

PROYECTO INTEGRADO

VTES-APP

Curso 2023/2024

IES Sotero Hernández



INDICE

Resumen **pag.4**

- Descripción del proyecto.
- Objetivos principales.
- Tecnologías utilizadas.

Introducción **pag.5**

- Contexto y motivación del proyecto.
- Problema que se pretende resolver.
- Objetivos específicos.

Descripción del Proyecto **pag.7**

- Funcionalidades principales.
- Flujo de usuario.

Arquitectura del Sistema **pag.8**

- Descripción general de la arquitectura.
- Diagrama de arquitectura (Frontend, Backend, Base de Datos, Despliegue).

Desarrollo del Proyecto **pag.9**

- Frontend (Angular)
- Estructura del proyecto.
- Componentes principales.
- Servicios utilizados.

- Comunicación con el Backend.
- Backend (Node.js con Express)

Base de Datos (MongoDB) **pag.12**

- Modelo de datos.
- Colecciones y documentos.

Despliegue **pag.13**

- Despliegue en AWS.
- Despliegue en Vercel.

Implementación Técnica **pag.16**

- Configuración del Entorno de Desarrollo
- Requisitos previos.
- Instalación y configuración de dependencias

Pruebas **pag.17**

- Herramientas utilizadas.

Guia de Usuario **pag.18**

Conclusiones **pag.31**

- Resumen de lo logrado.
- Dificultades encontradas.
- Posibles mejoras y futuras ampliaciones.

Referencias **pag.32**

- Bibliografía y recursos utilizados.

RESUMEN

Estoy desarrollando una aplicación utilizando Node.js, Express, MongoDB, Angular y Bootstrap. La aplicación es un conjunto de herramientas diseñado para facilitar la creación de mazos de un juego de cartas llamado Vampire: The Eternal Struggle (Vtes). La aplicación incluye varias secciones, como el listado y filtrado de cartas, la construcción y listado de mazos, un sistema de registro de usuarios con seguridad mejorada, la creación y gestión de cartas y el registro de torneos y su gestión.

Despliegue:

<https://vtesapp.duckdns.org/> - (backend)

<https://front-vtes.vercel.app> - (front)

Nota: Hoy, a día 25/05/2024 he tenido que detener el backend para no superar la capa gratuita, en caso de querer revisarlo antes del 1 de junio póngase en contacto conmigo.

Para acceder a cualquier cuenta la contraseña es "Vtes1" , y los usuarios van de test1@gmail.com a test20@gmail.com,

test1 es administrador.

INTRODUCCIÓN

La aplicación consiste en varias secciones:

Listado y filtrado de cartas:

- Dos páginas separadas, una para listar y filtrar los personajes y otra para listar y filtrar las cartas de la biblioteca. Para ello, me apoyaré en una API externa con la información de todas las cartas: [API de cartas](#).

Construcción y listado de mazos:

- Página con un formulario dividido en dos partes, Cripta y Biblioteca, donde se podrán añadir dinámicamente cartas de las listas anteriores para la creación de mazos.
- Los usuarios podrán guardar los mazos creados de manera pública para toda la comunidad o de forma privada.
- Opción para eliminar el mazo o eliminar una única fila dentro de la tabla.
- Los mazos podrán tener múltiples copias de la misma carta.
- Botones para imprimir el mazo en formato TXT o PDF con las imágenes de las cartas.
- Botón para realizar una copia de un mazo ya creado.
- Importación de listas de mazos en formato TXT.
- Página para listar todos los mazos propios o de la comunidad, permitiendo el filtrado por tipo o por autor.

Sistema de registro de usuarios:

- Mejora de seguridad utilizando JWT para asignar un token a la sesión y bcrypt para encriptar las contraseñas.
- Sistema de roles (SuperAdmin, Admin, Colaborador, Usuario) para acceder a diferentes opciones de la página.
- Guardianes en Angular para el acceso a las páginas según roles y autenticación.
- Los usuarios podrán añadir un avatar personalizado.

Creación y visionado de cartas personalizadas:

- Los usuarios podrán añadir cartas personalizadas desde un formulario para añadir información relevante de la misma.
- Los usuarios podrán visualizar las cartas creadas por la comunidad y filtrar por autor o nombre de la carta.

Creación y gestión de eventos:

- Los eventos serán torneos presenciales del juego de cartas, conteniendo toda la información relevante (día, hora, lugar, número máximo de jugadores, listado de jugadores inscritos).
- Listado de todos los jugadores inscritos con un campo indicando si han realizado el pago o están pendientes.
- Asignación aleatoria de posiciones para los jugadores y distribución en mesas.
- Visualización de mesas con el avatar, nombre de los jugadores y sus posiciones.
- Los torneos constan de 3 rondas y una final, con reglas específicas para evitar que los jugadores repitan posición en la mesa con respecto a otros jugadores en rondas consecutivas.
- Sistema de puntos basado en puntos de victoria y mesas obtenidas, con reglas detalladas para la obtención de estos puntos.

Perfil de Usuario:

- La aplicación tendrá diferentes roles para permitir el acceso a las páginas o funcionalidades. Admin, user, invitado, colaborador.
- Los invitados tendrán solo acceso de lectura a la aplicación sin posibilidad de editar.
- Los usuarios registrados podrán crear cartas y mazos personalizados y participar en foros aunque no los crearon.
- Los colaboradores tendrán los mismos permisos que los usuarios y también podrán crear eventos y nuevos foros de discusión.
- Y el rol de Administrador permitirá todo lo anterior y la posibilidad de eliminar o bloquear usuarios.

Épica de la Aplicación:

- La aplicación permitirá a los usuarios registrados la creación de nuevos mazos y la impresión de las cartas. Los mazos podrán ser públicos para todos los usuarios o no.
- La aplicación permitirá el diseño personalizados de cartas que podrán almacenarse como publicas o privadas en el perfil del usuario.
- La aplicación contendrá diferentes sistemas de filtrado para tipos de carta, dependiendo de si son cartas de personajes o de librería(cartas que pueden jugar los personajes), y otros filtrados más específicos por diferentes campos: nombre, tipo, title, disciplinas...
- Creación de eventos para gestionar partidas, con un panel personalizado para cada evento que muestre los participantes, las mesas asignadas a cada participante, la posición y los puntos obtenidos en el torneo.
- Listado de mazos y cartas personalizadas (publicas)
- (Si me da tiempo) Foros de discusión para diversos temas del juego.
- La aplicación deberá manejar los errores, mostrando mensajes personalizados para cada caso, implementando dichos mensajes en los modelos.
- La aplicación permitirá a los administradores eliminar o modificar permisos de otros usuarios, en caso de eliminar los usuarios se eliminaran todos los campos asociados.
- Los usuarios podrán eliminar mazos de sus libreras, eliminándolo a su vez de la lista publica en caso de existir.

DESCRIPCIÓN DEL PROYECTO

- Los usuarios podrán acceder a dos páginas separadas, una para listar y filtrar los personajes y otra para listar y filtrar las cartas de la biblioteca.
- La información de las cartas se obtendrá a través de una API externa.
- Los usuarios tendrán la capacidad de buscar y filtrar las cartas por diferentes criterios, como nombre, tipo, atributos, etc.
- Construcción y listado de mazos:
- Los usuarios contarán con un formulario dividido en dos secciones: Cripta y Biblioteca, donde podrán agregar dinámicamente las cartas seleccionadas para crear sus propios mazos.
- Los usuarios tendrán la opción de guardar los mazos creados, ya sea de forma pública para que toda la comunidad pueda verlos, o de forma privada.
- Existirá la posibilidad de eliminar un mazo completo o eliminar una única fila dentro de la tabla del mazo.
- Los usuarios podrán imprimir el mazo en formato TXT o PDF, incluyendo las imágenes de las cartas.
- Habrá una opción para realizar una copia de un mazo ya creado.
- Los usuarios podrán mandar reportes al administrador sobre cartas que les parezcan inapropiadas.
- Se implementará un sistema para la creación de torneos y el reparto de los jugadores en las diferentes mesas.
- Se contará con un panel de Administrador para poder ver los reportes de la comunidad, eliminar eventos, usuarios o cartas.

