Plan de Verificación y Validación

Versión 1.7

Historia de revisiones

Fecha	Versión	Descripción	Autor
22/01/2021	1.0	Primera versión	Francis Mori
22/01/2021	1.1	Agregados requerimientos por verificar	Diego León
23/01/2021	1.2	Agregados estrategias de verificación	Javier Portella
23/01/2021	1.3	Agregados recursos e hitos	León Chávez
24/01/2021	1.4	Agregados entregables	Francis Mori
24/01/2021	1.5	Agregados dependencias	Diego León
24/01/2021	1.6	Agregados riesgos	Javier Portella
25/01/2021	1.7	Agregar detalles y corregir	León Chávez

Contenido

1. Introducción

1.1. Propósito

El presente plan para el proyecto de SGEH, tiene los siguientes objetivos:

- Identificar los componentes de software y documentación que deben ser sometidos al proceso de verificación y validación.
- Enumerar los requerimientos que son recomendados para verificar, teniendo en cuenta las prioridades del cliente en cada fase.
- Describir las estrategias de verificación que serán utilizadas para cada tipo de verificación, esto es, verificación unitaria, de integración, funcional y de sistema.
- Identificar los recursos humanos y roles que serán necesarios en el proceso de verificación y validación.

1.2. Punto de partida

El sistema por construir es un sistema de gestión de eventos cuyo servicio correrá en las instalaciones de Cloud Computing de Microsoft Windows Azure. La principal funcionalidad del software es la publicación y reserva de eventos.

Estará conformado por 3 grandes componentes:

- Servidor ASP.Net
- Cliente Web
- Cliente Móvil

El objetivo de la verificación es poder encontrar la mayor cantidad de defectos tempranamente con el objetivo de entregar un software de calidad al cliente, y evitando las cuestiones de retrabajo ocasionadas por el descubrimiento de fallas sobre el final del proceso de desarrollo.

Para lograr esto, se requiere tener tanto una visión global del sistema, como visión en cada uno de los subsistemas que lo componen, y también de cada unidad de código fuente que componen los subsistemas. Es por esto por lo que la verificación se separa en 5 grandes tipos:

- Pruebas unitarias: Se verifica cada unidad de código fuente, para determinar si contiene defectos en su implementación, comprobándolo contra el diseño de este.
- Pruebas de integración: Se verifica el correcto funcionamiento de las interacciones entre las unidades de código fuente para determinar que no existen defectos en la comunicación de estos.
- Pruebas funcionales: Se verifica que el sistema cumple con todas las funcionalidades especificadas en los requerimientos de este.
- Pruebas del sistema: Se verifican conjuntos de funcionalidades del sistema, como ser ciclo de vida de entidades, interacciones entre las mismas, etc. De manera de obtener la aceptación por parte del cliente.
- Pruebas de requerimientos no funcionales: Se verifican que los requerimientos no funcionales especificados son cumplidos por el sistema

1.3. Alcance

A continuación, se pretende definir el alcance del proceso de verificación y validación, especificando las responsabilidades que deben asumir los roles involucrados.

La verificación unitaria será responsabilidad de cada implementador. Se sugiere que cada vez que una unidad de código es terminada, o avanzada, el implementador cree un conjunto básico de casos de pruebas para probar la funcionalidad de la unidad. También será responsable de realizar los informes de verificación que especifiquen los defectos encontrados, pero no corregidos en la iteración.

Las pruebas de integración serán responsabilidad del equipo de implementadores. Estos deberán organizarse para realizarlas y en caso de encontrarse defectos y no corregirlos, realizar el informe pertinente.

Las pruebas funcionales y del sistema serán responsabilidad del equipo de verificación, utilizando el documento de especificación de requerimientos para comprobar que el sistema cumple con ellos.

Las pruebas de requerimientos no funcionales serán responsabilidad del equipo de verificación. Existen inconvenientes técnicos que hacen imposible la verificación de ciertos requerimientos no funcionales, pero que, dada las características del proyecto, se puede inferir que el sistema las cumplirá. Esto sucede por ejemplo con el requerimiento de que el sistema debe ser escalable. A menos que se libere una versión del producto apta para todo público (lo cual no sucederá), este requerimiento no podrá ser verificado, por lo tanto, se asumirá como cumplido dadas las características de la plataforma Windows Azure en la cual el sistema se basa.

