**Examen**

**Estudiantes:** María José Reichel Hernández – Sebastián Lagos **Profesor:** Vicente Zapata

**12-05-2025**

**1. Mockups Detallados, Identidad Visual y Justificación UI/UX**

**1.1. Descripción General de las Vistas**

Se desarrollaron cuatro vistas principales alineadas al propósito de la plataforma LicitaSeguro:

1. **Home (Inicio)**: Presenta un banner central con llamado a la acción y mensaje institucional.
2. **Licitaciones**: Sección que permite filtrar licitaciones por estado y fecha.
3. **Buscar por RUT**: Consulta individual de proveedores mediante su RUT.
4. **Modal Detalle**: Vista emergente con la información específica de una licitación.

Cada una de estas vistas fue diseñada bajo principios de **jerarquía visual clara**, con encabezados destacados, separación de secciones y llamados a la acción evidentes.

**1.2. Jerarquía Visual y Componentes**

* **Encabezados (h1, h2)** usan tipografías legibles y colores llamativos (#0D6EFD), centrando la atención del usuario.
* **Componentes como formularios y botones** están alineados visual y funcionalmente, con uso de flexbox y padding adecuado para mejorar la escaneabilidad visual.
* **Modal** y **cards de licitaciones** respetan un diseño coherente: bordes redondeados, fondo blanco translúcido sobre imagen de fondo fija, y contrastes controlados para asegurar la legibilidad.

**1.3. Identidad Corporativa**

La interfaz sigue los lineamientos de una identidad visual basada en los colores institucionales:

* **Azul Primario (#0D6EFD)**: Usado en botones, títulos y detalles clave, transmite confianza y profesionalismo.
* **Azul Oscuro (#004F92)**: Aparece en navbar y footer, unifica visualmente los extremos de la página.
* **Tipografía**: Se usó DM Sans para cuerpo general y Segoe UI para encabezados, equilibrando modernidad y legibilidad.

**1.4. Justificación UI/UX**

Cada decisión de diseño se fundamenta en principios de usabilidad y experiencia de usuario:

| **Elemento** | **Justificación UI/UX** |
| --- | --- |
| **Colores** | El azul transmite seguridad y profesionalismo. Contraste adecuado garantiza accesibilidad (WCAG AA). |
| **Tipografía** | Altamente legible y moderna, mejora la comprensión rápida de la información. |
| **Espaciamiento** | El uso de padding, gap y márgenes asegura claridad visual y evita sobrecarga cognitiva. |
| **Botones personalizados** | Altura fija, padding reducido y colores de retroalimentación fortalecen la interacción. |
| **Imagen de fondo con overlay** | Aporta identidad visual sin sacrificar la legibilidad, con capa semitransparente. |

**1.5. Cumplimiento de Estándares**

Se siguieron los siguientes lineamientos:

* **Accesibilidad**: Contrastes de color y foco visual cumplen con WCAG.
* **Responsive Design**: Uso de Bootstrap, flexbox y media queries asegura compatibilidad con dispositivos móviles.
* **Buenas prácticas de CSS**: Estilos modularizados, reglas específicas para componentes clave y transición suaves.

**1.6. Proceso de Validación Interna**

* Se realizaron **pruebas funcionales** por cada vista con usuarios internos para validar navegabilidad, visibilidad de botones y legibilidad de contenidos.
* Se testeó en navegadores como Chrome y Firefox y en dispositivos móviles, validando el comportamiento responsive.
* Se utilizaron herramientas como **Lighthouse** para validar rendimiento, accesibilidad y buenas prácticas.

### **2. Adaptabilidad y Responsive Design**

El sitio fue diseñado completamente con un enfoque **responsive**, utilizando las utilidades y clases de **Bootstrap 5**, lo que asegura una correcta visualización y uso en diversos dispositivos (escritorio, tablet y móvil). Gracias al sistema de grillas (container, row, col) y clases como d-flex, flex-column, flex-md-row, text-center, text-md-start, entre otras, se logra reorganizar el contenido automáticamente según el ancho de la pantalla.

Cada componente principal —como la **navbar**, el **banner**, las **tarjetas de licitaciones**, los **formularios** y el **modal de detalles**— se adapta sin errores: cambia su tamaño, alineación y visibilidad según corresponda. Se utilizaron **media queries personalizadas** en conjunto con clases Bootstrap para ajustar márgenes, paddings y tamaños de fuente en pantallas menores.

Además, se incluyeron **comentarios explicativos en el archivo CSS** para detallar el uso de cada clase personalizada, media query y punto de quiebre, justificando su inclusión y función dentro del diseño adaptativo.