Sistema de registro de usuarios:

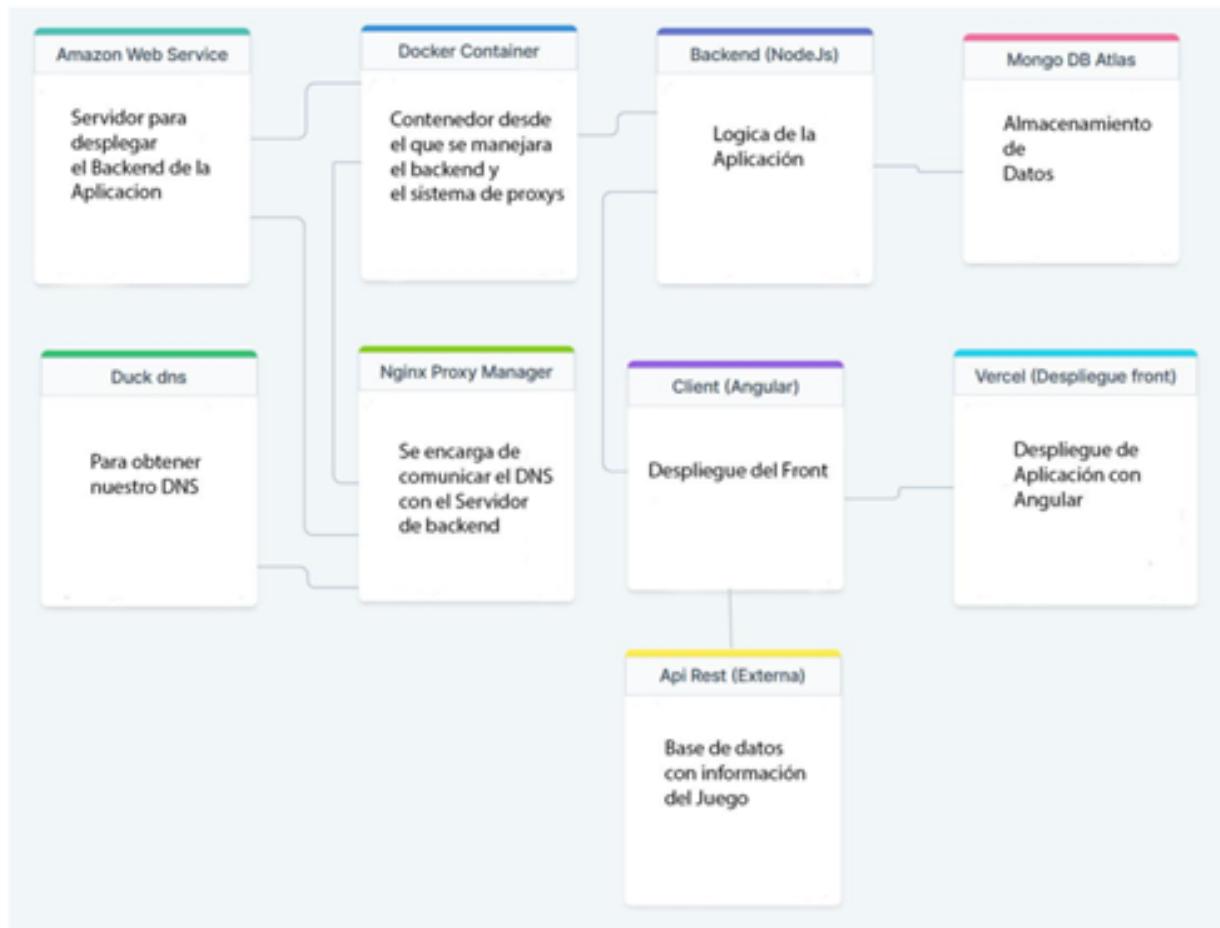
- Se implementará un sistema de registro y autenticación de usuarios con mejoras de seguridad, utilizando JWT y bcrypt.
- Habrá un sistema de roles (SuperAdmin, Admin, Colaborador, Usuario) para controlar el acceso a diferentes opciones de la aplicación.
- Los usuarios podrán agregar un avatar personalizado a su perfil.

ARQUITECTURA DEL SISTEMA

Descripción general de la arquitectura:

- La arquitectura de la aplicación sigue un enfoque de microservicios, donde el Frontend, el Backend y la Base de Datos se encuentran desacoplados y se comunican entre sí a través de API REST.
- Esta arquitectura modular permite una mayor escalabilidad, mantenibilidad y flexibilidad en el desarrollo y despliegue de la aplicación.
- El Frontend de la aplicación está desarrollado utilizando el framework Angular, que se encarga de la interacción con el usuario, la gestión de la interfaz de usuario y la lógica de presentación.
- El Backend, por su parte, está construido sobre Node.js y Express, y se encarga de la lógica de negocio, la gestión de datos y la exposición de los endpoints API.
- La Base de Datos utilizada es MongoDB, una base de datos NoSQL orientada a documentos, que permite almacenar y recuperar eficientemente la información relacionada con los usuarios, los mazos, las cartas y los eventos.

Diagrama de arquitectura:



DESARROLLO DEL PROYECTO

Estructura del proyecto Frontend (Angular):

- La estructura del proyecto Angular sigue la convención de Angular CLI, con las siguientes carpetas principales:
 - **src/app:** Contiene los componentes, servicios, pipes, y demás archivos de la aplicación.
 - **src/assets:** Almacena los recursos estáticos, como imágenes, fuentes y archivos de configuración.
 - **src/environments:** Contiene los archivos de configuración específicos para cada entorno (desarrollo, producción, etc.).

- Se ha utilizado la estructura de carpetas recomendada por Angular, donde cada componente, servicio y módulo se encuentra en su propia carpeta.

Componentes principales:

- **inicioComponent**: Muestra una página de presentación.
- **LoginComponent**: Maneja el proceso de inicio de sesión y registro de usuarios.
- **RegisterComponent**: Permite el registro de Usuarios.
- **Biblioteca y CryptoComponent**: Muestra el listado de cartas y permite filtrarlas por diferentes criterios.
- **Deck y ListaDeck**: Proporciona la funcionalidad de construcción , gestión de mazos y su listado.
- **Custom y uploadCustomCard**: Permite a los usuarios ver y añadir las cartas personalizadas, incluyendo la opción de reportes.
- **Event y Lista de Eventos**: Gestiona la creación, visualización y administración de eventos (torneos).

Servicios utilizados:

- **AuthService**: Encargado de la autenticación y autorización de usuarios, utilizando JWT.
- **CardService**: Proporciona métodos para obtener y filtrar la información de las cartas a través de la API.
- **DeckService**: Maneja la creación, actualización y eliminación de mazos.
- **EventService**: Permite la creación, actualización y consulta de eventos (torneos).
- **ReportService**: Permite enviar reportes que se almacenan en la bbdd.

Pipes Personalizadas:

- Diferentes pipes para el filtrado tanto de cartas como eventos o usuarios.

Comunicación con el Backend:

- Los servicios Angular se encargan de realizar las llamadas HTTP a los endpoints del Backend.
- Utilizan el módulo HttpClientModule de Angular para enviar solicitudes HTTP y recibir las respuestas.
- Implementan métodos que abstraen la lógica de comunicación con el Backend, ocultando los detalles de implementación a los componentes.
- Cuando se requiere autenticación, los servicios incluyen el token JWT en los encabezados de las solicitudes.

Estructura del proyecto Backend (Node.js con Express):

- La estructura del proyecto Backend sigue el patrón de diseño MVC (Modelo-Vista-Controlador).
- Las principales carpetas son:
 - **routes**: Contiene los archivos de definición de rutas y sus respectivos controladores.
 - **controllers**: Implementa la lógica de negocio y maneja las solicitudes HTTP.
 - **models**: Define los esquemas y modelos de datos utilizados en la aplicación.
 - **middleware**: Contiene funciones de middleware personalizadas, como la autenticación y autorización.
 - **service**: Almacena los archivos de configuración, como la conexión a la base de datos.

