	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 1 de 16
	Instructivo general para la calidad del desarrollo interno	

PROASISTEMAS S.A



Instructivo general para la calidad del desarrollo interno

Este documento es propiedad intelectual de Proasistemas S.A y queda prohibida su reproducción total o parcial en cualquier medio. El otorgamiento de una copia a terceros deberá ser con autorización escrita de la gerencia o en su defecto el responsable de Proasistemas S.A.



	PROASISTEMAS S.A.	Código: TI-MAI-I-004
	Instructivo general para la calidad del desarrollo interno	Versión: 02 Fecha: 29/04/2024 Página 2 de 16

Tabla de contenido

Introducción	3
Fase 1: Developer testing (pruebas del desarrollador - unit testing)	3
Fase 2: Peer review or pair programming testing	3
Fase 3: Tester testing	4
Estandarización de documentación	4
Protocolo de prototipo	4
Diagramas	4
Modelo entidad-relación	4
Diagrama de secuencia	6
Diagrama de casos de uso	7
Diagrama de paquetes	9
Diagrama de componentes	10
Estructura de almacenamiento de diagramas en NOTION	11
Control de cambios	12
Manual de uso	13
Cypress SI	14
Control de cambios	16

	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 3 de 16
	Instructivo general para la calidad del desarrollo interno	

Introducción

De conformidad con la política de desarrollo de software del área de Mantenimiento de Aplicaciones Internas, se adopta un enfoque estructurado de tres etapas de pruebas para mejorar la calidad del código y garantizar una entrega de software sólida. Este enfoque implica tres fases de pruebas: Developer Testing (Pruebas del Desarrollador), Peer Review or Pair Programming Testing (Revisión de Pares o Pruebas de Programación en Pareja) y Tester Testing (Pruebas del Tester).

Fase 1: Developer testing (pruebas del desarrollador - unit testing)

Responsabilidades del Desarrollador y Explicación: El desarrollador que escribió el código es responsable de las pruebas iniciales de su propio trabajo. Esta fase, conocida como "unit testing" (pruebas unitarias), implica:

Parámetros y aspectos para enfocar

- **Funcionalidad:** Verificar que el código realice las funciones previstas y cumpla con los requisitos.
- **Validación de Entradas:** Probar varios valores de entrada, incluyendo casos límite y condiciones extremas.
- **Manejo de Errores:** Comprobar que los mensajes de error y las excepciones se gestionen correctamente.
- **Rendimiento:** Evaluar la eficiencia del código y los tiempos de respuesta.
- **Cobertura de Código:** Apuntar a una alta cobertura de código para evaluar todos los caminos del código.
- **Estilo de Código:** Cumplir con los estándares de codificación y las mejores prácticas.
- **Puntos de Integración:** Si corresponde, probar las interacciones con sistemas o API externas.
- **Seguridad:** Identificar y abordar posibles vulnerabilidades de seguridad.


Fase 2: Peer review or pair programming testing (revisión de pares o pruebas de programación en pareja)

Responsabilidades del Revisor de Pares o Programador en Pareja y Explicación: Después de que el desarrollador complete sus pruebas iniciales, otro desarrollador, a menudo un par o compañero programador, revisa el código. Esta fase, denominada "code review" (revisión de código) o "peer review" (revisión entre pares), implica lo siguiente:

Parámetros y aspectos para enfocar

- **Calidad del Código:** Evaluar la claridad, legibilidad y mantenibilidad del código.
- **Estándares de Codificación:** Asegurarse de que se cumplan las directrices de codificación y las convenciones de nomenclatura.
- **Corrección:** Verificar la alineación con los requisitos y el diseño.
- **Manejo de Errores:** Comprobar el manejo adecuado de errores.
- **Optimización:** Evaluar oportunidades para mejorar la eficiencia del código.

Este documento es propiedad intelectual de Proasistemas S.A y queda prohibida su reproducción total o parcial en cualquier medio. El otorgamiento de una copia a terceros deberá ser con autorización escrita de la gerencia o en su defecto el responsable de Proasistemas S.A.

	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 4 de 16
	Instructivo general para la calidad del desarrollo interno	

- **Consistencia:** Asegurarse de la consistencia en el estilo de codificación y los patrones de diseño.
- **Comentarios y Documentación:** Verificar que el código esté bien documentado.
- **Cobertura de Pruebas:** Asegurarse de que se hayan escrito pruebas unitarias y que cubran el código.

