Oefa Organismo de Evaluación y Fiscalización Ambiental	**FICHA DE PROCEDIMIENTO**	Código: PA0302 Versión: ***03*** Fecha: 22/12/2022

NOMBRE DEL PROCEDIMIENTO	Desarrollo y mantenimiento de sistemas de información

APROBACIÓN		
Nombre y cargo	**Órgano o Unidad Orgánica**	**Firma y sello**
Elaborado por: Zico Alexis Yacila Espinoza Jefe de la Oficina de Tecnologías de la Información	Oficina de Tecnologías de la Información	Oefa Organismo de Evaluación y Fiscalización Ambiental Firmado digitalmente por: YACILA ESPINOZA Zico Alexis FAU 20521286769 soft Cargo: Jefe de la Oficina de Tecnologías de la Información Lugar: Sede Central - Lima\Lima\Jesus Maria Motivo: Soy el autor del documento
Revisado por: Elvis Romel Palomino Pérez Jefe de la Oficina de Planeamiento y Presupuesto	Oficina de Planeamiento y Presupuesto	Oefa Organismo de Evaluación y Fiscalización Ambiental Firmado digitalmente por: PALOMINO PEREZ Elvis Romel FAU 20521286769 soft Cargo: Jefe de la Oficina de Planeamiento y Presupuesto Lugar: Sede Central - Lima\Lima\Jesus Maria Motivo: Soy el autor del documento
Revisado por: Gonzalo Pinto Bazurco Mendoza Jefe de la Oficina de Asesoría Jurídica	Oficina de Asesoría Jurídica	Oefa Organismo de Evaluación y Fiscalización Ambiental Firmado digitalmente por: PINTO BAZURCO MENDOZA Gonzalo FAU 20521286769 soft Cargo: Jefe de la Oficina de Asesoría Jurídica Lugar: Sede Central - Lima\Lima\Jesus Maria Motivo: Soy el autor del documento
Aprobado por: Miriam Alegría Zevallos Gerenta General	Gerencia General	Oefa Organismo de Evaluación y Fiscalización Ambiental Firmado digitalmente por: ALEGRIA ZEVALLOS Miriam FAU 20521286769 soft Cargo: Gerenta General Lugar: Sede Central - Lima\Lima\Jesus Maria Motivo: Soy el autor del documento

CONTROL DE CAMBIOS		
Versión	**Sección del Procedimiento**	**Descripción del cambio**
00	-	Versión inicial del procedimiento[1]

[1] Aprobado mediante la Resolución N° 075-2019-OEFA/GEG de fecha 31 de diciembre de 2019.

Oefa Organismo de Evaluación y Fiscalización Ambiental	**FICHA DE PROCEDIMIENTO**	Código: PA0302
		Versión: ***03***
		Fecha: 22/12/2022

CONTROL DE CAMBIOS		
Versión	**Sección del Procedimiento**	**Descripción del cambio**
01	Todas las secciones	- Se modifica el objetivo y alcance del procedimiento. - Se actualiza la base normativa - Se modifican las consideraciones generales. - Se *incorporan* definiciones. - Se actualizan actividades - Se incluyen los Formatos PA0302-F13: *"Acta de pruebas de rendimiento"*, PA0302-F14: *"Acta de capacitación"*; y, PA0301-F10: *"Acta de reunión"*.[2]
02	Definiciones, siglas, actividad del procedimiento y documentos que se generan	- Se incluyó base normativa, se adiciona una definición, se añadieron dos siglas, se realizan precisiones en las actividades números 1, 2, 5, 8, 9, 19 y 22 del procedimiento. - Se adecúa el versionamiento de los formatos e instructivo a la versión del procedimiento[3].
03	***Alcance, base normativa, actividades, documentos que se generan y anexo del procedimiento***	- ***Se realiza precisiones en la base normativa, definiciones, y se precisa las actividades números 1,2,3,4,5,6,8,10,13,14,15,17,19 y 20.*** - ***Se modifica la denominación del formato PA0302-F05 "Control de Despliegue" por PA0302-F05 Control de cambios.*** - ***Se incluyen nuevos Anexos: Estándar de base de datos, Estándar de infraestructura en la nube, Estándar de seguridad de desarrollo seguro, Estándar para aplicaciones desplegadas en jboss, Estándar para aplicaciones desplegadas en Kubernetes[4].***

OBJETIVO	Establecer actividades para la ejecución del desarrollo y mantenimiento de los sistemas de información alineados con los requerimientos de las áreas usuarias.
ALCANCE	***El*** presente procedimiento es de cumplimiento obligatorio de la Oficina de Tecnologías de la Información y de las áreas del OEFA. Comprende desde la especificación de la solicitud del desarrollo de software hasta la comunicación de la atención de los documentos y elementos para el pase a producción.
RESPONSABLE DEL PROCEDIMIENTO	Jefe/a de la Oficina de Tecnologías de la Información
BASE NORMATIVA	- Ley N° 28612, Ley que norma el uso, adquisición y adecuación del software en la administración pública. - Decreto Legislativo N° 1412, Decreto Legislativo que aprueba la Ley de Gobierno Digital. - Decreto Supremo N° 024-2006-PCM, Decreto Supremo que aprueba el Reglamento de la Ley N° 28612, Ley que norma el uso, adquisición y adecuación del software en la administración pública. - Decreto Supremo N° 013-2017-MINAM, Decreto Supremo que aprueba el Reglamento de Organización y Funciones del Organismo de Evaluación y Fiscalización Ambiental - OEFA. - Decreto Supremo N° 029-2021-PCM, Decreto Supremo que aprueba el Reglamento del Decreto Legislativo N° 1412, Decreto Legislativo que aprueba la Ley de Gobierno Digital, y establece disposiciones sobre las condiciones, requisitos y uso de las tecnologías y medios electrónicos en el procedimiento administrativo.

[2] Modificado mediante la Resolución de Gerencia General N° 051-2021-OEFA/GEG de fecha 09 de junio de 2021.

[3] Modificado mediante la Resolución de Gerencia General N° 0063-2022-OEFA/GEG de fecha 26 de mayo de 2022.

[4] ***Modificado mediante la Resolución de Gerencia General N° 00114-2022-OEFA/GEG de fecha 22 de diciembre de 2022.***

	- ***Decreto Supremo N°103-2022-PCM, Decreto Supremo que aprueba la Política Nacional de Modernización de la Gestión Pública al 2030.*** - Resolución Ministerial N° 073-2004-PCM, que aprueba la Guía para la Administración Eficiente del Software Legal en la Administración Pública. - Resolución Ministerial N° 004-2016-PCM, que aprueba el uso obligatorio de la Norma Técnica Peruana *"NTP ISO/IEC 27001:2014 Tecnología de la Información. Técnicas de Seguridad. Sistemas de Gestión de Seguridad de la Información. Requisitos. 2a. Edición"*, en todas las entidades integrantes del Sistema Nacional de Informática. - Resolución Ministerial N° 041-2017-PCM, que aprueba el uso obligatorio de la Norma Técnica Peruana *"NTP ISO/IEC 12207:2016 - Ingeniería de Software y Sistemas. Procesos del ciclo de vida del software. 3a Edición"*, en todas las entidades integrantes del Sistema Nacional de Informática. - Resolución Ministerial N° 119-2018-PCM, que dispone la creación de un Comité de Gobierno Digital en cada entidad de la Administración Pública. - Resolución Ministerial N° 087-2019-PCM, que aprueba las disposiciones sobre la conformación y funciones del Comité de Gobierno Digital. - Resolución de Presidencia del Consejo Directivo N° 071-2018-OEFA/PCD, que conforma el Comité de Gobierno Digital del Organismo de Evaluación y Fiscalización Ambiental - OEFA. - Resolución de Presidencia del Consejo Directivo N° 00020-2022-OEFA/PCD, que aprueba la designación y funciones del Oficial de Seguridad y Confianza Digital del OEFA. Las referidas normas incluyen sus modificatorias.
CONSIDERACIONES GENERALES	- Para el desarrollo de software se utilizan los estándares de programación Java, Base de datos, OWASP (Open Web Application Security Project). - En los casos en que se tercerice el desarrollo de sistemas de información o aplicativos, es responsabilidad de las áreas usuarias que en sus términos de referencia se precise el uso de formatos u otros documentos que la OTI requiera, cuando corresponda.
DEFINICIONES	- **Ambiente de desarrollo:** Entorno de desarrollo de un proyecto o mantenimiento de software, donde se realizan los prototipos, la codificación del software y las pruebas unitarias. - **Ambiente de calidad:** Entorno de calidad donde se realizan las verificaciones y validaciones de software previo a su despliegue en el ambiente de producción. - **Ambiente de producción**: Entorno de producción de un proyecto de desarrollo de software, es el término que cubre todo lo que necesita el proyecto para desplegar el sistema y para que el/la usuario/a final lo pueda visualizar e iniciar con su uso. - **Aplicativo:** Programa informático que se ha diseñado como herramienta para permitir a los usuarios realizar uno o diversos tipos de tareas. - **Archivos de configuración:** Conjunto de datos que determina el valor de algunas variables de un programa o de un sistema operativo. Estas opciones generalmente son cargadas en su inicio del sistema. - ***Área Usuaria: Órganos, unidades orgánicas, coordinaciones y unidades funcionales establecidas por Resolución por la Alta Dirección del OEFA.*** - **Arquitectura de software:** Conjunto de patrones y abstracciones coherentes que proporcionan un marco definido y claro para interactuar con el código fuente del software. Se denomina también arquitectura lógica. - **Catálogo de servicios**: Base de datos o documento estructurado que contiene información sobre todos los servicios vigentes e incluye aquellos que se pueden implementar, forma parte del aplicativo de mesa de ayuda. - **Código fuente:** Conjunto de líneas de texto en un lenguaje de programación determinado con los pasos que debe seguir el computador para la correcta ejecución de un programa específico. - **Componentes de software:** Elementos de un sistema de software que ofrece un conjunto de servicios, o funcionalidades, a través de interfaces definidas. Los cuales pueden ser: código fuente, fuentes y scripts. - **Datos:** Información usada para la ejecución del plan de pruebas de los sistemas de información. - **Elementos de configuración:** Fuentes del aplicativo o sistema de información, archivos de configuración, script de base de datos, informe de pruebas unitarias, manuales del sistema, manual de el/la usuario/a, etc. - **Funcionalidad:** Permite la verificación del funcionamiento del sistema o aplicativo de acuerdo a los requerimientos especificados. - **Herramienta para la gestión de requerimientos:** Sistema informático en el que se registran los requerimientos solicitados por el área usuaria. - **Iteración:** incluye las actividades de desarrollo que dan lugar al release de un producto; es decir, una versión estable y ejecutable del producto software.

	- ***Líder del área usuaria: Servidor/a civil designado/a por el/la Jefe/a o Director/a del área usuaria quien consolida los cambios a nivel funcional de las aplicaciones. También es identificado como líder usuario/a.*** - **Plan de pruebas:** Define los objetivos de la prueba de un sistema, establece y coordina una estrategia de trabajo, y provee el marco adecuado para elaborar una planificación de las actividades de prueba. - **Proyecto tecnológico:** Creación, modificación o adaptación de un producto específico gracias al empleo de la tecnología. El producto tecnológico, que es el resultado del proceso, tiene como función satisfacer una necesidad, demanda o servicio. Los cuales se realizan a nivel de infraestructura, desarrollo de sistemas de información; y, servicios de tecnologías de información. - **Pruebas de calificación**: Pruebas para demostrar que el software cumple sus especificaciones y está listo para ser usado en su entorno de destino. - **Pruebas unitarias:** Método a través del cual se comprueba el correcto funcionamiento de un código. - **Reporte de pruebas:** Documento en el que se detalla las pruebas realizadas a las aplicaciones, tanto como observaciones y mejoras que se pueda encontrar. - **Requerimientos:** Conjunto de criterios o condiciones que deben configurarse para calificar que un software cumple con sus especificaciones y está listo para ser usado en su entorno de destino. - **Requerimientos funcionales:** Declaraciones de los servicios que proveerá el sistema, de la manera en que éste reacciona a entradas particulares. En ciertos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer. - **Requerimientos no funcionales:** Pedidos que no se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. Asimismo, pueden definir las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y la representación de datos que se utiliza en la interfaz del sistema. - **Script de base de datos:** Descripciones de las instrucciones utilizadas para crear una base de datos y sus objetos. - **Seguridad de la Información: Preservación de la confidencialidad, integridad y disponibilidad de la información.** - **Sistema de información:** Conjunto de datos que interactúan entre sí con un fin común, asimismo, ayudan a administrar, recolectar, recuperar, procesar, almacenar y distribuir información relevante para los procesos fundamentales de cada organización. - **Software**: Conjunto de programas y rutinas que permiten a la computadora realizar determinadas tareas.
SIGLAS	- CGTD: Comité de Gobierno y Transformación Digital. - OTI: Oficina de Tecnologías de la Información. - SGI: Sistema de Gestión Integrado. - SGSI: Sistema de Gestión de Seguridad de la Información. - TI: Tecnologías de la información.

REQUISITOS PARA INICIAR EL PROCEDIMIENTO	
Descripción del requisito	**Fuente**
Acta de requerimiento	Áreas usuarias
Solicitud de mantenimiento de sistema de información	

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
1	Especificar requerimiento del área usuaria	El/La analista funcional de sistemas conjuntamente con el ***líder del*** área usuaria realizan la especificación de los requerimientos a nivel funcional y no funcional en el Formato PA0302-F01: *"Acta de requerimiento"* ***y remite por correo institucional el/la***	Formato PA0302-F01: *"Acta de requerimiento"* ***Correo institucional***	Analista funcional de sistemas ***Líder*** del Área usuaria	OTI Área usuaria

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
		Analista programador/a para la revisión correspondiente. **Plazo:** Tres (3) días hábiles contabilizados a partir de la recepción de la solicitud del área usuaria.			
2	***Evaluar el contenido del acta del requerimiento a nivel técnico y verificar si es un nuevo sistema de información***	***Evalúa el contenido del Acta de requerimiento a nivel técnico según el Formato PA0302-F01.*** ***Plazo: de cuatro (4) días hábiles.*** ***¿El acta de requerimiento es conforme?*** ***No: Comunica, mediante correo institucional, a el/la Analista Funcional de Sistemas la no viabilidad y va a la actividad N° 3.*** ***Sí: Va a la actividad N° 4.*** ***¿Se trata de un nuevo sistema de información?*** ***Sí: El/La Analista de gestión de proyectos de tecnologías de información elabora el informe y lo remite a el/la Jefe/a de la OTI y se sigue las actividades descritas en el procedimiento PA0301 "Formulación e implementación de proyectos tecnológicos", mediante el SIGED.*** ***¿El CGTD lo encuentra viable?*** ***Sí: Se registra como acuerdo en el Acta de sesión del CGTD y va a la actividad N° 9.*** ***No: El/La Analista de proyectos de tecnologías de la información comunica mediante correo institucional al área usuaria. Va a la actividad N° 3.*** ***No: Va a la actividad N° 12.***	***Formato PA0302-F01: "Acta de requerimiento" revisado y validado técnicamente*** ***Correo institucional*** ***Informe*** ***Acta de sesión del CGTD*** ***SIGED***	***Analista programador/a*** ***Analista de proyectos de tecnologías de la información***	***OTI***
3	Informar no viabilidad	Informa al ***líder del*** área usuaria, mediante correo institucional, la no viabilidad en base a lo comunicado por el/la analista programador/a. **Plazo:** Un (1) día hábil de recibido el requerimiento. **Fin del procedimiento.**	Correo Institucional	Analista funcional de sistemas	OTI

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
4	Realizar la estimación del tiempo y elaborar el cronograma	Realiza la estimación del tiempo, elabora la propuesta del cronograma para la implementación del requerimiento ***y lo remite, mediante correo institucional, a el/la* Coordinador/a de desarrollo de sistemas para su verificación.**	Propuesta del Cronograma ***Correo Institucional***	Analista programador/a	OTI
5	Verificar cronograma	Verifica la propuesta del cronograma elaborada por el/la Analista Programador/a. **¿Es conforme?** **Sí**: Va a la actividad N° 6. **No:** Realiza comentarios, los remite por correo institucional y va a la actividad N° 4.	Cronograma **Correo Institucional**	Coordinador/a de desarrollo de sistemas	OTI
6	Priorizar, actualizar y comunicar los requerimientos	Prioriza y actualiza ***el Formato PA0302-F15: "Control de Requerimientos" con*** los requerimientos que serán atendidos, dentro de los dos (2) días hábiles, contados a partir de la verificación del cronograma. Comunica, mediante correo institucional, al ***líder del*** área usuaria el orden de atención de su requerimiento.	***Formato PA0302-F15: "Control de Requerimientos"*** Correo institucional	Coordinador/a de desarrollo de sistemas	OTI
7	Asignar la atención del requerimiento	Asigna, mediante correo institucional, a el/la analista programador/a la atención del requerimiento. *Nota:* *Para la asignación de la atención del requerimiento, realiza la evaluación de la carga de trabajo del equipo de desarrollo de sistemas.*	Correo institucional	Coordinador/a de desarrollo de sistemas	OTI
8	Analizar y diseñar el sistema	Analiza el requerimiento, elabora el diseño técnico y define componentes, conforme al formato PA0302-F02: *"Documento de Análisis y diseño"*. ***Para ello revisa los Anexos 1, 2 ,3 ,4 y 5, los cuales son los estándares con los que se diseñará el sistema de información.***	Formato PA0302-F02: *"Documento de Análisis y diseño"*	Analista programador/a	OTI
9	Dar conformidad de arquitectura de software	Da conformidad sobre la arquitectura de software, mediante correo institucional. **Plazo:** Dos (2) días hábiles contabilizados luego de recibir el documento de análisis y diseño remitido, por el/la Analista Programador/a, mediante correo institucional.	Correo Institucional	Analista de arquitectura de tecnológica	OTI

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
10	Revisar, validar y ***comunicar*** los aspectos de Seguridad de la información	Revisa y valida los aspectos de Seguridad de la Información, ***vinculados a la Política Específica del SGSI - item Adquisición, desarrollo y mantenimiento de sistemas, así como el Estándar de Ciclo de desarrollo seguro. Asimismo, comunica a los/as analistas programadores o contratistas, mediante correo institucional los temas relacionados al Estándar de desarrollo seguro.*** **Plazo:** Dos (2) días hábiles de recibida la conformidad. **¿Existe un entorno (ambiente) de desarrollo para el sistema de información?** **Sí:** Va a la actividad N° 12. **No**: Va a la actividad N° 11.	Correo Institucional	Analista de seguridad informática	OTI
11	Comunicar y preparar el ambiente de desarrollo para alojar el sistema de información	El/La Analista Programador/a mediante correo institucional, comunica el inicio de la preparación del ambiente de desarrollo a el/la Analista de redes y comunicaciones. El/***L***a Analista de Redes y Comunicaciones prepara el ambiente de desarrollo para alojar el sistema de información; a fin de generar el espacio en servidores informáticos, la creación de base datos para la construcción y pruebas.	Correo Institucional	Analista programador/a Analista de redes y comunicaciones	OTI
12	Codificar, versionar y ejecutar las pruebas unitarias	Codifica los componentes de software (o componentes de integración) en el sistema de control de versiones, conforme al Formato PA0302-F02 *"Documento de Análisis y diseño".* Versiona el código fuente, según los estándares de desarrollo vigentes. Ejecuta las revisiones especiales previstas. Diseña y ejecuta los casos de pruebas unitarias de acuerdo con el Formato PA0302-F03 "*Informe de pruebas unitarias".*	Formato PA0302-F02: *"Documento de Análisis y diseño"* Formato PA0302-F03: *"Informe de pruebas unitarias"*	Analista programador/a	OTI

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
13	Validar los requerimientos implementados en la aplicación y comunicar al área usuaria	Valida de forma preliminar los requerimientos implementados en la aplicación (prototipos de pantallas en la aplicación, entre otros) y lo comunica mediante correo institucional al área usuaria. El ***líder del*** área usuaria efectúa la validación. **¿Es conforme?** **No:** Va la actividad N° 12. **Sí:** Va la actividad N° 14.	Correo Institucional	***Líder del área usuaria*** Analista programador/a	OTI
14	Identificar ***y remitir*** los elementos de configuración	Identifica ***y completa*** los elementos de software o elementos de configuración en el ***Formato*** PA0302-F04 *"Solicitud de despliegue"* y documentos actualizados de acuerdo con lo dispuesto ***en los Formatos*** PA0302-F11 *"*y PA0302-F12; así como en lo establecido en el Instructivo I-OTI-PA0302-1: *"Instructivo de Desarrollo de Software"*.	Formato PA0302-F04: *"Solicitud de despliegue"* Formato PA0302-F11: *"Manual de Usuario"* Formato PA0302-F12: *"Manual de Sistema y Configuración"*	Analista programador/a	OTI
15	Revisar los requerimientos	Revisa los requerimientos (documentos y elementos de configuración), ***en el Formato PA0302-F04 "Solicitud de despliegue" y documentos actualizados de acuerdo con lo dispuesto en los Formatos PA0302-F11 "Manual de usuario" y PA0302-F12 "Manual de Sistema y configuración"; así como en lo establecido en el Instructivo I-OTI-PA0302-1: "Instructivo de Desarrollo de Software"*** **Plazo:** Un (1) día hábil de recibido el requerimiento. **¿Es conforme?** **Sí:** Va a la actividad N° 16. **No:** Va a la actividad N° 14.	-	Coordinador/a de informática y gestión administrativa	OTI
16	Remitir el requerimiento para despliegue en ambiente de calidad	Remite el requerimiento a el/la Analista de Calidad de Software, mediante correo institucional, según disponibilidad, para el despliegue del sistema de información en el ambiente de calidad.	Correo Institucional	Coordinador/a de informática y gestión administrativa	OTI

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
17	Desplegar el sistema de información en ambiente de calidad	Despliega el sistema de información en ambiente de calidad y lo registra en el Formato ***PA0302-F05: "Control de cambios".*** **¿Es conforme el despliegue?** **Sí:** Va a la actividad N° 18. **No:** Va a la actividad N° 12.	***Formato PA0302-F05: "Control de cambios"***	Analista de calidad de software	OTI
18	Planificar, diseñar y ejecutar las pruebas de calificación	Planifica las pruebas conforme al Formato PA0302-F06: "*Plan de pruebas",* detallando las responsabilidades, los plazos, las actividades y los criterios de aceptación de las pruebas de calificación. Diseña los casos de prueba, conforme al formato PA0302-F08: *"Casos de Prueba",* para las pruebas de calificación. Ejecuta los casos de prueba conforme a los Formatos PA0302-F09: *"Informe de pruebas"* y PA0302-F13: *"Acta de pruebas de rendimiento"* de acuerdo con los criterios de aceptación definidos en el Plan de pruebas. **Plazo:** Ocho (8) días hábiles por iteración de pruebas. **¿El resultado de las pruebas es conforme?** **Sí:** Va a la actividad N° 19. **No:** Va a la actividad N° 12.	Formato PA0302-F06: "*Plan de pruebas"* Formato PA0302-F08: *"Casos de Prueba"* Formato PA0302-F09: *"Informe de pruebas"* Formato PA0302-F13: *"Acta de pruebas de rendimiento"*	Analista de calidad de software	OTI
19	Presentar las funcionalidades al área usuaria	Presenta las funcionalidades (sistema o aplicativo) al ***líder del*** área usuaria quien realiza las pruebas de aceptación. El ***líder de***l área usuaria brinda conformidad, mediante el formato PA0302-F07: *"Acta de pruebas"* o a través del correo institucional. **Plazo**: Tres (3) días hábiles contados a partir del día siguiente de la ejecución de pruebas (por iteración). **¿El resultado de las pruebas es conforme?** **Sí:** Va a la actividad N° 20. **No:** Va a la actividad N° 12. *Nota 1:*	Formato PA0302-F07: "*Acta de pruebas"* Correo institucional Formato PA0302-F08: *"Casos de Prueba"*	Analista de calidad de software ***Líder del*** Área usuaria	OTI Área usuaria

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
		Los datos usados para las pruebas correspondientes en el ambiente de desarrollo y ambiente de calidad, son: ● *Datos reales homologados de producción (para mantenimientos de aplicaciones).* ● *Datos ficticios (para sistemas de información nuevos). Los cuales se mantienen en el ambiente de pruebas como evidencia mediante el Formato PA0302-F08: "Casos de Prueba", donde se coloca la data utilizada y la trazabilidad de la misma.* *Nota 2:* *En caso se usen datos sensibles, el área de infraestructura envía la información encriptada al solicitante de desarrollo y/o calidad.*			
20	Entregar los elementos de configuración verificados y elaborar la solicitud para desplegar el sistema en un ambiente de producción	Entrega ***a el/la Analista de calidad de software*** los elementos de configuración para el despliegue del sistema de información. Elabora la solicitud para desplegar el sistema en un ambiente de producción, conforme al Formato PA0302-F04: *"Solicitud de despliegue"*. Remite ***a el/la Analista de redes y comunicaciones***, a través del correo institucional, la solicitud para el despliegue.	Formato PA0302-F04: *"Solicitud de despliegue"* Correo Institucional	Analista de calidad de software	OTI
21	Realizar el despliegue del sistema en el ambiente de producción	Realiza el despliegue del sistema en ambiente de producción, ejecutando la secuencia indicada en la solicitud de despliegue. **¿La ejecución se da sin errores?** **Sí:** Va a la actividad N° 22. **No:** Va a la actividad N° 20.	-	Analista de redes y comunicaciones	OTI
22	Comunicar la atención del requerimiento	Elabora el formato PA0302-F10: *"Acta de conformidad general"* y comunica, mediante correo institucional, a el/la Supervisor/a de soporte técnico para la actualización del catálogo de servicios de TI. **Fin del procedimiento.** ***Nota:*** *Si el requerimiento es tercerizado se presentan los formatos PA0302-F14: "Acta de capacitación" y*	Formato PA0302-F10: *"Acta de conformidad general"* Correo Institucional Catálogo de servicios de TI	Coordinador/a de desarrollo de sistemas	OTI

Oefa Organismo de Evaluación y Fiscalización Ambiental	**FICHA DE PROCEDIMIENTO**	Código: PA0302 Versión: ***03*** Fecha: 22/12/2022

ACTIVIDADES				**EJECUTOR**	
N°	**ACTIVIDAD**	**DESCRIPCIÓN**	**REGISTROS**	**RESPONSABLE**	**UNIDAD DE ORGANIZACIÓN**
		PA0301-F10: "Acta de conformidad general".			

DOCUMENTOS QUE SE GENERAN:

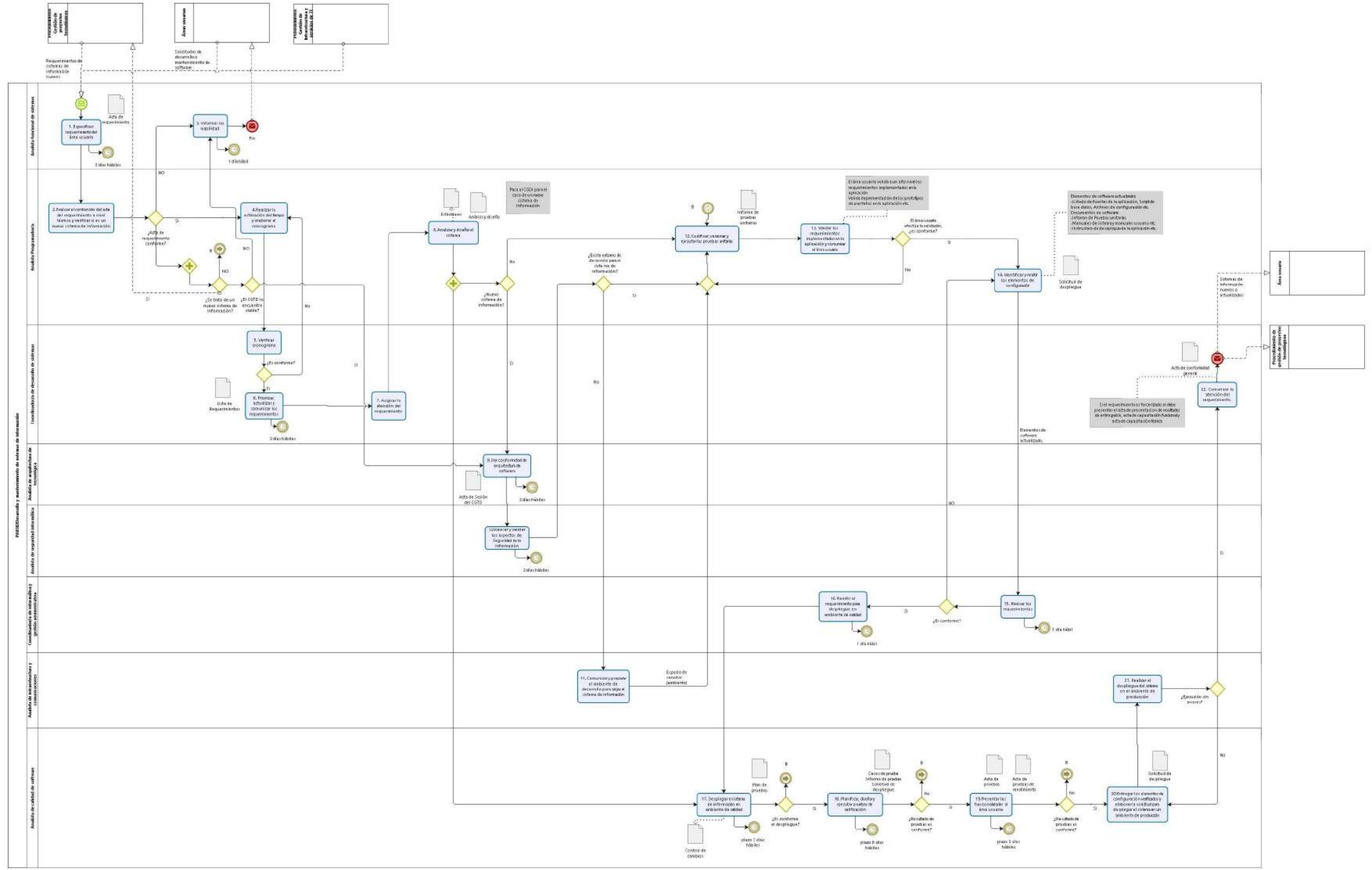
- Formato PA0302-F01: *"Acta de requerimiento*"
- Formato PA0302-F02: *"Documento de Análisis y diseño*"
- Formato PA0302-F03: *"Informe de Pruebas Unitarias*"
- Formato PA0302-F04: *"Solicitud de despliegue*"
- Formato PA0302-F05: *"Control de* ***Cambios***"
- Formato PA0302-F06: *"Plan de Pruebas"*
- Formato PA0302-F07: *"Acta de Pruebas"*
- Formato PA0302-F08: *"Casos de Prueba*"
- Formato PA0302-F09: "*Informe de pruebas"*
- Formato PA0302-F10: "*Acta de Conformidad General"*
- Formato PA0302-F11: "*Manual de Usuario"*
- Formato PA0302-F12: "*Manual del Sistema y Configuración"*
- Formato PA0302-F13: *"Acta de pruebas de rendimiento"*
- Formato PA0302-F14: *"Acta de capacitación"*
- ***Formato PA0302-F15: "Control de Requerimientos"***
- Formato PA0301-F10: *"Acta de reunión"*

ANEXOS DEL PROCEDIMIENTO:

Instructivo I-OTI-PA0302-1: *"Instructivo de Desarrollo de Software".*
Anexo N° 1: Estándar de base de datos.
Anexo N° 2: Estándar de infraestructura en la nube.
Anexo N° 3: Estándar de seguridad de desarrollo seguro.
Anexo N° 4: Estándar para aplicaciones desplegadas en jboss.
Anexo N° 5: Estándar para aplicaciones desplegadas en Kubernetes.

