

# Especificación de Requisitos según el estándar de IEEE 830

IEEE Std. 830-1998

## **Resumen**

Este documento presenta, en castellano, el formato de Especificación de Requisitos Software (ERS) según la última versión del estándar IEEE 830. Según IEEE, un buen Documento de Requisitos, pese a no ser obligatorio que siga estrictamente la organización y el formato dados en el estándar 830, si debería incluir, de una forma o de otra, toda la información presentada en dicho estándar. El estándar de IEEE 830 no está libre de defectos ni de prejuicios, y por ello ha sido justamente criticado por múltiples autores y desde múltiples puntos de vista, llegando a cuestionar incluso si es realmente un estándar en el sentido habitual que tiene el término en otras ingenierías. El presente documento no pretende pronunciarse ni a favor ni en contra de unos u otros: tan solo reproduce, con propósitos fundamentalmente docentes, cómo se organizaría un Documento de Requisitos según el estándar IEEE 830.

---

## **Especificación de Requisitos de Software**

**Proyecto: Magical Creature**

---

**Valentina Cruz Rivera**

**Jhampier Santos Ortiz**

**Carol Nicole Marentes Torres**

# Índice

<b>1. Introducción</b>	<b>3</b>
1.1. Propósito	3
1.2. Ámbito del Sistema	3
1.3. Definiciones, Acronimos y Abreviaturas	3
1.4. Referencias	4
1.5. Visión General del Documento	5
<b>2. Descripción General</b>	<b>5</b>
2.1. Perspectiva del Producto	5
2.2. Funciones del Producto	5
2.3. Características de los Usuarios	6
2.4. Restricciones	6
2.5. Suposiciones y Dependencias	6
2.6. Requisitos Futuros	7
2.7. Objetivos Generales y Específicos	7
<b>3. Recolección de datos</b>	<b>8</b>
3.1. Historias de Usuario	8
3.2. Entrevista	9
3.3. Evaluación Heurística	9
<b>4. Requisitos Específicos</b>	<b>10</b>
4.1. Interfaces Externas	10
4.2. Funciones	10
4.3. Requisitos de Rendimiento	11
4.4. Restricciones de Diseño	11
4.5. Atributos del Sistema	12
4.6. Otros Requisitos	12
<b>5. Apéndices</b>	<b>32</b>

# 1. Introducción

La presente introducción tiene como finalidad definir los lineamientos iniciales para el desarrollo del juego digital Magical Creature propuesto en el marco del reto SenaSoft el juego consiste en una competencia de cartas en la que los jugadores buscan acumular la mayor cantidad de cartas mediante la comparación de atributos de modelos predefinidos.

El sistema se implementará utilizando Java (Spring Boot) para la lógica de negocio y Css para la interfaz gráfica, asegurando una solución eficiente, moderna y atractiva para los participantes, Y de framework utilizaremos Bootstrap para simplificar y facilitar el desarrollo del juego de cartas

## 1.1. Propósito

El propósito de este documento define de manera clara y detallada los requisitos funcionales y no funcionales del sistema Magical Creature un juego de cartas digital desarrollado en el marco del reto SenaSoft, este documento servirá como guía principal para el equipo de desarrollo asegurando que todos los miembros comprendan el alcance, los objetivos y las funcionalidades claves que debe cumplir el sistema.

## 1.2. Ámbito del Sistema

El sistema abarca las siguientes funcionalidades:

- Plataforma web multijugador.
- Creación de salas de juego.
- Reparto aleatorio de cartas entre jugadores.
- Comparación de atributos por rondas.
- Determinación de ganadores.
- Visualización de cartas, atributos, turnos y puntajes.

Tecnologías utilizadas:

Backend: Java con Spring Boot

Frontend: CSS + Bootstrap

Base de datos: MySQL

## 1.3. Definiciones, Acronimos y Abreviaturas

SRS	Especificación de requisitos del software
-----	---

Usuario	Persona que usará el producto
ADSO	Análisis y Desarrollo de Software
RF	Requisitos Funcionales
RFN	Requisitos no Funcionales
MER	Modelo Entidad Relación
GUI	Interfaz gráfica de usuario

#### 1.4. Referencias

SERESMITOLOGICOS.(2018). Seres Mitológicos y Fantásticos.

<https://www.seresmitologicos.net/>

Mary Santos. (2018). Características de un usuario. Prezi.

<https://prezi.com/1hdbbej8zafu/caracteristicas-de-un-usuario/>

Blog de QAT Insights.(2025).Redacción de suposiciones y restricciones en una especificación de requisitos de software (SRS). QAT GLOBAL.

<https://qat.com/writing-assumptions-constraints-srs/>

María Alonso. (2024). Objetivos generales y específicos: Qué son y cómo redactarlos [Plantilla gratis]. Asana

<https://asana.com/es/resources/general-and-specific-objectives>

Alexander Menzinsky, Gertrudis López, Juan Palacio, Miguel Ángel Sobrino, Rubén Álvarez y Verónica Rivas. (2022). Historias de Usuario. Scrum Manager.

