UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

DEPARTAMENTO DE ELECTROTECNIA E INFORMÁTICA
INGENIERÍA EN INFORMÁTICA

# Implementación de Front-End responsivo para Amukan: Plataforma de Planificación Turística

Ashley Katrina Iturriaga Bastías

ashley.iturriaga@usm.cl

Pamela Gatica
Profesora Guía

Franco Jorquera
Profesor Correferente

**Resumen:** Amukan aborda la problemática de la fragmentación de la información turística, que dificulta la planificación de viajes al requerir la consulta de múltiples plataformas. El presente trabajo tuvo como objetivo desarrollar el frontend responsivo de Amukan, una plataforma web orientada a centralizar la oferta turística de Viña del Mar y facilitar el acceso a información sobre servicios, actividades y eventos. La implementación se realizó utilizando *React, Vite y Tailwind CSS,* mediante una metodología ágil basada en *Scrum* y organizada en tres *sprints*. Se desarrollaron módulos de exploración y búsqueda, generación de itinerarios personalizados con exportación a PDF, gestión de servicios para negocios locales y visualización de indicadores institucionales. La validación contempló auditorías técnicas mediante *Google Lighthouse* en las vistas mapa, dashboard, mi comercio y registro, considerando rendimiento, accesibilidad, buenas prácticas y *SEO*. Se incorporaron optimizaciones como carga diferida (*lazy loading*) y utilización de imágenes en formato *WebP*. Los resultados permitieron identificar fortalezas y oportunidades de mejora en el comportamiento técnico del *frontend*. El proyecto proporciona una base funcional y modular para integrar información turística en una misma plataforma, con posibilidades de optimización y evolución futura hacia una *Progressive Web App* (*PWA*).

**Palabras Clave:** frontend, responsividad, turismo, google lighthouse, react

## 1 Introducción

### 1.1 Contexto, motivación y problemática

El sector turístico de Viña del Mar enfrenta una serie de problemáticas relacionadas con la dispersión de la información y falta de integración tecnológica en la planificación de viajes o actividades. Actualmente, los turistas deben recurrir a múltiples plataformas de turismo para organizar, hospedaje, alimentación y entretenimiento, lo que genera una experiencia poco práctica, compleja y fragmentada. Estudios de mercado recientes indican que, en promedio, un viajero destina el equivalente a dos días laborales (aproximadamente 16 horas) a la planificación y reserva de un viaje [1]. Este volumen de tiempo invertido refleja lo engorroso del proceso.

# CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

**1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO**

**Tipo de monografía (marcar una opción):** ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

**Título del trabajo:** Implementación de Front-End responsivo para Amukan: Plataforma de Planificación Turística

**Nombre del candidato(a):** Ashley Katrina Iturriaga Bastías

**Carrera / Grado:** Ingeniería en Informática

**Campus:** Viña del Mar **Departamento:** Electrotecnia e Informática

**2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS**

Yo, Pamela Alexandra Gatica Caballero, en mi calidad de profesora guía del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

**3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)**

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por **(marcar una opción)**:

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

**Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):**

_____

_____

_____

**4.- FIRMAS**

**Profesor(a) guía o director(a) de memoria o tesis:**

**Fecha:** 28 de octubre de 2026 **Firma:** _____

**Estudiante o Candidato(a):**

**Fecha:** 28 de octubre 2026 **Firma:** _____

*Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.*

Por su parte, los negocios locales carecen de visibilidad digital y dependen de métodos de difusión tradicionales de bajo alcance, lo que restringe su competitividad en temporada baja o en zonas alejadas del centro. Este déficit tecnológico se enmarca en las brechas identificadas a nivel comunal en el sector turístico de Viña del Mar, asociadas a la necesidad de mayor acompañamiento técnico, digitalización de las *MIPYMES* y fortalecimiento de estrategias públicas de adopción tecnológica [2]. Las instituciones, en tanto, enfrentan sobrecarga operativa y ausencia de automatización.

A estas problemáticas se suma la percepción ciudadana recogida en el informe *PLADETUR* [2], que evidencia que el turismo en Viña del Mar se encuentra excesivamente centralizado en el borde costero y en actividades de verano, generando una oferta poco diversificada. La comunidad expresa el anhelo de ampliar el turismo hacia otras temporadas y territorios, incorporando espacios culturales y naturales de la comuna, lo que pone de relieve la necesidad de una estrategia más diversificada y sostenible, fortaleciendo el vínculo entre comunidad y territorio.

Con este contexto, la motivación del proyecto se centra en el desarrollo de una plataforma web responsiva que facilite el acceso a información sobre actividades, eventos y alimentación a los usuarios, con un frontend adaptable a dispositivos móviles, considerando que los turistas y residentes toman decisiones "en movimiento" y en tiempo real. Un diseño centrado en usabilidad y adaptabilidad permite ofrecer una experiencia fluida de planificación, integrar servicios dispersos y fortalecer la interacción entre visitantes, negocios, residentes e instituciones locales.

Aunque la implementación inicial se proyecta en Viña del Mar, la propuesta está pensada para ser adaptable a otras ciudades, municipios y destinos turísticos.

## 1.2 Definición del problema

La problemática central que este proyecto identifica es la fragmentación de la información turística de Viña del Mar y las dificultades que enfrentan los distintos actores involucrados en la actividad turística. Esta situación se refleja en el tiempo y esfuerzo que deben dedicar los usuarios para localizar, comparar y organizar información proveniente de distintas fuentes. La problemática se divide en cuatro puntos críticos:

a) En el caso de los turistas, la interacción con los servicios disponibles se caracteriza por una alta carga cognitiva y un bajo nivel de personalización, lo que los obliga a emplear múltiples plataformas y herramientas no integradas para completar sus tareas. Esta situación se encuentra respaldada por los resultados del estudio realizado por *Priceline* [1].
b) Para los negocios locales y pymes, la marcada estacionalidad del destino genera baja visibilidad y menor flujo de clientes fuera de la temporada alta, limitando su alcance y competitividad durante gran parte del año. El *PLADETUR* reconoce esta concentración de la actividad en verano (con saturación en sectores clave) y plantea como desafío la desestacionalización y el desarrollo de una oferta turística diversificada durante todo el año [2].
c) En las instituciones, la falta de planificación estratégica y de sistemas de información genera una gestión saturada en temporadas altas, y una carencia de datos que dificulta la toma de decisiones. El *PLADETUR* reconoce que *Viña del Mar* se encuentra en una fase de estancamiento del ciclo de vida turístico, con riesgo de declive [2].

d) Para los residentes, particularmente aquellos de sectores periféricos, se genera una menor integración con la oferta y los beneficios del turismo local. Según el *PLADETUR* la actividad turística y la distribución de sus beneficios no son homogéneas: se concentran en los sectores Plan y Reñaca, mientras que el resto de la comuna no absorbe esta dinámica, limitando las oportunidades de participación e inclusión territorial [2].

Todo esto provoca un ecosistema fragmentado que no aprovecha óptimamente las oportunidades de la digitalización, afectando la competitividad del destino.

## 1.3 Propuesta de solución

La solución propuesta consiste en un sitio web responsivo, orientado a distintos tipos de usuario: turistas, residentes, negocios locales, instituciones y organizaciones del ecosistema turístico. El objetivo es integrar información actualmente dispersa y optimizar la experiencia de planificación y gestión turística mediante herramientas digitales. Para ello, la plataforma contempla las siguientes funcionalidades principales.

- **Planificación turística**: permite a turistas y residentes explorar la oferta disponible y generar itinerarios de acuerdo con sus preferencias y disponibilidad.
- **Búsqueda y exploración**: centraliza información sobre actividades, eventos, gastronomía, alojamiento e incorporando filtros para facilitar su consulta.
- **Negocios locales:** permite a los negocios gestionar y difundir información sobre sus servicios, actividades y promociones, favoreciendo su exposición ante turistas y residentes.
- **Gestión institucional:** incorpora un panel orientado a instituciones y organizaciones para administrar información turística y consultar datos relacionados con la actividad del destino.
- **Responsividad**: el diseño responsivo permite acceder a las funcionalidades desde dispositivos móviles, tablets y computadores.

La elección de un enfoque responsivo responde a un requerimiento técnico asociado al contexto turístico, donde los usuarios puedan acceder a la plataforma desde distintos dispositivos mientras se desplazan por la ciudad. En este contexto, el aporte de este trabajo se concentra en el desarrollo del frontend, implementando React y Tailwind CSS, incluyendo la construcción de sus vistas, componentes, navegación y adaptación a dispositivos móviles, tablets y computadores. El alcance se limita a la capa de presentación e interacción del sistema, por lo que no contempla el desarrollo de backend, implementación de analítica ni un estudio específico sobre *UX/UI*.

## 1.4 Objetivo general y específicos

**Objetivo general:**
Desarrollar el *frontend* responsivo de Amukan, una plataforma web orientada a centralizar la información turística de Viña del Mar, incorporando funcionalidades de exploración, planificación personalizada de itinerarios y gestión para negocios locales e instituciones.

**Objetivos específicos:**

- ***OE1***: Diseñar la arquitectura de componentes del *frontend* y su integración con la *API* del sistema, definiendo la estructura de módulos, el manejo de estado y el control de acceso por rol.

- ***OE2***: Implementar los módulos de exploración y planificación turística, incorporando interfaces para la búsqueda y visualización de servicios, eventos y actividades, así como la presentación de itinerarios personalizados y su exportación a formato PDF.

- ***OE3***: Implementar el módulo de gestión para negocios locales, que permita publicar y editar servicios.

- ***OE4***: Implementar el panel de indicadores para instituciones y organizaciones, integrando en la interfaz las visualizaciones de la actividad turística registrada en la plataforma.

- ***OE5***: Validar la responsividad, el rendimiento y la accesibilidad de las vistas principales mediante pruebas en distintos anchos de pantalla y auditorías automatizadas sobre una compilación optimizada del *frontend*.

El cumplimiento del objetivo general se comprobará a través de la realización de sus objetivos específicos, abarcando dos dimensiones verificables:

- Verificación de la implementación del frontend: Se evaluará mediante la revisión de los componentes desarrollados, los fragmentos de código y evidencias visuales. Este procedimiento permitirá comprobar la incorporación de las funcionalidades contempladas en los objetivos específicos **OE1** al **OE4**, considerando los módulos de exploración, planificación de itinerarios, gestión comercial y panel institucional.