PROCESO RELACIONADO

PA03 - Tecnologías de la información


Mantenimiento Gestión de peticiones
Área usuaria
Mantenimiento Gestión de incidencias y soporte de TI
Requerimientos de sistemas de información (nuevos)
Solicitudes de desarrollo o mantenimiento de software
PM182Desarrollo y mantenimiento de sistemas de información
Analista funcional de sistemas
Analista Programadora
Coordinadora de desarrollo de sistemas
Analista de seguridad de la información y ciberseguridad
Analista de arquitectura de tecnología
Coordinadora de infraestructura y gestión cloud
Analista de infraestructura y comunicaciones
Analista de calidad de software
1. Especificar requerimiento del área usuaria
Acta de requerimiento
3 días hábiles
2.Evaluar el contenido del acta del requerimiento a nivel técnico y verificar si es un nuevo sistema de información
¿Acta de requerimiento conforme?
NO
SI
3. Informar no viabilidad
Fin
1 día hábil
4.Realizar la estimación del tiempo y elaborar el cronograma
¿Se trata de un nuevo sistema de información?
¿El CGTD lo encuentra viable?
5. Verificar cronograma
¿Es conforme?
Lista de Requerimientos
6. Priorizar, actualizar y comunicar los requerimientos
2 días hábiles
7. Asignar la atención del requerimiento
Para el CGDI para el caso de un nuevo sistema de información
Estándares
Análisis y diseño
8. Analizar y diseñar el sistema
¿Nuevo sistema de información?
¿Existe entorno de desarrollo para el sistema de información?
9. Dar conformidad de arquitectura de software
Acta de Sesión del CGTD
2 días hábiles
10.Obtener y validar los aspectos de Seguridad de la Información
2 días hábiles
11. Comunicar y preparar el ambiente de desarrollo para alojar el sistema de información
Espacio de servidor (ambiente)
12. Codificar, versionar y ejecutar las pruebas unitarias
Informe de pruebas unitarias
13. Validar los requerimientos implementados en la aplicación y comunicar al área usuaria
El área usuaria valida si los requerimientos implementados en la aplicación
Valida la presentación de los prototipos de pantallas en la aplicación etc.
El área usuaria verifica la validación, ¿Es conforme?
14. Identificar y mantener los elementos de configuración
Elementos de software actualizados
-Listado de fuentes de la aplicación, Scripts base datos, Archivos de configuración etc.
Documentos de software:
-Informe de Pruebas Unitarias.
-Manuales de sistemas y manuales usuario etc.
-Instructivo de despliegue de la aplicación etc.
Solicitud de despliegue
Elementos de software actualizado
15. Revisar los requerimientos
1 día hábil
¿Es conforme?
16. Revisar el requerimiento para desplegar en ambiente de calidad
1 día hábil
17. Desplegar el sistema de información en ambiente de calidad
Plan de pruebas
Control de cambios
plazo 1 días hábiles
¿Es conforme el despliegue?
18. Planificar, diseñar y ejecutar pruebas de ratificación
Casos de prueba Informe de pruebas Solicitud de despliegue
plazo 5 días hábiles
¿Resultado de pruebas es conforme?
19.Presentar las funcionalidades al área usuaria
Acta de pruebas
Acta de pruebas de rendimiento
plazo 3 días hábiles
¿Resultado de pruebas es conforme?
20.Entregar los elementos de configuración verificados y elaborados y solicitar que se despliegue el sistema en un ambiente de producción
Solicitud de despliegue
21. Realizar el despliegue del sistema en el ambiente de producción
¿Ejecución sin errores?
Acta de conformidad general
22. Comunicar la atención del requerimiento.
Si el requerimiento es tercerizado se debe presentar el acta de conformidad de resultados de entregables, acta de capacitación funcional y acta de capacitación técnica
Sistemas de información nuevos o actualizados
Área usuaria
Proceso Gestión de pruebas de tecnologías
SI
NO

Oefa Organismo de Evaluación y Fiscalización Ambiental	**ACTA DE REQUERIMIENTO**

Versión	
Sistema o Aplicativo	
Código del requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha	

Tipo de requerimiento	( ) Desarrollo ( ) Mantenimiento
NECESIDAD DEL ÁREA USUARIA: *Describir las limitaciones de la situación actual y las razones por las cuales se emprende el proyecto.*	
OBJETIVOS DEL PROYECTO: *Definir con claridad los objetivos del proyecto.*	

REQUISITOS FUNCIONALES Y NO FUNCIONALES

PRIORIDAD	**CRITICIDAD**	**REQUISITO**	
		N°	**DESCRIPCIÓN**

FIRMA Y SELLO	
Firma y Sello: Jefe/a del Área Usuaria Fecha / Hora:	Firma y Sello: Usuario/a Final (Líder de Área Usuaria) Fecha / Hora:

Oefa Organismo de Evaluación y Fiscalización Ambiental	**DOCUMENTO DE ANÁLISIS Y DISEÑO**

NOMBRE DEL SISTEMA DE INFORMACIÓN

ANÁLISIS Y DISEÑO

TABLA DE CONTENIDO

Pág.

SECCIÓN 1. ESPECIFICACIONES DE CASO DE USO.

1.1 Nombre

Título del caso de uso.

1.2 Actor

Actor del caso de uso.

1.3 Breve Descripción

Descripción simple de las actividades o pasos principales que realiza el Caso de Uso.

1.4 Pre-condiciones

Definen el estado en el que el sistema debe estar o las condiciones que deben ser satisfechas antes de que el caso de uso sea ejecutado.

1.5 Flujo de eventos

Flujo básico

Ruta tomada por la mayoría de usuarios la mayor parte del tiempo. Este flujo le permitirá al actor alcanzar su meta satisfactoriamente.

Flujos alternativos

Flujos opcionales, flujos de excepción.

Subflujos

Secciones que ejecutan comportamientos importantes pero que no son esenciales para el entendimiento del flujo básico de eventos.

Escenarios

Un escenario es una instancia (ocurrencia específica) de un caso de uso.

1.6 Post-condiciones

Definen el estado en el que el sistema se encuentra una vez que el caso de uso ha sido ejecutado.

1.7 Requerimientos especiales

Requerimientos no funcionales.

SECCIÓN 2. PANTALLAS Y REPORTES DEL CASO DE USO

1.8 Flujo general de navegación

Diagrama que muestra la navegación principal.

1.9 Descripción de Subsistemas / Módulos / Sub. - Módulos

Módulos

Módulo-01

Descripción del módulo 01.

Módulo-02

Descripción del módulo 02.

Pantallas

Pantalla/Reporte P001

a) **Descripción**

Ingresar descripción de la Pantalla/ Reporte P001.

b) **Descripción de campos**

Ingresar descripción de campos de la Pantalla/ Reporte P001.

c) **Descripción de botones**

Ingresar descripción de botones de la Pantalla/ Reporte P001.

Pantalla/Reporte P002

a) **Descripción**

Ingresar descripción de la Pantalla/ Reporte P002.

b) **Descripción de campos**

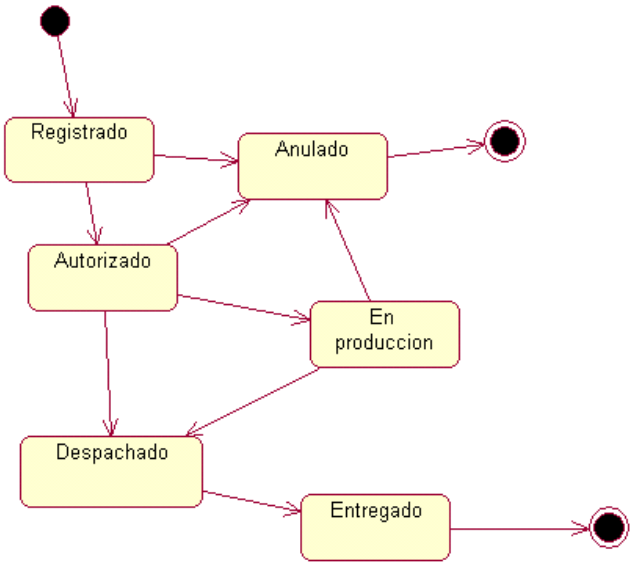
Ingresar descripción de campos de la Pantalla/ Reporte P002.

c) **Descripción de botones**

Ingresar descripción de botones de la Pantalla/ Reporte P002.

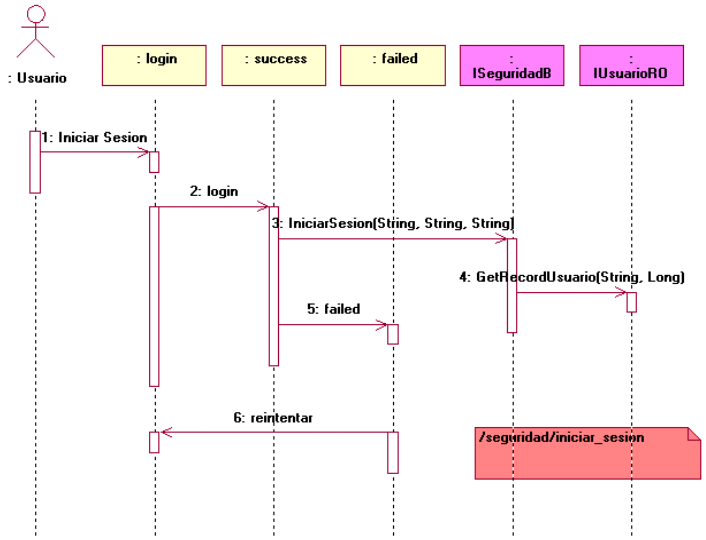
SECCIÓN 3. DIAGRAMAS DE ESTADO DEL CASO DE USO

Diagramas de estado asociados a este caso de uso. Ejemplo:

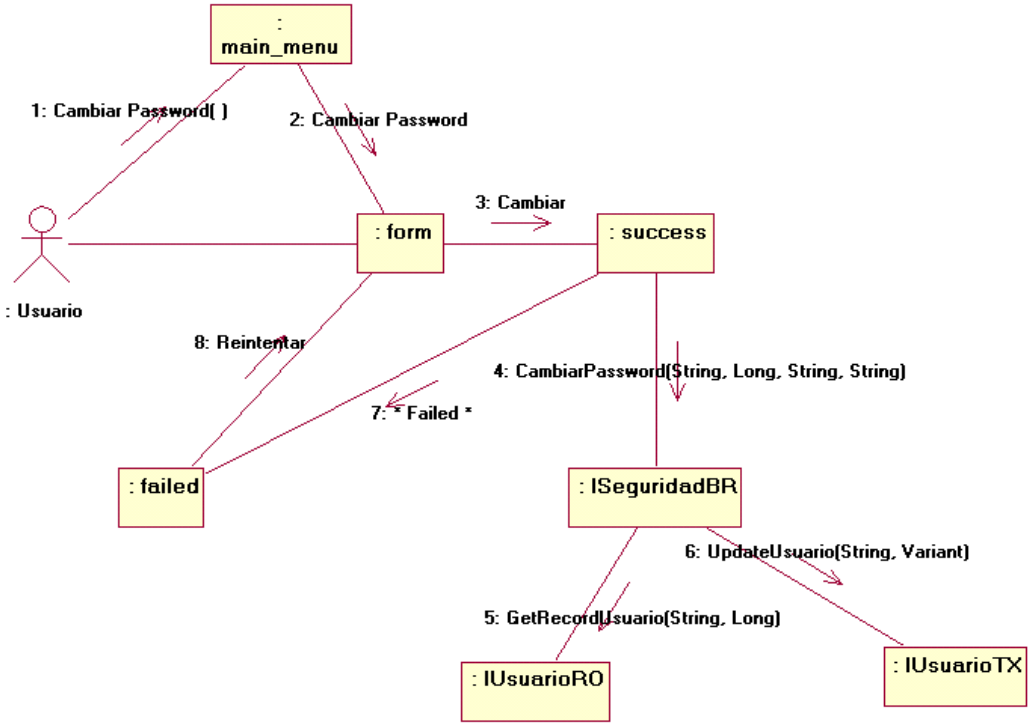


SECCIÓN 4. DIAGRAMAS DE SECUENCIA DEL CASO DE USO

Esta sección es opcional. Gráfico que muestra diagramas de Secuencia generales (que no está asociado a un caso de uso en particular) del sistema de software que se debe construir, muestra la interacción de componentes. Se sugiere un diagrama de secuencia o colaboración de UML. Ejemplo:



Diagramas de colaboración asociados a este caso de uso. Ejemplo:



SECCIÓN 4. DISEÑO DE COMPONENTES/PROGRAMAS

Inventario de componentes/programas

Lista de componentes/programas:

- Que deben ser modificados (MODIFICAR). Indicar si se usan otros componentes de referencia como ejemplo para la modificación.
- Que deben ser creados (CREAR) Indicar si se usan otros componentes/programas de referencia como ejemplo para crear el nuevo componente/programa.
- A reusar (REUSAR)
- Que serán elaborados por algún proveedor (TERCERIZAR) o
- Que serán adquiridos a algún proveedor (COMPRADO).

Indicar de manera explícita si no hay algún origen.

Id	Nombre del componente/programa	Tipo	Funcionalidad requerida	Origen	Responsable

Componentes/programas sensibles

Por sus características, los siguientes componentes/programas no deben poder ser accesados por todo el equipo de desarrollo, si no sólo por personas específicas.

Id	Nombre del componente o programa sensible	Otorgar acceso a

Descripción de cada componente/programa

Nombre / Clase:	
Librería:	
Descripción / Objetivo:	
Entradas (Input):	
Lógica / Descripción de Métodos/subrutinas:	
1. **2.**	
Salidas (Output):	
Interfases que usa:	

SECCIÓN 5. DISEÑO DE INTERFASES

Interfases externas

Identificación de interfases externas

Id	Origen	Destino	Tipo	Referencia a estructura
01				
02				

Diseño de estructura 1.

Diseño de estructura N.

Interfases internas

Identificación de interfases internas

Id	Origen	Destino	Tipo	Referencia a estructura
01				
02				

Diseño de estructura 1.

Agregar las estructuras necesarias.

Diseño de estructura N.

Agregar las estructuras necesarias.

SECCIÓN 6. DIAGRAMA DE CLASES (DISTRIBUIDOS)

Gráfico(s) que muestra cómo los componentes interactúan. Se sugiere, lo siguiente:

- Diagrama de clases
- Diagrama de objetos
- Diagrama de secuencia o
- Diagrama de colaboración

Usar estándar UML. Se puede incluir descripciones para aclarar detalles de las interacciones.
Se puede incluir una referencia al archivo y ubicación donde está(n) el(los) diagrama(s) si se usa alguna herramienta. También se pueden pegar los diagramas para mayor claridad de este documento si no todos tienen acceso a la herramienta.

Diagrama de Clases

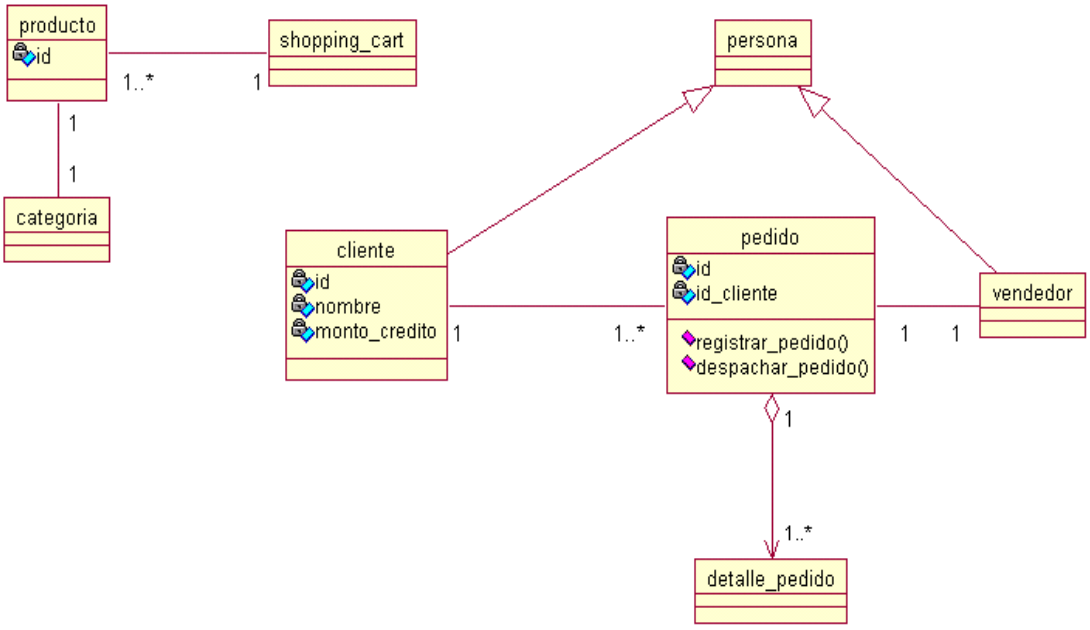


Diagrama de objetos

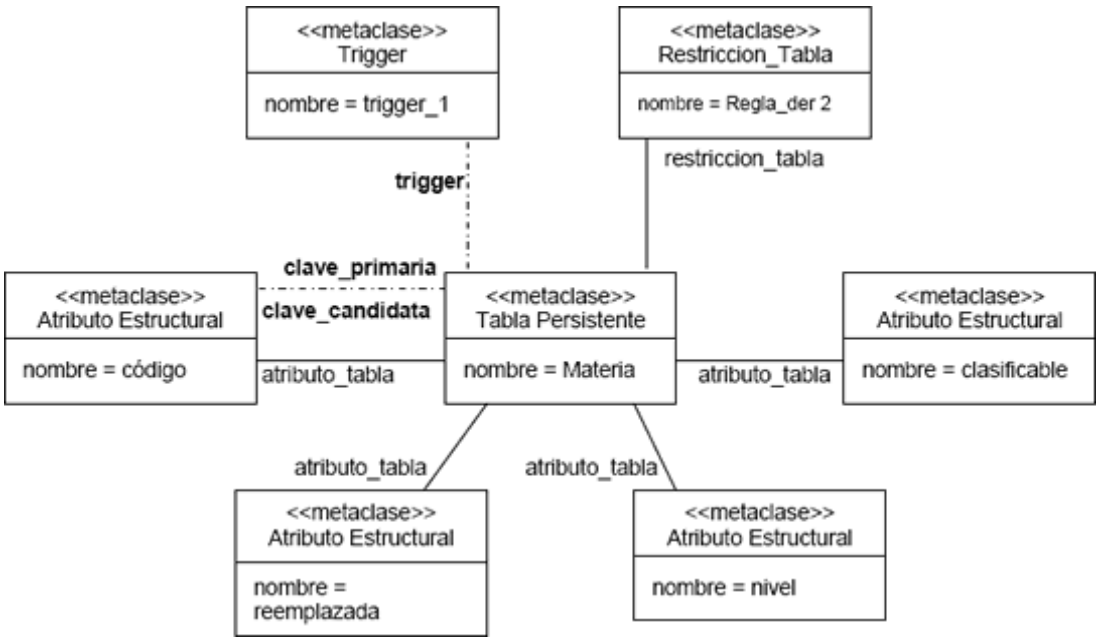


Diagrama de Secuencia

Gráfico que muestra cómo los componentes interactúan. Se sugiere un diagrama de secuencia o colaboración de UML. Se puede incluir descripciones para aclarar detalles de las interacciones.

.

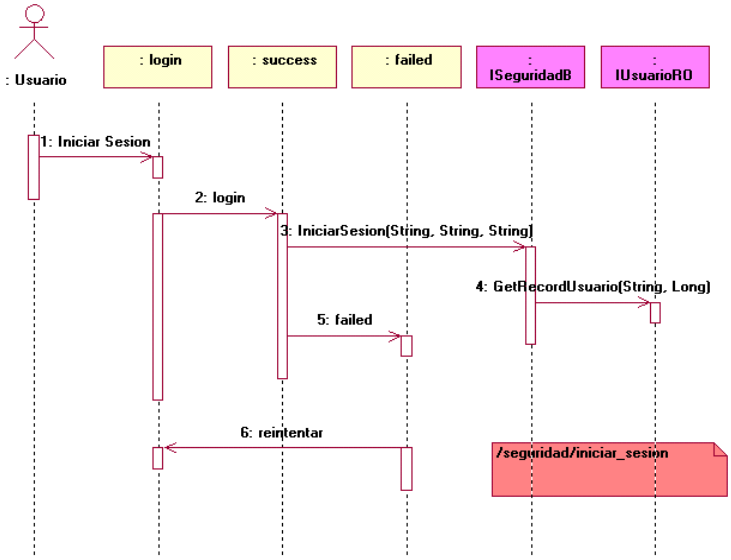
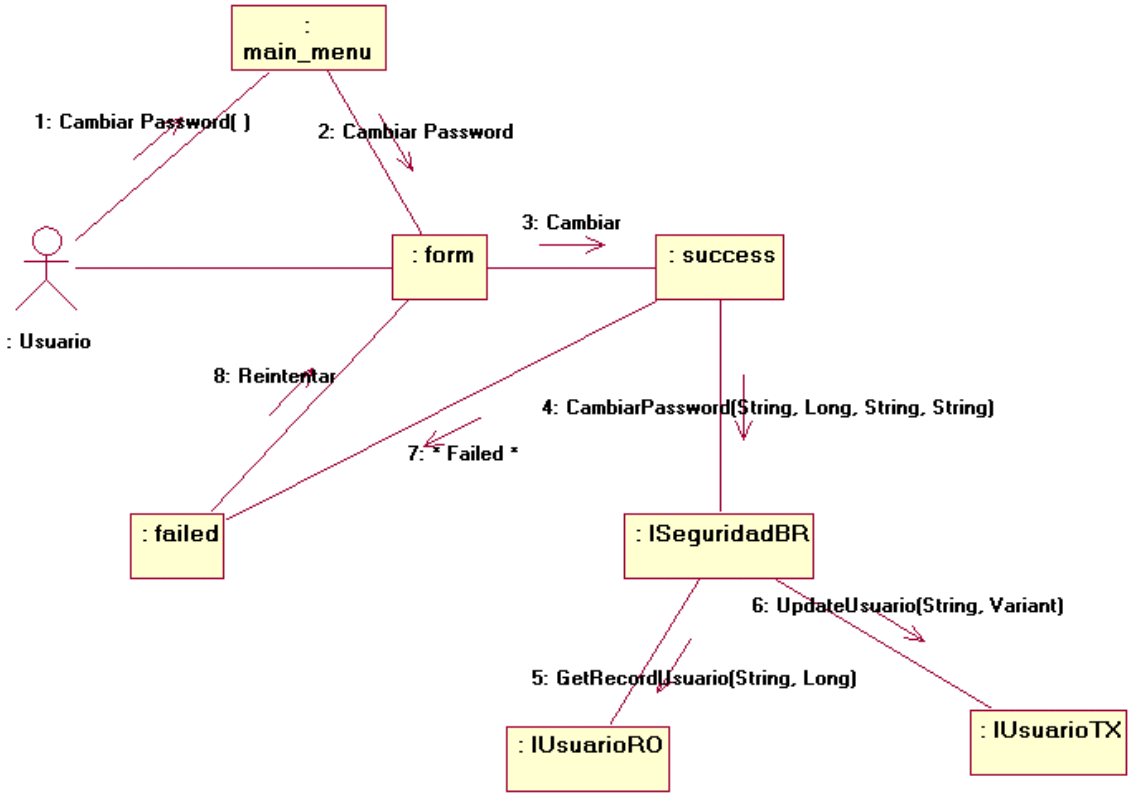


Diagrama de Colaboración



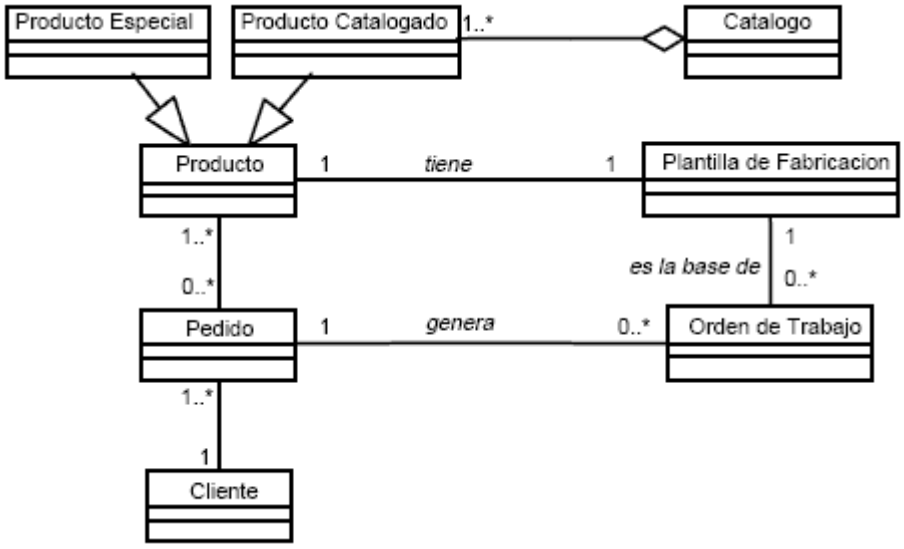
.

SECCIÓN 7. DISEÑO DE BASE DE DATOS (DISTRIBUIDOS)

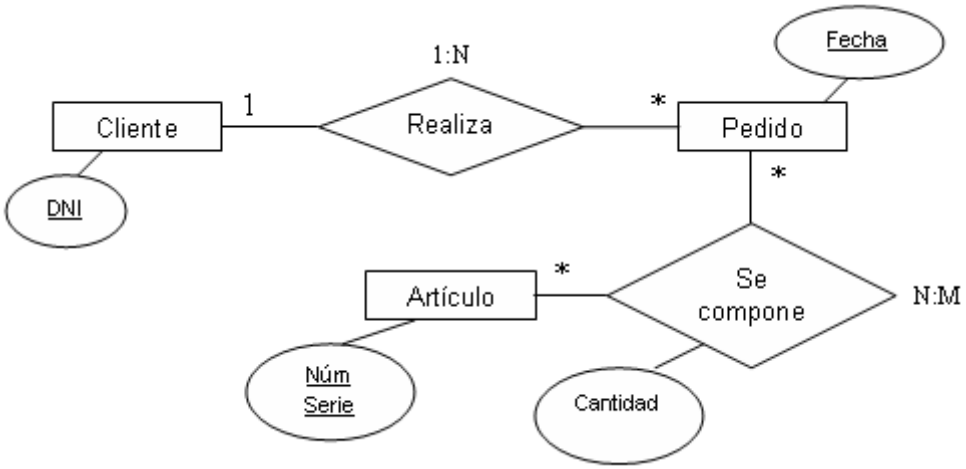
Modelo conceptual o lógico

Diagrama(s) entidad relación. Se puede incluir una referencia al archivo y ubicación donde está(n) el(los) diagrama(s) si se usa alguna herramienta. También se pueden pegar los diagramas para mayor claridad de este documento si no todos tienen acceso a la herramienta.

Ejemplo 1.



Ejemplo de Diagrama E-R:



Modelo físico

Descripción de bases de datos

Descripción de las bases de datos físicas. Se puede incluir una referencia al archivo y ubicación donde está(n) el(los) diagrama(s) si se usa alguna herramienta.

Descripción de tablas

Descripción de las tablas en cada base de datos. Se puede incluir una referencia al archivo y ubicación donde está(n) el(los) diagrama(s) si se usa alguna herramienta. También se pueden pegar los reportes que genera la herramienta para mayor claridad de este documento si no todos tienen acceso a la herramienta.

Descripción de archivos

Descripción de archivos (no son parte de base de datos, por ejemplo, archivos CICS, archivos de configuración, entre otros).

Diccionario de datos

Puede ser un anexo o se puede incluir una referencia al archivo y ubicación donde está(n) el(los) diagrama(s) si se usa alguna herramienta. También se pueden pegar los reportes que genera la herramienta para mayor claridad de este documento si no todos tienen acceso a la herramienta.

Id	Tabla	Campo	Tipo de dato	Descripción
01				
02				
03				
04				
05				

Oefa Organismo de Evaluación y Fiscalización Ambiental	**INFORME DE PRUEBAS UNITARIAS**

Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha del documento	

1. OBJETIVO

Proveer la información necesaria para planear y controlar los esfuerzos de pruebas de un proyecto tecnológico o iteración específicos.

2. CRITERIOS DE ENTRADA Y SALIDA

2.1. Criterio de Ejecución de Pruebas

2.2. Criterio de Terminación del Plan de Pruebas

2.3. Criterio de Suspensión del Plan de Pruebas

3. ESCENARIOS PROPUESTOS DE PRUEBA

Información General	
Identificador de caso de uso:	
Nombre de caso de uso:	
Descripción Prueba:	
Responsable:	
Prerrequisitos	
Descripción de Casos de Prueba	
Caso:	
Servicio (s) Probados:	
Instrucciones de Prueba 1. 2. 3.	
Criterios de Aceptación 1. 2.	

4. Resumen

Errores totales	Cerrado	Abierto	Suspendido	Total de Errores
Errores Graves				
Errores Medios				
Errores Leves				
Total				

Oefa Organismo de Evaluación y Fiscalización Ambiental	**SOLICITUD DE DESPLIEGUE**

Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha del documento	

Ambiente	:	____ Calidad ____ Producción	**Prioridad:**	___ Urgente ___ Media ___ Baja
Lenguajes de Programación	:	___ Java () ___ Power Builder ___ Otros: _____	**Servidor:**	___ internal.oefa.gob.pe ___ publico.oefa.gob.pe ___ apps.oefa.gob.pe ___ Otros: _____
Servidor de Aplicaciones	:	___ Tomcat () ___ WebLogic ___ JBOSS ___ Otros: _____	**Versión del sistema:**	
Motor de Base de datos	:	___ Oracle ___ SQL Server ___ MySQL ___ Otros _____	**Nombre de la Base de datos**	

1. DESCRIPCIÓN DEL DESPLIEGUE:

1.1. DESCRIPCIÓN
1.2. PASOS DE EJECUCIÓN
Copia de Seguridad Especificar a qué objetos se realizará el backup.
Base de Datos Especificar esquema donde se ejecutarán los scripts, package, procedure, entre otros elementos de base de datos. **Aplicación** Especificar los pasos a seguir como cambio de cadena de conexión a base de datos, conexión al Sistema de Seguridad u otras configuraciones.
Pruebas de Aceptación Se especificará solo para el área de calidad, en la cual indicará el/la usuario/a con quien realizó las pruebas o quien solicitó dicho requerimiento (esta información debe ser brindada por el equipo de desarrollo).
Marcha Atrás Especificar los pasos a seguir para restaurar el ambiente.

2. RELACIÓN DE OBJETOS

	Nombre	Tamaño Objeto (Kb)	Tipo	Servidor Destino
1	Se debe listar los nombres de scripts con la siguiente estructura: 01_Esquema_Acción_Tabla o paquete			

3. FIRMAS PARA EL PASE A PRODUCCIÓN

Firma y Sello: **Analista de gestión de proyectos de TI** Fecha / Hora:	Firma y Sello: **Analista de arquitectura tecnológica** Fecha / Hora:	Firma y Sello: **Coordinador/a de infraestructura y comunicaciones** Fecha / Hora:

CONTROL DE CAMBIOS

SOLICITUD						APLICACION			CORRELATIVOS			RESPONSABLES					FECHA							OBSERVACIONES
Petición	Año	Tipo	Tipo REQ	Número	Descripción	Siglas	Tipo	Versión	APFU	ACAP	SPRO	Desarrollo / Atención	Control de Calidad / Atención	Solicitud a Producción / Atención	Ejecución Producción	Nro. Iteración	Solicitud a Calidad / Atención	Fecha de asignación a Analista de Calidad	Fecha inicio de Atención Analista de Calidad	Fecha de devolución a desarrollo	Solicitud a Producción / Atención	Ejecución Producción / Atención	ESTADO	

PLAN DE PRUEBAS

TABLA DE CONTENIDO

Pág.

SECCIÓN 1. INTRODUCCIÓN

1.1 Objetivo

Dar la visión general de la fase de calificación, detallando los logros que queremos alcanzar en cada una de las funcionalidades del sistema de información a ser probado y los medios a utilizar para dicho fin.

1.2 Componentes Involucrados

Detallar los archivos, procesos, reportes y demás puntos que interrelacionan a todos los aplicativos de desarrollo, indicando claramente (de ser necesario), si la interfase es nueva, modificada o ya utilizada.
Si un proyecto realiza sucesivos pases a producción, debe existir y actualizarse un único plan de calificación.
En la columna donde se indica al *"Responsable"*, pueden ser: Grupo Calificación, Grupo Desarrollo o Factoría Desarrollo.

Tabla N° 1: Componentes Involucrados

IdAC	Aplicativo - Componente / Interfaz	Plataforma	Nuevo / Modificado	Responsable
1				
2				
3				
4				

1.3 Hitos de Pruebas

Indicar si algún componente tiene un pase a producción diferente por ser independiente.

La columna donde se indica *"Lista"* de IdAC sirve para que en las actividades EJECUCIÓN DE PRUEBAS quede establecido cuáles componentes se van a probar en dicha actividad (columna IdAC de la **Tabla N° 1**).

La Fecha de Pase a Producción sirve para indicar, según este plan, cuándo podría realizarse el pase a producción de la iteración que se está planificando.

Tabla N° 2: Hitos y Componentes pase a Producción

Id	Tarea	Lista de IdAC (Para saber Aplicativos - Componentes / Interfaces a probar)	Fecha de Inicio (DD/MM/AAAA)	Fecha de Fin (DD/MM/AAAA)	Recurso
1	ANÁLISIS DE REQUERIMIENTO				
2	PLANIFICACIÓN DE ACTIVIDADES				

Id	Tarea	Lista de IdAC (Para saber Aplicativos - Componentes / Interfaces a probar)	Fecha de Inicio (DD/MM/AAAA)	Fecha de Fin (DD/MM/AAAA)	Recurso
3	PREPARACIÓN DE TEST CASES				
4	EJECUCIÓN DE PRUEBAS				
5	EJECUCIÓN DE PRUEBAS	2,3			
6	EJECUCIÓN DE PRUEBAS	1,4			
7	EJECUCIÓN DE PRUEBAS	2,3,5			
8	REWORK EJECUCIÓN	1,2,3,4,5			
9	REPORTE FINAL				
Fecha de Pase a Producción (estimada):					

1.4 Restricciones y/o supuestos

Consignar en esta sección las restricciones y supuestos de las actividades necesarias para alcanzar los objetivos (costo, calendario, personas, objetivos de calidad):

Tabla N° 3: Restricciones y supuestos

N°	Restricciones y/o Supuestos
1	
2	
3	
4	

1.5 Criterios de Entrada y Salida

Indicar los prerrequisitos (criterios de entrada) y post-condiciones (criterios de salida) específicos (que no están en la descripción del proceso) para poder iniciar y dar por terminado este plan. Se pueden redactar aquí o indicar dónde están documentados.

SECCIÓN 2. PRUEBAS DE CALIFICACIÓN

2.1 Alcance

Delimitar en forma precisa desde el punto de vista funcional y/o técnico qué elementos de las aplicaciones serán sujetos a prueba. Si por algún tipo de consideración se dejará de probar alguno de los alcances, esto debe ser indicado explícitamente en este punto, indicando el motivo. Tomar en cuenta los siguientes criterios:

(i) El alcance de las pruebas debe ser el 100% de los requerimientos funcionales y no funcionales. Debe haber una trazabilidad explícita.
(ii) Cuando el alcance incluya modificaciones, eliminaciones o adiciones a uno o varios aplicativos existentes, debe evaluarse el alcance para las pruebas de regresión y para las pruebas de integración, que pueden ir más allá de los requerimientos funcionales y no funcionales.

2.2 Alcance Específico por sistema

Detallar claramente, los alcances de las pruebas por aplicativo, módulo y/o interfaz, que cubrirán la calificación.

2.3 Fuera del alcance

Detallar claramente, los aplicativos, módulos, funcionalidades, procesos y/o interfaces que se encuentran fuera del alcance de las pruebas.

2.4 Estrategia de pruebas

Indicar el orden en el que se realizan las pruebas, definir cuántos ciclos de pruebas y cuáles casos de prueba ejecutar en cada ciclo de prueba, las herramientas que se utilizarán, tipos de pruebas a simular y los diferentes tipos de técnicas de pruebas a usar.
Detallar por funciones desde los aplicativos, módulos, procesos y/o interfaces lo siguiente:

- Tipos de prueba a simular
- Procedimientos para las pruebas (secuencia de las pruebas)
- Poblamiento de tablas y data histórica- Indicar cuál será la modalidad de poblamiento de tablas y el schedule de mantenimiento de los datos.
- Identificar y diseñar en cuáles componentes enfocar más el testing en base a los siguientes criterios:
  - Uso de la funcionalidad
  - Impacto en el negocio (en caso de falla)
  - Atributos no funcionales
  - Pruebas especiales (como pruebas de regresión)

Tabla N° 4: Estrategia de Pruebas

ID	Ítems de la Lista de Requerimiento	Funcionalidad a probar	Escenario	Uso de la funcionalidad	Impacto si falla	Total casos de prueba
1						
2						
3						
4						

2.5 Riesgos del Proceso de Calificación

Identificar los riesgos que podrían poner en peligro la fase de Calificación.

