**Políticas de Mantenimiento**

**Aplicación Web Progresiva – Registro de Asistencia**

**I. Introducción**

Este documento establece las políticas de mantenimiento aplicables a la aplicación web progresiva (PWA) de Registro de Asistencia. La herramienta permite a los usuarios llevar el control de su asistencia personal de forma offline y online, adaptándose a entornos sin conectividad estable.

**II. Objetivo**

Garantizar el correcto funcionamiento, confiabilidad, seguridad y evolución del sistema a través de mantenimiento preventivo, correctivo y continuo, así como asegurar una buena experiencia para los usuarios.

**III. Marco de Referencia**

Las políticas de mantenimiento se guían por principios de buenas prácticas en desarrollo web, la protección de datos personales, el diseño centrado en el usuario y la tolerancia a fallos. Aunque no aplica aún legislación directa, se promoverán prácticas responsables de almacenamiento y recuperación de información.

**IV. Glosario de Términos**

* **PWA:** Aplicación Web Progresiva. Funciona offline gracias a un service worker.
* **Mantenimiento preventivo:** Acciones regulares para evitar errores.
* **Mantenimiento correctivo:** Corrección de fallos detectados.
* **Mantenimiento continuo:** Mejoras planeadas y soporte.
* **Backup:** Copia de seguridad de datos.
* **Usuario:** Persona que utiliza la app para registrar su propia asistencia.
* **Redundancia:** Copias sincronizadas de datos para evitar pérdidas.
* **Versión offline:** Funcionalidad de la app sin conexión a Internet.

**V. Políticas Generales**

1. **Responsabilidad del desarrollador:** El equipo creador será responsable de supervisar el mantenimiento, actualizaciones y correcciones.
2. **Colaboración del usuario:** Se espera que los usuarios reporten fallos a través de formularios o medios definidos.
3. **Confidencialidad de datos:** Los datos personales no se comparten ni se transmiten sin consentimiento.
4. **Arquitectura modular:** El sistema está dividido en módulos para facilitar el mantenimiento (PWA, módulos JS, datos en localStorage y PHP + MySQL).

**VI. Políticas de Mantenimiento Preventivo**

* **Revisiones periódicas:** Revisión funcional cada mes para comprobar correcto funcionamiento.
* **Backups:** Se realizarán copias de seguridad locales semanales y automáticas.
* **Pruebas funcionales:** Se aplicarán pruebas básicas de interfaz y lógica cada mes.
* **Notificaciones:** El sistema incluirá avisos visibles de mantenimiento futuro.
* **Control offline:** La funcionalidad offline se validará al menos una vez por ciclo de mantenimiento.

**VII. Políticas de Mantenimiento Correctivo**

* **Tiempo de respuesta:** Se buscará resolver errores en un plazo no mayor a 24 horas en días hábiles.
* **Clasificación de fallos:** Los errores se priorizarán según su impacto (críticos, menores, visuales).
* **Medios de solución:** Dependiendo del error, se aplicarán parches directos al JS.
* **Registro de errores:** Los errores serán documentados en una bitácora y se notificará al usuario si procede.
* **Pruebas del usuario:** Se requerirá al usuario describir el fallo (captura o mensaje visible).

**VIII. Mantenimiento Continuo (Soporte Evolutivo)**

* **Actualizaciones planificadas:** Se actualizará el sistema al menos una vez por trimestre.
* **Mejoras previstas:** Se considera incluir login con PHP y base de datos MySQL, sincronización de datos remotos, y gráficos de resumen.
* **Extensibilidad:** El sistema está diseñado modularmente para permitir nuevas funciones como notificaciones, calendario o multiusuario.
* **Compatibilidad:** Se probarán versiones actualizadas en los navegadores más comunes.
* **Retroalimentación del usuario:** Se evaluará la experiencia con encuestas simples o formularios web.

**IX. Identificación de Riesgos y Estrategias de Mitigación**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Factor de Riesgo | Probabilidad | Impacto | Estrategia de Mitigación | Responsable |
| Pérdida de conexión | Alta | Media | Uso de Service Worker y localStorage | Desarrollador |
| Eliminación accidental de datos locales | Media | Alta | Incluir advertencias antes de borrar y opción futura de respaldo automático | Desarrollador |
| Fallos en actualizaciones futuras | Media | Media | Probar cambios en entorno de prueba antes de subirlos a producción | Desarrollador |
| Resistencia al cambio del usuario | Baja | Baja | Mantener la interfaz simple, mínima curva de aprendizaje | Desarrollador |
| Integración futura con base de datos | Alta | Alta | Planificar fase de transición con pruebas y sincronización progresiva | Desarrollador / Backend |