- Validación Técnica (Responsividad y Rendimiento): Se validará mediante auditorías automatizadas utilizando *Google Lighthouse* (midiendo *Performance, Accessibility, Best Practices y SEO*) y comprobaciones prácticas de adaptabilidad en pantalla móvil, tablet y computador, con el propósito de evaluar el cumplimiento del **OE5***.*

## 1.5 Metodologías y plan de trabajo

Se adoptó la metodología ágil *Scrum* para organizar el desarrollo de Amukan, distribuyendo el trabajo en tres sprints orientados a la construcción progresiva de un Producto Mínimo Viable (*MVP*). El equipo contempló los roles de *Scrum Master*, *Product Owner* y responsables de las áreas de *UX/UI*, *frontend y backend*. En este contexto, la autora del presente trabajo asumió la responsabilidad del desarrollo frontend de la plataforma.

Para la organización y seguimiento de las actividades se utilizó *GitHub Projects,* mientras que *Milanote* permitió registrar errores menores, ajustes y mejoras pendientes. Asimismo, se realizaron reuniones periódicas de coordinación para revisar los avances y las dificultades del desarrollo.

El plan de trabajo se organizó en tres *Sprints* principales, orientados a alcanzar y pulir progresivamente un Producto Mínimo Viable (*MVP*):

- **Sprint 1**: Esta iteración inicial se centró en el levantamiento de requerimientos y la estructuración del *MVP* funcional básico. En esta etapa, el diseño de la interfaz y la experiencia de usuario (*UX*/*UI*) fue concebido y maquetado en Figma mediante un trabajo inicialmente colaborativo entre todo el equipo. Esta colaboración temprana permitió establecer las bases de la propuesta visual del sistema, tras lo cual el diseño y sus iteraciones continuaron bajo la responsabilidad exclusiva del rol de *UX*/*UI*. Al cierre del ciclo, el equipo generó una bitácora de feedback y un plan de mejoras.

- **Sprint 2**: Esta iteración se centró en el desarrollo de las funcionalidades principales de la plataforma. Se utilizó *Milanote* como herramienta complementaria a *GitHub Projects* para registrar errores menores, mejoras y tareas específicas, clasificándolas según su estado de avance. Desde la perspectiva del desarrollo frontend, los diseños fueron transformados en interfaces interactivas mediante *React, Vite* y *Tailwind CSS*, utilizando componentes modulares y estilos responsivos. Esta iteración culminó con una demostración funcional del *MVP* actualizado y la identificación de brechas pendientes para la entrega final.

- **Sprint 3:** La iteración final se orientó a la corrección de errores, la validación técnica y la consolidación del producto definitivo. Para el área del *frontend*, esta fase contempló dos procesos de validación: por un lado, la evaluación del rendimiento, accesibilidad, buenas prácticas y *SEO* de las vistas principales utilizando la herramienta automatizada *Google Lighthouse*; por otro lado, la verificación práctica de la responsividad y adaptabilidad visual mediante el uso de herramientas de desarrollo del navegador (*DevTools*) simulando distintos anchos de pantalla (móvil, tablet y computador). Finalmente, el equipo desarrolló la demostración del *MVP* final, acompañada de la preparación del material de exhibición visual y el *pitch* del proyecto.

## 1.6 Estructura del informe

Con el fin de facilitar la comprensión del desarrollo del proyecto, este informe se organiza en capítulos que abordan los elementos, metodologías y tecnologías utilizadas. A continuación, se presenta una síntesis de la estructura general del documento:

- **Capítulo 2: Marco Conceptual**. En este capítulo se reúnen los conceptos técnicos necesarios para comprender el funcionamiento del sistema, incluyendo fundamentos de *frontend*, tecnologías utilizadas y futuras expansiones.

- **Capítulo 3: Diseño y Arquitectura.** Se describen los Diagramas UML que moldean la jerarquía de actores, el contenido turístico, la planificación de itinerarios y la estructura de la interfaz detallando la arquitectura modular.

- **Capítulo 4: Desarrollo e Implementación del Frontend**. Este capítulo describe cómo se implementaron los Objetivos Específicos. Se detalla la lógica aplicada en módulos claves.

- **Capítulo 5: Pruebas y Validaciones.** Se presenta la metodología y los resultados de las evaluaciones técnicas del *frontend*, mediante pruebas de responsividad y auditorías automatizadas con *Google Lighthouse*. Asimismo, se analiza el cumplimiento de los objetivos específicos utilizando las evidencias de implementación y los resultados obtenidos.

- **Capítulo 6: Conclusiones y Trabajo Futuro**. Este capítulo final sintetiza el cumplimiento de los objetivos y evalúa la solución implementada. Se proponen las mejoras y expansión del proyecto a futuro.

# 2 Marco Conceptual

En este capítulo se presentan los fundamentos conceptuales y tecnológicos relacionados con el desarrollo frontend de aplicaciones web. Se abordan los principios de construcción y presentación de interfaces, el diseño responsivo y los criterios técnicos asociados con el rendimiento y la accesibilidad. Asimismo, se examinan distintas tecnologías y herramientas utilizadas en el desarrollo frontend, considerando sus características, ventajas y limitaciones. Estos antecedentes permiten establecer una base conceptual para comprender las decisiones tecnológicas y los criterios de evaluación abordados en este trabajo.

## 2.1 Introducción al frontend

El frontend corresponde a la parte de una aplicación web encargada de presentar la información y permitir la interacción directa con el usuario. Comprende los elementos de la interfaz, como botones, formularios, menús de navegación, imágenes y estilos visuales, mediante los cuales se accede a las funcionalidades del sistema. Su implementación se apoya principalmente en tecnologías como *HTML, CSS y JavaScript*, que permiten estructurar el contenido, definir su presentación y establecer comportamientos interactivos, respectivamente [3].

El frontend forma parte de la arquitectura de una aplicación web y se diferencia del backend, que se encarga del procesamiento y de la lógica que se ejecuta del lado del servidor. La interacción entre ambas partes permite presentar información y ofrecer funcionalidades a través de una interfaz. Además, el desarrollo frontend contempla aspectos técnicos como la organización de los componentes, la adaptación a distintos tamaños de pantalla y la presentación adecuada de los contenidos en diferentes dispositivos [3][4].

### 2.1.1 Diseño Responsivo

El diseño responsivo es un enfoque de desarrollo web que permite adaptar la presentación y distribución de los contenidos según las características del dispositivo y el espacio disponible en pantalla. Para ello, se emplean técnicas como cuadrículas flexibles, imágenes adaptables y consultas de medios (*media queries*), que permiten modificar la disposición de los elementos sin necesidad de desarrollar una versión independiente para cada dispositivo [4]. De esta manera, una misma aplicación puede mantener el acceso a sus contenidos y funcionalidades desde computadores y dispositivos móviles. Además de la adaptación visual, es importante que la interfaz

presente información comprensible y facilite la interacción de los usuarios, aspectos relacionados con los principios básicos de experiencia de usuario [5].

La importancia de este enfoque se relaciona con la diversidad de dispositivos utilizados para acceder a la web. Según las estadísticas de *StatCounter Global Stats* correspondientes a septiembre de 2026, la participación mundial del tráfico web se distribuyó entre dispositivos móviles (58,99 %), computadores de escritorio (39,26 %) y tablets (1,76 %) [6]. Estas cifras evidencian la relevancia de considerar diferentes tamaños de pantalla durante el desarrollo de aplicaciones web, aunque no representan necesariamente los patrones de uso de todos los contextos o grupos de usuarios.

La adaptación de una interfaz no depende únicamente de la distribución visual de sus elementos, sino también de la posibilidad de acceder a sus controles y contenidos en distintas condiciones de visualización. En este sentido, el diseño responsivo debe considerar la reorganización de los componentes, la legibilidad del contenido y la disponibilidad de los elementos interactivos en diferentes tamaños de pantalla [4]. Estos criterios permiten establecer una base técnica para evaluar si una interfaz conserva su funcionalidad al visualizarse desde distintos dispositivos.

## 2.2 Tecnologías principales para frontend

El desarrollo de plataformas web se sustenta en tres tecnologías base: *HTML5,* encargado de la estructura semántica del contenido; *CSS*, orientado a los estilos, la presentación visual y la adaptación responsiva; y *JavaScript*, que se encarga de la interactividad y el dinamismo [7]. Estas herramientas constituyen la base sólida de toda aplicación web. No obstante, al abordar proyectos de mayor calibre, su uso tiende a generar dificultades en la escalabilidad, el mantenimiento del código y la productividad en el trabajo colaborativo.

En este contexto, han surgido diversos frameworks y librerías que optimizan el flujo de trabajo y aportan soluciones a problemáticas comunes del desarrollo frontend. Entre las tecnologías analizadas destacan *React, Tailwind CSS* y *Vite*, las cuales presentan un conjunto de ventajas significativas en comparación con el uso aislado de *HTML, CSS y JavaScript.*

*React* es una biblioteca de *JavaScript* orientada a la construcción de interfaces de usuario mediante componentes reutilizables [8]. *Vite* es una herramienta de desarrollo y construcción (*build tool*) que ofrece un servidor de desarrollo ultrarrápido y *builds* optimizados [9]. *Tailwind CSS* es un *framework* de *CSS* utilitario (*utility-first*) que permite definir estilos de forma rápida, consistente y adaptable a diferentes tamaños de pantalla [11].

| Tecnología | Función principal | Ventajas Principales | Limitaciones |
|---|---|---|---|
| *React* | Construir interfaces de usuarios mediante componentes. | Reutilización de componentes, estructura modular y amplio ecosistema. | Requiere herramientas complementarias para cubrir necesidades como el enrutamiento. |
| Vue | Construir interfaces mediante un enfoque declarativo y basado en componentes. | Adopción progresiva y estructura de componentes. | Requiere definir e integrar herramientas complementarias según las necesidades del proyecto. |
| Angular | Desarrollar aplicaciones web mediante un framework con estructura y herramientas integradas. | Convenciones definidas y conjunto amplio de funcionalidades. | Su ecosistema y estructura pueden requerir una curva de aprendizaje mayor. |
| Next.js | *Framework* basado en *React* para desarrollar aplicaciones web. | Enrutamiento integrado, opciones de renderizado y optimización de recursos. | Mayor complejidad de configuración y convenciones que pueden resultar innecesarias en proyectos frontend sencillos. |
| Vite | Proporcionar un entorno de desarrollo y herramientas para construir aplicaciones web. | Servidor de desarrollo rápido, actualización en caliente y construcción optimizada. | No reemplaza una biblioteca de interfaz ni define por sí mismo la arquitectura de los componentes. |
| Tailwind CSS | Definir la presentación visual mediante clases *CSS* de utilidad. | Personalización de estilos, consistencia y adaptación responsiva. | Requiere familiarizarse con sus clases y puede generar marcado extenso. |
| Bootstrap | Construir interfaces mediante componentes, estilos y utilidades *CSS* | Componentes listos para utilizar y recursos para diseños responsivos. | La personalización puede requerir ajustes para alcanzar una apariencia específica. |

| | | | |
|---|---|---|---|
| | predefinidos. | | |

**Tabla 1.** Tabla comparativa de tecnologías
Fuente: elaboración propia

La combinación de *React*, *Vite* y *Tailwind CSS* permite alcanzar un equilibrio entre eficiencia, rendimiento y escalabilidad. *React* facilita la creación de interfaces mediante componentes reutilizables; *Vite* agiliza el desarrollo gracias a su servidor de compilación ultrarrápido y proceso de compilación optimizado; *Tailwind CSS* proporciona un diseño moderno, coherente y adaptable sin necesidad de escribir hojas de estilo extensas.

