MATERIAL DE LECTURA OBLIGATORIA / UNIDAD 06

Implementación, Accesibilidad, SEO y Métricas de Producto

Del prototipo validado a una interfaz implementable, inclusiva y medible



Asignatura
Técnicas de UX y Diseño de Interfaces
Ingeniería de Sistemas · UNICEN

Enfoque
Handoff, documentación, calidad web, accesibilidad, analítica y mejora del Proyecto Integrador

Material académico de cátedra · Noviembre 2026 · Lectura obligatoria · Edición ampliada

GUÍA DE LECTURA

Contenido de la unidad

El propósito de esta edición es comprender las decisiones que conectan el diseño de interacción con la construcción, la evaluación técnica y el seguimiento de un producto digital.

La lectura se organiza en los mismos bloques del programa analítico. Cada bloque combina fundamentos, ejemplos de la plataforma de canje de beneficios y procedimientos de aplicación. Las tablas y diagramas sintetizan conceptos que se desarrollan en el texto; no sustituyen su lectura.

Bloque	Contenidos	Páginas
I. Handoff	Transferencia, documentación, contratos, estados, aceptación y cambios.	3–13
II. Accesibilidad	Diseño inclusivo, WCAG 2.2, semántica, mejora progresiva y evaluación.	14–26
III. SEO y rendimiento	Rastreo, indexación, estructura, velocidad y estabilidad.	27–33
IV. Medición	KPIs, eventos, calidad de los datos y A/B testing.	34–42
V. Integración	Caso resuelto, aplicación al TP y referencias.	43–48

Cómo utilizar el material

En los ejemplos se distinguen los requisitos que puede representar un prototipo de aquellos que necesitan código, servicios o datos reales. Las cifras de canje, los eventos y las observaciones son hipotéticos y sirven para practicar. No constituyen evidencia de una aplicación en producción.

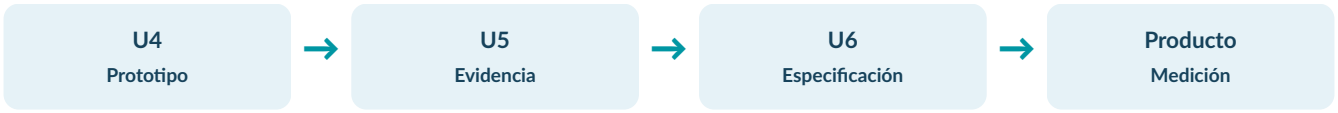
> **Pregunta transversal:** ¿qué debe ocurrir para que una persona complete una tarea, cómo sabremos que la implementación respeta esa intención y qué indicador permitirá revisar si el producto mejora?

Se recomienda revisar los ejemplos junto al prototipo corregido de la Unidad 5. Es importante comprender el razonamiento detrás de cada decisión antes de elegir herramientas, bibliotecas o plataformas.

01 / ENFOQUE DE LA UNIDAD

La interfaz también debe poder construirse

En la Unidad 5 comprobamos qué sucede cuando una persona usa el prototipo. Ahora debemos lograr que ese conocimiento llegue al código **sin perder intención, estados ni criterios de calidad**.



¿Qué aprenderemos en esta unidad?

Construir con acuerdos
Describir comportamiento, componentes, cambios, criterios de aceptación y responsabilidades entre diseño y desarrollo.

Construir para todas las personas
Introducir accesibilidad, HTML semántico, controles nativos y verificación manual y automática.

Construir para encontrarse y funcionar
Aplicar SEO técnico y performance a páginas públicas sin confundirlo con la visibilidad de áreas privadas.

Construir para aprender
Definir KPIs, eventos y experimentos que permitan tomar decisiones con evidencia.

> El objetivo no es formar especialistas en accesibilidad, SEO ni analítica: es que el ingeniero pueda **reconocer requisitos, implementarlos con criterio y trabajar con otros perfiles sin perder calidad de uso**.

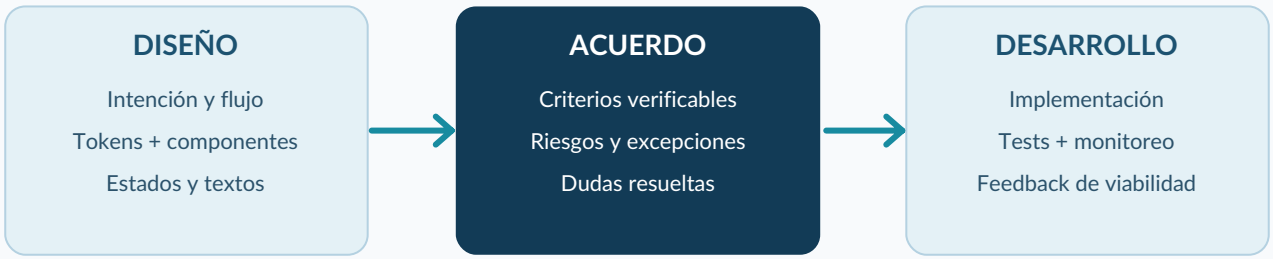
Utilizaremos como caso transversal una web de canje de beneficios. Las decisiones serán transferibles a reservas, inscripciones, turnos y otros flujos transaccionales.

El aporte del ingeniero no consiste únicamente en reproducir el aspecto de una pantalla. Debe transformar intenciones y hallazgos en condiciones observables de funcionamiento, detectar restricciones de implementación y devolver información al equipo de diseño. Las decisiones que afectan a las personas deben mantenerse explícitas incluso cuando cambie la tecnología elegida.

02 / ENTRE DISEÑO Y DESARROLLO

Handoff: pasar decisiones, no capturas

El *handoff* es la transferencia coordinada de decisiones, recursos y restricciones desde diseño hacia implementación. No es «entregar el Figma y listo».



El handoff funciona como conversación y contrato, no como archivo definitivo.

Contenido mínimo de una entrega

Elemento	Qué debería recibir el equipo
Flujo	Objetivo de usuario, pasos, entradas, salida y criterio de finalización.
Pantallas	Diseño responsive, grillas, tokens y componentes reutilizables.
Comportamientos	Qué ocurre al tocar, enviar, retroceder, cargar, fallar o repetir.
Accesibilidad	Orden del foco, etiquetas, mensajes, contraste y navegación por teclado.
Pruebas	Criterios de aceptación, ejemplos y casos límite.

> Una captura no responde: «¿qué pasa si falla el canje?», «¿qué comunica el sistema mientras espera?» o «¿cómo se usa sin mouse?».

02 / HANDOFF · DESARROLLO

Cómo se realiza una transferencia de diseño

Lo que se transfiere es un modelo compartido de la tarea, no una representación estática. El handoff comienza mientras se está diseñando y continúa durante la implementación.

En el modelo de trabajo de las unidades anteriores, el equipo investiga necesidades y construye un prototipo para validar sus hipótesis. El prototipo permite explorar las acciones visibles, pero deja abiertos muchos detalles: cuándo recuperar el saldo, quién autoriza una transacción, qué respuesta se muestra si falla la red. Si estos aspectos se resuelven en silencio durante el desarrollo, la interfaz final puede contradecir decisiones previamente validadas.

Antes, durante y después de la entrega

Antes. El equipo revisa el flujo validado, identifica las pantallas y casos representados, y señala las incógnitas. Se acuerdan la versión de referencia, los componentes que se reutilizarán, los textos y las reglas que condicionan cada acción. Las dudas de viabilidad deben resolverse con desarrollo antes de cerrar la especificación.

Durante. Diseño y desarrollo recorren una tarea concreta. En un canje, no basta con señalar dónde está ubicado el botón: hay que explicar en qué momento está disponible, qué sucede con el saldo y qué reconoce el usuario como finalización. Desarrollo aporta restricciones y alternativas. Si una solución no es viable, se conserva el objetivo de usuario y se acuerda otra manera de satisfacerlo.

Después. Se verifica la implementación contra los criterios acordados y se comparan los comportamientos reales con los del prototipo. Siempre que cambie una decisión, se actualizan las referencias y la prueba correspondiente. Esto hace del handoff una práctica iterativa, no una instancia de aprobación unilateral.

> **Ejemplo de una duda bien resuelta.** La prueba de usabilidad mostró que una persona no sabía si había canjeado el beneficio. Desarrollo informa que la respuesta puede demorar varios segundos. El equipo acuerda comunicar el procesamiento y mostrar un comprobante exclusivamente cuando el servidor confirma la operación. La decisión se registra con su motivo y sus pruebas.

Una entrega de calidad reduce las interpretaciones contradictorias. Si el equipo necesita preguntar, las preguntas deben poder dirigirse al responsable de cada decisión y quedar documentadas, en lugar de quedar dispersas en conversaciones privadas.

Referencias para profundizar: [17].

03 / DOCUMENTACIÓN FUNCIONAL Y TÉCNICA

Documentar para reducir ambigüedad

La documentación funcional explica **qué debe lograr el usuario y cómo debería comportarse el producto**. La técnica explica **cómo se implementará y verificará** en un entorno concreto.

Documento	Preguntas que responde	Ejemplo en canje
Funcional	¿Quién lo usa? ¿Para qué? ¿Qué reglas de negocio aplican?	No se puede canjear con saldo insuficiente.
Interacción/UI	¿Qué estados, textos, transiciones y errores ve la persona?	Botón deshabilitado + explicación del saldo faltante.
Técnico	¿Qué componentes, rutas, API, eventos y pruebas se necesitan?	POST de canje, código de error, componente de mensaje.
Validación	¿Qué evidencia muestra que cumple la intención?	Prueba de canje exitoso, fallido y con teclado.

No hace falta escribir una enciclopedia

Para una operación simple, una ficha clara puede resolver más que veinte páginas. Documentar lo que cambia decisiones, evita errores o elimina dudas.

La documentación debe evolucionar

Una especificación desactualizada es peor que una duda explícita. Agreguen versión, fecha y responsable de decisiones importantes.

Ficha funcional mínima

US-07 / Canjear beneficio. Usuario registrado con puntos disponibles.
Disparador: pulsa «Canjear» en un beneficio habilitado.
Regla: solo se procesa si su saldo alcanza.
Resultado esperado: aparece confirmación con comprobante y saldo actualizado.
Excepciones: sin saldo, error del servicio, solicitud repetida.

> La documentación se vuelve útil cuando otra persona puede implementar la operación **sin tener que adivinar los casos incómodos**.

03 / DOCUMENTACIÓN · DESARROLLO

De una necesidad a una especificación verificable

La documentación funciona como puente entre el problema de usuario, las reglas del dominio y las decisiones de implementación.

Una necesidad como «quiero utilizar mis puntos» no especifica todavía qué debe construir el equipo. Puede haber límites de saldo, vigencias, condiciones de disponibilidad o reglas de concurrencia. La documentación funcional transforma esa necesidad en una operación con precondiciones, un resultado y alternativas. El componente visual será una forma de representar dicha operación, no la operación misma.

Separar lo funcional de lo técnico

La documentación funcional no debería depender de la elección de React, Angular o cualquier otra tecnología. Por ejemplo, exigir un comprobante después de un canje exitoso es independiente de cómo se obtenga. La documentación técnica traduce esa exigencia a contratos del frontend y del servidor: qué datos devuelve la API, cómo se interpreta una respuesta y qué prueba automática verifica el resultado.

