NOMBRE DE LA INSTITUCIÓN:
Tecnológico de Monterrey

ENTREGABLE:
Etapa 2. Diseño

MATERIA:
TC3002C.101 Ciberseguridad informática II

INTEGRANTES:
Axel Ariel Grande Ruiz A01611811
Carlos Eugenio Saldaña Tijerina A01285600
Humberto Jasso Silva A01771184
Isaac Hernández Pérez A01198674
Víctor Misael Escalante Alvarado A01741176

NOMBRE DEL PROFESOR:
Luis Alberto Terrazas

FECHA:
1 de noviembre del 2025

Problemática que se va a resolver

El Museo MARCO ha enfrentado una *disminución significativa de visitantes* tras la pandemia en su mayoría debido a: cierres temporales, reducción de aforo y pérdida de interés del público. Esta situación afecta su misión de promover el arte contemporáneo y sensibilizar a la sociedad, reduciendo su alcance educativo y cultural.

Siguiendo el principio de "Security by Design", cada componente del sistema ha sido diseñado considerando la protección de datos, la prevención de vulnerabilidades y el cumplimiento de los requerimientos no funcionales de seguridad establecidos en el entregable anterior (Etapa 1. Requerimientos). El enfoque adoptado garantiza que la seguridad no sea un añadido posterior, sino un elemento fundamental de la arquitectura desde su concepción.

Propuesta de Diseño de Base de Datos

El presente esquema propone una base de datos que considera proteger la información sensible tanto de los usuarios como del sistema de la aplicación. Esto mediante hashing de contraseñas (considerado con Argon2), cifrado de correos electrónicos y datos personales, campos de auditoría con timestamps de la creación de los usuarios, y tokenización de información de pago sin almacenar datos completos de los metodos de pago de los usuarios (ubnicamente el estado de los pagos y la cantidad). Si bien se han tomado en cuenta distintas medidas de seguridad para la protección del intercambio de datos entre la base de datos y la aplicación. También se ha diseñado para que, en caso de ocurrir una fuga de información, no constituya un peligro para los usuarios, puesto que no se almacenan datos completos o en texto plano que dejen vulnerables a los usuarios.

La prevención de inyecciones se logra mediante queries parametrizadas obligatorias en PostgreSQL (Sequelize/TypeORM), validación exhaustiva de esquemas con tipos explícitos y validadores personalizados, sanitización que elimina operadores peligrosos como dollar-where, whitelist de caracteres permitidos, y longitudes máximas estrictas. El control de acceso implementa usuarios específicos con permisos granulares donde el servicio API solo tiene SELECT, INSERT y UPDATE en tablas necesarias.

Es de este modo que se propone el siguiente diseño de entidades para la base de datos:

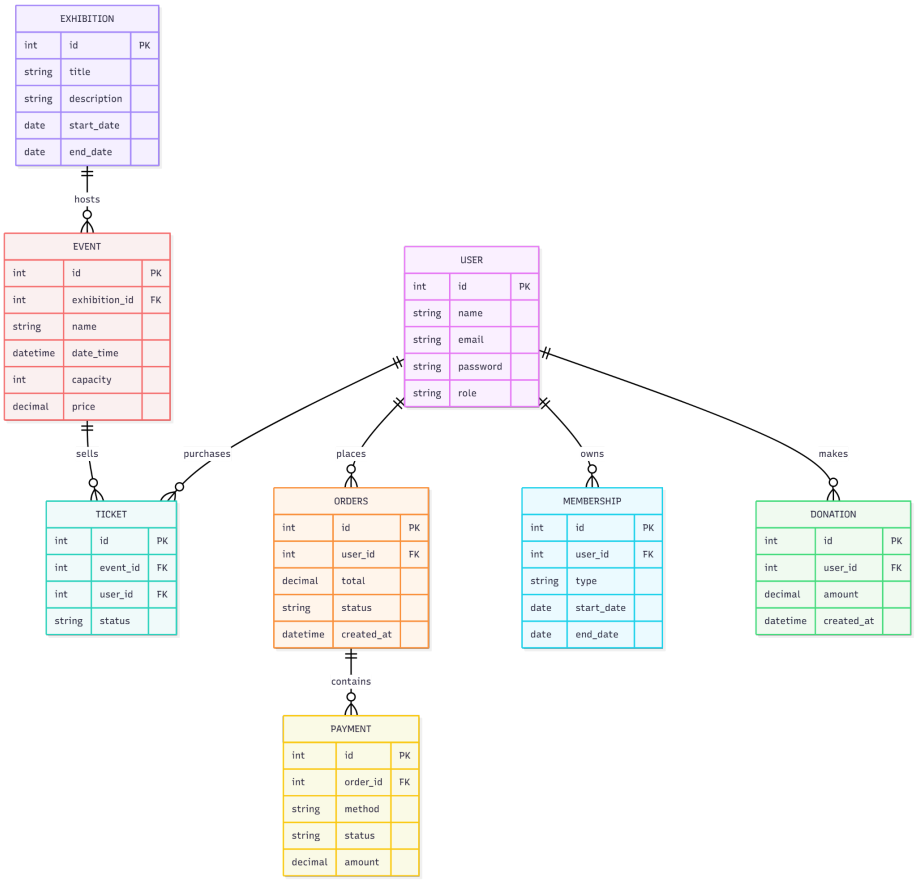


Diagrama de Arquitectura

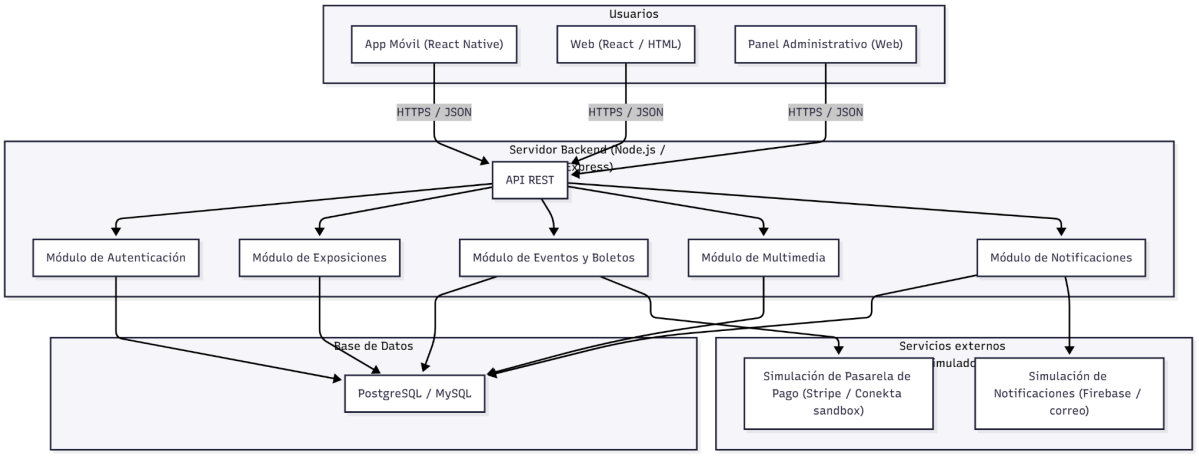
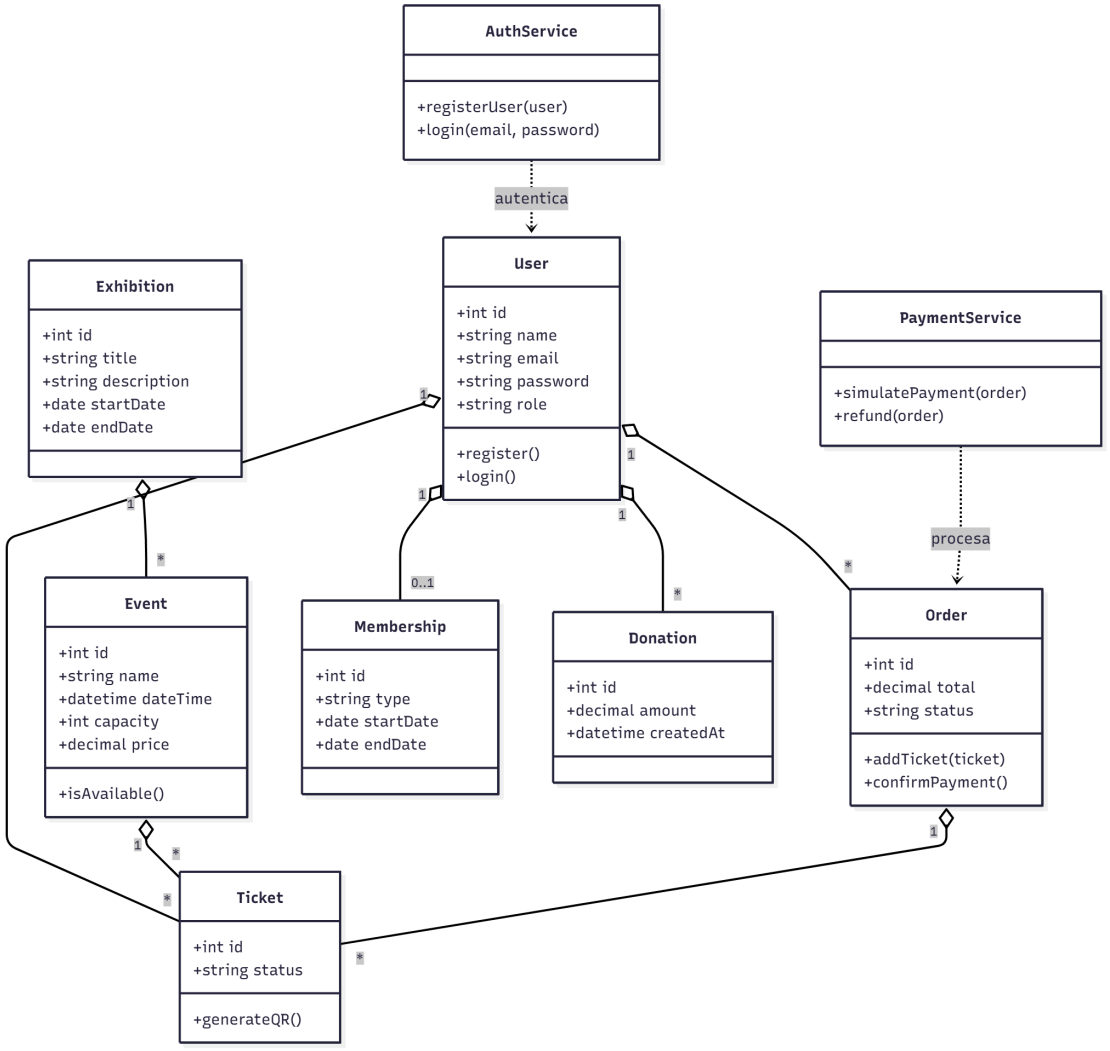


Diagrama de Clases



Diseño de las interfaces de usuario y Protección de Frontend

Aplicación Móvil (React Native)

La aplicación móvil está desarrollada en React Native, con un enfoque en seguridad multiplataforma. Para el almacenamiento seguro, se utiliza react-native-keychain, que guarda los tokens JWT y credenciales dentro del Keychain (iOS) o Keystore (Android).