Fase 3: Tester testing (pruebas del tester - pruebas funcionales, pruebas de integración, pruebas de aceptación del usuario)

Responsabilidades del tester y explicación: En esta fase, un probador de software dedicado realiza pruebas desde la perspectiva del usuario. Esta fase incluye los siguientes tipos de pruebas:

Parámetros y aspectos para enfocar

- **Requisitos funcionales:** Verificar que el software cumpla con los requisitos funcionales especificados.
- **Integración:** Probar las interacciones entre diferentes módulos o componentes.
- **Validez de datos:** Comprobar la corrección y la integridad de los datos manejados por el software.
- **Interfaz de usuario:** Evaluar la usabilidad, accesibilidad y coherencia de la interfaz de usuario.
- **Rendimiento:** Evaluar el rendimiento del sistema bajo diversas condiciones.
- **Seguridad:** Probar las vulnerabilidades de seguridad, validación de entradas y autenticación.
- **Compatibilidad:** Verificar la compatibilidad con navegadores, dispositivos y sistemas operativos.
- **Regresión:** Asegurarse de que los nuevos cambios no introduzcan regresiones.
- **Usabilidad:** Evaluar la facilidad de uso y la experiencia del usuario.
- **Escalabilidad:** Probar cómo el software se comporta con un aumento en el uso.
- **Accesibilidad:** Asegurarse del cumplimiento de estándares de accesibilidad.
- **Localización e internacionalización:** Probar para múltiples idiomas y regiones, si corresponde.

Estandarización de documentación

Protocolo de prototipo

Diagramas

Modelo entidad-relación

Situación: Antes de realizar cambios en un proyecto existente que requiere una actualización en la base de datos para agregar o modificar información.

Por qué: Este diagrama ayuda a entender cómo se estructuran y relacionan los datos en la base de datos. Es útil antes de implementar cambios para garantizar una gestión eficiente de la información.


	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 6 de 16
	Instructivo general para la calidad del desarrollo interno	

Diagrama de secuencia

Situación: Representación visual que muestra cómo las diferentes partes de un sistema interactúan entre sí y en qué orden durante la ejecución de una función o proceso. Es útil para comprender la lógica temporal de las interacciones en un sistema.

Por qué: Se emplea cuando se necesita entender y documentar la secuencia de eventos detallada entre actores (usuarios o sistemas) y objetos (componentes del sistema) en la implementación de una nueva función o proceso, como la autenticación de usuarios o la ejecución de transacciones en una aplicación.

Cuando no usarlo: Puede no ser necesario en situaciones simples donde las interacciones son directas y lineales, o cuando la lógica del proceso es evidente sin la necesidad de una representación visual detallada.

Parámetros:

- Ideal para visualizar procesos empresariales
- Revela la estructura del sistema
- Muestra la secuencia cronológica de mensajes e interacciones
- Admite iteraciones y ramificaciones simples
- Eficiente para escenarios multitarea
- Colores sugeridos:
- Letra: #000000 (Negro)
- Fondo neutro: #FFFFFF (Blanco)
- Elementos: #90EE90 (Verde claro)
- Mensajes: #FF6347 (Tomate)

