PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad Académica de Formación Técnica y Tecnológica – PUCE TEC

DESARROLLO DE UN VIDEOJUEGO EDUCATIVO PARA EL APRENDIZAJE INTERACTIVO DE LA HISTORIA DEL ECUADOR

Proyecto de titulación previo a la obtención del título de tecnólogo en desarrollo de software

Autor:

Freire Zambrano Jose Ismael

Tutor:

José Luis Galarraga Cañizares

Quito, Ecuador

2025

Tabla de contenidos

Lista de ilustraciones

DECLARACIÓN y AUTORIZACIÓN

Yo, Jose Ismael Freire Zambrano con C.I. 1727027110 autor(a) del trabajo de titulación intitulado: **"Desarrollo de un videojuego educativo para el aprendizaje interactivo de la historia del ecuador"**, previa a la obtención del título de **tecnólogo en desarrollo de software** en la Unidad Académica de Formación Técnica y Tecnológica PUCE TEC:

1.- Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través de sitio web de la Biblioteca de la PUCE el referido trabajo de titulación, respetando las políticas de propiedad intelectual de Universidad.

Quito, febrero 2025

Jose Ismael Freire Zambrano

C.I. 1727027110

Agradecimientos

A lo largo de este camino lleno de aprendizajes, desafíos y momentos inolvidables, muchas personas han sido parte fundamental de este logro. Esta tesis no es solo el resultado de mi esfuerzo, sino también del apoyo, la guía y el cariño de quienes han estado a mi lado en cada paso del camino.

A Dios, por darme la vida, la salud y la fortaleza para seguir adelante, incluso en los momentos más difíciles. Su luz ha sido mi guía y su presencia me ha dado paz en los momentos de incertidumbre.

A mi familia, por su amor y compañía en cada etapa de mi vida. Gracias por estar siempre ahí, por su confianza en mí y por hacerme sentir que nunca estoy solo en este camino.

A mi novia, Andreina, quien ha sido mi mayor fuente de inspiración y mi apoyo inquebrantable. Gracias, amor, por tu paciencia infinita, por creer en mí incluso cuando yo dudaba, por cada gesto de cariño que me motivó a seguir adelante y por ser mi refugio en los momentos más difíciles. Tu amor y comprensión han sido un regalo invaluable en este proceso.

A mis amigos, por su amistad sincera y su compañía en cada etapa de esta aventura. Gracias por compartir desvelos, largas horas de estudio, risas y palabras de ánimo. Su apoyo ha hecho que este camino sea más llevadero y lleno de momentos inolvidables.

A mis profesores y mentores, por compartir su conocimiento y enseñarme mucho más que teorías y conceptos. Gracias por su paciencia, su guía y por motivarme a dar siempre lo mejor de mí. Cada enseñanza recibida ha dejado huella en mi formación profesional y personal.

A todas aquellas personas que, de una u otra forma, contribuyeron a la realización de este proyecto, ya sea con su conocimiento, apoyo o palabras de aliento. Cada gesto, por pequeño que haya sido, ha significado mucho para mí.

Este logro no es solo mío, sino también de todos ustedes. Gracias por ser parte de este sueño hecho realidad.

Con profundo agradecimiento,

Ismael Freire

Introducción

Planteamiento del Problema

En la actualidad, la enseñanza d la historia enfrenta varios desafíos. Unos de los principales es la falta de interés de los estudiantes hacia los métodos tradicionales de aprendizaje, como las clases expositivas o los textos largos. Esto hace que conceptos importantes, como los eventos históricos del Ecuador no logren captar su atención ni ser retenidos con facilidad. Otro problema que se identifica es la falta de herramientas interactivas que combinen tecnología y educación. A pesar de los avances tecnológicos, todavía existen pocos recursos diseñados específicamente para hacer que la enseñanza sea más atractiva y adaptada a las nuevas generaciones. Este proyecto busca resolver este problema mediante un videojuego educativo que permita a los estudiantes interactuar con los eventos históricos del Ecuador de una manera entretenida, ayudando a mejorar la experiencia de aprendizaje.

Justificación

La historia ecuatoriana es una parte fundamental de nuestra identidad como país, pero muchas veces no se enseña de manera que motive a los estudiantes. Este proyecto tiene como objetivo principal ofrecer una solución que combine aprendizaje y tecnología, adaptándose a los intereses de los jóvenes. El videojuego educativo será una herramienta que no solo hará el aprendizaje más dinámico, sino que también fomentará el pensamiento crítico y la curiosidad por la historia. Además, al tratarse de un recurso digital, podrá ser utilizado tanto en aulas como en entornos virtuales, ampliando su alcance y utilidad.

Objetivos

Objetivo General

Diseñar y desarrollar un videojuego educativo que permita a los estudiantes aprender sobre la historia del Ecuador de manera interactiva y entretenida.

Objetivos Específicos

1. Crear tres niveles interactivos que representes eventos históricos clave, con la Fundación de Quito, La independencia de Guayaquil y la Batalla de Pichincha.
2. Desarrollar un sistema de juego que incluya cronometro y vidas para aumentar el nivel de desafío.
3. Diseñar una interfaz intuitiva y atractiva, utilizando un estilo pixel art.
4. Validar el videojuego con estudiantes para medir su impacto en el aprendizaje.

En este proyecto de tesis se propone desarrollar un videojuego educativo que sea interactivo y que facilite el aprendizaje de la historia ecuatoriana de manera accesible, atractiva y que sea entretenida para el estudiante.

El videojuego, nombrado como "Explorador del Tiempo", permite al estudiante viajar a eventos históricos cruciales, como la Fundación de Quito (1534), la Independencia de Guayaquil (1820) y la Batalla de Pichincha (1822), participando en misiones relacionadas con estos eventos. A través de mecánicas de juego basadas en el tiempo y la exploración, el proyecto lo que busca es que los estudiantes refuercen su comprensión y retención de estos contenidos mediante el videojuego educativo.

Este trabajo se enfoca en estas dos áreas principales, em el levantamiento de requisitos y diseño del sistema, y la implementación del videojuego en Unity esto servirá como base técnica para el proyecto. El juego integra componentes educativos mediante un sistema de vidas en forma de relojes de tiempo y un cronometro que agrega un nivel de desafío para que el jugador pueda arreglar la línea de tiempo a su forma original, incentivando el aprendizaje a través de la superación de retos históricos.

El objetivo del proyecto es incluir en el desarrollo de una interfaz intuitiva y visualmente atractiva mediante gráficos 2D con una vista Top and Down, la implementación de mecánicas de juego van a ir acordes al contenido educativo donde el jugador tendrá que

hablar con personajes de ese escenario para saber que cambio en la historia ya que el villano del juego lo altero y tendiendo conocimiento del evento que se alteró y el jugador tendrá que arreglarlo haciendo que evento ocurra, también detallando cada fase para garantizar que el videojuego cumpla con sus metas pedagógicas.

Capítulo I

Levantamiento de Requisitos y Diseño del Sistema

1.1 Introducción al Levantamiento de Requisitos

El levantamiento de requisitos es fundamental para establecer las funcionalidades necesarias del sistema para el videojuego “Explorador del Tiempo”, el objetivo principal del levantamiento es definir que aspectos educativos e interactivos se requieran para que el sistema lograra su propósito que es mejorar el aprendizaje de a historia ecuatoriana.

1.2 Metodología del Levantamiento de Requisitos

Para la recolección de requisitos, se aplicaron técnicas como entrevistas y encuestas a estudiantes y educadores, lo cual la recopilación incluyo lo siguiente:

- Entrevistas a profesores de historia para definir los contenidos de los niveles.
- Encuestas a estudiantes para entender sus preferencias de interacción y juego.
- Análisis de documentación sobre videojuegos educativos y pedagogía histórica para alinear los objetivos de aprendizajes.