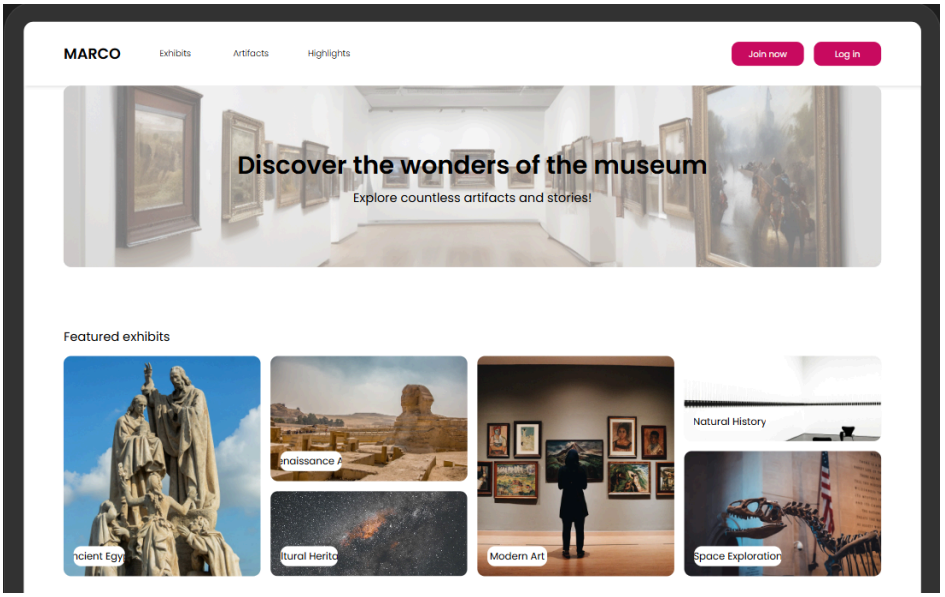
La autenticación biométrica se implementa mediante react-native-biometrics, permitiendo el uso de Face ID, Touch ID o huella digital. Esta autenticación se solicita tras 30 minutos de inactividad o para operaciones críticas, como pagos o acceso a información personal.La validación de certificados SSL se refuerza con react-native-ssl-pinning, evitando ataques Man-in-the-Middle, incluso cuando se presentan certificados aparentemente válidos pero no autorizados.Se emplea react-native-device-info para detectar si el dispositivo está rooteado (Android) o tiene jailbreak (iOS). En caso afirmativo, la aplicación muestra advertencias y limita las funciones sensibles, evitando comprometer datos o sesiones activas.

