**Plan de Pruebas de Software**

***PROYECTO R.A.G.S***

***TPS - 2926378***

***Fecha: [01/12/2024]***

**Tabla de contenido**

Historial de Versiones 4

Información del Proyecto 4

Aprobaciones 4

Resumen Ejecutivo 5

Alcance de las Pruebas 5

Elementos de Pruebas 5

Nuevas Funcionalidades a Probar 6

Pruebas de Regresión 6

Funcionalidades a No Probar 7

Enfoque de Pruebas (Estrategia) 7

Criterios de Aceptación o Rechazo 8

Criterios de Aceptación o Rechazo 8

Criterios de Suspensión 8

Criterios de Reanudación 9

Entregables 9

Recursos 10

Requerimientos de Entornos – Hardware 10

Requerimientos de Entornos – Software 10

Herramientas de Pruebas Requeridas 11

Personal 11

Entrenamiento 12

Planificación y Organización 12

Procedimientos para las Pruebas 12

Matriz de Responsabilidades 13

Cronograma 13

Premisas 14

Dependencias y Riesgos 14

Referencias 15

Glosario 15

**Historial de Versiones**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fecha** | **Versión** | **Autor** | **Organización** | **Descripción** |
| 01/12/24 | 1.0 | Juan Pablo Mosquera | R.A.G.S | En la presente versión el proyecto va orientado hacia un avance con respecto a la idea principal otorgada al mismo. |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Información del Proyecto**

|  |  |
| --- | --- |
| Empresa / Organización | SENA |
| Proyecto | R.A.G.S |
| Fecha de preparación | 01/12/2024 |
| Cliente | SENA |
| Patrocinador principal | ------ |
| Gerente / Líder de Proyecto | Juan Pablo Mosquera Ubaté |
| Gerente / Líder de Pruebas de Software | Juan Pablo Mosquera Ubaté |

**Aprobaciones**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nombre y Apellido** | **Cargo** | **Departamento u Organización** | **Fecha** | **Firma** |
| Albeiro Ramos | Instructor | Sena | Fecha revisión |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Resumen Ejecutivo**

El Plan Maestro de Pruebas de Software tiene como objetivo garantizar que la aplicación para registrar el ingreso y salida de los visitantes cumpla con los requisitos funcionales, de usabilidad, rendimiento y seguridad. El plan abarca las pruebas necesarias a lo largo del ciclo de desarrollo, con un enfoque en las funcionalidades clave y la eficiencia del sistema. Las restricciones incluyen un equipo limitado, presupuesto ajustado y tiempo corto, por lo que se priorizarán las pruebas más críticas. El esfuerzo de pruebas se centrará en validar el funcionamiento adecuado del sistema en distintos entornos y su seguridad.

**Alcance de las Pruebas**

**Elementos de Pruebas**

**Listado de Módulos, Componentes o Elementos a Probar**

Para este proyecto, el listado de módulos y componentes a probar se puede estructurar en dos niveles: alto y detallado.