Tabla N° 5: Riesgos

N°	**Evento** (riesgo identificado)	**Mitigación** (acciones para minimizar prob. del evento)	**Contingencia** (acciones si sucede el evento)
1			
2			
3			
…			

2.6 Casos de prueba y escenarios

En esta sección se especifica en forma detallada los casos de prueba y los resultados esperados para cada uno de ellos.

Para la elaboración de estos, se detallará la relación de módulos, procesos y funciones por aplicativo cumpliendo los pasos siguientes:

- Análisis: Tener una visión integral en cuanto a funcionalidad e impacto en el aplicativo a certificar.
- Desarrollo de los casos de prueba: Se elaborarán los casos de prueba, teniendo en cuenta:
  - Estructura modular del aplicativo.
  - Descripción de los escenarios de prueba.
  - Matriz de escenarios/casos de prueba.

El/La Analista de Calificación:

- Ingresa y/o actualizará los casos de prueba en el repositorio para casos de prueba
- Extrae los casos de prueba del repositorio para casos de prueba

En esta sección indicar el nombre del archivo o herramienta donde están los casos de prueba de este plan de calificación.

2.7 Ambiente de pruebas

Equivale a la sección *"4. Ambiente de Pruebas"*.

2.8 Requerimientos de Hardware y Software

Hardware y software necesario para el funcionamiento del aplicativo basándose en las definiciones del plan de calificación y que simula el ambiente de producción, especificando si este debe convivir con otros en los mismos equipos, servidores, etc. Incluye requerimientos para los puestos de trabajo del tester.

Tabla N° 6: Requerimientos de Hardware y Software

Id	Puesto de trabajo	Requisitos HW	Sistema Operativo	Requisitos SW
1				
2				
3				
…				

2.9 Requerimientos de accesos a sistemas

Se especifica qué permisos y perfiles de usuario se requerirá por sistema, para validación.

2.10 Requerimientos de datos de prueba

Indicar archivos o bases de datos, así como las autorizaciones requeridas para realizar las pruebas programadas.

2.11 Requerimientos para contingencias de la prueba

Indicar las consideraciones a tener para la ejecución de los casos de prueba, en lo referente a backups y restore.

2.12 Aplicaciones

Identificar las aplicaciones a instalar.

Tabla N° 7: Aplicaciones

Id	Aplicación	Versión	Ambiente Calificación	Ambiente UAT/ Preproducción	Ambiente Producción
1					
2					
3					
…					

2.13 Datos

Describir qué tipos de datos hay que conseguir.

2.14 Dispositivos externos

Identificar los dispositivos externos necesarios para las pruebas.

2.15 Pruebas especiales

Describir los criterios que sustenten la ejecución de los tipos de prueba:

2.16 Pruebas Integrales

Indicar la metodología para llevarla a cabo (flujo de los casos de prueba y su seguimiento).

2.17 Pruebas de Stress/Concurrencia

Describir en forma detallada la cantidad de concurrencias a simular, tiempo de respuesta esperado, etc. En el cuadro adjunto, describir el plan de pruebas de stress.

Tabla N° 8: Plan de pruebas de Stress

Id	Prueba	Fecha Inicio (dd/mm/aaaa)	Fecha Fin (dd/mm/aaaa)	Hora	Responsable	Resultados	Recomendaciones
1							
2							
3							
…							

2.18 Pruebas de Carga/Volumen

Indicar el tamaño o volúmenes de archivos a cargar / procesar y cuáles son los referentes para comparar resultados.

2.19 Pruebas de Ancho de Banda

Para aplicativos cuyo medio de acceso sea Internet / Intranet y la cantidad de información que se traslada sea considerable y haya la posibilidad de congestión en el tráfico; sumado a cantidad de usuarios; justifica la ejecución de esta prueba.

2.20 Pruebas de Seguridad de Información

Consignar aquí las pruebas de Seguridad de Información que se realizarán.

2.21 Pruebas de Instaladores

Si hay instaladores, consignar aquí las pruebas que se harán a los instaladores.

2.22 Otras Pruebas

Agregar todas las pruebas que sean necesarias.

2.23 Documentación de respaldo

Evidencias.
Indicar el tipo y el tiempo que se guardará de acuerdo a los procesos.

Conformidad de pruebas.
Especificar si se requiere aprobación del usuario de ser necesario.

Reporte de estado final de pruebas
(Adjuntar)

2.24 Plan de Conformidad final de pruebas en el ambiente de producción

Especificar si es necesaria la ratificación en producción o no. El Plan de Calificación se usa durante las pruebas finales de conformidad en producción, donde se debe indicar los casos de prueba a ser ejecutados como parte de las pruebas finales de conformidad en ambiente de producción.

(*) **Tipos de pruebas en producción**:

a. Dress rehearsal: son pruebas en vivo, sobre casos que no se pueden probar en ambiente de calificación. Usualmente para detectar errores y se revierten.

b. Marcha blanca: cuando es algo nuevo y se define un tiempo relativamente largo, pueden ser de 2 semanas a un mes. El usuario ejecuta algunas funcionalidades y algunos casos en producción, luego de lo cual se restaura y re-instala.

c. Paralelo: convivencia completa, se hace comparación y cuadre.

d. Piloto: pruebas focalizadas, ciertos segmentos, clientes y/o zonas geográficas. Hay capacitaciones involucradas, son cambios fuertes.

e. Pruebas en oficina virtual: cuando es un tema transaccional.

f. Producción controlada (Big family): es un piloto con un producto nuevo, se proporciona el nuevo producto o servicio a un segmento focalizado.

2.25 Tiempos y recursos requeridos (resumen de cronograma)

Esta sección muestra un resumen del esfuerzo y requerimientos de recursos tanto internos como externos, con el fin de programar la preparación y apoyo a la fase de calificación.

Esta sección resume información del cronograma de la fase de Calificación:

- Estimación del plazo para la calificación.
- Participantes en las pruebas. Recursos necesarios para la ejecución de las pruebas indicando si se requerirá la participación de un grupo de usuarios como apoyo en la calificación y entrenamiento necesario o se subcontratará recursos de proveedor o se contratará la calificación.
- Definición del apoyo necesario por parte de los técnicos de desarrollo.

Adicionalmente se debe indicar por cada recurso, las funciones que le han sido asignadas para su validación, detallando por cada función, la cantidad de casos de prueba de su responsabilidad y las horas estimadas para su ejecución-ver cuadro adjunto:

Tabla N° 9: Cuadro resumen de casos y horas por recurso de calificación

			Estimación preliminar	
Id	**Nombre del Recurso - Rol - Dedicación**	**Función a probar (Área - Aplicación)**	**# Casos Planificados**	**# Horas Estimadas**
1				
2				
3				
…				

			Estimación preliminar	
Id	**Nombre del Recurso - Rol - Dedicación**	**Función a probar (Área - Aplicación)**	**# Casos Planificados**	**# Horas Estimadas**
		Total General	Ingresar el número total de casos planificados.	Ingresar el número total de horas estimadas

Tabla N° 10: Cronograma

Tarea	Recurso	Duración
1. Checklist documentación		
2. Análisis de Requerimiento		
3. Estimación Inicial		
4. Planificación de Actividades		
5. Creación del Plan de Calificación		
6. Preparación de Casos de Prueba		
7. Ejecución de Pruebas		
8. Reporte Final		
Duración total		

Oefa Organismo de Evaluación y Fiscalización Ambiental	**ACTA DE PRUEBAS**

Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha	

I. RESUMEN EJECUTIVO

Tipo de solicitud	( ) Desarrollo ( ) Mantenimiento
Área solicitante	
Responsable(s) de las Pruebas	
Lugar	
Fecha - Hora de prueba	

II. RESULTADOS DE LAS PRUEBAS FUNCIONALES

N° REQ	Caso de prueba (CP) Asociado	Descripción	Conforme (SI/NO)
00x	CP00x	Breve descripción funcional del requerimiento X a probar	

III. RESULTADOS DE LAS PRUEBAS NO FUNCIONALES

N° REQ	Caso de prueba (CP) Asociado	Descripción	Conforme (SI/NO)
00x	CP00x	Breve descripción del requerimiento X a probar	

IV. OBSERVACIONES (*)

N° REQ	Descripción
00x	Breve descripción de la observación encontrada en el requerimiento o mantenimiento N° 00x.

(*) Solo se llenará en caso que existan observaciones, caso contrario indicar *"No aplica"*.

Firma: Analista de Calidad de Software Fecha / Hora:	Firma Usuario/a (Líder del Área Usuaria) Fecha / Hora:	Firma Jefe/a del Área Usuaria Fecha / Hora:

Oefa Organismo de Evaluación y Fiscalización Ambiental	**CASOS DE PRUEBA**

Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por	
Documento revisado por	
Fecha	

CASO DE PRUEBA 01					
Requerimiento o Caso de Uso asociado					
Objetivo					
Condición/es de Prueba					
Descripción					
Nombre de el/ la Analista de pruebas		**Cargo / Área**		**Fecha y Hora de Prueba**	

Paso	Instrucción	Data	Resultados Esperados	Resultados Reales	Estado	Tipo Error
1.	*Instrucción*	*Datos de ingreso, si fuera el caso*	*Resultados esperados*	*Resultados reales*		*Grave/Crítico/Medio/ Bajo*
2.						

Paso	Instrucción	Data	Resultados Esperados	Resultados Reales	Estado	Tipo Error
3.						
…						

Anexo de imágenes o pantallas del sistema o aplicativo esperado (en este punto se captura las pantallas o imágenes de cada instrucción a seguir en el caso de prueba en mención)

1. ***Imagen del paso 1***
2. ***Imagen del paso 2***

CASO DE PRUEBA 02					
Requerimiento o Caso de Uso asociado					
Objetivo					
Condición/es de Prueba					
Descripción					
Nombre de el/ la Analista de pruebas		**Cargo / Área**		**Fecha y Hora de Prueba**	

Paso	Instrucción	Data	Resultados Esperados	Resultados Reales	Estado	Tipo Error
1.	*Instrucción*	*Datos de ingreso, si fuera el caso*	*Resultados esperados*	*Resultados reales*		*Grave/Crítico/Medio/ Bajo*
2.						
3.						

Paso	Instrucción	Data	Resultados Esperados	Resultados Reales	Estado	Tipo Error
…						

Anexo de imágenes o pantallas del sistema o aplicativo esperado (en este punto se captura las pantallas o imágenes de cada instrucción a seguir en el caso de prueba en mención)

1. ***Imagen del paso 1***
2. ***Imagen del paso 2***

Observaciones	
1.	
2.	
3.	
…	

Sugerencias de Mejoras para el Sistema	
1.	Descripción de mejora
2.	
3.	
…	

ANALISTA DE CALIDAD DE SOFTWARE	**LIDER DE ÁREA USUARIA**
Nombre: Fecha:	Nombre: Fecha:

(*) Cuando las pruebas sean realizadas sólo por el analista de calidad, éste debe firmar solo en el cuadro izquierdo, no en ambos.

Oefa Organismo de Evaluación y Fiscalización Ambiental	**INFORME DE PRUEBAS**

1. RESUMEN DE PRUEBAS

N° de Informe				
Tipo de Requerimiento	( ) Desarrollo de Sistemas ( ) Mantenimiento	**N°**	N° de atención del ticket	
Responsable	**Construcción**		**Pruebas**	
Enfoque de Pruebas	( ) Caja Blanca - ( ) Caja Negra			
Fecha de Inicio y Fin de Pruebas				
Duración de Pruebas	( ) días - ( ) horas - ( ) minutos			
Navegador empleado	( ) Internet Explorer	( ) Google Chrome	( ) Firefox	
	( ) Opera	( ) Safari		
Cantidad de	**Errores / Incidencias**		**Oportunidades de Mejora**	

MÓDULO / OPCIÓN	**NIVEL DE COMPLEJIDAD**		
	Alto	**Medio**	**Leve**
Nombre del módulo			
	0	0	0
Total	**0**	**0**	**0**

2. DETALLE DE ERRORES / INCIDENCIAS

MÓDULO	**FORMULARIO EXTERNO**

Caso de Prueba				**Paso N°**	-
Opción					
Tipo de Error / Incidencia		**Nivel de Complejidad**		**Estado**	
Descripción					
1	Colocar imagen de la observación				

3. OPORTUNIDADES DE MEJORA

N°	Caso de Prueba asociado	Módulo	Oportunidad de mejora	Fecha

Oefa Organismo de Evaluación y Fiscalización Ambiental	**ACTA DE CONFORMIDAD GENERAL**

Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha	

CONFORMIDAD

Los/as firmantes dejan constancia que las funcionalidades del sistema…

OBSERVACIONES

La presente acta se firma:

Con Observaciones ☐
Sin Observaciones ☐

Observaciones:

LÍDER DEL ÁREA USUARIA	**ARQUITECTO/A DE TECNOLOGÌAS**
Nombre: Fecha:	Nombre: Fecha:
GESTOR/A DE PORTAFOLIO DE PROYECTOS	**SUPERVISOR/A DE PROYECTOS DE DESARROLLO DE SISTEMAS**
Nombre: Fecha:	Nombre: Fecha:

Oefa Organismo de Evaluación y Fiscalización Ambiental	**MANUAL DE USUARIO**

Versión	
Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por	
Documento revisado por	
Fecha	

1. **OBJETIVO DEL SISTEMA**
   Describir los objetivos del sistema.

2. **ALCANCE DEL SISTEMA**
   Descripción del alcance.

3. **DESCRIPCIÓN GENERAL DEL SISTEMA**
   - Descripción del sistema considerando los puntos principales del mismo
   - Descripción del acceso y seguridad del sistema

4. **FUNCIONALIDAD DEL SISTEMA**
   - Nombre de la funcionalidad 1
     Explicación de la funcionalidad 1

   - Nombre de la funcionalidad 2
     Explicación de la funcionalidad 2

5. **CONSULTAS Y REPORTES**
   a) Nombre de la consulta o reporte 1
      - Descripción
      - Forma de obtención

   b) Nombre de la consulta o reporte 2
      - Descripción
      - Forma de obtención

6. **GLOSARIO DE TÉRMINOS**
   Desarrollar un glosario de términos.

7. **GLOSARIO DE PREGUNTAS FRECUENTES**
   Describir las preguntas frecuentes sobre el manejo del software.

8. **GLOSARIO DE PROBLEMAS FRECUENTES**
   Describir los problemas frecuentes del software.

9. **DESCRIPCIÓN DE PERFILES**
   Descripción de los distintos perfiles de usuarios/as.

10. **MAPA DE NAVEGACIÓN DEL SISTEMA**
    Presentar en forma gráfica la forma de navegación del sistema y su estructura de menús.

Oefa Organismo de Evaluación y Fiscalización Ambiental	**MANUAL DEL SISTEMA Y CONFIGURACIÓN**

Nombre del Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha	

1. ALCANCE
Describir el alcance del documento.

2. DESCRIPCIÓN GENERAL DEL SISTEMA
Descripción breve del sistema.

3. RECURSOS
Especificar los componentes del sistema.

3.1. Recursos Hardware
Recursos hardware 1
- Descripción

Recursos hardware 2
- Descripción

3.2. Recursos software
Recursos software 1
- Descripción

Recursos software 2
- Descripción

4. ENTORNO OPERATIVO DEL SISTEMA

4.1. Entorno de trabajo
Describir brevemente los elementos que conforman el entorno de trabajo del usuario.

4.2. Perfiles de usuario
Describir una relación de los distintos perfiles de usuarios.

4.3. Funcionalidad del sistema
Describir de forma breve los procesos que han sido automatizados.

4.4. Sistemas relacionados
Describir los sistemas relacionados (diagrama de contexto).

4.5. Seguridad (permisos y roles)
Describir los parámetros de seguridad.

4.6. Instalación
Descripción de la instalación del sistema.

4.7. Especificaciones técnicas de ambiente HW y SW
Describir las especificaciones técnicas del ambiente hardware y software.

4.8. Ubicación física de archivos de BD, ejecutables, etc.
Describir específicamente la ubicación actual de los archivos de instalación.

5. INSTALACIÓN Y CONFIGURACIÓN DEL SOFTWARE BASE
Especificar la información para realizar la instalación y configuración del software base.

6. INSTALACIÓN Y CONFIGURACIÓN DEL SISTEMA
Descripción detallada de las actividades de instalación y configuración del sistema.

7. DESINSTALACIÓN DEL SISTEMA Y MARCHA ATRÁS
Describir la información necesaria para la desinstalación y marcha atrás del software.

Oefa Organismo de Evaluación y Fiscalización Ambiental	**Oficina de Tecnologías de la Información**
	Acta de Pruebas de Rendimiento

Sistema o Aplicativo	
Código requerimiento	
Documento elaborado por:	
Documento revisado por:	
Fecha	

I. RESUMEN EJECUTIVO

Tipo de Requerimiento	( ) Desarrollo ( ) Mantenimiento
Área Solicitante	
Lugar	
Fecha - Hora de prueba	

II. RESULTADOS DE LAS PRUEBAS

Se indica con qué tipo de herramienta se realizó las pruebas y con la cantidad de usuarios/as ejecutados/as.

N° REQ	CP Asociado	Descripción	Conforme (SI/NO)
	CP002		
	CP001		

Se debe mostrar la imagen de la cantidad de hilos ejecutados.

1. Imagen nro. 01
2. Imagen nro. 02

III. RECOMENDACIONES (*)

CP Asociado	Descripción
-	No aplica

(*) Solo se llenará en caso que existan observaciones, caso contrario indicar ***"No aplica"***.

FIRMA
_____ **ANALISTA DE CALIDAD DE SOFTWARE** Fecha / Hora:

Oefa Organismo de Evaluación y Fiscalización Ambiental	**Oficina de Tecnologías de la Información**
	Acta de Capacitación

Tipo de Capacitación	( ) Funcional ( ) Técnico
Área Solicitante	
Responsable de la capacitación	
Temario	
Lugar	
Fecha - Hora	

N°	Nombres y apellidos	Área	Correo institucional	Firma

Oefa

CONTROL DE REQUERIMIENTOS

SOLICITUD						APLICACION	RESPONSABLES	FECHA							OBSERVACIONES
Petición	Año	Tipo	Tipo REQ	Número ticket	Descripción	Siglas	Desarrollo / Atención	Solicitud de Atención	Fecha de asignación a Analista de Desarrollo	Fecha inicio de Atención Analista de Desarrollo	Fecha de revisión con usuario	Fecha solicitud a calidad	ESTADO		
			1. Incidencia 2. Mantenimiento 3. Proyecto												

Instructivo de Desarrollo de Software

1. OBJETIVO

Establecer las tareas para facilitar el uso de herramienta para control de versiones y el despliegue de las aplicaciones en ambiente de producción.

2. INSTRUCCIONES

2.1. Control de Versiones

2.1.1. El/La Analista Programador/a elabora el código y/o elementos del software para el funcionamiento de la aplicación.

2.1.2. Se realizan las pruebas unitarias y de integración. Si las pruebas son exitosas se continúa realizando esta acción, si las pruebas no fueron exitosas se revisan los errores en la codificación del software y retorna al Numeral 2.1.1 del presente instructivo.

2.1.3. Las fuentes, war, scripts y otros elementos de software son resguardados en la carpeta de desarrollo: \\oefa.gob.pe\oefa\OTI\Desarrollo. Se disponen carpetas con los nombres de cada aplicativo, donde se debe consignar la fecha, número de acta de requerimiento o ticket de atención.

2.1.4. El/La Analista Programador/a, completa los datos en el Formato PA0302-F04 *"Solicitud de despliegue"* donde se especifica los elementos del software y los pasos que debe seguir para el despliegue y configuración del software.

2.1.5. Se registra la información solicitada en el formato de registro de solicitud de requerimiento de despliegue en ambiente de calidad.

2.1.6. Revisar las fuentes, war, scripts y otros componentes necesarios para el correcto funcionamiento de la aplicación.

2.1.7. Realizar la configuración y despliegue de los elementos de software en ambiente de calidad.

2.1.8. El/La Analista de Calidad de Software realiza las pruebas de software de acuerdo a la documentación remitida por el/la Analista Programador/a. Si las pruebas son exitosas se continúa con la actividad señalada en el Numeral 3.1.9 del presente instructivo, si las pruebas no son exitosas se retorna al Numeral 2.1.1 del presente instructivo y se devuelve el requerimiento a el/la Analista Programador.

2.1.9. Subir a la herramienta de control de versiones (Gitlab[1]) las fuentes, war, scripts y/o componentes necesarios para el correcto funcionamiento de la aplicación.

2.1.10. El/La Coordinador/a de Informática y Gestión Administrativa, procede a revisar y verificar que las fuentes y demás componentes se encuentren registrados en la herramienta de control de versiones.

2.1.11. El/La Analista de Calidad de Software, mediante correo institucional, remite a el/la Especialista de Infraestructura o de base de datos, la atención del requerimiento para despliegue en ambiente de producción; asimismo, el formato PA0302-F04 *"Solicitud de despliegue"* es actualizado por el/la Analista de Calidad.

2.1.12. El/La Analista de Soporte Informático procede a revisar las fuentes, war, scripts y otros elementos de software necesarios para el correcto funcionamiento de la aplicación. Si los elementos de software son los indicados de acuerdo al Formato PA0302-F04 *"Solicitud de despliegue"*, se continúa con la actividad señalada en el Numeral 2.1.13 del presente instructivo. Si no es conforme procede a comunicar, mediante correo institucional, a el/la Analista de Calidad, a efectos de subir las versiones correctas y retorna al Numeral 2.1.6.

[1] Gitlab: es una suite completa que permite gestionar, administrar, crear y conectar los repositorios con diferentes aplicaciones y hacer todo tipo de integraciones, ofreciendo un ambiente y una plataforma para realizar varias etapas de su SDLC/ADLC y DevOps.

2.1.13. Realizar el proceso de despliegue en ambiente de producción de acuerdo a la aplicación. Si el despliegue es exitoso se procede a continuar en el Numeral 2.1.14. Si el despliegue no es exitoso se continúa con el Numeral 2.1.6 del presente instructivo y revisa los componentes junto con el/la Analista Programador.

2.1.14. Comunica, mediante correo institucional, el despliegue en ambiente de producción a el/la Coordinador/a de Informática y Gestión Administrativa, a el/la Analista de Calidad de Software y a el/la Analista Programador. Se verifica el correcto funcionamiento de la aplicación en producción y se comunica a el/la especialista de infraestructura, continúa la actividad del Numeral 2.1.15 del presente instructivo. Si existiera inconvenientes en producción, el/la Analista programador/a solicita el proceso del Rollback[2] de la aplicación.

2.1.15. El/La analista programador/a, procede a registrar y actualizar las versiones en el Aplicativo de Control de Versiones, de acuerdo a las fuentes puestas en producción. Estos componentes son descargados de la herramienta del ambiente de producción[3].

2.1.16. El/La Coordinador/a de desarrollo de sistemas, procede a revisar y verificar que las fuentes y demás componentes se encuentren registrados en el Aplicativo de Control de Versiones.

2.1.17. El/La analista de soporte informático, registra y actualiza las versiones en el Aplicativo de Control de Versiones. Estos elementos de software son las versiones desplegadas en producción.

2.1.18. El/La Coordinador/a de desarrollo de sistemas, procede a revisar y verificar que los elementos de software se encuentren registrados en la SVN.

2.2. Despliegue de fuentes de aplicaciones en ambiente de producción y envío para despliegue en producción

2.2.1. Se recibe el ticket[4], proveniente del área de desarrollo de sistemas hacia calidad.
En el ticket se comparte las rutas donde se encuentran las fuentes de la aplicación y la descripción de las modificaciones o el Formato PA0302-F11 *"Manual de Usuario".*

2.2.2. Acceder a la ruta y copiar las fuentes en el workspace.
Consultar el Formato PA0302-F12 *"Manual del Sistema y Configuración"* y verificar si se requiere algún software adicional para que la aplicación funcione correctamente en el ambiente local. Si requiere softwares complementarios, se debe proceder a instalarlos software descritos en el referido formato.

2.2.3. Utilizar el Integrated development environment (IDE)[5] que se especifica, para cargar las fuentes.

2.2.4. Configurar el IDE según las características necesarias registradas en el archivo *"Entorno Calidad Software"*, los datos a tener en cuenta en dicho archivo son lenguaje de programación, versión, servidor web.

2.2.5. Modificar el archivo de configuración de la aplicación, que está especificado en el Formato PA0302-F12 *"Manual del Sistema y Configuración"*, se debe cambiar los datos de desarrollo por los datos de calidad.

Los datos que se verifican para el cambio pueden ser:

a) Servicios web

---

[2] Rollback: reversión o flagare es una operación que devuelve a la base de datos a algún estado previo.

[3] Se accede como consulta.

[4] Ticket: constituye una consulta de cualquier tipo que realiza el área usuaria o área de la OTI.

[5] IDE: Un entorno de desarrollo integrado o entorno de desarrollo interactivo, es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.

b) Ruta de Alfresco
c) IP[6] o URL[7] de ambiente de despliegue
d) Sistema de seguridad
e) Cadena de conexión de base de datos
f) Base de datos, usuario y clave.
g) Otros campos específicos en el Formato PA0302-F12 *"Manual del Sistema y Configuración"*.

2.2.6. Se inicia la aplicación en el IDE, realizándose un despliegue local en el ambiente de trabajo.
2.2.7. Se realizan pruebas preliminares para comprobar que la aplicación está configurada correctamente.
2.2.8. Se genera el archivo empaquetado de la aplicación para ser desplegado en el ambiente de calidad.
2.2.9. Se carga el archivo empaquetado en el servidor web específico.
2.2.10. Se accede a las rutas de la aplicación y se realizan las pruebas correspondientes a la modificación implementada o de acuerdo al Formato PA0302-F11 *"Manual de Usuario"* adjunto al ticket.
2.2.11. Si las pruebas realizadas no son conformes se retorna la aplicación a desarrollo de sistemas, para las correcciones correspondientes.
2.2.12. Con la conformidad de las pruebas realizadas se procede a configurar las fuentes en el IDE para que se pueda generar el archivo empaquetado destinado a producción.
2.2.13. Se genera el archivo empaquetado para producción.
2.2.14. Se elabora el acta de pase a producción.
2.2.15. Se almacenan los archivos empaquetados y el acta de producción en el repositorio compartido.
2.2.16. Se deriva el ticket a infraestructura y operaciones, compartiendo la ruta del repositorio.
2.2.17. Se informa ***la atención del despliegue en ambiente de producción al usuario,*** mediante correo institucional a las áreas usuarias o usuarios/as finales.

[6] IP: Conjunto de números que identifica, de manera lógica y jerárquica, a una interfaz en red de un dispositivo que utilice el protocolo o, que corresponde al nivel de red del modelo TCP/IP.

[7] URL: Permite denominar recursos dentro del entorno de Internet para que puedan ser localizados.

Anexo N° 1

Estándar de Bases de Datos

1. ALCANCE

Lo dispuesto en el presente estándar es de aplicación obligatoria para el personal del equipo de Desarrollo de Sistemas de la Oficina de Tecnologías de la Información, incluyendo a los proveedores involucrados en el desarrollo de cualquier tipo de base de datos informática para el OEFA.

2. ABREVIATURAS

BD	:	Base de datos
DBA	:	Administrador de Base de datos
OTI	:	Oficina de Tecnologías de la Información
SQL	:	Structured Query Language
TDR	:	Términos de Referencia

3. DEFINICIONES

3.1. Base de Datos

Sistema Gestor Base Datos que recopila y organiza información. En las bases de datos, se puede almacenar información de todas las áreas en una Entidad, así como de personas, productos, pedidos; entre otros.

3.2. Base de Datos de Producción

- Es la Base de Datos que contiene la información real de la Entidad.
- Las instancias alojadas se encuentran en modo ARCHIVELOG.
- Todas las instancias productivas de base de datos se respaldan a cinta.
- La contraseña de usuario SYS y SYSTEM es un código alfanumérico con más de 08 caracteres de uso exclusivo del administrador de base de datos.
- Los usuarios y esquemas de base de datos relacionados a aplicaciones no deben tener el ROL DBA.

3.3. Base de Datos de Calidad o Certificación

- Se refiere a la Base de Datos utilizadas por los/as Analistas de calidad de Software para realizar pruebas y control de los aplicativos existentes en la Entidad, posterior a la implementación de la solución por parte de los/as desarrolladores de software.
- La actualización se realiza cada tres (3) meses, a menos que surja la necesidad de actualización antes del periodo indicado.
- La nomenclatura para la creación de una instancia de base datos en el ambiente de CALIDAD, se coloca el prefijo QA seguido del nombre de la instancia de base datos (máximo ocho (8) caracteres)
- Las instancias alojadas se encuentran en modo NOARCHIVELOG.
- Estas instancias no son respaldadas. Si se requiere realizar un respaldo debe ser solicitado por el área correspondiente.

- La contraseña de usuario SYS y SYSTEM es un código alfanumérico con más de ocho (8) caracteres de uso exclusivo del administrador de base de datos.
- Los usuarios y esquemas de base de datos relacionados a aplicaciones no deben tener el ROL DBA.

3.4. Base de Datos de Desarrollo

- Se refiere a la Base de Datos en la que los/as desarrolladores pueden realizar pruebas de los aplicativos existentes en la Entidad o desarrollar nuevos aplicativos, no contiene información actualizada de la base de datos de producción.
- La actualización se realiza cada tres (3) meses, a menos que surja la necesidad de actualización antes del periodo indicado.
- La nomenclatura para la creación de una instancia de base datos en el ambiente de DESARROLLO, se coloca el prefijo DV seguido del nombre de la instancia de base datos (máximo ocho (8) caracteres)
- Las instancias alojadas se encuentran en modo NOARCHIVELOG.
- Estas instancias no son respaldadas. Si se requiere realizar un respaldo debe ser solicitado por el área correspondiente.
- La contraseña de usuario SYS y SYSTEM es un código alfanumérico con más de ocho (8) caracteres de uso exclusivo del administrador de base de datos.
- Los usuarios y esquemas de base de datos relacionados a aplicaciones no deben tener el ROL DBA.

3.5. Esquema

Un esquema es una colección de estructuras lógicas de datos u objetos. Los objetos de esquema se pueden crear y manipular con SQL e incluir los siguientes tipos de objetos:

- Clusters
- Constraints
- Database links
- Dimensions
- External procedure libraries
- Functions
- Index-organized tables, Indexes, Index Types
- Java classes, Java resources, Java sources
- Materialized views, Materialized view logs
- Object tables, Object types, Object views
- Operators
- Packages
- Sequences
- Stored functions, stored procedures
- Synonyms
- Tables
- Triggers
- Views

3.6. Usuario de Aplicación

Este es un usuario de base de datos creado para ser utilizado en la cadena de

conexión de una aplicación específica.
Este usuario accede a los objetos de esquema con los mínimos accesos necesarios para cumplir las funciones de la aplicación.

3.7. Tabla (Table)
Objeto para almacenar datos. Está formada por una o más columnas, cada una con un tipo de dato asociado.

3.8. Índice (Index)
Estructuras opcionales asociadas a tablas, creadas para mejorar el tiempo de respuesta de una consulta de datos.

3.9. Integridad Referencia
Sistema de reglas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.

3.10. Llave Foránea (Foreign Key - FK)
Limitación referencial entre dos tablas. La clave foránea identifica una columna o grupo de columnas en una tabla (tabla hija o referendo) que se refiere a una columna o grupo de columnas en otra tabla (tabla padre o referenciada). Las columnas en la tabla referendo deben ser la clave primaria u otra clave candidata en la tabla referenciada. Permite mantener la integridad referencial de información almacenada en otra tabla.

3.11. Llave Primaria (Primary Key - PK)
Campo o grupo de campos que identifica en forma única un registro dentro de una tabla. Ningún otro registro puede tener la misma llave primaria. La clave o llave primaria se utiliza para distinguir un registro con el fin de que se pueda tener acceso a ellos, organizarlos y manipularlos.

3.12. Restricción de Unicidad (Unique Constraint)
Asegura valores únicos para cada registro es *"unique"*, esta restricción impide la duplicidad de claves alternas (no primarias), es decir, especifica que dos registros no puedan tener el mismo valor en un campo. Se pueden aplicar varias restricciones de este tipo a una misma tabla, y pueden aplicarse a uno o varios campos que no sean clave primaria.

3.13. Restricción de Verificación (Check Constraint)
Especifica los valores que acepta un campo, evitando que se ingresen valores inapropiados. Este tipo de restricción verifica los datos cada vez que se ejecutan inserciones y actualizaciones. Si la tabla contiene registros que no cumplen con la restricción que se va a establecer, la restricción no se puede establecer, hasta que todos los registros cumplan con dicha restricción.

3.14. Disparador (Trigger)
Procedimiento que define una acción que la base de datos debe llevar a cabo cuando se produce algún suceso relacionado con la misma. Pueden utilizarse para completar la integridad referencial, también para imponer reglas de negocio complejas o para auditar cambios en los datos. La ejecución de disparadores es transparente al usuario.

3.15. Enmascaramiento

Proceso mediante el cual se cambian ciertos elementos de los datos de una base de datos, cambiando su información, pero consiguiendo que la estructura permanezca similar, de forma que la información sensible quede protegida, lo cual garantiza que la información sensible del usuario/a no se encuentre disponible fuera de un ambiente productivo.

3.16. Funciones (Function)

Conjunto de sentencias que al ser ejecutadas retorna uno o más valores.

3.17. Paquete (Package)

Conjunto de procedimientos y funciones de manera lógica que puede encapsular toda la casuística asociada a un determinado tipo de tarea y pueden usar las mismas variables definidas.

Consta de dos elementos:

a) Especificación o cabecera: contiene las declaraciones públicas (es decir, accesibles desde cualquier parte de la aplicación) de sus programas, tipos, constantes, variables, cursores, excepciones, etc.
b) Cuerpo: contiene los detalles de implementación y declaraciones privadas, es decir, accesibles solamente desde los objetos del paquete.