Portal WEB

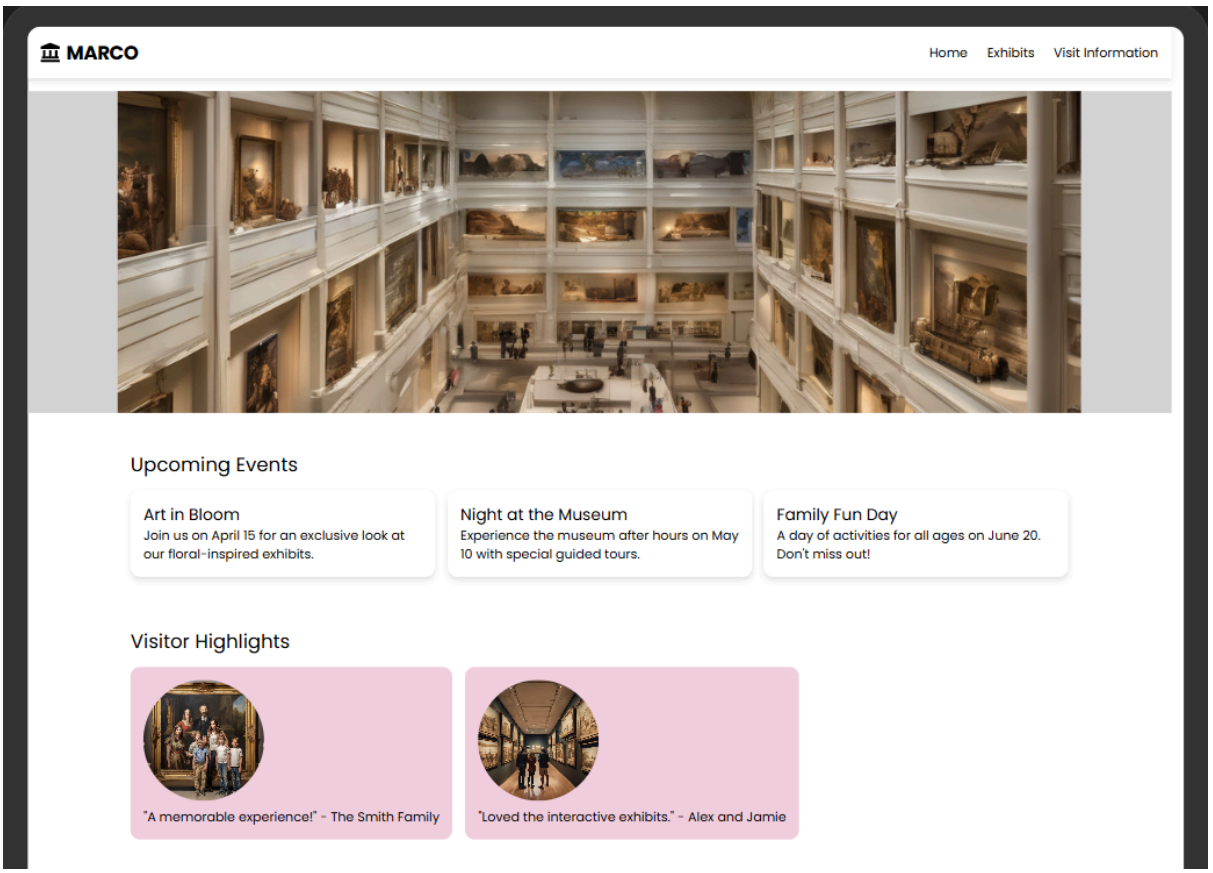
Las peticiones HTTP se manejan con axios, configurado para

- Forzar HTTPS en todas las solicitudes
- Establecer timeouts
- Implementar reintentos con backoff exponencial
- Incluir headers de autorización tipo Bearer Token

