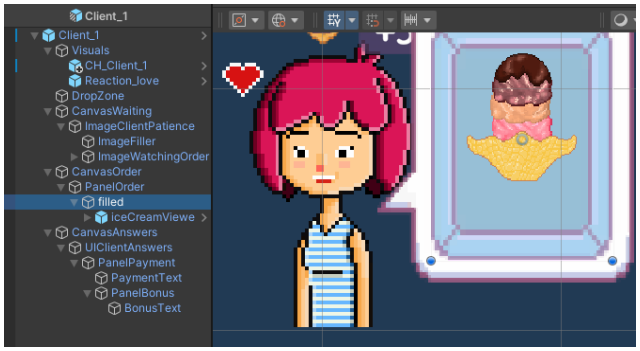

Figura 7.2. Prefab de un cliente

3. **Desarrollo de scripts de testing:** utilizando Unity Test Framework se configuran los scripts para probar el componente, una vez se desarrollan los scripts que prueben los módulos, se crea y se configura un paquete con ellos, para que se cargue al inicio del videojuego , como se evidencia en la **Figura 7.3**,

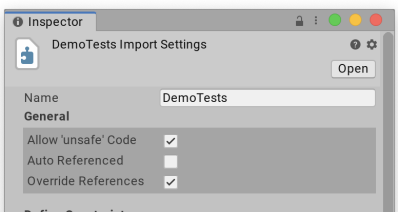

Figura 7.3. Configuración para iniciar el videojuego en modo testing

4. **Realizar pruebas unitarias:** se ejecutan cada uno de las pruebas unitarias, que valida si todos los componentes están funcionando correctamente, en caso de que no sea exitoso se vuelve al paso uno para hacer ajustes. Un ejemplo se puede observar en la **Figura 7.4**

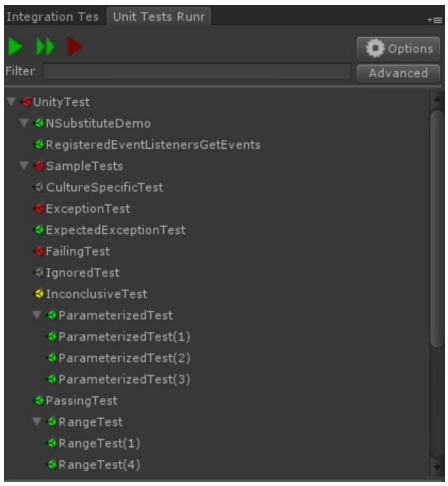

Figura 7.4. Resultados test ClientSpawner

5. Una vez el componente está funcional, se añade en escena junto a los demás, como se puede observar en la **Figura 7.5**.

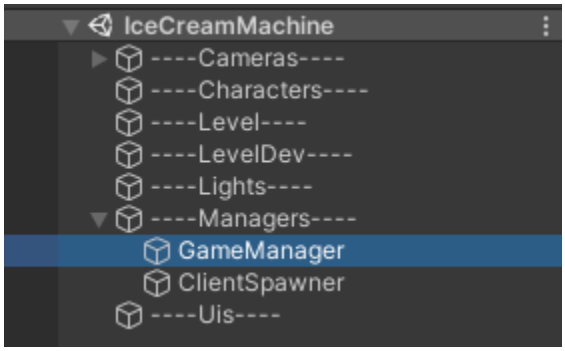

Figura 7.5. componentes en escena

D. Creación versiones funcionales: con el fin de presentar una versión funcional del videojuego al cliente a lo largo del proyecto, se crearon tres versiones : Alfa,Beta y Candidata. Cada una de ellas es descrita a continuación:

- **Versión Alfa:** se implementaron algunos elementos creados por el equipo de arte, con el objetivo de crear un prototipo para presentarle al cliente una idea base de cómo se vería el videojuego y como el usuario interactuaría con él, ver **Figura 8**.Una vez obtenida la aprobación del cliente, se añadieron los demás elementos de la interfaz de usuario y sistemas necesarios para completar la versión alpha del videojuego, la cual fue desplegada en la plataforma del cliente

