**RUMBO**

**Simón Carmona Toro**

**Nicolás Mateo Pérez Carvajal**

**Diego Alejandro Rodriguez Espinoza**

**Juan Pablo Agudelo Pérez**

**12/08/2024**

**Índice**

**Introducción ----------------------------------------------------------- pag 3**

**1.1 Descripción general del sistema--------------------------------------- pag 3**

**1.2 Propósito del documento---------------------------------------------- pag 3**

**1.3 Alcance del Sistema --------------------------------------------------- pag 3**

**Descripción general del sistema ------------------------------------------ pag 3**

**2.1 Contexto del sistema “RUMBO” ----------------------------------------pag 3**

**2.2 Distintos roles ----------------------------------------------------------pag 3**

**2.3 Suposiciones y dependencias -------------------------------------------pag 4**

### Gestión de Proyectos ---------------------------------------------------pag 5

### 3.1 Gestión de Tareas---------------------------------------------------pag 6

### Requerimientos No funcionales ---------------------------------------pag 7

**Arquitectura del Sistema------------------------------------------------pag 9**

**5.1 Visión General ----------------------------------------------------------pag 9**

**5.2 Herramientas -----------------------------------------------------------pag 9**

**5.3 Integraciones --------------------------------------------------------pag 10**

**6 Apéndice ---------------------------------------------------------------pag 10**

1. **Introducción**

**1.1 Descripción general del sistema:** Rumbo es un videojuego de rol de toma de decisiones que servirá en mejorar su memoria prospectiva, además es útil para mejorar el pensamiento crítico y el análisis de situaciones conflictivas o difíciles.

**1.2 Propósito del documento:** El propósito del Documento ERS es evidenciar las funcionalidades que se llevarán a cabo a lo largo de todo el proyecto, es importante para llevar un seguimiento para el equipo de desarrollo, y es crucial definir toda la documentación para llevar a cabo un correcto desarrollo.

**1.3 Alcance del Sistema:** Lo que se hará es un estudio de las capacidades neurológicas de los jóvenes para encontrar el punto clave y desarrollarlo, además se busca encontrar un desarrollo único para distinguirnos en el campo de los videojuegos de rol.

Lo que haremos es un videojuego que sea visualmente agradable para los jugadores y que les proporcione un buen entorno de juego.

1. **Descripción general del sistema**

**2.1 Contexto del sistema “RUMBO”** busca integrarse de manera positiva en la forma de pensar de los jóvenes y como estos toman acción en las diferentes situaciones para poder ejecutar el plan de mejoramiento en su desarrollo, esto por medio de interacciones reflexivas sin hacer énfasis por alguna religión o postura política, esto con el fin de hacer que el videojuego sea inclusivo.

**2.2 Distintos roles:**

* **Desarrollador tester:** busca actualizar constantemente el videojuego, dando solución a los errores que se presenten a lo largo del desarrollo y de las actualizaciones del videojuego.
* **Desarrollador backend:** Se encargará de codificar y optimizar el código fuente del videojuego logrando que este sea más ligero.
* **Diseñador:** Es aquel desarrollador que tiene la tarea de diseñar las texturas que se incluyen a lo largo de todo el videojuego.
* **Analista de datos:** Es el encargado de realizar las revisiones y comparaciones del desarrollo y que coincida en todos los aspectos con la documentación del videojuego.
* **Usuario:** Este rol se centra en la experiencia interactiva con el juego, incluyendo la toma de decisiones, el desarrollo de habilidades, la gestión de proyectos y tareas, y la progresión del personaje.

**2.3 Suposiciones y dependencias**

**Suposiciones:**

* **Rendimiento y Experiencia del Usuario:** Se supone queel juego mantendrá un rendimiento aceptable (30 FPS o más) en configuraciones mínimas.
* **Estabilidad del versionamiento:** Se asume que el versionamiento no tendrá actualizaciones disruptivas que afecten negativamente el desarrollo del videojuego.
* **Usuarios y Mercado:** Los jugadores objetivo tendrán acceso a computadoras de bajos recursos con capacidad para ejecutar juegos en 2.5D y gráficos de pixel art.