**Nivel Alto (Áreas Funcionales):**

1. **Interfaz de Usuario (UI)**: Validación de la facilidad de uso y navegación del sistema por parte de los guardias de seguridad para registrar el ingreso y salida de visitantes.
2. **Gestión de Registros**: Validación de la correcta captura, almacenamiento y consulta de los datos de los visitantes, como nombre, hora de ingreso, hora de salida, motivo de la visita, etc.
3. **Control de Accesos**: Pruebas sobre la autenticación y autorización de los usuarios (guardias de seguridad) para asegurarse de que solo personas autorizadas puedan acceder y modificar los registros.
4. **Generación de Reportes**: Validación de las funcionalidades para generar y visualizar informes de los registros de visitantes.
5. **Rendimiento**: Validación de la capacidad del sistema para manejar múltiples registros de visitantes simultáneamente, asegurando tiempos de respuesta adecuados.
6. **Seguridad**: Verificación de las medidas de protección de los datos sensibles de los visitantes, incluyendo pruebas contra vulnerabilidades comunes.
7. **Módulo de Login/Autenticación**: Pruebas de inicio de sesión, validación de credenciales y control de acceso.
8. **Módulo de Registro de Visitantes**: Validación de los campos de entrada de datos (nombre, documento, hora de ingreso/salida, etc.) y su correcta integración con la base de datos.
9. **Módulo de Base de Datos**: Verificación de la correcta creación, almacenamiento y consulta de los registros en la base de datos.
10. **Módulo de Reportes**: Pruebas en la generación de informes de registros de visitantes, tanto en formato visual como descargable (por ejemplo, PDF o Excel).
11. **Módulo de Seguridad**: Validación de la encriptación de datos y control de accesos para garantizar la protección de la información sensible.
12. **Módulo de Integración**: Verificación de la correcta interacción entre los diferentes módulos del sistema y su funcionamiento adecuado en conjunto.

Este enfoque cubre tanto las áreas funcionales clave como los módulos específicos que requieren pruebas detalladas para asegurar el correcto funcionamiento del sistema en su totalidad.

**Nuevas Funcionalidades a Probar**

**1. Registro de Visitantes**

* **Funcionalidad**: Los guardias de seguridad pueden registrar fácilmente los datos de los visitantes, como nombre, documento de identificación, fecha y hora de ingreso, y motivo de la visita.
* **Punto de Vista del Usuario**: El usuario (guarda de seguridad) puede llenar un formulario de manera rápida y eficiente, asegurándose de que toda la información relevante se capture correctamente sin confusión.

**2. Edición de Registros**

* **Funcionalidad**: Los guardias pueden editar información de los visitantes registrados si es necesario, por ejemplo, corregir errores o añadir información adicional.
* **Punto de Vista del Usuario**: El usuario debe tener una interfaz intuitiva donde, al seleccionar un registro, pueda modificar los datos de forma sencilla.

**3. Búsqueda Rápida de Visitantes**

* **Funcionalidad**: El sistema permite buscar registros previos de visitantes por nombre, documento de identificación o fecha.
* **Punto de Vista del Usuario**: El guardia puede encontrar rápidamente a un visitante en la base de datos mediante una barra de búsqueda o filtros, ahorrando tiempo y evitando búsquedas manuales.

**4. Generación Automática de Minutas**

* **Funcionalidad**: El sistema genera un resumen de la entrada y salida de los visitantes en formato de minuta que puede ser exportado o impreso.
* **Punto de Vista del Usuario**: El usuario puede generar un informe completo de las visitas registradas en un periodo de tiempo, el cual se puede imprimir o almacenar digitalmente.

**5. Notificación de Entrada/Salida**

* **Funcionalidad**: El sistema notifica a los usuarios (guardias de seguridad) cuando un visitante ha ingresado o salido, proporcionando alertas visuales o sonoras.
* **Punto de Vista del Usuario**: El guardia recibe una alerta instantánea cada vez que se registra un nuevo visitante o se marca la salida de un visitante, lo que mejora el monitoreo.

**6. Accesibilidad y Facilidad de Uso**

* **Funcionalidad**: La interfaz del sistema es fácil de navegar y accesible para los usuarios sin conocimientos técnicos.
* **Punto de Vista del Usuario**: Los guardias deben sentir que el sistema es intuitivo y rápido, sin necesidad de formación avanzada para usarlo efectivamente.

**7. Integración con Otros Sistemas**

* **Funcionalidad**: El sistema puede integrarse con otras plataformas, como el control de acceso o el sistema de cámaras de seguridad, para obtener información adicional.
* **Punto de Vista del Usuario**: El guardia puede visualizar información de acceso en tiempo real sin tener que cambiar de sistema, lo que optimiza el flujo de trabajo.