[https://www.scrummanager.com/files/scrum\\_manager\\_historias\\_usuario.pdf](https://www.scrummanager.com/files/scrum_manager_historias_usuario.pdf)

Santiago lopez. (2019). Formato requerimientos funcionales. Studocu.

<https://www.studocu.com/co/document/politecnico-grancolombiano/ingenieria-de-software-i/formato-requerimientos-funcionales/14047738>

Ayat Shukairy. (2025). Evaluación heurística: su guía completa para la evaluación heurística y la optimización de la conversión. Invesp.

<https://www.invespcro.com/blog/heuristic-evaluation-your-complete-guide>

## **1.5. Visión General del Documento**

Este documento describe los requisitos necesarios para desarrollar el juego Magical Creature. Incluye una visión general del sistema, sus funciones, restricciones, características del usuario y requerimientos técnicos. Sirve como guía para el equipo de desarrollo y asegura que todos trabajen alineados bajo los mismos objetivos.

## **2. Descripción General**

Magical Creature es un juego de cartas digital multijugador, ambientado en un universo mágico de criaturas del bosque. Cada jugador recibe 8 cartas con atributos numéricos como fuerza, velocidad, magia y resistencia. En cada ronda, se elige un atributo para competir, y quien tenga el valor más alto gana las cartas de esa ronda.

El juego tiene como propósito ofrecer una experiencia dinámica, sencilla y entretenida, reforzando habilidades lógicas y de análisis comparativo.

### **2.1. Perspectiva del Producto**

El juego será desarrollado como una aplicación web. Se Compondrá de dos módulos principales: el cliente (frontend) para interacción visual con las cartas y el servidor (backend) para manejar la lógica, turnos, reparto y resultados.

Se diseñará para ser autosuficiente, ejecutable para navegadores web.

### **2.2. Funciones del Producto**

- Crear salas y definir el número de jugadores.
- Repartir 8 cartas únicas por jugador.
- Visualizar cartas y atributos en pantalla.
- Permitir selección de atributos en cada ronda.
- Comparar valores y asignar cartas al ganador.
- Mostrar puntajes, turnos y resultados.
- Finalizar la partida y declarar un ganador.

### **2.3. Características de los Usuarios**

Nombre del Usuario	Actividad
Administrador	Creación de salas. Elegir número de jugadores por sala
Usuario	Creación de perfil para poder jugar multijugador. Unirse a las salas. Ganar puntos mediante las rondas

### **2.4. Restricciones**

- El proyecto debe ser usado con acceso a internet
- El proyecto debe cumplir con las políticas de seguridad para proteger la información del usuario.
- El proyecto debe de tener un rendimiento rápido y eficiente.
- El proyecto debe de ser intuitivo y fácil de usar.
- El proyecto debe ser compatible con diferentes navegadores web.

### **2.5. Suposiciones y Dependencias**

- Se espera que los usuarios tengan acceso a internet y a algún navegador web.
- Se espera que el usuario tenga conocimientos básicos sobre la temática del juego

## **2.6. Requisitos Futuros**

El planteamiento del problema es una parte esencial en el proyecto de investigación por la falta de dinámicas lúdicas innovadoras dificulta la participación activa y el aprendizaje colaborativo en eventos como SenaSoft. Se requiere un juego que combine estrategia y entretenimiento para motivar a los participantes, facilitando la interacción y el desarrollo de habilidades analíticas en un entorno competitivo.

La justificación de este proyecto es que el desarrollo del juego de cartas denominado Magical Creature responde a la necesidad de una solución tecnológica innovadora y dinámica en el marco del reto SenaSoft. Este proyecto permite integrar conocimientos en programación, diseño de interfaces y gestión de proyectos colaborativos, aplicando metodologías ágiles y buenas prácticas de desarrollo a la implementación del juego que tiene como estrategia la toma de decisiones y la competencia sana entre jugadores, convirtiéndose en una herramienta de entretenimiento interactivo que combina simplicidad en las reglas con dinamismo en la experiencia de usuario. Asimismo, el uso de Java (Spring Boot) para el backend y CSS para el frontend garantiza un diseño muy moderno y una buena interfaz hacia el usuario para que así le pueda agradar. Además, cuenta con un framework que sería Bootstrap que facilita mucho al momento de desarrollar el juego de cartas, por lo cual así cumpliríamos con los estándares actuales de desarrollo web. De esta manera, el proyecto no solo atiende las exigencias del concurso, sino que también constituye una oportunidad para fortalecer las habilidades técnicas del equipo.