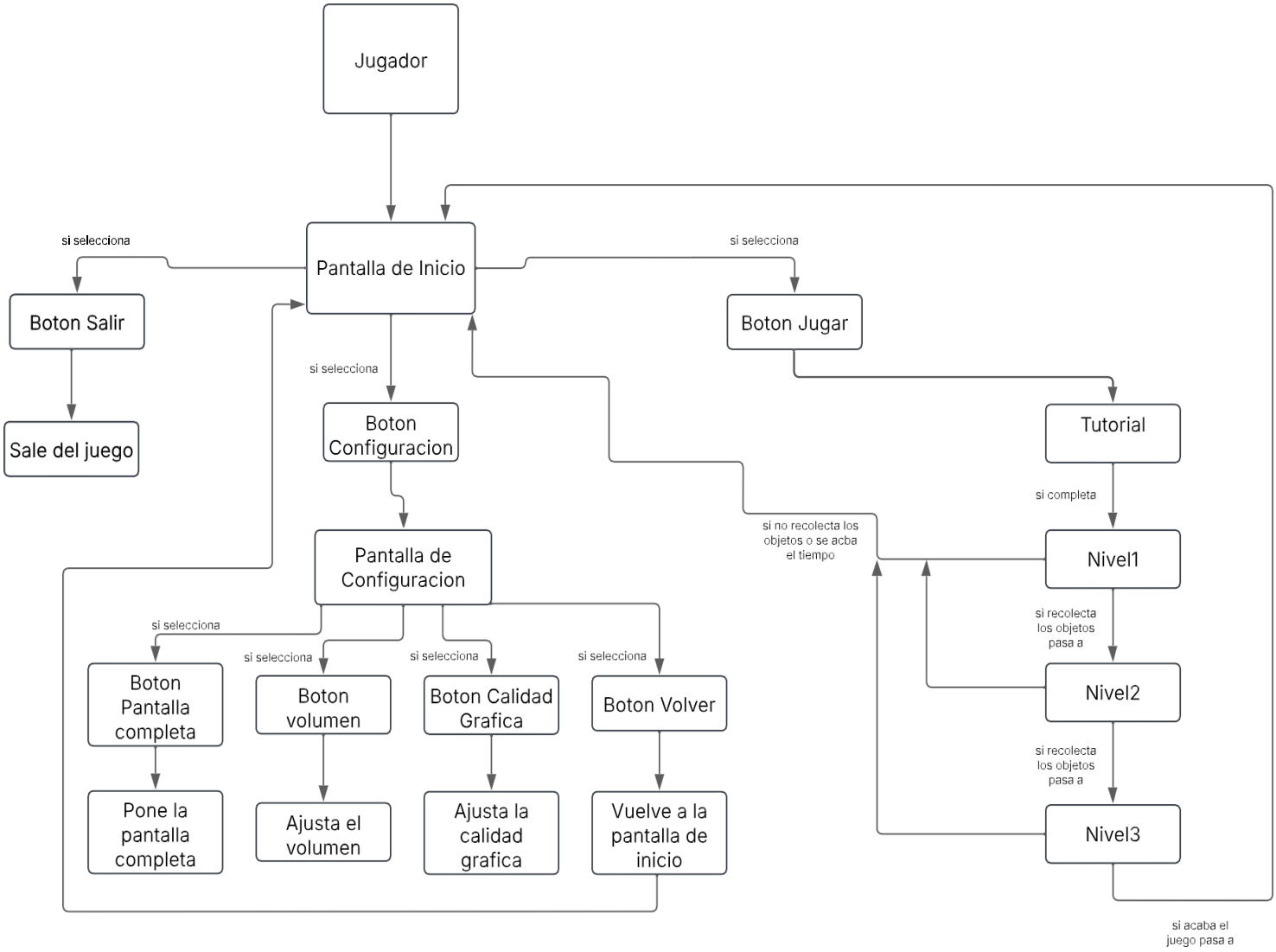
1.3 Resultados del Levantamiento de Requisitos

Modelado del Negocio

El presente diagrama de flujo detalla el funcionamiento del videojuego educativo “Explorador del Tiempo”, diseñado para enseñar la historia del Ecuador a los estudiantes mediante la interacción con personajes históricos y la recuperación de fragmentos de documentos importantes.

El flujo del juego comienza con un menú inicial que permite al jugador seleccionar entre las opciones de jugar, ajustar la configuración o salir del juego. Tras seleccionar la opción

de jugar, el jugador pasa por un tutorial donde aprende las mecánicas básicas del juego. Posteriormente, el jugador es transportado a diferentes épocas históricas del Ecuador para completar misiones específicas.



1. Modelado del Negocio

Pasos Detallados

1. **Menú Inicial:**
    - **Mostrar pantalla de bienvenida:** El juego presenta una pantalla de bienvenida con las opciones “Jugar”, “Configuración”, “Salir”.
    - **Opción: Jugar:** Si el jugador selecciona “Jugar”, es dirigido a la pantalla de tutorial.

- **Opción: Configuración:** Si el jugador selecciona “Configuración”, es dirigido a una pantalla donde puede ajustar la configuración del juego
- **Opción: Salir**: Si el jugador selecciona "Salir", el juego termina.

2. **Configuración:**
    - **Pantalla completa**: El jugador puede activar o desactivar el modo de pantalla completa.
    - **Ajustar volumen**: El jugador puede ajustar el volumen del juego.
    - **Cambiar calidad**: El jugador puede cambiar la calidad gráfica del juego.
    - **Volver al menú inicial**: El jugador puede regresar al menú inicial desde la pantalla de configuración.
3. **Tutorial:**

Hablar con NPC: El jugador interactúa con un NPC que explica las mecánicas básicas del juego.

Activar reloj: Tras completar el tutorial, se activa un reloj que transporta al jugador al primer nivel.

4. **Primer Nivel: Fundación de Quito:**

Encuentro con NPC principal (Sebastián de Benalcázar): El jugador habla con Sebastián de Benalcázar para obtener pistas sobre los fragmentos de documentos.

Explorar zona: El jugador explora la zona y activa nombres flotantes en lugares históricos.

Hablar con otros NPCs: El jugador interactúa con otros NPCs para obtener más pistas sobre los fragmentos.

Reunir fragmentos de documentos: El jugador busca y reúne los fragmentos de documentos necesarios.

Entrega de objeto: Si el jugador encuentra todos los fragmentos, los entrega a Sebastián de Benalcázar.

Progresar al siguiente nivel: Si el jugador completa el nivel, progresa al siguiente nivel.

Tiempo agotado: Si el tiempo se agota antes de completar el nivel, el jugador es dirigido a una pantalla de "Tiempo Agotado" y debe comenzar de nuevo.

5. **Segundo Nivel: Independencia de Guayaquil**:

Encuentro con NPC principal (José Joaquín de Olmedo): El jugador habla con José Joaquín de Olmedo para obtener pistas sobre los fragmentos de documentos.

Explorar zona: El jugador explora la zona y activa nombres flotantes en lugares históricos.

Hablar con otros NPCs: El jugador interactúa con otros NPCs para obtener más pistas sobre los fragmentos.

Reunir fragmentos de documentos: El jugador busca y reúne los fragmentos de documentos necesarios.

Entrega de objeto: Si el jugador encuentra todos los fragmentos, los entrega a José Joaquín de Olmedo.

Progresar al siguiente nivel: Si el jugador completa el nivel, progresa al siguiente nivel.

Tiempo agotado: Si el tiempo se agota antes de completar el nivel, el jugador es dirigido a una pantalla de "Tiempo Agotado" y debe comenzar de nuevo.

6. **Tercer Nivel: Batalla de Pichincha**:

Encuentro con NPC principal (Antonio José de Sucre): El jugador habla con Antonio José de Sucre para obtener pistas sobre los fragmentos de documentos.

Explorar zona: El jugador explora la zona y activa nombres flotantes en lugares históricos.

Hablar con otros NPCs: El jugador interactúa con otros NPCs para obtener más pistas sobre los fragmentos.

Reunir fragmentos de documentos: El jugador busca y reúne los fragmentos de documentos necesarios.

Entrega de objeto: Si el jugador encuentra todos los fragmentos, los entrega a Antonio José de Sucre.