**Dependencias:**

**Herramienta de desarrollo:** El equipo depende de softwares y herramientas específicas para desarrollar el videojuego de forma adecuada como lo son:

**lenguaje principal del código fuente:**

* **C#:** Dependemos de este lenguaje de programación ya que con este se realizará el código fuente del videojuego y lo que respecta a la sección del Backend.

**Diseño:**

* **Pixel Art:** Dependemos de la destreza del diseñador para realizar satisfactoriamente y en tiempos de trabajo cortos los diseños y texturas que tendrá el videojuego.

**Gestion general:**

* **Repositorios y versionamiento:** Dependemos del buen manejo de y almacenamiento de los repositorios para mantener cada rama del proyecto en perfecto control, y guardar correctamente todos los registros de cada nueva versión del proyecto.

**Producción auditiva:**

* **Sonidos:** Dependemos de una buena producción de efectos de sonido e integración de bandas sonoras para el videojuego el cual le proporcionará una mayor inmersión.

**Testeo:**

* **Hacer pruebas:** Dependemos de unas pruebas completas y que proporcionen una información que aporte para la continuación del proyecto.
* **Dependencia de jugadores:** Se depende de que la temática sea adecuada y atractiva. Para garantizar esto, es crucial realizar un riguroso proceso de testeo que evalúe la jugabilidad, narrativa, y experiencia del usuario. El feedback obtenido de estas pruebas permitirá ajustar y mejorar el juego para asegurar que cumpla con las expectativas y preferencias de los jugadores.

### 3.Gestión de Proyectos

**Requerimientos Funcionales:**

* **Creación de Proyectos:** El personaje podrá iniciar proyectos personales o laborales, asignando un nombre, una descripción, una fecha de inicio y fin.
* **Organización de Proyectos:** El personaje podrá priorizar y organizar proyectos según su relevancia o urgencia.
* **Gestión de Proyectos:** El personaje podrá editar, posponer o cancelar proyectos en función de las circunstancias del juego.

**Historia de Usuario:**

* **Como** personaje del juego,
* **Quiero** poder iniciar y gestionar mis propios proyectos,
* **Para** planificar mis metas y objetivos en el juego de manera efectiva.

### 3.1 Gestión de Tareas

**Requerimientos Funcionales:**

* **Creación de Tareas:** El personaje podrá crear tareas dentro de sus proyectos, especificando detalles como el título, la descripción, la fecha de vencimiento y el estado.
* **Seguimiento de Tareas:** El personaje podrá ver el progreso de sus tareas y recibir notificaciones cuando las tareas estén cerca de su fecha de vencimiento.
* **Cierre de Tareas:** El personaje podrá marcar tareas como completadas y revisar las tareas cerradas para aprender de sus experiencias.

**Historia de Usuario:**

* **Como** personaje del juego,
* **Quiero** poder crear y gestionar mis tareas,
* **Para** manejar mi tiempo y recursos de manera eficiente en el juego.

### Gestión de Usuarios y Permisos

**Requerimientos Funcionales:**

* **Perfil del Personaje:** El personaje podrá modificar su perfil y sus habilidades a lo largo del juego.
* **Permisos de Acción:** Las habilidades del personaje determinarán las acciones que puede realizar en el juego.

**Historia de Usuario:**

* **Como** personaje,
* **Quiero** desarrollar mis habilidades y gestionar mis acciones,
* **Para** progresar y enfrentar diferentes desafíos en el juego.

### Flujos de Trabajo

**Requerimientos Funcionales:**

* **Configuración de Flujos de Trabajo:** El personaje podrá establecer rutinas y procesos que reflejan sus prioridades y estilo de vida.
* **Gestión de Flujos de Trabajo:** El personaje podrá ajustar o cambiar sus rutinas según las necesidades y eventos del juego.
* **Automatización de Flujos de Trabajo:** El sistema podrá sugerir flujos de trabajo óptimos basados en las decisiones y acciones del personaje.