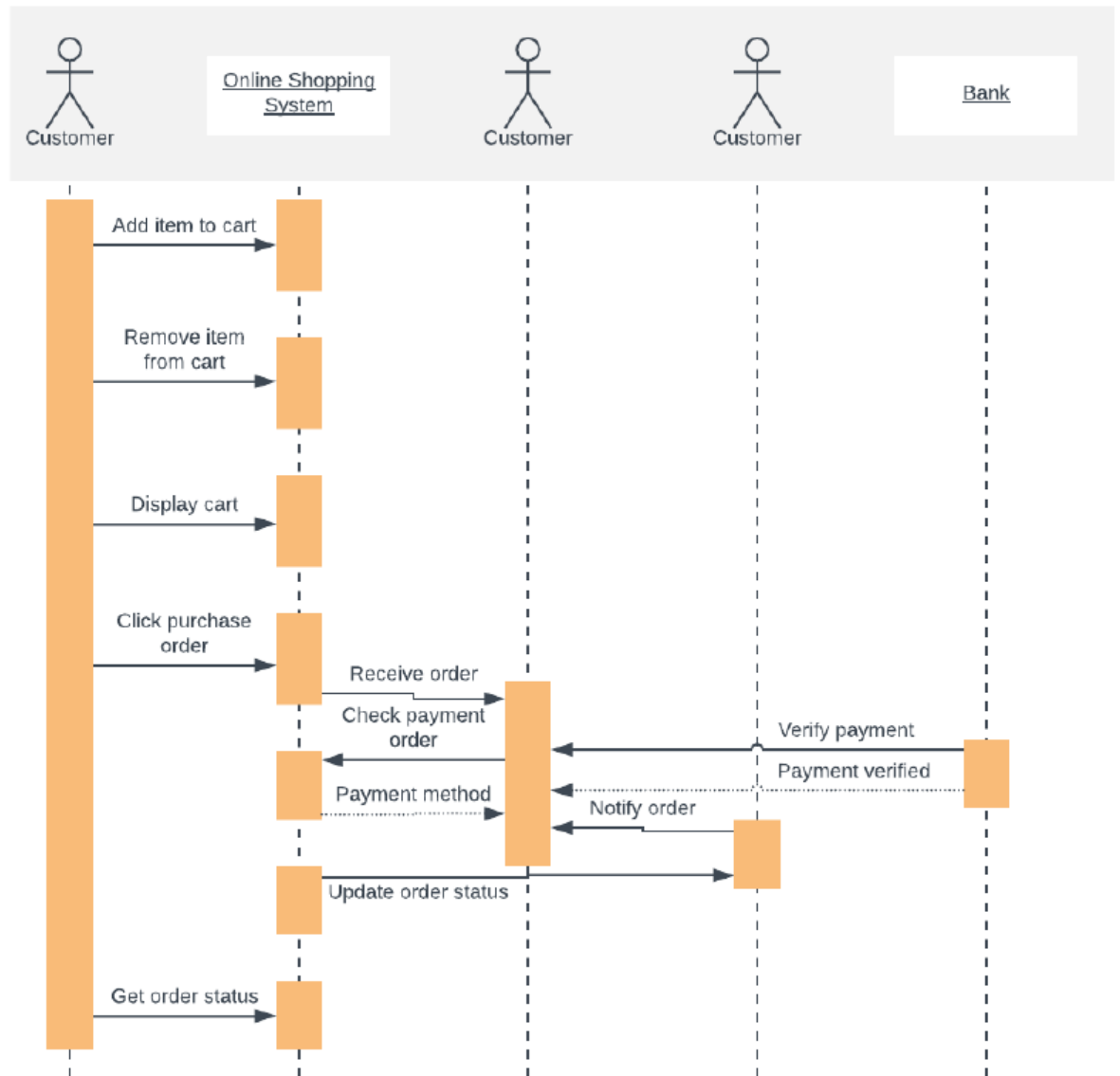


Diagrama de casos de uso

Situación: Al planificar la adición o modificación de un botón o característica interactiva a una interfaz de usuario.

Por qué: Este diagrama visualiza cómo los usuarios interactuarán con la función. Es útil para comprender los posibles escenarios de uso y requisitos.

Cuando no usarlo: No es necesario para cambios muy pequeños o cuando las interacciones son obvias.

Parámetros:

Este documento es propiedad intelectual de Proasistemas S.A y queda prohibida su reproducción total o parcial en cualquier medio. El otorgamiento de una copia a terceros deberá ser con autorización escrita de la gerencia o en su defecto el responsable de Proasistemas S.A.

- Describe las funcionalidades del sistema (Muestra lo que un sistema puede realizar mas no lo que no puede realizar)
- Un caso de uso es un conjunto de eventos que ocurren cuando un “actor” usa un sistema para completar un proceso
- Los actores (personas, organizaciones, aplicaciones) interactúan con el sistema
- Representación visual de los requisitos funcionales
- Colores sugeridos:
- Letra: #000000 (Negro)
- Fondo neutro: #ECECEC (Pastel claro)
- Actores: #ADD8E6 (Azul claro)
- Casos de Uso: #FFDAB9 (Naranja claro)