Principales endpoints:

- Login para el inicio de sesión de los usuarios.
- Metodos para la creación de nuevas cartas y su modificación.
- Metodo para obtener todas las cartas oficiales del juego.
- Metodo para almacenar las imagenes de las cartas y redimensionar las imagenes.
- Metodo para la impresion en formato txt con un listado completo de las cartas de los mazos
- Metodo para la impresion en formato pdf de las imagenes de las cartas
- Metodo para añadir usuarios a los eventos.

- Metodo para que los administradores puedan añadir nuevos usuarios a los eventos.
- Metodo para generar nuevos torneos.
- Metodo para actualizar los avatares de los usuarios
- Metodo para enviar reportes a los administradores sobre las cartas personalizadas
- Metodo para registrar nuevos usuarios

Middleware utilizado:

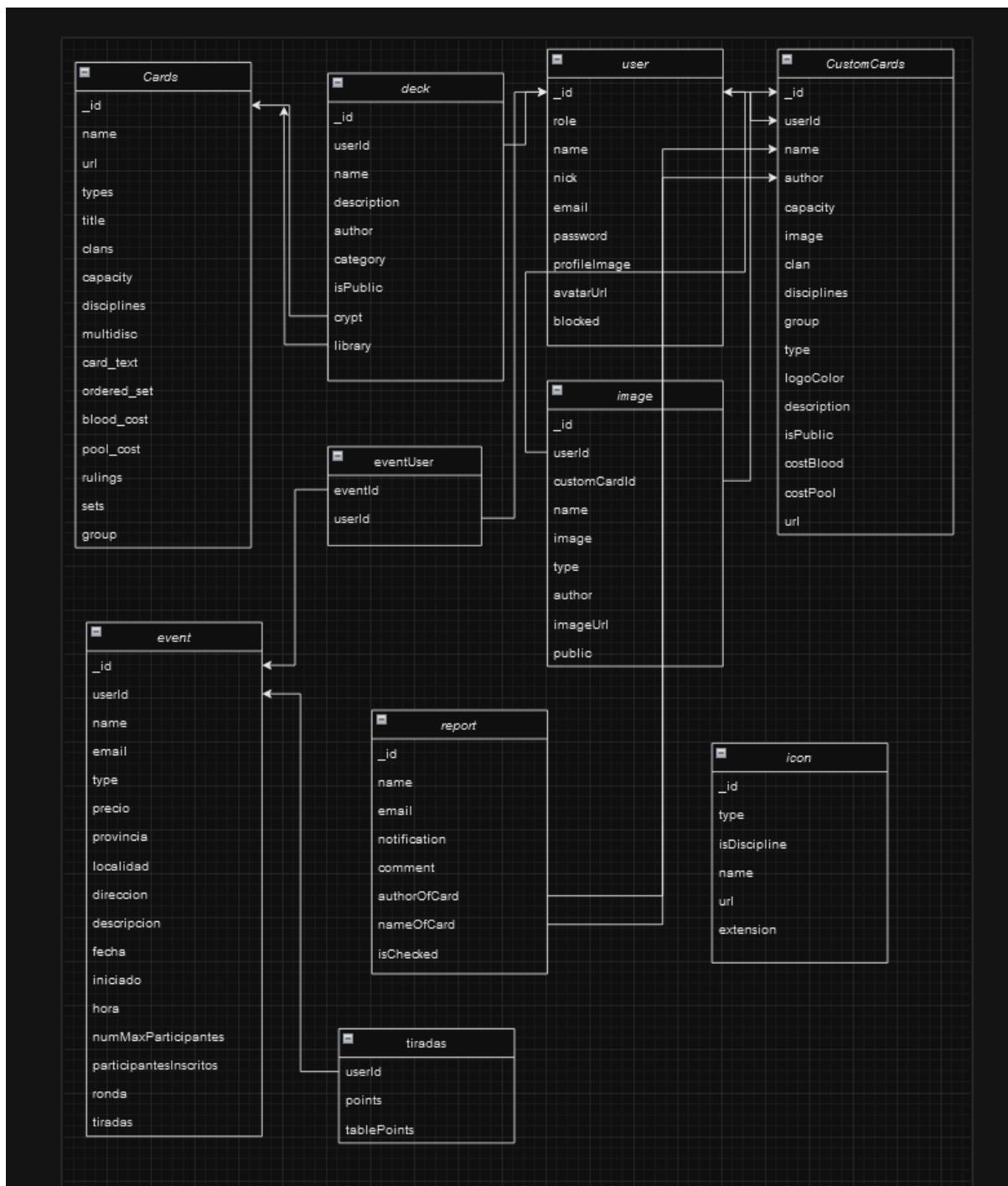
- **auth:** Verifica la autenticación y autorización de los usuarios mediante el token JWT.
- **validationMiddleware:** Realiza la validación de los datos enviados en las solicitudes HTTP.
- **error:** Maneja los errores que puedan ocurrir durante la ejecución del Backend.
- **Multer:** Para gestionar la subida de imágenes y redimensionarlas.
- **passwordHash:** Para encriptar las contraseñas de los usuarios.
- **printPDF:** Para la impresión de las imágenes en formato PDF.
- **printTxt:** Para imprimir el listado completo de las cartas en un mazo.

BASE DE DATOS (MongoDB)

Modelo de datos:

- La aplicación utiliza MongoDB como sistema de gestión de base de datos NoSQL. El modelo de datos se ha diseñado de acuerdo con las necesidades de la aplicación y sigue una estructura orientada a documentos.

COLECCIONES:



DESPLIEGUE

Despliegue de la Aplicación

Generación de Contenedor Docker

- Crear un archivo Dockerfile en el proyecto con la siguiente configuración:
 - Utilizar la imagen base de Node.js versión 20.
 - Establecer el directorio de trabajo en /app.
 - Copiar el archivo package.json y ejecutar npm install para instalar las dependencias.
 - Copiar el código de la aplicación.
 - Exponer el puerto 3000.
 - Establecer el comando de inicio de la aplicación.

Crear un archivo docker-compose.yml en el mismo directorio con la siguiente configuración:

- Definir el servicio app que utiliza la imagen construida en el paso anterior.
- Mapear el puerto 3000 del contenedor al puerto 3000 del host.
- Establecer las variables de entorno necesarias, como la URL de MongoDB.

Construir la imagen de Docker ejecutando el comando: docker build -t whejhe/vtes-backend-app .

- Publicar la imagen en Docker Hub: docker push whejhe/vtes-backend-app

Creación de Instancia en AWS

- Ir a la consola de AWS y seleccionar "Elastic Container Service" (ECS).
- Crear un nuevo clúster seleccionando "EC2 Linux + Networking".
- Configurar la instancia EC2, incluyendo el tipo de instancia, el par de claves y las opciones de red.
- Agregar un nuevo contenedor, seleccionando la imagen publicada en Docker Hub.

- Configurar los puertos y las variables de entorno según sea necesario.
- Crear el clúster.

Integración Continua

- Agregar las claves necesarias para la integración con GitHub.
- Configurar el despliegue automático en GitHub Actions o en un servicio de integración continua.

Despliegue del Frontend con Vercel

- Conectar el repositorio de GitHub a Vercel.
- Configurar las variables de entorno necesarias.
- Vercel se encargará del despliegue automático del frontend.

Relanzar Instancia

- Ir a la consola de AWS EC2 y seleccionar la instancia.
- Iniciar o detener la instancia según sea necesario.
- Copiar la dirección IP pública de la instancia.

Actualizar Dirección IP en GitHub

- Ir al repositorio de GitHub del proyecto.
- Actualizar el secreto AWS_Backend_host con la nueva dirección IP pública.