**Historia de Usuario:**

* **Como** personaje del juego,
* **Quiero** poder establecer mis propias rutinas y procesos,
* **Para** optimizar mis decisiones y adaptarse a diferentes situaciones.

### Informes y Dashboards

**Requerimientos Funcionales:**

* **Generación de Informes:** El personaje podrá acceder a informes sobre su progreso y logros en el juego.
* **Visualización de Datos:** El personaje tendrá un dashboard que muestra sus métricas clave y estadísticas personales.
* **Personalización de Dashboards:** El personaje podrá personalizar su dashboard para enfocarse en la información que considera más relevante para su desarrollo.

**Historia de Usuario:**

* **Como** personaje,
* **Quiero** ver informes y estadísticas de mis acciones,
* **Para** analizar mi progreso y mejorar mis decisiones en el juego.

**4. Requerimientos No funcionales**

### Rendimiento

**Requerimientos de Rendimiento:**

* **Tiempo de Carga:** El juego debe cargar completamente en menos de 10 segundos en equipos con especificaciones mínimas.
* **FPS Consistente:** Mantener un framerate mínimo de 30 FPS en las configuraciones más bajas para asegurar fluidez.
* **Uso Eficiente de Recursos:** Optimizar el uso de memoria y CPU para evitar sobrecargas en computadoras de bajos recursos.

### Seguridad

**Requerimientos de Seguridad:**

* **Autenticación Segura:** Utilizar protocolos seguros para gestionar el inicio de sesión de los usuarios.
* **Autorización de Roles:** Implementar niveles de acceso según el rol del usuario (p. ej., jugador, administrador) para controlar el acceso a diferentes funcionalidades.
* **Protección de Datos:** Encriptar los datos sensibles y asegurar que la información personal de los usuarios está protegida contra accesos no autorizados.

### Usabilidad

**Requerimientos de Usabilidad:**

* **Interfaz Intuitiva:** Diseñar una interfaz de usuario clara y sencilla que facilite la navegación y comprensión del juego.
* **Tutoriales y Ayuda:** Incluir tutoriales interactivos y secciones de ayuda que explican las mecánicas del juego a los jugadores nuevos.
* **Accesibilidad:** Asegurar que el juego sea accesible para personas con discapacidades, incluyendo opciones de audio, subtítulos y ajustes visuales.

### Compatibilidad

**Requerimientos de Compatibilidad:**

* **Plataformas:** El juego debe ser compatible con múltiples sistemas operativos, incluyendo Windows, macOS y Linux.
* **Integración con Servicios:** Facilitar la integración con plataformas de distribución como Steam para permitir descargas y actualizaciones.
* **Escalabilidad:** Diseñar el juego de manera que pueda adaptarse a diferentes resoluciones de pantalla y configuraciones de hardware.

**5.Arquitectura del Sistema**

**5.1 Visión General:** La arquitectura del sistema del videojuego estará basada en una estructura cliente-servidor, donde el cliente interactúa con un servidor central para gestionar el almacenamiento de datos y la lógica de juego.

**5.2 Herramientas:**

**Motor de Juego:**

* **Unity:** Dependemos de esta herramienta para llevar a cabo la gestión del proyecto el cual es encargado de hacer que el programa se ejecute.

**Software de Diseño Gráfico:**

* **Aseprite:** Dependemos de esta herramienta la cual está enfocada al diseño en estilo Pixel Art con el cual se harán todos los diseños y texturas del videojuego
* **Photoshop:** Dependemos de esta herramienta de diseño y arte digital para elaborar publicidad y diseños externos a lo que es el videojuego.

**Herramientas de Programación:**

* **Visual Studio Code o Visual Studio:** Dependemos de estos entornos de desarrollo integrados que soporta múltiples lenguajes de programación y ofrecen extensiones útiles para desarrollo de este videojuego.