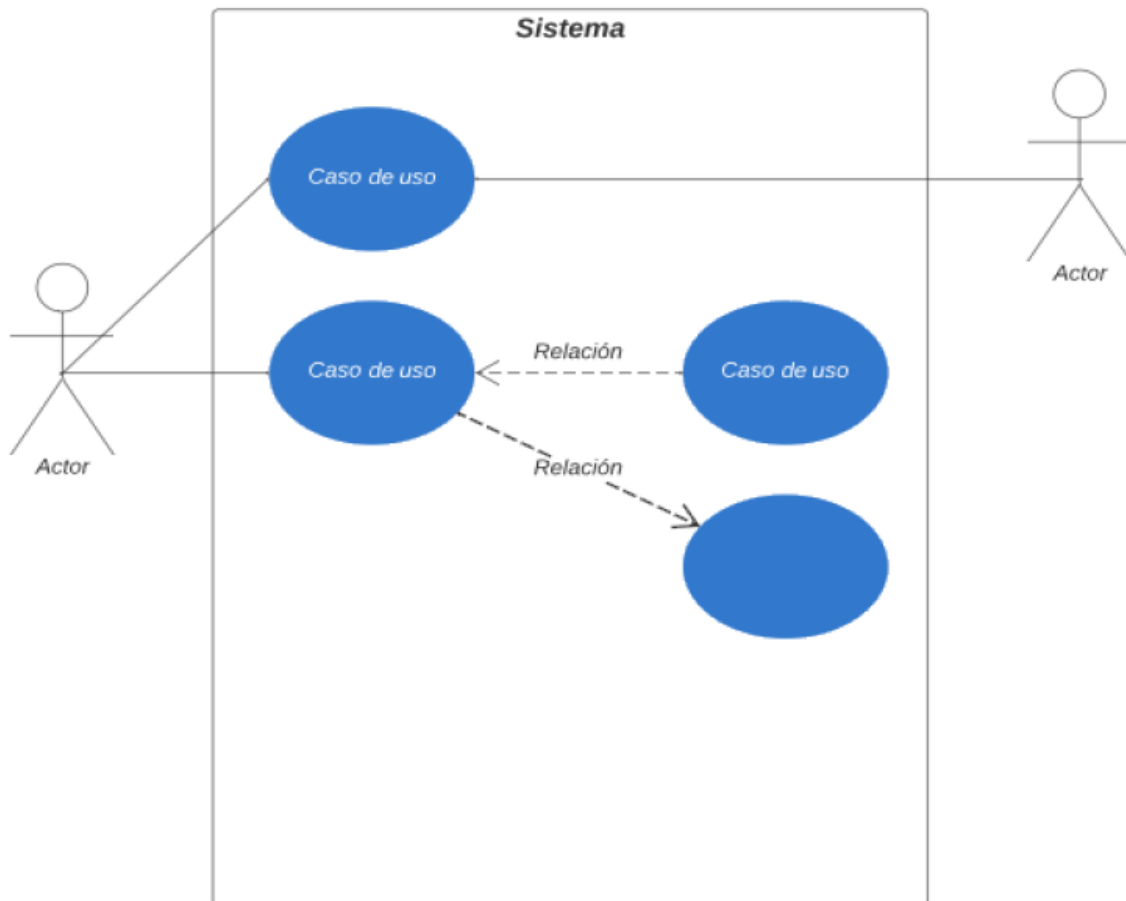


Diagrama de paquetes

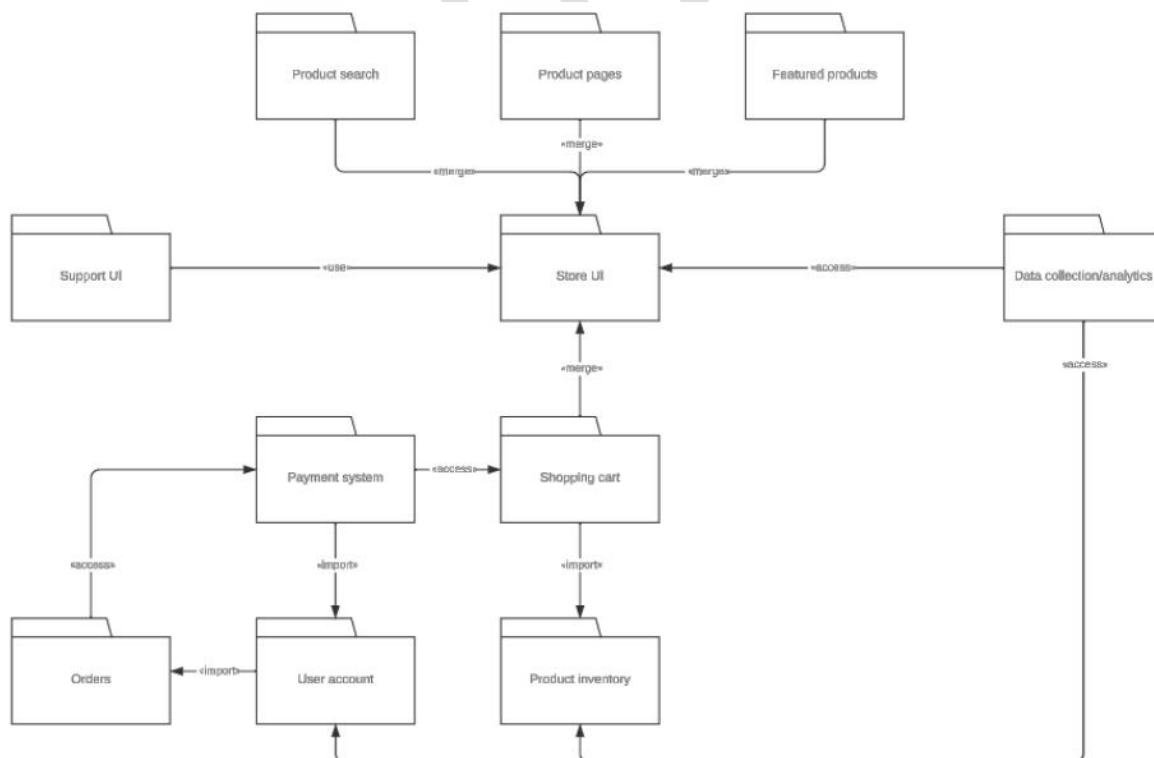
Situación: Antes de realizar ajustes en la organización de las funciones en un proyecto, como agrupar varias características relacionadas.

Por qué: Este diagrama muestra cómo diferentes partes del sistema están organizadas y dependen entre sí. Es útil para entender la estructura general y las relaciones entre módulos.

Cuando no usarlo: No es esencial si la aplicación web es simple y las funciones no están divididas en módulos distintos.