Final del juego: Si el jugador completa el nivel, el juego finaliza y el jugador es enviado de vuelta al menú inicial.

Tiempo agotado: Si el tiempo se agota antes de completar el nivel, el jugador es dirigido a una pantalla de "Tiempo Agotado" y debe comenzar de nuevo.

7. **Final del Juego**:

Regreso al menú inicial: Al finalizar el juego, el jugador es enviado de vuelta a la pantalla de menú inicial.

Diseño del Diagrama Caso de Uso

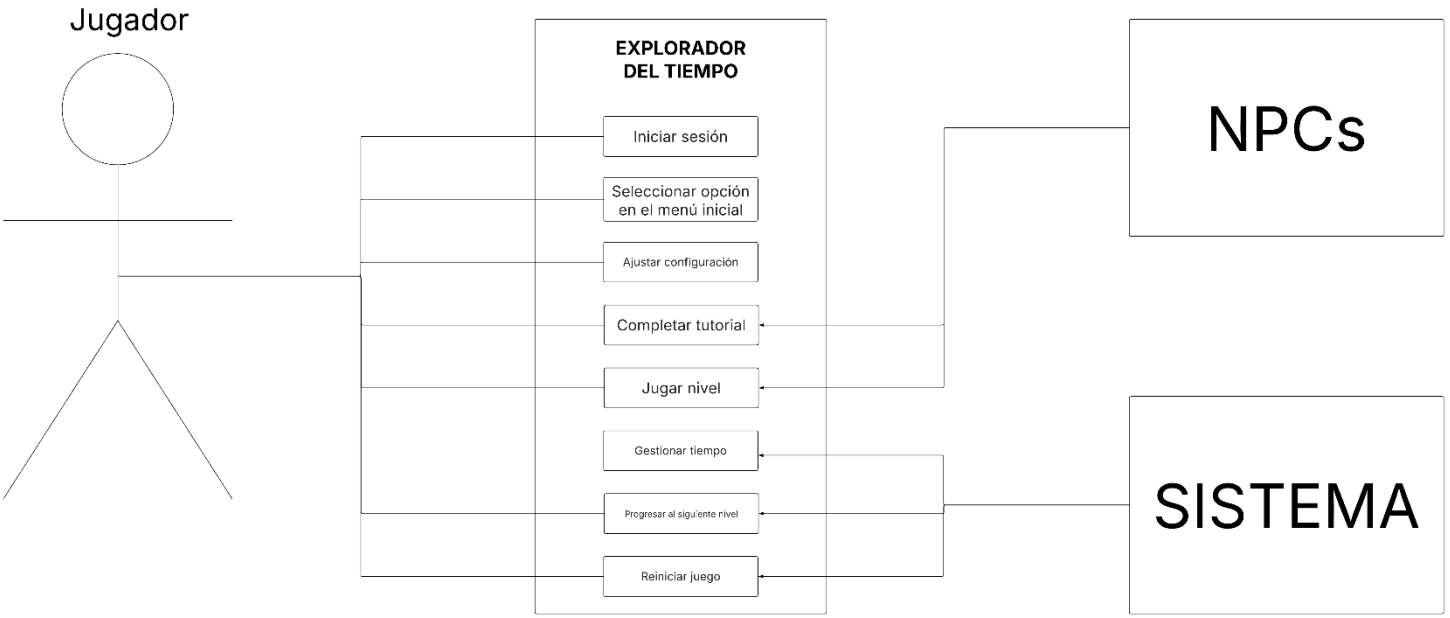
El diagrama de casos de uso proporciona una representación gráfica de las interacciones entre los actores (jugadores, NPCs, sistema) y los casos de uso (funcionalidades del juego). Este diagrama ayuda a visualizar cómo los usuarios interactúan con el sistema y qué funcionalidades están disponibles para ellos.

Diagrama de Casos de Uso:

- **Actores:**
    - Jugador
    - NPC (Personajes del juego)

- Sistema

- **Casos de Uso:**
  - Iniciar sesión
  - Seleccionar opción en el menú inicial
  - Ajustar configuración
  - Completar tutorial
  - Jugar nivel
  - Gestionar tiempo
  - Progresar al siguiente nivel
  - Reiniciar juego



2. Diseño del Diagrama Caso de Uso

Descripción de los Casos de Uso

1. **Iniciar sesión**: El jugador inicia sesión en el juego.
2. **Seleccionar opción en el menú inicial**: El jugador puede elegir entre jugar, ajustar configuración o salir.
3. **Ajustar configuración**: El jugador puede cambiar la configuración del juego (pantalla completa, volumen, calidad) y volver al menú inicial.
4. **Completar tutorial**: El jugador habla con un NPC para aprender las mecánicas del juego y activar el reloj que lo envía al primer nivel.
5. **Jugar nivel**: El jugador explora el nivel, habla con NPCs y reúne fragmentos de documentos.
6. **Gestionar tiempo**: El juego tiene un temporizador que, si se agota, muestra una pantalla de tiempo agotado y reinicia el juego.
7. **Progresar al siguiente nivel**: Si el jugador reúne todos los fragmentos y habla con el NPC principal, progresa al siguiente nivel.
8. **Reiniciar juego**: Si el tiempo se agota, el juego se reinicia y el jugador debe comenzar de nuevo.

1.4 Diseño del Sistema

El diseño del sistema define la arquitectura y los componentes esenciales que se utilizarán dentro del videojuego educativo "Explorador del Tiempo". A continuación, se detallan las tecnologías y herramientas que se emplearán tanto en el frontend como en el backend, así como la descripción de la base de datos.

Tecnologías por Utilizar

Frontend: Unity

- **Descripción**: Unity es una plataforma de desarrollo de juegos que permite crear experiencias interactivas en 2D y 3D. Es ampliamente utilizada debido a su flexibilidad y capacidad para desarrollar juegos multiplataforma.
- **Librerías y Bibliotecas**:
    - **Unity UI Toolkit**: Para la creación de interfaces de usuario interactivas y responsivas.
    - **Cinemachine**: Para la gestión avanzada de cámaras y efectos cinematográficos.
    - **TextMesh Pro**: Para la renderización de texto de alta calidad.
    - **Unity Analytics**: Para el seguimiento y análisis del comportamiento del jugador.
- **Uso de la Herramienta**:
    - **Desarrollo de Niveles**: Crear y diseñar los diferentes niveles del juego, incluyendo la Fundación de Quito, la Independencia de Guayaquil y la Batalla de Pichincha.
    - **Interacción con NPCs**: Implementar la lógica de interacción entre el jugador y los personajes históricos.
    - **Gestión de Objetos**: Manejar la búsqueda y recolección de fragmentos de documentos.

- **Interfaz de Usuario**: Diseñar menús, pantallas de configuración y elementos de HUD (Head-Up Display).

Backend: PlayFab

- **Descripción**: PlayFab es una plataforma de backend como servicio (BaaS) que proporciona una infraestructura robusta para juegos, incluyendo gestión de datos, autenticación de usuarios, y análisis.
- **Finalidad del Uso**:
  - **Conexión con Base de Datos**: Almacenar y gestionar datos de jugadores, progreso del juego, y estadísticas.
  - **Autenticación de Usuarios**: Implementar sistemas de inicio de sesión y gestión de cuentas.
  - **Protocolos de Seguridad**: Asegurar la integridad de los datos y proteger la información sensible de los jugadores mediante cifrado y autenticación segura.
  - **Análisis y Telemetría**: Recopilar y analizar datos sobre el comportamiento del jugador para mejorar la experiencia de juego.

Almacenamiento de Datos

Base de Datos: Firebase Firestore

- **Descripción**: Firebase Firestore es una base de datos NoSQL en tiempo real que permite almacenar y sincronizar datos entre los clientes y el backend de manera eficiente.
- **Uso en el Juego**:

- **Almacenamiento de Datos del Jugador**: Guardar información sobre el progreso del jugador, fragmentos de documentos recolectados, y estadísticas de juego.
- **Sincronización en Tiempo Real**: Asegurar que los datos del jugador se actualicen en tiempo real, permitiendo una experiencia de juego fluida y coherente.
- **Escalabilidad**: Manejar grandes volúmenes de datos y usuarios simultáneos sin comprometer el rendimiento.
- **Seguridad**: Implementar reglas de seguridad para controlar el acceso a los datos y proteger la información sensible.