Elemento	Ejemplo de especificación
Objetivo	Permitir que una persona use sus puntos sin incertidumbre sobre el resultado.
Precondiciones	Sesión válida, beneficio disponible, saldo suficiente al confirmar.
Flujo principal	Elegir beneficio, revisar costo, confirmar, recibir comprobante y saldo vigente.
Alternativas	Si falta saldo, informar cuánto falta. Si el resultado es incierto, consultar su estado antes de repetir.
Contrato técnico	Respuesta de éxito con identificador de operación y saldo confirmado; error con código interpretable.
Verificación	Caso de éxito, saldo insuficiente, fallo y repetición de solicitud.

Las decisiones de interacción se expresan en términos observables: «el sistema conserva la selección cuando hay un error recuperable» es verificable; «la interfaz debe ser intuitiva» no lo es. Una imagen puede acompañar, pero no reemplazar esa descripción.

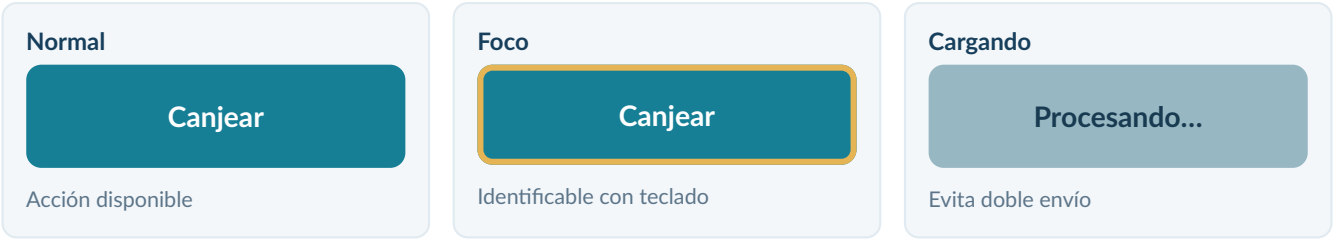
> **Para el TP:** incluir el enlace y la versión del prototipo, nombrar la operación y documentar los comportamientos que otro desarrollador debería conocer. No es necesario diseñar toda la API ni definir una arquitectura de producción.

04 / COMPONENTES Y COMPORTAMIENTOS

El contrato de un componente

Una biblioteca de UI define piezas reutilizables. Su contrato aclara **qué reciben, qué representan y qué hacen**. Los tokens establecen valores compartidos, pero la conducta requiere especificación propia.

Aspecto	Ejemplo: botón «Canjear»
Propiedades	label, disabled, loading, onActivate; variante primaria.
Estados	normal, hover, foco visible, presionado, deshabilitado y carga.
Resultado	Al activarse, inicia la solicitud una sola vez y comunica su progreso.
Accesibilidad	Nombre accesible, foco visible, activación con teclado; no depender del color.
Exclusiones	No decide el saldo ni actualiza la base de datos: esa lógica pertenece al dominio.



Contrato antes que estética

Separar responsabilidad

El componente se encarga de representar e interactuar; las reglas de negocio deben definirse fuera o en capas específicas.

Describir casos extremos

Texto más largo, conexión lenta, error y datos faltantes: son parte del diseño, no excepciones para después.

> No confundamos propiedades con permisos: que exista un estado *disabled* en el componente no sustituye una validación de seguridad en el servidor.

04 / COMPONENTES · DESARROLLO

Una ficha de entrega de componente

Un componente reutilizable es una unidad de interfaz con responsabilidad definida. Su documentación debe permitir implementarlo, utilizarlo correctamente y probarlo sin reconstruir las decisiones de diseño.

Las propiedades (props) expresan la información que recibe; las variantes corresponden a formas permitidas de representación, y los estados describen circunstancias temporales de uso. No es equivalente una variante secundaria a un botón deshabilitado: la primera modifica su función visual dentro de una jerarquía, el segundo modifica qué acciones están disponibles.

Campo	Ficha de ejemplo: RedemptionAction
Propósito y contexto	Confirmar un canje desde la pantalla de revisión. Usarlo para la acción final, no para entrar al catálogo.
Entradas	label: cadena; busy: booleano; disabled: booleano; variant: primary; onActivate: función.
Estados visuales	Normal, foco, hover, presionado, deshabilitado y procesando. Error y éxito pertenecen al flujo que lo contiene.
Comportamiento	Activación por clic o teclado. Mientras busy, impedir nuevos envíos; comunicar «Procesando…».
Accesibilidad	Elemento button nativo; nombre claro; indicador de foco; mensaje de progreso en región de estado.
Límites	No calcula saldo ni decide permisos. Recibe elegibilidad del flujo; el servidor vuelve a validar.
Aceptación	Con una solicitud en curso, un nuevo clic no genera otra. El resultado se comunica claramente.
Referencia	Vincular al componente del diseño y sus tokens. Registrar versión y responsable.

Por qué conviene limitar el contrato

Un componente que acumula las reglas de todas las operaciones termina difícil de reutilizar. Separar presentación y lógica permite cambiar el estilo de un botón sin alterar cálculos de saldo o seguridad. También ayuda a escribir pruebas pequeñas: el componente presenta estados; el flujo orquesta la solicitud; el servidor garantiza la transacción.

La implementación dependerá de la biblioteca elegida. Lo importante es que la interfaz pública del componente sea previsible y que sus variantes, estados y restricciones estén documentados junto a ejemplos de uso.

Referencias para profundizar: [17].

05 / ESPECIFICACIÓN DE ESTADOS

Estados de una operación, no solo de un botón

Un flujo transaccional necesita anticipar qué pasa antes, durante y después de enviar datos. Las transiciones entre estados son parte de la experiencia y del contrato técnico.



Figura. Máquina de estados simplificada: la interfaz debe comunicar transiciones y permitir recuperarse.

Estado	Qué muestra	Qué puede hacer la persona
Vacío	«Todavía no tenés beneficios» + explicación	Descubrir cómo obtener puntos o volver.
Listo	Saldo, detalle del beneficio y acción principal	Confirmar o cambiar elección.
Carga	Mensaje de espera y prevención de repetición	Esperar, sin disparar otra transacción.
Éxito	Confirmación, código de operación y saldo actualizado	Volver o descargar comprobante.
Error	Causa comprensible si se conoce; próximo paso	Reintentar de forma segura o pedir ayuda.

> La mejor especificación combina **estados, textos y reglas de transición**. Una pantalla fija no alcanza para describir una operación.

05 / ESTADOS · DESARROLLO

Pensar la interacción como una máquina de estados

Una operación no es una secuencia de pantallas necesariamente lineal. La interfaz representa un estado que cambia en respuesta a acciones o resultados del sistema.

Un estado combina información relevante: ¿hay una selección válida?, ¿hay una solicitud pendiente?, ¿existe una confirmación? Una transición expresa el paso entre ellos, provocado por una entrada o una respuesta. Es útil representar explícitamente los estados para evitar combinaciones imposibles, como un canje confirmado que sigue ofreciendo la misma acción de confirmación.

Transiciones de un canje

Desde	Evento o condición	Hacia / efecto
Listo	Confirmación válida	Enviando; informar progreso.
Enviando	Respuesta exitosa	Éxito; mostrar comprobante.
Enviando	Rechazo confirmado	Error; explicar causa y recuperación.
Enviando	Timeout o desconexión	Resultado incierto; consultar estado.
Error recuperable	Usuario vuelve a intentar	Enviando tras verificar condiciones.
Éxito	Volver al catálogo	Nuevo flujo; evitar repetir la operación anterior.

La respuesta de interfaz debe distinguir un rechazo de negocio de un problema de comunicación. Si el servidor confirma saldo insuficiente, puede explicarse cómo resolverlo. Si la red falla después de enviar la solicitud, no sabemos necesariamente si el servidor realizó el canje. Mostrar «falló, reintentá» sin comprobar el resultado puede provocar operaciones duplicadas.

En un sistema real, deshabilitar el botón ayuda a reducir envíos accidentales, pero no garantiza una sola transacción. Se necesita protección en el servidor, por ejemplo mediante una clave de idempotencia asociada a la operación lógica, junto con reglas de consistencia para actualizar saldo y emitir la confirmación. Son responsabilidades distintas de las de un control visual.

> **En el prototipo:** representar la carga, el éxito y los posibles errores. Una simulación ayuda a comprender el comportamiento esperado, pero no permite certificar que el servicio sea seguro o idempotente. Especificar qué deberá verificarse cuando exista una implementación real.

06 / CALIDAD Y CONTROL DE CAMBIOS

Criterios de aceptación y definición de terminado

Los criterios de aceptación describen lo que una funcionalidad debe cumplir. La *Definition of Done* (DoD) es un acuerdo de calidad que se aplica a la entrega, no solo a una historia.

Ejemplo de criterios verificables

Dado un usuario con 1.800 puntos y un beneficio de 1.200,
cuando confirma el canje,
entonces el sistema registra una sola operación, muestra un comprobante y actualiza el saldo a 600.

Dado un usuario con menos puntos que los requeridos,
cuando intenta acceder a ese canje,
entonces ve el saldo insuficiente y no puede completar la operación.

Una DoD razonable para este ejercicio

- ✓ El flujo principal y los estados relevantes funcionan de extremo a extremo.
- ✓ Las pantallas responden a tamaños móviles y de escritorio definidos.
- ✓ Es operable mediante teclado, con foco y mensajes comprensibles.
- ✓ Se verificaron los criterios de aceptación, errores y recuperación.
- ✓ La documentación, los eventos y la versión entregada coinciden.

Control de cambios

Cambio solicitado	Análisis breve antes de hacerlo
«Agreguemos confirmación doble»	Impacto en el flujo, fricción adicional, estados, tests y métricas.
«Cambiemos la etiqueta a Canjear ahora»	Efecto en comprensión, documentación y futuras pruebas.
«El backend devuelve otro error»	Actualizar contrato, copy, trazabilidad y caso de prueba.

> El cambio no se prohíbe: se registra su motivo, responsable, impacto y verificación. Es la forma de evitar que diseño y código evolucionen por separado.

06 / ACEPTACIÓN Y CAMBIOS · DESARROLLO

Cómo escribir criterios que puedan probarse

Los criterios de aceptación son condiciones particulares de una funcionalidad. Una definición de terminado (DoD) establece un nivel de calidad común al incremento de producto.

La técnica Dado–Cuando–Entonces ayuda a transformar una expectativa en un escenario de prueba. «Dado» describe la situación inicial; «cuando», el evento; «entonces», el resultado observable. Los criterios no deben imponer el detalle de implementación cuando solo importa el resultado. Deben tener suficiente precisión para que dos personas puedan comprobarlos y obtener la misma conclusión.

Dado / cuando	Entonces (resultado comprobable)
Saldo de 1.800 y costo 1.200; confirma	Una operación registrada, comprobante visible y saldo resultante 600.
Saldo insuficiente; consulta el beneficio	Se explica cuánto falta y el canje no puede confirmarse.
Solicitud pendiente; activa nuevamente el control	No se dispara una segunda operación lógica.
Error recuperable; vuelve a la pantalla	Conserva la selección y puede continuar según las reglas definidas.
Navega con Tab; llega a confirmar	El foco es visible y la acción puede activarse con teclado.