3.18. Procedimiento almacenado (Stored Procedure)

Conjunto de sentencias que realiza una tarea específica.

3.19. Secuencia (Sequence)

Objeto de base de datos que genera un número secuencial único cada vez que es invocado.

3.20. Sinónimo (Synonym)

Alias de un objeto de base de datos del tipo tabla, vista, secuencia o unidad de programa.

3.21. Tabla de Búsqueda (Lookup table)

Tabla que contiene datos de referencia que es utilizada por otras tablas de la base de datos.

3.22. Vista (View)

Presentación de información contenida en una o más tablas, su tratamiento es equivalente a una tabla.

4. CONSIDERACIONES GENERALES

- La aplicación de los estándares de base de datos es de uso obligatorio en el desarrollo de nuevos sistemas de información o el mantenimiento evolutivo de los mismos, ya sea mediante recursos propios o la contratación de proveedores de servicios.
- La OTI es responsable de brindar a los proveedores de servicios de desarrollo de sistemas que entregan código fuente, los estándares de base de datos definidos para asegurar un eficiente mantenimiento evolutivo.
- La OTI debe revisar y validar la aplicación de este Estándar, exigiendo el modelo entidad relación y el diccionario de datos correspondiente.
- Debe aplicarse sin excepción en las bases de datos de desarrollo, certificación y

producción.

5. DISPOSICIONES ESPECÍFICAS

5.1. ARQUITECTURA DE BASE DE DATOS

La arquitectura de las bases de datos en los ambientes se divide en:

Ambiente de Producción:

Motor	Versión	Modalidad	Sistema Operativo
Oracle Database 19c	19.13	RAC con 2 nodos	Oracle Linux x86 64-bit

Ambiente de Calidad y Desarrollo:

Motor	Versión	Modalidad	Sistema Operativo
Oracle Database 19c	19.13	Standalone	Oracle Linux x86 64-bit

Es importante mencionar que:

- El juego de caracteres de las bases de datos es WE8MSWIN1252 (NLS_CHARACTERSET= WE8MSWIN1252).
- El National Character set es AL16UTF16 (NLS_NCHAR_CHARACTERSET = AL16UTF16).

5.2. NOMENCLATURA PARA LA CREACIÓN DE OBJETOS EN SISTEMAS TRANSACCIONALES

5.2.1 CONSIDERACIONES GENERALES PARA LA NOMENCLATURA

- El nombre de un objeto de base de datos debe estar en singular, teniendo un máximo de treinta (30) caracteres. No se utilizan abreviaciones salvo que este exceda el límite.
- Adicionalmente, los nombres sólo pueden incluir caracteres, tales como [A-Z] y [0-9], separando cada palabra entre guión bajo (_).
- Las letras acentuadas se reemplazan con las equivalentes no acentuadas, y en lugar de la letra eñe (Ñ) se utiliza (NI). Ejm: ANIO_EXPEDIENTE.
- Se debe registrar metadata a la base de datos, agregando comentarios a las tablas y campos, sobre todo a los booleanos.

5.3. APLICACIÓN DE ESTÁNDARES A ESQUEMA

- El nombre tiene una longitud máxima de quince (15) caracteres
- Debe ser un nombre único en la base de datos que identifique al aplicativo, el cual es coordinado con el administrador de base de datos del OEFA.

5.4. APLICACIÓN DE ESTÁNDARES A TABLA

Nomenclatura:

- El nombre de una tabla debe cumplir las consideraciones generales descritas en el Subnumeral 5.2.1 del presente documento.
- Los nombres de las tablas se realizan conforme al siguiente formato:

Formato:
[T][XX][Y]_[ZZZZZZZZZZ]

Donde:	**Descripción**
T	Prefijo de Tipo de Objeto
XX	Tipo de tabla según su naturaleza, se encuentra compuesto por dos caracteres
Y	Tipo de tabla según su estructura, se encuentra compuesto por un carácter
ZZZZZZZZZZ	Nombre representativo de la tabla, se recomienda que no exceda los 10 caracteres

Los tipos de tabla según su naturaleza son:

MA	MAESTRO
MV	MOVIMIENTO
TM	TEMPORAL
GE	GENERAL

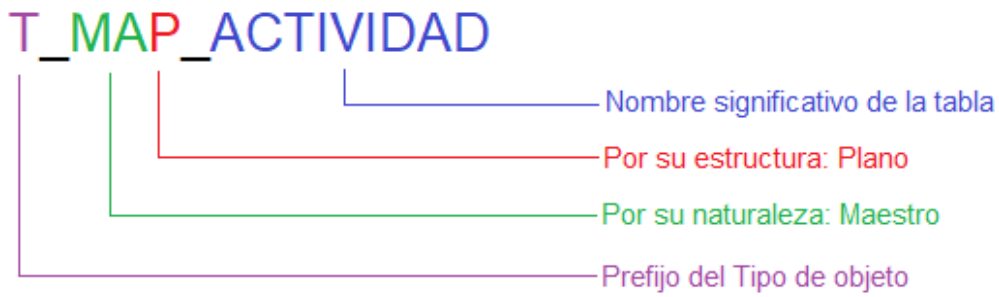
Los tipos de tablas según su estructura son:

P	Plano
C	Cabecera
D	Detalle

A continuación, algunos ejemplos de las combinaciones entre los tipos de tabla según su naturaleza y su estructura:
MAP = Maestra Plana
MVC = Movimiento Cabecera
MVD = Movimiento Detalle

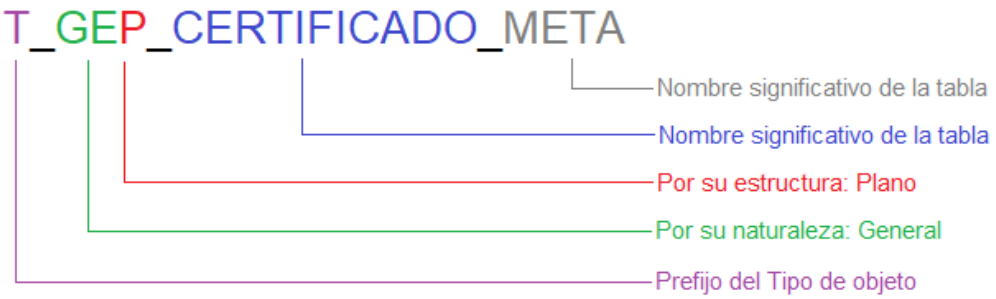
TMP = Temporal Plano

Ejemplo:



En el caso de que sean tablas provenientes de la relación de dos o más tablas (tablas relacionadas), se utilizan los nombres descriptivos de las tablas relacionadas, utilizando una letra "_" para separarlas.

Ejemplo:



Consideraciones para nombrar columnas

- El nombre de una columna debe cumplir las consideraciones generales descritas en el Subnumeral 5.2.1 del presente documento.
- El nombre de una columna debe cumplir el siguiente formato:

Formato: [TD]_[AAA_BB_CCC]

Donde:	Descripción
TD	Prefijo del tipo de dato almacenado en la columna
AAA_BB_CCC	Nombre representativo de la columna. Debe describir el dato que va a almacenar de manera entendible; si se usan abreviaturas,deben ser mnemotécnicas. Las columnas usadas como flag serán de tipo Char (1) y nombrarse con una palabra que indique su estado en verdadero. Las columnas de tipo Fecha y hora deben nombrarse

	con una palabra que indique si es Fecha y/u hora.

Los tipos de dato (TD) han sido agrupados de la siguiente manera:

TIPO DATO	CONTENIDO	NOMENCLATURA
NUMBER	Número	NU
INTEGER	Número	NU
FLOAT	Número	NU
BLOB	Binario	BL
CLOB	Texto Grande	CL
NCLOB	Texto Grande	CL
DATE	Fecha	FE
TIMESTAMP	Fecha	FE
RAW	Binario	RW
VARCHAR2	Texto	TX
CHAR	Texto	TX
NVARCHAR2	Texto	TX
NCHAR	Texto	TX

Ejemplo:
TX_DESCRIPCION
FE_FECHA_REGISTRO
FE_FECHA_HORA_MODIFICA
BL_FOTO_POSTULANTE

- Las columnas de tipo llave primaria deben ubicarse al inicio de la definición de la tabla, conforme al siguiente formato:

Formato: [TD]_[ID]_[ZZZZZZZZZZ]

Donde:	Descripción
TD	Prefijo del tipo de dato almacenado en la columna
ID	Prefijo de tipo de columna
ZZZZZZZZZZ	Nombre representativo de la tabla, se recomienda que no exceda los 10 caracteres

Ejemplo:
Tabla : T_MAP_MONEDA
Columna : NU_ID_MONEDA

Para las columnas de tipo clave foránea, su tratamiento es similar al de la clave primaria, y si hubieran más de dos (2) campos de tipo clave foránea referidos a la

misma tabla, se incorpora al medio un nombre más que complemente el dato.

- Finalmente, toda columna debe tener un comentario que indique el propósito de la misma. Asimismo, tener en cuenta las siguientes consideraciones:
  - Cuando el campo de la tabla es un dato confidencial, es necesario que se especifique para que quede registrado en la Base de Datos.
    Ejm:
    ```
    COMMENT ON COLUMN
     ESQUEMA.TABLA.CODIGO_DECLARACION_JURADA_TX IS
    Código de la declaración jurada. Dato (CONFIDENCIAL)';
    ```
  - Cuando se trate de un campo Primary key, y si este utiliza una secuencia, en el comentario deberá indicar el nombre de la secuencia.
    Ejm:
    ```
    COMMENT ON COLUMN
    ESQUEMA.TABLA.ID_ACTA_NU IS 'Identificador del registro en la tabla de Actas.
    Secuencia: NOMBRE SECUENCIA';
    ```
  - Cuando se trate de un campo identificador de llave foránea (Foreign Key), en el comentario deberá indicar la tabla a la que tiene referencia.
    Ejm:
    ```
    COMMENT ON COLUMN
    ESQUEMA.TABLA.ID_UNIDAD_FISCAL_NU IS 'Foránea de la tabla Unidad
    Fiscalizadora';
    ```
  - Cuando se trate un campo del tipo flag o booleano, se debe especificar los posibles valores que contendrá.
    Ejm:
    ```
    COMMENT ON COLUMN ESQUEMA.TABLA.ACTIVO IS 'Flag indicador si el
    registro se encuentra Activo = 1 o No_Activo = 0';
    ```

5.5. APLICACIÓN DE ESTÁNDARES A CONSTRAINT

- El nombre de un constraint debe cumplir las consideraciones generales descritas en el Subnumeral 5.2.1 del presente documento.
- El nombre del constraint debe ser descriptivo respecto de los objetos a los cuales representan y depende del tipo del mismo nombre, conforme al siguiente detalle:

Tipo de Contraints	**Prefijo**
Primary Key	PK
Foreign Key	FK
Unique	UK
Check Contraint	CK

Nomenclatura para Primary Keys (Llaves primarias)

Formato:
[PK]_[Nombre representativo de la Tabla]

Ejemplo:
Tabla : T_MAP_MONEDA
Primary key : PK_MONEDA

Nomenclatura para Foreign Key (Llaves foráneas)

Formato:
[FK]_[Nombre representativo Tabla Origen]_[Nombre representativo Tabla Destino]

Ejemplo:
Tabla origen : T_MAP_MONEDA
Tabla destino : T_MAP_PAGO
Foreign Key : FK_MONEDA_PAGO
Indica que la FK es PK en T_MAP_MONEDA.

Excepciones
Cuando se trate de una tabla origen a una tabla destino por más de una vez, es necesario usar el nombre representativo de la columna de la tabla destino al final del nombre de la llave foránea.
Ejemplo:
Tabla Origen : T_GENP_UNIDADMEDIDA
Tabla Destino : T_GENP_IMPOSICIÓN
Llaves foraneas: FK_UNIDADMED_IMPOSICION_LONG
FK_UNIDADMED_IMPOSICION_ALTURA

Nomenclatura para Unique (Restricciones de Unicidad)

Formato:
[UK]_[Nombre de la Tabla]_[Secuencia de unique de tabla]

Ejemplo:
Tabla : T_MAP_MONEDA
Unique Key Nº 01 de la Tabla: UK_MONEDA_01

Nomenclatura para Check Constraint (Restricciones de Verificación)

Formato:
[CK]_[Nombre de la Tabla]_[Secuencia check de tabla]

Ejemplo:
Tabla : T_MAP_MONEDA

Check Constraint Nº 01 de la Tabla: CK_MONEDA_01 *Check Constraint Nº 02 de la Tabla:* CK_MONEDA_02

5.6. APLICACIÓN DE ESTÁNDARES A INDICES

- La forma de nombrar un índice depende del nombre de la tabla y el nombre de la columna a indexar. Si se trata de un índice compuesto por más de una columna se debe señalar un nombre que describa al índice creado, conforme al siguiente detalle:

Formato:
[IDX]_[Nombre representativo de la Tabla]_[Nombre representativo de columna]

Ejemplo Indice simple:
Tabla : T_MAP_EMPRESA
Campo : RUC_NU
Índice : IDX_EMPRESA_RUC
Ejemplo Índice compuesto:
Tabla : T_MAP_EMPRESA
Campos : ID_EMPRESA_TX, RUC_NU
Índice : IDX_EMPRESA_ID_EMPRESA_RUC

5.7. APLICACIÓN DE ESTÁNDARES A SECUENCIAS

- Para el caso de una secuencia el nombre será el prefijo de tipo de objeto más el nombre representativo de la tabla, conforme al siguiente detalle:

Formato:
[Prefijo de Tipo de Objeto]_[Nombre representativo de Tabla]

Objeto de Base de datos	Prefijo de Tipo de Obj.
Secuencia	SEQ

Ejemplo:
Tabla : T_MAP_EMPRESA
Secuencia : SEQ_EMPRESA

5.8. APLICACIÓN DE ESTÁNDARES A VISTAS y VISTA MATERIALIZADA

- Para nombrar una vista se deben colocar nombres descriptivos, a fin de que se pueda utilizar para devolver el valor de varias tablas.
- La denominación de una vista depende de los nombres de la aplicación, de la tabla y de los objetos de base de datos, conforme al siguiente detalle:

Formato:
[Prefijo de Tipo de Objeto]_[Nombre de Vista]

Objeto de Base de datos	Prefijo de Tipo de Obj.
Vista	VW
Vista materializada	VM

Ejemplo:
Vista : VW_DATOS_PERSONA
Vista Materializada : VM_ DATOS_BOLETA_PAGO

5.9. APLICACIÓN DE ESTÁNDARES A SINÓNIMOS

- Los sinónimos son usados para ocultar el nombre del propietario, la ubicación o el nombre real de una tabla.
- Asimismo, son utilizados para proporcionar a los usuarios nombres de objetos menos complicados que los reales.

Formato:
[Nombre representativo del sinónimo]

Ejemplo:
Tabla : T_ESTADO_TERMINOREFERENCIA
Sinónimo : ESTADO_TERMINOREF

5.10. APLICACIÓN DE ESTÁNDARES A PAQUETES, PROCEDIMIENTOS, FUNCIONES Y DISPARADORES

- Para nombrar paquetes, procedimientos, funciones, y disparadores, la nomenclatura es el prefijo de tipo de objeto, seguido del nombre representativo del objeto, conforme al siguiente detalle:

Formato:
[Prefijo de Tipo de Objeto]_[Nombre de Objeto]

Objeto de Base de datos	Prefijo de Tipo de Obj.
Paquetes	PKG
Procedimientos	SP
Función	FN

Disparador	TRG

Ejemplo:
PKG_REGISTRO_EXPEDIENES
SP_PROC_UPD_ESTADO_SUP
FN_PUNTAJE_FINAL_MATRIZ
TRG_AUD_EFA_ACTIVIDAD

- Para los objetos de base de datos como: paquetes, procedimientos, funciones y disparadores incluir comentarios del propósito del objeto creado, en caso sea modificado debe indicarse en un comentario en la sección de revisiones. Como, por ejemplo:

```
CREATE OR REPLACE PACKAGE SIS_STD.PKG_REGISTRO_EXPEDIENTES IS
  ----------------------------------------------------------------------------------------------
  --Sistema              :     Sistema de Gestión Documentaria
  --Módulo               :     Modulo Mesa de Partes
  --Objetivo             :     Registrar los expedientes externos nuevos.
  --Desarrollado por     :     Pedro Perez
  --Fecha Creación       :     20-01-2018
  ----------------------------------------------------------------------------------------------
  ----------------------------------------------------------------------------------------------
  --Modificado por       :     Pedro Perez
  --Fecha Modificación   :     25-01-2018
  --Motivo               :     Cambio de la funcionalidad del Negocio.
  --Indicador cambio     :     001
  --Descripción cambio   :     Se añadió campo RUC como dato de la supervisión
                                principal
  ----------------------------------------------------------------------------------------------
```

- Para todos los objetos paquetes, procedimientos, funciones, y disparadores, cada vez que se requiera su actualización se debe colocar el identificador de inicio y fin del cambio, conforme al siguiente detalle:

Formato:
Inicio del Cambio: [@][Indicador Cambio][i]
Fin del Cambio: [@][Indicador Cambio][f]

Ejemplo:

```
/*@001i
SELECT SIS_STD.DIRECCION_SEQ.NEXTVAL INTO DIRECCION FROM DUAL;
INSERT INTO
DIRECCION(DIRECCION, REFERENCIA, ID_CLIENTE, UBIGEO, ES_DFISCAL)
VALUES(v_DIRECCION, v_REFERENCIA, v_ID_CLIENTE, v_UBIGEO,
v_ES_DFISCAL);
```

```
@001f */
```

5.11. APLICACIÓN DE ESTÁNDARES A ROLES

- La forma de consignar el nombre a un rol depende de la denominación de la aplicación y tipo de rol que se asigna al grupo de permisos, conforme al siguiente detalle:

Formato:
[Prefijo de Tipo de Objeto]_[Nombre Aplicacion]_[Nombre Rol]

Objeto de Base de datos	Prefijo de Tipo de Obj.
Rol	ROL

Ejemplo:
Aplicación : SINADA
Tipo Rol : ADMIN
Rol : ROL_SINADA_ADMIN

5.12. APLICACIÓN DE ESTÁNDARES A DATABASE LINKS

- La forma de poner nombre a un database link depende del nombre del esquema al cual desea conectarse, conforme al siguiente detalle:

Formato:
[Prefijo de Tipo de Objeto]_[Nombre Esquema Destino]

Objeto de Base de datos	Prefijo de Tipo de Obj.
Database Link	DBLINK

Ejemplo:
Desde la aplicación SIGA-OEFA (Esquema Origen) se requiere crear un database link para conectarse con la aplicación SSO (Esquema destino).

Database Link : DBLINK_SSO
Usuario Conexión: LNK_SSO

5.13. APLICACIÓN DE ESTÁNDARES A TABLESPACES

- La forma de poner nombre a un tablespace depende de la denominación de la aplicación y tipo de datos que almacena, estos pueden ser DATA (almacena datos) o INDEX (almacena índices), conforme al siguiente detalle:

Formato:
[Prefijo de Tipo de Objeto]_[Nombre Aplicacion]_[Tipo de Dato que almacena]

Objeto de Base de datos	Prefijo de Tipo de Obj.
Tablespace	TS

Ejemplo:
Aplicación *:* SINADA
Tablespace : TS_SINADA_DATA
TS_SINADA_INDEX

5.14. PROGRAMACIÓN

- Las operaciones para la manipulación de datos (SELECT, INSERT, UPDATE, DELETE) deben realizarse a nivel de base de datos a través de paquetes, procedimientos y/o funciones. Estos objetos tienen que ser creados en el esquema propietario de la aplicación, permitiendo un mantenimiento de las aplicaciones sin necesidad de modificar el código fuente.
- En ningún caso el desarrollador puede forzar el uso de índices en un PROCEDIMIENTO ALMACENADO ni colocar sentencias que alteren el comportamiento de los bloqueos.
- No se deben usar procedimientos encriptados.
- Las vistas deben ser utilizadas para simplificar los queries.
- En ningún caso se otorgan permisos sobre columnas, por ello se debe crear una vista para que se le otorgue permiso de SELECT a toda la vista.
- Los tipos de datos definidos por usuario (User Defined Datatypes) deben ser debidamente documentados.
- Se recomienda el uso de las tablas de UBIGEO alojadas en el esquema PADRONES, con la finalidad de no duplicar datos.

5.15. SEGURIDAD

- El estándar adoptado por el OEFA se emplea la Seguridad Estándar para el acceso a las Bases de Datos, es decir, que para cada una de ellas se crea un único Login con el que todos los/as usuarios/as acceden a la BD, asimismo, se crea Logins como roles requiera el proyecto. Todas las cuentas son creadas por el DBA.
- Para cada Sistema o Aplicativo existen dos esquemas, en el cual, uno es el propietario de los objetos de la base de datos y el otro es el usuario que cuenta con todos los beneficios para el normal funcionamiento de la aplicación, usuario con el que se conectará al aplicativo.
- La auditoría de la base de datos ORACLE es realizada utilizando el software propio de la base de datos, en cuanto a la auditoría a nivel aplicación debe ser

implementada por el desarrollador de la aplicación (*Ver Anexo 1*).
- En ningún caso las cuentas de usuarios utilizan el System Administrator (DBA).

5.16. ENMASCARAMIENTO DE LA INFORMACIÓN

- Cuando las bases de datos de calidad o desarrollo son actualizadas inmediatamente se inicia con el proceso de enmascaramiento de la información para evitar que datos confidenciales sean expuestos.
- Para determinar la información que debe ser enmascarada, se revisa el diccionario de datos de la aplicación, el cual debe indicar la columna que contiene datos sensibles/confidenciales.
- Asimismo, pueden ser enmascaradas las columnas o tablas que sean solicitadas por el propietario de la información mediante correo institucional.

5.17. INTEGRIDAD DE DATA

- La integridad referencial es manejada a través de Constraints Primary Key y Foreign Keys de acuerdo al Modelo de Datos definido.
- Los triggers deben ser utilizados sólo en los casos que se quiera hacer alguna acción en cascada sobre la integridad referencial o cuando se quiera evitar que una clave primaria sea modificada o si la funcionalidad del aplicativo así lo requiera, tomando en cuenta que el uso de los mismos disminuye la velocidad de actualización de las tablas.
- En caso se requieran hacer validaciones sobre columnas se debe utilizar el constraint de tipo CHECK, salvo que se quiera hacer una validación de un USER DEFINED DATATYPE.

5.18. PERFORMANCE Y TUNNING DE BASE DE DATOS

- Los desarrolladores deben realizar un test con un mínimo de cien (100) usuarios/as concurrentes, pudiendo aumentar la cantidad de usuarios/asr dependiendo del tipo de sistema, estas pruebas son supervisadas por el DBA y el/la Gestor/a del Proyecto, con el fin de garantizar el funcionamiento adecuado de la base de datos.
- Si el test no es aprobado satisfactoriamente, se deben corregir las consultas y en algunos casos el código de los sistemas.
- Previamente al pase a Producción de una Base de Datos, ésta debe ser analizada por el DBA en colaboración con los desarrolladores para optimizar al máximo el funcionamiento si es que la performance alcanzada no es adecuada.
- Todos los sistemas a desarrollarse deben optimizar el uso de conexiones por usuario/a.

5.19. PASE A PRODUCCIÓN

Para que se realice un pase a producción en capa de Base de Datos, el área de calidad de la Oficina de Tecnologías de la Información debe gestionar la documentación pertinente y esta debe estar registrada en el repositorio respectivo de pases a producción del servidor en el cual se detalla lo siguiente:

a. **Acta de Requerimiento:** Se debe consignar las características técnicas de hardware y software necesario, el tipo de licencias para el funcionamiento del sistema en todos los niveles y se debe contemplar un breve resumen general del sistema, señalando el responsable del mismo y las áreas usuarias relacionadas.
b. **Diccionario de datos:** Diseño completo de la Base de Datos, así como de las interfaces. Asimismo, se debe consignar la descripción de cada uno de los objetos que se utilizan en el aplicativo la cual cumplirá con la nomenclatura vigente y el detalle de conexiones con otras bases de datos en caso corresponda. En el diccionario de datos se debe incluir las columnas que contienen los datos sensibles /confidenciales.
c. **Proyección de crecimiento de la información:** Proyección anual de la información para determinar las necesidades de espacio en disco, asimismo, proceso de cierre periódico para el guardado y limpieza de tablas cuya información histórica podría ser extraída de la Base de Datos para la recuperación de espacio de Disco.
d. **Formato de Pase a Producción:** Documento que será adjuntado para el proceso al pase a producción.
e. **Archivo de script:** Los archivos de scripts SQL tiene que seguir las siguientes pautas y recomendaciones:
    - Se tienen que numerar los archivos con el número correspondiente al orden de ejecución de la instalación, la cual no incluirá espacios en blanco ni caracteres especiales. El orden de ejecución es el siguiente:
        - Permisos que permiten la creación de objetos
        - Creación de Objetos
        - Permisos de acceso a los objetos creados
    - El nombre del archivo debe ser representativo.
    - Todos los scripts a ejecutar deben contener de forma obligatoria el esquema al que pertenecen los objetos.
    - No se deben mezclar sentencias DDL y DML en el mismo archivo.
    - No se deben enviar archivos muy extensos, a fin de evitar fallas en algún punto del archivo.

Si el pase a producción es un nuevo proyecto, se solicitan todos los documentos descritos anteriormente. Para el mantenimiento de un sistema existente se requieren los documentos señalados en los Literales **a**, **d**, **e**, y opcionalmente **b**, solo si el mantenimiento implica la creación de nuevos objetos).

5.20. EJECUCIÓN DE SCRIPTS EN AMBIENTE DE PRODUCCIÓN (SÓLO DML)

Existen solicitudes del área usuaria para ejecutar sentencias de tipo DML en el ambiente de producción. Para que se lleve a cabo este tipo de solicitud el área de calidad se debe gestionar la documentación pertinente elaborada por el analista de desarrollo y revisado por el analista de calidad, la cual debe estar registrada en la ruta compartida de pases a producción del file server.

a. En la solicitud se debe incluir la siguiente información:

- Nombre de la Base de datos

- Nombre del esquema
- Nombre del Script
- Ruta de Ubicación del Script
- El código del GLPI asociado.
- La lista de tablas que deberán ser respaldadas antes de la ejecución del script.

b. Otras consideraciones:

- En el caso de UPDATE/DELETE de filas, se debe consignar el número de registros a los que afecta su ejecución.
- En la ruta donde se ubican los scripts, se adjunta la autorización de el/la usuario/a líder de la aplicación para la solicitud.

Anexo N° 1: Recomendaciones para Auditoría desde Aplicación

Para procesos de auditoría más específicos se recomida personalizar de acuerdo a la necesidad de la organización, para este tipo de auditoría personalizada debe ser parte del desarrollo de los proyectos informáticos, a continuación, unas recomendaciones para su implementación:

1) Definir las tablas que se requieren auditar.
2) Definir el nivel de granularidad (nivel de detalle almacenado en una tabla).
3) Definir la periodicidad del backup de estas tablas.
4) Definir la depuración de la auditoría almacenada.
5) Se almacena una tabla de auditoría por cada tabla auditada
6) La nomenclatura para estas tablas será AUD_NOMBRE_TABLA_AUDITADA. Por ejemplo: La tabla TACTIVIDAD, su tabla de auditoría será AUD_TACTIVIDAD.
7) La tabla de auditoría tendrá las siguientes columnas:
   a) Un identificador
   b) Nombre de la columna modificada
   c) Acción realizada (I: Insert, D: Delete, U: Update)
   d) Valor anterior del registro
   e) Nuevo valor del registro
   f) Fecha y Hora de ejecución
   g) Usuario de base de datos con el que se ejecutó la acción
   h) Usuario de sistema operativo con el que se ejecutó la acción
   i) Máquina con la que se ejecutó la acción

Anexo N° 2

Estándar de Arquitectura de Nube

1. OBJETIVO

Establecer criterios para la habilitación, configuración y operatividad de la infraestructura en la nube disponible para atención de los servicios tecnológicos del OEFA.

2. ALCANCE

Lo dispuesto en el presente documento es de aplicación obligatoria para el personal del equipo de infraestructura de la Oficina de Tecnologías de la Información, incluyendo a los proveedores involucrados en la prestación de servicios tecnológicos.

3. HABILITAR, CONFIGURAR Y PONER EN OPERATIVIDAD LA INFRAESTRUCTURA EN NUBE

Esta etapa comprende habilitar, configurar y poner en operatividad la infraestructura en nube con sus tres (3) ambientes (desarrollo, calidad y producción); así como establecer la conectividad con el centro de datos de la infraestructura en on-premise[1] del Organismo de Evaluación y Fiscalización Ambiental - OEFA. Para lo cual se debe considerar las siguientes actividades:

a) El proveedor debe proporcionar, dentro de los tres (3) días calendarios siguiente de notificada la conformidad del primer entregable, una lista mínima de dos (2) y máxima de tres (3) zonas de disponibilidad[2] geográfica y su nivel de latencia, con la finalidad de que la Oficina de Tecnologías de la Información - OTI al cabo de tres (3) días calendarios elija la zona de disponibilidad más cerca de los/as usuarios/as finales y con menor latencia.

b) Establecer en la configuración que los datos deben almacenarse única y exclusivamente en la región geográfica seleccionada por la OTI (según el numeral 3 literal a), salvo servicios de nube que sean de alcance global, definiendo con la OTI las excepciones y los protocolos a aplicar ante el traslado temporal o permanente de los datos almacenados en otra región geográfica a la elegida, el cual está incluida dentro de la lista proporcionada, señalada en el Numeral 3 literal a.

c) En la configuración debe habilitarse el acceso bajo demanda a los servicios en nube por parte de las aplicaciones informáticas, las cuales se encuentran alojadas en esta infraestructura, garantizando un acuerdo de disponibilidad de servicios en nube del 99.95%, cuyo incumplimiento es penalizado.

d) El proveedor debe implementar la infraestructura, habilitando y configurando los servicios en nube, en base a la arquitectura definida en el numeral 3.1. Asimismo, para la conectividad con el centro de datos de la infraestructura en on-premise del OEFA se debe realizar un enlace de alta disponibilidad definido en el numeral 3.1.

e) La habilitación y configuración de los servicios en nube para la implementación de la

[1] Instalaciones propias

[2] Las zonas de disponibilidad son ubicaciones concretas dentro de una región del proveedor de nube diseñadas para estar aisladas de los errores que se produzcan en las demás zonas de disponibilidad. Proporcionan conectividad de red y de baja latencia con las demás zonas de disponibilidad dentro de la misma región.

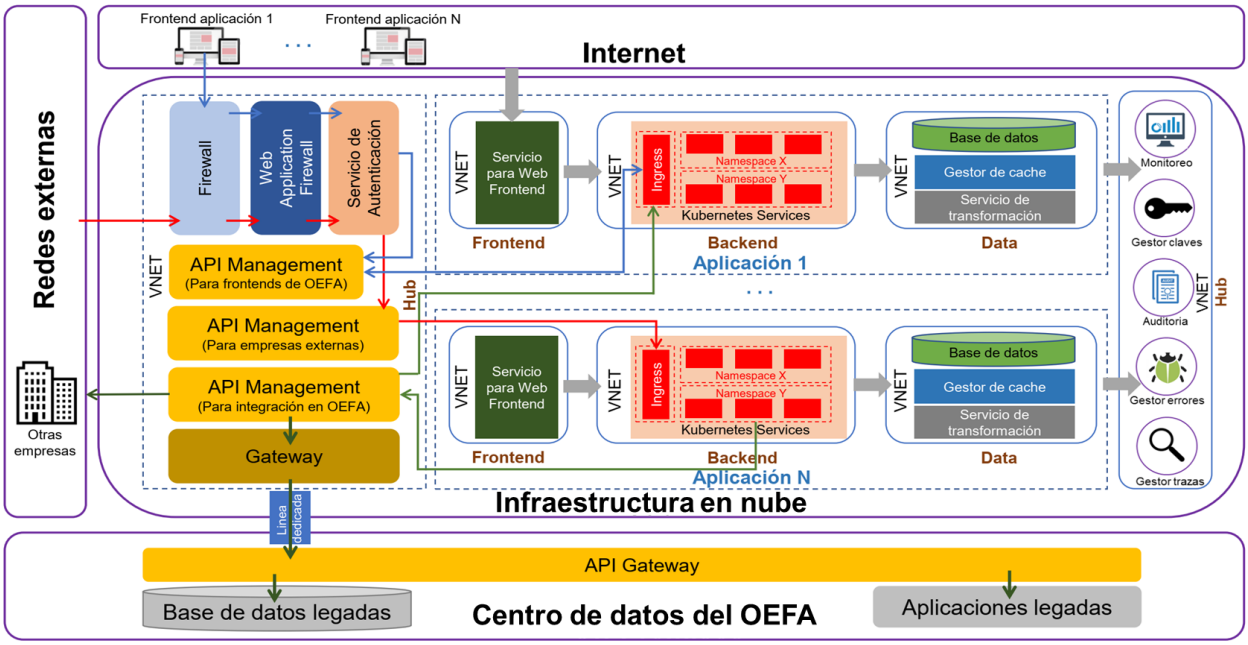
infraestructura en nube debe cumplir con las características definidas en el numeral 3.1.1, 3.1.2, 3.1.3, 3.1.4 para el ambiente de producción.

f) El proveedor debe implementar en la infraestructura en nube tres (3) ambientes denominados: *(i)* ambiente de desarrollo; *(ii)* ambiente de calidad; y, *(iii)* ambiente de producción. Considerando que el ambiente de desarrollo tendrá el 15% de la capacidad configurada para el ambiente de producción y el ambiente de calidad el 25% de la capacidad configurada para el ambiente de producción.

3.1 ARQUITECTURA A SER IMPLEMENTADA

El diagrama de la arquitectura para el proyecto se muestra en la figura N° 1:

Figura N° 1: Arquitectura a ser implementada



En la figura N° 1, se muestra la interacción entre la infraestructura en nube y la conectividad con el centro de datos del OEFA, así como, la composición de la infraestructura en nube con diversos elementos, tales como:

- VNET Data: Detalle en el numeral 3.1.1
- VNET Backend: Detalle en el numeral 3.1.2
- VNET Hub: Detalle en el numeral 3.1.3
- VNET Frontend: Detalle en el numeral 3.1.4

Asimismo, las flechas rojas, verdes y azules señaladas en la figura Nº 1 representan el flujo de comunicación de los servicios dentro de la infraestructura en nube; así como el flujo de comunicación de redes externas, internet y el centro de datos del OEFA, los cuales deben ser realizadas en la configuración de los servicios en nube.