Parámetros:

- Representa las dependencias entre los paquetes del modelo
- Muestra las relaciones entre los diversos componentes que forman un sistema complejo
- Colores sugeridos:
- Letra: #000000 (Negro)
- Fondo neutro: #F5F5F5 (Gris claro)
- Paquetes: #ECECEC (Pastel claro)




	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 10 de 16
	Instructivo general para la calidad del desarrollo interno	

Diagrama de componentes

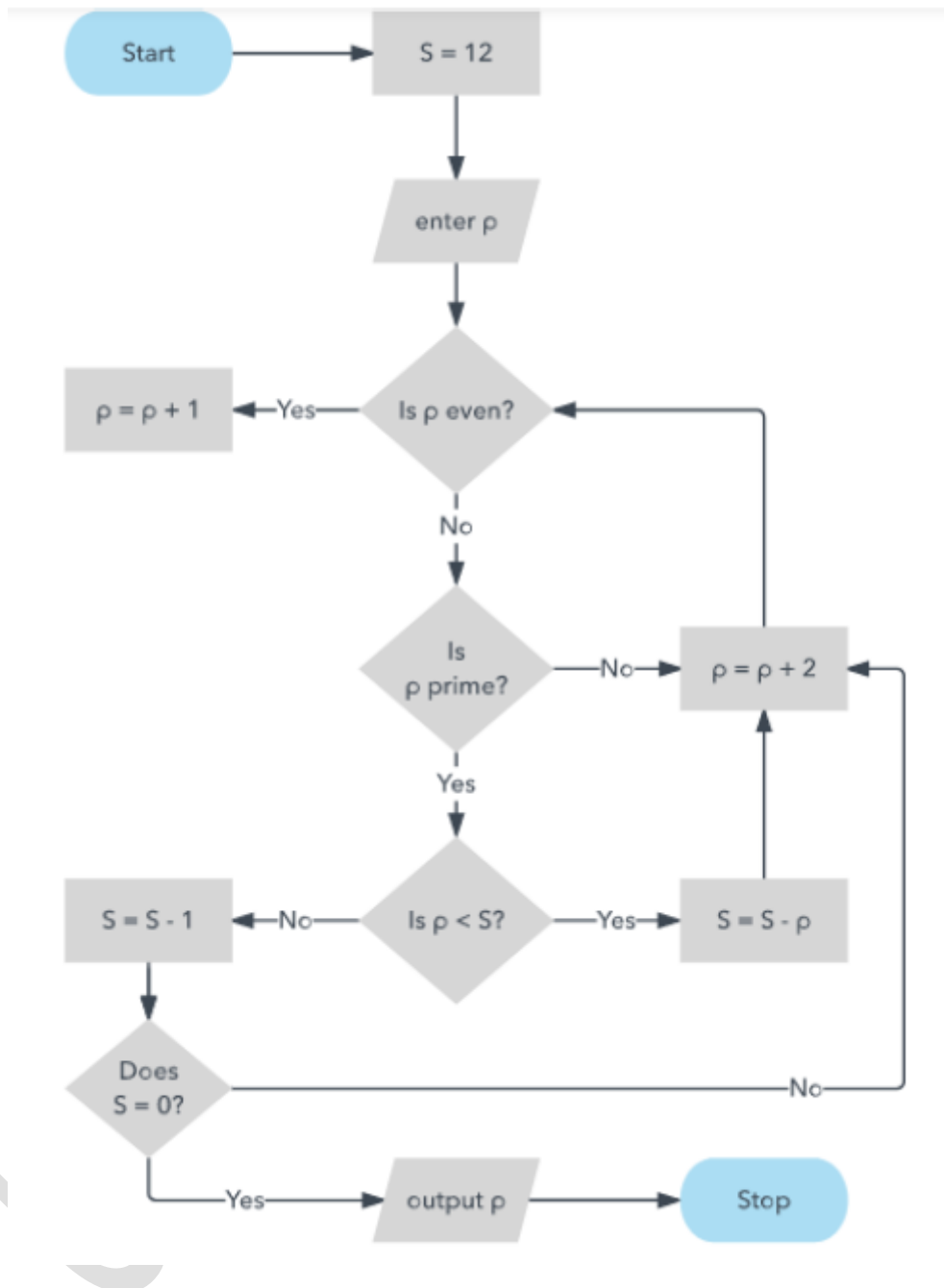
Situación: Al documentar el flujo de trabajo para procesar un formulario de contacto en un proyecto.

Por qué: Un diagrama de flujo muestra la secuencia de pasos y decisiones en un proceso. Es útil para entender cómo la información se mueve a través del sistema y cómo se manejan diferentes situaciones.

Cuando no usarlo: Puede no ser necesario para procesos extremadamente simples o si el flujo de trabajo es obvio y directo.