1.4.1 Arquitectura del Sistema

Se seleccionó una arquitectura cliente-servidor para el videojuego educativo "Explorador del Tiempo", permitiendo una experiencia de usuario eficiente y escalable. Esta arquitectura consta de un frontend desarrollado en Unity y un backend que gestiona los datos y el progreso del jugador.

Funcionamiento del Juego con la Arquitectura Cliente-Servidor

1. **Cliente (Frontend)**:
    - **Descripción**: El cliente es la aplicación de juego desarrollada en Unity que actúa como la interfaz principal para el jugador.

- **Interacción con el Usuario**: El jugador interactúa con el juego a través de la interfaz de usuario, que incluye menús, niveles, personajes y elementos interactivos.
- **Conexión al Servidor**: Al iniciar el juego, el cliente establece una conexión con el servidor para autenticar al jugador y sincronizar los datos.

2. **Servidor (Backend)**:
    - **Descripción**: El servidor es responsable de procesar las operaciones del juego y almacenar la información relevante. Utiliza PlayFab para la gestión de datos y Firebase Firestore como base de datos.
    - **Procesamiento de Operaciones**: Una vez establecida la conexión, el servidor maneja todas las operaciones posteriores, como la autenticación de usuarios, el seguimiento del progreso del juego, y la gestión de estadísticas.
    - **Almacenamiento de Información**: El servidor almacena información sobre el progreso del jugador, fragmentos de documentos recolectados, y estadísticas de juego en Firebase Firestore.

Flujo de Datos en la Arquitectura Cliente-Servidor

1. **Inicio de Sesión**:
    - El jugador inicia sesión en el juego desde el cliente (Unity).
    - El cliente envía las credenciales al servidor (PlayFab) para autenticación.
    - El servidor verifica las credenciales y responde con la confirmación de inicio de sesión.

2. **Sincronización de Datos**:

    - Una vez autenticado, el cliente solicita los datos del jugador (progreso, estadísticas) al servidor.
    - El servidor recupera los datos de Firebase Firestore y los envía al cliente.
    - El cliente actualiza la interfaz de usuario con los datos recibidos.

3. **Interacción en el Juego**:

    - Durante el juego, el jugador interactúa con NPCs, recolecta fragmentos de documentos y completa misiones.
    - El cliente envía las acciones del jugador al servidor para su procesamiento.
    - El servidor actualiza los datos del jugador en Firebase Firestore y envía la confirmación al cliente.

4. **Gestión del Progreso**:

    - Al completar un nivel, el cliente envía la información de progreso al servidor.
    - El servidor actualiza el estado del jugador en Firebase Firestore y envía la confirmación al cliente.
    - Si el tiempo se agota, el servidor notifica al cliente para mostrar la pantalla de "Tiempo Agotado".

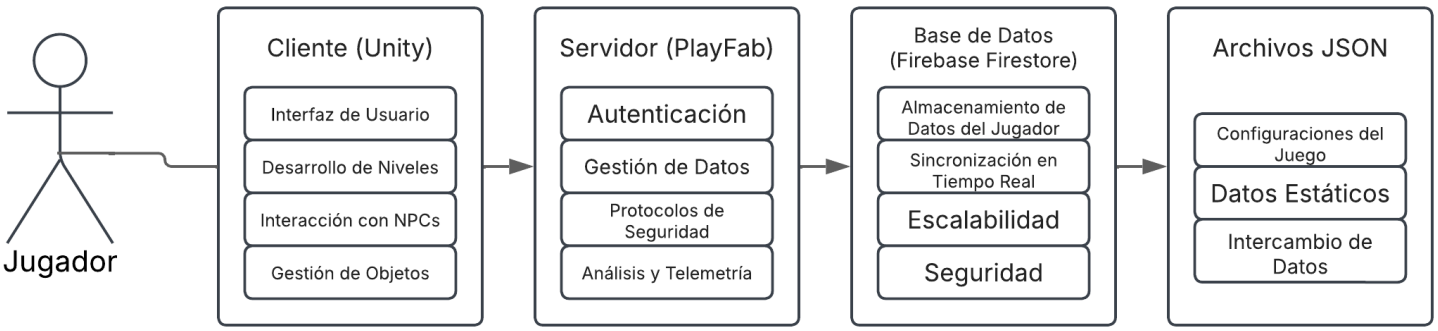
5. **Final del Juego**:

    - Al finalizar el juego, el cliente envía una solicitud al servidor para guardar el progreso final.

- El servidor almacena los datos finales en Firebase Firestore y envía la confirmación al cliente.
- El cliente redirige al jugador al menú inicial.

Diagrama de Arquitectura del Sistema

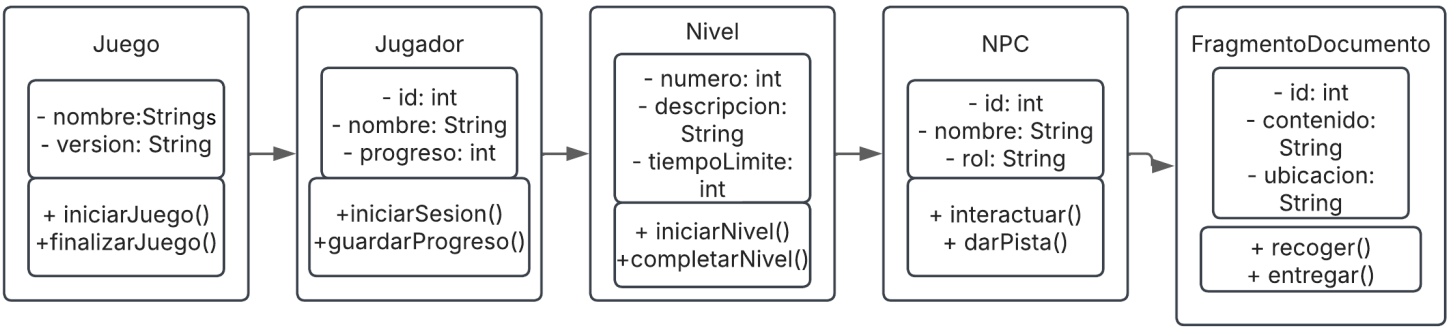
El diagrama de arquitectura del sistema para el videojuego educativo "Viajero del Tiempo" ilustra cómo se integran y funcionan los componentes tecnológicos clave, incluyendo el frontend, el backend, la base de datos y los archivos JSON. A continuación, se describe cada uno de estos componentes y su interacción dentro del sistema.



3. Diagrama de Arquitectura del Sistema

Diagrama de Clases

El diagrama de clases para el videojuego educativo "Explorador del Tiempo" detalla las diversas interacciones y relaciones entre las clases que componen la lógica interna del juego. A continuación, se presenta una descripción técnica de las clases principales y sus relaciones.