Un criterio puede satisfacerse en forma aislada y aun así la entrega no estar terminada. Un canje que funciona con mouse pero no con teclado incumple una condición general de calidad. La DoD hace visibles esas condiciones: pruebas pertinentes aprobadas, documentación actualizada y verificaciones de accesibilidad acordadas, entre otras.

Gestionar el cambio

Si se decide agregar una confirmación intermedia, registrar el origen (hallazgo o requisito), el motivo, el alcance y los artefactos afectados. La decisión puede cambiar el flujo, la medición de inicio y finalización y los casos de prueba. Después se actualizan las versiones de diseño y código, se verifica el nuevo comportamiento y se registra el resultado. Un registro pequeño con fecha, decisión, responsable y prueba asociada suele alcanzar para un proyecto de aula.

Referencias para profundizar: [18].

07 / CALIDAD PARA PERSONAS REALES

Accesibilidad digital y diseño inclusivo

La accesibilidad busca que personas con distintas capacidades, dispositivos y contextos puedan percibir, comprender y operar un producto. No es una mejora estética opcional ni se limita a usuarios con discapacidad permanente.

01
Visual
Bajo contraste, texto pequeño, zoom, visión reducida.

02
Motora
Navegación sin mouse, objetivos táctiles, precisión limitada.

03
Auditiva
Subtítulos, alternativas a información sonora.

04
Cognitiva
Lenguaje claro, procesos previsibles, recuperación de errores.

Diseñar para situaciones diversas

Permanentes
Una persona ciega necesita información que no dependa de la imagen ni de una posición visual.

Temporales
Una lesión en la mano puede hacer necesario usar teclado, comandos de voz o navegación con una sola mano.

Situacionales
Luz intensa, mala conexión, uso del teléfono en movimiento o ruido ambiental cambian la experiencia.

Combinadas
No basta con “resolver para un perfil”: las barreras interactúan. Contraste, textos, foco y tiempos pueden fallar simultáneamente.

> Diseño inclusivo significa incorporar perspectivas diversas desde el inicio. No equivale a afirmar que un producto es accesible solo porque pasa un validador.

Fuentes para profundizar: [1], [2], [3]

La discapacidad no explica por sí sola una dificultad de uso; también importa la interacción entre las características de la persona y las barreras del entorno. Un formulario ilegible con zoom o una tarea que requiere arrastrar elementos sin alternativa genera una barrera que el equipo puede reducir. El diseño inclusivo aprovecha la diversidad de situaciones para anticipar requisitos en lugar de delegarlos a una revisión final.

08 / ESTÁNDARES DEL W3C

WCAG 2.2: principios y niveles

Las **Web Content Accessibility Guidelines (WCAG)** son recomendaciones del W3C para hacer contenido web más accesible. La versión 2.2 mantiene cuatro principios, criterios de éxito comprobables y niveles de conformidad A, AA y AAA.

P
Perceptible
Texto alternativo, subtítulos y suficiente contraste.

O
Operable
Teclado, tiempo suficiente, navegación y foco.

U
Comprensible
Lenguaje claro, consistencia y ayuda ante errores.

R
Robusto
Compatibilidad con tecnologías de asistencia.

Cómo leer los niveles

Nivel	Qué representa	Enfoque de esta materia
A	Requisitos fundamentales.	Identificar y evitar bloqueos graves.
AA	Criterios más amplios de accesibilidad; objetivo frecuente de proyectos.	Reconocer y verificar criterios seleccionados.
AAA	Exigencias adicionales, no siempre aplicables a todo tipo de contenido.	Comprender su existencia, sin exigir conformidad total en el TP.

> Las WCAG son un estándar, no un kit de componentes. Cumplir una lista de criterios requiere evaluación del producto real. Una captura de Figma no demuestra conformidad.

WCAG 2.2 fue publicada como Recomendación del W3C en 2023 y actualizada editorialmente en 2024; sus nuevos criterios amplían la atención a foco, tamaño de objetivos, interacción y autenticación.

Fuentes para profundizar: [1], [2]

08 / WCAG 2.2 · DESARROLLO

Cómo interpretar una norma de accesibilidad

El W3C produce estándares y recomendaciones para la Web; su iniciativa WAI aborda específicamente la accesibilidad. WCAG 2.2 organiza sus requisitos en cuatro principios, pautas y criterios de éxito verificables. Los documentos *Understanding* y *Techniques* ayudan a interpretar los criterios y muestran estrategias posibles, pero no son una lista única de implementaciones obligatorias.

Los cuatro principios se complementan. Una etiqueta visible puede ser perceptible para quien ve la pantalla, pero no robusta si su relación con un campo no está disponible para las tecnologías de asistencia. Un botón con excelente contraste sigue siendo inoperable si no se activa con teclado. Un mensaje de error audible puede ser comprensible o no, dependiendo de si indica qué pasó y cómo corregirlo.

Conformidad y alcance

El nivel AA supone cumplir los criterios aplicables de A y AA, no obtener un promedio favorable. Una evaluación de unas pocas pantallas no permite declarar conformidad de todo un producto. Los requisitos también abarcan páginas completas y los pasos que forman un proceso, junto con otras condiciones de conformidad. El alcance de la evaluación debe registrarse.

Pregunta	Criterio de ejemplo	Verificación
¿Se percibe el texto?	1.4.3 Contraste mínimo (AA)	Medir pares de colores reales.
¿Se opera sin mouse?	2.1.1 Teclado (A)	Recorrer y completar la tarea.
¿Se entiende el error?	3.3.1 Identificación de errores (A)	Provocar error y evaluar indicación textual.
¿Se conocen nombre y rol?	4.1.2 Nombre, función, valor (A)	Inspección semántica y tecnología de asistencia.

> **Ejemplo.** Un prototipo en Figma permite medir contraste de colores especificados y observar el orden visual. No permite concluir que la implementación final asigna nombres accesibles, anuncia estados o respeta el orden real de foco. Informar lo verificado y lo pendiente es parte de un análisis riguroso.

Las obligaciones legales y contractuales pueden variar según el contexto. En esta materia se utiliza WCAG 2.2 como referencia técnica de trabajo, no como certificación jurídica.

Referencias para profundizar: [1], [2], [19].

09 / CRITERIOS QUE CONVIENE SABER REVISAR

WCAG 2.2 en decisiones de interfaz

Estos ejemplos no reemplazan la norma. Sirven para reconocer problemas frecuentes en una web transaccional y empezar a verificarlos.

Criterio	Qué exige o persigue	Ejemplo de revisión
1.4.3 · Contraste mínimo (AA)	Texto normal, al menos 4,5:1; texto grande, 3:1, con excepciones.	Texto sobre botón, placeholder y mensajes.
1.4.11 · Contraste no textual (AA)	Componentes e información gráfica necesaria con contraste adecuado, normalmente 3:1.	Bordes, estados y foco cuando son necesarios.
2.1.1 · Teclado (A)	Operar funciones mediante teclado, salvo excepciones específicas.	Completar canje sin mouse.
2.4.7 / 2.4.11 · Foco (AA)	Foco visible; el componente enfocado no debe quedar totalmente oculto.	Encabezado fijo sobre el campo activo.
2.5.8 · Tamaño de objetivo (AA)	Objetivos de al menos 24 × 24 CSS px o excepciones por espaciado/otros casos.	Cerrar modal y pequeños iconos táctiles.
3.3.8 · Autenticación accesible (mínima, AA)	No exigir pruebas cognitivas sin alternativas permitidas.	Evitar “recordá esta clave” como único método.

> **Prueba de clase:** recorran con Tab un formulario, amplíen el zoom del navegador y desactiven la apariencia de hover. ¿Se reconoce dónde está el foco, qué está habilitado y cómo continuar?

No confundir el mínimo de 24 px del criterio 2.5.8 con otras recomendaciones de tamaño más generosas. El cumplimiento depende del detalle de la excepción aplicable.

Fuentes para profundizar: [1], [2], [3], [4]

09 / CONTRASTE Y OPERABILIDAD · DESARROLLO

De un requisito WCAG a una comprobación concreta

Una verificación de contraste no consiste en decidir a simple vista si un texto «se lee bien». Exige conocer el par de colores que realmente aparece en la interfaz.

WCAG utiliza una razón basada en luminancia relativa; el cálculo compara el color más claro contra el más oscuro. Para texto normal, el criterio 1.4.3 establece 4,5:1 como mínimo (AA), y para texto grande 3:1. Se considera grande el texto de 18 puntos o 14 puntos en negrita, aproximadamente 24 o 18,67 CSS px, respectivamente. Existen excepciones como logotipos y controles inactivos.

$$\text{Contraste} = (L_{\text{clara}} + 0{,}05) / (L_{\text{oscura}} + 0{,}05)$$

Combinación	Razón aprox.	Interpretación para texto normal
#777 sobre blanco	4,48:1	No alcanza 4,5:1; no redondear para aprobar.
#595959 sobre blanco	7,0:1	Cumple el requisito de contraste AA.
Blanco sobre #174D78	8,86:1	Cumple el requisito de contraste AA.

Si hay transparencia, degradados o una fotografía de fondo, deben considerarse los colores resultantes en las condiciones menos favorables. Un estado de error no debe depender únicamente de rojo: el texto puede decir «Saldo insuficiente» y explicar qué hacer a continuación.

Verificar objetivos táctiles y foco

El tamaño de un icono no equivale necesariamente a su área interactiva. Un icono de 16 px dentro de un botón de 44 px puede ser cómodo de activar. El criterio 2.5.8 de WCAG 2.2 establece un mínimo de 24 × 24 CSS px o excepciones definidas, entre ellas determinadas separaciones entre objetivos pequeños. Un indicador de foco debe ser visible y, en AA, no quedar completamente oculto por elementos creados por el autor, como una barra fija.

En la auditoría se debe registrar el criterio, la evidencia y el resultado. Una sugerencia estética sin medición ni pasos de reproducción no alcanza para verificar estos requisitos.

Referencias para profundizar: [4], [19].

10 / EL NAVEGADOR NECESITA ENTENDER LA PÁGINA

HTML semántico y estructura documental

Las etiquetas semánticas describen la función del contenido: ayudan a navegadores, lectores de pantalla, buscadores y a los equipos que mantienen el código. No alcanza con que algo *se vea* como un título o un botón.

```
<header>Mi Beneficio</header>
<nav aria-label="Principal"> ... </nav>
<main id="contenido">
  <h1>Canje de beneficios</h1>
  <section aria-labelledby="disponibles">
    <h2 id="disponibles">Beneficios disponibles</h2>
    <article> ... </article>
  </section>
</main>
<footer>Ayuda y condiciones</footer>
```

Landmarks y encabezados

Elemento	Función semántica
`header`	Cabecera general cuando corresponde al contexto de página; no cada header interno es un landmark.
`nav`	Región de navegación. Puede llevar nombre accesible cuando hay varias.
`main`	Contenido principal de la página; normalmente una única región principal.
`h1 … h6`	Jerarquía temática, no escala visual. Usar niveles con sentido.
`button` / `a`	Botón para ejecutar acción; enlace para navegar a otra dirección.