Parámetros:


- Ilustra la agrupación lógica de elementos y sus relaciones
- Simplifica sistemas complejos en componentes más pequeños
- Cada pieza se muestra con una caja rectangular con su nombre
- Los conectores definen las relaciones y dependencias entre los diferentes componentes
- Colores sugeridos:
- Letra: #000000 (Negro)
- Fondo neutro: #FFFFFF (Blanco)
- Componentes: #90EE90 (Verde claro)



Estructura de almacenamiento de diagramas en NOTION

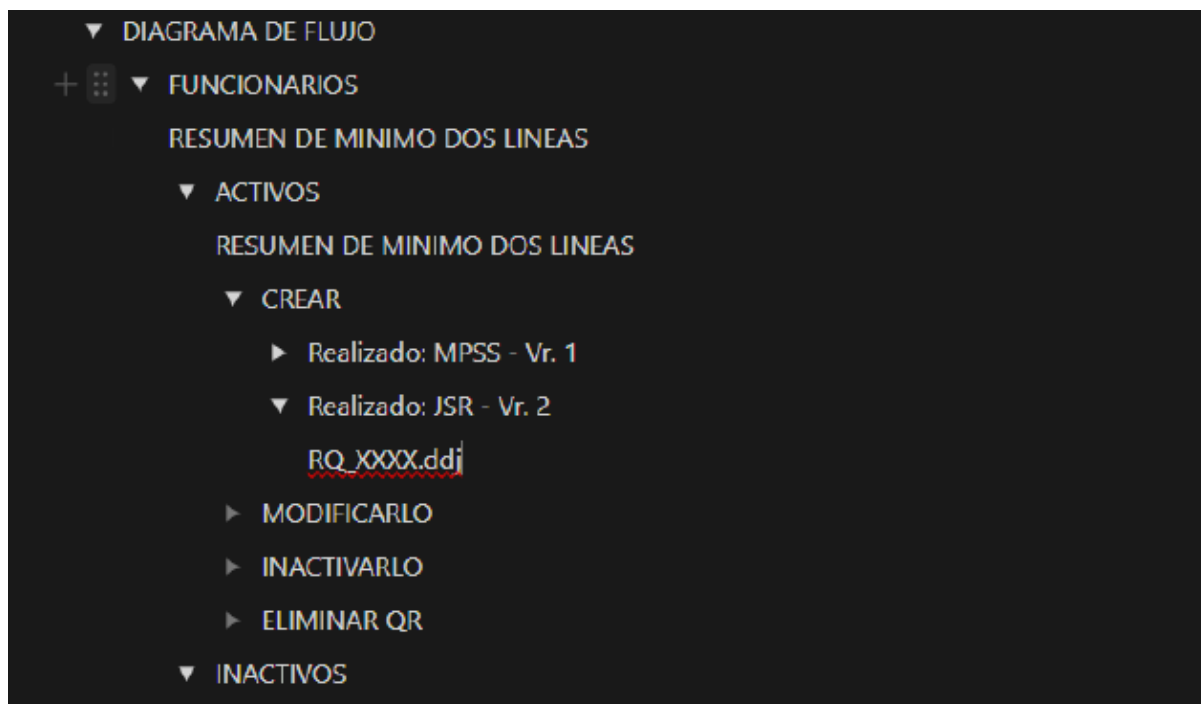
Se deben almacenar dentro de los desplegables iniciando por la pestaña afectada del proyecto.

- Dentro de la función afectada con su respectivo nombre

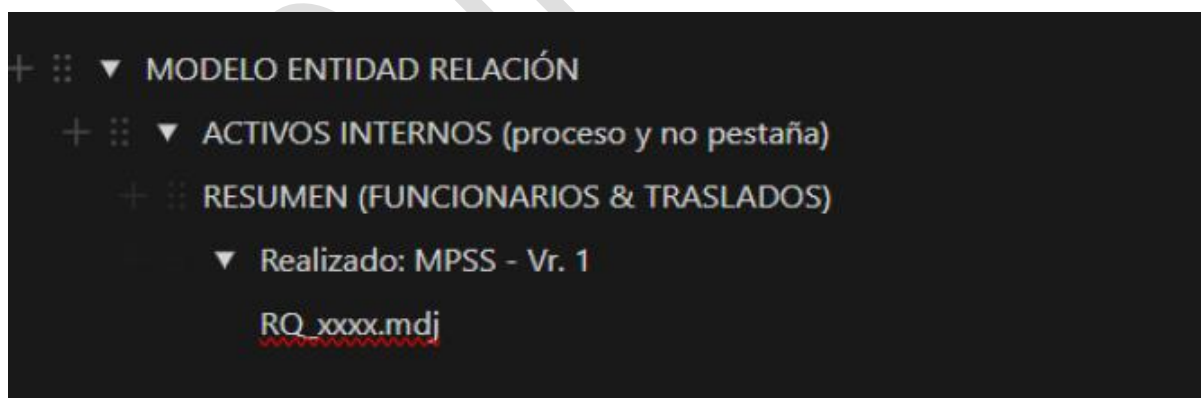
	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 12 de 16
	Instructivo general para la calidad del desarrollo interno	

En desplegable donde se guarda el documento debe estar nombrada con las iniciales del programador que ejecuto junto con el número de versión.

