Documentación Técnica del Proyecto: CyberWallet

# 1. Resumen Ejecutivo

CyberWallet es una billetera virtual de próxima generación desarrollada con tecnologías modernas y arquitecturas robustas para asegurar calidad, escalabilidad y seguridad. El objetivo principal es simular un ecosistema financiero real, donde se pueden validar identidades, realizar transferencias, interactuar con divisas y más, todo siguiendo estándares como RFC 7807 para errores y JWT para autenticación.

# 2. Tecnologías Utilizadas

- Java 21, Spring Boot 3.2.5, Spring Security 6  
- Maven, JUnit 5, RestAssured, Allure  
- PostgreSQL, Flyway, H2 (test)  
- React 18 + Vite + TypeScript  
- TailwindCSS, ShadCN, Lucide Icons  
- APIs externas: restcountries.com, countrystatecity.in, dolarapi.com

# 3. Arquitectura General

El proyecto está dividido en múltiples módulos claramente separados:  
- Módulo de Autenticación: manejo de login, registro, recuperación de contraseña, logout con JWT y blacklist.  
- Módulo de Transferencias: soporte a transferencias vía CVU, alias, carga con tarjeta, validaciones múltiples.  
- Validaciones geográficas: verificación de países y provincias vía APIs oficiales.  
- Módulo de normalización: inputs sanitizados automáticamente.  
- Frontend desacoplado, consume los endpoints y presenta UI con diseño Glassmorphism/Neumorphism.

# 4. Manejo de Errores

Se implementa un esquema centralizado basado en RFC 7807 (Problem Details) con GlobalExceptionHandler.  
- ErrorCode enum: catálogo tipado de errores.  
- BusinessException: excepción controlada con metadatos.  
- fieldErrors: lista de errores de validación por campo.

# 5. Testing Avanzado

Testing exhaustivo aplicado en todos los niveles:  
- Unit tests con cobertura de servicios, validadores y utilidades.  
- Integration tests con flujos reales, incluyendo validaciones geográficas, alias, errores extremos.  
- E2E simulando transferencias concurrentes.  
- Logging estratégico y limpieza de base de datos por test.  
- Todos los errores validados vía asserts RFC 7807.

# 6. Estándares y Buenas Prácticas

- Logs a nivel DEBUG estratégicamente colocados.  
- Validación dinámica de datos con Faker y UUID.  
- Tests anotados con @Order, @Severity, @Epic, @Story.  
- Limpieza de base de datos en @BeforeEach y @AfterEach.  
- Uso correcto de ResponseEntity y DTOs en todas las capas.

# 7. Agradecimientos

Gracias a los equipos de desarrollo de las siguientes APIs públicas utilizadas para validación y simulación:  
- https://restcountries.com  
- https://api.countrystatecity.in  
- https://dolarapi.com

# 8. Estado Actual del Proyecto

El backend está completo, con cobertura robusta y pruebas exhaustivas.  
El frontend está en construcción, con Landing Page responsive terminada y diseño moderno aplicado.  
Próximos pasos: autenticación frontend, dashboard de usuario, carga visual de cotizaciones, perfil.