

INTEGRANTES: Kevin Paúl Asimbaya Celi

Jhon Daniel Morales Navarrete

Francis Belén Velastegui Armas

Roberto Jhoel Narváez Sillo

GRUPO: VertexEvolution

FECHA: 01/12/2024

ESTÁNDARES

1. Introducción

Este documento establece los estándares técnicos y metodológicos que se deben seguir durante el desarrollo, implementación y mantenimiento de la aplicación. Los estándares garantizarán consistencia, calidad y facilidad de mantenimiento en todas las etapas del proyecto.

2. Estándares de Desarrollo

2.1. Organización del Código

- **Frontend:**

- Utilizar Angular para la organización del proyecto, siguiendo la estructura modular.
- Dividir componentes por funcionalidad: *registro*, *consulta* y *comparación*.
- Gestionar un directorio para estilos generales (por ejemplo, */assets/styles*) y otro dedicado a almacenar recursos visuales como imágenes.

- **Backend:**

- Configurar la aplicación utilizando Node.js y Express.
- Seguir una arquitectura RESTful para la organización de rutas.
- Utilizar controladores y servicios para separar lógica de negocio y datos.

2.2. Nombres de Variables y Archivos

- Nombres en inglés y en *camelCase* para variables y funciones.
- Archivos en *kebab-case* para mantener consistencia (*user-service.js*, *purchase-history.component.html*).
- Rutas de API en minúsculas y descriptivas (*/api/v1/purchases*, */api/v1/compare*).

2.3. Estilo de Código

- Seguir la guía de estilo de *JavaScript Standard Style* para el backend.
- Utilizar *Prettier* o *ESLint* para mantener el código limpio y uniforme.

2.4. Gestión de Dependencias

- **Frontend:** Manejar dependencias a través de *npm*.
- **Backend:** Listar todas las dependencias en *package.json* con las versiones exactas.

3. Estándares de Interacción con la API Pública (Finnhub)

3.1. Consumo de API

- Utilizar *Axios* para gestionar solicitudes HTTP.
- Definir una función reutilizable para peticiones, como *fetchStockData(symbol, token)*.
- Manejar errores de la API con bloques *try-catch*.

3.2. Frecuencia de Solicitudes

- Respetar los límites de solicitud establecidos por la API Finnhub.
- Implementar un *rate limiter* si es necesario.

3.3. Validación de Datos

- Verificar que los datos recibidos de la API tengan los campos requeridos (*price*, *symbol*, *timestamp*).
- Manejar casos de respuestas vacías o errores con mensajes claros al usuario.

4. Estándares de Validación de Datos de Usuario

- Validar entradas en el frontend y backend para evitar datos inconsistentes.
- Ejemplo de validaciones:
 - El símbolo debe ser un string alfabético y en mayúsculas (e.g., AAPL).
 - La cantidad debe ser un número entero positivo.
 - El precio debe ser un número decimal positivo.
 - La fecha debe seguir el formato YYYY-MM-DD.

5. Estándares de Diseño y Usabilidad

5.1. Diseño de Interfaz

- Seguir principios de diseño responsivo utilizando Bootstrap.
- Colores y tipografía:
 - Fondo: Blanco (#FFFFFF).
 - Texto principal: Negro (#000000).
 - Botones: Azul (#007BFF) con texto blanco (#FFFFFF).

5.2. Accesibilidad

- Proporcionar etiquetas descriptivas para formularios y botones.
- Asegurar que la navegación sea posible mediante teclado.

6. Estándares de Seguridad

6.1. Protección de Datos

- Asegurar las conexiones a la API Finnhub con HTTPS.

6.2. Manejo de Errores

- No exponer errores técnicos al usuario final.

6.3. Seguridad del Backend

- Validar todas las entradas del cliente para prevenir inyecciones de código.

7. Estándares de Control de Versiones

7.1. Flujo de Trabajo en Git

- Utilizar ramas para cada funcionalidad o corrección:
 - main: Versión estable.
 - feature/<nombre>: Nuevas funcionalidades.
 - bugfix/<nombre>: Correcciones de errores.
- Realizar *pull requests* para integrar cambios, con revisiones obligatorias por al menos un miembro del equipo.

7.2. Mensajes de Commits

- Estructura: [Tipo]: Descripción breve.

- Tipos: feat, fix, docs, style, refactor, test, chore.
- Ejemplo: feat: add user registration form.

8. Estándares de Pruebas

8.1. Tipos de Pruebas

- Pruebas unitarias para componentes del frontend y funciones del backend.
- Pruebas de integración para verificar la interacción con la API Finnhub.
- Pruebas funcionales en la interfaz de usuario.

9. Conclusión

El cumplimiento de estos estándares garantiza la calidad, la seguridad y la mantenibilidad de la aplicación web. Todos los miembros del equipo deben familiarizarse y seguir estos lineamientos durante el desarrollo del proyecto.