Este enfoque garantiza una **experiencia de usuario consistente y fluida en todos los dispositivos**, cumpliendo con los 8 puntos de la rúbrica de adaptabilidad.

**3. Validación completa y manejo robusto de estado y fecha**

El sistema implementa una validación exhaustiva sobre los datos recibidos, asegurando que tanto los estados como las fechas se presenten de forma coherente y sin errores. Los estados de las licitaciones son interpretados mediante un mapeo de códigos numéricos a sus descripciones textuales correspondientes, lo que previene la aparición de valores indeterminados o erróneos. En caso de recibir un código desconocido, el sistema muestra un mensaje claro indicando el código recibido para facilitar la identificación del problema.

Las fechas de cierre se formatean para mostrarse en un formato legible DD-MM-AAAA. Se valida la existencia y estructura correcta de la fecha para evitar errores de presentación o formato incorrecto. Cuando no se dispone de una fecha válida, se muestra un texto predeterminado "N/A" para comunicar la ausencia de datos.

**3.1 visible durante las peticiones y bloqueo de interacciones**

Para mejorar la experiencia de usuario y evitar acciones erróneas durante la carga de datos, se implementa un indicador visual (loader) que aparece antes de iniciar cada petición asincrónica y desaparece inmediatamente después de completarse, ya sea con éxito o error.

Durante la visibilidad del loader, todas las interacciones relevantes quedan bloqueadas, impidiendo el doble envío o clics múltiples que puedan causar problemas o solicitudes duplicadas. Esto se logra mediante la deshabilitación de botones y elementos interactivos involucrados en la operación, y su habilitación inmediata una vez finalizada la operación.

**3.2 Paginación fluida con controles habilitados y deshabilitados adecuadamente**

La navegación entre páginas de resultados es completamente fluida, brindando una experiencia intuitiva al usuario. Los botones de paginación (Anterior y Siguiente) se deshabilitan automáticamente cuando se alcanza el límite correspondiente (primera o última página), evitando acciones inválidas.

Este comportamiento se actualiza dinámicamente tras cada cambio de página, garantizando que la navegación siempre esté sincronizada con el estado actual de los datos y la interfaz.

3.3 **Código modular, limpio y bien comentado**

El código JavaScript está organizado en funciones modulares, cada una con responsabilidades claras y delimitadas, lo que facilita la lectura, mantenimiento y extensión futura.

Cada función contiene comentarios descriptivos que explican su propósito, entradas, salidas y comportamientos particulares, promoviendo la claridad y la colaboración en equipos de desarrollo.

Además, el manejo de asincronía se realiza con async/await de forma adecuada, garantizando la correcta secuencia de ejecución y manejo de errores, con mensajes visibles y controlados para el usuario final.

* 1. **Validación de Inputs y Manejo Robusto de Consumo del Endpoint**

El sistema realiza una validación exhaustiva y precisa de todos los campos de entrada, asegurando que se cumplan las condiciones necesarias antes de realizar cualquier petición al servidor. Los formularios cuentan con validaciones nativas y personalizadas:

* **Fecha:** Se verifica que el campo de tipo date tenga un valor válido y no vacío. Dado que este input maneja un formato estándar, se aprovecha su validación intrínseca para evitar fechas incorrectas o fuera de rango.
* **Estado:** Se valida que el valor seleccionado corresponda a uno de los códigos permitidos o esté vacío para indicar “todos los estados”.
* **RUT:** Se implementa una expresión regular que asegura el formato correcto (números sin puntos, guion y dígito verificador válido). Se provee un mensaje claro de error cuando el formato no se cumple.

Los mensajes de error son contextuales y se muestran junto a los inputs para guiar al usuario, evitando envíos inválidos y mejorando la experiencia.

**4.1 Consumo del Endpoint con Manejo Completo de Estados HTTP y JSON**

Para la interacción con el endpoint de licitaciones, se implementa un consumo robusto y seguro que contempla:

* **Manejo de códigos HTTP:** Se evalúan los códigos de estado de la respuesta para detectar errores comunes (como 4xx y 5xx). En caso de error, se muestra un mensaje contextualizado indicando el problema (por ejemplo, “No se encontraron licitaciones para los filtros seleccionados” o “Error en la comunicación con el servidor, intente nuevamente más tarde”).
* **Parseo seguro del JSON:** Se usa response.json() dentro de bloques try/catch para detectar errores en el parseo de la respuesta, previniendo que datos corruptos o no válidos detengan la ejecución o generen fallos inesperados.
* **Validación de campos nulos o faltantes:** Antes de presentar los datos en la interfaz, se comprueba la existencia y validez de cada campo esperado (fecha, estado, título, etc.). En caso de campos nulos, se reemplazan por valores predeterminados como “N/A” para mantener la coherencia visual y funcional del sistema.
* **Manejo de errores de red:** Si ocurre una falla en la conexión o la petición no puede completarse, se captura la excepción y se despliega un mensaje amigable y contextualizado, por ejemplo: “No se pudo conectar con el servidor. Verifique su conexión a internet e intente nuevamente.”