> Un `div onclick` que parece botón no trae automáticamente semántica, foco ni interacción con teclado. Siempre que sea posible, **preferí controles nativos**.

Fuentes para profundizar: [3], [5]

10 / HTML SEMÁNTICO · DESARROLLO

Lo que el navegador y el lector de pantalla interpretan

El navegador no infiere la función de un elemento a partir de su aspecto visual. Construye una estructura de elementos a partir del HTML y expone información semántica a las tecnologías de asistencia mediante un árbol de accesibilidad. Este árbol incorpora nombres, roles, estados y relaciones que pueden diferir de la representación gráfica.

Los *landmarks* identifican regiones importantes de la página: navegación, contenido principal, cabecera o pie en el contexto adecuado. Los encabezados permiten reconocer la organización temática. Una persona que usa lector de pantalla puede saltar directamente a la navegación o recorrer los títulos sin leer todo el contenido anterior. La semántica también beneficia a quienes mantienen el código, porque hace visible su propósito.

```
<a href="#contenido">Ir al contenido principal</a>
<header><a href="/">Mi beneficio</a></header>
<nav aria-label="Principal">
  <a href="/beneficios">Beneficios</a>
  <a href="/historial">Historial</a>
</nav>
<main id="contenido">
  <h1>Beneficios disponibles</h1>
  <section aria-labelledby="cafe">
    <h2 id="cafe">Cafeterías</h2>
    <a href="/beneficios/cafe">Ver café por 800 puntos</a>
  </section>
</main>
<footer><a href="/ayuda">Ayuda</a></footer>
```

En el ejemplo, el enlace inicial permite omitir bloques de navegación repetidos. La cabecera y el menú ofrecen destinos; `main` identifica el contenido principal. El encabezado de nivel 1 describe la página y el de nivel 2 denomina una sección. Su tamaño visual se decide con CSS: no se debe usar un h4 solo porque se quiere un texto más pequeño.

Los enlaces llevan a destinos; el botón ejecuta una acción. Elegir correctamente elimina trabajo manual de semántica y teclado. Un enlace con `href` también ofrece comportamientos propios, como abrir el destino en otra pestaña. ARIA puede complementar la información cuando HTML por sí solo no alcanza, pero no añade automáticamente el comportamiento que necesita un widget.

Referencias para profundizar: [3], [5], [19].

11 / SEMÁNTICA APLICADA A LA TAREA

Formularios, foco y mensajes comprensibles

Las pantallas transaccionales suelen concentrar barreras de accesibilidad en formularios y mensajes. Cada campo necesita nombre claro, indicación de requisitos y relación con sus errores.

```
<form action="/canje" method="post">
  <label for="beneficio">Beneficio</label>
  <select id="beneficio" name="beneficio" required>
    <option value="">Seleccionar beneficio</option>
    <option value="cafe">Café por 800 puntos</option>
  </select>
  <button type="submit">Confirmar canje</button>
</form>
```

Con controles nativos, el navegador ofrece comportamiento de teclado y semántica básica. Para errores de negocio y estados asincrónicos se necesitarán mensajes asociados y una estrategia de foco.

Antes de enviar

Etiquetas visibles, requisitos claros, acciones diferenciadas. No usar el placeholder como única etiqueta ni el color como única señal.

Después de enviar

Confirmación identificable o error que explique qué pasó y cómo recuperar la tarea. Si el mensaje cambia dinámicamente, considerar anuncios accesibles.

Verificación mínima

- ✓ ¿El orden del Tab sigue la lógica visual y de la tarea?
- ✓ ¿Los selectores y botones son operables con teclado?
- ✓ ¿Se identifica el error sin depender solo de rojo o de un icono?
- ✓ ¿El mensaje de éxito tiene un lugar claro y es legible?

> No conviertan cada elemento en un widget ARIA personalizado. El HTML nativo suele resolver mejor el comportamiento básico con menos código y menos riesgos.

Fuentes para profundizar: [3], [5], [6]

11 / FORMULARIOS · DESARROLLO

Cómo comunicar errores, foco y cambios de estado

Los formularios son una conversación entre el usuario y el sistema. Una etiqueta inicia esa conversación; la validación y la respuesta permiten continuarla.

El nombre de un control debe identificar su propósito. Vincular un `label` mediante `for` con el `id` del campo permite que el nombre se exponga a tecnologías de asistencia y amplía el área desde la que puede activarse el control. El placeholder puede aportar un ejemplo, pero desaparece cuando se escribe y no debe ser la única etiqueta.

```
<label for="codigo">Código de beneficio</label>
<input id="codigo" name="codigo" required
       aria-describedby="ayuda error" aria-invalid="true">
<p id="ayuda">Usá el código de 6 caracteres.</p>
<p id="error">Ingresá los 6 caracteres del código.</p>
<div role="status" id="resultado"></div>
```

Este ejemplo representa el estado posterior a un error. `aria-invalid` indica que el valor no es válido y `aria-describedby` vincula el campo con la ayuda y el mensaje. Cuando se corrige el valor, el estado y el texto de error se actualizan. No se debe marcar un campo como inválido antes de intentar validarlo, salvo que el contexto lo justifique. La información debe estar visible además de disponible para tecnología de asistencia.

Foco y comunicación dinámica

El foco indica qué elemento recibirá la siguiente interacción de teclado. El orden natural se obtiene normalmente de la estructura del documento; alterar arbitrariamente el orden con tabindex positivos dificulta el recorrido. No eliminar el indicador visual del foco sin ofrecer una alternativa visible. Si un diálogo modal se abre, el foco se ubica dentro de él y al cerrarlo vuelve normalmente al control que lo abrió.

Cuando un canje termina de manera asincrónica, modificar una región existente con `role="status"` puede permitir que el lector de pantalla anuncie un resultado breve sin mover el foco. Para mensajes urgentes existen otras estrategias; no se debe anunciar cada cambio de forma intrusiva. Si se navega a una nueva pantalla, se comprueba que el título y el punto de inicio de lectura tengan sentido.

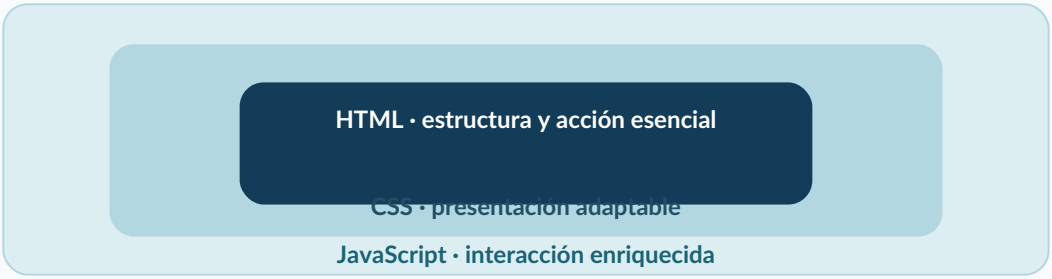
> **Verificación:** ingresar un código inválido con teclado, localizar el mensaje, corregirlo y completar la tarea. Registrar qué ocurre visualmente, qué se anuncia y si se puede continuar. El código es ilustrativo: el manejo de estados y las validaciones reales deben implementarse.

Referencias para profundizar: [5], [6], [19].

12 / QUÉ QUEDA SIN CSS NI JAVASCRIPT

Mejora progresiva: la capa base importa

La **mejora progresiva** parte de una experiencia funcional con HTML y agrega presentación con CSS y comportamientos enriquecidos con JavaScript. Si una capa falla, el contenido y las acciones esenciales deberían degradarse de forma comprensible.



Capas conceptuales: la estructura semántica no debería depender de la decoración.

Qué probar concretamente

Sin…	Qué esperaría en una página bien concebida
CSS	Texto, títulos, orden, enlaces y controles siguen siendo comprensibles.
JavaScript	Las páginas públicas siguen siendo accesibles y los formularios importantes pueden contar con una ruta funcional basada en servidor, si el sistema la implementa.
Imágenes	El texto alternativo útil comunica el propósito de contenido informativo; las imágenes decorativas no agregan ruido.
Animaciones	La acción se entiende sin depender del movimiento; se respeta preferencia por movimiento reducido cuando corresponde.

> Un prototipo HTML estático sin backend puede simular un canje, pero no completarlo realmente sin una implementación de servidor. “Funcionar sin JavaScript” es una decisión arquitectónica que necesita soporte real.

Fuentes para profundizar: [5], [7]

12 / MEJORA PROGRESIVA · DESARROLLO

Cómo diseñar una experiencia que tolere fallas

Un sitio web atraviesa diferentes condiciones de ejecución. La hoja de estilos puede tardar, un recurso JavaScript puede fallar y la conexión puede interrumpirse. Diseñar solo para el escenario en que todos los recursos funcionan oculta esas dependencias. La mejora progresiva propone una base que sea significativa por sí misma y capas posteriores que enriquezcan el comportamiento cuando estén disponibles.

HTML: contenido, estructura y controles fundamentales. **CSS:** organización visual, adaptación y jerarquía. **JavaScript:** comportamientos enriquecidos, actualizaciones dinámicas y respuestas inmediatas. Esta división no prohíbe aplicaciones complejas con JavaScript; pide razonar sobre qué parte de la experiencia necesita cada capa y qué sucede cuando falta.

```
<!-- Estructura base: el servidor debe aceptar el POST -->
<form action="/canje" method="post">
  <button type="submit">Confirmar canje</button>
</form>
```

El formulario puede enviar una petición al servidor sin código JavaScript, siempre que exista una ruta real que procese la operación, valide sesión y reglas de negocio, proteja la solicitud y responda con HTML adecuado. Luego se puede incorporar JavaScript para interceptar el envío, mostrar carga y actualizar el comprobante sin recargar la página. Ambos caminos deben respetar las mismas reglas y no duplicar la operación.

Ejercicio de inspección

Condición	Qué observar
Sin CSS	Orden del contenido, enlaces, etiquetas, lectura de la tarea.
Sin JavaScript	Contenido público, navegación, alternativa de operación o explicación de la limitación.
Con red lenta	Qué sucede antes de que llegue la respuesta; si el usuario entiende la espera.
Con imágenes bloqueadas	Información alternativa cuando la imagen aporta significado.

El objetivo no es imponer que cualquier aplicación funcione íntegramente sin JavaScript. Por ejemplo, una herramienta de edición compleja puede requerirlo. En ese caso se debe declarar la dependencia y brindar una experiencia de fallo comprensible; el contenido informativo y las tareas compatibles con una base más simple siguen siendo oportunidades para aplicar el principio.

Referencias para profundizar: [5], [7].

13 / EVALUACIÓN DE ACCESIBILIDAD

Auditoría automática + verificación humana

Las herramientas automáticas pueden detectar fallas recurrentes y acelerar la revisión. Pero **no pueden certificar por sí solas que una experiencia es accesible**. La evaluación manual sigue siendo necesaria.