Un riesgo importante que puede afectar negativamente al proceso de verificación es que el proyecto se basa en tecnologías nuevas que ninguno de los integrantes conoce de antemano. Esto puede ocasionar que las estimaciones de tiempo sean imprecisas, causando el posible problema de pasarse de fecha en la entrega de código y por lo tanto afectando al proceso de verificación. Incluso, si el calendario se aprieta mucho, es posible que el equipo de desarrollo no realice las correspondientes pruebas de verificación, o que las realice, pero con un bajo nivel de calidad.

Un requerimiento no funcional que puede ser problemático en el proceso de verificación es el que especifica que el sistema debe correr en los browsers: IE (desde la versión 6 en adelante), Mozilla Firefox, Google Chrome, y Safari. Este requerimiento obliga a que las pruebas funcionales se realicen varias veces, una vez en cada browser, relenteciendo el proceso de verificación.

1.4. Identificación del proyecto

Los documentos usados para elaborar el Plan de Verificación son los siguientes:

- Documento de requisitos funcionales y no funcionales (SGEH-RF y SGEH-RNF)
- Especificaciones de Casos de Uso (SGEH-ECU)

1.5. Estrategia de evolución del Plan

El responsable del monitoreo del Plan de Verificación y Validación es el Responsable de Verificación. Este debe cerciorarse que el plan se está cumpliendo en cada etapa del proyecto.

Las modificaciones al plan no están agendadas por el momento. Se esperará a evolucionar en las fases para planificar los cambios al plan, en especial, se esperarán los cambios en los requerimientos del sistema.

El equipo de verificación evaluará los posibles cambios al Plan de Verificación que se crean necesarios. El resto del equipo podrá sugerir un cambio al plan si lo cree necesario, y el equipo de verificación lo discutirá y analizará.

Los cambios al plan serán registrados en el Documento de Evaluación y Ajuste del Plan de V&V, y dependiendo de la importancia del cambio, se informará o no por mail al resto del grupo, o solo a la parte que propuso el cambio.

2. Requerimientos para verificar

En la lista a continuación se presentan los elementos, casos de uso, requerimientos funcionales y requerimientos no funcionales, que serán verificados.

Estos se corresponden con los casos de usos prioritarios para esta fase que el cliente dio a entender al equipo de analistas. Luego, en la próxima fase se verificarán los restantes requerimientos y/o casos de usos.

- Registrarse (Usuario no registrado).
- Loguearse (Usuario no logueado).
- Desloguearse (Usuario logueado).
- Registrar eventos nuevos (Usuario logueado).
- Visualizar detalles de los eventos (Usuario en general).
- Comprar eventos (Usuario logueado).
- Actualizar información de la cuenta (Usuario logueado).
- Ingresar información de la cuenta (Usuario logueado).
- Visualizar información de la cuenta (Usuario logueado).
- Obtener listado de eventos favoritos (Usuario logueado).
- Obtener listado de los eventos solicitados (Usuario logueado).
- Obtener listado de los productos solicitados (Usuario logueado).

Requerimientos no funcionales que serán verificados a lo largo de todo el proyecto (podrán modificarse en el futuro)

- Se podrá acceder a dicha aplicación desde los siguientes navegadores: Internet Explorer (superior a la versión 11.0), Mozilla Firefox, Safari y Chrome.
- Almacenamiento de datos en una base de datos.
- Validación de operaciones
- El tiempo estimado de visualización de los eventos debe ser mejor o igual a 4 segundos si se cuenta con un ancho de banda de 510 Kb/segundos.
- La aplicación para ejecutar en Windows Mobile bastará con que funcione en un emulador.

3. Estrategia de Verificación

Esta sección presenta el enfoque recomendado para la verificación. Describe como se verificarán los elementos.

Se indicarán las técnicas usadas y el criterio para saber cuándo una prueba se completó (criterio de aceptación).

3.1. Tipos de pruebas

3.1.1. Prueba de Funcionalidad

La prueba de funcionalidad se enfoca en requerimientos para verificar que se corresponden directamente a casos de usos o funciones y reglas del negocio. Los objetivos de estas pruebas son verificar la aceptación de los datos, el proceso, la recuperación y la implementación correcta de las reglas del negocio.

3.1.1.1. Objetivo de la prueba

Asegurar la funcionalidad apropiada del objeto de prueba, incluyendo la navegación, entrada de datos, proceso y recuperación.