- Dentro del desplegable se debe subir el diagrama nombrado con el RQ al que pertenece.




En el modelo de entidad relación se presenta primero con el proceso en lugar de como pestaña.



Control de cambios

- En el proceso de control de cambios se generará en el orden estipulado en la imagen adjunta
- Las creaciones de funciones deben mencionar también a cuál tarea del story está relacionada.

	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 13 de 16
	Instructivo general para la calidad del desarrollo interno	

- En las modificaciones se debe mostrar cómo funciona antes y después, de igual forma, se debe mencionar a cuál tarea está relacionado.

PROCESO NUEVO

- CAMBIOS BD
 - 1.1. sentencias generadas
- CAMBIOS FRONTEND (TAREA 1)
 - 2.1. Se creo un botón
- CAMBIOS BACKEND (TAREA 1)
 - 3.1. Direccionar a un formulario
- CAMBIOS FRONTEND (TAREA 2)
 - 4.1. Formulario con 6 campos select


MODIFICAR PROCESO EXISTENTE

- CAMBIOS BD
 - 1.1. sentencias generadas
- CAMBIOS FRONTEND (TAREA 1)
 - 2.1. Se MODIFICO un botón

Antes & Ahora

Input & botton


- CAMBIOS BACKEND (TAREA 1)
 - 3.1. Direccionar a un formulario
- CAMBIOS FRONTEND (TAREA 2)
 - 4.1. Formulario con 6 campos select



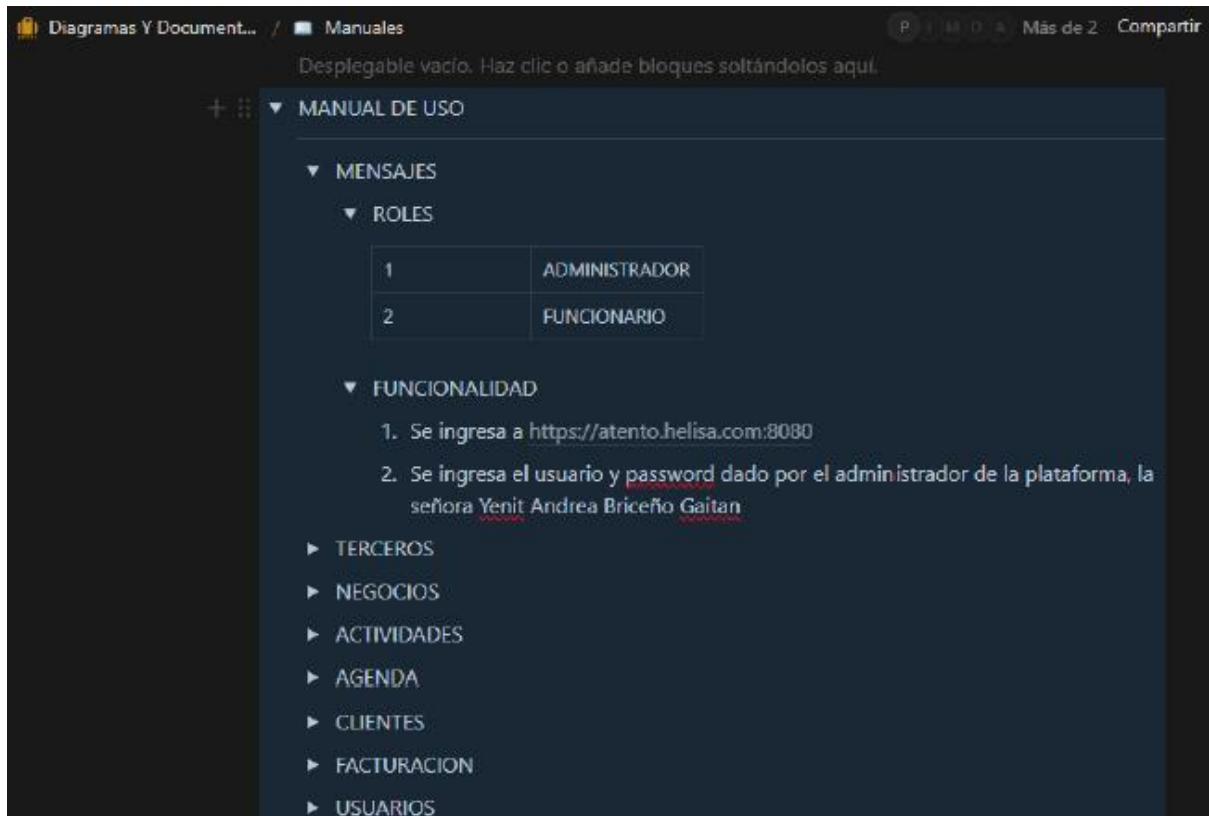
Manual de uso

- La creación de manuales se realizará con desplegables en notion de la forma en la que se visualiza en la imagen adjunta, mencionando las funciones, roles e instrucciones.