### 2.2.1 Progressive Web Apps (PWAs)

Las *Progressive Web Apps (PWAs)* son aplicaciones web que utilizan tecnologías modernas para ofrecer una experiencia similar a la de una aplicación nativa, combinando lo mejor de la web y de las aplicaciones de plataforma. Funcionan principalmente a través de un *service worker* (*scripts* que se ejecutan en segundo plano) y un archivo de manifiesto web, lo que permite funcionalidades avanzadas como el acceso sin conexión, notificaciones push y la instalación directa en el dispositivo del usuario desde el navegador [12].

Entre sus principales ventajas se destacan:

- Funcionamiento offline o con conectividad limitada, gracias al almacenamiento en caché de recursos.
- Posibilidad de instalarse como una aplicación nativa sin necesidad de pasar por tiendas de aplicaciones.
- Envío de notificaciones *push* para mantener la interacción con los usuarios.
- Mejor rendimiento, con tiempos de carga más rápidos y una navegación más fluida.

Las *Progressive Web Apps* pueden integrarse de forma natural con tecnologías como *React, Vite y Tailwind CSS*. *React* proporciona una arquitectura de componentes reutilizables ideal para construir interfaces robustas y mantenibles. *Vite*, mediante el plugin oficial *vite-plugin-pwa*, facilita significativamente la implementación de *service workers* y el archivo de manifiesto web, simplificando la conversión de una aplicación en *PWA* [10]. Por su parte, *Tailwind CSS* asegura que el diseño mantenga consistencia visual y adaptabilidad total en todos los dispositivos, aspecto fundamental para cumplir con los criterios de calidad que definen a una *PWA* moderna [12].

## 2.3 Herramientas de Análisis y Optimización del Frontend

Además de las tecnologías de desarrollo, existen herramientas que permiten evaluar la calidad, el rendimiento y la accesibilidad de una aplicación web. Estas plataformas entregan métricas cuantificables que facilitan identificar oportunidades de mejora en la carga, el uso de recursos y la experiencia de uso.

### 2.3.1 Google Lighthouse

*Google Lighthouse* es una herramienta automatizada de auditoría de páginas web, integrada en *Chrome DevTools* y también disponible mediante línea de comandos y *PageSpeed Insights.* Evalúa cuatro categorías con una puntuación de 0 a 100: *Performance* (rendimiento), *Accessibility* (accesibilidad), *Best Practices* (buenas prácticas) y *SEO* (optimización para motores de búsqueda) [13]. Su uso no se limita al entorno profesional: investigaciones de ingeniería de software la emplean como base de medición en estudios de caso industriales de optimización del rendimiento web [14].

Cada categoría resume un conjunto de auditorías técnicas. A continuación se describen las cuatro.

- **Performance (Rendimiento)**

Mide qué tan rápido se carga la página y en qué momento se vuelve usable. Lighthouse calcula esta puntuación como un promedio ponderado de cinco métricas de laboratorio [15].

| Métrica | Sigla | Qué mide | Interpretación |
|---|---|---|---|
| First Contentful Paint | FCP | Tiempo hasta que aparece el primer elemento visible en pantalla | En móvil, cerca de 1,8 s o menos se asocia a una puntuación alta [15] |
| Largest Contentful Paint | LCP | Tiempo en que carga el elemento más grande de la vista principal | ≤ 2,5 s se considera bueno [16] |
| Speed Index | SI | Velocidad con la que el contenido se vuelve visualmente completo | En móvil, 0–3,4 s se codifica como rápido [17] |
| Total Blocking Time | TBT | Tiempo en que el hilo principal queda bloqueado impidiendo interacción | ≤ 200 ms se asocia a una puntuación alta en móvil [18] |
| Cumulative Layout Shift | CLS | Estabilidad del diseño durante la carga | ≤ 0,1 se considera bueno [19] |

**Tabla 2.** Tabla explicativa de *Performance*
Fuente: *Documentación de Lighthouse*

Estas métricas permiten identificar elementos que retrasan o bloquean la interacción del usuario, como imágenes pesadas, *JavaScript* o recursos sin optimización.

- **Accessibility (Accesibilidad)**

Comprueba, de forma automática, si la página presenta barreras para personas con discapacidad visual, motriz o cognitiva. La detección automática cubre solo un subconjunto de los problemas; no garantiza por sí sola la accesibilidad de la aplicación [13]. Entre los criterios que revisa se incluyen:

- Contraste de color suficiente.
- Imágenes con texto alternativo (*alt*).
- Navegación mediante teclado.
- Uso válido de atributos *ARIA*.
- etiquetas asociadas a campos de formulario.

- **Best Practices (Buenas prácticas)**

Analiza el cumplimiento de normas técnicas relacionadas con seguridad, estabilidad y uso apropiado de recursos.

Incluye aspectos como:

- Uso de *HTTPS* para comunicación segura.
- Evitar dependencias vulnerables o desactualizadas.
- Correcto manejo de recursos multimedia.
- Optimización y compresión aplicada en imágenes y *scripts*.
- Uso de Apis vigentes.

- **SEO (Optimización para motores de búsqueda)**

Comprueba si la página sigue recomendaciones básicas para que los motores de búsqueda puedan rastrearla e interpretarla. No cubre todos los factores de posicionamiento [13]. Entre los elementos que evalúa se encuentran:

- Presencia de títulos y meta descripciones.
- Enlaces con texto descriptivo.
- Uso adecuado de etiquetas *header*, *main*, *footer*.
- Atributos *alt* en imágenes para interpretación del contenido.
- Correcto funcionamiento en dispositivos móviles.

# 3 Diseño y Arquitectura

A continuación, se mostrarán diagramas realizados en *UML*, con el fin de describir de forma clara la estructura y funcionamiento del sistema. *UML* es un estándar de modelado que permite representar visualmente los componentes, procesos e interacciones, facilitando la comprensión del diseño y asegurando que la arquitectura propuesta cumpla con los requisitos del proyecto.

## 3.1 Diagrama de Casos de Uso

Este diagrama permite identificar las principales interacciones entre los actores y las funcionalidades del sistema. En la plataforma se consideran tres actores principales: el usuario, que engloba tanto a turistas como a residentes; el negocio local, que dispone

de funcionalidades para gestionar sus servicios; y la institución u organización, que puede acceder al panel de indicadores turísticos.

El diagrama permite visualizar tareas comunes como registrarse, iniciar sesión, editar perfil o eliminar cuenta, así como las funcionalidades que el usuario puede realizar al explorar la oferta disponible: aplicar filtros, revisar servicios y eventos, y generar itinerarios personalizados según sus intereses. Por su parte, los comercios validados pueden registrar, editar o eliminar sus servicios y acceder a estadísticas relacionadas con su actividad dentro de la plataforma.

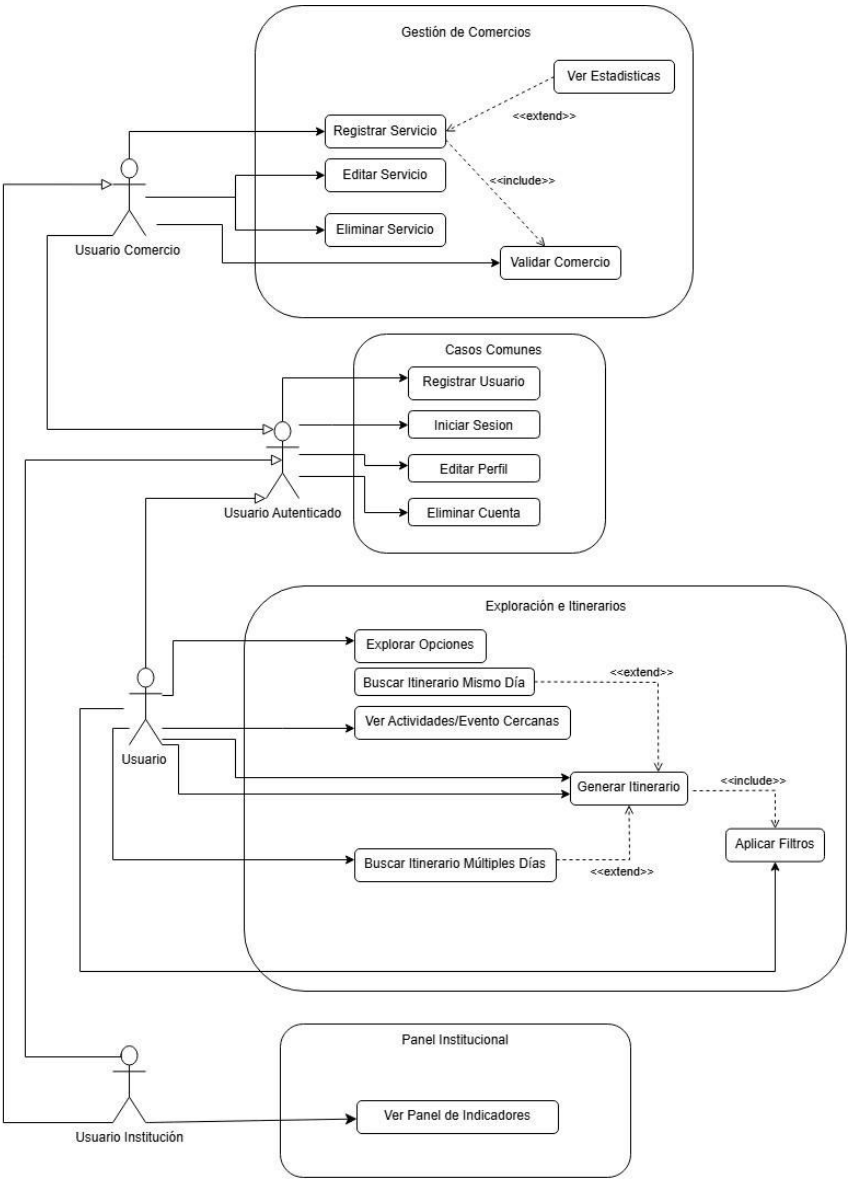



**Figura 1.** Diagrama Casos de Uso.
Fuente: Elaboración propia.

## 3.2 Diagrama de Clases

Para describir la arquitectura de la plataforma, el Diagrama de Clases se ha dividido en cuatro módulos temáticos que representan las áreas funcionales del sistema: gestión de usuarios y roles; contenido turístico y ubicación; planificación y componentes de interfaz.

