Plan de Gestión de Calidad para la Aplicación EduShare.js

Introducción

Este plan de gestión de calidad establece los procedimientos y estándares que se implementarán en el desarrollo, operación y mantenimiento de EduShare.js, una aplicación orientada a ofrecer cursos de refuerzo educativo. La gestión de calidad garantiza que la plataforma cumpla con las expectativas de los usuarios, minimice errores y ofrezca un servicio eficiente y confiable.

1. Objetivos del Plan de Gestión de Calidad

Asegurar que la aplicación EduShare.js cumpla con los requerimientos funcionales y no funcionales.

Optimizar la experiencia del usuario mediante una interfaz intuitiva y tiempos de respuesta eficientes.

Identificar y mitigar problemas en las primeras etapas del desarrollo.

Fomentar la mejora continua a través de evaluaciones periódicas y retroalimentación.

2. Definición de Calidad

La calidad en EduShare.js se define como:

Funcionalidad completa: La plataforma debe permitir una navegación fluida y acceso a todos los cursos sin errores.

Usabilidad: Debe ser fácil de usar, accesible y contar con un diseño atractivo.

Confiabilidad: La app debe operar de manera continua, con un mínimo de interrupciones o fallos.

Eficiencia: Optimización de recursos tecnológicos para garantizar tiempos de carga rápidos.

3. Requerimientos de Calidad

Requerimientos funcionales:

Registro de usuarios, gestión de cursos y sistema de evaluación del progreso.

Notificaciones automáticas de tareas o eventos relevantes.

Requerimientos no funcionales:

Tiempo de respuesta: No superior a 3 segundos por acción.

Disponibilidad: 99% del tiempo durante horarios pico.

Accesibilidad: Cumplimiento con estándares WCAG 2.1 para garantizar el acceso a personas con discapacidades.

4. Roles y Responsabilidades

|  |  |
| --- | --- |
| Rol | Responsabilidad |
| Gerente de Calidad | Supervisar la implementación de estándares de calidad y realizar auditorías internas. |
| Equipo de Desarrollo | Implementar funcionalidades siguiendo las mejores prácticas y realizar pruebas unitarias. |
| Equipo de Pruebas (QA) | Diseñar y ejecutar casos de prueba y reportar errores. |
| Soporte Técnico | Monitorear el rendimiento de la aplicación y resolver problemas reportados. |
| Usuarios finales | Proporcionar retroalimentación sobre la experiencia de uso. |

5. Procesos de Aseguramiento de Calidad (QA)

Revisión de Requisitos:

Verificar que los requerimientos del sistema estén claros y alineados con los objetivos.

Pruebas Unitarias:

Cada módulo desarrollado será probado de manera aislada por el equipo de desarrollo.

Pruebas de Integración:

Verificar la interacción entre los diferentes componentes de la aplicación.

Pruebas de Carga y Estrés:

Evaluar el rendimiento del sistema bajo condiciones extremas de uso.

Pruebas de Aceptación del Usuario (UAT):

Validar la funcionalidad con un grupo de usuarios finales antes del lanzamiento oficial.

Auditorías Internas:

Realizar auditorías trimestrales para evaluar el cumplimiento de estándares de calidad.

6. Herramientas de Gestión de Calidad

Jira: Para la gestión de tareas y seguimiento de errores.

Postman: Para pruebas de API.

Selenium: Para pruebas automatizadas en la interfaz.

Google Analytics: Para monitorear el comportamiento de los usuarios en la app.

7. Indicadores Clave de Desempeño (KPIs)

|  |  |  |
| --- | --- | --- |
| Indicador | Objetivo | Frecuencia de Medición |
| Tasa de errores reportados | Menos de 1 error por cada 100 usuarios activos. | Mensual |
| Tiempo de respuesta | Menos de 3 segundos por acción. | Semanal |
| Nivel de satisfacción del usuario | 85% o más en encuestas. | Trimestral |
| Disponibilidad del sistema | 99% de disponibilidad. | Mensual |

8. Gestión de Mejora Continua

Recopilación de Retroalimentación: A través de encuestas a usuarios y análisis de los comentarios recibidos.

Revisión de Rendimiento: Evaluación trimestral del desempeño basado en los KPIs.

Implementación de Ajustes: Incorporación de mejoras a partir de las lecciones aprendidas.

Actualización de Estándares: Ajuste de políticas y procesos según cambios tecnológicos o de mercado.

9. Plan de Contingencia

Falla en el sistema: Activación de servidores de respaldo para garantizar la continuidad del servicio.

Error crítico en la aplicación: Notificación inmediata a los usuarios y despliegue de un parche de emergencia.

Insatisfacción del usuario: Implementación de encuestas rápidas para identificar y resolver los problemas reportados.

10. Conclusión

Este plan de gestión de calidad proporciona una hoja de ruta clara para garantizar que EduShare.js cumpla con altos estándares de calidad, asegurando una experiencia satisfactoria para los usuarios. A través de procesos rigurosos de pruebas, monitoreo continuo y mejora constante, la plataforma podrá mantener su eficiencia, confiabilidad y accesibilidad. La aplicación de estas prácticas también permitirá anticipar problemas y fomentar una cultura de excelencia en el equipo de desarrollo.