Comprobar Ejecución del Proxy Manager

- Conectarse a la instancia de AWS mediante SSH: ssh -i ubuntu.pem ubuntu@(ip_publica)
- Verificar el estado del contenedor Docker: sudo docker ps
- Revisar los registros del contenedor: sudo docker logs (id_contenedor)

Comandos Útiles

- sudo docker-compose up -d: Iniciar los contenedores en segundo plano.
- sudo docker-compose down: Detener los contenedores.

- sudo docker-compose pull: Actualizar las imágenes de los contenedores.
- sudo docker ps: Listar los contenedores en ejecución.
- sudo docker logs (id_contenedor): Ver los registros de un contenedor.
- sudo docker exec -it (id_contenedor) sh: Acceder al interior de un contenedor.

IMPLEMENTACIÓN TÉCNICA

Configuración del Entorno de Desarrollo

Requisitos previos:

- Tener instalado Node.js versión 20.
- Tener instalado Angular CLI versión 17.
- Tener instalado MongoDB versión 4.4 o superior.
- Tener instalado un editor de código, Visual Studio Code.

Instalación y configuración de dependencias:

Front:

Instalación global de Angular

npm install -g @angular/cli

Crear un nuevo proyecto Angular utilizando Angular CLI:

Ng new vtes-frontend

Instalar las dependencias necesarias para el proyecto:

Npm install

Otras dependencias Instaladas

npm install yarn:

Yarn es un administrador de paquetes que se utiliza para instalar, actualizar y administrar dependencias en proyectos de Node.js. Ofrece una alternativa a npm (Node Package Manager) y se caracteriza por ser más rápido y seguro.

npm install hammerjs:

Hammer.js es una biblioteca de JavaScript que proporciona reconocimiento de gestos multi-táctiles para aplicaciones web y móviles. Se utiliza para agregar funcionalidades de gestos a elementos HTML, como swipe, tap, pinch, etc.

ng add @angular/material:

Angular Material es una biblioteca de componentes de interfaz de usuario (UI) para Angular. Proporciona una amplia variedad de componentes visuales y directivas para crear aplicaciones con una apariencia y comportamiento consistentes con el diseño de Material Design de Google.

npm install bootstrap --save:

Bootstrap es un framework de CSS y JavaScript que se utiliza para crear aplicaciones web con una apariencia y comportamiento consistentes. Ofrece una amplia variedad de componentes visuales, como barras de navegación, formularios, botones, etc., y es ampliamente utilizado para crear aplicaciones web responsivas.

npm i bootstrap@5.3.3:

Esta instrucción instala una versión específica de Bootstrap (5.3.3). Esto es útil cuando se necesita una versión determinada de la biblioteca para mantener la compatibilidad con otros proyectos o para evitar problemas de versiones.

npm install jwt-decode:

jwt-decode es una biblioteca de JavaScript que se utiliza para decodificar tokens JSON Web Tokens (JWT). Los JWT son una forma común de autenticar y autorizar a usuarios en aplicaciones web, y esta biblioteca facilita el proceso de decodificar y verificar la autenticidad de estos tokens.

Comando para generar archivos:

ng generate service services/json-service:

Este comando genera un nuevo servicio en Angular llamado json-service dentro de la carpeta services. Un servicio es una clase que proporciona funcionalidades que pueden ser utilizadas en diferentes partes de la aplicación.

ng generate guard guards:

Este comando genera un nuevo guard en Angular llamado guards. Un guard es una clase que se utiliza para controlar el acceso a rutas en una aplicación Angular, permitiendo o denegando el acceso a una ruta según ciertas condiciones.

ng generate component components/header:

Este comando genera un nuevo componente en Angular llamado header dentro de la carpeta components. Un componente es una parte fundamental de una aplicación Angular que representa una sección de la interfaz de usuario.

ng generate component pages/main:

Este comando genera un nuevo componente en Angular llamado main dentro de la carpeta pages. Este componente probablemente se utilizará como la página principal de la aplicación.

ng generate module pages/pages:

Este comando genera un nuevo módulo en Angular llamado pages dentro de la carpeta pages. Un módulo es una forma de organizar y agrupar componentes, servicios y otros elementos relacionados en una aplicación Angular.

ng generate module components --module=app:

Este comando genera un nuevo módulo en Angular llamado components y lo agrega al módulo principal app. Esto permite agrupar componentes relacionados en un módulo separado.

ng generate pipe pipes/my-custom:

Este comando genera un nuevo pipe en Angular llamado my-custom dentro de la carpeta pipes. Un pipe es una clase que se utiliza para transformar y formatear datos en una aplicación Angular.

ng g environments:

Este comando genera archivos de configuración de entorno para una aplicación Angular. Estos archivos permiten definir variables de entorno que se pueden utilizar en diferentes entornos, como desarrollo, producción, etc.

Backend:

- Instalar Node.js desde el sitio web oficial: <https://nodejs.org>
- Verificar la versión instalada:

Node –versión:

Este comando muestra la versión actual de Node.js instalada en el sistema.

Crear un nuevo proyecto de Node.js y Express:

npm init -y:

Dependencias instaladas:

- Express.js, un framework de Node.js para crear aplicaciones web.
- Morgan, un middleware de registro de solicitudes HTTP para Node.js.
- Multer, un middleware para manejar archivos subidos en Node.js.
- Mongoose, una biblioteca de Node.js para interactuar con bases de datos MongoDB.
- Nodemon, una herramienta que reinicia automáticamente el servidor Node.js cuando se realizan cambios en el código. La opción -D indica que se instala como una dependencia de desarrollo.
- Dotenv, una biblioteca que carga variables de entorno desde un archivo .env en Node.js.
- Cross-env, una biblioteca que permite definir variables de entorno de manera consistente en diferentes plataformas.
- CORS (Cross-Origin Resource Sharing), una biblioteca que permite habilitar el acceso a recursos desde orígenes cruzados en Node.js.

- Bcrypt, una biblioteca para hashear contraseñas de manera segura.
 - JSON Web Tokens (JWT), una biblioteca para crear y manejar tokens de autenticación.
 - Nodemailer, una biblioteca para enviar correos electrónicos desde Node.js.
 - PDFKit, una biblioteca para generar archivos PDF en Node.js.
 - fs (File System), que es un módulo nativo de Node.js para interactuar con el sistema de archivos.
 - Sharp, una biblioteca para procesar imágenes en Node.js, y sus tipos de TypeScript (@types/sharp).
 - Got, una biblioteca para realizar solicitudes HTTP en Node.js de manera sencilla y eficiente.
-

PRUEBAS

Herramientas utilizadas:

- **Postman:** Para pruebas de integración de la API.

GUIA DE USUARIOS

INICIO:

La página de inicio contiene una breve introducción del juego y enlaces con las diferentes secciones de la página.

Vampire: The Eternal Struggle (VTES) es un juego de cartas colecciónables ambientado en el universo de Vampiro: La Mascarada, uno de los juegos de rol más populares del mundo. Lanzado por primera vez en 1994 por la editorial White Wolf, VTES se ha convertido en un clásico entre los amantes de los juegos de cartas estratégicos.

En VTES, los jugadores asumen el papel de poderosos vampiros que luchan por el control de la ciudad. Cada jugador construye un mazo de cartas que representa su clan, disciplinas y recursos, con el objetivo de eliminar a sus oponentes y convertirse en el vampiro dominante.

La historia de VTES se remonta a los orígenes del juego de rol Vampiro: La Mascarada, que introdujo a los jugadores en el fascinante mundo de las criaturas de la noche. Cuando White Wolf decidió adaptar este universo a un juego de cartas, la comunidad de jugadores acogió la idea con entusiasmo.

A lo largo de los años, VTES ha mantenido una base de seguidores fieles gracias a su profunda estrategia, su rica ambientación y su constante evolución. Cada expansión y set de cartas ha traído nuevas mecánicas, clanes y personajes, manteniendo fresco y emocionante el juego para los veteranos y atrayendo a nuevos jugadores.

Hoy en día, VTES sigue siendo un juego de culto entre los amantes de los juegos de cartas estratégicos. Su comunidad internacional organiza torneos y eventos en todo el mundo, donde los jugadores se reúnen para demostrar sus habilidades y disfrutar de la emoción de la eterna lucha entre los vampiros.