## **2.7. Objetivos Generales y Específicos**

Objetivo general: Actualmente no existe una versión digital del juego Magical Creature que permite a varios jugadores competir en tiempo real mediante la comparación de atributos de cartas esto genera la necesidad de desarrollar una plataforma web que gestione las partidas, controle los turnos de manera aleatoria, garantizando una experiencia ágil y entretenida entre 2 y 7 jugadores

Objetivo específicos: Desarrollar un juego de cartas digital multijugador llamado Magical Creature, que permita a los jugadores competir mediante la comparación de atributos que cuenta con algunas cosas específicas:

- Diseñar la estructura de cartas y jugadores para soportar partidas entre 2 y 7 participantes
- Implementar la lógica del juego en Java (Spring Boot), incluyendo el reparto de cartas, control de turnos y cálculo de ganadores
- Cada jugador contendrá 8 cartas aleatoria, con un total de 56 cartas por todos los jugadores que están jugando claro si hay 7 jugadores jugando
- La duración de la partida estará estimado por ahi de 5 o 8 minutos



### 3. Recolección de datos

#### 3.1. Historias de Usuario

Historia de Usuario #1:
Como: jugador nuevo Quiero: ver un tutorial con ejemplos de cartas y jugadas Para: entender rápidamente las reglas sin tener que leer demasiado.
Criterios de Aceptación: Se muestra una diapositiva o pantalla inicial con las reglas resumidas. Incluye imágenes de cartas de ejemplo. El jugador entiende en menos de 2 minutos cómo jugar.

Historia de Usuario #2:
Como: jugador en medio de la partida Quiero: ver claramente cuántas cartas tiene cada participante Para:saber quién va ganando en todo momento.
Criterios de Aceptación: La interfaz muestra el conteo actualizado de cartas por jugador. El número cambia automáticamente cuando se gana o pierde una carta. Se resalta visualmente al jugador con más cartas.

Historia de Usuario #3:
Como: jugador competitivo Quiero: que el sistema me avise cuando cometo un error o jugada inválida Para:saber quién va ganando en todo momento.

**Criterios de Aceptación:**

La interfaz muestra el conteo actualizado de cartas por jugador. El número cambia automáticamente cuando se gana o pierde una carta. Se resalta visualmente al jugador con más cartas.

### **3.2. Entrevista**

¿Cuál considera que es el principal objetivo del reto Siigo SenaSoft?

¿Qué competencias espera que desarrollemos con este tipo de actividades?

¿Qué tan importante es la organización del equipo para lograr buenos resultados en este reto?

¿De qué manera se evaluará nuestro desempeño durante el desarrollo del reto?

¿Qué errores comunes deberíamos evitar al enfrentar un reto como este?

Según su experiencia, ¿qué aspectos son fundamentales para que el juego sea atractivo y funcional?

¿Qué recomendación nos daría para la duración de la partida?

¿Qué tipo de alcance tiene este proyecto?

¿Cómo cree que se debería mejorar la dinámica del juego?

¿Cómo se podría mejorar este juego?

### **3.3. Evaluación Heurística**

- Principio: Visibilidad del estado del sistema

Heurística: El sistema debe mantener a los usuarios informados sobre lo que está pasando, a través de retroalimentación adecuada y de forma clara.

Evaluación: En el juego de cartas, es importante que el jugador siempre sepa en qué parte del juego está, es decir si es su turno, cuantas rondas quedan, cuántas cartas ha ganado, etc.

- Principio: Relación entre el sistema y el mundo real

Heurística: El juego debe usar conceptos familiares para el usuario y basarse en convenciones del mundo real, haciendo que este sea comprensible y accesible.

Evaluación: El uso de atributos de las criaturas místicas como fuerza, velocidad, magia, etc., se alinean con la temática del bosque donde los jugadores pueden identificar fácilmente las cartas y lo que representan.

- Principio: Control y libertad del usuario

Heurística: El usuario debe tener control sobre el sistema y ser capaz de

deshacer o corregir acciones para que este pueda evitar frustraciones.

Evaluación: En el juego actual el jugador no puede deshacer una carta si ya ha sido seleccionada para hacer la jugada, esto se hace para evitar que el jugador no se pueda arrepentir de su selección.

## **4. Requisitos Específicos**

### **4.1. Interfaces Externas**

Interfaz de Usuario (IU): Diseño web con CSS + Bootstrap.

Base de Datos: MySQL para persistencia de usuarios, partidas y estadísticas

API: Desarrollada en Java Spring Boot para la lógica del juego

### **4.2. Funciones**

Aquí se detallan los requisitos funcionales del sistema:

- El sistema debe permitir a los usuarios crear un perfil.
- El sistema debe permitir a los usuarios iniciar y cerrar sesión.
- El administrador de la sala debe poder crear nuevas salas de juego.
- El administrador de la sala debe poder definir el número de jugadores permitidos en una sala (entre 2 y 7).
- Los usuarios deben poder unirse a salas existentes.
- El sistema debe notificar a los jugadores cuando una sala está llena.
- Al inicio de una partida, el sistema debe repartir 8 cartas aleatorias y únicas a cada jugador.
- El total de cartas en el juego es de 56, si lo máximo son 7 jugadores.
- El sistema debe mostrar las cartas en la mano del jugador actual.
- El sistema debe mostrar los atributos de las cartas de forma clara.
- El sistema debe mostrar el número de cartas de cada participante
- El sistema debe indicar claramente de quién es el turno.
- El sistema debe asignar el primer turno de forma aleatoria.
- El jugador en turno debe poder seleccionar un atributo de su carta para competir.
- El sistema debe comparar el valor del atributo seleccionado entre todas las cartas jugadas en la ronda.
- El sistema debe asignar todas las cartas jugadas en la ronda al jugador

con el atributo de mayor valor.