Automatización puede señalar…	Una persona debe comprobar…
Atributos alternativos ausentes o etiquetas mal vinculadas.	Si el texto alternativo comunica el significado correcto.
Contrastes calculables y ciertos errores de estructura.	Si la interacción y el foco son comprensibles en el flujo.
Referencias ARIA o IDs inconsistentes.	Si el lector de pantalla anuncia resultados y errores de forma útil.
Algunas violaciones de reglas WCAG detectables en el DOM.	Si el contenido mantiene sentido al ampliar zoom o cambiar dispositivo.

Secuencia de auditoría de aula

Lighthouse / axe hallazgos iniciales → **Teclado + zoom** recorrido manual → **Lector de pantalla** muestra de tareas → **Registro** corrección y retest

Herramientas introductorias

Lighthouse (auditoría web), axe DevTools y WAVE (análisis de barreras). Consultar el alcance de cada herramienta antes de interpretar resultados.

Qué registrar

Pantalla, criterio afectado, pasos para reproducir, impacto, evidencia, solución y verificación posterior.

> «Lighthouse 100» no equivale a cumplir toda WCAG 2.2. Los puntajes resumen un subconjunto de verificaciones en condiciones específicas.

Fuentes para profundizar: [8], [9]

13 / AUDITORÍA · DESARROLLO

Una auditoría de aula, paso a paso

La evaluación automática analiza propiedades que pueden expresarse como reglas. La revisión manual interpreta si las personas pueden comprender y utilizar el producto.

Una herramienta puede detectar que una imagen no tiene atributo alt, pero no determinar de manera fiable si su descripción es pertinente. Puede calcular ciertos contrastes cuando los estilos son accesibles para su motor, pero no evaluar todos los estados, combinaciones o fondos variables. Un resultado sin errores detectados significa que la herramienta no encontró violaciones en las reglas ejecutadas, no que el producto sea accesible en su totalidad.

Procedimiento sugerido

1. Seleccionar una tarea y declarar la versión, el dispositivo, las páginas y estados evaluados. **2.** Ejecutar una herramienta como axe o Lighthouse y revisar cada hallazgo; conservar el detalle, no solo la puntuación. **3.** Completar la tarea con teclado, revisar foco, orden, errores y recuperación. **4.** Ampliar texto y viewport, y comprobar con lector de pantalla una muestra de la tarea. **5.** Registrar, priorizar y repetir las comprobaciones después de corregir.

Registro	Ejemplo aplicado
Pantalla / criterio	Confirmación de beneficio · 2.4.7 Foco visible (AA).
Reproducción	Usar Tab hasta Confirmar en vista escritorio.
Evidencia	El foco llega al control, pero el estilo elimina el contorno sin alternativa.
Impacto	La persona no sabe qué acción recibirá Enter.
Corrección	Indicador visual de foco distinguible.
Verificación	Repetir recorrido con teclado; registrar resultado y versión.

Una herramienta de auditoría automática complementa, no reemplaza, técnicas de prueba con personas con discapacidad. La revisión experta comprueba criterios y detecta barreras; las pruebas con usuarios muestran cómo afectan una tarea real. Estas formas de evidencia son complementarias, como se estudió en la Unidad 5.

En Figma se puede observar contraste y especificación de estados, pero cualquier verificación del DOM, del foco nativo o de un lector de pantalla debe quedar pendiente del código. En el TP no se exige una auditoría formal completa.

Referencias para profundizar: [8], [9], [19].

14 / BÚSQUEDA Y ESTRUCTURA DOCUMENTAL

SEO técnico: hacer visible lo que es público

El SEO técnico ayuda a que un buscador **descubra, procese y comprenda** contenido público. No sustituye la relevancia, el contenido útil ni el propósito del sitio.

1
Descubrimiento
Enlaces rastreables, sitemap, arquitectura de URLs.

2
Procesamiento
Respuesta HTTP, HTML, recursos y renderizado.

3
Indexación
Decisión de incluir o excluir contenido de búsqueda.

4
Presentación
Título, fragmentos, datos estructurados cuando aplican.

Lista técnica inicial

- ✓ URLs comprensibles y navegación mediante enlaces reales.
- ✓ `<title>` único, encabezado principal y descripción útil por página.
- ✓ Estado HTTP correcto, redirecciones y enlaces internos verificables.
- ✓ Sitemap cuando corresponde, y robots.txt sin bloqueos accidentales.
- ✓ Canonical, meta robots y datos estructurados solo cuando hay necesidad concreta.

> `robots.txt` controla el rastreo, no es un mecanismo universal para impedir indexación. `noindex` necesita que el buscador pueda acceder a la página para leer la indicación. Y un sitemap no garantiza aparecer en resultados.

Fuentes para profundizar: [10], [11]

14 / SEO TÉCNICO · DESARROLLO

Cómo llega una página a un buscador

El SEO técnico no es una lista de etiquetas mágicas: consiste en reducir obstáculos para que el contenido público sea descubierto, interpretado y considerado para aparecer en la búsqueda.

El proceso comienza cuando un rastreador conoce una URL por enlaces, sitemaps u otras fuentes. Al recuperar la URL, recibe una respuesta HTTP y puede procesar el documento y sus recursos. Google también puede renderizar JavaScript. Después evalúa si el contenido merece incorporarse al índice y, ante páginas similares, puede seleccionar una versión canónica. Finalmente, cuando alguien busca, decide qué resultados son pertinentes. Una URL rastreada no necesariamente se indexa; una URL indexada tampoco garantiza una posición determinada.

Ejemplo: ficha pública de un beneficio

```
<head>
  <title>Café por 800 puntos | Mi beneficio</title>
  <meta name="description" content="Conocé el beneficio,
    sus condiciones y cómo utilizar tus puntos.">
  <link rel="canonical"
    href="https://ejemplo.com/beneficios/cafe">
</head>
<main>
  <h1>Café por 800 puntos</h1>
  <p>Consultá las condiciones del canje.</p>
  <a href="/beneficios">Ver todos los beneficios</a>
</main>
```

El título comunica la finalidad específica de la página, la meta descripción propone un resumen útil y el encabezado principal identifica el contenido visible. El enlace con href permite descubrir otra URL. El canonical señala la versión preferida cuando existen variantes equivalentes; es una señal, no una orden absoluta. Las direcciones son ficticias y deben reemplazarse al implementar.

El buscador puede utilizar un título o fragmento distinto según la consulta y el contenido, por lo que un título y una descripción adecuados no garantizan que se muestren literalmente. La meta descripción ayuda a describir la página; no debe confundirse con una garantía de posicionamiento.

> **Diagnóstico:** si una ficha pública no aparece, primero comprobar su respuesta, accesibilidad al rastreador, contenido renderizado e indicaciones de indexación. No comenzar cambiando palabras clave sin identificar el problema técnico.

Referencias para profundizar: [10], [11], [20].

15 / PÁGINAS PÚBLICAS Y PRIVADAS

SEO y flujos transaccionales: no todo se indexa

Una web con transacciones tiene páginas de naturaleza distinta. La ficha pública de un beneficio puede ser útil en buscadores; el comprobante o saldo personal no debería aparecer como resultado de búsqueda.

Página	Intención de indexación	Decisión técnica orientativa
Home / información general	Normalmente pública	HTML descriptivo, title y enlaces internos.
Catálogo público de beneficios	Depende del producto	URLs útiles, texto real, performance y enlaces rastreables.
Login o perfil	Generalmente no aporta como contenido de búsqueda	Evaluar noindex cuando corresponda; proteger acceso.
Saldo personal / comprobante	Privada, nunca exponer datos	Autenticación y autorización reales; no confiar en noindex como seguridad.
Resultado de error o página vacía	No indexar como contenido canónico	Estado HTTP correcto, recuperación y mensaje útil.

JavaScript y renderizado

Los buscadores modernos pueden renderizar JavaScript, pero eso agrega pasos. Si el contenido público importante se entrega en HTML inicial o renderizado en servidor, suele ser más predecible para visitantes y rastreadores. En una SPA, aseguren URLs accesibles y enlaces con `href`.

```
<head>
  <title>Beneficios disponibles | Mi Beneficio</title>
  <meta name="description"
        content="Consultá beneficios, requisitos y opciones de canje.">
  <link rel="canonical" href="https://ejemplo.com/beneficios">
</head>
```

> La privacidad no es SEO. Un comprobante sensible debe estar protegido en servidor aunque ningún buscador lo indexe.

Fuentes para profundizar: [10], [11]

15 / SEO · DESARROLLO

Páginas privadas, indexación y JavaScript

Las decisiones de SEO deben partir de la arquitectura de información. Una página de catálogo puede ayudar al público a encontrar el producto; un saldo personal o comprobante tiene valor para una persona identificada, no para una consulta abierta. Hay rutas que pueden ser públicas pero no resulta útil indexar, como ciertos pasos transaccionales o variantes de filtro.

Recurso / condición	Tratamiento razonable
Catálogo y ficha pública	Respuesta 200, enlaces rastreables, contenido descriptivo y canonical coherente cuando corresponda.
Confirmación o saldo privado	Autenticación y autorización en el servidor; no exponer datos en HTML público.
Página pública que no se desea indexar	Directiva noindex mediante meta o encabezado HTTP compatible; permitir que el buscador la lea.
Contenido eliminado	404 o 410 cuando corresponde; 301/308 si existe reemplazo permanente pertinente.
Variantes realmente equivalentes	Canonical a la URL representativa; mantener señales coherentes.

Qué cambia al usar JavaScript

Si la respuesta inicial contiene solamente un elemento vacío y el contenido se obtiene después con JavaScript, el rastreador necesita ejecutar recursos adicionales para interpretar la página. Google puede hacerlo, pero existen fallas posibles: archivos bloqueados, errores de API, contenido dependiente de acciones o respuestas incorrectas. El renderizado desde servidor o la generación estática pueden entregar el contenido principal desde el inicio; deben elegirse por las necesidades de cada proyecto y verificarse.

La revisión tiene que distinguir el HTML inicial del DOM renderizado. Se puede inspeccionar la respuesta del servidor y luego observar el contenido en el navegador; una vez publicado, Search Console permite inspeccionar URLs y revisar problemas de indexación. El sitemap facilita descubrimiento, pero no obliga a indexar. El archivo robots.txt regula rastreo; bloquear una ruta no garantiza que su dirección desaparezca del índice y puede impedir leer noindex.

> **Seguridad no es SEO.** Noindex, robots.txt y canonical no protegen datos personales. Una página privada debe controlar acceso y autorización, independientemente de las preferencias de rastreo.

Referencias para profundizar: [10], [11], [20].

16 / CORE WEB VITALS

Performance y experiencia: medir lo que se siente

Una interfaz clara puede fracasar si tarda demasiado, responde con demora o mueve botones mientras se usa. Las **Core Web Vitals** resumen dimensiones de carga, respuesta y estabilidad visual con datos de usuarios reales.

LCP
Contenido principal
≤ 2,5 s (bueno)

INP
Respuesta a interacciones
≤ 200 ms (bueno)

CLS
Estabilidad visual
≤ 0,1 (bueno)

Qué significan en la práctica

Métrica	Qué evalúa	Qué puede mejorarla
LCP	Cuándo se presenta el mayor elemento de contenido elegible en el viewport.	Optimizar imágenes, servidor y recursos críticos.
INP	Latencia de respuesta a las interacciones durante la visita.	Reducir tareas largas y trabajo del hilo principal.
CLS	Cambios inesperados de disposición visual.	Reservar espacio a imágenes/ads y evitar inserciones sorpresivas.