3.1.1.2. Técnica

Ejecutar los casos de uso usando datos válidos y no válidos, para verificar lo siguiente:

- Se obtienen los resultados esperados cuando se usan datos válidos.
- Cuando se usan datos no válidos se despliegan los mensajes de error o advertencia apropiados.
- Se aplica apropiadamente cada regla del negocio.

3.1.1.3. Criterio de aceptación

Todas las pruebas planificadas se realizaron. Todos los defectos encontrados han sido debidamente identificados.

3.1.1.4. Consideraciones especiales

Las pruebas de funcionalidad se verán afectadas si se retrasa el desarrollo de la implementación, o si el software entregado al área de verificación contiene muchos defectos que impidan las pruebas de las restantes funcionalidades.

Dentro de lo posible se intentará realizar tests cases que sean automáticos, de manera de poder reproducirlos para realizar los tests de regresión. Esto no es fácil dado que se cuenta con una interfaz web que correrá sobre varios browsers de internet y el armado de tests cases automatizados es más complejo que el manual.

3.1.2. Prueba de Ciclo del Negocio

Esta prueba debe simular las actividades realizadas en el proyecto en el tiempo. Se debe identificar un período, que puede ser un año, y se deben ejecutar las transacciones y actividades que ocurrirían en el período de un año. Esto incluye todos los ciclos diarios, semanales y mensuales y eventos que son sensibles a la fecha.

En este caso, el único requerimiento que determina un ciclo diario es el resumen por día de las noticias comentadas. Si en el futuro surge algún otro requerimiento, se agregará debidamente.

3.1.2.1. Objetivo de la prueba

Asegurar que la aplicación funcione de acuerdo con los requerimientos del negocio.

3.1.2.2. Técnica

La prueba debe simular ciclos de negocios realizando lo siguiente:

Las pruebas de funcionalidad se deben modificar para aumentar la cantidad de veces que se ejecuta cada función, simulando varios usuarios diferentes en un período determinado.

Todas las funciones sensibles a la fecha se deben ejecutar con fechas válidas y no válidas o períodos de tiempo válidos y no válidos.

Para cada prueba realizada verificar lo siguiente:

- Se obtienen los resultados esperados cuando se usan datos válidos.
- Cuando se usan datos no válidos se despliegan los mensajes de error o advertencia apropiados.
- Se aplica apropiadamente cada regla del negocio.

3.1.2.3. Criterio de aceptación

Todas las pruebas planificadas se realizaron. Todos los defectos encontrados han sido debidamente identificados.

3.1.2.4. Consideraciones especiales

Las fechas del sistema y eventos requieren actividades de soporte especiales. Esto es, poder cambiar la fecha y hora del reloj de servidor donde correrá el sistema.

3.1.3. Prueba de Interfaz de Usuario

Esta prueba verifica que la interfaz de usuario proporcione al usuario el acceso y navegación a través de las funciones apropiadas. Además, asegura que los objetos presentes en la interfaz de usuario se muestren como se espera y conforme a los estándares establecidos por la empresa o de la industria.

3.1.3.1. Objetivo de la prueba

Verificar que: la navegación a través de los elementos que se están probando reflejen las funciones del negocio y los requerimientos, incluyendo manejo de ventanas, campos y métodos de acceso; los objetos de las ventanas y características, como menúes, tamaño, posición, estado funcionen de acuerdo con los estándares.

3.1.3.2. Técnica

Crear o modificar pruebas para cada ventana verificando la navegación y los estados de los objetos para cada ventana de la aplicación y cada objeto dentro de la ventana.

3.1.3.3. Criterio de aceptación

Cada ventana ha sido verificada exitosamente siendo consistente con una versión de referencia o estándar establecido.

3.1.4. Prueba de Performance

En esta prueba se miden y evalúan los tiempos de respuesta, los tiempos de transacción y otros requerimientos sensitivos al tiempo. El objetivo de la prueba es verificar que se logren los requerimientos de performance. La prueba de performance es implementada y ejecutada para poner a punto los destinos de pruebas de performance como función de condiciones de trabajo o configuraciones de hardware. Para este sistema, las pruebas de performance son respecto a los tiempos de carga para las páginas.

3.1.4.1. Objetivo de la prueba

Verificar la performance de determinadas transacciones o funciones de negocio bajo ciertas condiciones:

- condiciones de trabajo normales conocidas.
- peores casos de condiciones de trabajo conocidas.