- El jugador que gana la ronda inicia la siguiente.
- El sistema debe indicar claramente de quién es el turno.
- La partida debe finalizar cuando un jugador ha acumulado todas las cartas o cuando se cumpla un tiempo límite estimado.
- El sistema debe declarar al jugador con más cartas al final de la partida como el ganador.
- El sistema debe mostrar un mensaje claro si el jugador intenta realizar una jugada inválida.

#### 4.3. Requisitos de Rendimiento

- Tiempo de respuesta: Las interacciones del usuario (selección de cartas, actualización de puntajes) deben tener un tiempo de respuesta inferior a 2 segundos.
- Concurrencia: El sistema debe soportar al menos 7 partidas simultáneas con 7 jugadores cada una sin degradación significativa del rendimiento.
- Eficiencia en el reparto: El reparto aleatorio de cartas debe ser instantáneo al inicio de la partida.
- Escalabilidad: El diseño del *backend* debe permitir el escalado horizontal para soportar un mayor número de usuarios simultáneos en el futuro.
- Tecnología backend: El backend debe ser desarrollado utilizando Java con Spring Boot.
- Tecnología frontend: El *frontend* debe ser desarrollado utilizando CSS y Bootstrap.
- Base de datos: Se debe utilizar MySQL como sistema de gestión de bases de datos.
- Compatibilidad: La aplicación web debe ser compatible con las últimas versiones de los navegadores web principales (Chrome, Firefox, Edge, Safari).

#### 4.4. Restricciones de Diseño

- Tecnología Backend: El backend debe ser desarrollado utilizando Java con Spring Boot.
- Tecnología Frontend: El frontend debe ser desarrollado utilizando CSS y Bootstrap.

■ **Base de Datos:** Se debe utilizar MySQL como sistema de gestión de bases de datos.

■ **Compatibilidad:** La aplicación web debe ser compatible con las últimas versiones de los navegadores web principales (Chrome, Firefox, Edge, Safari).

#### 4.5. Atributos del Sistema

■ **Fiabilidad:** El sistema debe garantizar la persistencia de los datos de las partidas y perfiles de usuario, con un mínimo de pérdida de información en caso de fallos.

■ **Usabilidad:** La interfaz de usuario debe ser intuitiva y fácil de aprender, incluso para usuarios sin experiencia previa en juegos de cartas digitales.

■ **Mantenibilidad:** El código debe estar bien documentado, modularizado y seguir principios de diseño que faciliten futuras modificaciones y adiciones.

■ **Seguridad:** El sistema debe proteger la información del usuario y las partidas contra accesos no autorizados y manipulaciones.

#### 4.6. Otros Requisitos

##### 4.6.1. Requerimientos Funcionales

<b>Identificador:</b> RF 01		<b>Nombre:</b> Crear perfil	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A		<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> <ul style="list-style-type: none"><li>- Datos del usuario (nombre de usuario, correo electrónico, contraseña, etc.)</li></ul>		<b>Salida:</b> <ul style="list-style-type: none"><li>- Confirmación de creación de perfil o mensaje de error.</li></ul>	

<b>Descripción:</b> El sistema debe permitir a los usuarios registrarse y crear un perfil dentro de la aplicación.
<b>Manejo de situaciones anormales</b> Si los datos ingresados son inválidos (ej. correo electrónico ya registrado, contraseña no cumple requisitos), el sistema debe mostrar un mensaje de error claro y guiar al usuario para corregirlos. Si hay un error de conexión con la base de datos, el sistema debe informar al usuario e intentar nuevamente o indicar que lo intente más tarde.
<b>Criterios de aceptación</b> El usuario puede acceder al formulario de registro. El sistema valida los datos ingresados antes de crear el perfil. Al completar el registro, el usuario recibe una confirmación de que su perfil ha sido creado exitosamente. El usuario puede iniciar sesión con el perfil recién creado.

<b>Identificador:</b> RF 02		<b>Nombre:</b> Iniciar sesión	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> <ul style="list-style-type: none"><li>- Correo electrónico</li><li>- Contraseña</li></ul>		<b>Salida:</b> <ul style="list-style-type: none"><li>- Acceso al sistema o mensaje de error (credenciales incorrectas).</li></ul>	
<b>Descripción:</b> El sistema debe permitir a los usuarios iniciar sesión con sus credenciales registradas			
<b>Manejo de situaciones anormales</b> <ul style="list-style-type: none"><li>● Si las credenciales son incorrectas, el sistema debe mostrar un mensaje indicando que el correo electrónico o la contraseña son incorrectos</li><li>● Opción de "Olvidó su contraseña" para recuperación de acceso</li></ul>			