### 3.2.1 Módulo de Gestión de Usuarios y Roles

Este módulo define la jerarquía de los actores del sistema, sus roles especializados y las entidades de soporte que almacenan sus preferencias y actividad básica (intereses, favoritos y reservas).

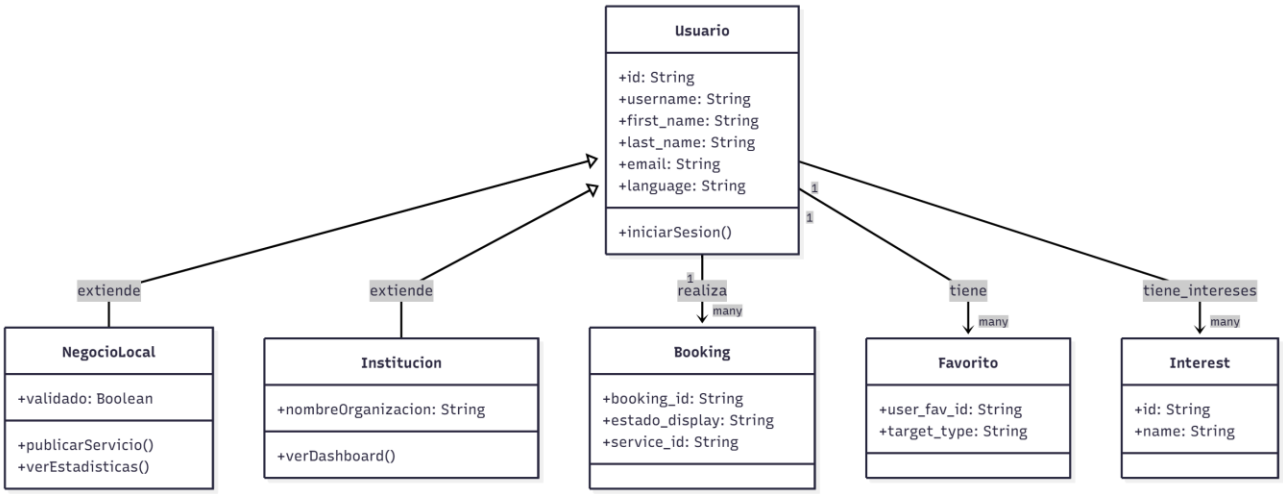



**Figura 2.** Diagrama de clase
Fuente: Elaboración propia

### 3.2.2 Módulo de Contenido Turístico y Ubicación

Este módulo modela la información turística central de la plataforma, incluyendo la estructura de las ubicaciones geográficas y la especialización de las ofertas de servicios.

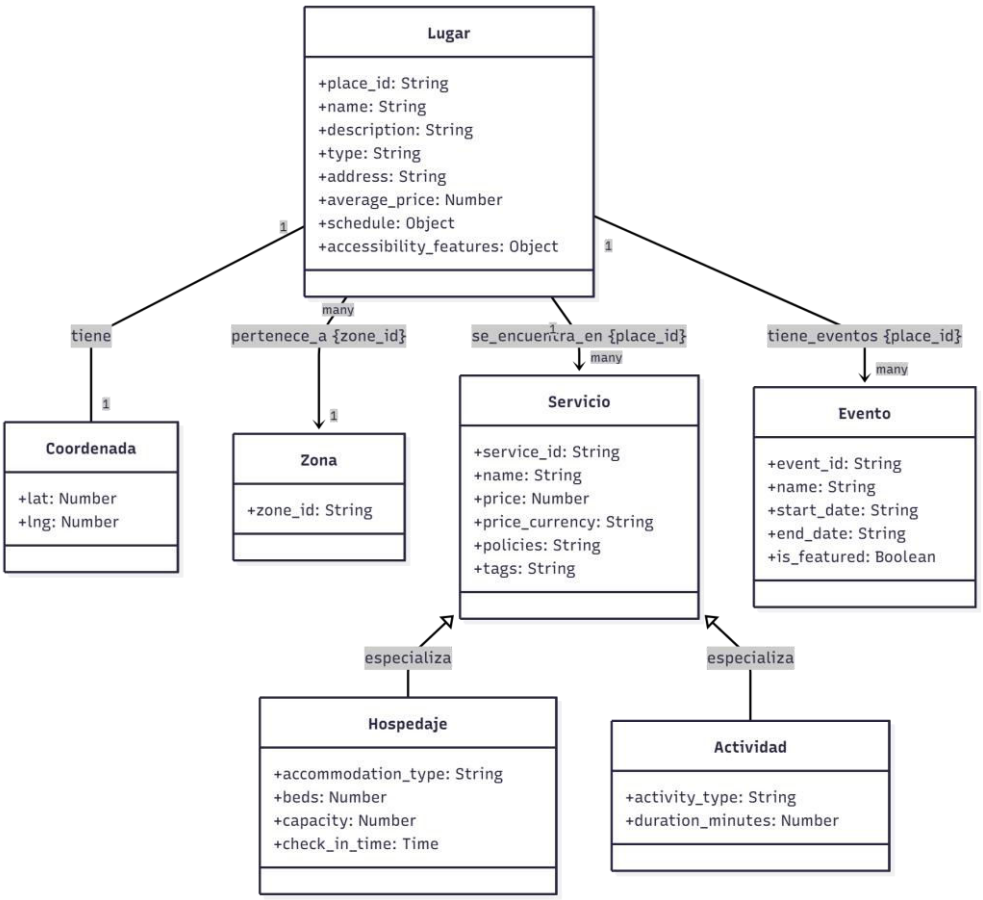



**Figura 3.** Diagrama Clase – Módulo de Turismo y Ubicación
Fuente: Elaboración propia

### 3.2.3 Módulo de Contenido Planificación

Este módulo define la funcionalidad principal de la plataforma: la creación y gestión de itinerarios personalizados, vinculando los servicios y eventos específicos.

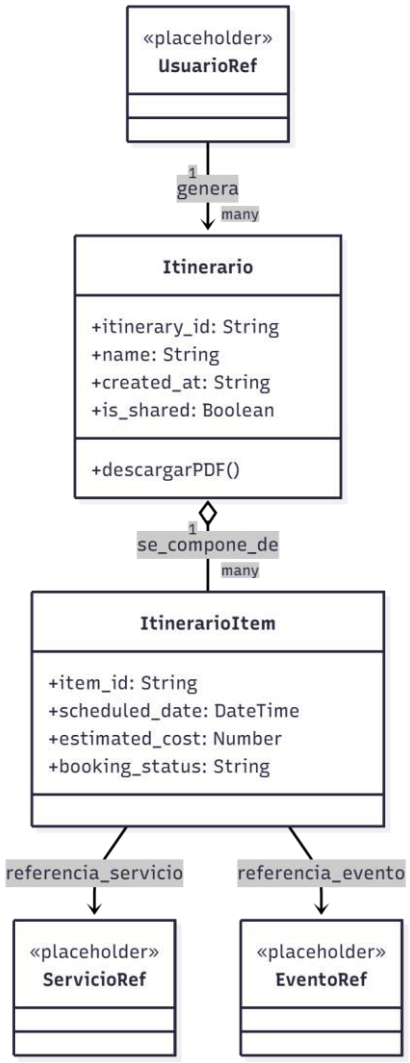



**Figura 4.** Diagrama Clase – Módulo de Itinerarios
Fuente: Elaboración propia

### 3.2.4 Módulo de Componentes de Interfaz y Flujo

Este módulo representa los componentes de React que controlan los flujos de usuario complejos y la visualización de los datos del dominio.

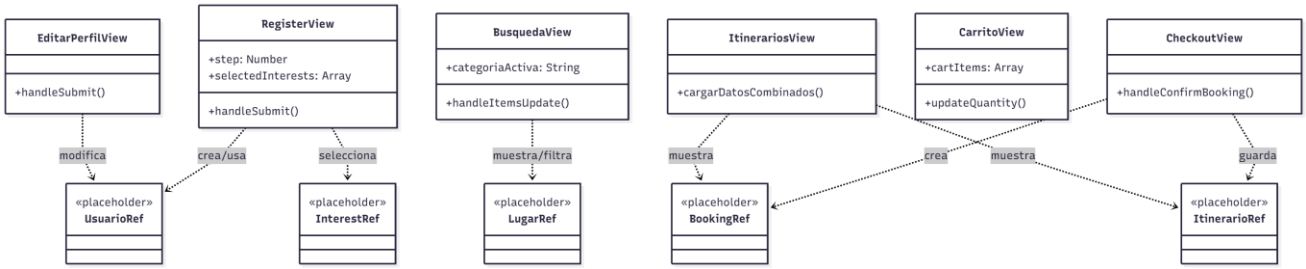



**Figura 5.** Diagrama Clase – Módulo de Interfaz y Flujo
Fuente: Elaboración propia

## 3.3 Arquitectura Basada en Componentes

La arquitectura del *frontend* de Amukan se construye siguiendo un enfoque modular basado en componentes reutilizables, característico de *React*. Esta organización permite mantener un código más ordenado, escalable y fácil de extender a medida que la plataforma incorpore nuevas funcionalidades o módulos. Para ello, el proyecto se estructuró en carpetas que agrupan elementos de acuerdo con su rol dentro del sistema, favoreciendo la separación de responsabilidades y la coherencia interna del código.

### 3.3.1 Diagrama de Componentes

El diagrama de componentes describe la arquitectura lógica del sistema Amukan, detallando la interacción entre módulos del *frontend*, el *backend* desarrollado en *Django REST y* la base de datos basada en *PostgreSQL/PostGIS*. Cada componente expone o consume interfaces definidas, permitiendo una estructura modular mantenible y desacoplada.

En el *frontend*, construido con *React* y *Vite*, se distinguen los componentes encargados de la navegación, la gestión del estado global, y los distintos módulos que componen la experiencia de usuario. Entre ellos se incluyen las vistas principales: *Home*, buscador/exploración, *dashboard*/gestión e itinerarios. Adicionalmente, el sistema incorpora componentes *core* como el mapa interactivo, basado en *MapLibre*, y los elementos visuales reutilizables (*UI Shared Components*) que conforman la interfaz.
Toda la comunicación con el servidor se realiza a través del *API Client,* que centraliza las solicitudes *HTTP*, la gestión de *tokens* y el manejo de flujos asíncronos.