**8. Historial de Visitas**

* **Funcionalidad**: El sistema permite acceder al historial completo de visitas de un visitante específico, con detalles de cada entrada.
* **Punto de Vista del Usuario**: El guardia puede acceder al historial de visitas de un visitante en cualquier momento, lo que facilita la verificación de la actividad y mejora la seguridad.

**9. Alertas de Seguridad**

* **Funcionalidad**: El sistema emite alertas cuando se detecta un visitante con un perfil sospechoso (por ejemplo, en caso de que la identificación esté marcada como bloqueada).
* **Punto de Vista del Usuario**: El guardia recibe una advertencia visible que le permite verificar la situación rápidamente y tomar acciones de seguridad si es necesario.

**10. Acceso a Reportes Detallados**

* **Funcionalidad**: Los usuarios pueden generar reportes detallados sobre los registros de visitantes durante un determinado periodo de tiempo, incluyendo gráficos, análisis de flujo de visitantes, etc.
* **Punto de Vista del Usuario**: El guardia puede acceder a reportes listos para imprimir o enviar por correo, lo que facilita la revisión y análisis del flujo de personas en el lugar.

Estas son solo algunas ideas generales. Si tienes funcionalidades específicas en mente o más detalles sobre el sistema, puedo ayudarte a definirlas aún más.

**Pruebas de Regresión**

**1. Sistema de Control de Acceso**

* **Funcionalidad**: El sistema de control de acceso puede estar vinculado a la información de los visitantes, y aunque no sea parte del desarrollo principal, las interacciones con las minutas pueden verse afectadas.
* **Punto de Vista del Usuario**: El guardia debe asegurarse de que el sistema de control de acceso siga funcionando correctamente, permitiendo la entrada de los visitantes según la información registrada.

**2. Interfaz de Usuario (UI) de la Aplicación**

* **Funcionalidad**: Si el diseño o la estructura de la interfaz de usuario ha sido modificada por cambios en el sistema de minutas, aunque no sea la parte principal del desarrollo, debe verificarse que la interfaz sigue siendo accesible y fácil de usar.
* **Punto de Vista del Usuario**: El guardia debe comprobar que las funcionalidades de la interfaz no solo siguen estando disponibles, sino que también son fáciles de acceder, y que no hay elementos que se hayan desplazado o mal implementado.

**3. Sistema de Backup de Datos**

* **Funcionalidad**: La creación de copias de seguridad o el sistema de recuperación de datos podría no ser parte de los cambios, pero sigue siendo afectado por la gestión de registros.
* **Punto de Vista del Usuario**: El guardia debe asegurarse de que las copias de seguridad continúan realizándose correctamente y que la restauración de datos funciona si es necesario.

**4. Autenticación de Usuarios**

* **Funcionalidad**: Si el sistema de minutas está vinculado a una plataforma con control de acceso de usuarios, la autenticación y permisos de los mismos podrían verse indirectamente afectados.
* **Punto de Vista del Usuario**: El usuario debe asegurarse de que el sistema de autenticación (como inicio de sesión y permisos de usuario) sigue funcionando sin inconvenientes, garantizando que los accesos sean los adecuados según el perfil.

**5. Sistema de Impresión de Minutas o Informes**

* **Funcionalidad**: El proceso de impresión de las minutas o informes podría no ser parte del desarrollo actual, pero cualquier cambio en el sistema de minutas podría afectar la capacidad de imprimir los documentos correctamente.
* **Punto de Vista del Usuario**: El guardia debe verificar que la impresión de los documentos generados (como las minutas) se siga realizando de manera correcta y sin errores de formato o contenido.