Listado de Todas las Cartas del juego

El listado de Cartas en la página web de Elysium Eternal permite a los usuarios explorar y filtrar la extensa biblioteca de cartas del juego. Los visitantes podrán buscar por diversos criterios, como nombre, clan, disciplina y tipo de carta, facilitando la identificación de las cartas que necesitan para sus mazos.

Cada carta tendrá su propia página de detalles, con

Activar Windows
Ve a Configuración para activar

MENU:

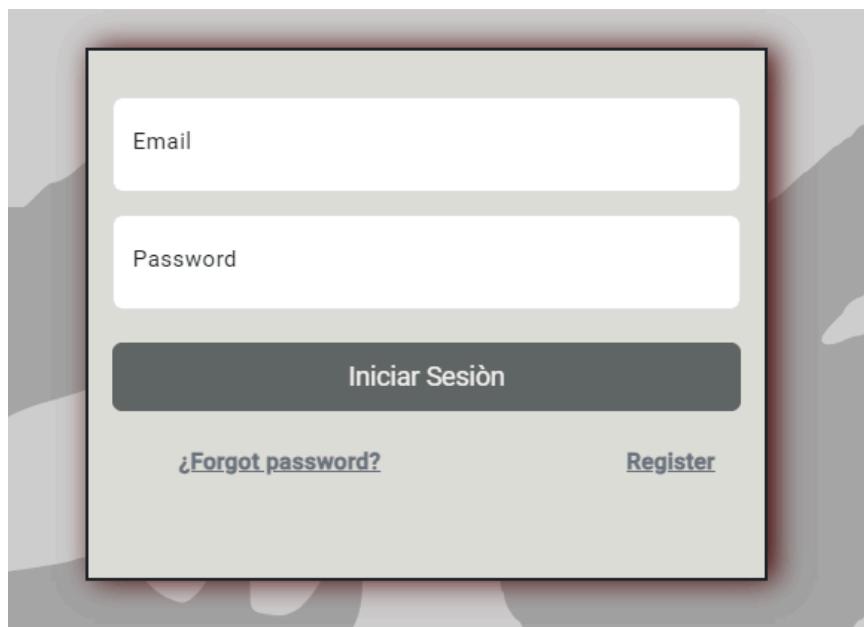
El menu se encuentra presente en toda la aplicación y contiene una lista con los enlaces necesarios para navegar por las diferentes páginas, el título de la página que también es un enlace con la página de inicio y una zona para el avatar del usuario que te manda al login si no tienes sesión abierta.

Elysium Eternal

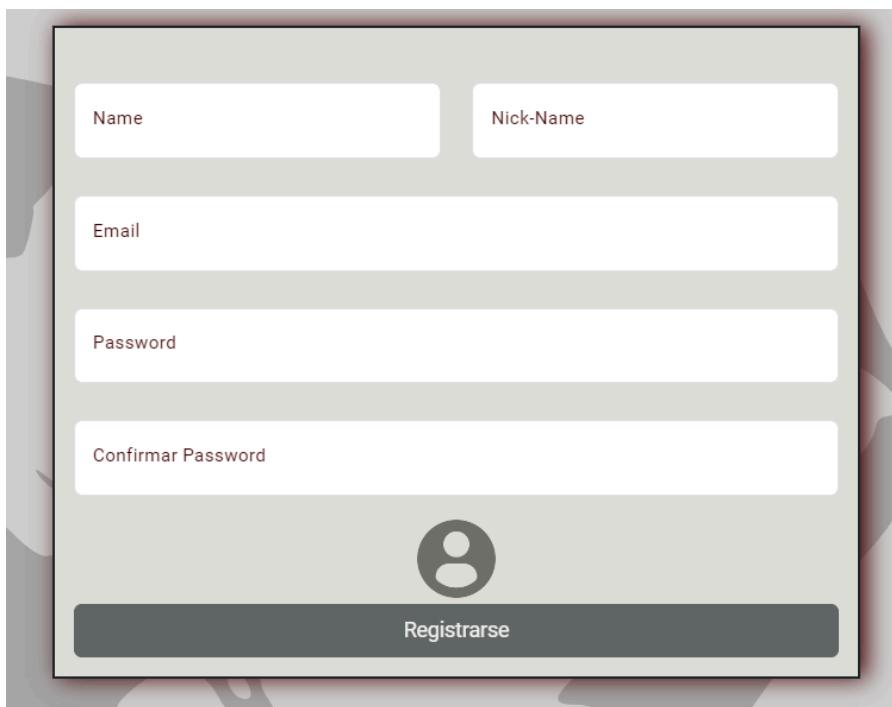
Home List Cards Custom Cards Decks Events

LOGIN:

El login contiene un enlace a la página de registro.

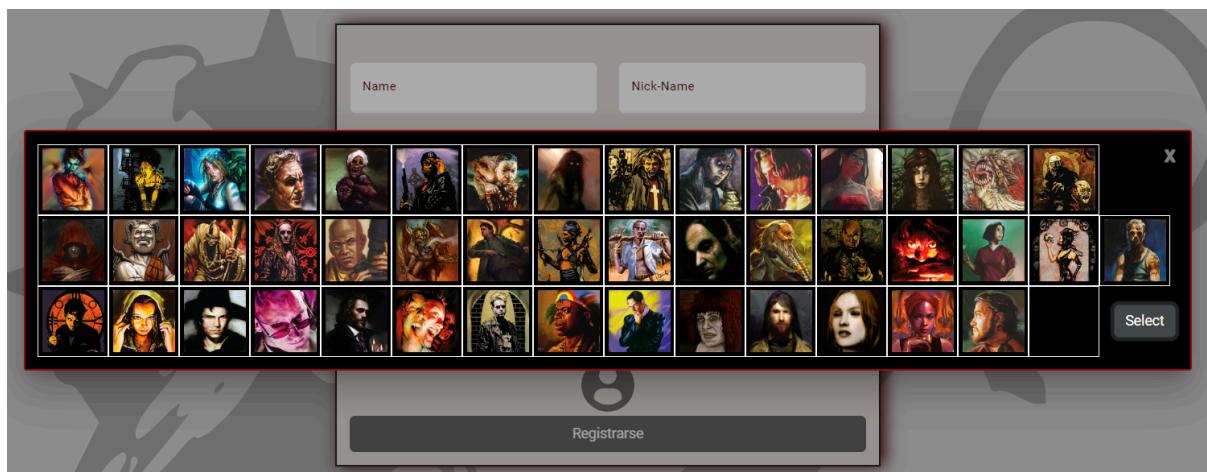


REGISTRO:



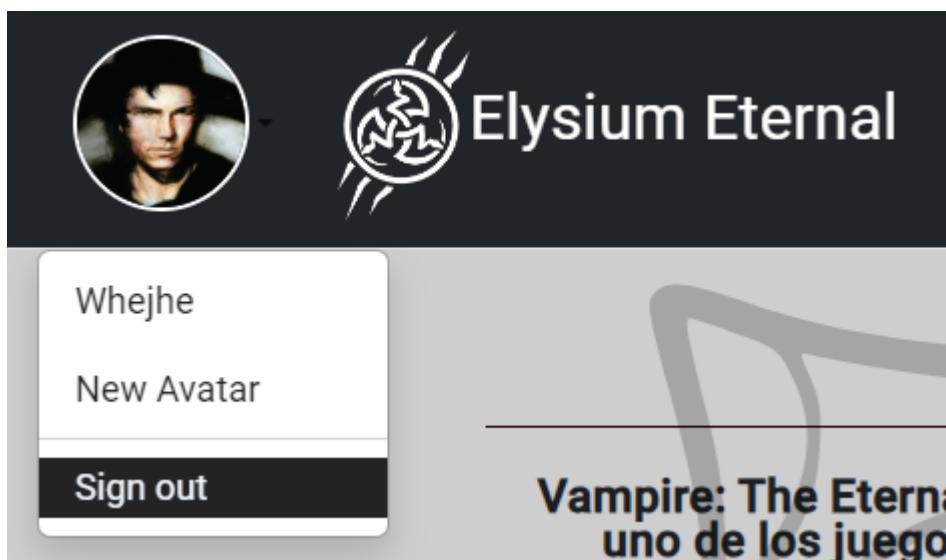
AVATAR-REGISTRO:

Desde el registro puedes añadir un avatar predeterminado para los usuarios.