**Criterios de aceptación**

El usuario puede introducir su correo electrónico y contraseña en el formulario de inicio de sesión. Si las credenciales son válidas, el usuario es redirigido a la interfaz principal del juego. Si las credenciales son inválidas, se muestra un mensaje de error adecuado. La opción de recuperar contraseña funciona correctamente

<b>Identificador:</b> RF 03		<b>Nombre:</b> Cerrar sesion	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Acción del usuario (clic en botón "Cerrar Sesión")		<b>Salida:</b> - Redirección a la página de inicio de sesión o pantalla principal no autenticada	
<b>Descripción:</b> El sistema debe permitir a los usuarios cerrar su sesión activa de forma segura			
<b>Manejo de situaciones anormales</b> Si la sesión ya está expirada o es inválida, el sistema debe manejarlo sin errores y redirigir al usuario a la pantalla de inicio de sesión			
<b>Criterios de aceptación</b> <ul style="list-style-type: none"><li>● Existe un botón o enlace "Cerrar Sesión" visible para los usuarios autenticados</li><li>● Al hacer clic en "Cerrar Sesión", la sesión del usuario se invalida</li><li>● El usuario es redirigido a la página de inicio de sesión o a la pantalla principal sin autenticar</li></ul>			

<b>Identificador:</b> RF 04		<b>Nombre:</b> Crear salas de juego	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A		<b>¿Crítico?</b> Si

<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup
<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Nombre de la sala, número de jugadores (entre 2 y 7)</li> </ul>	<b>Salida:</b> <ul style="list-style-type: none"> <li>- Sala de juego creada y visible para otros usuarios</li> </ul>
<b>Descripción:</b> El administrador de la sala debe poder crear nuevas salas de juego, definiendo el número de jugadores permitidos	
<b>Manejo de situaciones anormales</b>  Si el número de jugadores está fuera del rango permitido (2-7), el sistema debe mostrar un mensaje de error y no permitir la creación de la sala. Si el nombre de la sala ya existe, el sistema debe notificar al administrador y solicitar un nombre diferente	
<b>Criterios de aceptación</b> El administrador tiene una opción para crear una nueva sala. El administrador puede especificar un nombre para la sala. El administrador puede seleccionar el número de jugadores entre 2 y 7. La sala creada aparece en la lista de salas disponibles para otros usuarios.	

<b>Identificador:</b> RF 05		<b>Nombre:</b> Unirse a salas existentes
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	
<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Selección de una sala disponible</li> </ul>		<b>Salida:</b> <ul style="list-style-type: none"> <li>- Usuario unido a la sala o mensaje de sala llena</li> </ul>



<b>Descripción:</b> Los usuarios deben poder unirse a salas de juego existentes
<b>Manejo de situaciones anormales</b> <ul style="list-style-type: none"> <li>• Si el usuario intenta unirse a una sala que ya está llena, el sistema debe mostrar un mensaje claro de "Sala llena"</li> <li>• Si la sala es privada o requiere contraseña (requisito futuro), el sistema debe solicitar la contraseña o negar el acceso</li> </ul>
<b>Criterios de aceptación</b> <ul style="list-style-type: none"> <li>• El usuario puede ver una lista de salas disponibles.</li> <li>• El usuario puede seleccionar una sala para unirse.</li> <li>• Si la sala tiene espacio, el usuario es agregado a la sala.</li> <li>• Si la sala está llena, el sistema notifica al usuario.</li> </ul>

<b>Identificador:</b> RF 06		<b>Nombre:</b> Notificación sala llena	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Intento de unirse a una sala llena		<b>Salida:</b> - Mensaje de notificación "Sala llena"	
<b>Descripción:</b>  El sistema debe notificar a los jugadores cuando una sala está llena y no se puede unir			
<b>Manejo de situaciones anormales</b>			
<b>Criterios de aceptación</b> Cuando un usuario intenta unirse a una sala que ya alcanzó su número máximo de jugadores, se muestra un mensaje de "Sala llena" de forma prominente. El mensaje es claro y comprensible			

<b>Identificador:</b> RF 07		<b>Nombre:</b> Reparto aleatorio de cartas	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Inicio de una partida		<b>Salida:</b> 8 cartas únicas repartidas a cada jugador	
<b>Descripción:</b> Al inicio de una partida, el sistema debe repartir 8 cartas aleatorias y únicas a cada jugador. El total de cartas en el juego es de 56 (máximo 7 jugadores)			
<b>Manejo de situaciones anormales</b> <ul style="list-style-type: none"><li>Si no hay suficientes cartas para repartir 8 a cada jugador (ej. configuración incorrecta), el sistema debe generar un error y no iniciar la partida</li></ul>			
<b>Criterios de aceptación</b> <ul style="list-style-type: none"><li>Se asegura que todas las cartas repartidas sean únicas.</li><li>El número total de cartas en juego sea 56 si hay 7 jugadores, distribuidas</li></ul>			

Identificador: RF 08		Nombre: Mostrar atributos de las cartas	
Tipo: Usuario	Requerimiento que lo utiliza o especializa: N/A	¿Crítico? Si	
Prioridad de desarrollo: Alta	Documentos de visualización asociados:  Mockup		

<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Carta seleccionada o en visualización.</li> </ul>	<b>Salida:</b> <ul style="list-style-type: none"> <li>- Valores de los atributos de la carta.</li> </ul>
<b>Descripción:</b>  El sistema debe mostrar los atributos (fuerza, velocidad, magia, resistencia) de las cartas de forma clara y legible.	
<b>Manejo de situaciones anormales</b>	
<b>Criterios de aceptación</b>  Los atributos de cada carta se muestran de forma legible (ej. con etiquetas y sus valores numéricos).  Los atributos son fácilmente identificables y comparables.	