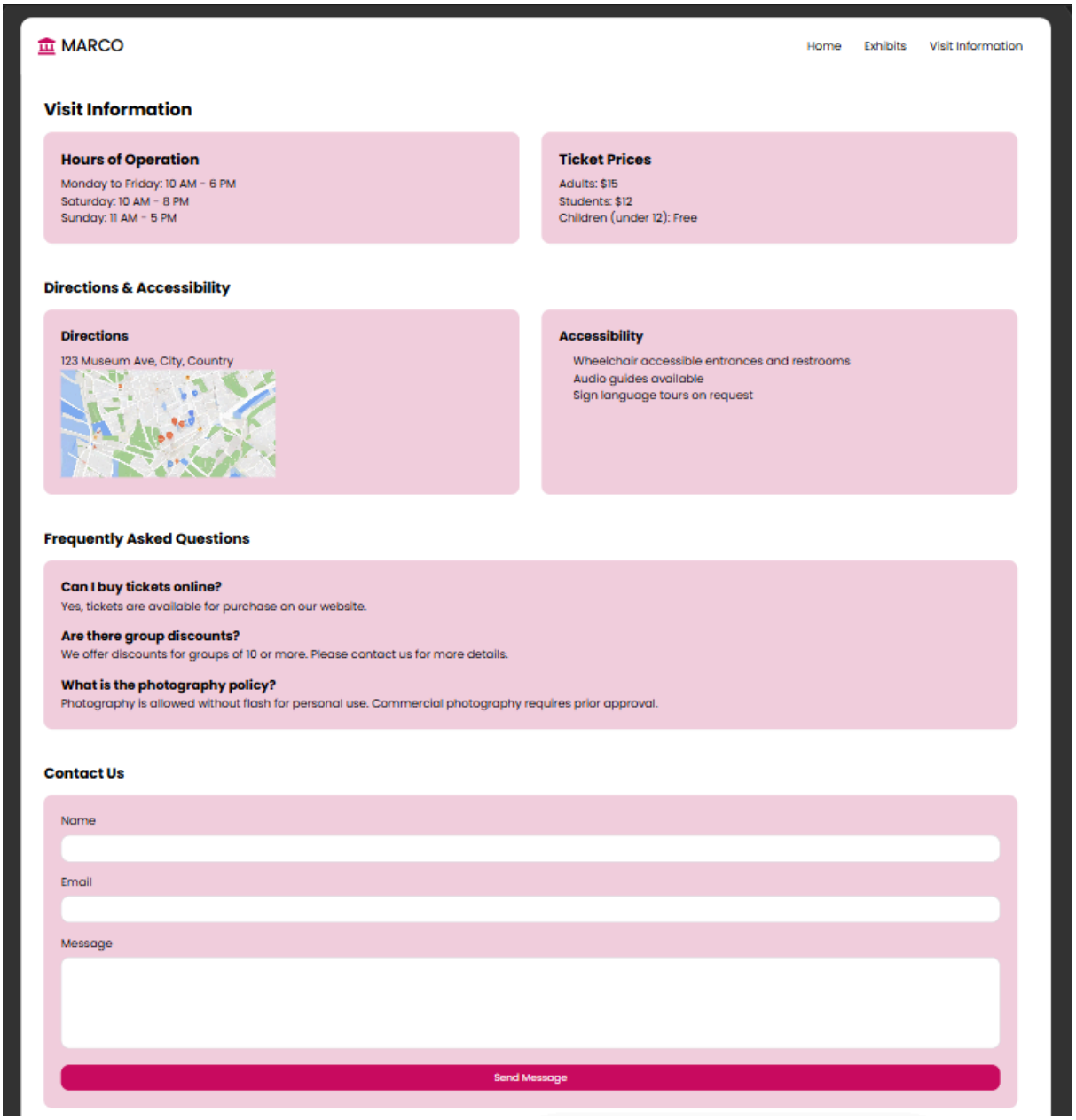
Interfaces de Usuario



1. Pagina de inicio



Pagina de tour virtual



Información adicional del museo

Prácticas de Desarrollo Seguro

El pipeline CI/CD del proyecto está implementado con GitHub Actions, ejecutándose automáticamente con cada push. En esta etapa se realiza un análisis estático con SonarCloud, que identifica vulnerabilidades y bloquea las integraciones si se detectan fallos críticos o si la cobertura de pruebas es inferior al 80%. Además, se incluye un escaneo de dependencias con npm audit y Snyk, herramientas que sugieren actualizaciones y generan pull requests automáticos cuando se detectan vulnerabilidades.

Durante la fase de desarrollo, se integra GitLeaks para la detección de secretos en el código. Esta herramienta analiza las solicitudes de contribución en busca de credenciales, claves API o tokens expuestos. También se realiza un escaneo de contenedores Docker con Trivy, el cual bloquea el despliegue si se detectan vulnerabilidades críticas o altas.

Para la protección de secretos, además de GitLeaks, se utilizan git-secrets (AWS) y una configuración estricta de .gitignore. Esto evita commits que incluyan credenciales, contraseñas o certificados.En caso de exposición, el procedimiento de remediación incluye rotación inmediata del secreto, limpieza del historial con BFG Repo-Cleaner y notificación al equipo.En producción, los secretos se almacenan en AWS Secrets Manager o HashiCorp Vault, garantizando cifrado, control de acceso mediante IAM, rotación automática, auditoría y versionado.Este pipeline busca garantizar un desarrollo seguro, automatizado y auditable para una aplicación web y móvil destinada al Museo MARCO, fortaleciendo la calidad del código, la gestión de riesgos y la protección de la información en cada etapa del ciclo de vida del software.