**Documentación Final - Prueba Técnica Superintendencia de Bancos**

**Proyecto: Plataforma Analítica Avanzada de Datos Financieros**

**Candidato:** Ean A. Jimenez  
**Organización:** Superintendencia de Bancos (SB)  
**Fecha:** Junio 2025  
**Repositorio:** <https://github.com/Prometean/GobiernoDatos_SB_POC>

**Ejercicio 1: Diseño del Equipo de Gobierno de Datos**

**Objetivo**

Constituir un equipo interdisciplinario responsable de la implementación y monitoreo de las iniciativas de calidad, ingestión, procesamiento, y analítica de datos regulatorios para la SB.

**Estructura del Equipo**

|  |  |  |
| --- | --- | --- |
| **Rol** | **Perfil Profesional** | **Responsabilidades Clave** |
| Jefe de Gobierno de Datos | Profesional senior en datos, con experiencia en regulación financiera y liderazgo | Dirigir la estrategia general, priorizar iniciativas, representar el equipo ante dirección |
| Ingeniero/a de Datos | Ingeniero/a con experiencia en ETL, Python, APIs, bases de datos y Docker | Desarrollar y mantener pipelines de ingestión y almacenamiento |
| Analista de Calidad de Datos | Especialista en validaciones, reglas de negocio y herramientas como Great Expectations | Diseñar y ejecutar validaciones automatizadas sobre reportería |
| Analista de Negocio | Profesional con conocimiento del MRI y procesos bancarios | Traducir requerimientos de negocio a validaciones técnicas y reportes |
| Desarrollador/a BI | Experto/a en visualización con Superset o Power BI | Diseñar dashboards institucionales para diferentes stakeholders |
| Data Steward | Conocimiento en catalogación y gobernanza | Administrar metadatos y asegurar calidad semántica de los datos |

**Operación del Equipo**

* **Scrum Quincenal:** Cada sprint de 2 semanas incluye planning, revisión y retrospectiva.
* **Herramientas:** Jira para gestión, GitHub para versionado, Superset para visualización.
* **Reunión diaria:** 15 minutos para seguimiento.
* **Indicadores de éxito:** % de reportes recibidos a tiempo, % de registros con errores, NPS usuarios internos.

**Recepción de Reportes**

La Superintendencia de Bancos debe recibir los reportes por parte de las entidades supervisadas según las obligaciones establecidas en el Manual de Requerimientos de Información (MRI). Esta recepción se gestiona mediante los siguientes elementos:

* **Tipología de entidades:** Cada tipo de entidad está obligado a entregar distintos reportes, con estructuras y periodicidades específicas.
* **Estandarización:** Los formatos deben cumplir con estructuras XML/CSV/JSON definidas por la SB.
* **Control de cumplimiento:** El equipo de Gobierno de Datos verifica mediante indicadores si los reportes han sido entregados correctamente y a tiempo.
* **Herramientas de soporte:** Se utilizan flujos automatizados en Airflow y sistemas de validación con Great Expectations para detectar inconsistencias.

**Ejercicio 2: Requerimiento para Servicio de Recepción Optimizada de Reportes**

**Objetivo**

Plataforma para recepción automatizada de reportes regulatorios, con capacidad de:

* Adaptarse a diferentes niveles tecnológicos de entidades.
* Almacenar reportes históricos.
* Generar alertas de cumplimiento.

**Arquitectura Propuesta**

|  |  |
| --- | --- |
| Componente | Descripción |
| API Gateway | Punto de entrada para entidades (REST + autenticación tokenizada) |
| Servicio de Recepción (Flask/Nginx) | Endpoint que valida formato, estructura y metadatos |
| Almacenamiento en MinIO | Histórico de archivos reportados (versionado, s3-like) |
| PostgreSQL o SQLite | Registro de reportes, fechas, entidad, errores |
| Scheduler (Airflow o Prefect) | Planifica y verifica cumplimiento de fechas de entrega |
| Interfaz Web (React/Bootstrap) | Panel para ver estatus de entrega y cargar manualmente |

**Features Requeridos**

* Validación automática de estructuras según MRI
* Notificación por email a entidades que incumplan
* Dashboards con status por entidad
* Logs de auditoría
* APIs REST para integración con SIMBAD u otros sistemas internos

**Justificación y Beneficios**

* Mejora trazabilidad y transparencia.
* Disminuye errores por carga manual.
* Permite escalar con nuevas entidades y nuevos reportes.