> Los umbrales de “bueno” se evalúan habitualmente al **percentil 75** de visitas válidas. No confundir un informe de laboratorio con experiencia real de todos los dispositivos.

Herramientas de medición

Lighthouse sirve para auditar y diagnosticar en laboratorio; **PageSpeed Insights** combina información de laboratorio y de campo cuando hay datos disponibles; **Search Console** ofrece reportes de Core Web Vitals por grupos de URL.

SEO y UX comparten consecuencias de performance, pero un resultado “verde” no garantiza visibilidad ni convierte automáticamente una interfaz en usable.

Fuentes para profundizar: [12], [13]

16 / PERFORMANCE · DESARROLLO

LCP, INP y CLS: tres experiencias diferentes

Una interfaz rápida no es solamente una que termina de descargar sus archivos pronto. Una persona necesita ver el contenido importante, obtener una respuesta al actuar y conservar la posición de los elementos mientras avanza. Las Core Web Vitals permiten investigar estas dimensiones por separado; no son un puntaje único ni sustituyen la evaluación de una tarea completa.

LCP · Largest Contentful Paint

Mide el tiempo desde el inicio de navegación hasta que se presenta el mayor elemento de contenido elegible dentro del viewport, por ejemplo una imagen grande o un bloque de texto. En un catálogo, una fotografía principal que tarda en aparecer puede producir una pantalla que parece incompleta. Revisar el tiempo de respuesta del servidor, la prioridad y tamaño de la imagen, los recursos bloqueantes y el retraso de renderizado. No aplicar carga diferida indiscriminadamente a la imagen candidata a LCP.

INP · Interaction to Next Paint

Evalúa la capacidad de respuesta durante la visita mediante las interacciones de clic, teclado y toque, tomando una latencia representativa de las más lentas. Incluye el tiempo de espera antes de procesar la entrada, la ejecución del manejador y el tiempo hasta que el navegador presenta el siguiente cambio visual. Un botón puede tardar en mostrar el estado de carga aunque el servidor remoto sea veloz, si el hilo principal está ocupado ejecutando JavaScript. Mostrar el estado oportunamente y reducir tareas largas mejora esa respuesta. INP no mide por sí solo cuánto demora el servidor en completar un canje.

CLS · Cumulative Layout Shift

Mide la inestabilidad visual mediante el mayor conjunto de desplazamientos inesperados agrupados en ventanas de sesión. No es una duración en segundos. Si una imagen carga sin dimensiones reservadas, el contenido cambia de lugar y la persona puede activar un control distinto del que pretendía. CLS no cuenta de la misma manera todos los cambios: ciertos desplazamientos próximos a una interacción deliberada quedan excluidos.

2,5 s	200 ms	0,1
LCP bueno: hasta este valor	INP bueno: hasta este valor	CLS bueno: hasta este valor

Los umbrales se interpretan al percentil 75 de las visitas medidas, diferenciando móvil y escritorio cuando se evalúa el cumplimiento de los objetivos de Core Web Vitals.

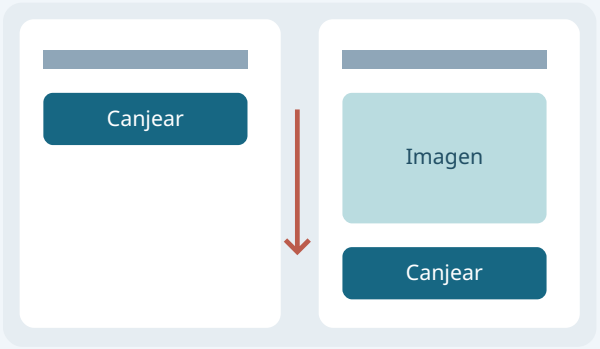
Referencias para profundizar: [12], [13].

16 / PERFORMANCE · APLICACIÓN

Diagnosticar antes de optimizar

Las mediciones permiten identificar dónde se produce una demora o una inestabilidad. La optimización se debe elegir en función de ese diagnóstico.

Supongamos que una ficha de beneficio carga una fotografía a partir de una conexión lenta. Al principio se ve un título y un botón, y segundos después aparece la fotografía y desplaza el botón. Aunque la ficha termine de cargar correctamente, la posición de la acción cambia en un momento en que la persona está por utilizarla.




```
<img src="beneficio.webp" width="800" height="450"
     alt="Taza de café incluida en el beneficio">
/* CSS: img { max-width: 100%; height: auto; } */
```

Las dimensiones intrínsecas permiten que el navegador conozca la proporción antes de descargar la imagen. CSS conserva una adaptación fluida. Si cambian los tamaños de anuncios, banners o contenidos dinámicos, también debe considerarse cómo se reserva espacio y cómo se incorporan a la página.

Laboratorio y campo

Un ensayo de Lighthouse representa condiciones simuladas de carga; permite reproducir problemas y comparar cambios bajo condiciones similares. Los datos de campo reflejan visitas reales y pueden variar por dispositivo, red o versión. PageSpeed Insights presenta ambas fuentes cuando existen datos disponibles. INP necesita interacciones reales o una sesión de diagnóstico interactiva; no debe confundirse con TBT, métrica de laboratorio relacionada con bloqueo del hilo principal. Se debe comparar antes y después, mantener condiciones comparables y revisar el impacto sobre la tarea, además del valor de cada métrica.

Referencias para profundizar: [12], [13].

17 / INDICADORES ESTRATÉGICOS

KPIs: conectar producto, usuarios y negocio

Una métrica cuantifica algo. Un **KPI** es una métrica elegida para seguir un objetivo relevante y actuar sobre él. Muchas mediciones posibles no implican muchas decisiones útiles.

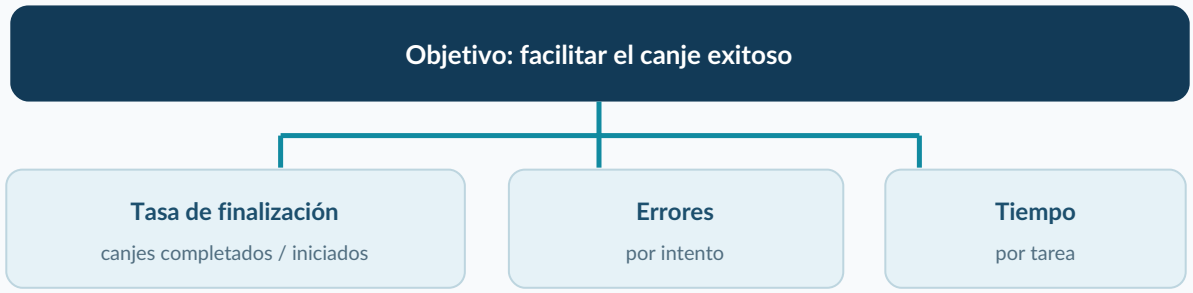


Figura. Un KPI central y métricas de diagnóstico que explican su variación.

Fichar el indicador antes de implementarlo

Campo	Ejemplo
Nombre	Tasa de canje completado.
Definición	Canjes completados / canjes iniciados × 100, dentro del mismo período y población.
Desglose	Dispositivo, tipo de beneficio y versión de experiencia.
Meta	Objetivo fijado desde línea base, no inventado al cerrar el reporte.
Responsable y frecuencia	Product owner; seguimiento semanal y revisión de cambios.

> Definir el denominador importa tanto como medir el numerador: sin una definición estable de «inicio», la tasa puede variar sin que cambie la experiencia.

17 / INDICADORES · DESARROLLO

Diseñar KPIs antes de generar reportes

Un indicador clave de desempeño traduce un objetivo en una medida que permite decidir. No toda métrica sirve como KPI y no toda mejora numérica representa una mejora del producto.

El objetivo estratégico explica qué resultado se busca, por ejemplo que las personas elegibles utilicen sus beneficios con menos errores. En cambio, el número de visitas, clics o impresiones describe actividad. Puede ayudar a diagnosticar qué ocurre, pero no indica necesariamente que la tarea se haya completado. Un KPI requiere una definición de unidad de análisis, período, población, fuente y criterio de interpretación.

Dos KPIs para el caso de canje

Ficha	KPI 1 · finalización	KPI 2 · recuperación
Objetivo	Completar el canje sin abandonar.	Retomar un flujo que sufrió un error recuperable.
Unidad	Intento lógico de canje.	Intento lógico con error recuperable.
Cálculo	Intentos completados / iniciados × 100.	Intentos con error que luego completan / intentos con error recuperable × 100.
Fuente	Eventos de inicio y éxito, emparejados por intento.	Eventos de error y éxito, emparejados por intento.
Ventana	Cohorte semanal de inicios, seguimiento de 24 h (ejemplo).	Misma cohorte; seguimiento de 24 h.
Decisión	Investigar pasos con pérdidas de finalización.	Revisar mensajes y opciones de recuperación.

La ventana y el criterio de deduplicación importan. Si el mismo usuario inicia dos canjes, se contarán dos intentos lógicos; si reenvía una solicitud por una falla temporal dentro del mismo intento, no se convierte automáticamente en otro. Se deben decidir estas reglas antes de obtener números para que los períodos y versiones sean comparables.

Las metas deben establecerse con una línea de base y el contexto. Una tasa puede subir porque cambiaron los beneficios disponibles o el perfil de los usuarios, no porque la interfaz haya mejorado. Se recomienda acompañar los KPIs con medidas de diagnóstico, como errores por tipo y tiempo del proceso, y revisar efectos no deseados.

> **Para el TP:** proponer dos indicadores vinculados al problema original del proyecto y explicar qué harían si el indicador cambia. No hay que producir datos ficticios como si fueran resultados reales.

18 / EVENTOS Y EMBUDOS

Analítica de producto: instrumentar un recorrido

La analítica de producto registra acciones relevantes dentro de un flujo. Permite entender dónde se abandona una tarea y relacionar decisiones de interfaz con resultados medibles.



Embudo ficticio para explicar eventos; no representa resultados reales de alumnos.

Evento sugerido	Cuándo emitir	Parámetros útiles
`benefit_view`	Se visualiza el detalle de un beneficio.	benefit_id, categoría, origen.
`redemption_start`	La persona inicia el proceso de canje.	benefit_id, versión del flujo.
`redemption_error`	La operación no pudo completarse.	error_code no sensible, paso.
`redemption_complete`	El sistema confirma éxito real.	benefit_id, tipo de beneficio.

Los nombres anteriores son **eventos personalizados de ejemplo**. Antes de crearlos revisen si la plataforma ya ofrece un evento recomendado que cubra el caso (por ejemplo, `login`, `search` o `purchase` para una compra real). No emitan `purchase` si solo se canjean puntos sin una transacción comercial definida.

> Un botón pulsado no demuestra un resultado. Diferencien eventos de **intención, intento, error y éxito confirmado**.