**Funcionalidades a No Probar**

1. **Integración de la base de datos con sistemas externos**  
   **Razón**: El proyecto no incluye integración con sistemas externos (por ejemplo, sistemas de gestión de acceso o bases de datos ajenas). Solo se está probando la funcionalidad interna de la aplicación, como el ingreso de datos por parte de los guardas y su almacenamiento en la base de datos local.  
   **Riesgo**: Al no probar la integración con sistemas externos, se corre el riesgo de que, si en el futuro se requiere integrarse con otros sistemas, puedan surgir problemas de compatibilidad o de manejo de datos.
2. **Pruebas de escalabilidad y carga**  
   **Razón**: El enfoque del proyecto en esta fase es principalmente funcional, centrado en las operaciones básicas del sistema (login, registro, ingreso de minutas, etc.), sin realizar pruebas de alto tráfico o múltiples usuarios simultáneos.  
   **Riesgo**: No realizar pruebas de carga podría generar problemas en la performance de la aplicación cuando se use con muchos usuarios al mismo tiempo, como caídas del servidor o lentitud en el acceso a datos.
3. **Pruebas de seguridad avanzadas (inyección SQL, XSS, CSRF, etc.)**  
   **Razón**: Aunque la aplicación utiliza PHP y puede tener vulnerabilidades inherentes si no se valida adecuadamente, en esta fase no se realizarán pruebas avanzadas de seguridad. El enfoque inicial está en comprobar la funcionalidad básica.  
   **Riesgo**: Existen riesgos de seguridad si no se validan y protegen correctamente las entradas de usuario, lo que podría permitir ataques como inyecciones de SQL, Cross-Site Scripting (XSS) o Cross-Site Request Forgery (CSRF).
4. **Pruebas en dispositivos móviles**  
   **Razón**: Aunque la aplicación está desarrollada en HTML, CSS y JS, y podría ser accesible en dispositivos móviles, no se realizarán pruebas específicas de usabilidad o diseño en diferentes dispositivos móviles en esta fase. El diseño está orientado a su funcionamiento en escritorio.  
   **Riesgo**: No probar la interfaz en dispositivos móviles podría generar una mala experiencia de usuario en teléfonos o tabletas, especialmente si el diseño no es completamente adaptable o responsive.
5. **Sistema de notificaciones (por ejemplo, alertas de ingreso o salida)**  
   **Razón**: El sistema de notificaciones para alertar a los guardas de seguridad sobre los ingresos y salidas de visitantes no está implementado en esta fase del proyecto.  
   **Riesgo**: Sin un sistema de notificaciones, los guardas podrían no estar al tanto de cambios en tiempo real, lo que podría generar confusión o retrasos en el proceso de registro de visitantes.
6. **Pruebas de recuperación ante desastres o copias de seguridad automáticas**  
   **Razón**: El proyecto no incluye funcionalidades avanzadas de recuperación ante desastres o copias de seguridad automáticas de la base de datos en esta etapa.  
   **Riesgo**: La falta de estas medidas podría resultar en la pérdida de datos importantes en caso de fallo del sistema, ya sea por error humano o problemas técnicos.
7. **Soporte para múltiples idiomas**  
   **Razón**: El proyecto está desarrollado únicamente en un idioma (español) en esta fase, sin planes inmediatos de implementación de soporte multilingüe.  
   **Riesgo**: Los usuarios que no hablen el idioma principal de la aplicación no podrán utilizarla correctamente, limitando la accesibilidad a un público más amplio.

**Enfoque de Pruebas (Estrategia)**

**1. Objetivo de las Pruebas**

El objetivo principal de las pruebas es verificar que el sistema de gestión de minutas de seguridad funcione correctamente en todas sus funcionalidades básicas (login, registro, gestión de minutas, y ver usuarios), garantizando la integridad y fiabilidad de los datos. Esto incluye comprobar que el sistema responde adecuadamente a los requerimientos funcionales y no funcionales, y que se mantenga seguro y escalable.

**Criterios de Aceptación o Rechazo**

**Criterios de Aceptación o Rechazo**

El **Plan de Pruebas de Software** será considerado completado con éxito si cumple con los criterios de aceptación establecidos, incluyendo una alta tasa de éxito en pruebas funcionales, seguridad, rendimiento y usabilidad, así como la corrección de defectos críticos. Si alguno de los criterios de rechazo se presenta, las pruebas no se considerarán satisfactorias hasta que se resuelvan los problemas identificados.