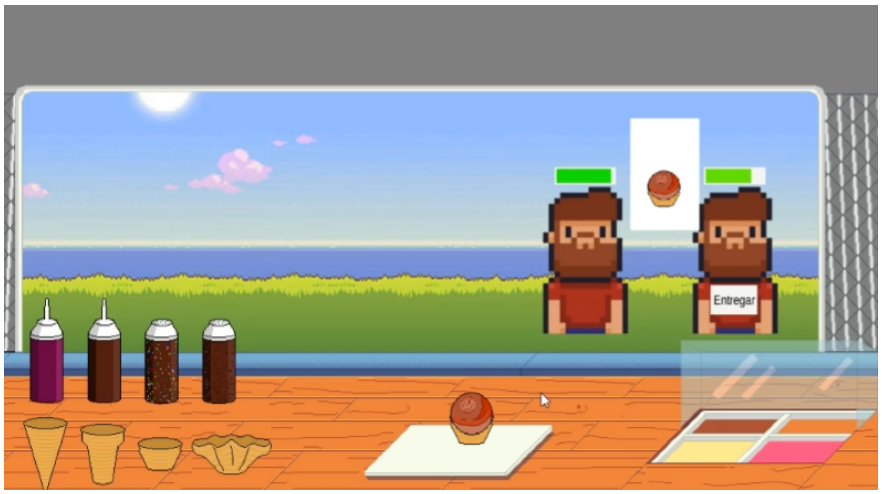

Figura 8. Visualización del gameplay del videojuego en su versión alfa.

- **Versión Beta:** en esta versión, ver **Figura 8.1**, se desarrollaron funcionalidades adicionales como un sistema de guardado, se añadieron más niveles y mecánicas como poder visualizar la orden del cliente de nuevo y se implementaron todos los elementos creados por el equipo de arte y se corrigieron los errores que fueron reportados por el cliente.

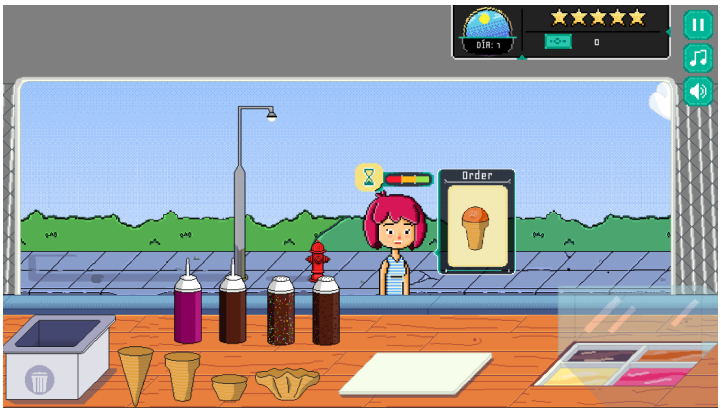

Figura 8.1 Visualización del gameplay del videojuego en su versión beta.

- **Versión Candidata**: esta es la versión final del videojuego, donde ya se arreglaron todos los errores, se añadieron funcionalidades adicionales como tener la oportunidad de ganar tiempo extra o entregar varias órdenes de forma simultánea, entre otras. Además se actualizaron algunos elementos por parte del equipo de arte, como se puede evidenciar en la **Figura 8.2** y la **Figura 8.3**

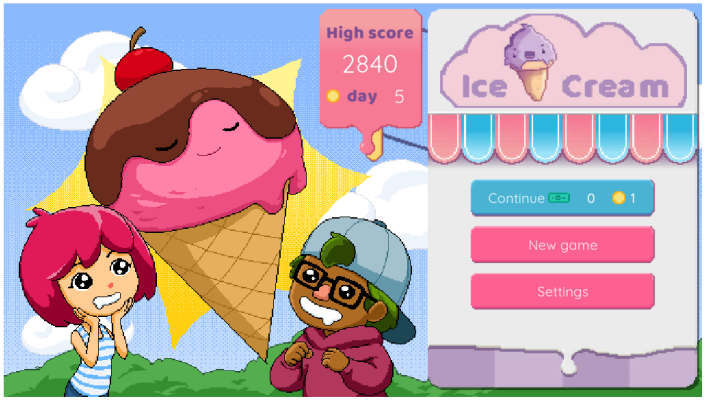

Figura 8.2. Menú principal del videojuego en su versión candidata.

Figura 8.3. Visualización del gameplay del videojuego en su versión candidata.

D. Definición y asignación de actividades :

Con la ayuda de la plataforma ClickUp y la productora a cargo del proyecto, se realiza la asignación de responsabilidades y fechas estimadas. En el siguiente diagrama Gantt, ver **Figura 9**, se evidencia cada una de las actividades a realizar durante cada uno de los

meses que dura el proyecto, las lineas de color verde indican actividades de desarrollo y las lineas azuales, son tares del equipo de arte:

