VALENTINA ELISA AGUILERA

Profesor: Paulina González Peña y Lillo  Fecha: 5 de diciembre de 2024

calidad de software

**Evaluación 3 de Calidad de Software**

**Introducción**

Este informe presenta los resultados obtenidos durante la evaluación parcial de la asignatura *Calidad de Software*. En el documento se detalla el desarrollo de un proyecto relacionado con personajes de la saga **Harry Potter**, donde se implementaron diversos aspectos clave de calidad de software, como Linters, cumplimiento de requerimientos funcionales y no funcionales, uso de herramientas de control de versiones, y aplicación de marcos de calidad reconocidos.



**Implementación y configuración de Linter**

En este caso utilice el video numero 1 parq guiarme en la implementación de Linter.

Implementé ESLint como herramienta de análisis estático de código, con la siguiente configuración base:

Procedo a ejecutar el código **NPM INSTALL STANDARD -D**

|  |
| --- |
|  |

En este caso mi programa me detecta una desactualización entonces busqué información para poder actualizarla con el código **NPM INSTALL ESLINT@LATEST -D**

|  |
| --- |
|  |

Esta configuración está orientada a:

La detección de errores potenciales, la estandarización de estilo de código y la mejora de la legibilidad y mantenibilidad

**Análisis de Requerimientos**

**Requerimiento Funcional:** Filtrado de Personajes

Capacidad de filtrar personajes por casa de Hogwarts.

**Línea 27 a la 31 del componente MiApi.jsx de mi código**

|  |
| --- |
|  |

Este método permite una selección dinámica y eficiente de personajes, en este caso filtra por casa seleccionada (Gryffindor, Slytherin, Ravenclaw, Hufflepuff) cumpliendo con el principio de funcionalidad específica.

Requerimiento No Funcional: Diseño Responsivo

Adaptabilidad de la interfaz a diferentes dispositivos.

**Línea 39 a la 52 del componente App.css de mi código**

|  |
| --- |
|  |

Se implementa de media queries para garantizar una experiencia de usuario óptima en múltiples dispositivos.

**OWASP - Análisis de Seguridad**

1. **Axios y solicitudes seguras**: Utilice Axios para obtener datos del servidor de manera segura y que las solicitudes usen HTTPS para proteger la información.
2. **Manejo de errores**: Si algo sale mal al hacer una solicitud (por ejemplo, el servidor no responde), se muestra un mensaje al usuario para evitar reveelar detalles técnicos.
3. **Prevención de inyecciones**: Si los usuarios deben seleccionar una opción (como una casa en un filtro), nos aseguramos de que solo puedan elegir entre opciones válidas, evitando que ingresen algún código malicioso.

**Marco de Calidad Investigado: Principios SOLID**

Aplicación de Principios

En el desarrollo del proyecto apliqué conceptos de los principios SOLID:

* **Principio de responsabilidad única:** separación en componentes como Buscador y MiApi.
* **Principio abierto/cerrado:** los componentes de React fueron diseñados para ser extensibles sin necesidad de modificar el código base.