Fuentes para profundizar: [14], [15]

18 / ANALÍTICA · DESARROLLO

Leer un embudo sin confundir visitas con operaciones

Un evento registra una ocurrencia definida: la visualización de una ficha, el inicio de una operación o la confirmación del resultado. Una propiedad describe su contexto, como la categoría del beneficio o la versión del flujo. Al reunir eventos relacionados, se puede reconstruir un recorrido, siempre que se defina qué significa pertenecer a la misma secuencia.

Un embudo compara etapas de una tarea. Para calcular sus tasas se debe mantener la misma unidad de análisis. La figura de la página anterior muestra sesiones de ejemplo: no equivale necesariamente a operaciones, pues una sesión podría contener varias. Para este ejercicio supongamos que el embudo está construido sobre 100 sesiones que cumplen la secuencia en orden, dentro del mismo período, contando cada sesión una sola vez por etapa.

Etapa	Sesiones	Tasa desde etapa anterior
Vista del beneficio	100	—
Inicio de canje	64	64 / 100 = 64 %
Canje confirmado	41	41 / 64 ≈ 64,1 %

Finalización desde la vista: 41 / 100 = 41 %

Se observa que 23 de las 64 sesiones que iniciaron no alcanzaron la confirmación en la ventana definida. Esto indica dónde investigar, no la causa. Puede haber saldo insuficiente, fallas de red, dudas, abandonos voluntarios o eventos mal instrumentados. Para distinguirlas se requiere contrastar errores, registros técnicos, segmentación y pruebas con usuarios.

Qué no debe hacer el analista

Comparar un total de clics con un total de personas, o considerar completado un canje cuando se pulsó el botón de confirmación, introduce un error de definición. También es incorrecto restar conteos de eventos sin comprobar su relación con el mismo grupo inicial. El embudo se debe construir con reglas de pertenencia y orden. Para los KPIs por intento de la página anterior, el conteo se realizaría sobre intentos lógicos vinculados y deduplicados, no sobre sesiones.

> La analítica revela patrones de comportamiento agregado; la investigación de la Unidad 5 ayuda a comprender por qué pueden ocurrir. Combinar ambas fuentes permite formular nuevas hipótesis.

Referencias para profundizar: [14], [15].

19 / PLAN DE MEDICIÓN

Especificar eventos sin perder privacidad

No basta con programar llamadas a analytics. Cada evento necesita disparador, propiedades, condición de duplicación, interpretación y responsable de calidad de datos.

Ficha de evento: redemption_complete

Disparador: el servidor confirma la operación y la UI muestra comprobante.
Parámetros: benefit_id (catálogo no sensible), flow_version, device_type.
Se emite: una vez por canje exitoso, con estrategia de deduplicación.
No enviar: nombre, correo, documento, teléfono, texto libre ni identificadores sensibles.
QA: probar caso éxito, fallo y recarga; comprobar que no se duplique.

Separar datos necesarios de datos tentadores

Pregunta	Decisión
¿Para qué medimos esta acción?	Definir hipótesis o KPI al que contribuye.
¿Se puede medir sin identificar a la persona?	Preferir datos agregados o pseudónimos adecuados y minimización.
¿Existe consentimiento o base legal aplicable?	Revisar la política de privacidad y obligaciones del contexto.
¿Quién valida la calidad de los datos?	Asignar responsable y test de instrumentación.

> Un panel de analítica no habilita automáticamente el seguimiento de todo. Revisar consentimiento, privacidad, políticas del proveedor y la normativa aplicable antes de instrumentar eventos.

La finalidad educativa es comprender el contrato de medición. Un prototipo local puede documentar eventos sin enviar información a un servicio externo.

Fuentes para profundizar: [14], [15]

19 / INSTRUMENTACIÓN · DESARROLLO

Del plan de eventos a la verificación de datos

Instrumentar es implementar y comprobar los puntos de observación del producto. Un contrato de eventos permite interpretar los números meses después, incluso si cambia el equipo.

En el inicio del flujo se genera un evento con datos permitidos. Si se desea medir intentos lógicos de operación, el sistema debe disponer de un mecanismo de correlación y deduplicación adecuado. Un evento de confirmación tiene que originarse cuando existe éxito verificable, no cada vez que el componente se renderiza ni cada vez que se abre el comprobante. La fuente puede ser el servidor o una entrega desde el cliente basada en el resultado confirmado, según la arquitectura y las garantías definidas.

Evento	Disparador inequívoco	Prueba de calidad
redemption_start	Se crea un intento lógico de canje al iniciar el flujo.	Un doble clic no crea otro inicio para el mismo intento.
redemption_error	Se informa un fallo registrado en el intento.	Distinguir fallo recuperable y definitivo con códigos controlados.
redemption_complete	El servidor confirma el éxito del intento.	Una recarga del comprobante no emite otro éxito.

```
// Ejemplo ilustrativo con gtag ya configurado.
// Emitir solo tras confirmar y deduplicar el resultado.
gtag('event', 'redemption_complete', {
  benefit_id: 'cafe',
  flow_version: 'v2'
});
```

El código muestra la forma de enviar un evento personalizado; no implementa deduplicación, consentimiento ni lógica de negocio. En algunos sistemas los parámetros requieren registrarse como dimensiones para aparecer en informes habituales. Revisar el catálogo de eventos recomendados y evitar crear diferentes nombres para el mismo significado.

Antes de publicar, usar herramientas de depuración y revisar la secuencia para casos de éxito, error, retroceso y recarga. Controlar duplicados y datos faltantes, distinguir entorno de pruebas de producción y verificar que la medición respete las preferencias de privacidad y las reglas aplicables. No enviar nombres, correos, documentos ni texto libre que pueda contener datos personales. La correlación entre eventos requiere considerar también la minimización y las restricciones del proveedor. Un prototipo puede simular estos eventos localmente sin enviar información externa.

Referencias para profundizar: [14], [15], [21].

20 / INTRODUCCIÓN A EXPERIMENTOS

A/B testing: comparar cambios de forma controlada

Un test A/B asigna participantes o sesiones a variantes para evaluar el efecto de una modificación sobre una métrica. La asignación aleatoria y un tamaño de muestra adecuado son centrales para interpretar diferencias.

Variante A · actual

Canjear
botón al final

Tasa de finalización observada: línea base.

Variante B · propuesta

Confirmar canje
saldo visible antes

Hipótesis: más claridad, menos errores.

Condiciones para una comparación válida

Antes	Durante	Después
Definir una sola hipótesis principal.	Asignación aleatoria, misma población y reglas estables.	Comparar métrica primaria y errores; registrar límites.
Calcular muestra según efecto mínimo y línea base.	No detener el experimento porque una variante “parece ganar”.	No confundir diferencia observada con causalidad probada sin análisis.

> El TP Integrador no necesita un A/B estadísticamente concluyente: **dos usuarios en una prueba cualitativa no sirven para demostrar una mejora cuantitativa entre versiones**. En esta materia, A/B se presenta como metodología introductoria.

Un test de usabilidad detecta obstáculos; un A/B con suficiente tráfico puede ayudar a comparar el efecto agregado de dos alternativas. Responden preguntas distintas y pueden complementarse.

Fuentes para profundizar: [16]

20 / A/B TESTING · DESARROLLO

Cómo diseñar una comparación válida

Un experimento A/B busca estimar el efecto de un cambio de interfaz en una población, bajo condiciones controladas. Se diferencia de comparar los datos de dos semanas consecutivas.

Las pruebas con usuarios permiten descubrir dificultades y explicar hipótesis. Un A/B asigna aleatoriamente las unidades elegibles a dos variantes simultáneas, procurando que las diferencias sistemáticas entre grupos se deban al cambio evaluado. La asignación suele mantenerse estable para una persona o cuenta cuando puede volver al producto; elegir una unidad adecuada evita contaminación entre variantes.

Preparación del experimento

Primero se formula una hipótesis: «Mostrar el costo y saldo restante antes de confirmar reducirá abandonos por incertidumbre». Luego se define el indicador primario, la población y la unidad de asignación, las variantes, los criterios de inclusión y la ventana de observación. Se calculan los requisitos de muestra a partir de la tasa de base, el efecto mínimo que interesa detectar, el nivel de significación y la potencia, entre otros supuestos del diseño. También se establecen métricas de protección: por ejemplo, que no aumenten errores o solicitudes duplicadas.

A · Control
Revisión con costo del beneficio y botón de confirmación.

B · Variante
La misma revisión incorpora una vista explícita del saldo restante.

Durante el ensayo se comprueba que la asignación sea correcta, que las mediciones tengan las mismas definiciones y que no haya cambios externos que afecten de manera diferente a los grupos. No es conveniente terminar al observar una diferencia favorable sin una regla estadística válida de detención. Deben definirse previamente el análisis y la duración, teniendo en cuenta ciclos de actividad y tráfico.

Si se cambian simultáneamente el texto, el color, la posición y la lógica de la pantalla, un experimento puede estimar el efecto del conjunto, pero no aislar el efecto de cada modificación. Elegir un cambio relevante facilita interpretar la evidencia y reutilizar lo aprendido.

> El TP no exige implementar un experimento. El propósito es saber distinguir una hipótesis de una prueba cualitativa y de un diseño de evaluación cuantitativa controlada.

Referencias para profundizar: [16].

20 / A/B TESTING · APLICACIÓN

Interpretar diferencias: un ejemplo resuelto

En este ejemplo ficticio se asignan 1.000 personas elegibles a cada variante. Todas tienen la misma ventana de observación. El indicador primario es el porcentaje de personas asignadas que completan al menos un canje dentro de esa ventana. Se cuentan personas y no clics ni intentos múltiples.

Variante	Asignadas	Completaron	Tasa
A · Control	1.000	400	40 %
B · Variante	1.000	440	44 %

Diferencia absoluta: 44 % − 40 % = 4 puntos porcentuales

Cambio relativo: (44 − 40) / 40 × 100 = 10 %

El incremento observado es de 4 puntos porcentuales, o un 10 % relativo respecto de A. No son expresiones equivalentes: decir «4 % de mejora» sin especificar cuál puede confundir. El resultado por sí solo no demuestra una diferencia estadísticamente concluyente.

Como aproximación pedagógica, si las dos muestras son independientes y el diseño permite usar una comparación simple de proporciones, el error estándar estimado de la diferencia es √[0,40 × 0,60 / 1.000 + 0,44 × 0,56 / 1.000] ≈ 0,0221. Un intervalo de confianza aproximado del 95 % es 0,04 ± 1,96 × 0,0221, es decir, de −0,3 a +8,3 puntos porcentuales. Incluye cero: estos datos son compatibles con ausencia de mejora y también con mejoras relevantes.

El ejemplo no enseña una fórmula universal para experimentos; muestra por qué se debe evaluar incertidumbre además de observar tasas. Un análisis real debe respetar el diseño, la unidad de aleatorización y las reglas de análisis previstas. También se revisan métricas de protección, efectos prácticos, errores de instrumentación y posibles grupos desbalanceados.

Volver a la evidencia cualitativa

Si el TP evaluó dos participantes y ambos comprendieron mejor el saldo después de modificar el diseño, esa observación apoya una hipótesis exploratoria. No implica una mejora poblacional del 100 % ni autoriza a afirmar que B gana un A/B. El valor de las pruebas pequeñas es orientar decisiones y nuevas evaluaciones.