4. Diagrama de Clases

Clases Principales y Relaciones

1. **Clase Juego**
    - **Atributos:**
        - nombre: String
        - version: String
    - **Métodos:**
        - iniciarJuego()
        - finalizarJuego()
    - **Relaciones:**
        - Asociación con Jugador

- Asociación con Nivel

2. **Clase Jugador**

   - **Atributos:**

     - id: int
     - nombre: String
     - progreso: int

   - **Métodos:**

     - iniciarSesion()
     - guardarProgreso()

   - **Relaciones:**

     - Asociación con Juego
     - Asociación con Nivel
     - Asociación con FragmentoDocumento

3. **Clase Nivel**

   - **Atributos:**

     - numero: int
     - descripcion: String
     - tiempoLimite: int

   - **Métodos:**

     - iniciarNivel()

        - completarNivel()
    - **Relaciones:**
        - Asociación con Juego
        - Asociación con Jugador
        - Asociación con NPC
        - Asociación con FragmentoDocumento

4. **Clase NPC**
    - **Atributos:**
        - id: int
        - nombre: String
        - rol: String
    - **Métodos:**
        - interactuar()
        - darPista()
    - **Relaciones:**
        - Asociación con Nivel

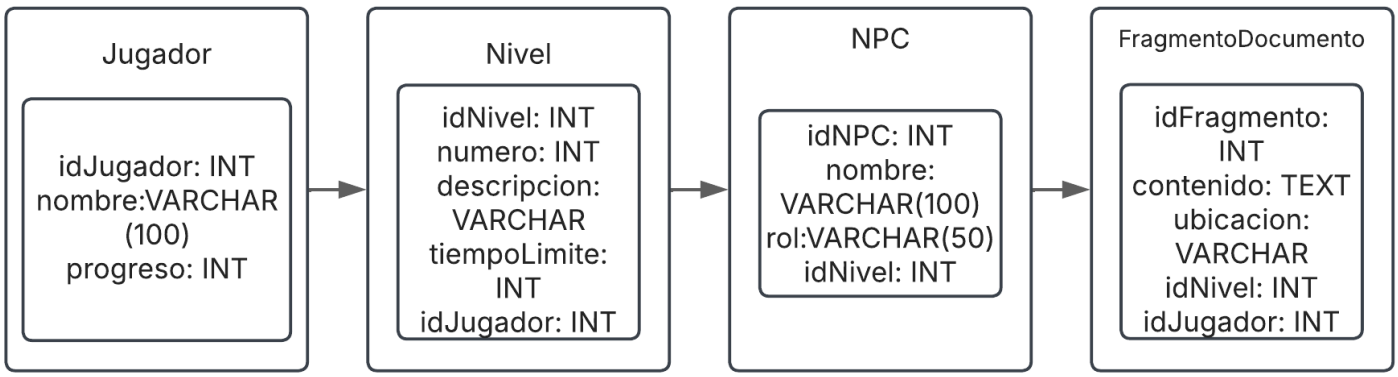
5. **Clase FragmentoDocumento**
    - **Atributos:**
        - id: int
        - contenido: String

- ubicacion: String

- **Métodos:**
  - recoger()
  - entregar()
- **Relaciones:**
  - Asociación con Jugador
  - Asociación con Nivel

Diagrama de Base de Datos

El diseño de la base de datos se realiza utilizando el modelo entidad-relación (ER), que define las entidades, sus atributos y las relaciones entre ellas. A continuación, se presenta el modelo ER con las tablas, campos y relaciones.



5. Diagrama de Base de Datos

Modelo Entidad-Relación

1. **Entidad Jugador**
   - **Campos**:

- idJugador: INT (Clave primaria)
- nombre: VARCHAR(100)
- progreso: INT

2. **Entidad Nivel**
    - **Campos**:
        - idNivel: INT (Clave primaria)
        - numero: INT
        - descripcion: VARCHAR(255)
        - tiempoLimite: INT
        - idJugador: INT (Clave foránea)
3. **Entidad NPC**
    - **Campos**:
        - idNPC: INT (Clave primaria)
        - nombre: VARCHAR(100)
        - rol: VARCHAR(50)
        - idNivel: INT (Clave foránea)
4. **Entidad FragmentoDocumento**
    - **Campos**:
        - idFragmento: INT (Clave primaria)
        - contenido: TEXT

- ubicacion: VARCHAR(255)
- idNivel: INT (Clave foránea)
- idJugador: INT (Clave foránea)

Relaciones

- **Jugador - Nivel**: Un jugador puede tener múltiples niveles asociados (1:N).
- **Nivel - NPC**: Un nivel puede tener múltiples NPCs asociados (1:N).
- **Nivel - FragmentoDocumento**: Un nivel puede tener múltiples fragmentos de documentos asociados (1:N).
- **Jugador - FragmentoDocumento**: Un jugador puede recolectar múltiples fragmentos de documentos (1:N).

1.4.2 Diseño de la Interfaz de Usuario

La interfaz de usuario (UI) del videojuego educativo "Viajero del Tiempo" está diseñada para proporcionar una experiencia intuitiva y atractiva para los jugadores. A continuación, se describen los componentes principales de la UI, se presentan capturas de pantalla de las diferentes pantallas del juego y se incluye un diagrama que muestra cómo están organizadas estas pantallas.

Componentes de la UI

1. **Pantalla de Inicio**:
    - **Descripción**: La pantalla de inicio presenta las opciones de juego y configuración.
    - **Opciones**:

- Jugar
- Configuración
- Salir

2. **Pantalla de Configuración**:
    - **Descripción**: La pantalla de configuración permite al jugador ajustar opciones como pantalla completa, volumen y calidad gráfica.
    - **Opciones**:
        - Pantalla completa
        - Volumen
        - Calidad gráfica
        - Volver al menú inicial
3. **Pantalla de Tutorial**:
    - **Descripción**: La pantalla de tutorial introduce al jugador a las mecánicas básicas del juego mediante la interacción con un NPC.
    - **Elementos**:
        - Interacción con NPC
        - Activación del reloj
4. **Pantallas de Misión para Cada Nivel Histórico**:
    - **Descripción**: Cada nivel histórico tiene su propia interfaz de misión, incluyendo un cronómetro y un medidor de vidas.

- **Niveles**:
    - Nivel 1: Fundación de Quito
    - Nivel 2: Independencia de Guayaquil
    - Nivel 3: Batalla de Pichincha

5. **Pantalla de Tiempo Agotado**:
    - **Descripción**: Esta pantalla aparece cuando el tiempo se agota antes de completar un nivel.
    - **Opciones**:
        - Volver al menú inicial

Diagrama de las Diferentes Pantallas Diseñadas

El siguiente diagrama muestra cómo están organizadas y conectadas las diferentes pantallas del juego:

Pantalla de Inicio

6. Diagrama de las pantallas

Pantalla de Configuración

Pantalla de Tutorial

Pantalla de Tiempo Agotado

Pantallas de Misión para Cada Nivel Histórico

Descripción del Diagrama de Pantallas:

1. **Pantalla de Inicio**:

    - **Descripción**: La pantalla de inicio es la primera pantalla que ve el jugador al iniciar el juego. Presenta las opciones de Jugar, Configuración y Salir.

    - **Transiciones**:

- Al seleccionar "Jugar", el jugador es dirigido a la Pantalla de Tutorial.
- Al seleccionar "Configuración", el jugador es dirigido a la Pantalla de Configuración.
- Al seleccionar "Salir", el juego se cierra.

2. **Pantalla de Configuración**:
   - **Descripción**: La pantalla de configuración permite al jugador ajustar las opciones del juego, como pantalla completa, volumen y calidad gráfica.
   - **Transiciones**:
     - Al ajustar las configuraciones, el jugador puede volver al menú inicial seleccionando "Volver al menú inicial".
3. **Pantalla de Tutorial**:
   - **Descripción**: La pantalla de tutorial introduce al jugador a las mecánicas básicas del juego mediante la interacción con un NPC.
   - **Transiciones**:
     - Después de completar el tutorial, el jugador es dirigido a la Pantalla de Misión del primer nivel histórico.
4. **Pantallas de Misión para Cada Nivel Histórico**:
   - **Descripción**: Cada nivel histórico tiene su propia interfaz de misión, incluyendo un cronómetro y un medidor de vidas.
   - **Transiciones**:

- Al completar un nivel, el jugador es dirigido al siguiente nivel histórico.
- Si el tiempo se agota, el jugador es dirigido a la Pantalla de Tiempo Agotado.

5. **Pantalla de Tiempo Agotado**:
    - **Descripción**: Esta pantalla aparece cuando el tiempo se agota antes de completar un nivel.
    - **Transiciones**:
        - El jugador puede volver al menú inicial seleccionando "Explorador del Tiempo".