DESPLEGABLE-AVATAR:

Una vezlogueado el avatar aparecerá en la cabecera y contendrá un desplegable con el nick del usuario y una opción para cerrar la sesión.



CRIPTA:

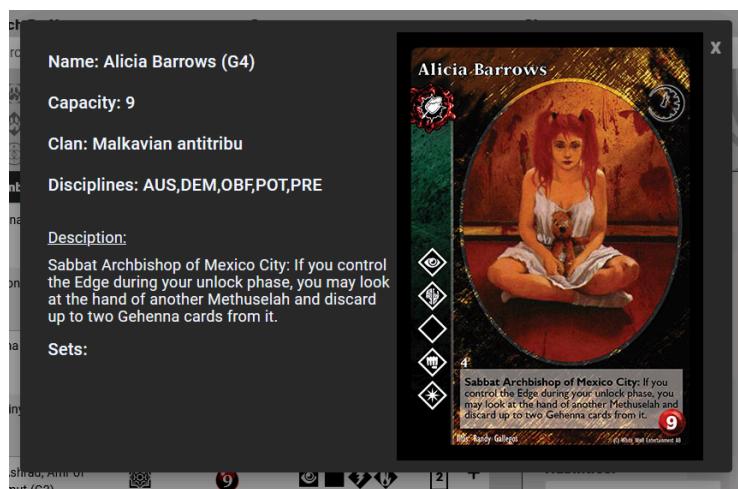
En la página de cripta los usuarios podrán visualizar todas las cartas de personajes con múltiples filtros para facilitar la búsqueda de estos, también he añadido una pequeña caja con enlaces relacionados con el juego.

The screenshot shows a search interface for a database of characters. At the top, there are three search fields: 'Search By Name:' with a dropdown menu for 'Group' and a list of clan symbols; 'Clan:' with a dropdown menu; and 'Search By Card Text:' with dropdown menus for 'Title', 'Sect', and 'Habilities', and input fields for 'Min. Capacity' and 'Max. Capacity'. Below these is a table of character cards:

Nombre	Clan	Capacidad	Disciplinas	Grupo
Aabbt Kindred (G2)	∅	4	∅ * ?	2 +
Aaron Bathurst (G4)	∅	4	∅ ■ ■	4 +
Aaron Duggan, Cameron's Toady (G2)	∅	2	∅	2 +
Abaddon (G7)	∅	8	∅ ■ □ △ ▯	7 +
Abd al-Rashid (G2)	∅	5	■ ■	2 +
Abdelsobek (G5)	∅	5	∅ ■ ■ ■ ■ ■	5 +
Abebe (G4)	∅	4	∅ ■ ■ ■	4 +
Abid bin Haji, Scholar of Mirrors (G6)	∅	7	∅ ■ ■ ■ ■ ■	6 +
Abiku (G4)	∅	6	∅ ■ ■ ■ ■ ■	4 +

VISTA-CARTAS:

Las cartas cuentan con un enlace para visualizar la información relevante.



BIBLIOTECA:

Desde el enlace de biblioteca se podrá acceder a todas las cartas de librería, y al igual que en cripta cuenta con multiples filtros para las búsquedas.

The screenshot shows a library search interface with various filters and a card list. At the top, there are three search fields: 'Search By Name:', 'Type:', and 'Clan:'. Below these are several filter icons. A large table lists cards with columns for 'Nombre', 'Disciplines', 'Tipo', 'Coste', and 'Agregar'. The table includes entries like '.44 Magnum' (Equipment), '419 Operation' (Action), '47th Street Royals' (Ally), etc. To the right, there's a sidebar with 'Search By Card Text:' and 'Search By Traits:' fields. A bottom right corner button says 'Activar Windows' and 'Ve a Configuración'.

Nombre	Disciplines	Tipo	Coste	Agregar
.44 Magnum		Equipment	2	+
419 Operation		Action		+
47th Street Royals		Ally		+
Aaron's Feeding Razor		Equipment	1	+
Abactor		Action		+
Abandoning the Flesh	¶	Combat,Reaction		+
Abbot		Action		+
Abjure	◎	Power		+
Ablative Skin	▲	Action		+
Abombwe		Master		+

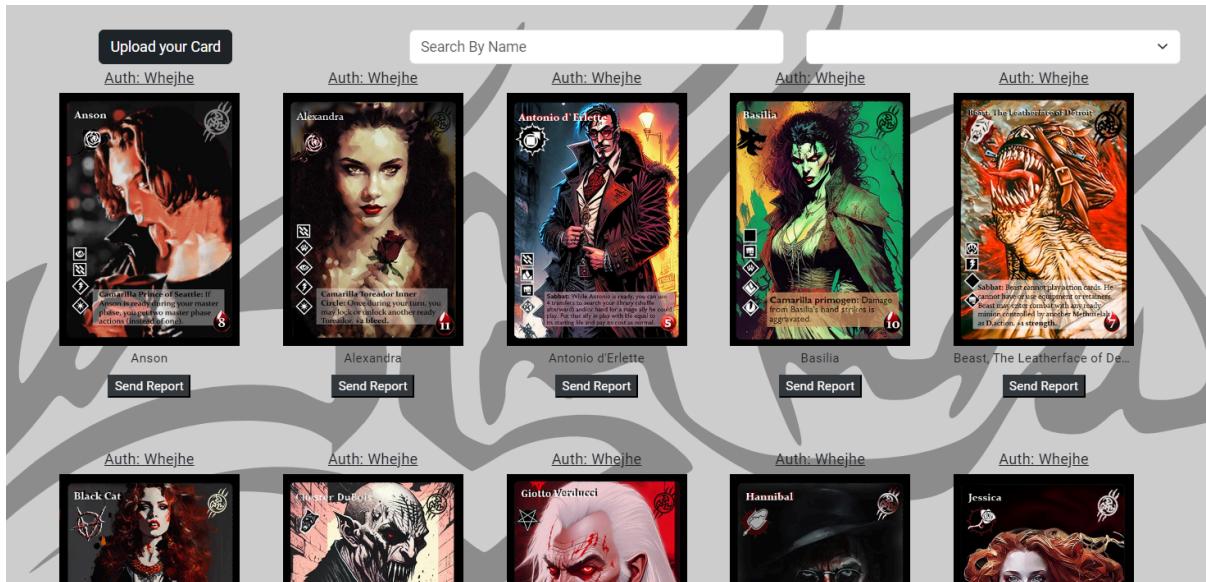
UPLOAD-CARD:

Desde el enlace de custom los usuarios pueden añadir cartas que ellos mismos hayan diseñado y añadirles la información relevante de las mismas.

The screenshot shows a card creation form. At the top is a toolbar with various icons. Below it are fields for 'Name:' (with placeholder 'Nombre'), 'Image:' (with 'Seleccionar archivo' and 'Ningú...ionado'), 'Capacity:' (set to 1), 'Clan:', 'Group:' (set to 'Publico'), and 'Type:'. A large text area labeled 'not defined' is at the bottom left. A 'Guardar' (Save) button is at the bottom right.