3.1.4.2. Técnica

- Levantar el servidor ASP en la plataforma Azure utilizando la cuenta de developer. Luego utilizar el software OpenSTA para realizar las pruebas de tiempos de respuesta en las páginas del portal Web.

3.1.4.3. Criterio de aceptación

Las paginas más importantes (las de los Casos de Uso más prioritarios) tienen un tiempo de respuesta aceptable según los requerimientos no funcionales.

3.1.5. Prueba de Carga

3.1.5.1. Objetivo de la prueba

Verificar que el sistema responderá adecuadamente bajo condiciones de carga importantes que simulen lo mas realista posible un escenario real al que se podría enfrentar el sistema en producción. El objetivo es determinar la cantidad más razonable de usuarios que puede soportar un nodo, para luego extrapolar a varios nodos simultáneamente.

3.1.5.2. Técnica

- Levantar el servidor ASP en la plataforma Azure utilizando la cuenta de developer.

- Luego utilizar el software OpenSTA para realizar las pruebas de carga simulando un uso racional en las acciones de los usuarios, por ejemplo, un 60% de los usuarios solo ingresara a la portada sin registrarse. Un 20% se registrará y mirará su portada personalizada. Un 10% se registrará, realizará búsquedas y verá algunas noticias y un 10% se registrará y subirá alguna noticia.

- Luego incrementar el número de usuarios manteniendo la proporción de las clases de equivalencia

3.1.5.3. Criterio de aceptación

El objetivo de la prueba es determinar la carga de usuarios más común que puede soportar un único nodo, por lo que no se tiene un criterio de aceptación.

3.1.6. Prueba de Esfuerzo (stress, competencia por recursos, bajos recursos)

Las pruebas de stress serán similares a las de carga, pero con la diferencia que, en vez de simular una carga balanceada, se simularán condiciones límite, por ejemplo, el logueo repentino de muchos usuarios. La búsqueda simultanea de distintas noticias filtradas por *tags*, la subida de noticias a la vez, etc. El objetivo es encontrar un límite a las capacidades del sistema.

3.1.7. Prueba de Volumen

No aplica ya que la plataforma Windows Azure se compromete a dar un servicio de storage altamente escalable a las aplicaciones que corren dentro de la nube.

3.1.8. Prueba de Seguridad y Control de Acceso

La Prueba de Seguridad y Control de Acceso se enfoca en dos áreas de seguridad:

- Seguridad en el ámbito de aplicación, incluyendo el acceso a los datos y a las funciones de negocios.
- Seguridad en el ámbito de sistema, incluyendo conexión, o acceso remoto al sistema.

La seguridad en el ámbito de aplicación asegura que, basado en la seguridad deseada los actores están restringidos a funciones o casos de uso específicos o limitados en los datos que están disponibles para ellos.

La seguridad en el ámbito de sistema asegura que, solo los usuarios con derecho a acceder al sistema son capaces de acceder a las aplicaciones y solo a través de los puntos de ingresos apropiados. En este caso, dado que la aplicación será de uso público, no se requiere este tipo de seguridad.

3.1.8.1. Objetivo de la prueba

Seguridad en el ámbito de aplicación: Verificar que un actor pueda acceder solo a las funciones o datos para los cuales su tipo de usuario tiene permiso.

3.1.8.2. Técnica

Seguridad en el ámbito de aplicación: Identificar y hacer una lista de cada tipo de usuario y las funciones y datos sobre las que cada tipo tiene permiso.

Crear pruebas para cada tipo de usuario y verificar cada permiso creando operaciones específicas para cada tipo de usuario.

3.1.8.3. Criterio de aceptación

Para cada tipo de actor conocido las funciones y datos apropiados están disponibles, y todas las operaciones funcionan como se espera y ejecutan las pruebas de Funcionalidad de la aplicación.

3.1.9. Prueba de Fallas y Recuperación

Las Pruebas de Fallas y Recuperación aseguran que el software puede recuperarse de fallas de hardware, software o mal funcionamiento de la red sin pérdida de datos o de integridad de los datos.

La Prueba de Recuperación es un proceso en el cual la aplicación o sistema se expone a condiciones extremas, o condiciones simuladas, para causar falla, como fallas en dispositivos de Entrada/Salida o punteros a la base de datos inválidos. Los procedimientos de recuperación se invocan y la aplicación o sistema es monitoreado e inspeccionado para verificar que se recupera apropiadamente la aplicación o sistema y se logre la recuperación de datos.