A. Enlace dedicado de baja latencia.

El proveedor debe realizar dos enlaces independientes y dedicados de baja latencia (menor o igual a 30 ms) para la alta disponibilidad, dichos enlaces deben provenir del mismo punto de conexión del proveedor, pero por diferentes rutas físicas para la conectividad directa entre la infraestructura en nube y los centros de datos en on-premise del OEFA, los cuales se ubican en el piso 18 de la sede Central del OEFA, con la finalidad de asegurar la conectividad y estabilidad de las comunicaciones.
El enlace debe contar con las siguientes características:

Tabla N° 1: Características del enlace dedicado de baja latencia

Medio de transmisión	La fibra óptica debe ser canalizada y subterránea en la última milla del proveedor, es decir hasta el buzón o puntos de acceso o CTO (caja terminal óptica) más cercano, para cada enlace, hasta la sede central del OEFA.
Ancho de banda en Gbps	1 Gbps simétrico con overbooking 1:1
Alta disponibilidad, es decir 2 enlaces independientes)	Si
Conectividad directa con el Proveedor de Servicio de Nube, es decir que el proveedor del servicio en nube esté conectado directamente el centro de datos en on-premise del OEFA)	Si

3.1.1 VNET DATA: CARACTERÍSTICAS DE LOS SERVICIOS PARA LA REDVIRTUAL DE DATOS

Los servicios involucrados dentro de la red virtual de datos brindarán soporte a tres (3) servicios: *(i)* el servicio de mensajería; (*ii)* el servicio de cache de base de datos; y, *(iii)* el servicio de base de datos no relacional. Dichos servicios son configurados dentro de una opción de plataforma.

Asimismo, cada uno de ellos debe ampliarse en las capacidades solicitadas acorde al crecimiento de nuevos procesos que se dan en el OEFA. Las solicitudes de ampliación, se requerirán, mediante correo institucional, a el/la Jefe/a de la OTI o a el personal asignado por este.

Las características y la configuración se especifican en la siguiente tabla, si el proveedor requiere características adicionales propias del fabricante, estas deben ser incluidas como parte de su oferta.

Tabla N° 2: VNET DATA

DENOMINACIÓN	CARACTERÍSTICAS
NETWORKING TRANSVERSAL VIRTUAL	✓ Este servicio debe considerar la configuración de la red para el servicio de datos. ✓ La configuración de la red comprenderá notación CIDR /26. ✓ Toda la comunicación entre la red de datos y otras redes será interna, no habrá acceso desde internet a esta red ✓ Dicha red debe estar dividida como mínimo en dos subnet: • Subnet Mensajería con una distribución para dar soporte a 62 IP's. • Subnet para Base de Datos con una distribución para dar soporte a 62 IP's. ✓ La red virtual que sea configurada debe tener las siguientes características: • Ofrecer baja latencia, que no supere los 1ms, entre los recursos dentro de las diferentes redes virtuales requeridas • No debe tener tiempo de inactividad en los recursos de la red virtual al momento de crear la unión de las virtuales requeridas. • El tráfico entre redes virtuales emparejadas debe ser privada y encriptada. • Debe poder transferirse datos[3] entre las diferentes instancias o zonas asignadas de los servicios en nube.
BASE DE DATOS NO	✓ Servicio de base de datos NoSQL totalmente

[3] Un paquete de datos es una unidad de datos hecha en un solo paquete que viaja a lo largo de una ruta de red dada. Los paquetes de datos se utilizan en transmisiones de Protocolo de Internet (IP) para datos que navegan por la Web y en otros tipos de redes.

DENOMINACIÓN	CARACTERÍSTICAS
RELACIONAL	administrado para el desarrollo de aplicaciones. ✓ Tiempos de respuesta garantizados de milisegundos de un solo dígito y disponibilidad del 99,9994 por ciento, en caso de incumplimiento se aplica la penalidad. ✓ Escrituras y lecturas de los datos en cualquier parte del mundo con la capacidad de poder replicar de datos y/o escrituras a diferentes regiones. ✓ Las características mínimas incluidas como aprovisionamiento para la base de datos no relacional serán las siguientes: ● Escritura/Lectura en 01 única región geográfica. ● Un mínimo de 01 GB de espacio contratado para el ambiente transaccional ● Se debe mantener la reserva de capacidad hasta el final del periodo contratado para el mantenimiento o ampliación de la capacidad reservada. ● Debe considerar una (1) copia periódica (quincenal) de respaldo, el cual debe ser verificable a través de la consola o panel de administración del servicio de nube. ● El tamaño mínimo del bloque de datos es de 45 KB. ● Se debe proporcionar como un servicio bajo demanda, con un mínimo de mil solicitudes por segundo. **Cálculo de la disponibilidad de la base de datos no relacional** El OEFA calcula la disponibilidad, de forma mensual, para la base de datos no relacional, de acuerdo al siguiente detalle: Disponibilidad = ((TT – TE) x 100) / TT Dónde: TT = Cantidad de minutos del mes – brindadas por el proveedor – para la provisión del servicio de base de datos no relacional. TE = Total de minutos sin servicio en el mes. Ejemplo: Si el servicio de base de datos no relacional tuviera 5 caídas en 1 mes de 1 hora de duración cada caída por causas atribuibles al proveedor, la disponibilidad será: TT = 60x24x30 (en 1 mes con 30 días calendario) = 43200

[4] Para verificar el nivel de cumplimiento de la disponibilidad del 99,999% de la base de datos no relacional, se requiere una herramienta de monitoreo que permita visualizar el rendimiento, los logs y la disponibilidad del servicio ofertado.

DENOMINACIÓN	CARACTERÍSTICAS
	minutos TE = 5 horas = 300 minutos Disponibilidad = ((43200 – 300) x 100) / 43200= 99.3% De presentarse una afectación al servicio por causas externas ajenas al proveedor, se debe sustentar mediante un informe presentado a través de mesa de partes virtual o presencial en la sede central del OEFA y dirigido a la OTI, para su verificación, y de corresponder, no configurarse un supuesto de penalidad.
CACHÉ DE BASE DE DATOS	✓ El servicio de caché de Base de Datos debe permitir acelerar la capa de datos mediante el almacenamiento en caché que permita mejorar el rendimiento para atender millones de solicitudes por segundo con una latencia inferior a un milisegundo. ✓ Dicho servicio debe tener la capacidad de proveer el escalamiento de aplicaciones sin tener que rediseñar la base de datos. ✓ El servicio de caché de base de datos debe permitir la configuración en modo clúster habilitado en dos o más nodos. ✓ La capacidad mínima requerida por instancia productiva es de 200 MB de Caché.
MENSAJERÍA	✓ El servicio de mensajería debe ser completamente administrado y debe tener la capacidad de desacoplarse y ajustarse a la implementación de microservicios, sistemas distribuidos y aplicaciones sin servidor. ✓ El servicio de mensajería debe contemplar: ● Equilibrio de carga del trabajo ● Enrutamiento y transferencia de datos de forma segura y control entre los límites de aplicaciones y servicios ✓ El servicio debe poder soportar inicialmente para el ambiente de producción no menos de un millón de operaciones de mensajería al mes.
FIREWALL	✓ Servicio administrado que facilita la implementación de protecciones de red básicas. ✓ Se debe poder definir reglas de firewall que le brindan un control sobre el tráfico de la red. ✓ Debe tener la capacidad de importar reglas en formatos de reglas de código abierto comunes. ✓ Debe incluir características que brindan protección contra las amenazas comunes de la red. ✓ El firewall debe poder observar los flujos de tráfico,

DENOMINACIÓN	CARACTERÍSTICAS
	como el seguimiento de conexiones y la identificación de protocolos, para hacer cumplir políticas como evitar que la red virtual pueda ser accedida a dominios mediante un protocolo no autorizado. ✓ El sistema de prevención de intrusiones (IPS) debe proporcionar una inspección activa del flujo de tráfico para que pueda identificar y bloquear las vulnerabilidades mediante la detección basada en firmas. ✓ Se debe considerar como mínimo un (01) firewall por cada Subnet que es parte de la red virtual de datos
NIVELES DE SOPORTE	✓ Los niveles de soportes deben ser del tipo 24 x 7 provisto por la marca propuesta por el proveedor y en idioma español. ✓ Asesoramiento, configuración y resolución de problemas de interoperabilidad de software de terceros como sistemas operativos, plataformas y componentes de pilas de aplicaciones comunes. ✓ Acceso sin interrupción las 24 horas de los siete (7 días) de la semana al servicio de soporte técnico por correo institucional y/o vía telefónica. ✓ Tiempo de respuesta para atención de incidentes: ● Impacto mínimo: tiempo de atención en un plazo de treinta (30) minutos. ● Impacto moderado: tiempo de atención en un plazo de treinta (30) minutos. ● Impacto crítico: tiempo de atención en un plazo de treinta (30) minutos. ✓ Tiempo de respuesta para solución de incidentes: ● Impacto[5] mínimo: tiempo de solución[6] en un plazo de cuatro (4) horas. ● Impacto moderado: tiempo de solución en un plazo de dos (2) horas ● Impacto crítico: tiempo de solución en un plazo de una (1) hora. ✓ El soporte técnico debe permitir: ● Planificar implementaciones y migraciones ● Mejoras de rendimiento ● Buenas prácticas para mejorar la confiabilidad y la recuperabilidad de los servicios. ● Mejorar la configuración de seguridad. ✓ Es transversal a toda la infraestructura en nube a ser

[5] La modalidad de impacto se define en: *(i)* crítico si se detienen los servicios que imposibilita el uso de cualquier aplicativo alojado en el servidor de nube; *(ii)* moderado si se detiene uno o más servicios que imposibilita el uso de un aplicativo alojado en el servicio de nube y *(iii)* mínimo si se presenta algún fallo en algún servicio de la nube que no detiene el funcionamiento de cualquier aplicativo desplegado.

[6] El tiempo de solución inicia desde la notificación del incidente.

DENOMINACIÓN	CARACTERÍSTICAS
	provista por el proveedor. **Tiempo de atención:** Tiempo que transcurre desde el momento de reportado el incidente por parte de la OTI, el cual será a través de correo electrónico y/o vía telefónica, hasta que el proveedor responda consignando el Ticket de incidente para dar inicio a la solución, el cual debe ser comunicado al correo institucional de la persona que reportó el incidente. **Tiempo de solución:** Tiempo que transcurre desde que se consigna el ticket de incidente por parte del proveedor a la persona que reportó el incidente por parte de la OTI hasta la solución del mismo. La cual será de manera presencial y/o remoto determinado por el proveedor en coordinación con la OTI. En caso supere el tiempo de solución se aplica la penalidad indicada en el numeral 6.15 (OTRAS PENALIDADES) en la tabla Nº 12 "Por incumplimiento de los niveles de servicio del soporte"
IOT[7] DEVICE MANAGEMENT	✓ Las Implementaciones de IoT Device Management contienen diversos dispositivos tecnológicos, por lo que se realiza el seguimiento, monitoreo y administración de las flotas de dispositivos conectados a la red. Debe asegurarse de que sus dispositivos de IoT funcionen correctamente y de forma segura después de su implementación. También necesita proteger el acceso a sus dispositivos, controlar su estado, detectar y solucionar problemas de forma remota y administrar las actualizaciones de software y firmware. ✓ IoT Device Management facilita el registro, la organización, la monitorización y la administración remota de dispositivos IoT a gran escala. Asimismo, se puede registrar los dispositivos conectados de forma individual o masiva y administrar fácilmente los permisos para que los dispositivos permanezcan seguros. También puede organizar los dispositivos, monitorizarlos y solucionar problemas de funcionalidad en ellos, consultar el estado de cualquier dispositivo IoT de su flota y enviar actualizaciones de firmware inalámbricas (OTA), a través de una aplicación web totalmente administrada por el proveedor. IoT Device Management es independiente del tipo de dispositivo y del sistema operativo, lo que permite administrar dispositivos desde microcontroladores básicos hasta automóviles conectados, todo con el mismo servicio. IoT Device

[7] IoT proceso que permite conectar elementos físicos cotidianos al Internet

DENOMINACIÓN	CARACTERÍSTICAS
	Management le permite ajustar la escala de las flotas y reducir el costo y el esfuerzo necesario para administrar grandes implementaciones de dispositivos IoT heterogéneos. ✓ Se requiere como cuota mensual soportar veinte (20) dispositivos. ✓ Se requieren por dispositivo 30 000 paquetes de datos por mes. ✓ Se requiere una cuota de 1 KB por paquete de datos a ser indexados o procesados.
CHAT BOT	Es un servicio de inteligencia artificial (IA) administrado con modelos avanzados de lenguaje natural que sirve para diseñar, crear, probar e implementar interfaces de conversación en las aplicaciones. ✓ Se requiere soportar 1 millar de mensajes como cuota mensual.

3.1.2 VNET BACKEND: CARACTERÍSTICAS DE LOS SERVICIOS PARA LA RED VIRTUAL DE LA ZONA BACKEND

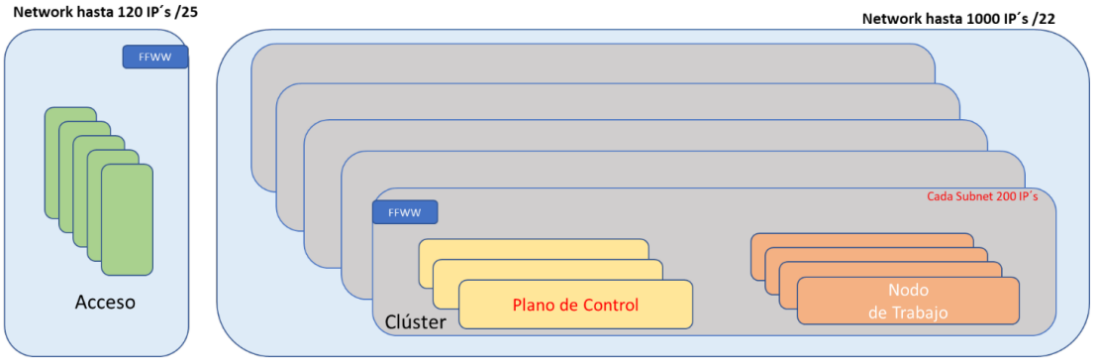
La Zona Backend de servicios contempla una plataforma portable y extensible de código abierto para administrar cargas de trabajo y microservicios. Dicha plataforma debe permitir automatizar la implementación, el escalado y la administración de aplicaciones en contenedores.

La implementación de esta plataforma debe tener en consideración la ejecución de versiones previas, hasta dos versiones anteriores por lo menos. La decisión de la versión final a implementar es evaluada por el/la Jefe/a de la OTI o a quien este designe, al momento del despliegue de los servicios.

Se debe tener en consideración que la configuración de los Pods que residen en los nodos de trabajo debe realizarse de tal manera que los microservicios que se implementarán estén asociados uno a uno. Cada espacio de nombres (o namespace) debe asignarse a cada aplicación que se despliegue.

Debe suceder con cada nueva aplicación para así mantener la división ordenada de los recursos entre múltiples clústeres. Como se aprecia en la figura N° 2.

Figura N° 2: Ejemplo de distribución por cluster



Como parte del servicio, cada clúster se añade a la infraestructura en nube según las necesidades del OEFA.

Todas las características requeridas para la zona backend hacen referencia al ambiente de producción, los ambientes de desarrollo y calidad[8] deben ser provisto por el proveedor bajo el siguiente criterio:

- Ambiente de Desarrollo = 15% de la capacidad configurada para el ambiente de producción
- Ambiente de Calidad = 25% de la capacidad configurada para el ambiente de producción

Tabla N° 3: VNET Backend

DENOMINACIÓN	CARACTERÍSTICAS
NETWORKING TRANSVERSAL VIRTUAL	✓ Este servicio debe considerar la configuración de la red para el servicio de BackEnd. ✓ Esta configuración comprenderá dos redes: • Red de Acceso CIDR /25 con subnets de 24 IP´s • Red de Cluster CIDR /22 con subnets de 200 IP´s ✓ La red virtual que sea configurada debe tener las siguientes características: • Ofrecer baja latencia, entre los recursos dentro de las diferentes redes virtuales requeridas • No debe tener tiempo de inactividad en los recursos de la red virtual al momento de crear la unión de las virtuales requeridas. • El tráfico entre redes virtuales emparejadas deberá ser privado

[8] Los ambientes de desarrollo y calidad deberán tener la posibilidad de habilitarse y deshabilitarse conforme a la demanda requerida.

DENOMINACIÓN	CARACTERÍSTICAS
	• Debe permitir la transferencia de datos entre las diferentes instancias o zonas asignadas de los servicios en nube.
NODO DE TRABAJO	✓ Los nodos de trabajo son los encargados de ejecutar las aplicaciones en los PODs. ✓ Los nodos deben incluir los agentes que permitan procesar las solicitudes de orquestación desde el plano de control y la programación de la ejecución de los contenedores solicitados en forma automatizada. ✓ El servicio de gestión debe evaluar el estado de salud de los contenedores en los nodos recopilando métricas de procesador, memoria, nodos, controladores del clúster y aplicaciones implementadas. ✓ El nodo de trabajo (Worker Node) debe cumplir las siguientes características mínimas: • Cantidad mínima de nodos =04 • Número de vCPUs = 2 • Memoria RAM = 08 GB • Almacenamiento = SSD no menos de 16GB • Operatividad = 24 horas • Configuración en alta disponibilidad[9] en diferentes zonas que permitan proteger los nodos en diferentes ubicaciones físicas. En caso de incumplimiento se aplicará la penalidad. ✓ Los nodos de trabajo son reservados entre las diferentes regiones seleccionadas por el OEFA (ver numeral 3 literal a) y a otros tamaños según cambien las necesidades de las cargas de trabajo.
PLANO DE CONTROL Y CONTROLADOR DE NUBE	✓ El Plano de Control (o Máster Node) proporciona los servicios básicos de aplicaciones basadas en contenedores y sus componentes de red y almacenamiento asociados. ✓ Los nodos del plano de control deben incluirse como parte del servicio de gestión de contenedores que permita reducir la complejidad de la implementación y de tareas básicas de administración, como la coordinación de actualizaciones. ✓ Debe integrarse con el servicio de Directorio Activo provisto por el proveedor, detallado en el

[9] La alta disponibilidad (HA) es la capacidad de garantizar la continuidad de los servicios, incluso en situaciones de deficiencias (es decir, hardware, software, corte de energía). Es decir, las características del sistema no se pueden detener.

DENOMINACIÓN	CARACTERÍSTICAS
	numeral 3.1.3 ✓ La seguridad y administración para el plano de control y demás recursos del clúster debe poder realizarse mediante el control de acceso basado en roles. Estas políticas deben controlar por lo menos el acceso a los recursos y al espacio de nombres. Los roles de acceso mínimos que deben ser configurados son: ● Rol Tipo I: Permite a los desarrolladores crear y actualizar flujos de trabajo, cuentas de integración y conexiones API. ● Rol Tipo II: Rol de acceso completo para administrar todos los recursos, incluida la capacidad de asignar roles. ● Rol Tipo III: Rol de tipo arquitecto para la gestión del clúster ✓ El servicio de gestión debe evaluar el estado de salud de los contenedores recopilando métricas de procesador, memoria, nodos, controladores del clúster y aplicaciones implementadas.
ACCESO	✓ Este servicio actúa como un controlador de ingreso que debe proporcionar la función de proxy inverso, enrutamiento de tráfico que pueda ser configurable, de tal forma que se pueda aplicarse reglas de entrada y rutas para los servicios del clúster. ✓ Los nodos deben incluir los agentes que permitan procesar las solicitudes de orquestación desde el plano de control y la programación de la ejecución de los contenedores solicitados en forma automatizada. ✓ El servicio de gestión debe evaluar el estado de salud de los contenedores en los nodos recopilando métricas de procesador, memoria, nodos, controladores del clúster y aplicaciones implementadas. ✓ Las características del nodo para soportar esta funcionalidad son: ● Cantidad mínima de nodos =02 ● Número de vCPUs = 2 ● Memoria RAM = 04 GB ● Almacenamiento = HD no menos de 128GB ● Operatividad = 24 horas
NETWORK FILE SYSTEM (NFS)	✓ Servicio de compartición de archivos entre los nodos del cluster. ✓ Los nodos deben incluir los agentes que permitan procesar las solicitudes de

DENOMINACIÓN	CARACTERÍSTICAS
	orquestación desde el plano de control y la programación de la ejecución de los contenedores solicitados en forma automatizada. ✓ El servicio de gestión debe evaluar el estado de salud de los contenedores en los nodos recopilando métricas de procesador, memoria, nodos, controladores del clúster y aplicaciones implementadas. ✓ Las características del nodo para soportar esta funcionalidad son: ● Cantidad mínima de nodos =01 ● Número de vCPUs = 8 ● Memoria RAM = 16 GB ● Almacenamiento = HD no menos de 128GB ● Operatividad = 24 horas
FIREWALL	✓ Servicio administrado que facilita la implementación de protecciones de red básicas. ✓ Se debe poder definir reglas de firewall que le brindan un control sobre el tráfico de la red. ✓ Debe tener la capacidad de importar reglas en formatos de reglas de código abierto comunes. ✓ Debe incluir características que brindan protección contra las amenazas comunes de la red. ✓ El firewall debe poder observar los flujos de tráfico, como el seguimiento de conexiones y la identificación de protocolos, para hacer cumplir políticas como evitar que la red virtual pueda ser accedida a dominios mediante un protocolo no autorizado. ✓ El sistema de prevención de intrusiones (IPS) debe proporcionar una inspección activa del flujo de tráfico para que pueda identificar y bloquear las vulnerabilidades mediante la detección basada en firmas. ✓ Se debe considerar por lo menos un firewall por cada Subnet que es parte de los servicios para la red virtual de la zona de backend
NIVELES DE SOPORTE	✓ Del tipo 24 x 7 provisto por la marca propuesta por el proveedor y en idioma español. ✓ Asesoramiento, configuración y resolución de problemas de interoperabilidad de software de terceros como sistemas operativos, plataformas y componentes de pilas de aplicaciones comunes. ✓ Acceso ininterrumpido las 24 horas de los 7 días de la semana al servicio de soporte técnico por

DENOMINACIÓN	CARACTERÍSTICAS
	correo electrónico y por teléfono. ✓ Tiempo de respuesta para atención de incidentes: ● Impacto mínimo; tiempo de atención en un plazo de 30 minutos ● Impacto moderado; tiempo de atención en un plazo de 30 minutos ● Impacto crítico; tiempo de atención en un plazo de 30 minutos ✓ Tiempo de respuesta para solución de incidentes: ● Impacto mínimo; tiempo de solución en un plazo de cuatro horas ● Impacto moderado; tiempo de solución en un plazo de dos horas ● Impacto crítico; tiempo de solución en un plazo de una hora. ✓ El soporte técnico debe permitir: ● Planificar implementaciones y migraciones ● Mejoras de rendimiento ● Buenas prácticas para mejorar la confiabilidad y la recuperabilidad de los servicios. ● Mejorar la configuración de seguridad. ✓ Es transversal a toda la infraestructura en nube a ser provista por el proveedor. **Tiempo de atención:** Tiempo que transcurre desde el momento de reportado el incidente por parte de OTI (el reporte del incidente será vía telefónica y/o por correo electrónico), hasta que el proveedor responda consignando el Ticket de incidente para dar inicio a la solución, el cual deberá ser comunicado al correo electrónico de la persona que reportó el incidente. **Tiempo de solución:** Tiempo que transcurre desde que se consigna el ticket de incidente por parte del proveedor a la persona que reportó el incidente (OTI), hasta la solución del mismo (presencial y/o remoto determinado por el proveedor en coordinación con OTI), en caso supere el tiempo de solución se aplicará la penalidad.

3.1.3 VNET HUB: CARACTERÍSTICAS DE LOS SERVICIOS PARA LA RED VIRTUAL DE LA ZONA HUB

La red virtual de la zona Hub involucra las siguientes aplicaciones:

- Firewall
- Firewall de Aplicaciones Web (WAF)
- Gateway de Aplicaciones (API GW)
- Servicio de gestión de API's

Debe incluir las herramientas de GESTIÓN DE CLAVES y MONITOREO. Así como herramientas de administración de identidades (AD) y un servicio de hospedaje para dominios (DNS)

Esta configuración cubre las necesidades del ambiente de producción. Para los ambientes de desarrollo y calidad la configuración es la siguiente:

- Ambiente de Desarrollo = 15% de la capacidad configurada ambiente el ambiente de producción.
- Ambiente de Calidad = 25% de la capacidad configurada para el ambiente de producción.

Tabla N° 4: VNET Hub

DENOMINACIÓN	CARACTERÍSTICAS
NETWORKING TRANSVERSAL VIRTUAL	✓ Este servicio debe considerar la configuración de la red para el servicio de HUB. ✓ Dicha configuración comprenderá lo siguiente: ● Red HUB con un CIDR /24 con 250 IP´s ✓ La red virtual que sea configurada debe tener las siguientes características: ● Ofrecer baja latencia, entre los recursos dentro de las diferentes redes virtuales requeridas ● No debe tener tiempo de inactividad en los recursos de la red virtual al momento de crear la unión de las virtuales requeridas. ● El tráfico entre redes virtuales emparejadas deberá ser privado ● Debe poder transferirse datos entre las diferentes instancias o zonas asignadas de los servicios en nube.
GATEWAY (VPN GW)	✓ Servicio que permite establecer conectividad entre la red de la Infraestructura en nube Pública y la Red de la Infraestructura On Premises del OEFA. ✓ El ancho de banda mínimo es 1 Gbps. ✓ Debe considerar 01 TB de datos a transferir (entrada y salida) a través de la configuración del Gateway. ✓ El proveedor debe realizar el enlace entre la nube pública y el centro de datos del OEFA de acuerdo a lo indicado en el numeral 3.1 literal A.

DENOMINACIÓN	CARACTERÍSTICAS
FIREWALL DE APLICACIONES WEB (WAF)	✓ La implementación del servicio de firewall de aplicaciones debe permitir el escalado automático en función de los cambios en los patrones de carga de tráfico. ✓ Debe considerarse que debe soportar la redundancia por zonas para la implementación del firewall de aplicaciones. ✓ Debe integrarse con el servicio de Directorio Activo provisto por el proveedor, detallado en el numeral 3.1.3 ✓ Debe tener integración con el servicio de gestión de claves. ✓ Debe configurarse la seguridad de la capa de transporte (TLS): ● Protocolos de transporte del lado de backend: TLS1.2 activo ● Protocolos de transporte del lado de cliente: TLS1.2 activo ✓ Debe proveerse la configuración de filtros IP. ✓ La recolección de eventos de los logs se realizará a través de la herramienta de monitoreo provisto por el proveedor, detallado en el numeral 3.1.3. ✓ La cantidad mínima de horas activas del firewall deben supera las 720 horas mensuales y no menos de 01TB de datos transferidos y/o procesados. ✓ Debe soportar no menos de 1000 conexiones persistentes.
GATEWAY DE APLICACIONES (API GW)	✓ Servicio de Gateway de Aplicaciones completamente administrado que facilita la creación, la publicación, el mantenimiento, el monitoreo de aplicaciones a cualquier escala. ✓ Esta capacidad debe realizarse a través de un equilibrio de carga de HTTP y control, de entrega, el cual debe tener compatibilidad con HTTP/2 o HTTP/1.0 o HTTP/1.1 o HTTP/1.2 ✓ La cantidad mínima de horas activas del firewall deben superar las 720 horas mensuales y no menos de 01 TB de datos transferidos y/o procesados.
FIREWALL	✓ Servicio administrado que facilita la implementación de protección de red básica. ✓ Se debe poder definir reglas de firewall que le brindan un control sobre el tráfico de la red. ✓ Debe tener la capacidad de importar reglas en formatos de reglas de código abierto comunes.

DENOMINACIÓN	CARACTERÍSTICAS
	✓ Debe incluir características que brindan protección contra las amenazas comunes de la red. ✓ El firewall debe poder observar los flujos de tráfico, como el seguimiento de conexiones y la identificación de protocolos, para hacer cumplir políticas como evitar que la red virtual pueda ser accedida a dominios mediante un protocolo no autorizado. ✓ El sistema de prevención de intrusiones (IPS) debe proporcionar una inspección activa del flujo de tráfico para que pueda identificar y bloquear las vulnerabilidades mediante la detección basada en firmas. ✓ Las capacidades del firewall deben poder brindar: inspección del tipo TLS, IDPS, filtro de URL y la denegación o permiso de acceso a páginas web. ✓ La cantidad mínima de horas activas del firewall deben superar las 720 horas mensuales y no menos de 01 TB de datos procesados o transferidos.
NIVELES DE SOPORTE	✓ Del tipo 24 x 7 provisto por la marca propuesta por el proveedor y en idioma español. ✓ Asesoramiento, configuración y resolución de problemas de interoperabilidad de software de terceros como sistemas operativos, plataformas y componentes de pilas de aplicaciones comunes. ✓ Acceso ininterrumpido las 24 horas de los 7 días de la semana al servicio de soporte técnico por correo electrónico y por teléfono. ✓ Tiempo de respuesta para atención de incidentes: ● Impacto mínimo; tiempo de atención en un plazo de 30 minutos ● Impacto moderado; tiempo de atención en un plazo de 30 minutos ● Impacto crítico; tiempo de atención en un plazo de 30 minutos ✓ Tiempo de respuesta para solución de incidentes: ● Impacto mínimo; tiempo de solución en un plazo de cuatro horas ● Impacto moderado; tiempo de solución en un plazo de dos horas ● Impacto crítico; tiempo de solución en un plazo de una hora. ✓ El soporte técnico debe permitir: ● Planificar implementaciones y migraciones ● Mejoras de rendimiento ● Buenas prácticas para mejorar la confiabilidad y la recuperabilidad de los servicios.

DENOMINACIÓN	CARACTERÍSTICAS
	● Mejorar la configuración de seguridad. ✓ Es transversal a toda la infraestructura en nube a ser provista por el proveedor. **Tiempo de atención:** Tiempo que transcurre desde el momento de reportado el incidente por parte de la OTI, el cual será a través de correo electrónico y/o vía telefónica, hasta que el proveedor responda consignando el Ticket de incidente para dar inicio a la solución, el cual deberá ser comunicado al correo institucional de la persona que reportó el incidente. **Tiempo de solución:** Tiempo que transcurre desde que se consigna el ticket de incidente por parte del proveedor a la persona que reportó el incidente, hasta la solución del mismo, de manera presencial y/o remoto determinado por el proveedor en coordinación con la OTI, en caso supere el tiempo de solución se aplicará la penalidad.
GESTOR DE CLAVES	✓ El servicio de gestión de claves debe permitir el cifrado automatizado de datos. Las características del servicio deben ser: ● El servicio de almacenamiento de llaves deberá realizarse por software. ● Nivel FIPS 140-2 ● Capacidad de soportar la opción de Bring your own key ● Longitud de clave asimétrica: RSA de 2048 bits, 3072 bits y 4096 bits ✓ Se debe configurar con la capacidad de soportar no menos de 21 millones de operaciones por mes.
HERRAMIENTA DE MONITOREO	✓ La herramienta de monitoreo debe almacenar y analizar toda la telemetría operativa en un almacén de datos centralizado, totalmente administrado y escalable. ✓ Debe tener la capacidad de integrarlo con herramientas como DevOps. ✓ Debe tener la capacidad de analizar y optimizar el rendimiento de la infraestructura, incluidas las máquinas virtuales (VM), el servicio de clúster, base de datos. ✓ Debe tener la capacidad, a través de la activación de paquetes, de poder efectuar el diagnóstico de problemas de enrutamiento. ✓ El servicio debe poder realizar el análisis de logs no menos de 01 GB/día de datos y debe retener seis (6) o más meses de logs.

DENOMINACIÓN	CARACTERÍSTICAS
	✓ Se debe considerar para el análisis toda VM que se haya requerido como parte de la solución. ✓ Asimismo, el análisis debe incluir el monitoreo de aplicaciones en vivo, detectando automáticamente anomalías en el rendimiento a través de herramientas de análisis que permitan diagnosticar problemas y comprender cómo trabaja la aplicación. ✓ La información debe ser provista en base a métricas o registros de información, se debe tener la capacidad de poder analizar no menos de 20 señales o indicaciones que provengan del componente de la solución a analizar.
SERVICIO DE DIRECTORIO ACTIVO	✓ Solución en la nube para la administración de la identidad y el acceso que combina servicios de directorio, administración del acceso a las aplicaciones (debe tener la capacidad de integrarse con aplicaciones SaaS) y protección de la identidad. ✓ El servicio debe considerar que los usuarios deben poder tener la capacidad de acceder a aplicaciones que sean parte de la red corporativa, la red interna y aplicaciones que sean desarrolladas en la nube. ✓ Asimismo, debe poder gestionar la administración de identidades de ambientes híbridos, así como el acceso (IAM) a autoservicio (restablecimiento de contraseña en autoservicio) y seguridad en la nube. ✓ Se tiene que considerar que el servicio debe ser provisto para no menos de 100 usuarios, con una cantidad de autentificaciones por hora superior a 3000 para más de 30000 objetos.
SERVICIO DE HOSPEDAJE PARA DOMINIOS	✓ El proveedor del servicio de nube debe ofrecer un servicio de resolución de nombres. ✓ El servicio de resolución de nombres debe permitir administrar y resolver nombres en una red virtual sin tener que crear y administrar una solución personalizada. ✓ Este servicio debe integrarse con los demás servicios que son parte del presente requerimiento para optimizar el proceso de implementación. ✓ El servicio debe poder configurarse con no menos de 20 zonas hospedadas y soporte a más de 1M de consultas DNS.