El *backend*, organizado como un conjunto de servicios *REST*, expone interfaces independientes para la administración y consulta de puntos de interés y servicios turísticos (*Point of Interest (POI)*/Servicios *API*), la gestión de itinerarios y reservas (Itinerario/*Booking API*), la generación de recomendaciones personalizadas (*Recommendations API*) y la autenticación de usuarios (*Auth API*). Cada módulo mantiene su propia responsabilidad, permitiendo la escalabilidad y la reutilización del modelo de datos.

Finalmente, la base de datos *PostGIS* funciona como el repositorio central de información geoespacial y operacional del sistema. El *backend* accede a esta capa mediante una

interfaz de almacenamiento especializada, necesaria para ejecutar consultas espaciales, almacenar itinerarios y administrar los datos de puntos de interés.

En conjunto, el diagrama evidencia una arquitectura orientada a componentes donde el *frontend*, el *backend* y la base de datos interactúan mediante interfaces definidas, contemplando la modularidad, la mantenibilidad y la integración futura de nuevas funcionalidades.

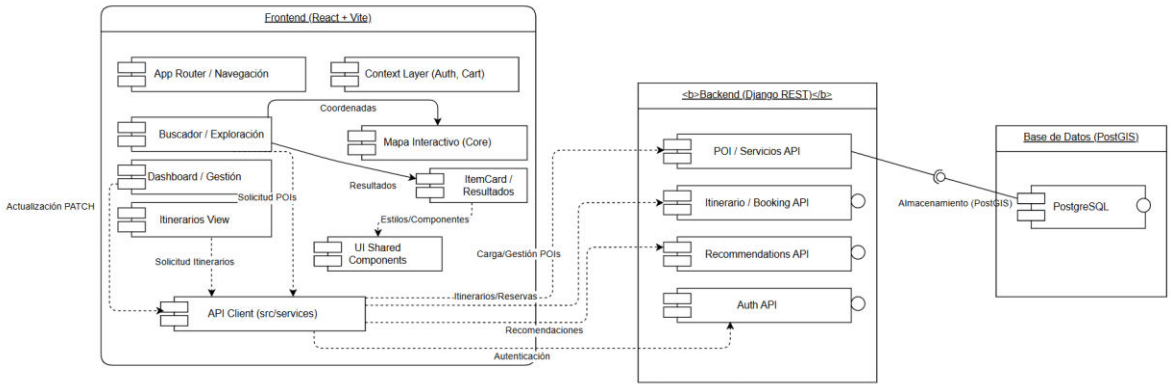



**Figura 6.** Diagrama de Componentes
Fuente: Elaboración propia

### 3.3.2 Estructura General del Proyecto

La estructura de carpetas del *frontend* se diseñó para optimizar el flujo de trabajo y la navegación interna del código, alineándose con el entorno de desarrollo de *Vite.*




**Figura 7.** Estructura de las carpetas
Fuente: Equipo Amukan

Cada carpeta cumple con un rol específico dentro del proyecto.

- ***assets***: Carpeta que contiene recursos estáticos, tales como imágenes o íconos.

- ***componentes***: Contiene componentes reutilizables, tales como tarjetas con servicios, formularios, *headers*, *footers* y elementos comunes utilizados en distintas páginas.
- ***context***: Almacena los contextos globales del proyecto para manejar roles, sesión o datos compartidos entre múltiples componentes.

- ***hooks***: Contiene *hooks* personalizados que encapsulan lógica reutilizable, tales como funciones para manejar filtros, gestionar estados específicos, obtener datos o centralizar comportamientos usados en múltiples componentes.

- ***services***: Incluye funciones que permiten obtener, procesar o transformar datos desde los archivos *JSON* u otras fuentes. Su función principal es abstraer el acceso a datos y simplificar la comunicación entre la interfaz y la capa lógica.

- ***utils***: Carpeta destinada a funciones utilitarias que cumplen roles transversales dentro del proyecto, como formateo de fechas, normalización de datos o funciones de apoyo para cálculos y validaciones.

- ***vistas***: Contiene las vistas o páginas principales del sistema. Cada carpeta dentro de vistas representa una sección completa (por ejemplo, inicio, registro, mapa, *dashboard*, etc.) y combina componentes, *hooks* y servicios para formar la estructura funcional de cada módulo.

# 4 Desarrollo e Implementación del Frontend

Este capítulo describe la implementación del frontend de Amukan, enfocándose en cómo se aplicó la arquitectura modular basada en componentes y cómo se desarrollaron las principales funcionalidades utilizando *React.js, Vite* y *Tailwind CSS*. Las decisiones de construcción priorizaron la adaptabilidad, la separación de responsabilidades y la correcta adaptación a distintos dispositivos.

## 4.1 Uso de *Milanote* para Organización

Durante el desarrollo del *frontend* se utilizó la herramienta *Milanote* como apoyo en la planificación y coordinación del proyecto. Esta plataforma permitió registrar y visualizar de manera colaborativa las tareas pendientes menores, errores detectados y mejoras propuestas. Además, facilitó la asignación de responsabilidades dentro del equipo y sirvió como un repositorio de documentación referente al avance del desarrollo, manteniendo un control ordenado de las modificaciones realizadas.




**Figura 8.** Extracto de Tablero de Organización Frontend para Amukan
Fuente: Equipo Amukan

## 4.2 Módulos de Autorización y Gestión de Usuarios

La gestión de usuarios y roles se implementó mediante estados globales y contextos, permitiendo controlar el acceso a funcionalidades específicas según el tipo de cuenta. La gestión de usuarios y roles se relaciona principalmente con el ***OE1*** y proporciona funcionalidades transversales a los demás módulos.

### 4.2.1 Control de Acceso y Roles

El acceso a secciones restringidas se administra a través del *AuthContext, ubicado en src/context/,* el cual almacena la información del usuario autenticado y su rol.

La lógica se implementa mediante *renderizado condicional,* un patrón común en React para restringir vistas sin recargar la página. Este mecanismo evita la renderización del contenido de paneles administrativos cuando el usuario no está autorizado, lo que contribuye indirectamente a la eficiencia y orden del *layout*.

- **Protección de vistas:**
  - En *Dashboard.jsx*, se valida que el usuario esté autenticado y además posea un tipo de organización (Negocio Local o Institución). Si no se cumple con estas condiciones, la vista se bloquea.

```
// Fragmento de src/vistas/Dashboard.jsx: Control de acceso basado en roles

if (!isAuthenticated) {
    return <Login />;
} // Se bloquea si no está logeado

if (!user.is_superuser &&! hasOrganization) {
    return (
        <div className="acceso-restringido">
            <h2 className="titulo">Acceso Restringido</h2>
            <p>Debes pertenecer a una organización para acceder.</p>
        </div>
    ); // Bloquea si no tiene el rol de Organización
}

// Si la validación pasa, se renderiza el Dashboard
```

**Figura 9.** Extracto de Código Control de Acceso
Fuente: Equipo Amukan

Este patrón garantiza que los componentes de *InteriorDashboard.jsx* solo se inicialicen si la validación del rol es exitosa.

- **Gestión de perfil:**
  - La vista *EditarPerfil.jsx* muestra cómo el frontend se comunica con el backend para actualizar solo los datos modificados.

  - Los *select* (género, nacionalidad, idioma) se obtienen dinámicamente, y la actualización del perfil se realiza mediante una solicitud *PATCH,* enviando únicamente los campos cambiados.

```
// Fragmento de src/vistas/EditarPerfil.jsx: Construcción del payload PATCH

const updatedFields = Object.fromEntries(
  Object.entries(formData).filter(([key, value]) => value !== originalData[key])
);

// Se verifica que updatedFields no esté vacío antes de enviar el PATCH.
const res = await api.patch("/user/me/", updatedFields);
```

**Figura 10.** Extracto de Código Construcción de *payload PATCH*
Fuente: Equipo Amukan

Este patrón evita enviar información innecesaria y optimiza la comunicación con el backend.

### 4.2.2 Flujo de Registro Personalizado

El registro de usuario se implementó en *Register.jsx* mediante un flujo dividido en cuatro pasos. Esto permite recopilar datos personales y preferencias de forma ordenada. Además, se implementaron validaciones asíncronas desde el navegador para verificar de manera temprana si el username o el email se encuentran previamente registrados. Estas verificaciones actúan como controles de interfaz que mejoran la experiencia del usuario al entregar retroalimentación inmediata, complementando el proceso de autenticación y las validaciones de seguridad definitivas que aseguran la integridad de los datos en el *backend*.

```
// Fragmento de src/vistas/Register.jsx: Validación Asíncrona

const checkUsernameAvailability = async (username) => {
  if (!username) return setUsernameStatus(null);
  try {
    const res = await fetch(`${API_BASE_URL}/user/check-username/?username=${username}`);
    const data = await res.json();
    setUsernameStatus(data.available ? "available" : "taken");
  } catch (err) {
    console.error("Error al verificar usuario:", err);
    setUsernameStatus(null);
  }
};

useEffect(() => {
  if (!pendingUsername) {
    setUsernameStatus(null);
    return;
  }

  // Implementación de debounce para reducir peticiones a la API
  const timer = setTimeout(() => {
    checkUsernameAvailability(pendingUsername);
  }, 600);

  return () => clearTimeout(timer); // Limpieza de timer
}, [pendingUsername]);
```

**Figura 11.** Extracto de Código Validación asincrónica
Fuente: Equipo Amukan

La combinación de validación *asincrónica* y *debounce* permite evitar solicitudes innecesarias al backend.

## 4.3 Módulos de Exploración y Resultados de Búsqueda (OE2)

Estos módulos permiten al usuario explorar servicios, actividades y paquetes turísticos desde una vista centralizada.

### 4.3.1 Contenedor de Búsqueda

El componente *Busqueda.jsx* funciona como controlador principal de la búsqueda. Administra:

- La categoría activa (hospedaje, paquetes, gastronomía, etc.)
- La lista de resultados
- Las coordenadas utilizadas por el mapa

La función *handleItemsUpdate* toma los ítems seleccionados y sus coordenadas, enviando la información al componente *mapa* para la visualización integrada.

```
//Fragmento de src/vistas/Busqueda.jsx: Función handleItemsUpdate

const handleItemsUpdate = useCallback(
    (itemsParaLista, coordsParaMapa) => {
        setItems(itemsParaLista);
        if (coordsParaMapa && coordsParaMapa.length > 0) {
            // Usa las coordenadas proporcionadas para el mapa y el zoom
            setCoordenadas(coordsParaMapa);
            setCoordenadasZoom(coordsParaMapa);
        } else {
            // Si no hay coordenadas explícitas, las extrae de los ítems
            const coords = itemsParaLista
                .map((i) => {
                    // ... lógica para extraer [lat, lon] de i.lugar[PG19.1]
                    return i.lugar;
                })
                .filter((c) => c !== null);
            setCoordenadas(coords);
            setCoordenadasZoom(null);
        }
    },
    [categoriaActiva]
);
```

**Figura 12.** Extracto de Código Función *HandleItemsUpdate*
Fuente: Equipo Amukan

El uso del *useCallback* garantiza que la función *handleItemsUpdate* mantenga la misma referencia, lo cual optimiza el rendimiento al evitar renderizados innecesarios de los componentes hijos. La lógica de control establece una única fuente de verdad (coordenadas) para el componente mapa, logrando la centralización de la información de búsqueda.