<b>Identificador:</b> RF 09		<b>Nombre:</b> Mostrar cartas en mano	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Mano del jugador actual		<b>Salida:</b> - Representación visual de las cartas del jugador	
<b>Descripción:</b> El sistema debe mostrar las cartas que el jugador tiene en su mano de forma clara y accesible			
<b>Manejo de situaciones anormales</b>			

<b>Criterios de aceptación</b> <ul style="list-style-type: none"> <li>Las cartas del jugador actual son visibles en su interfaz de juego</li> <li>Cada carta se muestra con su diseño y atributos correspondientes</li> <li>El tamaño y la disposición de las cartas permiten una fácil visualización y selección</li> </ul>

<b>Identificador:</b> RF 10		<b>Nombre:</b> Mostrar número de cartas por participante	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Estado de las cartas de cada jugador.		<b>Salida:</b> - Conteo de cartas por jugador visible en la interfaz.	
<b>Descripción:</b>  El sistema debe mostrar el número de cartas que cada participante tiene en su poder, incluyendo el propio jugador.			
<b>Manejo de situaciones anormales</b>			
<b>Criterios de aceptación</b> La interfaz de juego muestra el número actual de cartas de cada jugador.El conteo de cartas se actualiza en tiempo real después de cada ronda o acción. El número de cartas de cada jugador es visible para todos los participantes			

<b>Identificador:</b> RF 11		<b>Nombre:</b> Indicación de turno	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	

<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	
<b>Entrada:</b> <ul style="list-style-type: none"><li>- Cambio de turno</li></ul>		<b>Salida:</b> <ul style="list-style-type: none"><li>- Resaltado visual o mensaje indicando el jugador en turno.</li></ul>
<b>Descripción:</b>  El sistema debe indicar claramente de quién es el turno de jugar.		
<b>Manejo de situaciones anormales</b>		
<b>Criterios de aceptación</b>  El jugador cuyo turno es se resalta visualmente en la interfaz. Se muestra un mensaje o ícono que indica "Es tu turno" o "Turno de (Nombre del jugador)". La indicación es persistente hasta que el turno cambia.		

<b>Identificador:</b> RF 12		<b>Nombre:</b> Asignación aleatoria del primer turno	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Inicio de partida		<b>Salida:</b> - Jugador inicial seleccionado aleatoriamente	
<b>Descripción:</b>  El sistema debe asignar el primer turno de forma aleatoria al inicio de cada partida.			

<b>Manejo de situaciones anormales</b>
<b>Criterios de aceptación</b> <ul style="list-style-type: none"> <li>● Al comenzar una partida, el sistema elige aleatoriamente a uno de los jugadores para iniciar.</li> <li>● El primer jugador en turno se indica claramente.</li> <li>● La aleatoriedad del turno inicial es consistente y no predecible.</li> </ul>

<b>Identificador:</b> RF 13		<b>Nombre:</b> Selección de atributo para competir	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b>  - Carta en mano del jugador en turno, selección de atributo.		<b>Salida:</b>  - Atributo seleccionado para la ronda	
<b>Descripción:</b>  El jugador en turno debe poder seleccionar un atributo de una de sus cartas para competir en la ronda actual.			
<b>Manejo de situaciones anormales</b>  Si el jugador intenta seleccionar una carta que no está en su mano o un atributo inválido, el sistema debe mostrar un mensaje de error.			
<b>Criterios de aceptación</b>  <ul style="list-style-type: none"><li>● El jugador en turno puede seleccionar una carta de su mano.</li><li>● Al seleccionar una carta, el jugador puede elegir uno de sus atributos para competir.</li></ul>			

- La selección del atributo se confirma antes de proceder con la comparación.

<b>Identificador:</b> RF 14		<b>Nombre:</b> Comparación de atributos	
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b> Mockup		
<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Atributo seleccionado y cartas jugadas por todos los participantes</li> </ul>		<b>Salida:</b> <ul style="list-style-type: none"> <li>- Determinación del ganador de la ronda</li> </ul>	
<b>Descripción:</b>  El sistema debe comparar el valor del atributo seleccionado entre todas las cartas jugadas en la ronda para determinar el ganador.			
<b>Manejo de situaciones anormales</b>			
<b>Criterios de aceptación</b> <ul style="list-style-type: none"> <li>• El sistema compara los valores del atributo elegido de todas las cartas en juego para la ronda</li> <li>• Se identifica correctamente la carta con el valor más alto del atributo</li> <li>• En caso de empate, se define una regla de desempate clara (ej. el que jugó primero o el jugador a la izquierda del actual)</li> </ul>			