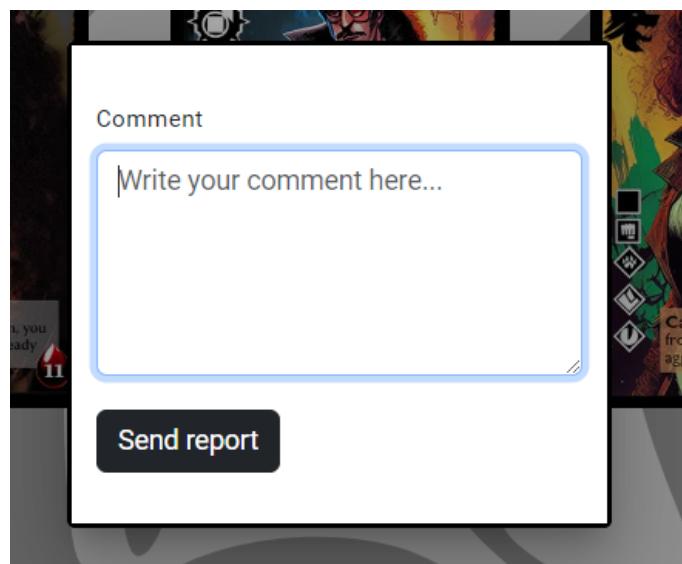
GALERÍA:

En galería los usuarios pueden ver todas las cartas creadas por la comunidad, buscar por nombre o por autor y mandar reportes al administrador en caso de que piensen que la carta no es apropiada.



SEND-REPORT:

Los reportes solo contendrán visualmente un campo de texto para los reportes, pero tambien se añaden como campos ocultos el nombre de la carta ,el autor de esta, y el email y el nombre de quien manda el reporte.



LIST OF DECKS:

Desde la sección de deck pueden visualizarse todos los mazos públicos creados por la comunidad, junto a unos filtros por autor y tipo (mecánicas del juego).

Author	Type			
All Authors	All			
Name	Type	Description	Author	Nº Cards
Using the Advantage	Bleed	Mazo de Sangrada con Enkil Cog para mantener el fi...	Whejhe	98
Brujah Bleed y Señales de tráfico	Combat	Mazo de Potencia, Presencia y Celeridad	Whejhe	102
Nephandus	Aily	Mazo de Tremores Antivirus con aliados y Frontal A...	Whejhe	102
Malkavian Derange	Bleed	Mazo de Bleed y Derange	Whejhe	102
Secret from the Magaji	Wall	Mazo de Magajis con Animalismo y Anarch Convert	Whejhe	102
Sin nombre		Descripción	Whejhe	0
Copy of Copy of Using the Advantage	Bleed	Mazo de bleed basado en cailean	Whejhe	102
Sin nombre		Descripción	Whejhe	0
Copy of Sin nombre		Descripción	Whejhe	1
Copy of Brujah Bleed y Señales de tráfico	Combat	Mazo de Potencia, Presencia y Celeridad	mig	87

DECKS:

Desde el apartado de decks los usuarios pueden crear sus propios mazos, para añadir las cartas tiene un input de búsqueda para añadir el nombre que desplegará las posibles opciones. Cada carta cuenta con un campo para aumentar o reducir el numero de copias y otro para eliminar la carta.

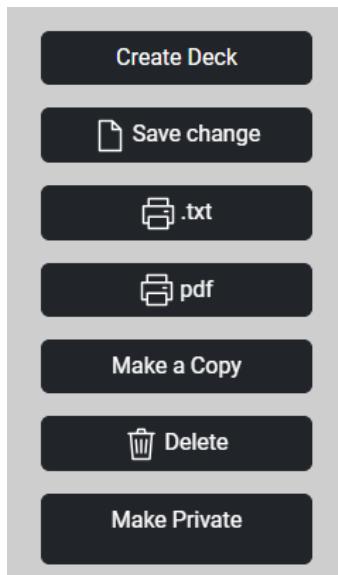
The screenshot shows the 'Deck' creation interface. At the top, there are search fields for 'Name' (containing 'mig') and 'Type' (set to 'Combat'). Below these are two main sections: 'Crypt' and 'Library'. The 'Crypt' section lists cards with their counts and trash icons. The 'Library' section lists cards with their counts and trash icons. To the right, there is a sidebar with various buttons: 'Create Deck', 'Save change', '.txt', '.pdf', 'Make a Copy', 'Delete', and 'Make Private'. A watermark for 'Activar Windows' is visible at the bottom right.

Name	Count
Theo Bell (G6)	2
Aline Gádeke (G6)	2
Chun Hei (G6)	2
Gold Pan Dan (G5)	2
Octane (G6)	2
Tara (G5)	2

Name	Count
Esprit de Corps	4
Show of Force	11
Iron Glare	11
Immortal Grapple	8
Pursuit	5
Quickness	8
Roundhouse	9
Slam	5
Taste of Vitae	5
Torn Signpost	9

BUTTONS-DECK:

Desde el menú de botones se puede generar una nueva vista en blanco para la creación de mazo, salvar los cambios realizados, imprimir la lista de todas las cartas en formato txt, imprimir las imágenes de las cartas en formato PDF, hacer una copia completa de un mazo existente para hacer modificaciones si lo desean, eliminar el mazo en caso de ser el propietario y cambiar la visibilidad de los mismos.



LIST OF EVENTS:

Otra sección con la lista de los torneos y la información relevante.

Evento	Fecha	Provincia	Localidad	
nuevo evento	2024-06-10	Sevilla	Nervion	
Nuevo	2024-05-21	Provincia	Localidad	
hjkdlshdlsf	2024-05-22	Provincia	Localidad	
grgd	2024-05-09	Provincia	Localidad	
dsdd	2024-05-23	Provincia	Localidad	
dawad	2024-05-01	Provincia	Localidad	
Ultimo Torneo	2024-06-01	Provincia	Localidad	
fsefesdfesd	2024-05-24	Provincia	Localidad	

EVENTOS:

Cada evento cuenta con un panel con la información del mismo, un temporizador con el tiempo restante para el comienzo y una serie de botones para darse de baja o alta en el torneo. En caso de ser administrador tambien tendrá un botón para dar comienzo a los torneos.

Los usuarios apuntados aparecerán en una lista.

The screenshot shows a tournament registration interface. At the top right is a button labeled "Agregar Usuario por Correo Electrónico". Below it, the event title is listed as "Titulo: fsefesdfesd". A large red "X" is overlaid on the right side of the page. On the left, it says "Comienza en 0 Dias" and "22h:31:37s". The event details include:

- Type: Torneo Europeo
- Provincia: Provincia
- Localidad: Localidad
- Direccion:
- Fecha: 2024-05-24
- Hora: 10:00
- Max. Participantes:
- Description:

At the bottom are buttons for "Apuntarse", "Darse de Baja", "Back to list of Events", "Comenzar torneo", and "Ranking".

Below this, a section titled "Lista de Jugadores Inscritos" displays a table of registered players:

Image	Participantes	Email	Acciones
	Carlos	dominguezalacid@gmail.com	
	test1	test1@gmail.com	
	test2	test2@gmail.com	
	test3	test3@gmail.com	
	test4	test4@gmail.com	

COMENZAR-TORNEO:

Desde el botón de comenzar torneo se generar las mesas de forma aleatoria para las tres rondas del torneo, para cada mesa se pueden añadir manualmente los puntos obtenidos por los jugadores que actualizaran una lista general de todos los jugadores para llevar el ranking.

Ronda N° 1			Ronda N° 2		
Mesa N° 1	Ronda N° 1	Update Points	Mesa N° 1	Ronda N° 2	Update Points
test5	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test3	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test9	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	Carlos	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test1	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test1	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test2	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test4	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test7	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test7	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
Mesa N° 2	Ronda N° 1	Update Points	Mesa N° 2	Ronda N° 2	Update Points
test8	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test9	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test3	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test8	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
Carlos	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test5	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test6	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test2	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
test4	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>	test6	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>
Ronda N° 3					
Mesa N° 1	Ronda N° 3	Update Points			
test5	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>			
test8	<input type="text" value="0"/>	Mesa: <input type="checkbox"/>			

CREAR EVENTO:

Otro formulario para gestionar la creación de los torneos.

The screenshot shows a form for creating a tournament. It includes fields for:

- Nombre del Evento
- Provincia
- Direccion
- Categoría del Torneo
- Hora: example: 10:00
- Localidad
- Precio
- Nº Max. Participantes
- Fecha: dd/mm/aaaa
- Añade la información relevante (with a text area)
- Enviar button

ADMIN-LISTA:

Desde el panel de administración se pueden listar los usuarios para eliminarlos o modificar los roles (permisos).