Responsividad:
La vista de búsqueda incorpora un diseño responsivo que se adapta tanto a pantallas grandes como a dispositivos móviles:

- **En escritorio**: se utilizan paneles laterales colapsables que permiten al usuario ocultar o mostrar el panel de resultados.
- **En dispositivos móviles**: se utiliza un panel deslizable para optimizar el espacio y facilitar la navegación.

```
<div
  className={`relative transition-all duration-500 ease-in-out ${
    panelIzquierdoVisible
      ? "md:w-[60%] lg:w-[65%] xl:w-[70%]"
      : "w-full"
  }`}
>
  <Mapa ... />
</div>
```

**Figura 13.** Extracto de Código Responsivo para *Busqueda.jsx*
Fuente: Equipo Amukan

Este patrón permite que, si el panel lateral está visible, el mapa ocupe un porcentaje del ancho (60–70 % del ancho disponible). Cuando el panel está oculto, el mapa se expande a *w-full*, utilizando todo el espacio disponible.

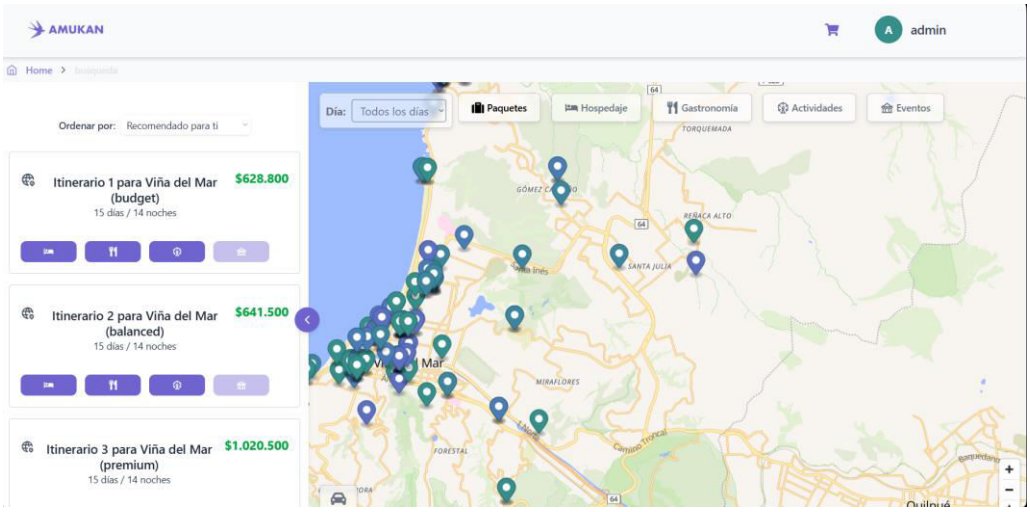



**Figura 14.** Vista de escritorio del mapa + itinerario
Fuente: Equipo Amukan

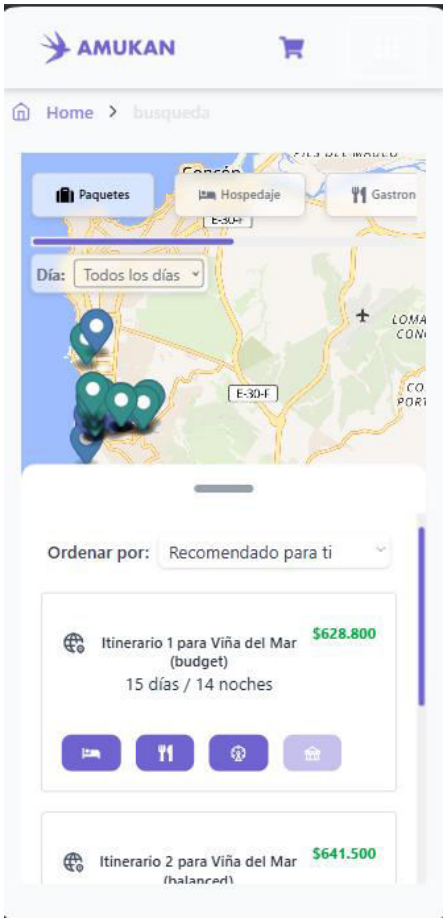



**Figura 15.** Vista para móvil del Mapa + Itinerario
Fuente: Equipo Amukan

### 4.3.2 Función "Cerca Mío"

El componente *CercaMio.jsx* implementa el flujo de exploración destinado a los usuarios residentes o que requieran una actividad o evento para ese día, permitiendo identificar actividades y servicios cercanos a su ubicación. Este módulo responde directamente a las necesidades de la comunidad local, ofreciendo una experiencia de búsqueda rápida y orientada al contexto geográfico.
En este flujo, el componente *PuntosCercanos.jsx* (desarrollado por otro integrante del equipo) es el encargado de manejar la lógica geoespacial, específicamente:

- Obtener la ubicación actual del usuario.
- Filtrar los resultados dentro de un radio definido.
- Retornar los resultados listos para ser representados en el mapa.

Desde el frontend, *CercaMio.jsx* se centra en presentar los resultados de forma inmediata y en mantener la coherencia visual con la vista general de búsqueda. Su diseño responsivo sigue un enfoque similar al de *Busqueda.jsx*, ya que ambos comparten el objetivo de mostrar un mapa acompañado de resultados filtrados dinámicamente. La diferencia principal es que *CercaMio.jsx* incluye la visualización de la ubicación del usuario y un radio aproximado de 5 km, destacando las actividades y eventos cercanos dentro de esa área.

Este enfoque permite ofrecer una experiencia simple y eficiente, alineada con el objetivo de contar con un frontend adaptable, especialmente orientado a usuarios móviles que necesitan información georreferenciada en tiempo real.

## 4.4 Módulo de Planificación y Gestión de Itinerarios

El módulo de planificación desarrollado en la vista *Itinerarios.jsx* concentra toda la actividad turística del usuario en una vista unificada, mostrando tanto los itinerarios planificados como las reservas individuales realizadas en la plataforma. Este componente cumple un rol fundamental dentro del flujo de experiencia de usuario, ya que presenta de forma clara y responsiva la información más relevante de su actividad dentro de Amukan.

Desde la perspectiva del frontend, la implementación prioriza la eficiencia en la carga, la eliminación de duplicidad de información y la adaptación visual a diferentes dispositivos, coherente con los objetivos generales del proyecto.

### 4.4.1 Integración de Itinerarios y Reservas Individuales

Para construir una vista consolidada, el componente realiza la carga de datos provenientes de dos endpoints distintos: uno correspondiente a los itinerarios creados por el usuario y otro a sus reservas individuales. Para obtener esta información, se utilizó *Promise.all(),* permitiendo ejecutar ambas solicitudes de forma concurrente. Esta estrategia evita procesarlas secuencialmente, con el propósito de optimizar la recuperación de datos desde la interfaz. Adicionalmente, se implementó un manejo de errores para estas peticiones; en caso de que alguna de las dos solicitudes falle (por ejemplo, por problemas de red o del servidor), la interfaz captura la excepción para evitar el quiebre de la aplicación y notifica al usuario, manteniendo un comportamiento controlado.

Una vez obtenidas las respuestas, el frontend aplica una lógica de filtrado que evita mostrar información duplicada. Para ello, se construye un *Set* que almacena los identificadores de las reservas que ya formen parte de algún itinerario.

Posteriormente, las reservas individuales se filtran dejando únicamente aquellas que no estén asociadas a ningún plan, conocidas como reservas independientes.

```jsx
// src/vistas/Itinerarios.jsx

const cargarDatosCombinados = async () => {
    const [itinerariosResponse, bookingsResponse] = await Promise.all([
      api.get("/travel/my-itineraries/"),
      api.get("/bookings/")
    ]);

    const itineraryBookingIds = new Set();
    itinerariosResponse.data.forEach(itinerario => {
      itinerario.items.forEach(item => {
        if (item.booking_id) itineraryBookingIds.add(item.booking_id);
      });
    });

    const standalone = bookingsResponse.data.filter(
      booking => !itineraryBookingIds.has(booking.booking_id)
    );

    setItinerarios(itinerariosResponse.data);
    setStandaloneBookings(standalone);
};
```

**Figura 16.** Extracto de Código Carga de Datos
Fuente: Equipo Amukan

Esta estrategia garantiza que el usuario no vea elementos repetidos y que la información se muestre de manera clara y organizada.

### 4.4.2 Diseño de Interfaz y Tarjetas Responsivas

Los elementos visuales del módulo se presentan mediante tarjetas responsivas, tanto para itinerarios como para reservas. Para ello, se utilizan clases de *Tailwind CSS* que permiten adaptar la disposición del contenido según el tamaño de pantalla:

- **En dispositivos móviles**: Las tarjetas utilizan una estructura vertical (*flex-col*) que favorece la lectura y la navegación.
- **En pantallas de escritorio**: Se opta por una distribución horizontal (*md:flex-row*), incorporando imágenes o íconos en una columna lateral para mantener una jerarquía visual más informativa.

Además, se aplican límites dinámicos como *max-w-6xl, w-full y md:w-1/3* para asegurar que los elementos mantengan proporciones adecuadas sin importar la resolución del dispositivo.

Esta estrategia refuerza la consistencia visual entre los distintos módulos de Amukan, cumpliendo con el objetivo de ofrecer una interfaz adaptable y accesible.

```jsx
// Fragmento de src/vistas/Itinerarios.jsx: Definición Responsiva de Tarjeta

<div
  className="bg-white flex flex-col md:flex-row shadow-lg rounded-lg w-full max-w-6xl overflow-hidden"
>
  {/* Columna 1: Imagen/Ícono (Visible solo a partir de md:, ocupa 1/3) */}
  <div className="hidden md:block w-full md:w-1/3">
    {/* ... Imagen ... */}
  </div>

  {/* Columna 2: Detalles (En móvil es columna, en escritorio es parte de la fila) */}
  <div className="p-6 w-full md:w-2/3 text-left flex flex-col justify-center">
    {/* ... Contenido ... */}
  </div>
</div>
```

**Figura 17.** Ejemplo de Código Responsividad de Cards
Fuente: Equipo Amukan

## 4.5 Panel de Visualización de Datos e Indicadores

El panel de indicadores corresponde al espacio destinado a instituciones y organizaciones, permitiendo visualizar métricas clave sobre la actividad turística registrada en Amukan. Aunque el procesamiento de datos se realiza desde *backend* y *Metabase*, el *frontend* cumple el rol de presentar esta información en una interfaz clara y accesible para la toma de decisiones.