<b>Identificador:</b> RF 15	<b>Nombre:</b> Asignación de cartas al ganador de la ronda
-----------------------------	---

<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	
<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Resultado de la comparación de atributos.</li> </ul>		<b>Salida:</b> <ul style="list-style-type: none"> <li>- Cartas de la ronda transferidas al ganador.</li> </ul>
<b>Descripción:</b> El sistema debe asignar todas las cartas jugadas en la ronda al jugador con el atributo de mayor valor.		
<b>Manejo de situaciones anormales</b>		
<b>Criterios de aceptación</b> <ul style="list-style-type: none"> <li>• El jugador ganador de la ronda recibe todas las cartas que fueron jugadas en esa ronda.</li> <li>• Las cartas se añaden a la mano o pila de cartas del jugador ganador.</li> <li>• El número de cartas de cada jugador se actualiza visualmente.</li> </ul>		

<b>Identificador:</b> RF 16		<b>Nombre:</b> Inicio de la siguiente ronda
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	



<b>Entrada:</b> <ul style="list-style-type: none"> <li>- El jugador gana la ronda</li> </ul>	<b>Salida:</b> <ul style="list-style-type: none"> <li>- El sistema inicia la siguiente ronda, con el turno asignado al ganador de la ronda anterior.</li> </ul>
<b>Descripción:</b>  El sistema debe asegurar que el jugador que ganó la ronda anterior sea el que inicie la siguiente ronda.	
<b>Manejo de situaciones anormales</b>	
<b>Criterios de aceptación</b>  El jugador que ganó la ronda anterior es claramente indicado como el jugador en turno al inicio de la nueva ronda. La transición entre rondas es fluida y sin interrupciones.	

<b>Identificador:</b> RF 17		<b>Nombre:</b> Indicador de turno	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Cambio de turno		<b>Salida:</b> - Resaltado visual o mensaje indicando el jugador en turno.	
<b>Descripción:</b>  El sistema debe indicar claramente de quién es el turno de jugar.			
<b>Manejo de situaciones anormales</b>			

**Criterios de aceptación**

El jugador cuyo turno es se resalta visualmente en la interfaz. Se muestra un mensaje o ícono que indica "Es tu turno" o "Turno de (Nombre del jugador)". La indicación es persistente hasta que el turno cambia.

<b>Identificador:</b> RF 18		<b>Nombre:</b> Finalización de partida por cartas	
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A		<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> - Un jugador acumula todas las cartas		<b>Salida:</b> - La partida finaliza	
<b>Descripción:</b> La partida debe finalizar cuando un jugador ha acumulado todas las cartas			
<b>Manejo de situaciones anormales</b>			
<b>Criterios de aceptación</b> <ul style="list-style-type: none"><li>• Cuando un jugador tiene la totalidad de las 56 cartas, el sistema debe reconocerlo automáticamente y finalizar la partida</li><li>• Un mensaje claro de "¡Has ganado todas las cartas!" o similar debe ser mostrado al jugador ganador</li></ul>			

<b>Identificador:</b> RF 19		<b>Nombre:</b> Finalización de partidas por cartas	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	

<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	
<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Un jugador acumula todas las cartas.</li> </ul>		<b>Salida:</b> <ul style="list-style-type: none"> <li>- La partida finaliza.</li> </ul>
<b>Descripción:</b>  La partida debe finalizar cuando un jugador ha acumulado todas las cartas.		
<b>Manejo de situaciones anormales</b>		
<b>Criterios de aceptación</b>  Cuando un jugador tiene la totalidad de las 56 cartas, el sistema debe reconocerlo automáticamente y finalizar la partida. Un mensaje claro de "¡Has ganado todas las cartas!" o similar debe ser mostrado al jugador ganador.-		

<b>Identificador:</b> RF 20		<b>Nombre:</b> Declaración de ganador
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	
<b>Entrada:</b> <ul style="list-style-type: none"> <li>- Finalización de la partida (por cartas o tiempo)</li> </ul>		<b>Salida:</b> <ul style="list-style-type: none"> <li>- El jugador con más cartas es declarado ganador</li> </ul>
<b>Descripción:</b>  El sistema debe declarar al jugador con más cartas al final de la partida como el ganador.		

<b>Manejo de situaciones anormales</b>
<b>Criterios de aceptación</b>  Al finalizar la partida, el sistema compara el número de cartas de cada jugador. El jugador con el mayor número de cartas es identificado y declarado como el ganador.