The screenshot shows the "Panel Admin" interface with a table of users:

Nombre	Nick	Email	Rol	Acciones
Carlos	Whejhe	dominguezalacid@gmail.com	SUPER_ADMIN	ADMIN Eliminar
test2	test2	test2@gmail.com	COLLABORATOR	COLLABORATOR Eliminar
test3	test3	test3@gmail.com	USER	USER Eliminar
test4	test4	test4@gmail.com	USER	USER Eliminar
test5	test5	test5@gmail.com	USER	USER Eliminar
test6	test6	test6@gmail.com	USER	USER Eliminar
test7	test7	test7@gmail.com	USER	USER Eliminar

LISTA-REPORTE:

Desde aqui los administradores pueden ver los reportes de las cartas.

<u>Panel Admin</u>					
<u>List of Reports</u>					
Nombre Remitente	View	Email Remitente	Author of Card	Name of Card	
Carlos		dominguezalacid@gmail.com	Whejhe	Basilia	
Carlos		dominguezalacid@gmail.com	Whejhe	Chester DuBois	
Carlos		dominguezalacid@gmail.com	Whejhe	Antonio d'Erlette	

VISTA DE REPORTES:

Los reportes incluyen la referencia a la carta y el usuario que realiza el reporte. También un campo checkbox para marcar el reporte como leído.

Nombre Remitente: Carlos	Email Remitente: dominguezalacid@gmail.com
Author of Card: Whejhe	Name of Card: Basilia
Comentario: <p> Lorem ipsum dolor, sit amet consectetur adipisicing elit. Quasi eos ut quae, natus recusandae sed veritatis ducimus asperiores nesciunt laboriosam, dolor dignissimos a reprehenderit itaque repudiandae, quia voluptate maxime obcaecati. Non tempora, reprehenderit maxime omnis excepturi tempore quam, alias id rerum nesciunt dolore consectetur est! Ab voluptatum reiciendis dolor amet nulla officia nisi vel laudantium debitis qui. Fuga, voluptate aspernatur! Magnam odio sint debitis, nostrum nisi molestiae ipsam laboriosam eum, dignissimos, dolore beatae neque dolorem enim placeat exercitationem nulla quis maiores architecto et quo. Doloremque itaque alias quisquam accusamus? Modi. Rem aliquid repellendus perspiciatis officiis nulla quisquam debitis nemo veritatis sed est? Facilis, dolore ducimus! Atque libero hic, esse dolorum maiores necessitatibus quis numquam est quo sint ipsum, suscipit autem! Accusamus voluptates totam unde fugiat minus hic repellendus magni deserunt neque? Doloribus eum tenetur laboriosam veniam est deleniti eius repudiandae qui reprehenderit cupiditate recusandae nostrum neque possimus, modi optio amet. Quam, expedita tempora rerum quos ipsam architecto enim ad aut earum dicta provident eum odit obcaecati, beatae, mollitia quas dolorum unde eligendi voluptas doloremque fuga suscipit. Itaque necessitatibus minima suscipit! Molestiae, maiores iure cum, dolorum ullam omnis dolore nam quidem enim ipsa, inventore perspiciatis dicta est delectus rem velit excepturi pariatur expedita eum. Fugit illo eveniet placeat maxime aliquid! Laboriosam. Nemo repudiandae iure sunt, repellendus vero ipsa. Modi corporis odio magni ea, nobis delectus iure ut nisi, fuga repudiandae iusto voluptates nulla. Ratione consequuntur nam voluptatibus, quae querat distinctio maxime. Sequi recusandae fugit deleniti sunt temporibus enim aliquam repudiandae tempora sint? Nobis molestiae maxime exercitationem. Delectus voluptatem optio recusandae, cumque, facere quia, eos a ea possimus illum maiores doloremque ipsam! Iure deserunt fugit laboriosam error fugiat vero dicta ab nulla ratione rerum, incidentur saepe pariatur inventore cupiditate minima. In error harum sapiente sequi explicabo. Ducimus minima dignissimos recusandae in hic?</p>	Marcar como leido: <input checked="" type="checkbox"/>

LISTA DE EVENTOS:

Desde aqui los administradores pueden listar y eliminar los eventos.

Panel Admin				
List of Users	Reports	Events	Custom Cards	List of Decks
Evento	Fecha	Provincia	Localidad	
nuevo evento	2024-06-10	Sevilla	Nervion	
Nuevo	2024-05-21	Provincia	Localidad	
hjkdlfhdsif	2024-05-22	Provincia	Localidad	
grgdg	2024-05-09	Provincia	Localidad	
dsdd	2024-05-23	Provincia	Localidad	
dawad	2024-05-01	Provincia	Localidad	
Ultimo Torneo	2024-06-01	Provincia	Localidad	
fsefesdfesd	2024-05-24	Provincia	Localidad	

LISTA DE CARTAS:

Otra sección para lista y eliminar las cartas personalizadas.

Panel Admin				
List of Users	Reports	Events	Custom Cards	List of Decks
Custom Cards				
Name	Type	Author		
Anson	Vampire	Whejhe		
Alexandra	Vampire	Whejhe		
Antonio d'Erlette	Vampire	Whejhe		
Basilia	Vampire	Whejhe		
Beast, The Leatherface of Detroit	Vampire	Whejhe		
Black Cat	Vampire	Whejhe		
Chester DuBois	Vampire	Whejhe		
Giotto Verducci	Vampire	Whejhe		
Hannibal	Vampire	Whejhe		
Jessica	Vampire	Whejhe		
Jessica-Adv	Vamnire	Whejhe		

LISTA DE MAZOS:

Y una más para listar los mazos.

Name	Type	Description	Author	Nº Cards	Action
Using the Advantage	Bleed	Mazo de Sangrada con Enkil Cog para mantener el fi...	Whejhe	98	
Brujah Bleed y Señales de trafico	Combat	Mazo de Potencia, Presencia y Celeridad	Whejhe	102	
Nephandus	Ally	Mazo de Tremores Antirivus con aliados y Frontal A...	Whejhe	102	
Malkavian Derange	Bleed	Mazo de Bleed y Derange	Whejhe	102	
Secret from the Magaji	Wall	Mazo de Magajis con Animalismo y Anarch Convert	Whejhe	102	
Sin nombre		Descripcion	Whejhe	0	
Copy of Copy of Using the Advantage	Bleed	Mazo de bleed basado en cailean	Whejhe	102	
Sin nombre		Descripcion	Whejhe	0	
Copy of Sin nombre		Descripcion	Whejhe	1	

CONCLUSIONES

Resumen de lo logrado:

- Desarrollamos una aplicación completa para la gestión de mazos y eventos del juego de cartas Vtes, utilizando tecnologías modernas y prácticas de desarrollo seguro.
- Implementamos un sistema robusto de autenticación y autorización.
- Proporcionamos herramientas para la creación, visualización e impresión de mazos.
- Facilitamos la gestión de eventos, incluyendo la asignación aleatoria de posiciones y el seguimiento de inscripciones.

Dificultades encontradas:

- Gestión de la autenticación y autorización entre el frontend y el backend.
- Optimización de la comunicación entre los servicios para mejorar el rendimiento.
- Implementación de las reglas específicas para la organización de torneos.

Posibles mejoras y futuras ampliaciones:

- Implementar un sistema de notificaciones en tiempo real para actualizaciones de eventos y mazos.
- Mejorar la interfaz de usuario con funcionalidades adicionales y una experiencia más intuitiva.
- Ampliar las funcionalidades de filtrado y búsqueda de cartas y mazos.

REFERENCIAS

- **Bibliografía y recursos utilizados:**
 - Documentación oficial de Node.js: [Node.js](#)
 - Documentación oficial de Angular: [Angular](#)
 - Documentación oficial de MongoDB: [MongoDB](#)
 - Guía de Docker: [Docker](#)
 - Guía de AWS ECS: [AWS ECS](#)