1.4.3 Requisitos Mínimos de Hardware y Software

Requisitos de Hardware

- **Procesador**: CPU con al menos dos núcleos.
- **Memoria RAM**: Mínimo 4 GB.
- **Tarjeta Gráfica**: GPU integrada compatible con DirectX 10.
- **Almacenamiento**: Al menos 2 GB de espacio libre en disco.
- **Resolución de Pantalla**: Mínimo 1280x720 píxeles.

Requisitos de Software

- **Sistema Operativo**: Windows 7 o superior, macOS 10.12 o superior.
- **Motor de Juego**: Unity 2020.3 LTS o superior.
- **Bibliotecas y Dependencias**:

- Unity UI Toolkit
- TextMesh Pro
- Unity Analytics
- Photon Unity Networking (PUN) (si es necesario)

1.4.4 Consideraciones Éticas y de Privacidad

Consideraciones Éticas

- **Representación Justa y Equitativa**:
  - Asegurar que el juego represente de manera justa y precisa a personas de diferentes géneros, etnias y orientaciones sexuales.
  - Evitar la promoción de estereotipos negativos o discriminación.
- **Contenido Apropiado**:
  - Evitar la inclusión de contenido violento o inapropiado que pueda influir negativamente en el comportamiento de los jugadores.
  - Promover valores positivos como la empatía, la justicia y la colaboración.
- **Responsabilidad Social**:
  - Fomentar el desarrollo de habilidades cognitivas y sociales a través del juego.
  - Incluir elementos educativos que enriquezcan la experiencia del jugador.

Consideraciones de Privacidad

- **Protección de Datos Personales**:

    - Implementar medidas de seguridad para proteger la información sensible de los jugadores, como el cifrado de datos y la autenticación segura.
    - Cumplir con las regulaciones de privacidad de datos, como el GDPR.
- **Transparencia**:
    - Informar a los jugadores sobre cómo se recopilan, utilizan y almacenan sus datos.
    - Proporcionar opciones para que los jugadores gestionen sus preferencias de privacidad.

1.5 Conclusión del Levantamiento de Requisitos y Diseño

El levantamiento de requisitos y el diseño proporcionaron una base sólida para la fase de desarrollo del videojuego, garantizando que el sistema cumpla con los objetivos de enseñanza de la historia ecuatoriana.

Capítulo II

Construcción del Sistema

1.1 Metodología de Desarrollo

La construcción del sistema se realizó utilizando el motor de desarrollo Unity, lo cual es aprovechado sus capacidades de renderizado 2D para lograr un estilo en pixel art. En cada nivel fue creado siguiendo un esquema de navegación top-down que facilita la exploración por parte del usuario.

1.2 Herramientas de Desarrollo y Control de Versiones

Herramientas de Desarrollo:

1. **Unity 2023.1.8f:** Motor de videojuegos utilizado para el diseño, desarrollo y renderizado en 2D.
2. **C#:** Lenguaje de programación utilizado para las mecánicas del videojuego.
3. **Visual Studio Code:** IDE empleado para escribir y depurar el código.
4. **GIMP/Photoshop:** Creación de sprites y elementos visuales en estilo pixel art

Herramienta de Control de Versiones:

- **Git:** Herramienta para el control de versiones, permitiendo registrar los cambios realizados durante el desarrollo.
- **GitHub:** Plataforma en línea utilizada para gestionar el repositorio del proyecto.

1.3 Metodología de Trabajo

Para la construcción del videojuego educativo se utilizó la metodología ágil Scrum, que permite organizar el desarrollo en ciclos iterativos llamados sprints. Este enfoque es ideal para videojuegos, ya que facilita la constante revisión y mejora de las funcionalidades implementadas, asegurando que el producto final cumpla con los objetivos educativos y de entretenimiento.

Scrum en el Desarrollo del Videojuego

Duración del Sprint: 3 semanas

Roles del Equipo:

Product Owner: Responsable de definir los objetivos principales del videojuego y priorizar funcionalidades.

Scrum Master: Facilitador del proceso ágil, encargado de eliminar bloqueos y asegurar el cumplimiento de los sprints.

Desarrollador: Responsable de la implementación del código, el diseño de mecánicas y la optimización del rendimiento del juego.

Fases del Desarrollo

1. **Planificación del Sprint:**
    - Definición de tareas y objetivos a desarrollar en cada sprint.
    - Priorización de funcionalidades en el backlog del producto.
    - Diseño inicial de mecánicas, niveles y elementos gráficos.
2. **Desarrollo:**
    - Implementación de la lógica del juego en Unity utilizando C#.
    - Creación de niveles, escenarios y sprites en pixel art.
    - Programación de mecánicas principales, como el sistema de vidas, cronómetro y enemigos.
    - Integración de efectos de sonido y música de fondo.
3. **Pruebas:**
    - Evaluación de la jugabilidad y detección de errores.
    - Corrección de bugs y optimización del rendimiento.
    - Revisión de mecánicas y ajuste de la dificultad según pruebas de usuario.
4. **Revisión Final:**
    - Validación del videojuego con pruebas de usuarios reales.

- Ajustes finales en mecánicas y gráficos según el feedback recibido.
- Revisión de la accesibilidad y facilidad de uso del juego.

5. **Entrega Final:**
    - Documentación del proyecto, incluyendo el manual técnico y manual de usuario.
    - Generación del instalador o empaquetado para distribución.
    - Publicación en una plataforma de distribución o entrega en formato digital.

Organización del Trabajo en Unity

El desarrollo del videojuego en Unity se organizó mediante las siguientes fases:

Levantamiento de Requisitos

- Identificación de las funcionalidades educativas y de entretenimiento necesarias.
- Definición de los objetivos del videojuego y su impacto en el aprendizaje.
- Creación del backlog del producto con historias de usuario.

Diseño del Sistema

- Creación del flujo de pantallas con el recorrido del jugador.
- Definición de los niveles históricos, NPCs y mecánicas interactivas.
- Diseño de la interfaz de usuario (UI/UX) con menús, HUD y controles.

Implementación

- Desarrollo en Unity, programando las mecánicas principales en C#.
- Integración de gráficos en pixel art y animaciones.
- Creación del sistema de colisiones, movimiento del personaje y eventos del juego.
- Optimización del código y assets para mejorar el rendimiento en distintas plataformas.

Pruebas

- Validación del sistema para garantizar que cumple con los objetivos educativos y de entretenimiento.
- Ejecución de pruebas de estrés para evaluar el rendimiento.
- Revisión de compatibilidad en diferentes dispositivos y resoluciones.
- Corrección de errores y ajustes de balance en la jugabilidad.

Publicación y Entrega

- Generación de una versión final jugable en formato ejecutable (.exe para PC).
- Creación de documentación, incluyendo el manual técnico y manual de usuario.
- Publicación del juego en plataformas educativas o de descarga.
- Evaluación de la recepción del videojuego por parte de los usuarios.

1.4 Implementación de los Requisitos en Unity

Lenguaje de Desarrollo

El desarrollo del videojuego "Explorador del Tiempo" se realizará utilizando **C#** como lenguaje de programación principal. C# es el lenguaje preferido para Unity debido a su robustez, flexibilidad y la amplia documentación y recursos disponibles.

Editor de Código

Para la edición del código, se recomienda utilizar **Visual Studio** o **Visual Studio Code**. Ambos editores son compatibles con Unity y ofrecen características avanzadas que facilitan el desarrollo:

- **Autocompletado de código**: Sugerencias de código en tiempo real.
- **Depuración**: Herramientas para identificar y corregir errores.

- **Integración con Unity**: Funcionalidades específicas para proyectos de Unity.