3.1.4 VNET FRONTEND: CARACTERÍSTICAS DE LOS SERVICIOS PARA LA RED VIRTUAL DE LA ZONA FRONTEND

Tabla N° 5: VNET Frontend

DESCRIPCIÓN	CARACTERÍSTICAS
NETWORKING TRANSVERSAL VIRTUAL	✓ Este servicio debe considerar la configuración de la red para el servicio de BackEnd. ✓ Esta configuración comprenderá una red: ● Red de Cluster CIDR /22 con subnets de 200 IP´s ✓ El proveedor deberá proveer para esta y cualquier red virtual solicitada un medio de conexión que brinde las siguientes características mínimas: ● Ofrecer baja latencia, entre los recursos dentro de las diferentes redes virtuales requeridas ● No debe tener tiempo de inactividad en los recursos de la red virtual al momento de crear la unión de las virtuales requeridas. ● El tráfico entre redes virtuales emparejadas deberá ser privado ● Debe poder transferir datos sin necesidad de tener el mismo tipo de suscripción entre las diferentes instancias o zonas con el proveedor de servicios en nube.
CONTENEDORES PARA APLICACIONES WEB	✓ El servicio para aplicaciones Web debe permitir implementar aplicaciones basadas en contenedores. Debe contemplar la herramienta para poder extraer imágenes de contenedores e implementarlas en cuestión de segundos a través de Git, GitHub y/o DevOps. ✓ Esta acción permitirá un escalado vertical y horizontal de forma automática en función de la aplicación. ✓ La plataforma se encargará de la aplicación de revisiones del sistema operativo, el aprovisionamiento de la capacidad de cómputo y memoria. ● Cantidad mínima de contenedores =05 ● Número de vCPUs = 1 ● Memoria RAM = 04 GB ● Almacenamiento = SSD no menos de 16GB ● Operatividad = 24 horas por día
SERVICIO DE APLICACIONES	✓ Servicio gestionado del tipo PaaS que permite a través de HTTP hospedar aplicaciones web. ✓ Tener la capacidad de soportar escalado

DESCRIPCIÓN	CARACTERÍSTICAS
	automático de aplicaciones y administración del tráfico. ✓ Las características del servicio deben contemplar los siguientes valores mínimos: ● Cantidad mínima de instancias =05 ● Número de vCPUs = 1 ● Memoria RAM = 1.5 GB ● Almacenamiento = 10 GB ● Operatividad = 24 horas por día ✓ Debe considerarse no menos de dos conexiones certificadas SSL basado en IP.
NIVELES DE SOPORTE	✓ Del tipo 24 x 7 provisto por la marca propuesta por el proveedor y en idioma español. ✓ Asesoramiento, configuración y resolución de problemas de interoperabilidad de software de terceros como sistemas operativos, plataformas y componentes de pilas de aplicaciones comunes. ✓ Acceso ininterrumpido las 24 horas de los 7 días de la semana al servicio de soporte técnico por correo electrónico y por teléfono. ✓ Tiempo de respuesta para atención de incidentes: ● Impacto mínimo; tiempo de atención en un plazo de 30 minutos ● Impacto moderado; tiempo de atención en un plazo de 30 minutos ● Impacto crítico; tiempo de atención en un plazo de 30 minutos ✓ Tiempo de respuesta para solución de incidentes: ● Impacto mínimo; tiempo de solución en un plazo de cuatro horas ● Impacto moderado; tiempo de solución en un plazo de dos horas ● Impacto crítico; tiempo de solución en un plazo de una hora. ✓ El soporte técnico debe permitir: ● Planificar implementaciones y migraciones ● Mejoras de rendimiento ● Buenas prácticas para mejorar la confiabilidad y la recuperabilidad de los servicios. ● Mejorar la configuración de seguridad. ✓ Es transversal a toda la infraestructura en nube a ser provista por el proveedor. **Tiempo de atención:** Tiempo que transcurre desde el momento de reportado el incidente por parte de OTI (el reporte del incidente será vía telefónica y/o por correo electrónico), hasta que el proveedor responda consignando el Ticket de incidente para dar inicio a la

DESCRIPCIÓN	CARACTERÍSTICAS
	solución, el cual deberá ser comunicado al correo electrónico de la persona que reportó el incidente. **Tiempo de solución:** Tiempo que transcurre desde que se consigna el ticket de incidente por parte del proveedor a la persona que reportó el incidente (OTI), hasta la solución del mismo (presencial y/o remoto determinado por el proveedor en coordinación con OTI), en caso supere el tiempo de solución se aplicará la penalidad.

3.2 TRANSFERENCIA DE CONOCIMIENTOS

Esta etapa se realiza después de la habilitación, configuración y puesta en operatividad de la infraestructura en nube. La transferencia de conocimientos consiste en:

a) El proveedor deberá realizar la transferencia de conocimiento de los servicios en nube que componen la infraestructura en nube implementada a personal designado por la OTI en la siguiente modalidad.

Tabla Nº 6: Modalidad de transferencia de conocimientos

	Modalidad	Detalle
1	Teórica	Documentación técnica y configuración de cada servicio configurado en la nube.
2	Práctica	Configuración de una aplicación informática designada por la OTI en los recursos asignados al ambiente de desarrollo de la nube.

b) El Proveedor debe proporcionar los manuales e instructivos, en forma impresa y digital en idioma español, un día calendario antes de la realización de la transferencia de conocimientos señalados en el numeral 3.2. literal a.

c) Debe incluirse las casuísticas más frecuentes para solución de incidencias.

d) Las personas que están a cargo de recibir la transferencia de conocimientos son designadas por la OTI, las cuales se encuentran en la etapa de “plan de trabajo”.

e) La transferencia de conocimiento debe ser remota, debe proporcionarse el material grabado y todo material que se usó en la capacitación en un drive a la OTI, con acceso a los usuarios que participaron por un periodo mínimo de un mes.

3.3 TRANSFERENCIA DE CREDENCIALES Y CUENTAS DE ACCESO

Esta etapa se realiza después de la transferencia de conocimientos y comprende la entrega del control y acceso exclusivo a la infraestructura en nube implementada por el proveedor al OEFA. Y consisten en:

a) Transferir todas las credenciales y cuentas de acceso relacionadas a la infraestructura en nube, así como todos los servicios en nube implementados en

la infraestructura en nube, detallados en el numeral 3, a la OTI.

3.4 INFORME DE CONSUMO MENSUAL DE LOS SERVICIOS EN NUBE

Esta etapa se realiza después de la transferencia de credenciales y cuentas de acceso (ver numeral 3.3), y consiste en generar un informe que detalle el consumo mensual de los servicios en la nube. Se debe realizar las siguientes actividades:

a) Elaborar un informe del consumo de los servicios en nube identificando los parámetros, el cálculo y el resultado total del período mensual correspondiente.

b) Presentar un reporte conteniendo todos los incidentes atendidos por el soporte por período mensual correspondiente.

c) Presentar un reporte conteniendo las horas, minutos y segundos de la no disponibilidad de los servicios en nube por período mensual correspondiente.

4. RECURSOS A SER PROVISTOS POR EL PROVEEDOR:

El Proveedor es responsable de suministrar cualquier recurso que requiera para la ejecución y cumplimiento de la prestación del servicio.

5. RECURSOS Y FACILIDADES A SER PROVISTOS POR LA ENTIDAD

El OEFA brindará el acceso al personal del proveedor al centro de datos de la Entidad para el cumplimiento del presente servicio.

Anexo N° 3

Estándar de Seguridad para el Desarrollo Seguro

1. OBJETIVO

Establecer lineamientos para el desarrollo seguro de las aplicaciones informáticas, a fin de fabricar productos seguros y confiables.

2. ALCANCE

Este documento se aplica al desarrollo y mantenimiento de todos los servicios, arquitectura, software y sistemas que se desarrollan en la organización.

3. TÉRMINOS Y DEFINICIONES

Caracteres Especiales: Aquellos caracteres que no son alfanuméricos como por ejemplo: <> ” ’ % ( ) & + \ / \’ \”.
Debugging: Proceso de encontrar, fijar y eliminar errores en un programa. [ISO/IEC 24765-2010]

Entropía. Medida de desorden, aleatoriedad o variabilidad en un sistema cerrado. [ISO/IEC 117706:2016]

FIPS 140-2. Estándar de seguridad de ordenadores de los Estados Unidos para la acreditación de módulos criptográficos. Federal Information Processing Standard 140-2: Security Requirements for cryptographic modules. [NIST SP 800-35]

Framework. Una estructura de soporte alrededor de la cual se puede construir algo. [Cambridge Dictionary]

Hash o función hash. Función que mapea cadenas de bits de longitud variable (pero generalmente delimitada) a cadenas de bits de longitud fija. [ISO/IEC 11770-6:2016]

Metadatos: datos acerca de los datos [Catálogo Nacional de metadatos IGP-PERÚ]

NIST. Es el Instituto Nacional de Estándares y Tecnología. National Institute of Standards and Technology.

Referer: Cabecera HTTP que identifica la dirección de la página web anterior de la que provenía el enlace a la página actual [RFC1945]

Request: Mensaje de solicitud de un cliente a un servidor en el protocolo HTTP [RFC 1945]

Salt. Valor utilizado como entrada para una función de derivación de clave, una función de expansión de clave o una función de extracción de clave, que podría no ser un secreto. [ISO/IEC 11770-6:2016]

Sanitización: Proceso de examinar un parámetro o documento y producir uno nuevo sin caracteres o valores considerados inseguros. [OWASP JAVA HTML SANITIZER PROJECT]

Timestamp : Marca de tiempo que es una secuencia de caracteres que registran la fecha y la hora de un evento [DOCS.MICROSOFT.COM]

Token: Firma cifrada que permite identificar al usuario [SESSION_MANAGEMENT_CHEAT_SHEET_ESPAÑOL]

Transport Layer Security (TLS). Protocolo de seguridad de capa de transporte diseñado para evitar las escuchas ilegales, la manipulación y la falsificación de mensajes. [RFC 8446]

URL: Localizadores Uniformes de Recursos utilizado para la localización y acceso a recursos a través de internet [RFC1738-es]

4. LINEAMIENTO DE SEGURIDAD PARA EL DESARROLLO SEGURO

4.1. Librerías y marcos de trabajo

4.1.1. Utilizar librerías y marcos de trabajo (Frameworks) de fabricantes o autores confiables que cumplan con los siguientes requisitos:
- Cuenta con repositorios oficiales.
- Cuenta con procesos de actualización.
- Cuentan con servicios de soporte.
- Se utilizan ampliamente.
- No han tenido fallos de seguridad de envergadura.

4.1.2. Mantener un inventario de librerías para el control de actualizaciones.

4.1.3. Utilizar librerías y frameworks de seguridad de aplicaciones informáticas como medida base en el desarrollo de aplicaciones.

4.2. Validación de datos

4.2.1. Toda la lógica del negocio debe estar del lado del servidor.

4.2.2. Realizar todas las validaciones de datos del lado servidor.

4.2.3. Identificar todas las fuentes de datos y clasificarlos como confiables o no confiables.

4.2.4. Validar todos los datos provenientes de fuentes no confiables (por ejemplo: bases de datos, secuencias de archivo, etc.).

4.2.5. Utilizar la librería o framework de validación de datos de entrada.

4.2.6. Estandarizar los juegos de caracteres apropiados, tales como UTF-8, para todas las fuentes de entrada.

4.2.7. Todas las fallas en la validación deben terminar en el rechazo del dato de entrada.

4.2.8. Determinar si el sistema soportará juegos de caracteres UTF-8 extendidos y, de ser así, validarlos luego de terminada la decodificación del UTF-8.

4.2.9. Verificar que los valores de la cabecera tanto en solicitudes como en respuestas contengan sólo caracteres ASCII.

4.2.10. Validar los datos que son redireccionados de otros sitios.

4.2.11. Validar rangos de datos.

4.2.12. Validar la longitud de los datos.

4.2.13. Validar toda entrada con una lista blanca que contenga una lista de caracteres aceptados.

4.2.14. Si es necesario, permitir el ingreso de algún carácter especial que pueda ser considerado peligroso. Asegurar de implementar controles adicionales tales como la codificación de la salida, API de seguridad y el registro del uso de tales datos a lo largo de la aplicación.

4.2.15. La librería o framework de seguridad debe realizar las siguientes validaciones:
- Compruebe si hay bytes nulos (%00).
- Compruebe si hay caracteres de nueva línea (%0d, %0a, \r, \n).
- Compruebe si hay caracteres de alteraciones de ruta “punto, punto, barra (../ o ..\).
- En los casos en que se soportan sets de caracteres UTF-8 extendidos, implemente representaciones alternativas tales como: %c0%ae%c0%ae/.

4.2.16. Para la salida de datos se deben pasar por la sanitización todos los flujos de datos, venga del contexto que sea, como HTML, JS, datos numéricos, etc.

4.3. Mecanismo de autenticación segura

4.3.1. Todos los recursos y páginas de la aplicación deben requerir autenticación excepto aquellas específicamente clasificadas como públicas.
4.3.2. Todos los controles de autenticación deben realizarse de lado del servidor.
4.3.3. Establecer y utilizar servicios de autenticación estándares y probados. Utilizar una implementación centralizada para todos los controles de autenticación.
4.3.4. Si la aplicación administra un almacenamiento de credenciales, se debe asegurar que se almacena el hash con sal (salt hash) de las contraseñas y que el archivo/tabla que guarda las contraseñas y claves sólo puede ser escrito por la aplicación (evitar el algoritmo de hash MD5).
4.3.5. El hash de las contraseñas debe ser implementado del lado del servidor.
4.3.6. Validar los datos de autenticación únicamente luego de haber completado todos los datos de entrada.
4.3.7. Las respuestas a los fallos en la autenticación no deben indicar cuál parte de la autenticación fue incorrecta.
4.3.8. Utilizar autenticación para conexiones a sistemas externos que involucren información o funciones sensibles.
4.3.9. Las credenciales de autenticación para acceder a servicios externos a la aplicación deben ser encriptadas y almacenadas en ubicaciones protegidas.
4.3.10. Nunca incluir en el código fuente ningún tipo de credenciales.
4.3.11. Utilizar únicamente los pedidos del tipo HTTPS POST para la transmisión de credenciales de autenticación.
4.3.12. Utilizar únicamente las conexiones encriptadas o datos encriptados para el envío de contraseñas que no sean temporales.
4.3.13. Contraseñas temporales como aquellas asociadas con reseteos por correo electrónico, pueden ser una excepción.
4.3.14. Los criterios para establecer una contraseña segura son:

- Longitud mínima de ocho (8) caracteres
- No estar asociadas a datos personales que permitan su deducción
- No usar contraseñas anteriores.
- Estar conformada por al menos tres (3) características de las siguientes: *(i)* caracteres alfabéticos en mayúsculas y/o en minúsculas, *(ii)* caracteres numéricos; y, *(iii)* caracteres especiales o extendidos.
- La validez de las contraseñas no podrá superar los seis (6) meses.
- Se realiza el bloqueo de la cuenta luego de cinco (5) reiterados intentos fallidos de inicio de sesión.

4.3.15. Para el ingreso de las contraseñas en las aplicaciones se debe utilizar el uso de teclado digital.
4.3.16. No se debe desplegar en la pantalla la contraseña ingresada.
4.3.17. Deshabilitar las cuentas luego de un número establecido de intentos inválidos de ingreso al sistema.
4.3.18. Aplicar los mismos niveles de control en el cambio y reseteo de contraseñas que los mismos niveles de control en la creación y autenticación de cuentas.
4.3.19. Contemplar un amplio rango de respuestas aleatorias para el reseteo de contraseñas por el mecanismo de *"preguntas secretas"*.

4.3.20. Si se utiliza un reseteo por correo institucional, únicamente enviar un link o contraseñas temporales a casillas previamente registradas en el sistema.
4.3.21. Las contraseñas y links temporales deben tener un periodo de validez no mayor a 24 horas.
4.3.22. Forzar el cambio de contraseñas temporales luego de su utilización.
4.3.23. Notificar a los usuarios cada vez que se produce un reseteo de contraseña.
4.3.24. Prevenir la reutilización de contraseñas.
4.3.25. Las contraseñas deben tener al menos un día de antigüedad antes de poder ser cambiadas, de forma que se pueda evitar ataques de reutilización de contraseñas.
4.3.26. Cambiar todos los usuarios y contraseñas por defecto o deshabilitar las cuentas asociadas.
4.3.27. Solicitar la re-autenticación de usuarios antes de la realización de operaciones críticas.
4.3.28. Utilizar autenticación multi-factor para las cuentas más sensibles o de mayor valor, es decir, cuentas con altos privilegios.
4.3.29. Si se utiliza un código de una tercera parte para la autenticación, se debe examinar, minuciosamente para asegurar que no se encuentre afectado por cualquier código malicioso.

4.4. Del manejo de sesiones y cookies

4.4.1. Utilizar los controles del servidor o del framework para la administración de sesiones. La aplicación solo debe reconocer estos identificadores como válidos.
4.4.2. La creación de identificadores de sesión solo debe ser realizada de lado del servidor.
4.4.3. Los controles de administración de sesiones deben utilizar algoritmos que generen identificadores con un alto grado de entropía.
4.4.4. Definir el dominio y ruta para las cookies que contienen identificadores de sesión autenticados con un valor apropiadamente estricto para el sitio.
4.4.5. La función de logout debe terminar completamente con la sesión o conexión asociada.
4.4.6. La función de logout debe estar disponible en todas las páginas protegidas por autenticación.
4.4.7. Establecer un tiempo de vida de la sesión lo más corto posible, no mayor a 15 minutos.[1]
4.4.8. Si una sesión fue establecida antes del login, cerrar dicha sesión y establecer una nueva luego de un login exitoso.
4.4.9. Generar un nuevo identificador de sesión luego de cada re-autenticación.
4.4.10. No permitir logeos concurrentes con el mismo usuario.
4.4.11. No exponer identificadores de sesión en URLs, mensajes de error ni logs.
4.4.12. Los identificadores de sesión sólo deben ser ubicados en la cabecera de la cookie HTTPS.
4.4.13. Proteger la información sobre las sesiones del lado del servidor implementando los controles de acceso apropiados.
4.4.14. Generar un nuevo identificador de sesión y desactivar el anterior de forma periódica.
4.4.15. Usar siempre HTTPS en lugar de cambiar entre HTTP y HTTPS, configuración del mecanismo de seguridad HSTS.
4.4.16. Configurar el atributo *"seguro"* para las cookies transmitidas sobre una conexión TLS.
4.4.17. Configurar las cookies con el atributo HttpOnly.

4.5. De los tokens

4.5.1. Los tokens de sesión deben generarse con algoritmos de cifrados fuertes (mínimo SHA256) y altamente resistentes a la predicción de valores.

[1] Industria de tarjetas de pago (PCI) Norma de seguridad de datos para las aplicaciones de pago.

4.5.2. Determinar un alto grado de entropía en el cifrado de la información del token.

4.6. Control de acceso

4.6.1. Utilizar una sola librería o framework para el control de autorizaciones en toda la aplicación. Se deben incluir librerías que llamen a servicios de autorización externos.
4.6.2. Los controles de acceso en caso de fallas deben actuar en forma segura.
4.6.3. Denegar todos los accesos en caso de que la aplicación no pueda acceder a la información de configuración de seguridad.
4.6.4. Requerir controles de autorización en cada solicitud o pedido, incluyendo aquellos creados por scripts en el servidor, librerías y pedidos desde cualquier componente desde el lado del cliente.
4.6.5. Separar lógica privilegiada de otro código de la aplicación.
4.6.6. Restringir acceso a ficheros u otros recursos, incluyendo aquellos fuera del control directo de la aplicación, únicamente a usuarios autorizados.
4.6.7. Restringir el acceso a URLs protegidas, solo a usuarios autorizados.
4.6.8. Restringir el acceso a funciones protegidas, solo a usuarios autorizados.
4.6.9. Restringir las referencias directas a objetos, solo a usuarios autorizados.
4.6.10. Restringir el acceso a servicios, solo a usuarios autorizados.
4.6.11. Restringir el acceso a información de la aplicación, solo a usuarios autorizados.
4.6.12. Restringir el acceso a usuario, atributos y política de información utilizada por los controles de acceso.
4.6.13. Restringir el acceso a información relevante de la configuración, solo a usuarios autorizados. Las reglas de control de acceso implementadas del lado del servidor y en la capa de presentación, deben coincidir.
4.6.14. Si existen datos de estado que deben ser guardados en el cliente, use cifrado y chequeos de integridad del lado del servidor para poder evaluar el estado.
4.6.15. Limitar el número de transacciones que un usuario común o un dispositivo puede desarrollar en un cierto período de tiempo.
4.6.16. No utilizar el header *"referer"* como verificación de autorización, ya que es posible modificarlo.
4.6.17. Cuentas de servicio o cuentas de soporte deben tener el mínimo privilegio.

4.7. Criptografía

4.7.1. Todas las funciones de criptografía de la aplicación deben ser implementadas del lado del servidor.
4.7.2. Todas las fallas en los módulos de criptografía deben ser capturados por excepciones y tener un tratamiento seguro.
4.7.3. Los módulos criptográficos del sistema deben cumplir con FIPS 140-2 o con su estándar equivalente.

4.8. Manejo de errores y Logs

4.8.1. No mostrar información sensible en respuestas de error, como detalles del sistema, identificadores de sesión o información de usuarios.
4.8.2. Utilizar manejadores de errores que no muestren información de debugging o de memoria.
4.8.3. Implementar mensajes de error genéricos y utilizar páginas de error adaptadas.
4.8.4. Liberar espacio de memoria en cuanto una condición de error ocurra.

4.8.5. La lógica para la gestión de errores debe estar asociada a que los controles de seguridad no permitirán acceso por defecto.
4.8.6. Todos los controles de logging deben estar implementados en sistemas confiables.
4.8.7. El logging de controles de acceso debe incluir tanto los casos de éxito como de falla.
4.8.8. Asegurar que los datos del registro de log contengan información importante.
4.8.9. Asegurar que los logs de entrada que incluyen información no segura no serán ejecutados en interfaces o aplicativos.
4.8.10. Restringir el acceso a los logs, solo a personal autorizado.
4.8.11. Utilizar una librería, framework o rutina estandarizada y centralizada para todas las operaciones de logging.
4.8.12. No guardar información sensible en logs, incluyendo detalles innecesarios del sistema.
4.8.13. Registrar en un log todas las fallas de validación.
4.8.14. Registrar en un log todos los intentos de autenticación, en particular los fallidos.
4.8.15. Registrar en un log todas las fallas en los controles de acceso.
4.8.16. Registrar en un log todos los eventos de intento de evasión de controles, incluyendo cambios en el estado de la información no esperados.
4.8.17. Registrar en un log todos los intentos de conexión con tokens inválidos o vencidos.
4.8.18. Registrar en un log todas las excepciones del sistema.
4.8.19. Registrar en un log todas las funciones administrativas, incluyendo cambios en la configuración de seguridad.
4.8.20. Registrar en un log, todas las fallas de conexión de TLS.
4.8.21. Registrar en un log las fallas de los módulos criptográficos.
4.8.22. Utilizar una función de hash para validar la integridad de los logs.

4.9. De los datos

4.9.1. El/La Oficial de Seguridad y Confianza Digital debe mantener clasificado y actualizado el inventario de datos.
4.9.2. Implementar el mínimo privilegio: restringir el acceso de los usuarios solamente a la funcionalidad, datos y sistemas de información que son necesarios para realizar sus tareas.
4.9.3. Proteger todos los almacenes temporales (en memoria o archivo) de los datos sensibles, eliminar todos los archivos y memoria de trabajo temporal tan pronto como no sean requeridos.
4.9.4. Encriptar toda la información altamente sensible almacenada, como datos del tarjetahabiente.
4.9.5. No almacenar contraseñas, cadenas de conexión u otra información sensible en texto claro o de forma que no sea criptográficamente segura del lado del cliente.
4.9.6. Remover los comentarios en el código de producción que sea accesible por el usuario, que puedan revelar información sobre los servidores o información sensible.
4.9.7. Remueva cualquier aplicación que no sea necesaria y la documentación de los sistemas que pueda revelar información útil para los atacantes.
4.9.8. No incluya información sensible en los parámetros del HTTP GET.
4.9.9. Deshabilite las funcionalidades de completar automáticamente en aquellos formularios que contienen información sensible, incluyendo la autenticación.
4.9.10. Deshabilite el almacenamiento temporal del lado del cliente de páginas que contienen información sensible.
4.9.11. La aplicación debe de soportar la eliminación de datos sensibles cuando los datos ya no son requeridos.

4.9.12. Implementar controles de accesos apropiados para los datos sensibles almacenados en el servidor como memoria temporal, archivos temporales y datos que solamente puedan ser accedidos por usuarios específicos del sistema.

4.10. De las comunicaciones

4.10.1. Implementar encriptación para todas las transmisiones de información sensible utilizando al menos TLS 1.2.
4.10.2. Los certificados TLS deben de ser válidos y contener el nombre de dominio correcto, no deben estar expirados y deberán ser instalados con los certificados intermedios si son requeridos.
4.10.3. Las conexiones TLS que fallen no deben de transformarse en una conexión insegura.
4.10.4. Utilizar conexiones TLS para todo el contenido que requiera acceso autenticado y para todo otro tipo de información sensible.
4.10.5. Utilizar TLS para las conexiones a sistemas externos que involucren funciones o información sensible.
4.10.6. Utilizar una única implementación estándar de TLS que se encuentre configurada correctamente.
4.10.7. Especificar los caracteres de codificación para todas las conexiones.

4.11. De los repositorios

4.11.1. Elegir un repositorio de código y control de versiones como estándar de uso en la Entidad.
4.11.2. No almacenar ninguna credencial en el repositorio de código y control de versiones.
4.11.3. Limpiar el historial del repositorio de código y control de versiones.
4.11.4. Utilizar doble factor de autenticación para las cuentas de acceso al repositorio de código y control de versiones.
4.11.5. No se deben compartir credenciales para las cuentas de acceso al repositorio de código y control de versiones.
4.11.6. Cualquier computador o dispositivo que tenga acceso al código fuente debe ser debidamente asegurado.
4.11.7. Revocar el acceso de los usuarios que ya no están trabajando en la Entidad.
4.11.8. Solo deben tener acceso al repositorio de código y control de versiones los usuarios que necesitan de acuerdo con su rol en la Entidad.
4.11.9. Para el acceso de aplicaciones escritas por terceros se validan los privilegios de acceso, la credibilidad del autor y su postura de seguridad.
4.11.10. Renueve periódicamente los tokens de acceso a servicios del repositorio de código y control de versiones.

4.12. De la base de datos

4.12.1. Utilizar autenticación segura a la base de datos.
4.12.2. Solo los usuarios autorizados tienen acceso a la base de datos.
4.12.3. A los usuarios se les otorgan los permisos mínimos necesarios para su función de trabajo en la base de datos. Los permisos se administran a través de roles o grupos, y no mediante concesiones directas a las ID de usuario cuando sea posible.
4.12.4. Las contraseñas seguras en la base de datos se aplican cuando es técnicamente posible, y las contraseñas de la base de datos se cifran cuando se almacenan en la base de datos o se transmiten a través de la red.
4.12.5. Las aplicaciones requieren inicio de sesión de base de datos individual / contraseña y roles cuando sea posible. Cuando no es posible, las cuentas de la aplicación pueden

ser utilizadas. Sin embargo, el ID de inicio de sesión y la contraseña deben estar protegidos en este caso, y esta información no existe en la estación de trabajo del cliente.

4.12.6. Las aplicaciones deben administrar los permisos y auditorías de los usuarios para cumplir con los requisitos de los Propietarios de datos.

4.12.7. Los objetos de la base de datos de usuarios con datos restringidos no tienen concesiones públicas cuando es posible. Documentar cualquier subvención pública si es necesario en bases de datos con datos restringidos.

4.12.8. Utilice consultas parametrizadas con tipos de datos fuertemente tipados.

4.12.9. Utilice validación de las entradas y codificación de las salidas y asegúrese de manejar los metadatos. Si esto falla, no ejecute el comando de la base de datos.

4.12.10. Asegúrese de que todas las variables tengan tipos de datos asociados.

4.12.11. La aplicación debe de utilizar el mínimo nivel de privilegios cuando accede a la base de datos.

4.12.12. Utilice credenciales seguras para acceder a la base de datos.

4.12.13. Las cadenas de conexión a la base de datos no deben de estar incluidas en el código de la aplicación. Las cadenas de conexión a la base de datos deben de estar en un archivo de configuración separado en un sistema confiable y debería de estar encriptado.

4.12.14. Utilice procedimientos almacenados para abstraer el acceso a los datos y elimine los permisos de las tablas en la base de datos.

4.12.15. Cierre la conexión a la base de datos tan pronto como sea posible.

4.12.16. Remueva o cambie todas las contraseñas administrativas por defecto. Utilice contraseñas fuertes o frases o implemente autenticación de múltiples factores.

4.12.17. Deshabilite todas las funcionalidades innecesarias de la base de datos (por ejemplo: procedimientos almacenados innecesarios, servicios no utilizados, paquetes de utilidades, instale solo el conjunto mínimo de funcionalidades y opciones requeridas (reduce el área de ataque)).

4.12.18. Eliminar el contenido innecesario incluido por el proveedor (por ejemplo: esquemas de ejemplo).

4.12.19. Deshabilitar las cuentas por omisión que no son necesarias para la operativa del negocio.

4.12.20. La aplicación debería conectarse a la base de datos con credenciales diferentes para cada nivel de confianza (por ejemplo: usuarios, usuarios solo lectura, invitados, administrador).

4.12.21. Las cuentas que no pertenecen a DBA no permiten otorgar roles o permisos en ningún entorno con datos restringidos (control de calidad, producción, desarrollo).

4.12.22. Las cuentas de la base de datos se bloquean después de un máximo de tres inicios de sesión fallidos.

4.12.23. Un informe de todos los derechos de acceso para los usuarios es proporcionado al propietario de los datos por los DBA de forma regular. Dos veces al año es el intervalo recomendado

4.13. De las APIs

4.13.1. Siempre use HTTPS para la comunicación de datos de las APIs.

4.13.2. Nunca exponga información en las URL, evite el uso del método GET.

4.13.3. Considere agregar un valor TIMESTAMP como encabezado en el REQUEST HTTP.

4.13.4. Todos los datos de entrada de la API deben utilizar los lineamientos de la sección 4.2 Validación de datos de este documento.

4.13.5. Utilizar el principio de mínimo privilegio para el acceso a las acciones a las que están autorizadas.

4.13.6. Las aplicaciones deben validar los derechos de acceso de la API a todos sus recursos para garantizar que estén permitidos.

4.14. Manejo de archivos

4.14.1. La aplicación no debe aceptar archivos grandes que puedan llenar el almacenamiento o provocar un ataque de denegación de servicio.
4.14.2. Exigir autenticación antes de permitir la transferencia de un archivo al servidor.
4.14.3. Permite transferir al servidor únicamente los tipos de archivo (por ejemplo: pdf, doc, entre otros) requeridos por la operativa del negocio.
4.14.4. Validar los tipos de archivo transferidos verificando la estructura de los encabezados. La validación del tipo de archivo únicamente por la extensión no es suficiente.
4.14.5. No guardar los archivos transferidos en el mismo contexto que la aplicación web. Los archivos deben almacenarse en un repositorio específico o en una base de datos.
4.14.6. Evite o restrinja la transferencia de archivos que puedan ser interpretados por el servidor web (por ejemplo: asp, php, jsp, etc).
4.14.7. Eliminar los permisos de ejecución a los archivos transferidos.
4.14.8. Utilizar el método de lista blanca de nombres y extensiones como validación de archivos.
4.14.9. No utilizar información provista por el usuario para la generación de redirecciones dinámicas.
4.14.10. No incluir en parámetros nombres de directorios o rutas de archivos, en su lugar utilizar índices que internamente se asocian a directorios o rutas predefinidas.
4.14.11. Nunca envíe la ruta absoluta de un archivo al usuario.
4.14.12. Asegúrese que los archivos y recursos de la aplicación sean de solo lectura.
4.14.13. Revise los archivos transferidos por los clientes en busca de virus y malware.

5. AUDITORÍA DE SEGURIDAD DE APLICACIONES

Los/as especialistas de la OTI cuentan con responsabilidades en el desarrollo y mantenimiento de los servicios, arquitecturas y software que se fabrican y despliegan en la Entidad, para lo cual deben cumplir con los criterios y disposiciones del presente documento.

La auditoría de cumplimiento se realiza durante y después del desarrollo de las soluciones de software, utilizando como herramienta la lista de verificación *"Checklist Lineamiento de Seguridad para el desarrollo seguro"*, para ambos casos se obtendrá el nivel de cumplimiento.

El/la Oficial de Seguridad y Confianza Digital verifica la realización de auditoría de seguridad de aplicaciones.

Estándar para el desarrollo de Aplicaciones desplegadas en Jboss

1. OBJETIVO

Establecer lineamientos para el desarrollo del Estándar de Programación de Sistemas aplicable a la construcción de un aplicativo informático en JBOSS entorno web.

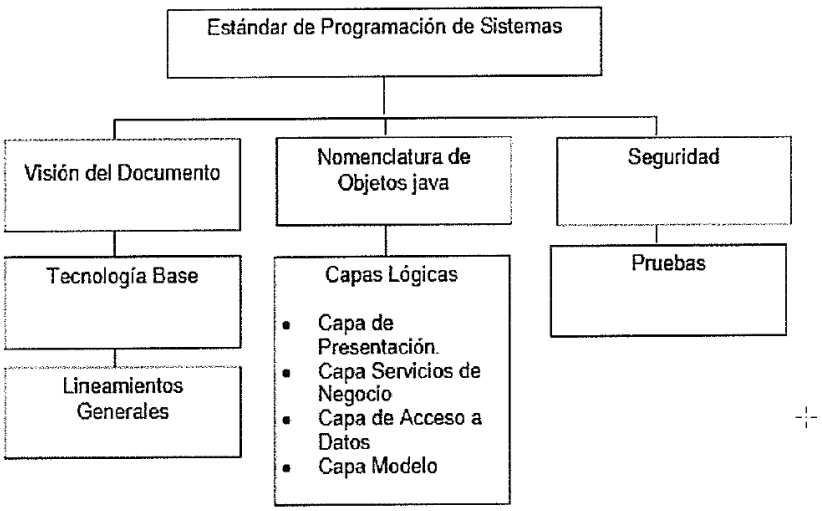
2. ALCANCE

Lo dispuesto en el presente documento es de aplicación obligatoria para el personal del equipo de desarrollo y contratistas de la Oficina de Tecnologías de la Información.