**Ejercicio 3: Flujo Institucional de Calidad de Datos**

**Objetivo**

Garantizar que los datos regulatorios cumplan con las reglas de negocio y estructuras establecidas en el MRI.

**Flujo End-to-End**

1. **Recepción del Reporte**
   * Fuente: MinIO / API
   * Registro en tabla recepcion\_reportes
2. **Validación Automática** (Great Expectations)
   * Reglas por tipo de entidad y estructura MRI
   * Resultado se guarda en resultados\_validacion
3. **Notificación de Discrepancias**
   * Email generado desde registros con errores
   * Historial en tabla notificaciones
4. **Reenvío Corregido**
   * Mismo proceso de recepción + comparación

**Tablas Clave**

CREATE TABLE recepcion\_reportes (

id SERIAL PRIMARY KEY,

entidad\_id TEXT,

tipo\_reporte TEXT,

fecha\_envio TIMESTAMP,

archivo\_url TEXT,

estado TEXT

);

CREATE TABLE resultados\_validacion (

id SERIAL PRIMARY KEY,

reporte\_id INT REFERENCES recepcion\_reportes(id),

regla TEXT,

resultado BOOLEAN,

mensaje TEXT

);

CREATE TABLE notificaciones (

id SERIAL PRIMARY KEY,

reporte\_id INT REFERENCES recepcion\_reportes(id),

fecha TIMESTAMP,

tipo TEXT,

destinatario TEXT

);

**Tecnologías Open Source Propuestas**

|  |  |  |
| --- | --- | --- |
| Fase | Herramienta | Justificación |
| Recepción | Flask + MinIO | Liviano, escalable, compatible con entornos heterogéneos |
| Validación | Great Expectations | Diseñado para validación de datos, trazabilidad, documentación automática |
| Orquestación | Apache Airflow | Definición clara de DAGs, integración con Python y notificaciones |
| Almacenamiento | PostgreSQL / SQLite | Base de datos relacional robusta y ampliamente usada |
| Visualización | Apache Superset | Dashboards interactivos, filtros nativos, soporte OLAP |
| Auditoría | OpenMetadata | Catálogo de datos y control de versiones de estructuras |

**Automatización**

El flujo completo puede ser orquestado desde Airflow:

* Tareas de ingesta
* Tareas de validación con Great Expectations
* Tareas de notificación vía email
* Carga en Superset o actualización de catálogos

**Análisis Visual en Apache Superset**

Se construyó el dashboard "**Análisis Financiero Alpha Vantage**" con 4 visualizaciones claves:

1. **Revenue Timeline**: Evolución temporal de los ingresos totales
2. **Operating Cash Flow**: Barras con flujo de caja operativo
3. **Assets vs Liabilities**: Comparación temporal entre activos y pasivos
4. **Bank Overview**: Tabla con información de cada banco

Los datos fueron obtenidos desde la API de Alpha Vantage y almacenados en SQLite. Cada tabla fue cargada en Superset, y el dashboard se construyó sobre estos datasets utilizando visualizaciones nativas.

**Conclusión**

Esta prueba de concepto muestra la factibilidad de establecer una plataforma de análisis financiero y gobernanza de datos regulatorios utilizando herramientas open-source, diseño modular y una estrategia de gobierno de datos clara y ejecutable.

El desarrollo está alineado a los principios de escalabilidad, auditabilidad y adaptabilidad que demanda la Superintendencia de Bancos.