Al igual que en casos anteriores, la plataforma Windows Azure provee mecanismos de alta disponibilidad para los servicios que estén alojados en la nube, teniendo réplicas de las máquinas virtuales de los servidores en máquinas físicamente distintas y en dominios de fallas diferentes. Esto mejora la disponibilidad del sistema en cuanto a caídas de los servidores.

En cuanto a recuperación de los datos, el servicio de storage de Windows Azure realiza 3 copias de toda la información que se mueve en una cuenta, y asegura la alta disponibilidad de esta para los servicios que corren en la nube.

Dadas estas condiciones, las pruebas de falla y recuperación carecen de sentido.

3.1.10. Prueba de Configuración

No aplica debido a las características mencionadas de la plataforma.

3.1.11. Prueba de Instalación

El cliente no requiere que el sistema sea instalado realmente. Una instalación real requiere que se paguen los costos requeridos para subir el sistema a los datacenters de Microsoft que dan soporte a la nube.

3.1.12. Prueba de Documentos

La Prueba de Documentos debe asegurar que los documentos relacionados al software que se generen en el proceso sean correctos, consistentes y entendible. Se incluyen como documentos los Materiales para Soporte al Usuario, Documentación Técnica, Ayuda en Línea y todo tipo de documento que forme parte del paquete de software.

3.1.12.1. Objetivo de la prueba

Verificar que el documento objeto de prueba sea:

- Correcto, esto es, que cumpla con el formato y organización para el documento establecido en el proyecto.
- Consistente, esto es, que el contenido del documento sea fiel a lo que hace referencia. Si el documento es Documentación de Usuario, que la explicación de un procedimiento sea exactamente como se realiza el procedimiento en el software, si se muestran pantallas que sean las correctas.
- Entendible, esto es, que al leer el documento se entienda correctamente lo que expresa y sin ambigüedades, además que sea fácil de leer.

3.1.12.2. Técnica

Para verificar que el documento es correcto se debe comparar con el estándar definido si existe o con las pautas de documentación y ver que el documento cumple con ellas.

Para verificar que el documento es consistente se debe ejecutar el programa siguiendo el documento en caso de los Materiales de Soporte al Usuario y comprobar que lo que se explica en estos documentos es exactamente lo que se ejecuta en el programa. En caso de Documentación Técnica se debe revisar el código al cual corresponde la documentación y comprobar que dicha documentación describe el código.

Para verificar que el documento es entendible, debe comprobar que se entiende correctamente, que no tiene ambigüedades y que sea fácil de leer.

3.1.12.3. Criterio de aceptación

El documento expresa exactamente lo que debe expresar, no hay diferencias entre lo que está escrito y el objeto de la descripción (operación de software, código de programa, decisiones técnicas) y se entiende fácilmente.

3.1.12.4. Consideraciones especiales

La documentación más importante para el cliente es la técnica, por lo tanto, se hará mayor énfasis en la verificación de esta.

3.2. Herramientas

Utilizaremos la herramienta de reporte de incidentes Mantis, ejecutando en servidores de la facultad de ingeniería. Dentro de las características más importantes de esta herramienta se encuentran:

- Especificación del componente en cada reporte de incidente.
- Asignación de prioridades a los incidentes reportados.
- Asignación a la persona encargada de ejecutar la corrección del incidente y envío automático de mail a ésta.

Para realizar los tests cases de pruebas funcionales se utilizará la herramienta o*pen source Selenium* (http://seleniumhq.org/) en su versión 1.0. Esta herramienta permite la creación de pruebas automáticas sobre navegadores Web de manera muy fácil e intuitiva y permite luego la reproducción de las pruebas a distintas velocidades y con parámetros ajustables. Además, permite guardar los casos de pruebas utilizando distintos lenguajes de programación como ser C#, Java, Perl, Python, Ruby, Perl, etc. También permite la utilización de varios navegadores para correr las pruebas, por ejemplo, Firefox 2 y 3, IE 7 y 8, Safari 2 y 3 entre otros.

Para realizar pruebas de escalabilidad, stress y carga se utilizará el software *open source OpenSTA* 1.4.4. Todavía no ha sido estudiada en profundidad esta herramienta.

Se utilizará el IDE Microsoft Visual Studio Code 2020, con SP1 + Windows Azure SDK + Windows Mobile SDK para levantar el ambiente de prueba.

Se utilizará Microsoft SQL Server 2018 para la administración de la BD.