Referencias para profundizar: [16].

21 / INTEGRACIÓN DE LAS UNIDADES 4, 5 Y 6

Caso transversal: de hallazgo a implementación

Supongamos que, durante el test de la Unidad 5, dos personas no reconocieron que el canje había terminado y tocaron nuevamente «Confirmar».

Etapa	Decisión trazable
Evidencia · Unidad 5	Dos participantes intentaron confirmar más de una vez; la interfaz no mostraba resultado claro.
Diseño · Unidad 4	Se incorpora, tras el TP 5, una confirmación con comprobante y un estado de carga.
Especificación · Unidad 6	Definir transición listo → procesando → éxito / error y reglas de prevención de repetición.
Aceptación	Canje exitoso: se procesa una vez, se muestra comprobante y cambia saldo.
Accesibilidad	El estado se comunica también mediante texto y es reconocible por teclado/tecnologías de asistencia.
Medición	Emitir redemption_start y redemption_complete; registrar errores sin datos sensibles.
Verificación	Prueba de flujo, estados, teclado y revisión de eventos sin duplicados.

> Lo importante no es cuántas herramientas se usaron. Es poder seguir la cadena: **hallazgo → decisión → contrato → código → prueba → métrica**.

Documento de transferencia de una pantalla

- ✓ Captura/URL del prototipo y versión.
- ✓ Descripción de la operación y estado inicial.
- ✓ Estados, reglas y mensajes.
- ✓ Criterios de aceptación y alcance de pruebas.
- ✓ Notas de accesibilidad, performance y eventos relevantes.

21 / INTEGRACIÓN · DESARROLLO

Un hallazgo convertido en una decisión de ingeniería

El siguiente recorrido reúne los contenidos del módulo sobre un mismo problema. No constituye un nuevo requisito del práctico: es una demostración de cómo relacionar los conceptos.

En la prueba hipotética, dos participantes activaron dos veces «Confirmar» tras no reconocer ningún cambio después de la primera acción. El registro permite sostener que hubo repetición y ausencia de confirmación visible; aún no demuestra que se haya duplicado una transacción. La respuesta de diseño debe hacer observable el procesamiento y el resultado.

De la evidencia a la entrega

Decisión	Justificación y comprobación
Contrato del flujo	Al iniciar, se muestra progreso. Al confirmar el servidor, aparece un comprobante. Ante resultado incierto, consultar estado.
Componente	El control no permite iniciar otro intento lógico mientras está ocupado; conserva nombre accesible e indicador de foco.
Criterio de aceptación	Dado saldo suficiente, si se confirma una vez y se recibe éxito, se registra una sola operación y se muestra el saldo actualizado.
Accesibilidad	Progreso y resultado mediante texto; verificar foco y anuncios con teclado y tecnología de asistencia en la implementación.
SEO y performance	La confirmación personal requiere control de acceso. Reservar espacio para el contenido dinámico de la pantalla.
Medición	Inicio y éxito vinculados; error registrado cuando corresponda. Revisar finalización y recuperación por intento.

El equipo puede verificar por separado el componente, el manejo de estados y el comportamiento de la operación completa. El diseño puede representar carga y resultado; la prueba de integración comprobará que los datos del servidor, los mensajes y los eventos correspondan al mismo intento. Las pruebas con personas permitirán reevaluar la claridad de la experiencia.

La trazabilidad evita perder el motivo original de los cambios. Si en una futura versión se elimina la pantalla de confirmación, debería recordarse por qué existía y evaluar qué otra solución satisface esa misma necesidad. Documentar el fundamento, junto con el resultado de las pruebas, permite modificar decisiones sin repetir problemas conocidos.

22 / CIERRE APLICADO AL PROYECTO INTEGRADOR

Aplicación al TP Integrador N.º 6

El práctico vigente toma el **prototipo corregido en la Unidad 5** como punto de partida. La consigna solicita preparar una entrega técnica breve, no desarrollar un sistema nuevo.

Requisitos de la entrega vigente

Informe PDF de **2 a 3 carillas** y enlace al prototipo actualizado. Elegir un componente, revisar al menos dos aspectos de accesibilidad con evidencia y una mejora posible, justificar indexación y metadatos cuando correspondan e identificar una decisión de performance. Proponer **dos KPIs y tres eventos** con definición y disparador. Distinguir verificaciones realizadas de cuestiones pendientes en código. No se exige backend, publicación, instalación de analítica ni nuevas funcionalidades.

Actividad orientativa (laboratorio)

Elegir una pantalla crítica → **Especificar** estados + aceptación → **Auditar** semántica + WCAG → **Medir** KPI + eventos

Una ficha, no veinte páginas

Describir nombre del componente, propósito, props relevantes, estados, mensajes, accesibilidad y criterios de aceptación.

Una verificación técnica

Si hay HTML, navegar solo con teclado, revisar semántica, correr una herramienta automática y registrar qué no pudo verificar.

Checklist de transferencia

- ✓ La documentación explica qué problema se resuelve y quién lo usa.
- ✓ Los estados contemplan vacío, carga, error, éxito y recuperación.
- ✓ Los criterios de aceptación son observables, no «debe verse lindo».
- ✓ El HTML tiene etiquetas y jerarquía semánticas, y se puede recorrer con teclado.
- ✓ Las páginas públicas tienen título, enlaces y tratamiento SEO adecuados.
- ✓ Se distinguieron métricas de diagnóstico de KPIs del objetivo.
- ✓ La instrumentación propuesta evita datos personales innecesarios.

> En el examen final podrán defender **qué decisiones se implementaron, cómo se verificarían y qué medirían para saber si funcionaron**.

22 / AUTOEVALUACIÓN

Preguntas para revisar lo aprendido

1. ¿Puede una interfaz funcionar y no estar terminada?

Un canje devuelve el saldo correcto al usar mouse, pero un usuario de teclado no puede acceder al botón. El criterio funcional principal se cumple en un contexto; la DoD no se cumple si requiere operación con teclado. Se debe ampliar la verificación y registrar el defecto.

2. ¿Qué permite confirmar un prototipo sin código?

Puede identificar decisiones de diseño y verificar aspectos como contraste de los colores especificados. No permite confirmar la semántica del DOM, el orden de foco de la implementación final, la protección de una operación ni los datos efectivamente registrados.

3. ¿Cómo evitar que un saldo personal aparezca públicamente?

El acceso debe restringirse mediante autenticación y autorización. Un archivo robots.txt o una indicación noindex no son mecanismos de protección de información. En las rutas públicas, los mecanismos de rastreo e indexación se eligen según el objetivo del contenido.

4. ¿Qué métrica investigar cuando tarda el feedback?

Si una interacción demora en producir un cambio visible, INP puede ayudar a diagnosticar capacidad de respuesta. Si la confirmación del servidor llega tarde, también se debe investigar latencia de la operación y su estado de espera; no es lo mismo que medir INP.

5. ¿Por qué una tasa de finalización puede subir sin mejorar el diseño?

Puede cambiar la selección de usuarios o beneficios, la ventana de medición o el disparador de los eventos. Es necesario comprobar definiciones y calidad de los datos, junto con otras explicaciones posibles.

6. ¿Qué aporta un A/B frente a dos pruebas Think Aloud?

El A/B permite estimar diferencias agregadas bajo un diseño controlado y con suficiente evidencia. Think Aloud ayuda a descubrir y comprender dificultades. Ninguno reemplaza automáticamente al otro.

23 / BIBLIOGRAFÍA Y RECURSOS

Referencias y lecturas de profundización

Se priorizaron normas, guías oficiales y documentación de herramientas. Los enlaces se incluyen para que el estudiante pueda ampliar cada tema o revisar versiones posteriores.

Normas, semántica y accesibilidad

[1] W3C WAI · WCAG 2 Overview
Versiones, principios y alcance de WCAG 2.2.

[2] W3C WAI · What's New in WCAG 2.2
Nuevos criterios de foco, tamaño, interacción y autenticación.

[3] W3C WAI · Page Regions Tutorial
Landmarks, regiones de página y HTML5.

[4] W3C WAI · Understanding Contrast Minimum
Contraste de texto y excepciones.

[5] MDN · HTML: A good basis for accessibility
Estructura semántica y controles nativos.

[6] W3C WAI · Forms Tutorial
Etiquetas, instrucciones y errores en formularios.

[7] MDN · Progressive enhancement
Estrategia de mejora progresiva.

[8] W3C WAI · Evaluating Web Accessibility
Evaluación automática y manual.

[9] W3C WAI · Selecting Evaluation Tools
Alcances y límites de las herramientas.

SEO, métricas e ingeniería UI

[10] Google Search Central · SEO Starter Guide
Rastreo, estructura y prácticas básicas.

[11] Google Search Central · JavaScript SEO Basics
Procesamiento JS, rutas y contenido indexable.

[12] web.dev · Core Web Vitals
Definición de LCP, INP y CLS.

[13] Google Search Central · Core Web Vitals
Umbrales y relación con búsqueda.

[14] Google Analytics · Recommended events
Eventos y parámetros de GA4.

[15] Google Analytics · Recommended events by vertical
Eventos recomendados según tipo de producto.

[16] Nielsen Norman Group · A/B Testing 101
Hipótesis, muestra, duración e interpretación.

[17] Storybook · Automatic documentation
Documentación viva de componentes.

> Este material es **introductorio-aplicado**. Para verificar conformidad formal con estándares o requisitos legales específicos, corresponde consultar la norma completa y realizar una evaluación profesional del producto.

Consultas y revisión: noviembre de 2026. Las especificaciones y herramientas pueden evolucionar; se recomienda comprobar su versión vigente.

23 / BIBLIOGRAFÍA · COMPLEMENTO

Referencias normativas y metodológicas

Las siguientes fuentes complementan las referencias de la página anterior. Se recomienda consultar primero los documentos introductorios y luego las secciones específicas que correspondan a los requisitos evaluados.

[18] Schwaber, K. y Sutherland, J. (2020). The Scrum Guide.
Definición de terminado, calidad del incremento y transparencia.

[19] W3C. Web Content Accessibility Guidelines (WCAG) 2.2.
Texto normativo y criterios de conformidad.
Understanding Conformance · Understanding Target Size (Minimum).

[20] Google Search Central. In-Depth Guide to How Google Search Works.
Descubrimiento, rastreo, renderizado, indexación y presentación de resultados.

[21] Google Analytics. Set up event parameters.
Parámetros de eventos recomendados y personalizados.

Lectura y verificación

Los diagramas, fichas, operaciones y resultados numéricos de este apunte son ejemplos didácticos preparados para la cátedra. Los estándares y las métricas técnicas se remiten a sus fuentes oficiales; los umbrales y recomendaciones no deben confundirse con una garantía general de calidad o conformidad. Las herramientas pueden actualizar sus interfaces y funciones sin que cambien los objetivos metodológicos de la asignatura.

> **Idea de cierre:** la calidad de una interfaz no termina en el prototipo. Se expresa en el comportamiento implementado, el acceso efectivo de personas diversas y la capacidad de verificar y aprender de los resultados.