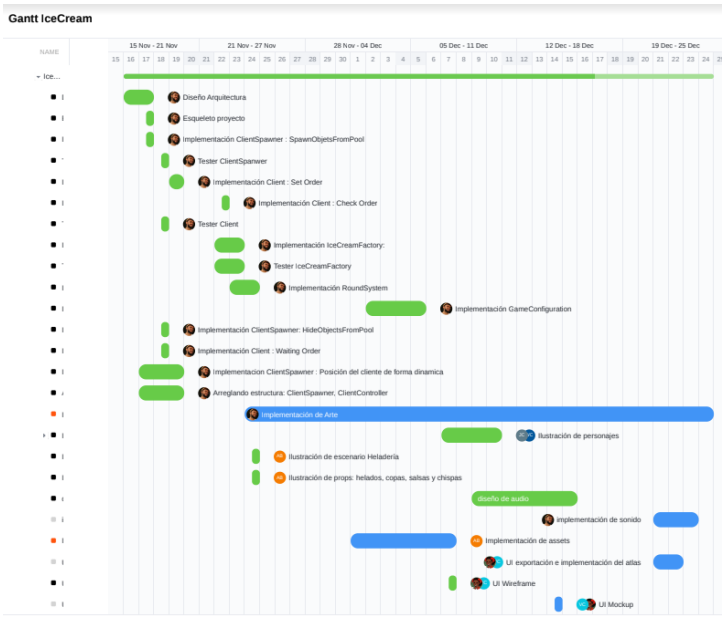

Figura 9. Diagrama de Actividades

5 Análisis

- **Creación del GDD:** fue de crucial importancia la creación de este documento, ya que al definir las características del videojuego, ayudaba a delimitar las funcionalidades que se solicitaban, permitiendo que los objetivos centrales del videojuego no se perdieran.

- **Arquitectura:** al tener en cuenta los principios de diseño SOLID, especialmente el de responsabilidad única y abierto/cerrado, ayudó en el momento en que se reportaron fallas en el aplicativo, encontrar con facilidad la clase responsable y hacer el ajuste, sin tocar los demás componentes. También, cuando se solicitaron nuevas funcionalidades, permitió extender las clases, sin modificar el código base, logrando que al realizar los test automatizados y manuales, no se tuviera que probar todo el videojuego.

- **Versionado de software:** durante la entrega de resultados, versionar el software, al menos en Alpha, Beta y Candida, permitió realizar compilados de cada una de forma rápida, logrando compararla de forma más precisa. También, al momento de realizar nuevas funcionalidades, si al cliente no le gustaban o presentaban fallas, simplemente se descartaban y se volvía a la versión inmediatamente anterior, permitiendo tener siempre una versión estable del videojuego.

- **Metodología de desarrollo:** ayudó a tener una constante comunicación con el cliente, permitiendo que él pudiera ver y realizar sugerencias sobre los resultados que se estaban obteniendo durante la etapa de desarrollo, con el propósito de que el producto final cumpliera los requerimientos solicitados.

6 Conclusiones

- Fue posible concluir con éxito el proyecto trabajando de forma remota, gracias a la continua comunicación a través de las herramientas con las que cuenta la compañía para este fin, ya que ayudó a una rápida solución ante las dudas o problemas presentados durante la etapa de desarrollo.

- Los testing automatizados ayudan a agilizar el desarrollo, especialmente para completar las tareas en los tiempos estimados. Pero los testing manuales fueron más importantes para asegurar la calidad del software, ya que a pesar de tener poca cobertura de los errores que se presentaron, logró solucionar los escenarios más complejos, permitiendo entregar un producto mucho más estable al cliente.

- Las asignaturas del pensum académico, fueron la base para asumir los principales retos presentados durante las prácticas, especialmente la idea de detenerse a pensar, antes de escribir código, desarrollando de una forma más comprensible y eficaz. Pero es importante profundizar más en temas de calidad de software, trabajo en equipo y realizar una continua práctica del análisis y diseño de sistemas a lo largo del pregrado. Ya que las empresas al manejar unos altos estándares de calidad, es valioso el ejercicio de motivar al estudiante que interiorice la idea de que aunque sí es importante que el software funcione, es más relevante enfocarse en que hay que hacerlo bien, siendo rápido, dándole al software flexibilidad que merece, con un código lo suficientemente entendible no para una máquina, sino para un ser humano.

Referencias

Smith J. (2021). *About Teleperformance*. https://www.teleperformance.com//about-tp/

ClickUp.(2021). https://www.clickup.com/

IceCream Videogame.(2021). https://icecreammachine.azurewebsites.net/

Indie Level Studio .(2019) *Portafolio*. https://indielevelstudio.com/portfolio-category/indie/

Chet Copra (2019) Medium. Pathfinding Algorithms. https://medium.com/swlh/pathfinding-algorithms-6c0d4febe8fd

Tim Ryan (1999) . The Anatomy of a Design Document. Gamasutra.

Rajlich, V. T., & Bennett, K. H. (2000). A staged model for the software life cycle. Computer, 33(7), 66–71. doi:10.1109/2.869374

Unity (2021). https://unity.com/

Unity Test Framework (2021). *Unity Docs*. https://docs.unity3d.com/Packages/com.unity.test-framework@1.1/manual/index.html

Brown A. (2022). *About GitLab*. https://about.gitlab.com/

FileZilla (2021). https://filezilla-project.org/

Rider JetBrains (2021). https://www.jetbrains.com/es-es/lp/dotnet-unity/