Desarrollo en Unity

El desarrollo en Unity sigue un flujo de trabajo estructurado que incluye los siguientes pasos técnicos:

1. **Configuración del Proyecto**:
    - Crear un nuevo proyecto en Unity y seleccionar la plantilla 2D.
    - Configurar los ajustes del proyecto en el menú **Edit > Project Settings**, incluyendo la resolución de pantalla y las opciones de compilación.
2. **Importación de Recursos**:
    - Importar los recursos necesarios (sprites, sonidos, fuentes) a través del menú **Assets > Import New Asset**.
    - Organizar los recursos en carpetas dentro del proyecto para mantener el orden y facilitar el acceso.
3. **Creación de Escenas**:
    - Crear nuevas escenas desde el menú **File > New Scene**.
    - Configurar las escenas añadiendo elementos de la interfaz de usuario (UI) utilizando el **Unity UI Toolkit**.
    - Utilizar el **Scene View** para posicionar y ajustar los elementos visuales.
4. **Programación de la Lógica del Juego**:

- Escribir scripts en C# para definir el comportamiento de los objetos del juego. Los scripts se pueden crear desde el menú **Assets > Create > C# Script**.

- Utilizar componentes de Unity como **Rigidbody2D** y **BoxCollider2D** para manejar la física y las colisiones.

- Implementar la lógica del juego, como el movimiento del personaje, la detección de colisiones y las interacciones con el entorno. Ejemplo de un script básico para el movimiento del personaje:

```
using UnityEngine;
public class PlayerMovement : MonoBehaviour
{
   public float speed = 5f;
 private Rigidbody2D rb;
 void Start()
  {
      rb = GetComponent<Rigidbody2D>();
   }

   void Update()
   {
      float moveX = Input.GetAxis("Horizontal");
      float moveY = Input.GetAxis("Vertical");
      Vector2 movement = new Vector2(moveX, moveY);
      rb.velocity = movement * speed;
   }
}
```

5. **Pruebas y Depuración**:
    - Probar el juego regularmente utilizando el **Game View** para identificar y corregir errores.
    - Utilizar las herramientas de depuración de Visual Studio o Visual Studio Code para solucionar problemas en el código, como puntos de interrupción y seguimiento de variables.
6. **Optimización**:
    - Optimizar el rendimiento del juego ajustando la calidad gráfica y reduciendo el uso de recursos innecesarios.
    - Utilizar el **Profiler** de Unity para monitorear el rendimiento y detectar cuellos de botella.
7. **Compilación y Distribución**:
    - Compilar el juego para las plataformas objetivo (Windows, macOS, etc.) desde el menú **File > Build Settings**.
    - Configurar las opciones de compilación y seleccionar la plataforma deseada.
    - Preparar el juego para su distribución, incluyendo la creación de instaladores y la documentación necesaria.

1.5 Desarrollo de los Niveles Históricos

Para el desarrollo de cada nivel histórico se tomó la decisión de sumergir al jugador en una época especifica lo cual será un nivel en específico:

- Nivel 1: Fundación de Quito (1534), con un escenario colonial con sus personajes de la época y personajes históricos.
- Nivel 2: Independencia de Guayaquil (1820), en un entorno portuario y con sus personajes históricos.
- Nivel 3: Batalla de Pichincha (1822), ambientado en una zona montañosa también implementando sus personajes.

1.6 Programación de Mecánicas de Juego

Para que el videojuego tenga una experiencia educativa, se programaron las siguientes interfases.

- Un cronometro que desafía al jugador a completar la misión dentro de un límite de tiempo.
- Sistema de vidas que reduce la cantidad de intentos si el jugador falla en sus objetivos o al no completarlos.
- Interacciones con NPCs con los cuales enriquecen el contenido histórico y dan contexto a las misiones.

1.7 Implementación de Funcionalidades Especificas

1. **Cronometro**:

Se desarrollo un cronometro para agregar un desafío adicional al jugador. Este cronometro cuenta regresivamente desde un tiempo predefinido hasta cero. Si el tiempo llega a cero, el jugador pierde automáticamente el nivel. La funcionalidad fue implementada usando la capacidad de Unity de actualizar constantemente el tiempo restante en cada cuadro (frame) del juego.

2. **Sistema de Vidas:**

Cada jugador comienza con tres vidas al inicio del nivel. Cada vez que falla en completar un objetivo o agota el tiempo disponible, pierde una vida. Si el jugador pierde todas las vidas, se muestra una pantalla de derrota, y el nivel debe reiniciarse. Esta funcionalidad fue diseñada para ofrecer un equilibrio entre dificultad y aprendizaje.

3. **Interacciones con NPCs:**

Los personajes no jugables (NPCs) fueron programados para dar información relevante y asignar misiones al jugador. Por ejemplo, en el primer nivel, un NPC

proporciona pistas para encontrar un documento perdido. Las interacciones fueron gestionadas mediante sistemas de detección y eventos en Unity, activándose al acercarse el jugador

1.8 Implementación del Backend

Dado que el videojuego no utiliza un backend ni una base de datos externa, el progreso del jugador se gestiona localmente dentro de Unity. Para almacenar los datos, se utilizó el sistema PlayerPrefs de Unity, que permite guardar información básica como el nivel actual, las vidas restantes y configuraciones del juego (volumen y dificultad).

- **Guardar Progreso:**

  Se implemento una función que almacena el nivel actual y las vidas restantes del jugador al completar o salir de un nivel. Estos datos se guardan en el dispositivo del jugador.

- **Cargar Progreso:**

  Al seleccionar la opción “Continuar” en el menú principal, el juego recupera los datos guardados y permite al jugador retomar el ultimo nivel completado. Esto garantiza una experiencia continua sin necesidad de backesnd.

Capítulo III

Pruebas y Estabilización

3.1 Introducción

Para garantizar la estabilidad y funcionalidad del videojuego educativo, se llevaron a cabo pruebas de usuario con diferentes jugadores. Estas pruebas tenían como objetivo detectar errores, evaluar la jugabilidad y recopilar comentarios sobre la experiencia general. Se seleccionaron tres jugadores con distintos niveles de experiencia en videojuegos para obtener una visión más amplia sobre el rendimiento y posibles mejoras.

A continuación, se detallan los resultados obtenidos durante las sesiones de prueba.

3.2 Pruebas de Jugabilidad y Errores Encontrados

Jugador 1: Andrés (Jugador ocasional, 22 años)

Experiencia de juego:

"Me gustó bastante el estilo pixel art y la música. La mecánica es sencilla y fácil de entender, aunque al principio no estaba muy claro qué debía hacer. Después de un rato me acostumbré y lo disfruté."

Errores encontrados:

- El personaje se queda atascado en esquinas: En algunas partes del escenario, especialmente en los bordes de los niveles.
- El botón de pausa no responde a veces: Presioné la tecla de pausa (Escape), pero en algunas ocasiones no funcionó y tuve que presionarlo varias veces.

- El texto de algunos diálogos sale de manera diferente: En ciertas resoluciones, los cuadros de texto no se ajustan correctamente y algunas palabras.

Sugerencias:

- Hacer que el botón de pausa siempre responda bien.
- Ajustar la fuente del texto para que salga bien.

Jugador 2: María (Estudiante, 17 años)

Experiencia de juego:

"El juego es entretenido y me gustó la temática educativa. Es fácil de aprender, aunque sentí que algunas partes eran un poco repetitivas. Los gráficos están bonitos y la música no es molesta."

Errores encontrados:

- Las colisiones a veces no funcionan bien: Hay momentos en los que paso por un lado de un personaje y no me dejan pasar.
- El personaje a veces no responde a los controles: En ciertos momentos, sobre todo después de pasar el juego, el personaje no responde al primer intento de moverse.

Sugerencias:

- Revisar las colisiones para que reaccionen de forma más precisa.
- Mejorar la respuesta de los controles después de pasar.

Jugador 3: Miguel (Desarrollador y jugador experimentado, 46 años)

Experiencia de juego:

"Me gusta la idea del juego y se nota que tiene potencial. La jugabilidad es sólida, aunque hay algunos detalles que podrían mejorar. El sistema de puntuación y vidas funciona bien, pero algunas mecánicas pueden pulirse más."

Errores encontrados:

- Los FPS bajan en ciertos niveles: En una sección con muchos elementos en pantalla, el rendimiento del juego bajó y se notaron algunos tirones.
- El sonido se desincroniza si cambias la resolución del juego: Al cambiar la resolución en las opciones, la música a veces deja de sonar hasta que reinicias.