Este documento es propiedad intelectual de Proasistemas S.A y queda prohibida su reproducción total o parcial en cualquier medio. El otorgamiento de una copia a terceros deberá ser con autorización escrita de la gerencia o en su defecto el responsable de Proasistemas S.A.


	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 14 de 16
	Instructivo general para la calidad del desarrollo interno	

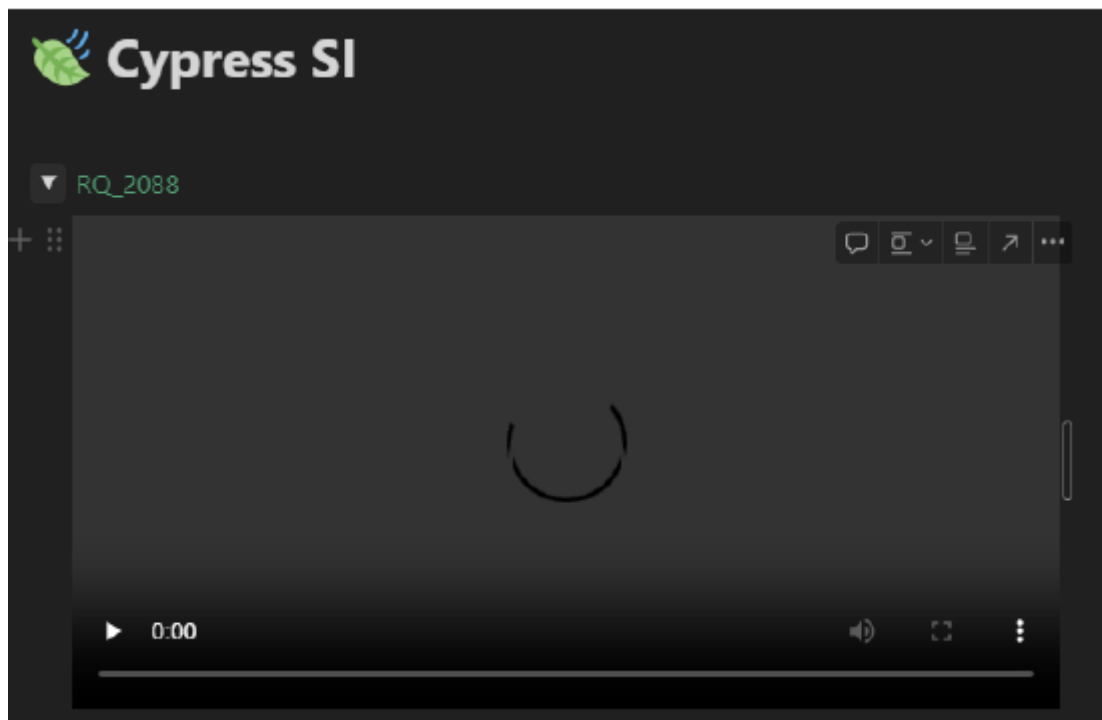
- Se debe documentar en la pestaña de manuales en la documentación del respectivo proyecto.



Cypress SI

- La Documentación Cypress debe estar registrada en la carpeta Cypress dentro de la documentación del respectivo proyecto.
- Debe estar adjunto dentro de un desplegable con el RQ como nombre
- Dentro del desplegable debe estar el video adjunto.
- Cuando algún dato, función o información no pueda ser visualizada en el video de Cypress, esta información debe estar especificada en la respectiva tarea mencionando que no se muestra en el video.

	PROASISTEMAS S.A.	Código: TI-MAI-I-004 Versión: 02 Fecha: 29/04/2024 Página 15 de 16
	Instructivo general para la calidad del desarrollo interno	



Recomendación:

- Mantener una documentación de pruebas completa, que incluya planes de pruebas, casos de pruebas, datos de pruebas y resultados de pruebas.
- Seguir las normas y directrices de documentación de la empresa.

Al manejar las tres etapas de pruebas mencionadas al inicio de este instructivo, se pretender detectar y corregir defectos tempranos, mejorar la calidad del código y garantizar la entrega confiable de software que cumpla con las políticas de desarrollo. La colaboración efectiva y la comunicación entre desarrolladores, revisores y probadores son clave para su éxito. Se trabaja conjuntamente para mejorar el proceso de desarrollo de software y ofrecer soluciones de alta calidad.

HELISA Software para el trabajo	PROASISTEMAS S.A.		Código: TI-MAI-I-004
	Instructivo general para la calidad del desarrollo interno		Versión: 02 Fecha: 29/04/2024 Página 16 de 16

Control de cambios

Control de cambios					
N° Versión	Ítem cambiado	Descripción de cambio realizado	Observaciones	Fecha de cambio	Responsable
01	-	Creación de documento	-	08/09/2023	Directora de procesos
02	-	Actualización de documento	-	29/04/2024	Asistente de procesos