3. DEFINICIONES Y ACRÓNIMOS

Ítem	Acrónimo	Definición
1	JAVA EE	Java Enterprise Edition
2	JDK	Java Developer Kit
3	JVM	Máquina Virtual de Java
4	Framework	Marco de Trabajo
5	Log4J	Log For Java
6	SLF4J	Simple Logging Facade for Java
7	JDBC	Java Database Connectivity
8	SGBD	Sistema de Gestión de Base de Datos
9	JAX-WS	Java API for XML Web Services
10	IDE	Entorno integrado de desarrollo

4. ESTRUCTURA



5. TECNOLOGÍA BASE

La tecnología base que se utiliza en la construcción de un nuevo aplicativo informático en entorno web, de deberá desarrollar conforme a lo siguiente:

Tabla N°. 1: Tecnología Base

	Servicio	Tecnología
Entorno de Desarrollo	Lenguaje de Programación	Java EE
	Entorno Integrado de Desarrollo	Eclipse
	Sistema de Gestión de Base de Datos	Oracle 11g r2
	Control de versiones	Subversion, Gitlab
	Máquina Virtual	Java JDK 1.7 o superior
Arquitectura	Presentación y Servicios de Negocio	Capa Backend SpringBoot versión 1.5.21.RELEASE. Capa presentación Angular versión 8.
	Acceso a Datos y Modelo	JDBC, MyBatis
	Autenticación y Autorización	Java Security
	Trazas	Log4J
	Librería	Javascript JQuery
	Reportes	Jas JaspertReport, IReport
Integración	Servicio Web	Oracle JAX-WS
	Servicio de Gestión Documental	Alfresco
Herramientas	Documentos PDF	ÍText
	Documentos Excel	Apache POI
	Envío de Correo	Java Mail
	Ejecución de tareas planificadas	Quartz Scheduler
	Prueba Unitaria e Integración	JUnit
	Prueba de Aceptación	Selenium
	Pruebas de Servicios Web	SoapUI
	Prueba de Carga	JMeter
	Análisis estático de código	Checkstyle, PMD, CPD, Findbuqs y Sonar

6. LINEAMIENTOS GENERALES

6.1. PARÁMETROS Y VARIABLES

- El nombre debe estar compuesto por una o más palabras en sustantivo, con la primera letra de cada palabra en minúscula y el resto de primeras letras de cada palabra interna en mayúscula (IowerCamelCase).
- Debe nombrarse en singular (excepto variables que almacenen un conjunto de elementos).
- Se debe evitar nombrar usando un sólo carácter, excepto variables temporales usadas en bucles como pueden ser i, j, k, m, y n.
- Un parámetro de un método debe ser instancia de un objeto.

6.2. MÉTODO

- El nombre debe estar compuesto por una o más palabras en verbo infinitivo, con la primera letra de cada palabra en minúscula y el resto de primeras letras de cada palabra interna en mayúscula (IowerCamelCase).

6.3. CONSTANTE

- Debe nombrarse usando palabras en mayúsculas.

- Debe usarse el carácter subrayado "_" cuando el nombre de una constante tiene más de una palabra, por ejemplo:
- public static final String URL_SER VICIO —— "url";
- Se debe utilizar constantes en reemplazo de números mágicos y en circunstancias en las cuales el código es cambiante.

6.4. INDENTACIÓN

- Se deben emplear cuatro espacios como unidad de indentación. La construcción exacta de la indentación (espacios en blanco contra tabuladores) no se especifica.
- Los tabuladores deben ser exactamente cada 8 espacios (N° 4).

6.5. TAMAÑO DEL CÓDIGO FUENTE

- Se debe evitar el código repetido utilizando la refactorizacíón.
- Se debe evitar la codificación de excesiva funcionalidad dentro de un mismo método, clase o página web.
- Deben evitarse los archivos de gran tamaño que contengan más de 2000 líneas.
- Evitar las líneas de más de 80 caracteres, ya que no son manejadas bien por muchas herramientas.

6.6. ESTRUCTURA DE CONTROL LÓGICA

Se debe utilizar las estructuras For, While, if, ifElse, Swich para entender la secuencia de ejecución del bloque de código. Los bloques de código deben de estar englobados entre llaves y seguirán las siguientes reglas:

6.6.1. SENTENCIAS FOR

Una sentencia for debe tener la siguiente forma:

for (inicializacion,' condicion, actualización) { sentencias,"

Una sentencia for vacía (en la que todo el trabajo se hace en las cláusulas inicialización, condición, y actualización) debe tener la siguiente forma:

for (inicializacion; condicion; actualización),

Al usar el operador coma en la cláusula de inicialización o actualización de una sentencia for, evitar la complejidad de usar más de tres variables. Si se necesita, usar sentencias separadas antes de bucle for (para la cláusula de inicialización) o al final del bucle (para la cláusula de actualización).

6.6.2. SENTENCIAS WHILE

Una sentencia while debe tener la siguiente forma:

while (condición)

(sentencias;

Una sentencia while vacía debe tener la siguiente forma:

while (condición);

6.6.3. SENTENCIAS DO-WHILE

Una sentencia do-while debe tener la siguiente forma:

do {

sentencias;

) while (condición);

6.6.4. SENTENCIAS SWITCH

Una sentencia switch debe tener la siguiente forma:

switch

(condicion) (

case ABC:

sentencias,"

/ este caso se propaga */*

case DEF.

sentencias,"

break;

case XYZ.

sentencias;

break,

default:

sentencias;

break;

Cada vez que un caso se propaga (no incluye la sentencia break), añadir un comentario donde la sentencia break se encontraría normalmente. Esto se muestra en el ejemplo anterior con el comentario "este caso se propaga "/".

Cada sentencia switch debe incluir un caso por defecto. El break en el caso por defecto es redundante, pero prevé que se propague por error si luego se añade otro caso.

6.6.5. SENTENCIAS TRY-CATCH

Una sentencia try-catch debe tener la siguiente forma:

t/y *(*

sentencias,"

) catch (ExceptionClass e)

{ sentencias,"

Una sentencia try-catch puede ir seguida de un *finally*, cuya ejecución se ejecuta independientemente de que el bloque *ip* se haya completado con éxito o no.

try {

sentencias,"

) catch (ExceptionClass e) {

sentencias,"

) finally {

sentencias,"

6.7. DECLARACIONES DE CLASS E INTERFACES

Al codificar clases e interfaces de Java, se siguen las siguientes reglas de formato:

- Ningún espacio en blanco entre el nombre de un método y el paréntesis "(" que abre su lista de parámetros.
- La llave de apertura "(" aparece al final de la misma línea de la sentencia declaración.
- La llave de cierre "}" empieza una nueva línea indentada para ajustarse a su sentencia de apertura correspondiente, excepto cuando no existen sentencias entre ambas, que debe aparecer inmediatamente después de la de apertura.

  "(" *class Ejemplo extends Object {*
  int ivar1;
  int ivar2,
  Ejemplo(int i, int j) (
  ivar1 —— i;
  ivar2 —— j,'
  int metodo Vacio() {}

- Los métodos se separan con una línea en blanco

6.8. ESPACIOS EN BLANCO

Se utilizará un espacio en blanco en los siguientes casos:

- Separar una palabra clave y su paréntesis asociado. Se aplica a las siguientes palabras clave: catch, for, if, switch, synchronized, while.
- Separar una palabra clave que lleva un argumento, por ejemplo: return true
- Entre dos palabras clave.
- Entre una palabra clave o un paréntesis de cierre y una apertura de llaves "{".
- Después de una coma en una lista.
- Después de un punto y coma en una sentencia for: for (expr1; expr2; expr3)(.
- No se utilizará un espacio en blanco en las siguientes situaciones:
  - Entre el nombre de un método y la apertura de paréntesis
  - Después de una apertura o cierre de paréntesis o corchetes.

6.9. LÍNEAS EN BLANCO

Se utilizarán líneas en blanco en los siguientes casos:

- Para separar los comentarios de cabecera del fichero, sentencia *package* y sentencias *import.*
- Entre declaraciones de clases.
- Entre declaraciones de métodos.

- En una clase, entre la última variable y el primer método.
- Para agrupar secciones de código que estén lógicamente relacionadas.

6.10. BÚSQUEDA DE INFORMACIÓN

- Se debe ejecutar una búsqueda de información sin discriminar mayúsculas, minúsculas o acentuación.

6.11. ROMPIENDO LÍNEAS

Cuando una expresión no entre en una línea, separarla de acuerdo con estos principios:

- Separar después de una coma.
- Separar antes de un operador.
- Preferir roturas de alto nivel (más a la derecha que el “padre”) que de bajo nivel (más a la izquierda que el “padre”).
- Alinear la nueva línea con el comienzo de la expresión al mismo nivel de la línea anterior.
- Si las reglas anteriores llevan a código confuso o a código que se aglutina en el margen derecho, indentar justo 8 espacios en su lugar.
- Saltar de líneas por sentencias /f deberá seguir generalmente la regla de los 8 espacios, ya que la indentación convencional (4 espacios) hace difícil ver el cuerpo.

Por ejemplo:

//NO USAR ESTA INDENTACION

if ((condicion 1 && condicion2)

|| *(condicion3 && condicion4)*

II' *condicion5 && condicion6)) (//MALOS SALTOS*

hacerAlgo(); //HACEN ESTA LINEA FACIL

DE OLVIDAR

//USE ESTA INDENTACION

if ((condicion 1 && condicion2)

(condicion3 && condicion4)

||*!(condicion5 && condicion6)) {*

hacerAlgo(),

6.12. CODIFICACIÓN

- Se debe adoptar el método de codificación UTF-8 para el código fuente en lugar de ISO-8859-1 (por defecto en MS Windows, no así en Linux o Mac OS), para evitar problemas posteriores a la hora de generar documentación desde el código fuente, como el Javadoc o el código fuente navegable (HTML).
- Se debe configurar el editor de texto IDE utilizado para que almacene el código fuente en UTF-8 (Sólo es necesaria en Windows porque Linux trabaja por defecto con UTF-8).

6.13. COMENTARIOS

Se definen dos tipos de comentarios:

6.13.1. COMENTARIOS DE IMPLEMENTACIÓN

Estos comentarios se utilizan para describir el código (“el cómo”), y en ellos se incluye información relacionada con la implementación, tales como descripción de la función de variables locales, fases lógicas de ejecución de un método, captura de excepciones, entre otros.

Distinguimos tres tipos de comentarios de implementación:

Comentarios de bloque:

Permiten la descripción de ficheros, clases, bloques, estructuras de datos y algoritmos.

Esto es un comentario de bloque

Comentarios de línea:

Son comentarios cortos localizados en una sola línea y tabulados al mismo nivel que el código que describen. Si ocupa más de una línea se utiliza un comentario de bloque. Deben estar precedidos por una línea en blanco.

/ Esto es un comentario de línea */*

// Esto es otro comentario de línea

Comentario a final de línea

Comentario situado al final de una sentencia de código y en la misma línea.

int contador —— 4 + 10; //Inicialización del contador

contador++; */* Incrementamos el contador */*

6.13.2. COMENTARIOS DE DOCUMENTACIÓN

Los comentarios de documentación, también denominados *“comentarios javadoc”*, se utilizan para describir la especificación del código, desde un punto de vista independiente de la implementación, de forma que pueda ser consultada por desarrolladores que probablemente no tengan acceso al código fuente[1]. Los comentarios de documentación describen clases Java, interfaces, constructores, métodos y atributos. Cada comentario de documentación se encierra con los delimitadores de comentarios /**...*/, con un comentario por clase, interface o miembro (método o atributo). Este comentario debe aparecer justo antes de la declaración:

** La clase Ejemplo ofrece ...*

public class Ejemplo {...

Cabe precisar, que las clases e interfaces de alto nivel son indentadas; es decir mover el bloque de la sentencia hacia la derecha insertando espacios o tabuladores, mientras que sus miembros los están. La primera línea de un comentario de documentación (/**) para clases e interfaces no está indentada,

subsecuentes líneas tienen cada una un espacio de indentación (para alinear verticalmente los asteriscos). Los miembros, incluidos los constructores, tienen cuatro espacios para la primera línea y 5 para las siguientes.

Si se necesita dar información sobre una clase, interface, variable o método que no es apropiada para la documentación, usar un comentario de implementación de bloque o de una línea para comentarlo inmediatamente después de la declaración. Por ejemplo, detalles de implementación de una clase deben ir en un comentario de implementación de bloque siguiendo a la sentencia *class,* no en el comentario de documentación de la clase.

Los comentarios de documentación no deben colocarse en el interior de la definición de un método o constructor, ya que Java asocia los comentarios de documentación con la primera declaración después del comentario.

7. NOMENCLATURA DE OBJETOS JAVA

El nombramiento de objetos Java debe cumplir lo siguiente:

- Debe ser claro y conciso específicamente relacionado a los objetos del negocio y la funcionalidad que implementan.
- No debe contener caracteres especiales, a excepción del carácter de subrayado "_" para nombrar una CONSTANTE.
- Se debe evitar el uso de abreviaturas que dificulten la comprensión del código (a excepción que se usen ampliamente como URL o HTML).

7.1. CANTIDAD POR LÍNEA

Se recomienda una declaración por línea, ya que facilita los comentarios. En otras palabras, se prefiere

int nivel, // nivel de indentación int
tam, // tamaño de la tabla

Antes que:

int level, size,"

No poner diferentes tipos en la misma línea. Ejemplo:

int foo, fooarray[j, //ERROR!

Los ejemplos anteriores usan un espacio entre el tipo y el identificador. Una alternativa aceptable es usar tabuladores, por ejemplo:

int level, // nivel de indentación

int size; //tamaño de la tabla

Object currentEntry; // entrada de la tabla seleccionada actualmente

7.2. INICIALIZACIÓN

Intentar inicializar las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir.

7.3. DECLARACIONES DE CLASS E INTERFACES

- Ningún espacio en blanco entre el nombre de un método y el paréntesis "(" que abre su lista de parámetros.
- La llave de apertura "{" aparece al final de la misma línea de la sentencia declaración- La llave de cierre "}" empieza una nueva línea indentada para

ajustarse a su sentencia de apertura correspondiente, excepto cuando no existen sentencias entre ambas, que debe aparecer inmediatamente después de la de apertura "{".

```
class Ejemplo extends Object (
    int ivar1,"
        int ivar2;
        Ejemplo(int i, int j) {
                ivar1 —— i,
                ivar2 —— j,

        int metodo Vacio() {}
```

- Los métodos se separan con una línea en blanco.

7.4. PAQUETE

- Se debe nombrar en minúscula y en singular.
- El paquete raíz será: pe.gob.oefa.
- Se debe aplicar la siguiente jerarquía de paquetes:

  pe.gob.oefa. **[APLICACION].[MODULO].[CAPA].[TIPO]**

Dónde:

- APLICACIÓN, es el nombre del aplicativo.
- MODULO, es el nombre de un módulo o subsistema del aplicativo. Es opcional dependiendo del tamaño del aplicativo.
- CAPA, representa la capa lógica como, por ejemplo: paquete persistencia.
- TIPO, son paquetes definidos por capa, por ejemplo: la capa persistencia incluye los paquetes dao, impl e interfase.

La estructura de paquetes del aplicativo debe ser:

ESTRUCTURA DE PAQUETES
Capa de Presentación WebC onte nt Css Ima gen Js Page [modulo 1] Admin [sub modulo 1] [registrarObjeto.xhtml] [listarObjeto.xhtml] [sub modulo 2] WEB- INF

Lib

Capa
Controlador
datasource.x
ml
log4j.properti
es pe
gob
oefa
[aplicación]
[modulo1]
Controller
[Objeto1Controller]
[modulo2]
Controller
[Objeto2Controller]

comun
recurso
[objeto.properties]

util
U
til.java
Constantes.java
AbstractBean.java
Servicio
[
Objeto1Servicio]
[Objeto2Servicio]

Capa
Modelo
[modulo
1]
Persitencia
Dao
AdministradoDAO.java]
Impl
AdministradoImpl.java]
Interfase

7.5. CLASE

- Debe estar compuesta por una o más palabras en sustantivo o adjetivo, con la primera letra de cada palabra en mayúscula (CamelCase).
- Todas las clases deben implementar la interface Serializable.
- Cuando una clase implementa una funcionalidad característica o un patrón de diseño definido, deben de indicarlo mediante un sufijo en el nombre de la clase, lo suficientemente descriptivo *para* favorecer la comprensión de *la* funcionalidad de la clase. Por ejemplo: CuentaSessionFacade.

Tabla N° 2

Nombramiento de Clases por Paquete

Paquete	Funcionalidad	Nombre de Clase
Mbean	Controlador	[nombre]MBean
Servicio	Servicio de comunicación con la interface de persistencia	[nombre]Servicio
persistencia.dao	Data Access Object (Comunicación con Base de Datos)	[nombre]DAO
persistencia.impl	Data Access Object Interface)	[nombre]Impl
persistencia.interfase	Data Access Object Interface)	[nombre]Interface
VO	Value Object (Transporte de datos entre capas)	[nombre]VO

7.6. INTERFACE

- Debe estar compuesta por una o más palabras en sustantivo o adjetivo, con la primera letra de cada palabra en mayúscula (CamelCase).
- La clase de tipo interfaz lleva el sufijo Interface.
- La clase que implementa una interfaz lleva el sufijo Impl. Por ejemplo:

Interface: AdministradoInterface
Implementación de Interface: AdministradoImpl

DOCUMENTACIÓN

- La documentación del código debe explicar y exponer su funcionalidad interna.
- Se debe incluir las etiquetas de Javadoc en todas las clases Java, tanto en sus métodos como en sus propiedades.
- Se debe incluir la etiqueta @autor en la cabecera de las clases Java.

CAPAS LÓGICAS

- La construcción de una aplicación web se debe enfocar en una arquitectura de capas lógicas y niveles físicos para separar distintos aspectos de la construcción como presentación, lógica de negocio, servicios y acceso a datos. Por tal razón la capa lógica consta de un servicio desarrollado en SpringBoot versión mencionada líneas arriba.
- Una capa se refiere a las distintas partes de código en que una aplicación web se divide desde un punto de vista lógico. En cambio, un nivel corresponde a la forma física en que se organiza una aplicación.
- Las capas inferiores deben brindar servicios a las capas superiores a través de una interfaz, es decir, una capa solamente debe utilizar lo que la interfaz de la capa inferior le brinda. La aplicación web debe dividirse en siguientes capas lógicas:

  - Capa de Presentación.
  - Capa de Servicios de Negocio.
  - Capa de Acceso a Datos.
  - Capa de Modelo (Entidades de Dominio)

Figura N° 1
Capas lógicas de la aplicación

Nivel	Capa Lógica	
Cliente	Presentación	Modelo
Servicio	Servicios de Negocio	
	Acceso a Datos	
Datos	Datos	

7.7. CAPA DE PRESENTACIÓN

En esta capa la aplicación debe recepcionar o presentar la información al usuario, y no deberá procesar datos ni tomar decisiones en ella.

- Se debe implementar los componentes de la tecnología base indicada en la Tabla N° 1 (Angular 8). Los componentes Angular a implementar son:

  ✓ Página html; es una página xhtml que usa etiquetas especiales definidas por la especificación JSF.
  ✓ Componente; es una clase typeScript que define propiedades y métodos (funciones) para los componentes de interfaz de usuario de las páginas.

7.7.1. PÁGINA SPA

- Se debe implementar como documentos XHTML con extensión ".xhtml".
- Se debe emplear el sistema de codificación UTF-8 (encoding = "UTF-8").
- Se debe ubicar en el folder (carpeta): Web Content > page > [modulo] > [sub modulo].
- Debe ser compatible con los navegadores de internet más comunes en sus versiones más actuales (Internet Explorer, Mozilla Firefox, Chrome).
- Una página se debe diseñar y modularizar de tal forma que no tome una dimensión excesiva de código.

7.7.2. NOMENCLATURA DE UNA PÁGINA

- Debe estar compuesta por una o más palabras, con la primera letra de cada palabra en minúscula y el resto de primeras letras de cada palabra interna en mayúscula (lowerCamelCase).
- Debe iniciar con una palabra en verbo y las demás palabras en sustantivo y debe nombrarse en singular.
- Es recomendable crear una página principal y páginas modales para implementar funcionalidad relacionada a la página principal.

7.7.3. TIPOS DE PÁGINAS

- Ayuda: Debe implementarse una página de ayuda explicando la funcionalidad de todas las páginas del aplicativo.
- Búsqueda: Debe contener los botones Buscar y Cancelar. Debe implementar paginación (por defecto de 10 en 10) en la tabla de resultados. Debe mostrar el total de registros encontrados después de ejecutar la búsqueda. Los campos de búsqueda deben ubicarse. Aprovechando el espacio horizontal y luego el vertical. En la parte superior derecha de la tabla

de resultados debe implementarse el botón *"Exportar a Excel"* y *"Exportar a PDF"*.

- Inicio de sesión: Debe contener el encabezado y el pie de página establecido para todas las páginas de la aplicación.

- Registro de datos: Debe contener los botones Grabar y Cancelar. Los campos deben ubicarse aprovechando el espacio horizontal y luego el vertical. Debe agruparse campos relacionados. Los campos de sólo lectura deben mostrarse como textos o deshabilitados.

7.7.4. ESTRUCTURA DE UNA PÁGINA

- Para estructurar las páginas se deben utilizar plantillas implementadas con facelets (incluido en el framework java server faces).

- Se debe estructurar la página de acuerdo a: Encabezado, menú, contenido y pie de página.

Encabezado:

- Debe contener el nombre de la aplicación (siglas en mayúsculas y el nombre completo en mayúsculas y minúsculas) en la parte superior central.

- Debe contener el logo de la institución en la parte superior izquierda; conservando las dimensiones reguladas por la oficina de comunicaciones y atención al ciudadano.

- Se debe agregar una barra de información que incluya:
  - ✓ En la parte izquierda: fecha y hora.
  - ✓ En la parte derecha: login del usuario que ingresó a la aplicación y la opción para *"Cerrar sesión"*. Si se define un tiempo de cierre de sesión automático entonces incluir el tiempo restante para el cierre de sesión.

Menú:

- Debe ubicarse en la parte lateral izquierda, debajo del encabezado.
- Debe mostrar de manera horizontal los sub menú de las opciones del menú lateral.

Contenido:

- Debe mostrar las páginas que implementan la funcionalidad del aplicativo.
- El contenido de las páginas debe implementar su tamaño en porcentaje. Pie de página:
- Debe ubicarse en la parte inferior de la página.
- Debe contener la lista de navegadores en la cual ha sido probada la aplicación, por ejemplo: Versión optimizada para Google Chrome, Microsoft Edge y navegadores Mac.
- Debe contener el horario de atención, el número telefónico, correo institucional de soporte informático, la dirección de la Entidad, el nombre del área de desarrollo de aplicaciones web y el año de implementación, por ejemplo:

Si necesita ayuda o asistencia técnica comuníquese de Lunes a Viernes, de 8:30 am a 5:30 pm al teléfono 01204-9900 anexo 6000 o al correo soporte@oefa.gob.pe. Av. Faustino Sánchez Carrión 603, Jesús María. OEFA. Oficina de Tecnologías de Información (OTI).

7.7.5. ASPECTO GRÀFICO

Está relacionado con la Interfaz Gráfica de Usuario

- La Entidad de la página (interfaz, formulario) debe considerar que los/as usuarios/as leen de arriba abajo y de izquierda a derecha.
- Se debe evitar el uso de animaciones FLASH porque distraen la vista de el/la usuario/a.
- Se debe aplicar los colores estándares de la Entidad en todas las páginas reportes del sistema a implementar.
- Se debe establecer una resolución (ancho, alto) máxima recomendada para mostrar información en las páginas de acuerdo al requerimiento solicitado.
- Se debe aplicar texto explicativo en partes de la página que faciliten el entendimiento al usuario, por ejemplo: Tooltip text o leyendas.
- Se debe aplicar paginado, scroll horizontal y vertical, columnas fijas cuando la información a mostrar supere la resolución máxima establecida de la página.
- Se debe utilizar los mismos textos de los botones que implementan la misma funcionalidad en distintas pantallas de la aplicación.
- El menú principal se debe ubicar siempre en el lado superior y su sentido debe ser de izquierda a derecha para facilitar la lectura del usuario.
- La ubicación de los botones de acción (buscar, grabar, cancelar) debe estar centrada y al final de un grupo de campos de ingreso de datos.
- El botón para cerrar de la página, debe usar el texto “Salir”, una imagen que exprese la acción y debe ubicarse en la parte superior derecha.
- Se debe utilizar el botón cancelar para detener un proceso de llenado continuo de información y regresar a su estado inicial o para ejecutar la acción de borrar los valores digitados en un grupo de campos de ingreso de datos.
- Se debe utilizar controles adecuados (menús de tipo acordeón, tipo Tab Panel, pestañas) para una mejor organización del contenido de la página.
- Se debe utilizar el carácter “(*)” para expresar que un dato es obligatorio y se incluirá al final de la página una leyenda con el significado del carácter.

7.7.6. FUNCIONALIDAD

- Se debe habilitar el uso del tabulador considerando el orden de arriba hacia abajo, izquierda a derecha de los campos contenidos en las páginas.
- Se debe implementar el uso de atajos de teclado, por ejemplo: Grabar (teclas ALT + G), Salir (tecla Escape) para ventanas emergentes, Ayuda (ALT + F1).
- Se debe mostrar en una leyenda las teclas de atajo implementadas en el aplicativo.

7.8. SE DEBE BLOQUEAR LOS BOTONES DE GRABAR, INICIAR SESIÓN O BUSCAR DESPUÉS DE EJECUTAR LA ACCIÓN PARA EVITAR DOBLE INVOCACIÓN.

7.8.1. RECURSOS ESTÁTICOS

- Los recursos estáticos (imágenes, estilos y javascript) debe ubicarse en la carpeta Web Content.
- Otros recursos estáticos como documentos relacionados al aplicativo deben ubicarse en el Gestor de Documentos ALFRESCO. Si son documentos como Resoluciones deben obtenerse del Gestor de Contenidos del Portal Web utilizando rutas relativas por código de identificación del documento.

IMÁGENES

- Se debe crear un sólo folder (carpeta, paquete) con el nombre imagen para contener las imágenes que se utilizarán en todas las páginas del aplicativo.
- Se debe utilizar imágenes con formatos png (cuando se requiera fondo transparente), gif (menor peso y pocos colores) y jpg (mejor calidad y resolución).
- Se debe utilizar imágenes significativas e intuitivas (ejemplo: impresora para imprimir) para expresar la misma funcionalidad en distintas pantallas de la aplicación y evitar imágenes duplicadas que representen lo mismo.
- Se debe crear imágenes considerando los lineamientos de diseño gráfico establecidas por la Oficina de Comunicaciones y Atención al Ciudadano.

ESTILOS

- Se debe crear un sólo folder (carpeta, paquete) con el nombre “css” para contener un solo archivo de estilos que se utilizará en todas las páginas del aplicativo.
- Se debe crear un archivo de estilos por cada dispositivo (navegador web, móvil, otros) donde se visualizará la aplicación.
- El archivo de estilos debe contener atributos relativos al aspecto de la página y no dejar ninguno al código: fuente, colores, tablas, controles visuales, menú.
- Se debe utilizar los estilos por defecto del richfaces implementados a través de un SKIN. Por defecto, el SKIN será “wine”.

JAVASCRIPT

- Se debe crear un sólo folder (carpeta, paquete) con el nombre “js” para contener un solo archivo de funciones de validación como ingreso de datos, cargar, ocultar, mostrar datos.
- Se debe utilizar la versión de javascript que sea compatible con todos los navegadores web en versiones recientes (Internet Explorer, Firefox, Chrome).

7.8.2. VALIDACIÓN DE ENTRADA

- Se debe utilizar los componentes de validación de entrada del framework JSF.
- Se debe validar el ingreso de datos por el tipo de dato (letras, números enteros o decimales), obligatoriedad, formato específico (correo electrónico, fecha, teléfono, etc.) o rango de valores.
- Se debe mostrar el mensaje de no cumplimiento de validación cuando se presione el botón grabar o buscar.
- Se debe mostrar los campos que no cumplieron la validación con un color rojo y debe enfocarse sobre los campos para ingresar el valor adecuado.
- Se debe controlar el tamaño y la cantidad de datos de los controles de entrada de datos para soportar el tamaño de almacenamiento en los campos

de la tabla de base de datos donde persistirán.

7.8.3. MENSAJE DE CONFIRMACIÓN, ÉXITO Y ERROR

- Los mensajes de confirmación, éxito y error se deben ubicar en la parte central de toda la pantalla a fin de que se fácil de ubicar.
- Se debe implementar mensajes de confirmación para las acciones de grabar, actualizar y eliminar, utilizando oraciones en pregunta, por ejemplo:
- ¿Está seguro de grabar los datos?
- Se debe mostrar un mensaje de éxito de color verde, cuando se graba o actualiza datos, por ejemplo: *"Se grabó correctamente los datos"*. Debe personalizarse el mensaje indicando el nombre del objeto de negocio sobre el cual se ejecutó la acción.
- Se debe mostrar un mensaje de error de color rojo, cuando se interrumpe o existe error durante las acciones de grabar, actualizar, eliminar o buscar datos; por ejemplo: *"Hubo un error y no se pudo registrar correctamente la información"*.

7.8.4. REPORTE

- Debe contener en la parte superior izquierda el logo de la organización, el título del reporte en el centro y la denominación del año actual en la parte derecha.
- Debe mostrar la fecha de impresión en la parte inferior izquierda y el texto *"página actual / total de páginas"* en la parte inferior derecha.
- Por defecto, los reportes se deben generar en formato PDF.

7.8.5. PAGINACIÓN

- Se debe implementar un esquema de paginación que evite el tráfico de grandes volúmenes de datos y facilite la navegación.
- Para grandes volúmenes de data, se recomienda realizar la paginación por base de datos utilizando el método basado en el valor (Value based method).
- Por defecto, los resultados de las páginas deben paginarse de 10 en 10.

7.9. CAPA DE SERVICIOS DE NEGOCIO

7.9.1. SERVICIO

- El acceso a la capa de modelo desde una clase managed bean se debe implementar a través de una clase servicio evitando el acceso directo a las clases DAO.
- Se debe implementar la capa de presentación utilizando el framework JSF, richfaces para los controles visuales y Ajax, facelets para la definición de plantillas.
- Se debe implementar páginas con extensión XHTML.

7.10. CAPA DE ACCESO A DATOS

7.10.1. OBJETO DE ACCESO A DATOS (DAO)

El patrón DAO permite encapsular el acceso, manipular datos, gestionar la conexión con la fuente de datos (bases de datos relacionales, ficheros planos, sistema de ficheros remotos, etc.) para obtener y almacenar dichos datos. Las clases que se implementan con el patrón DAO no tienen estado, no almacena en caché resultados de ninguna consulta, ni datos que el cliente pueda necesitar posteriormente. Esto provoca que los objetos DAO sean simples y

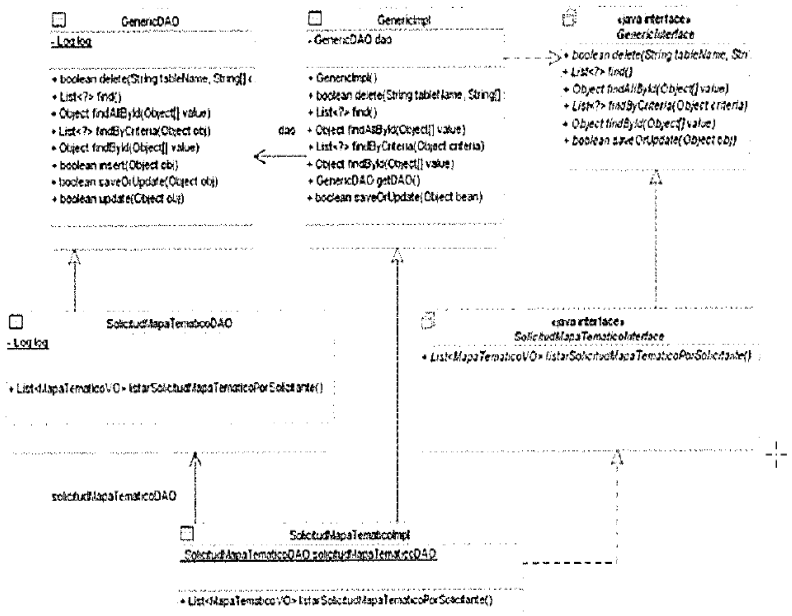
evita problemas potenciales de concurrencia.

- Se deben crear las siguientes clases en los paquetes indicados para implementar acceso a datos de un objeto:

PAQUETE		CLASE	
[modulo'	.persistencia.interfase	objeto	Interface
[modulo	.persistencia.impl	[objeto	Impl
[modulo	.persistencia.dao	[objeto	DAO

- Las clases DAO de tipo Interface e Impl (implementación de la interface) deben heredar de las clases GenericoInterface y GenericoImpl (contienen implementados métodos básicos) respectiv amente y sólo implementar métodos que sean específicos de las clases DAO.

- Se debe incorporar en las clases GenericoInterface y GenericoImpl los métodos comunes a todas las clases DAO que se identifiquen durante la construcción del aplicativo informático.



Los métodos de operación de las clases DAO deben de seguir la nomenclatura:

OPERACIÓN	NOMENCLATURA
Listar	listar(Objeto]
Insertar	insertar[Objeto]
Modificar	modificar[Objeto]
Eliminar	eliminar[Objeto]
Obtener	obtener[Objeto]

7.10.2. EJECUCIÓN DE CONSULTAS

- No se debe incrustar sentencias SQL en los métodos de las clases DAO.
- Se debe evitar el uso de consultas nativas.

7.10.3. LLAMADA A PROCEDIMIENTOS ALMACENADOS

- Se debe ejecutar operaciones de las clases DAO utilizando llamadas a procedimientos almacenados.

7.10.4. MANEJO DE EXCEPCIONES

- Se debe tratar las excepciones de este nivel (Exception, SQLException) en la capa controlador a través de la clase ServiceExceptíon.