Sugerencias:

- Optimizar los niveles para que el juego corra de forma fluida sin bajones de FPS.
- Verificar que el audio se mantenga estable al cambiar la configuración.

3.3 Conclusión y Plan de Mejora

Las pruebas realizadas permitieron identificar errores en mecánicas clave del juego, como colisiones, pausas y reinicio de partidas. Además, se detectaron pequeños problemas de rendimiento y gráficos que pueden mejorarse en futuras actualizaciones.

Para corregir estos errores, se establecen las siguientes acciones:

1. Corrección de colisiones: Ajustar las detecciones de contacto entre el personajes y el entorno.

2. Mejora en la pausa: Asegurar que el juego se detenga correctamente sin afectar el temporizador ni los controles.
3. Optimización de rendimiento: Reducir la carga gráfica en ciertos niveles para mejorar los FPS.
4. Corrección de problemas de audio: Revisar la sincronización de sonido al cambiar la resolución o configuración del juego.
5. Ajuste del sistema de reinicio: Garantizar que al regresar al menú principal se restablezcan correctamente todas las variables del juego.

Conclusiones

El desarrollo del videojuego "Explorador del Tiempo" ha sido un proceso integral y meticuloso que abarca desde la definición de requisitos hasta la implementación y estabilización del juego. A lo largo de este proyecto, se han alcanzado varios hitos importantes que garantizan la calidad y funcionalidad del producto final.

En primer lugar, se establecieron requisitos mínimos de hardware y software ajustados para asegurar la accesibilidad del juego en una amplia gama de dispositivos. Esto incluyó la selección de una CPU de dos núcleos, 4 GB de RAM, una GPU integrada compatible con DirectX 10 y al menos 2 GB de espacio libre en disco. Además, se implementaron consideraciones éticas y de privacidad para garantizar una representación justa y equitativa de diferentes géneros, etnias y orientaciones sexuales, así como la protección de datos personales de los jugadores.

El desarrollo en Unity, utilizando C# como lenguaje de programación y Visual Studio como editor de código, siguió un flujo de trabajo estructurado que incluyó la configuración del proyecto, importación de recursos, creación de escenas, programación de la lógica del juego, pruebas y optimización. Este enfoque iterativo permitió asegurar la calidad y funcionalidad del juego en cada etapa del desarrollo.

Las pruebas y la estabilización fueron componentes críticos del proyecto. Se aplicaron diversas metodologías de pruebas, incluyendo pruebas unitarias, de integración, de sistema y de aceptación, para asegurar que todas las características del juego funcionaran correctamente y que el rendimiento fuera óptimo. Las estrategias de estabilización, como la gestión de errores, la optimización del rendimiento y las pruebas de regresión, fueron fundamentales para asegurar una experiencia de usuario satisfactoria.

En conclusión, el proyecto "Explorador del Tiempo" ha demostrado ser un esfuerzo exitoso en la creación de un videojuego educativo y entretenido. A través de un enfoque meticuloso en el diseño, desarrollo, pruebas y estabilización, se ha logrado un producto final que cumple con los objetivos planteados y ofrece una experiencia de juego de alta calidad. La implementación de consideraciones éticas y de privacidad asegura que el juego no solo sea divertido, sino también seguro y respetuoso para todos los jugadores. Este proyecto no solo ha alcanzado sus metas técnicas, sino que también ha establecido una base sólida para futuros desarrollos en el ámbito de los videojuegos educativos.

Recomendaciones

A partir del desarrollo y la implementación del videojuego "Explorador del Tiempo", se han identificado varias áreas de mejora y recomendaciones para futuros proyectos similares. Estas recomendaciones están diseñadas para optimizar el proceso de desarrollo, mejorar la calidad del producto final y asegurar una experiencia de usuario aún más satisfactoria.

1. Mejora Continua del Proceso de Desarrollo

- **Documentación Detallada**: Mantener una documentación detallada y actualizada de cada etapa del desarrollo facilita la colaboración entre los miembros del equipo y asegura la continuidad del proyecto en caso de cambios de personal.

2. Optimización del Rendimiento

- **Uso Eficiente de Recursos**: Continuar optimizando el uso de recursos gráficos y de procesamiento para asegurar que el juego funcione de manera eficiente en dispositivos con especificaciones mínimas.
- **Pruebas de Estrés**: Realizar pruebas de estrés adicionales para identificar y corregir posibles cuellos de botella en el rendimiento del juego bajo condiciones extremas.

3. Ampliación de Pruebas y Feedback

- **Beta Testing Abierto**: Implementar un programa de beta testing abierto para obtener feedback de una audiencia más amplia y diversa, lo que puede ayudar a identificar problemas que no se detectaron durante las pruebas internas.

- **Análisis de Datos**: Utilizar herramientas de análisis de datos para monitorear el comportamiento de los jugadores y ajustar el juego en función de sus preferencias y patrones de uso.

4. Enfoque en la Experiencia del Usuario

- **Accesibilidad**: Implementar características de accesibilidad, como opciones de control personalizables y soporte para jugadores con discapacidades, para asegurar que el juego sea inclusivo para todos los usuarios.

5. Consideraciones Éticas y de Privacidad

- **Revisión Continua de Políticas de Privacidad**: Mantenerse actualizado con las regulaciones de privacidad de datos y ajustar las políticas del juego en consecuencia para asegurar el cumplimiento continuo.
- **Educación y Sensibilización**: Incluir elementos educativos dentro del juego que promuevan la sensibilización sobre temas éticos y de privacidad, fomentando un entorno de juego seguro y respetuoso.

6. Innovación y Actualización

- **Actualizaciones Regulares**: Planificar y ejecutar actualizaciones regulares del juego para introducir nuevas características, corregir errores y mantener el interés de los jugadores.
- **Exploración de Nuevas Tecnologías**: Investigar y adoptar nuevas tecnologías y herramientas que puedan mejorar el desarrollo y la experiencia del juego, como la inteligencia artificial y la realidad aumentada.

Referencias bibliográficas

- Ayala Mora, E. (2008). *Reseña histórica del Ecuador*. Ediciones Abya-Yala.
- Crespo Troyal, A. (1980). *La Independencia del Ecuador*. Editorial Casa de la Cultura Ecuatoriana.
- **Tobar, O.** (2015). *Game Development - From the Vision to the Final Product*. Lappeenranta University of Technology. Recuperado de UPC Commons.
- **Neupane, P.** (2015). *Essential Elements of Game Development: A Case Study*. Theseus. Recuperado de Theseus.fi.
- **Unity Technologies**. (2020). *Unity User Manual 2020.3 (LTS)*. Recuperado de Unity Documentation.
- **Microsoft**. (2020). *C# Programming Guide*. Recuperado de Microsoft Docs.
- **Rouse, R.** (2004). *Game Design: Theory and Practice*. Plano, TX: Wordware Publishing, Inc.

Anexos

El videojuego proporcionar dos manuales esenciales para la correcta implementación y uso del videojuego educativo desarrollado en Unity.

Por un lado, el Manual Técnico está dirigido a desarrolladores y personal de mantenimiento del software, detallando la arquitectura del videojuego, los requisitos del sistema, la instalación, la estructura del código y los procedimientos de depuración y actualización. Este manual permite garantizar la continuidad del desarrollo, facilitando futuras mejoras o correcciones en el sistema.

Por otro lado, el Manual de Usuario está diseñado para los docentes y estudiantes que utilizarán el videojuego. En él se explican los pasos de instalación, las funciones principales del juego y las mecánicas de interacción, así como soluciones a problemas comunes que puedan surgir durante su uso.

Ambos manuales han sido estructurados con un enfoque práctico y accesible, asegurando que tanto los desarrolladores como los usuarios finales puedan comprender y aplicar la información de manera efectiva. Estos documentos contribuyen a la documentación integral del proyecto, garantizando su operatividad y facilitando su implementación en entornos educativos.