**Criterios de Suspensión**

* **Defectos críticos no resueltos** (errores que bloquean funcionalidades clave o problemas graves de seguridad).
* **Tasa de fallos excesiva** (por ejemplo, más del 20% de casos fallidos).
* **Problemas de rendimiento graves** (tiempos de respuesta inaceptables o caídas del sistema bajo carga).
* **Problemas de usabilidad importantes** que impiden una interacción efectiva con el sistema.
* **Falta de recursos** para continuar con las pruebas de manera efectiva

**Entregables**

* **Diagrama de Despliegue**: Representación de la arquitectura del sistema en producción, mostrando los componentes y su interacción.
* **Diagrama de Clases**: Descripción de las clases del sistema, sus atributos, métodos y relaciones.
* **Plan Maestro de Pruebas**: Documento con los objetivos, estrategias, recursos, plazos y criterios de aceptación para las pruebas.
* **Plan de Casos de Prueba**: Detalle de los casos de prueba a ejecutar, especificando pasos, datos de entrada y resultados esperados.
* **Lista de Chequeo**: Elementos a verificar antes, durante y después de las pruebas.
* **Documentación de Pruebas**: Registros de ejecución de pruebas, logs y resultados.
* **Manual Técnico**: Guía para desarrolladores y administradores sobre la configuración, despliegue y mantenimiento del sistema.
* **Manual de Usuario**: Instrucciones para los usuarios finales sobre cómo usar el sistema y resolver problemas comunes.

**Recursos**

**Requerimientos de Entornos – Hardware**

**1. Servidores de Aplicación**

* **CPU**: 2-4 núcleos (mínimo 2.0 GHz)
* **RAM**: 4GB
* **Almacenamiento**: 1 GB libres
* **Sistema Operativo**: Windows 10 ó superior / Linux / MacOS
* **Red**: Conexión mínima de 100 Mbps

**2. Servidor de Bases de Datos**

* **CPU**: 2-4 núcleos
* **RAM**: 4-8 GB
* **Almacenamiento**: 30 GB libres
* **Sistema Operativo**: Linux (Ubuntu, CentOS) o Windows Server
* **Red**: Conexión mínima de 100 Mbps

**3. Conectividad y Red**

* **Red Local (LAN)**: Conexión de 1 Gbps
* **Acceso a Internet**: Conexión estable y de alta velocidad para pruebas de integración

**Requerimientos de Entornos – Software**

* **Sistema Operativo**:
* **Windows 10** o **macOS** para testers.
* **Linux (Ubuntu o CentOS)** para servidores de aplicación y bases de datos.
* **Navegadores Web**:
* **Google Chrome**, **Mozilla Firefox**, **Microsoft Edge**.
* **Servidor Web**:
* **Apache** o **Nginx** para el entorno de pruebas.
* **Base de Datos**:
* **MySQL** o **MariaDB** para la gestión de datos.
* **Entorno de Desarrollo y Pruebas**:
* **PHP 7.x o superior**.
* **XAMPP** o **WAMP** para servidores locales.
* **Herramientas de Pruebas**:
* **Selenium** para pruebas automatizadas de UI.
* **PHPUnit** para pruebas unitarias y de integración.
* **Postman** para pruebas de APIs.
* **Herramientas de Control de Versiones**:
* **Git** o plataformas como **GitHub** o **GitLab**.
* **Herramientas de Reportes y Documentación**:
* **Jira** o **Trello** para gestión de incidencias.
* **Microsoft Office** o **Google Workspace** para informes.

**Planificación y Organización**

**PROCESO GENERAL:**

* **Planificación**: Definir casos de prueba, cronograma y recursos necesarios.
* **Ejecución**: Ejecutar pruebas, registrar resultados y detectar defectos.
* **Evaluación**: Analizar los resultados de las pruebas para identificar problemas o desviaciones.
* **Informe**: Generar reportes de incidencias y resultados, proponer soluciones y acciones correctivas.