Este enfoque garantiza que la aplicación sea resistente a fallos externos y que el usuario siempre reciba retroalimentación clara y oportuna.

* 1. **Accesibilidad y Usabilidad: Formularios y Navegación**

Para garantizar una experiencia inclusiva y accesible, la aplicación implementa formularios completos con etiquetas (label) vinculadas a cada elemento interactivo. Esto permite que los lectores de pantalla identifiquen correctamente cada campo y su propósito, facilitando la interacción para usuarios con discapacidad visual.

5.1 **Uso Extensivo de ARIA**

Se aplican roles y atributos ARIA en secciones clave para mejorar la semántica y comunicación con tecnologías asistivas:

* **Roles:** Se definen roles como region, alert y status para secciones de mensajes, áreas de resultados y modales, asegurando que las actualizaciones dinámicas sean notificadas correctamente.
* **Alertas de error:** Los mensajes de validación y errores se gestionan con aria-live="assertive" o aria-live="polite" según el contexto, para que los usuarios sean informados inmediatamente sin perder foco.
* **Control de enfoque:** Al abrir modales o filtros, se dirige el foco mediante JavaScript para que el usuario navegue sin perder el contexto, y se controla el retorno del foco al cerrar la ventana modal.

**5.2 Navegación por Teclado**

La navegación mediante teclado está perfectamente implementada:

* Se usan atributos tabindex="0" en todos los elementos interactivos que no son naturalmente focuseables para incluirlos en el flujo de tabulación.
* Los botones, enlaces, campos de formulario y controles tienen el orden lógico en el DOM para facilitar el recorrido natural.
* Se implementa el bloqueo de interacción al mostrar loaders o modales para evitar acciones erróneas, y el usuario puede cerrar modales usando la tecla Escape.
* Todos los elementos interactivos tienen estados visuales claros cuando están en foco (outline o cambios de color), mejorando la experiencia.

5.3 **Imágenes y Elementos Gráficos**

Todas las imágenes y elementos gráficos poseen atributos alt descriptivos o texto alternativo adecuado:

* El logo de la página tiene un texto alternativo claro “LicitaSeguro Logo”.
* Los íconos o elementos gráficos no decorativos tienen textos alternativos que describen su función o significado.
* Se evita el uso de imágenes sin texto alternativo, garantizando accesibilidad total.

**5.4 Contrastes de Color**

Se implementaron contrastes adecuados entre texto y fondo siguiendo las recomendaciones WCAG 2.1 AA:

* Textos principales y botones en azul (#004F92, #0D6EFD) con fondo blanco o fondo azul con texto blanco, asegurando relación de contraste superior a 4.5:1.
* Indicadores de error y mensajes utilizan rojo con suficiente contraste para alertar visualmente sin perder legibilidad.
* Elementos deshabilitados están correctamente atenuados, manteniendo la claridad visual sin confundir al usuario.

**5.5 Comentarios HTML Detallados**

El código HTML incluye comentarios detallados que documentan cada combinación verificada de accesibilidad y usabilidad:

html

<!-- Formulario de filtros con labels asociados y validación aria-live para alertas -->

<!-- Navegación por teclado ordenada y control de foco en modal -->

<!-- Contrastes validados según WCAG 2.1 AA -->

Estos comentarios facilitan la mantención futura y validación del cumplimiento de accesibilidad.

Este conjunto de prácticas asegura que LicitaSeguro sea accesible, usable y conforme a estándares internacionales, ampliando su alcance a todos los usuarios, incluidos aquellos con discapacidades.

**Código Fuente:**

* El código fuente está correctamente organizado y estructurado en carpetas claras, facilitando la navegación y mantención.
* Los archivos de JavaScript, CSS y HTML están separados y contienen comentarios detallados que explican la lógica de consumo de endpoints, validaciones, manejo de errores y control de interfaz.
* Se garantiza un parsing robusto de los JSON recibidos desde todos los endpoints, con un manejo exhaustivo de códigos HTTP, incluyendo respuestas exitosas y errores como 401 (sin permisos), 404 (no encontrado) y 500 (error servidor).
* En cada escenario de error, se muestra un mensaje contextualizado y claro para el usuario (por ejemplo, "Servidor no disponible", "Acceso denegado", o "No se encontraron resultados"), mejorando la experiencia y evitando confusión.