4. Recursos

En esta sección se presentan los recursos recomendados para el proyecto SGEH, sus principales responsabilidades y su conocimiento o habilidades.

4.1. Roles

En la tabla a continuación se muestra la composición de personal para el proyecto Cloud News en el área Verificación del Software.

Rol	Cantidad mínima de recursos recomendada	Responsabilidades
Responsable de verificación	1	• Identifica, prioriza e implementa los casos de prueba. • Genera el Plan de Verificación. • Genera el Modelo de Prueba. • Evalúa el esfuerzo necesario para verificar. • Proporciona la dirección técnica. • Adquiere los recursos apropiados. • Proporciona informes sobre la verificación.
Asistente de verificación	4	• Ejecuta las pruebas • Registra los resultados de las pruebas. • Documenta los pedidos de cambio.
Implementador	Equipo de implementación completo	• Ejecuta las pruebas unitarias • Diseña las pruebas de integración planificadas • Ejecuta las pruebas de integración

4.2. Sistema

En la siguiente tabla se establecen los recursos de sistema necesarios para realizar la verificación.

Es recomendable que el sistema simule el entorno de producción, reduciendo los accesos y los tamaños de bases de datos si fuera apropiado.

Recurso	Nombre/Tipo
Servidor de base de datos	SQL Server 2015

PC Cliente para pruebas	PC con acceso a internet
Requerimientos especiales	Development Fabric y Development Storage
Repositorio de pruebas	Development Storage
IDE	Visual Studio Code 2020

5. Hitos del proyecto de Verificación

La verificación del Cloud News debe incorporar actividades de prueba para cada verificación identificada en las secciones anteriores

Actividad que determina el hito	Esfuerzo	Fecha de comienzo	Fecha de finalización
Planificar la verificación	10 hs	semana 13	semana 14
Elaborar casos de prueba	20 hs	semana 13	semana 14
Ajuste y Control de Verificación	10 hs	semana 13	semana 14
Ejecutar la verificación	50 hs	semana 13	semana 14
Evaluar la verificación	10 hs	semana 13	semana 14

Nota: Esta sección de hitos del proyecto es muy probable que sufra varias modificaciones en posteriores versiones.

6. Entregables

6.1. Modelo de Casos de Prueba

Documento	**Modelo de Casos de Prueba**
Creado por	El responsable de verificación, Javier Portella.
Para quien	Es la guía para realizar las pruebas del sistema y lo usarán los Asistentes de verificación y el responsable de verificación cuando se ejecuten las pruebas del sistema.
Fecha de liberación	Será liberado luego de la fase inicial.

6.2. Informes de Verificación

Documento	Se genera un documento **Informe de Verificación Unitaria** por cada prueba unitaria que se realice al sistema.
Creado por	Las personas que ejecutan las pruebas.
Para quien	Es el retorno para los implementadores de la tarea de verificación, que detalla los errores encontrados para que puedan ser corregidos.
Fecha de liberación	Será liberado luego de cada verificación unitaria.

Documento	Se genera un documento **Informe Consolidación** por cada consolidación que se realice al sistema.
Creado por	Las personas que ejecutan las pruebas.
Para quien	Es el retorno para los implementadores de la tarea de consolidación, que detalla los errores encontrados para que puedan ser corregidos.
Fecha de liberación	Será liberado luego de cada consolidación.

Documento	Se genera un documento **Informe de Verificación de Integración** por cada prueba de integración que se realice al sistema.
Creado por	Las personas que ejecutan las pruebas.
Para quien	Es el retorno para los implementadores de la tarea de verificación, que detalla los errores encontrados para que puedan ser corregidos.
Fecha de liberación	Será liberado luego de cada verificación de integración.

Documento	Se genera un documento **Informe de Verificación de Sistema** por cada prueba de sistema que se realice.
Creado por	Las personas que ejecutan las pruebas.
Para quien	Es el retorno para los implementadores de la tarea de verificación, que detalla los errores encontrados para que puedan ser corregidos.
Fecha de liberación	Será liberado luego de cada verificación de sistema.

6.3. Evaluación de la verificación

Documento	Se genera un documento **Evaluación de la verificación** por cada prueba que se realice al sistema. Este documento contiene las fallas encontradas en el sistema, la cobertura de la verificación realizada y el estado del sistema.
Creado por	El responsable de verificación, que toma como fuente de su trabajo los Informes de verificación.
Para quien	Es el resumen de la tarea de verificación y es el retorno para todo el equipo de trabajo del estado del sistema.
Fecha de liberación	Será liberado luego de cada verificación, unitaria, de integración y de sistema.