Desde la perspectiva del desarrollo *frontend*, este módulo utiliza una estructura basada en componentes reutilizables que reciben los datos ya procesados desde la *API*. Para cada indicador, se implementan tarjetas informativas y espacios para gráficos integrados.

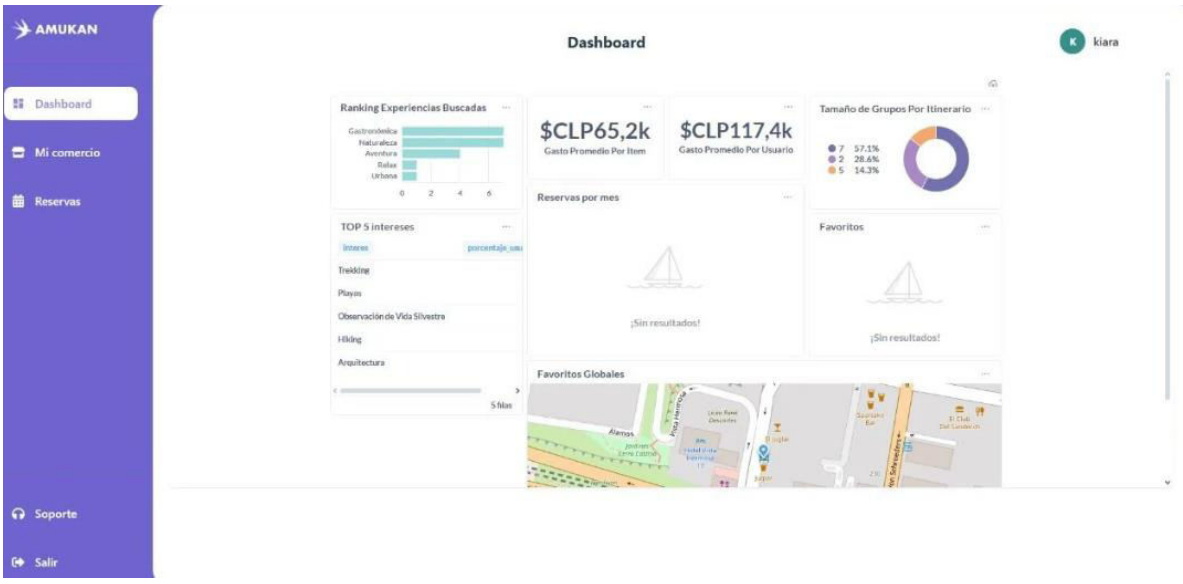



**Figura 18.** Panel para Instituciones y organizaciones.
Fuente: Equipo Amukan

### 4.5.1 Integración de Gráficos

Cada gráfico o tarjeta de datos se implementa como un componente independiente ubicado en *src/componentes/InteriorDashboard.jsx.* El componente principal organiza la información recibida de la API mapeando los datos para crear cada tarjeta e indicador.

La estructura del *layout* de esta sección se basa en una cuadrícula flexible (*grid*) de Tailwind CSS. Esta implementación permite que los indicadores (tarjetas de datos, rankings y gráficos) se distribuyan de forma organizada y que la interfaz se adapte para mostrar más o menos contenido simultáneamente en la pantalla de escritorio.

Aquí se muestra la implementación de la cuadrícula que organiza los indicadores.

```
// Fragmento de src/componentes/InteriorDashboard.jsx: Distribución de la Cuadrícula

<div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-4 gap-4 w-full">
    {/* Grid con el primer gráfico a todo el ancho */}
    <div className="col-span-1 sm:col-span-2 lg:col-span-4">
        {/* Componente que aloja el iframe de Metabase */}
        <EmbeddedMetabaseFrame ... />
    </div>

    {/* Tarjeta 1: Ranking */}
    <RankingComponent ... />

    {/* Tarjeta 2: Gasto Promedio */}
    <IndicadorCard ... />

    {/* Tarjeta 3: Tamaño de Grupos */}
    <GraficoComponent ... />
</div>
```

**Figura 19.** Extracto de Código Distribución de Cuadrículas.
Fuente: Equipo Amukan

Este diseño permite que los indicadores se organicen automáticamente según el tamaño de la pantalla, manteniendo una presentación clara y fácil de leer en cualquier dispositivo (**OE4**).

## 4.6 Módulo de Gestión para Negocios Locales

Este módulo entrega a los Negocios locales y pymes un espacio para publicar servicios, promociones y actividades turísticas. Su diseño se integra con el flujo general de la plataforma, permitiendo que estas publicaciones aparezcan en la vista de búsqueda y en las recomendaciones generales.

### 4.6.1 Publicación y Edición de Servicios

Las vistas desarrolladas en este módulo permiten crear nuevos servicios y editar información existente. El acceso a estas vistas está restringido, utilizando *AuthContext* para verificar que el usuario pertenece a una organización.

En los componentes *CrearServicio.jsx* y *EditarServicio.jsx* se implementó una lógica de renderizado condicional que permite presentar campos específicos según la categoría del

servicio, como hospedaje o actividades. De esta manera, los formularios solicitan la información correspondiente al tipo de publicación seleccionado.

```jsx
// Fragmento de src/vistas/CrearServicio.jsx: Renderizado condicional de campos

const renderCamposEspecificos = () => {
    switch(serviceType) {
        case "accommodation":
            return (
                <div className="grid grid-cols-1 md:grid-cols-2 gap-4 mt-4">
                    {/* Campos: accommodationType, beds, roomCapacity, checkInTime, etc. */}
                </div>
            );
        case "activity":
            return (
                <div className="grid grid-cols-1 md:grid-cols-2 gap-4 mt-4">
                    {/* Campos: activityType, durationMinutes, guideIncluded, etc. */}
                </div>
            );
        default: return null;
    }
};
```

**Figura 20.** Extracto de Código Renderizado Condicional.
Fuente: Equipo Amukan

Este módulo emplea solicitudes *HTTP POST, PUT* y *PATCH* para la creación y actualización, enviando únicamente los campos modificados al backend para optimizar la comunicación.

### 4.6.2 Responsividad y consistencia Visual

La interfaz de los formularios se adapta a distintos tamaños de pantalla, manteniendo una estructura y presentación consistentes en dispositivos móviles, tablets y computadores.

- Adaptabilidad del Layout: Los formularios de publicación aprovechan el diseño responsivo de Tailwind CSS para organizar los campos de manera eficiente.
    - **En móvil:** Los formularios se muestran como bloques verticales simples (*flex-col*), con inputs de ancho completo (*w-full*).
    - **En escritorio:** La interfaz utiliza *layouts* de dos columnas (*grid grid-cols-1 md:grid-cols-2*) para mejorar la legibilidad y la organización de los campos.

Este patrón de cuadrícula adaptable es fundamental para la usabilidad en pantallas grandes:

```jsx
// Fragmento de src/vistas/CrearServicio.jsx: Cuadrícula Responsiva para la Organización de Campos

<div className="grid grid-cols-1 md:grid-cols-2 gap-4">
    {/* Primer input (ej. Nombre) */}
    <div>
        <label>Nombre</label>
        <input type="text" ... />
    </div>

    {/* Segundo input (ej. Apellido/Tipo) */}
    <div>
        <label>Tipo de Lugar</label>
        <select ... />
    </div>
</div>
```

**Figura 21.** Extracto de Código Cuadrícula Responsiva.
Fuente: Equipo Amukan




**Figura 22.** Vista Crear Servicio.
Fuente: Equipo Amukan.

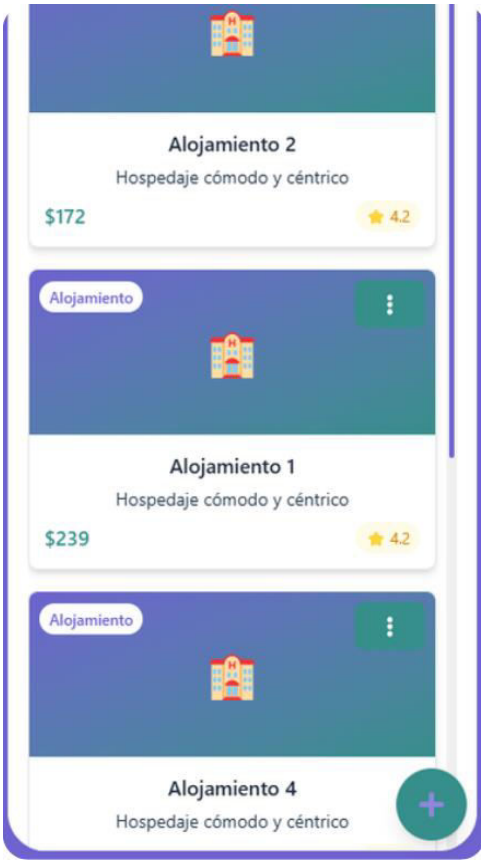



**Figura 23.** Servicios Creados - Alojamiento.
Fuente: Equipo Amukan

La implementación de este módulo se relaciona con el **OE3**, correspondiente al desarrollo de herramientas de gestión para negocios locales. Las vistas desarrolladas incorporan formularios de publicación y edición de servicios, organizados mediante componentes responsivos que permiten adaptar la disposición de los campos a distintos tamaños de pantalla.

## 4.7 Exportación de Itinerarios a PDF

Como parte del módulo de planificación turística asociado al **OE2**, el *frontend* de Amukan incorpora una funcionalidad que permite exportar los itinerarios generados en formato PDF. Esta opción facilita el almacenamiento del documento en el dispositivo del usuario para su posterior consulta, sin requerir que la plataforma funcione sin conexión a Internet. La implementación se encuentra integrada en la vista *ItinerarioGenerado.jsx.*

### 4.7.1 Descarga de Itinerarios

El botón de descarga se encuentra dentro de la vista *ItinerarioGenerado.jsx,* permitiendo exportar la planificación completa del usuario en formato PDF para su almacenamiento y posterior consulta.