**Premisas**

**1. Limitaciones de Tiempo:**

* **Plazo limitado** para completar todas las pruebas debido a la fecha de entrega del proyecto.
* **Tiempo insuficiente** para realizar pruebas exhaustivas de todas las funcionalidades del sistema, lo que llevó a priorizar ciertos casos de prueba.
* Las pruebas de **regresión** no se pudieron ejecutar en su totalidad debido a la presión del tiempo y a la introducción de nuevas funcionalidades de última hora.
* **Disolución del grupo de proyecto:** El modo en que tuvo que concluir el proyecto y se llevara a cabo la separación de los integrantes del proyecto generó retrasos considerables en el desarrollo y preparación del mismo.

 **Dependencias y Riesgos**

1. **Dependencias con Desarrollos**

* **Riesgo**: Las pruebas dependen de la finalización de las funcionalidades.
* **Mitigación**: Coordinar entregas y validaciones paralelas.

2. **Dependencias con Otros Proyectos**

* **Riesgo**: Otros proyectos afectan el entorno de pruebas.
* **Mitigación**: Mantener comunicación y coordinar interdependencias.

.3 **Disponibilidad de Recursos**

* **Riesgo**: Falta de personal o herramientas.
* **Mitigación**: Asegurar recursos desde el inicio y planificar capacitación.

4. **Restricciones de Tiempo**

* **Riesgo**: Plazos ajustados afectan la cobertura de pruebas.
* **Mitigación**: Priorizar pruebas críticas y automatizar tareas repetitivas.

5. **Premisas Erróneas**

* **Riesgo**: Asumir condiciones incorrectas.
* **Mitigación**: Validar suposiciones con las partes interesadas.

**Referencias**

**Plan de Proyecto**: Define el alcance, objetivos y recursos, proporcionando contexto para las pruebas.

**Especificaciones de Requerimientos**: Detallan lo que el sistema debe hacer, base para crear casos de prueba.

**Diseño General**: Describe la estructura del sistema y la interacción entre sus componentes.

**Manual de Usuario**: Instrucciones para el usuario final, útil para pruebas de usabilidad y aceptación.

**Informe de Riesgos**: Describe riesgos potenciales, ayudando a priorizar pruebas críticas.

**Registro de Incidencias**: Historial de errores encontrados en las pruebas para hacer seguimiento y correcciones.

**Glosario**

* **Pruebas de Software**: Actividades para verificar que un sistema cumple con los requisitos y funciona correctamente.
* **Pruebas Funcionales**: Validan que las funcionalidades del sistema operan como se espera.
* **Pruebas de Regresión**: Aseguran que los cambios no afecten funcionalidades existentes.
* **Caso de Prueba**: Condiciones y resultados esperados para verificar una funcionalidad.
* **Plan de Pruebas**: Documento que define estrategias, recursos y cronograma de pruebas.
* **Automatización de Pruebas**: Uso de herramientas para ejecutar pruebas automáticamente.
* **Incidencia/Defecto**: Error identificado durante las pruebas que impide el correcto funcionamiento.
* **Minuta**: Registro de entrada y salida de visitantes en instalaciones, utilizado por guardias de seguridad.
* **Pruebas de Usabilidad**: Evaluación de la facilidad de uso del sistema.
* **Pruebas de Seguridad**: Identificación de vulnerabilidades en el sistema.
* **Pruebas de Rendimiento**: Evaluación del sistema bajo condiciones de carga.
* **Pruebas de Integración**: Verificación de la interacción entre componentes del sistema.
* **Pruebas Unitarias**: Validación de módulos individuales del sistema.
* **Cobertura de Pruebas**: Porcentaje de código o funcionalidades probadas.
* **Verificación**: Comprobación de que el software cumple con los requisitos técnicos.
* **Validación**: Confirmación de que el software cumple con las expectativas del cliente.
* **Reporte**: Documento que detalla los resultados y hallazgos de las pruebas realizadas.