6.4. Informe final de verificación

Documento	El documento **Informe final de verificación** es el resumen de la verificación final del sistema antes de que sea liberado al entorno del usuario.
Creado por	El responsable de verificación, que toma como fuente

	de su trabajo los Informes de verificación.
Para quien	Indica el estado del sistema.
Fecha de liberación	Será liberado luego de la verificación final del sistema.

7. Dependencias

7.1. Dependencia de personal

Será necesaria en la fase de construcción la participación de los cuatro asistentes de verificación y del responsable de verificación.

7.2. Dependencia de software

El software por verificar debe haber sido verificado previamente por los desarrolladores (pruebas unitarias y pruebas de integración), y debe ser entregado en una fecha apropiada.

7.3. Dependencia de hardware

Las pruebas que verifiquen la funcionalidad de la interfaz web del sistema, serán ejecutadas de manera local, por lo que debe tener preparado el ambiente de pruebas para el mismo.

Las pruebas que verifiquen el requerimiento no funcional de tiempo de carga de la página requieren una maquina cliente ejecutando el browser que accede a la página, y el sistema corriendo sobre Cloud Computing en la cuenta de developer.

Las pruebas que verifiquen la funcionalidad del dispositivo móvil requieren de un emulador del dispositivo móvil con la aplicación cargada, y una máquina servidor que ejecutará el sistema emulando la nube.

8. Riesgos

En esta sección se detallan los riesgos detectados que puedan afectar la normal realización de las tareas de verificación.

8.1. Planificación

Las pruebas de verificación se verán afectadas si se retrasa el desarrollo de la implementación, o si el software entregado al área de verificación contiene muchos defectos que impidan las pruebas de las restantes funcionalidades.

Evaluación del riesgo: Alto.

Probabilidad de ocurrencia: Alta.

Contingencia: Realizar la verificación atrasada del componente. Si es necesario, se modifica la agenda establecida.

8.2. Gestión

Debido a la inexperiencia del equipo, el responsable de verificación puede planificar de forma incompleta la verificación.

Evaluación del riesgo: Medio, ya que el modelo de proceso es incremental, existe la posibilidad de enmendar las omisiones posteriormente.

Probabilidad de ocurrencia: Alta.

Contingencia: Completar el plan posteriormente.

9. Apéndice

9.1. Niveles de gravedad de error

En muchas actividades del proceso de verificación se deben clasificar los errores según su nivel de gravedad. Se asigna un nivel de gravedad a los errores para poder capturar de alguna manera su impacto en el sistema. Además, para poder evaluar la verificación y el sistema.

A continuación, se da una sugerencia de cuatro niveles diferentes de gravedad de error:

- **Catastrófico**: un error cuya presencia impide el uso del sistema.
- **Crítico**: un error cuya presencia causa la pérdida de una funcionalidad crítica del sistema. Si no se corrige el sistema no satisfará las necesidades del cliente.
- **Marginal**: un error que causa un daño menor, produciendo pérdida de efectividad, pérdida de disponibilidad o degradación de una funcionalidad que no se realiza fácilmente de otra manera.
- **Menor**: un error que no causa perjuicio al sistema, pero que requiere mantenimiento o reparación. No causa pérdida de funcionalidades que no se puedan realizar de otra manera.

9.2. Niveles de aceptación para los elementos verificados

Se debe establecer un nivel de aceptación para los elementos verificados para poder establecer el estado en el que se encuentra el proyecto.

En esta sección defina niveles de aceptación y los criterios de pertenencia a cada nivel.

Como ejemplo de niveles de aceptación:

- **No aprobado**: el elemento verificado tiene errores catastróficos (uno o varios) que impiden su uso o tiene errores críticos (uno o varios) que hacen que el elemento verificado no sea confiable. El usuario no puede depender de él para realizar el trabajo.
- **Aprobado con observaciones**: el elemento verificado no tiene errores catastróficos, ni errores críticos, pero tiene errores marginales (uno o varios) que hacen que el elemento de software se degrade en algunas situaciones.
- **Aprobado**: el elemento verificado no tiene errores o tiene errores menores que no afectan el normal funcionamiento del elemento.

9.3. Glosario

SGEH	Sistema de Gestión de Eventos HoliMori