7.11. SEGURIDAD

Se tienen las siguientes consideraciones de seguridad:

- Implementar tablas de auditoría para la identificación de las actualizaciones realizadas en el expediente electrónico durante su ciclo de vida.
- Utilizar las librerías actualizadas para evitar las vulnerabilidades en la aplicación y mantener la seguridad del sistema.
- Implementar etiquetas de seguridad en la cabecera del response de la aplicación.
- Incluir al inicio de Sesión el código Capcha y asegurar que este funcione en el Sistema.
- Si la aplicación utiliza la cabecera *"Strict-Transport-Security"* para forzar el uso de HTTPS, entonces se deberá configurar la cabecera *"Strict-Transport-Security"*.
- Verificar que las referencias a invocación de páginas existan, sin embargo, si la respuesta es una página que no existe, entonces devolverá una página de error. Esta página de error no debe contener el número de versión del servidor web y lista de módulos habilitados en este servidor, para este caso deberá manejarse con una página estándar de error.
- Deshabilitar el autocomplete del ingreso de contraseña del usuario (la aplicación debe obligar el ingreso de la contraseña cada vez que inicie sesión).
- La aplicación deberá eliminarse el código o no usado o comentado en las páginas HTML (front end), las cuales podrían revelar información importante para el aplicativo.
- La seguridad y autenticación de usuarios deberá realizarse mediante la integración al componente SSO (Single Sign-On) del OEFA. Esto implica la creación del menú y opciones, así como su consumo en la aplicación mediante servicios REST.

7.11.1. TRATAMIENTO DE EXCEPCIONES

- Se debe usar la tecnología base de trazas (log4j) para capturar y trazar excepciones en un archivo log.
- No se debe mostrar los errores por consola (no usar System.out o System.err).
- El archivo log debe contener diferentes tipos de información:
  - ✓ Todos los eventos de autenticación (inicio de sesión, cierre de sesión, intento de acceso fallido, etc.) que permitan detectar ataques de fuerza bruta y también ataques por adivinación.
  - ✓ Información significativa, como el nombre del método y los parámetros recibidos por el método.
- Se debe usar try y varios catch para capturar y trazar todas las excepciones, desde el más específico hasta el menos específico.
- Se debe usar finally para liberar recursos, como conexiones o ficheros.
- Se debe evitar concatenar los mensajes de la excepción, es decir, se debe mostrar toda la excepción.
- Se debe evitar declarar un método como throws Exception, ya que enmascara todas las excepciones, sin permitir gestionar el error concreto.
- Si es necesario personalizar el tratamiento de una excepción, se debe crear una

clase que herede de la clase Exception.
- Se debe evitar capturar y trazar excepciones constantemente o la misma excepción en distintos métodos de distintas capas.
- Se debe evitar trazar excepciones en los métodos de la capa de presentación para que no se muestre el error en el navegador.
- Se debe evitar presentar al usuario una traza o información técnica de la excepción ocurrida.
- Se debe evitar la presentación al usuario de los mensajes de error detallados, tales como trazas que devuelven información privada, es decir, se debe utilizar un mensaje de error genérico inclusive para los códigos de respuesta de estado de http (404 o 500).

7.11.2. VALIDACIÓN DE DATOS

La validación apropiada de la entrada de datos a una aplicación evita ataques de inyección, ataques locale/Unicode, ataques al sistema de archivos y desbordamientos de memoria.
- Se debe prevenir la manipulación de parámetros, revisando los mismos antes de utilizarlos por primera vez.
- Se debe evitar el uso de campos ocultos para almacenar información sensible.
- Se debe realizar validaciones en el servidor previamente a utilizarlos, siempre que sea necesario debido a que la validación del lado del cliente (Javascript) no es completamente segura y puede ser vulnerada.
- Se debe validar la entrada del usuario a los archivos de registro (histórico, transacciones, archivo log) para evitar su manipulación o la inserción de contenido malicioso.
- Se debe controlar el valor de retorno de los métodos para evitar que un valor devuelto no sea el esperado y no pueda ser controlado.

8. PRUEBAS

La ejecución de pruebas de software asegura que un aplicativo informático cumpla las especificaciones funcionales y no funcionales requeridas por los usuarios del aplicativo.

- Se debe ejecutar pruebas durante todo el ciclo de vida de la construcción del aplicativo informático y lo más temprano posible para detectar y corregir los defectos y errores aplicando la refactorización del código.

8.1. TIPOS DE PRUEBAS

Los tipos de pruebas que se deben aplicar durante todo el ciclo de vida de la construcción del aplicativo informático son:

8.1.1. PRUEBA UNITARIA

La prueba unitaria comprueba que una parte del código funcione de acuerdo a lo solicitado en el requerimiento funcional.
- Se debe implementar pruebas unitarias a la vez que se construye código del aplicativo.
- Se debe utilizar la herramienta de pruebas unitarias indicada en la Tabla N° 1: Tecnología Base.
- Las clases java de pruebas unitarias se deben ubicar en [modulo] > prueba y los recursos de configuración para ejecutar las pruebas se deben ubicar en [modulo] > prueba > recurso.
- Se debe retirar las clases de prueba de la aplicación cuando se despliega en ambiente de producción.

8.1.2. PRUEBA DE INTEGRACIÓN

La prueba de integración se ejecuta después que se aprobaron las pruebas unitarias con el objetivo de comprobar que todas las partes del código funcionen en conjunto y de acuerdo a lo solicitado en el requerimiento

funcional.

- Las clases java de pruebas unitarias se deben ubicar en [modulo] > prueba y íos recursos de configuración para ejecutar ías pruebas se deben ubicar en [modulo] > prueba > recurso.
- Se debe retirar las clases de prueba de la aplicación cuando se despliega en ambiente de producción.

8.1.3. PRUEBA DE ACEPTACIÓN

La prueba de aceptación permite al usuario final comprobar que todos los requerimientos funcionales de la aplicación estén implementados.

- Se debe utilizar la herramienta de pruebas de aceptación indicada en la Tabla N° 1: Tecnología Base.
- Las clases java de pruebas unitarias se deben ubicar en [modulo] > prueba y los recursos de configuración para ejecutar las pruebas se deben ubicar en [modulo] > prueba > recurso.
- Se debe retirar las clases de prueba de la aplicación cuando se despliega en ambiente de producción.

8.1.4. PRUEBA DE SERVICIOS WEB

La prueba de servicios web permite comprobar su correcto funcionamiento y automatizar dichas pruebas para repetirlas en cualquier momento.

- Se debe utilizar la herramienta de pruebas de servicios web indicada en la Tabla N° 1: Tecnología Base.
- Las clases java de pruebas de servicios web se deben ubicar en [modulo] > prueba y los recursos de configuración para ejecutar las pruebas se deben ubicar en [modulo] > prueba > recurso.
- Se debe retirar las clases de prueba de la aplicación cuando se despliega en ambiente de producción.

8.1.5. PRUEBA DE CARGA Y STRESS

La prueba de carga y stress evalúa el comportamiento y rendimiento del aplicativo en situaciones extremas, tanto en carga como en tiempo, donde los usuarios y peticiones se producen de forma concurrente.

Se debe utilizar la herramienta de pruebas de carga y stress indicada en la Tabla N° 1: Tecnología Base.

- Las clases java de pruebas de carga y stress se deben ubicar en [modulo] > prueba > jmeter y los recursos de configuración para ejecutar las pruebas se deben ubicar en [modulo] > prueba > recurso.
- Se debe retirar las clases de prueba de la aplicación cuando se despliega en ambiente de producción.

8.1.6. ANÁLISIS DE CÓDIGO ESTÁTICO

El análisis de código estático es la revisión del código por el propio programador que se ejecuta frecuentemente.

- Se debe utilizar la herramienta de análisis de código estático indicada en la Tabla N° 1: Tecnología Base.

8.2. PLAN DE PRUEBAS

El Plan de Pruebas es un documento que describe el alcance, enfoque, recursos y calendario de actividades de prueba.

- Se debe elaborar un plan de pruebas por el aplicativo o por cada módulo del aplicativo.
- El plan de pruebas debe incluir los tipos de pruebas, los datos utilizados y el resultado.

9. ENTREGA

- Se debe versionar el código fuente del aplicativo informático desde el inicio de su construcción utilizando la herramienta de Control de Versiones existente.
- Se debe realizar entregas parciales potencialmente funcionales para ser presentadas al usuario.
- AI terminar la construcción y pruebas del aplicativo informático se debe entregar las fuentes, el archivo ejecutable (WAR) y scripts necesarios utilizando el formato de Solicitud de Pase a Producción.

10. HÁBITOS DE PROGRAMACIÕN

10.1. PROPORCIONANDO ACCESO A VARIABLES DE INSTANCIA Y DE CLASE

No hacer ninguna variable de instancia o clase pública sin una buena razón. A menudo las variables de instancia no necesitan ser asignadas/consultadas explícitamente, a menudo esto sucede como efecto lateral de llamadas a métodos.

Un ejemplo apropiado de una variable de instancia pública es el caso en que la clase es esencialmente una estructura de datos, sin comportamiento. En otras palabras, si se utiliza la palabra struct en lugar de una clase (si Java soportara struct), entonces es adecuado hacer las variables de instancia públicas.

10.2. REFERENCIAS A VARIABLES Y MÉTODOS DE CLASE

Evitar usar un objeto para acceder a una variable o método de clase “static". Usar el nombre de la clase en su lugar. Por ejemplo:

metodoDeClase(), //OK

UnaClase. metodoDeClase(); //OK

unObjeto. metodoDeClase(); //EVITAR!

10.3. CONSTANTES

Las constantes numéricas (literales) no se deben codificar directamente, excepto -1, 0, y 1, que pueden aparecer en un “bucle for” como contadores.

10.4. ASIGNACIONES DE VARIABLES

Evitar asignar el mismo valor a varias variables en la misma sentencia. Es difícil de leer. Ejemplo:

fooBar. fChar —— barFoo.Ichar —— ’c’, // EVITAR!

No usar el operador de asignación en un lugar donde se pueda confundir con el de igualdad. Ejemplo:

if (c++ —— d++) {// EVITAR! (Java lo rechaza)

se debe escribir:

if ((c++ —— d++) !—- 0) {

No usar asignación embebidas como un intento de mejorar el *rendimient* en tiempo de ejecución. Ese es el trabajo del compilador. Ejemplo:

d -— (a —— b + c) + r, // EVITAR!

se debe escribir:

a = *b + c,*

d —- a +

r,

10.5. HÁBITOS VARIOS

10.5.1. PARÉNTESIS

En general es una buena idea usar paréntesis en expresiones que implican distintos operadores para evitar problemas con el orden de precedencia de los operadores. Incluso sí parece claro el orden de precedencia de los operadores, podría no ser así para otros, no se debe asumir que otros programadores conozcan el orden de precedencia.

if (a ——-— b && c ——-- d) // EVITAR!if ((a ------ b) && (c d)) // CORRECTO

10.5.2. VALORES DE RETORNO

Intentar hacer que la estructura del programa se ajuste a su intención. Ejemplo:

if
(expresionBooleana
) { return true;
) else {
return
false,'

En su lugar se debe escribir:

return expressionBooleana;

Similarmente,

if
(condicion)
{ return x;

return y,"

Se debe escribir:

return (condicion ? x . y),

10.5.3. COMENTARIOS ESPECIALES

Usar V en un comentario para indicar que algo tiene algún error, pero funciona. Usar *FIXME* para indicar que algo tiene algún error y no funciona.

10.6. TIPOS DE ARCHIVOS QUE SE UTILIZAN EN UNA APLICACIÓN WEB

En la siguiente tabla se describen los tipos de archivos implementados en una aplicación Web.

Tipo de archivo	Extensiones de archivo	Descripción
Estructurales	HTML, IHTML	En esta categoría se incluyen los archivos que describen lógicamente las interfaces de los objetos que componen una aplicación web.
Funcionales (servidor)	php, jsp, asp, pl, cgi	Estos son los archivos que implementan las funcionalidades de la aplicación web desde el lado del servidor.
Funcionales (cliente)	js, css, vs	Son archivos que implementan funcionalidades o apariencia de la aplicación web desde el lado del cliente.
Funcionales (incrustados)	class, swf, dir	Estos son archivos de funcionalidad externa que se visualizan mediante una funcionalidad agregada al navegador.
Imágenes	gif, jpg, bmp	Son archivos binarios de imágenes en varios formatos.
Documentos	pdf, ps, doc, xls, ppt, rtf	Este tipo de archivos, comprende los documentos que pueden ser desplegados o descargados de la aplicación web por el cliente mediante el navegador.

Anexo N° 5

Estándar para la integración, entrega y despliegue continuo en Kubernetes

1. OBJETIVO

Establecer lineamientos para la integración, entrega y despliegue continuo en KUBERNETES.

2. ALCANCE

Los lineamientos del presente documento son de aplicación obligatoria para el personal de la Oficina de Tecnologías de la Información (OTI).

3. DEFINICIONES

3.1. ¿QUÉ ES CI/CD?

Cuando hablamos de CI/CD tenemos que tomar en cuenta 3 términos: continuous integration, continuous delivery y continuous deployment. El término CI/CD engloba estos 3, sin embargo, no son dependientes unos de otros. En cambio, cada uno es una evolución o un paso más allá que el anterior.

3.2. CONTINUOUS INTEGRATION

La integración continua es el primer paso y está conformado por 2 etapas clave: la construcción del proyecto o *"build stage"* y las pruebas al proyecto o *"test stage"*. Este proceso se desencadena cuando se realiza una actualización del repositorio remoto (push) con nuevos cambios o mejoras (commits). Es posible decidir en qué condiciones se realizan ciertas etapas.

3.3. CONTINUOUS DELIVERY

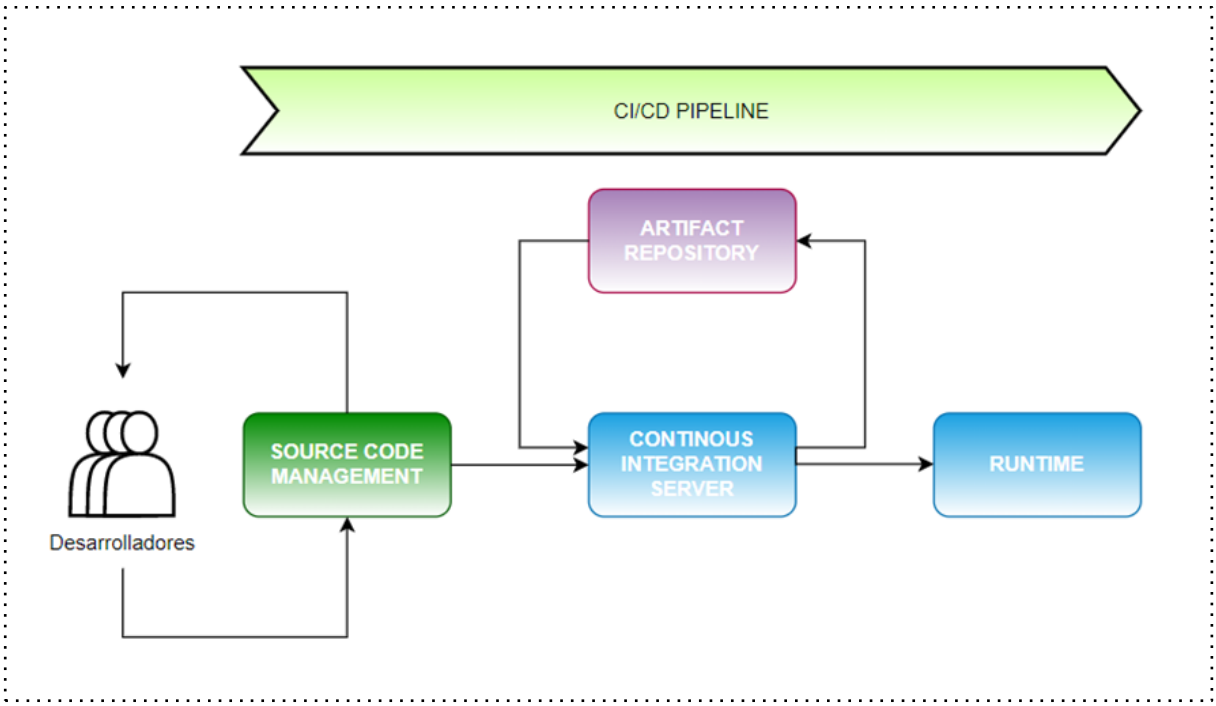
La entrega continua es un paso más allá de la integración continua. El proyecto no solo se construye y prueba en cada actualización al repositorio remoto, sino que, como paso adicional, también se despliega de forma continua en un ambiente de producción, aunque este paso se activa manualmente.

Este método garantiza que el código se verifique automáticamente, pero requiere que el dueño del repositorio realice el despliegue final de manera manual.

3.4. CONTINUOUS DEPLOYMENT

El despliegue da un paso más allá que la entrega continua. En este caso eliminamos la interacción manual completamente. Cuando utilizamos despliegue continuo nuestro proyecto es desplegado en un ambiente de producción de manera automática.

Tabla N°. 1: ARQUITECTURA DE CI/CD



Source Code Management
El sistema utilizado es Gitlab CE, el cual es accedido mediante un usuario y contraseña, a través de la siguiente URL:

https://sistemas.oefa.gob.pe/gitlab

4. SEGURIDAD

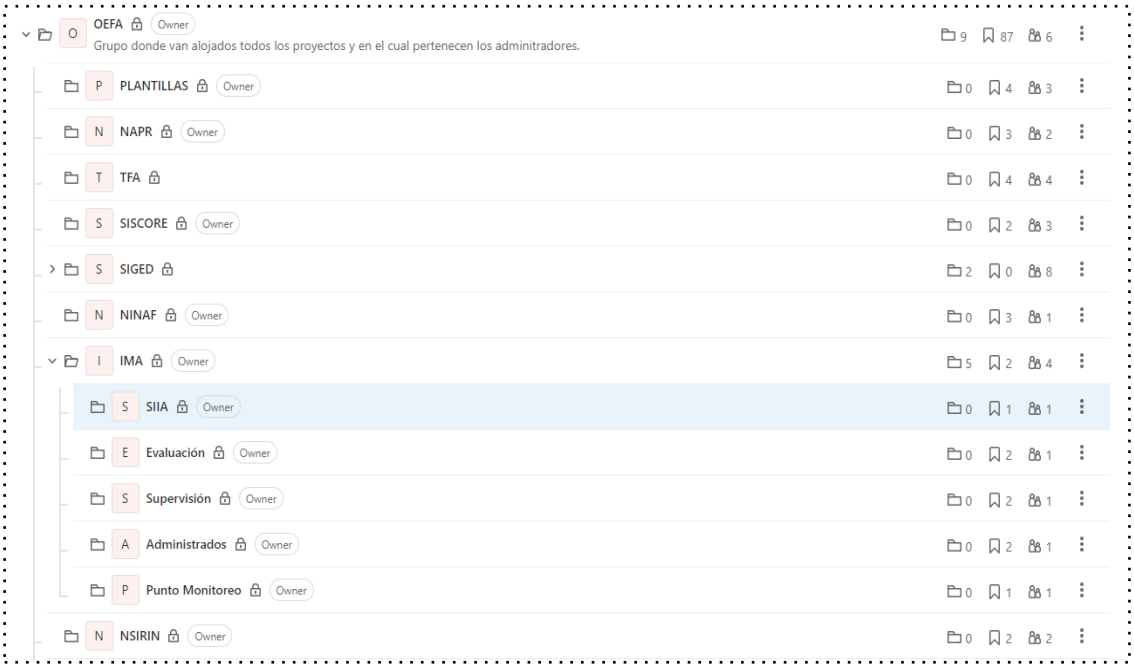
Se tienen las siguientes consideraciones de seguridad:

- Implementar tablas de auditoría para la identificación de las actualizaciones realizadas en el expediente electrónico durante su ciclo de vida.
- Utilizar las librerías actualizadas para evitar las vulnerabilidades en la aplicación y mantener la seguridad del sistema.
- Implementar etiquetas de seguridad en la cabecera del response de la aplicación.
- Incluir al inicio de sesión el código Capcha y asegurar que este funcione en el Sistema.
- Si la aplicación utiliza la cabecera *"Strict-Transport-Security"* para forzar el uso de HTTPS, entonces se deberá configurar la cabecera *"Strict-Transport-Security"*.
- Verificar que las referencias a invocación de páginas existan, sin embargo, si la respuesta es una página que no existe, entonces devolverá una página de error. Esta página de error no debe contener el número de versión del servidor web y lista de módulos habilitados en este servidor, para este caso deberá manejarse con una página estándar de error.
- Deshabilitar el autocomplete del ingreso de contraseña del usuario (la aplicación debe obligar el ingreso de la contraseña cada vez que inicie sesión).
- La aplicación deberá eliminar el código o no usado o comentado en las páginas HTML (front end), las cuales podrían revelar información importante para el aplicativo.

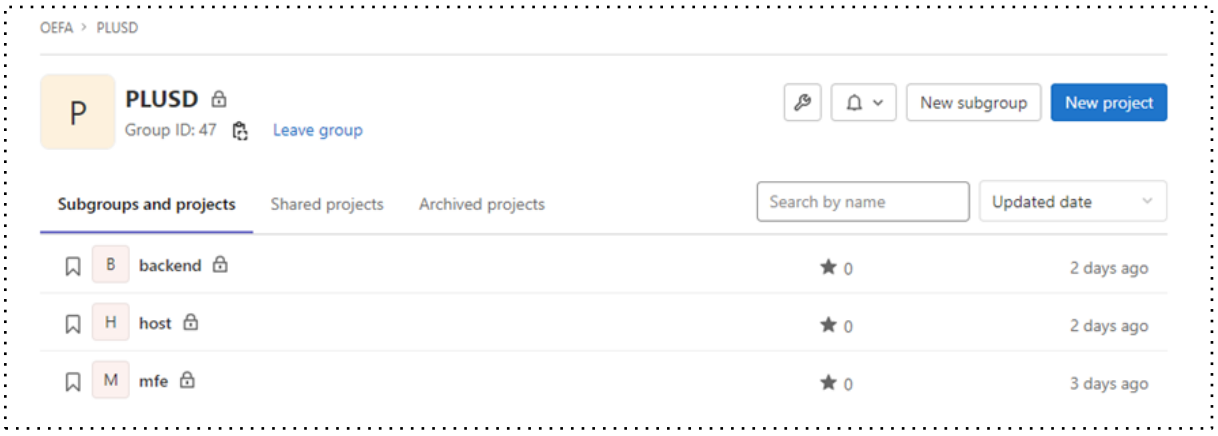
- La seguridad y autenticación de usuarios deberá realizarse mediante la integración al componente SSO (Single Sign-On) del OEFA. Esto implica la creación del menú y opciones, así como su consumo en la aplicación mediante servicios REST.

5. TAXONOMÍA DE REPOSITORIOS

La estrategia de alojamiento y gestión de código es bajo el esquema MULTI REPOSITORIO, es decir por cada microservicio se debe de usar un repositorio. Todos los repositorios asociados a un servicio deben estar alojados bajo un grupo dentro de la rama principal llamada OEFA. A continuación, se muestran los subgrupos, tanto de primer y segundo nivel.



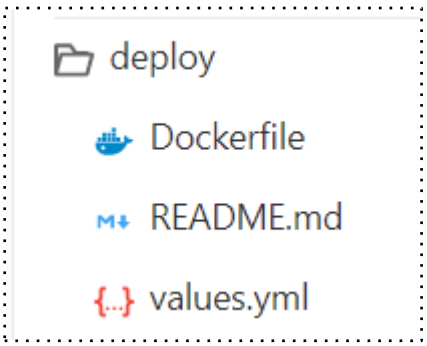
Dentro de cada subgrupo estarán los repositorios de cada microservicio. Estos repositorios parten del propio diseño de la aplicación o funcionalidad a desarrollar.



Referente al contenido dentro de cada repositorio, se debe considerar lo siguiente:

- El código fuente del desarrollo debe estar alojado en la raíz del repositorio.
- En la raíz del repositorio existirá la carpeta deploy, dentro de ella irán alojado dos archivos, el primero será *"Dockerfile"* el cual es diseñado y creado por parte del desarrollador para construir la imagen contenedor del microservicio y el archivo *"value.yml"* donde se configura los datos de

construcción de los objetos en kubernetes que forman parte del despliegue del microservicio (Ingress, Service, Deployment, Secret, entre otros).

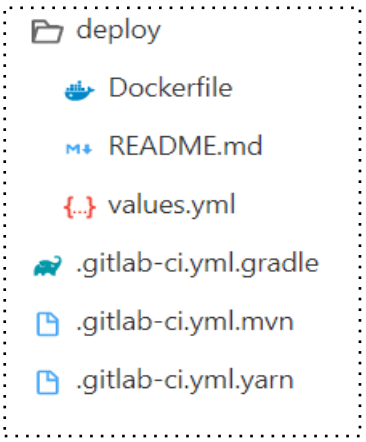


- Finalmente dentro de cada repositorio debe existir el archivo llamado *".gitlab-ci.yml"* el cual contiene las instrucciones para la ejecución del pipeline. Este archivo se modificará acorde a lo necesitado.

6. PLANTILLAS

Los repositorios serán inicializados con lo siguiente:

- Tres ramas, desarrollo, calidad y producción con lo indicado a continuación. Cabe resaltar que la rama *"desarrollo"* funge de rama maestra. Los desarrolladores que manejen dicho repositorio tendrán la función de *"developer"*, el cual puede realizar pull y push sobre dicha rama únicamente.
- La carpeta *"deploy"*, dentro de ella existe el archivo *"Dockerfile"* en blanco así como el archivo *"values.yml"* el cual debe modificarse acorde al despliegue.
- Tres plantillas de pipeline, las cuales difieren en la etapa *"package"* (mediante gradle, maven o yarn) pudiendo modificar dicha etapa a la medida de lo necesario, siendo estas plantillas una guía y no algo mandatorio, el desarrollador debe encontrar la forma óptima de empaquetar y generar los binarios necesarios para su código fuente acorde al lenguaje de programación usado.
- Las demás etapas del pipeline son manejadas de manera centralizada y no es posible manipularlas directamente.



7. RAMAS

Se utilizan tres ramas base las cuales corresponden a los tres ambientes de implementación y por los cuales se despliega cualquier sistema utilizado por la Entidad. Estas ramas son:

- ✓ Desarrollo
- ✓ Calidad
- ✓ Producción

7.1. Rama desarrollo:

Donde se ubica todo código fuente de la aplicación actualizado, así como los archivos necesarios para la integración, entrega y despliegue continuo (de acuerdo con lo indicado en la sección 3.1.1.). La finalidad de esta rama es brindar una ubicación de trabajo para el desarrollo, tanto externo como interno, así como eliminar la dependencia del desarrollador para re-construir y re-desplegar cualquier sistema.

7.2. Rama calidad:

Donde se ubica la última versión aprobada por el área de desarrollo para la etapa de pruebas.

7.3. Rama producción:

Donde se ubica la última versión aprobada por el área de calidad como versión productiva.

8. VERSIONAMIENTO

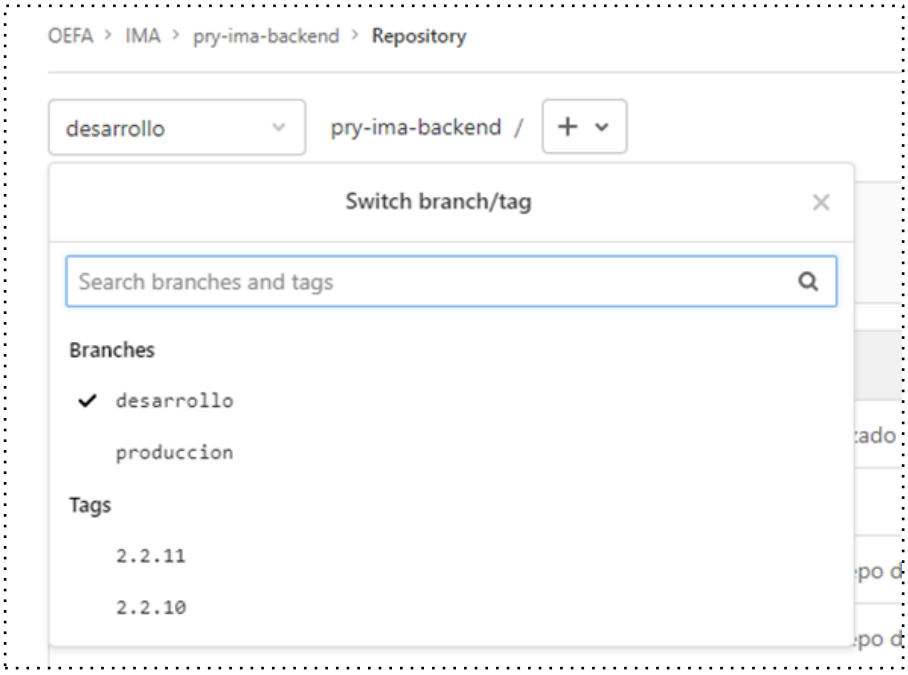
Existen distintas técnicas para la numeración de versiones, proponiendo para nuestra política la más extendida dentro de los sistemas de software libre. Es una notación numérica compuesta por tres números separados por puntos con la siguiente notación:

major.minor.revision

Cada uno de estos números tiene el siguiente significado:

- ✓ **major**: indica la versión principal del software, consistiendo en un conjunto de funcionalidades concretas que son recogidas y cubiertas en dicha versión.
- ✓ **minor**: indican funcionalidad menor cubierta en la versión de software entregada.
- ✓ **revisión**: se modifican cuando hay revisiones de código ante fallos de la aplicación.

NOTA: Debe existir tantos tags como entregas de software hacia el área de calidad existieron.



La rama principal se llama *"desarrollo"* la cual debe ser actualizada de manera constante. Si el trabajo sobre un repositorio es realizado por más de un desarrollador, y se ve en la necesidad de crear ramas alternas, estas deben ser creadas de manera temporal bajo el siguiente estándar.

desarrollo-X

De tal forma que X sea un número, empezando por 1. Finalmente, y una vez logrado una versión de software a entregar (hacia el área de calidad), esta rama debe ser fusionada con la rama principal (desarrollo) y así mismo debe crearse un tag con la versión correspondiente. Dicho tag (versión del software) es el que será usado por las diferentes áreas subsiguientes una vez entregado y aprobada dicha versión.

9. CONTINUOUS INTEGRATION SERVER

Esta etapa es realizada a través del sistema GitLab CE el cual brinda la herramienta de CI/CD de manera integrada por cada repositorio. La lógica de integración, entrega y despliegue, es manejada a través de un archivo llamado *".gitlab-ci.yml"*, el cual es proporcionado por la entidad y estará alojado en la raíz de cada repositorio. Dicho archivo puede ser visto en dos grandes partes:

- La primera, ubicada en el comienzo del archivo (variables), la cual está destinada a ser modificada de acuerdo con ciertos parámetros de desarrollo, por ejemplo, espacios de trabajo, nombre de recurso, url de publicación, contexto de publicación y recursos a ser usados por el desarrollo, así como ciertas variables que son usadas en la etapa de construcción.

- La segunda, todo aquello ubicado después de la anterior, la cual está destinada al manejo del pipeline. Esta segunda parte no debe ser modificada. Si se requiere alguna modificación se deberá informar ello para su validación y aplicación en caso se vea conveniente.

  La activación del pipeline se realiza de manera automática cada vez que se actualiza (realiza un push o merge) a una de las tres ramas principales (desarrollo, calidad o producción).

10. ARTIFACT REPOSITORY

Para esta etapa se hace uso del repositorio de artefactos integrado a la herramienta GitLab y su uso es automático, mediante el pipeline de CI/CD. Cada repositorio de código posee su propio repositorio de artefactos, donde se pueden almacenar paquetes (Packages Registry) o imágenes de contenedores (Container Registry). Para el caso puntual de esta primera etapa, se hará uso del Container Registry, donde irán alojadas todas las versiones (tag) del desarrollo.

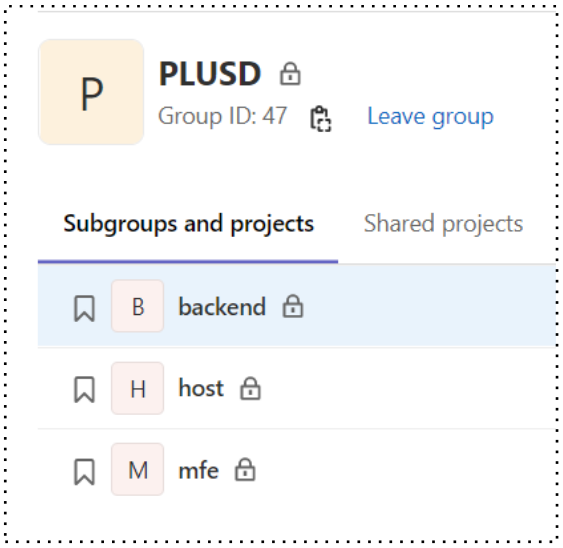
10.1. RUNTIME

La integración en esta primera etapa tendrá como sistema de ejecución la arquitectura Kubernetes. El despliegue será automático mediante el pipeline de CI/CD. Para la depuración de logs de todo lo desplegado dentro del ambiente Kubernetes, será necesario solicitar el acceso al espacio de trabajo con permisos de lectura sobre los elementos Pods y Deployments.

10.2. KUBERNETES

Cada grupo de microservicios relacionados entre sí, se desplegarán en su propio espacio de trabajo (namespace). Dicho nombre de espacio de trabajo será el mismo que el grupo de repositorios en minúsculas. Así mismo, dentro de este espacio de trabajo es donde se desplegarán los diversos microservicios asociados a dicho grupo.

Citemos como ejemplo los componentes de la Plataforma Única de Servicios Digitales *"PLUSD"*. El mismo consta de tres microservicios tal como se ve reflejado en la imagen contigua:



Contará con un espacio de trabajo llamado *"plusd"* con tres despliegues, uno por cada microservicio (repositorio).

```
[root@srvdevk8smaster01 ~]# kubectl get namespaces plusd
NAME    STATUS   AGE
plusd   Active   39d
[root@srvdevk8smaster01 ~]# kubectl -n plusd get deployments.apps
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
backend   1/1     1            1           2d19h
host      1/1     1            1           2d20h
mfe       1/1     1            1           2d19h
```

El acceso a cada espacio de trabajo es mediante el archivo *"kubeconfig"* correspondiente, con el rol correspondiente. Este acceso será proporcionado por la Oficina de Tecnologías de la Información.