**Gestión de Versiones:**

* **Git:** Dependemos de esta herramienta para controlar versiones del código y colaborar entre los miembros del equipo de desarrollo.
* **GitHub:** Dependemos de esta plataforma para alojar el repositorio Git y facilitar la colaboración.

**Herramientas de Sonido y Música:**

* **Audacity:** Hay dependencia de esta herramienta la cual se utilizará para editar los efectos de sonido y demás efectos multimedia.
* **FL Studio:** Dependemos de estas herramientas para componer y crear bandas sonoras y efectos de sonido los cuales irán incluidos en el videojuego.

**Herramientas de Prueba:**

* **Unity Test Runner:** Dependemos de esta herramienta integrada en Unity para realizar las pruebas y todo el testeo del progreso del desarrolló y confirmar su correcta realización.

**5.3 Integraciones:**

* **Plataformas de Distribución:** Integración con Steam para distribución y gestión de licencias.
* **Sistemas de Pago:** Integración con sistemas de pago para compras dentro del juego, si corresponde.
* **Sistemas de Chat y Soporte:** Si se incluye soporte para interacciones entre jugadores, podría ser necesaria la integración con plataformas de chat o foros para soporte técnico y comunidad.

**6. Apéndice:**

**6.1 Glosario**

**1.** Videojuego de rol: Un juego en el que el jugador asume el rol de un personaje en un mundo ficticio, tomando decisiones que afectan el desarrollo de la historia.

**2.** Memoria prospectiva: Capacidad de recordar realizar acciones planificadas en el futuro.

**3.** Backend: Parte de un sistema que gestiona la lógica de la aplicación, la base de datos, y la comunicación con el servidor.

4. Pixel Art: Estilo de arte digital en el que las imágenes se crean o editan pixel por pixel.

**5.** FPS (Frames Per Second): Tasa de imágenes por segundo que muestra una pantalla, utilizada para medir el rendimiento de gráficos en videojuegos.

**6.** Git: Sistema de control de versiones distribuido utilizado para gestionar cambios en el código fuente de un proyecto.

**7.** GitHub: Plataforma de desarrollo colaborativo basada en la nube que permite alojar repositorios Git.

**8.** C#: Lenguaje de programación utilizado para desarrollar aplicaciones, incluyendo videojuegos.

**9.** Unity: Motor de desarrollo de videojuegos que permite la creación de juegos en 2D y 3D.

**10.** Aseprite: Software especializado en la creación de gráficos de Pixel Art.

**11.** Visual Studio Code: Editor de código fuente ligero y multiplataforma con soporte para múltiples lenguajes de programación.

**12.** Audacity: Software de edición de audio de código abierto utilizado para editar efectos de sonido.

**13.** FL Studio: Estación de trabajo de audio digital (DAW) utilizada para la producción de música y efectos de sonido.

**14.** Steam: Plataforma de distribución digital de videojuegos que permite su descarga y actualización.

**15.** Integración de sistemas: Proceso de enlazar diferentes sistemas de software para que funcionen juntos.

**16.** Requerimientos no funcionales: Especificaciones que definen cómo debe funcionar un sistema, como rendimiento, seguridad y usabilidad.

**17.** Autenticación segura: Método para verificar la identidad de un usuario de manera segura.

**18.** Versionamiento: Proceso de gestionar diferentes versiones de un software a lo largo de su desarrollo.

**19.** Testeo: Proceso de verificar que un software funcione correctamente a través de pruebas.

**20.** FPS consistente: Tasa de fotogramas por segundo que se mantiene estable para garantizar una experiencia de juego fluida.

**21.** Motor de juego: Software que proporciona funcionalidades básicas para el desarrollo de videojuegos, como física, gráficos, y sonido.

**22.** Escalabilidad: Capacidad de un sistema para manejar un aumento en la carga de trabajo.

**23.** Interfaz intuitiva: Diseño de usuario fácil de entender y utilizar sin necesidad de instrucciones extensivas.

24. Usabilidad: Medida de lo fácil que es para los usuarios aprender a utilizar y operar un sistema.