<b>Identificador:</b> RF 21		<b>Nombre:</b> Mensaje de jugada inválida	
<b>Tipo:</b> Usuario	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> <ul style="list-style-type: none"><li>- Intento de jugada inválida por parte del jugador</li></ul>		<b>Salida:</b> <ul style="list-style-type: none"><li>- Mensaje claro de error indicando la jugada inválida.</li></ul>	
<b>Descripción:</b>  El sistema debe mostrar un mensaje claro si el jugador intenta realizar una jugada inválida (ej. seleccionar una carta que no está en su mano, intentar jugar fuera de turno, seleccionar un atributo no existente).			
<b>Manejo de situaciones anormales</b>			
<b>Criterios de aceptación</b>  Cuando un jugador intenta una acción que no está permitida según las reglas del juego, se muestra instantáneamente un mensaje de error legible y comprensible. El mensaje debe indicar específicamente qué fue inválido y, si es posible, cómo corregirlo.			

#### 4.6.2. Requerimientos No Funcionales

<b>Identificador:</b> RNF 1		<b>Nombre:</b> Tiempo de respuesta	
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b> <ul style="list-style-type: none"><li>- Interacciones del usuario (selección de cartas, actualización de puntajes)</li></ul>		<b>Salida:</b> <ul style="list-style-type: none"><li>- Respuesta de la interacción</li></ul>	
<b>Descripción:</b> Las interacciones del usuario (selección de cartas, actualización de puntajes) deben tener un tiempo de respuesta inferior a 2 segundos.			
<b>Manejo de situaciones anormales</b> Si el tiempo de respuesta excede los 2 segundos, el sistema debe investigar y optimizar los procesos para garantizar la fluidez			
<b>Criterios de aceptación</b> Las interacciones del usuario deben completarse en menos de 2 segundos de forma consistente			

Identificador: RNF 2		Nombre: Concurrencia	
Tipo: Sistema	Requerimiento que lo utiliza o especializa: N/A	¿Crítico? Si	
Prioridad de desarrollo: Alta	Documentos de visualización asociados:  Mockup		

<b>Entrada:</b>  Múltiples partidas y jugadores simultáneos	<b>Salida:</b>  Rendimiento estable del sistema
<b>Descripción:</b>  El sistema debe soportar al menos 7 partidas simultáneas con 7 jugadores cada una sin degradación significativa del rendimiento	
<b>Manejo de situaciones anormales</b>  Si se detecta una degradación del rendimiento con el número especificado de partidas y jugadores, se deben optimizar los recursos del servidor y la lógica del juego.	
<b>Criterios de aceptación</b>  El sistema debe mantener un rendimiento óptimo (sin ralentizaciones o caídas) cuando se ejecutan 7 partidas con 7 jugadores cada una simultáneamente.	

<b>Identificador:</b> RNF 3		<b>Nombre:</b> Eficiencia en el reparto de cartas	
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si	
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b>  Inicio de una partida		<b>Salida:</b>  Cartas repartidas instantáneamente a los jugadores	
<b>Descripción:</b>  El reparto aleatorio de cartas debe ser instantáneo al inicio de la partida.			

<p><b>Manejo de situaciones anormales</b></p> <p>Si el reparto no es instantáneo, se deben revisar y optimizar los algoritmos de reparto y la carga de datos.</p>
<p><b>Criterios de aceptación</b></p> <p>Las 8 cartas únicas por jugador deben ser visibles en la interfaz del jugador inmediatamente después de que la partida comience.</p>

<b>Identificador:</b> RNF 4		<b>Nombre:</b> Escalabilidad	
<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A		<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup		
<b>Entrada:</b>  Futuro aumento de usuarios simultáneo		<b>Salida:</b>  Capacidad de la aplicación para manejar más usuarios	
<b>Descripción:</b>  El diseño del backend debe permitir el escalado horizontal para soportar un mayor número de usuarios simultáneos en el futuro.			
<b>Manejo de situaciones anormales</b> Si el sistema no puede escalar horizontalmente, se debe refactorizar la arquitectura del backend para permitir la adición de más recursos de servidor.			
<b>Criterios de aceptación</b>  La arquitectura del backend debe ser modular y permitir la fácil adición de nuevas instancias de servidor para manejar un aumento de la demanda.			

<b>Identificador:</b> RNF 5	<b>Nombre:</b> Compatibilidad del navegador
-----------------------------	--

<b>Tipo:</b> Sistema	<b>Requerimiento que lo utiliza o especializa:</b> N/A	<b>¿Crítico?</b> Si
<b>Prioridad de desarrollo:</b> Alta	<b>Documentos de visualización asociados:</b>  Mockup	
<b>Entrada:</b> Acceso a la aplicación web desde diferentes navegadores		<b>Salida:</b>  Funcionamiento correcto en navegadores principales
<b>Descripción:</b> La aplicación web debe ser compatible con las últimas versiones de los navegadores web principales (Chrome, Firefox, Edge, Safari).		
<b>Manejo de situaciones anormales</b> Si la aplicación no funciona correctamente en alguno de los navegadores especificados, se deben realizar los ajustes necesarios en el código frontend.		
<b>Criterios de aceptación</b> La aplicación debe cargar y funcionar sin errores visuales o funcionales en las últimas versiones de Chrome, Firefox, Edge y Safari.		

## 5. Apéndices

<https://www.figma.com/design/0u7UkppVARrXQBZBYtlWB/Untitled?m=auto&t=d2bwGI5CkuyjKDwP-6>