La función *generarPDF* en el frontend se encarga de:

- Clonar el contenido de la vista (utilizando *html2canvas*).
- Generar el documento PDF (*jspdf*).
- Forzar la descarga en el navegador con el nombre del itinerario.

```
// Fragmento de src/componentes/ItinerarioGenerado.jsx: Botón de Descarga

<button
    onClick={generarPDF}
    disabled={generandoPDF}
    className="bg-purple-600 px-6 py-3 flex items-center gap-3 text-white font-semibold rounded-lg ..."
>
    <FaFilePdf />
    {generandoPDF ? 'Generando PDF...' : 'Descargar como PDF'}
</button>
```

**Figura 24.** Extracto de Código Botón Descarga PDF
Fuente: Equipo Amukan.




**Figura 25.** Vista para Guardar Itinerario como PDF.
Fuente: Equipo Amukan.

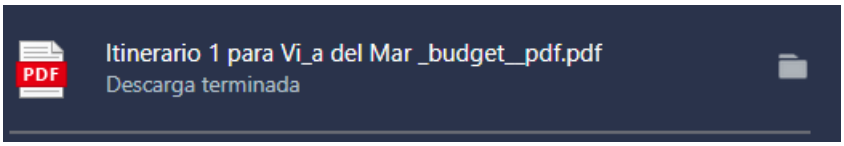



**Figura 26.** Itinerario guardado como PDF
Fuente: Elaboración propia

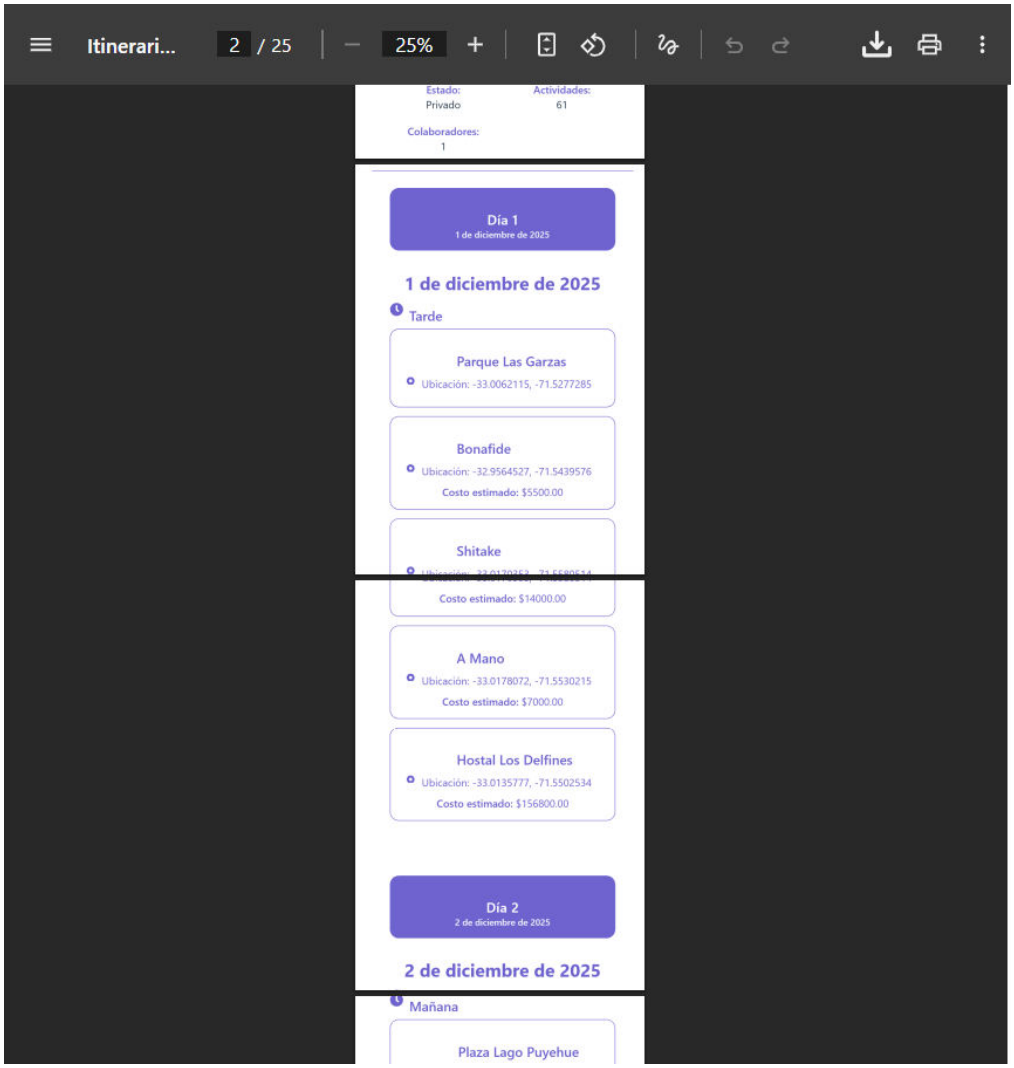



**Figura 27.** Itinerario como PDF
Fuente: Elaboración propia

Como limitación de esta implementación, la generación del PDF mediante la captura visual del contenido puede producir documentos cuyo texto no sea seleccionable ni interpretable por lectores de pantalla. Asimismo, la división del contenido entre páginas puede requerir ajustes adicionales. La generación de documentos PDF con texto accesible y una paginación más precisa se considera una oportunidad de mejora futura.

# 5 Pruebas y Validaciones

Este capítulo presenta la validación técnica del *frontend* desarrollado para Amukan, con el propósito de evaluar su comportamiento en términos de responsividad, rendimiento, accesibilidad y buenas prácticas de desarrollo web. Asimismo, se considera la evidencia de implementación documentada en los capítulos anteriores para determinar el cumplimiento de los objetivos establecidos en el proyecto.

La evaluación técnica se organizó en dos procedimientos complementarios. En primer lugar, se realizaron pruebas de responsividad mediante la emulación de resoluciones correspondientes a dispositivos móviles, tablets y computadores de escritorio, con el fin de examinar la adaptación de los componentes y la disponibilidad de los elementos principales de las interfaces. En segundo lugar, se efectuaron auditorías automatizadas mediante *Google Lighthouse*, considerando indicadores de rendimiento (*Performance*), accesibilidad (*Accessibility*), buenas prácticas (*Best Practices*) y optimización para motores de búsqueda (*SEO*).

Para ambas evaluaciones se seleccionaron las vistas registro, mapa, mi comercio y panel institucional, debido a que presentan diferentes estructuras y componentes dentro del *frontend*. Las pruebas se realizaron sobre una compilación optimizada de la aplicación, generada mediante *Vite* y ejecutada en un entorno local controlado, manteniendo la comunicación con los servicios disponibles del *backend*.

Los resultados obtenidos se contrastaron con los criterios de aprobación establecidos para el proyecto, permitiendo identificar fortalezas y limitaciones de la implementación. Finalmente, se evaluó el cumplimiento de los objetivos específicos mediante las evidencias presentadas en los capítulos 3 y 4 y los resultados técnicos desarrollados en el presente capítulo. Este análisis proporcionó los antecedentes necesarios para fundamentar las conclusiones y las oportunidades de mejora del trabajo.

## 5.1 Metodología de Validación y Criterios

La validación técnica del *frontend* de Amukan se estructuró mediante verificaciones de responsividad y auditorías automatizadas con *Google Lighthouse*, con el propósito de evaluar el comportamiento de las interfaces implementadas bajo condiciones definidas. Estos procedimientos complementaron las evidencias de implementación presentadas en los capítulos anteriores y permitieron analizar el cumplimiento de los requisitos técnicos establecidos para el proyecto.

Debido a que Amukan no cuenta actualmente con una versión desplegada públicamente, las evaluaciones se desarrollaron en un entorno local controlado. Para ello, se utilizó una compilación del *frontend* optimizada para producción, generada mediante *Vite* y ejecutada a través de su herramienta de previsualización (*preview*). Esta configuración permitió analizar el comportamiento del código compilado, manteniendo la comunicación con el *backend Django* mediante contenedores *Docker*, sin considerar dicho entorno como un despliegue productivo definitivo.

La evaluación de responsividad se realizó mediante las herramientas de desarrollo del navegador, emulando resoluciones de 430 × 932 píxeles para dispositivos móviles, 820 × 1180 píxeles para tablets y 1440 × 900 píxeles para computadores de escritorio. Estas verificaciones permitieron examinar la disposición de los componentes, la visibilidad de

los elementos interactivos y la adaptación de las interfaces a diferentes tamaños de pantalla. Se consideró que estas pruebas correspondieron a una emulación de las condiciones de visualización y no a evaluaciones efectuadas sobre dispositivos físicos.

| Parámetro | Configuración utilizada |
|---|---|
| Herramienta | Chrome DevTools, modo de emulación de dispositivos (Device Toolbar). |
| Navegador | Chromium 152.0.0.0. |
| Entorno de ejecución | *Frontend* compilado para producción mediante *Vite*, ejecutado localmente. |
| Dispositivo móvil | Resolución emulada de 430 × 932 píxeles. |
| Tablet | Resolución emulada de 820 × 1180 píxeles. |
| Computador de escritorio | Resolución emulada de 1440 × 900 píxeles. |
| Perfil de red | Sin limitación artificial (No throttling) durante las comprobaciones de responsividad. |
| Vistas seleccionadas | mapa, registro, mi comercio y dashboard. |
| Criterios de evaluación | Distribución de componentes, legibilidad, visibilidad de controles, navegación, interacción y ausencia de desbordamiento horizontal. |
| Evidencia | Capturas de pantalla y observaciones del comportamiento de cada vista. |
| Fecha de evaluación | Desde el 14 de octubre de 2026. |
| Tipo de prueba | Simulación de resoluciones mediante navegador, sin utilizar dispositivos físicos. |

**Tabla 3**. Parámetros utilizados para la evaluación de responsividad
Fuente: Elaboración propia

Por otra parte, las auditorías automatizadas mediante *Google Lighthouse* se definieron para las vistas mapa, registro, mi Comercio y *dashboard*. Se establecieron tres ejecuciones independientes por vista, manteniendo el mismo dispositivo emulado y las mismas condiciones de evaluación. Debido a que los puntajes de *Lighthouse* pueden presentar variaciones entre ejecuciones, se definió la mediana como medida representativa de los resultados correspondientes a rendimiento (Performance), accesibilidad (*Accessibility*), buenas prácticas (*Best Practices*) y optimización para motores de búsqueda (*SEO*).

| Parámetro | Configuración registrada |
|---|---|
| Herramienta | *Google Lighthouse* |
| Versión | 13.4.1 |
| Navegador | Chromium 152.0.0.0 |
| Dispositivo emulado | Moto G Power |
| Perfil de red | Slow 4G |
| CPU | 4× slowdown |
| Método de limitación | Simulado (*simulate*) |
| Tipo de auditoría | Carga inicial de página |
| Fecha | 14 de octubre de 2026 |
| Entorno | Compilación de producción ejecutada localmente |
| Ejecuciones realizadas | 3 por vista (12 en total) |
| Tratamiento de resultados | Mediana de las tres ejecuciones |

**Tabla 4**. Parámetros de configuración de las auditorías con Google Lighthouse
Fuente: Elaboración propia

La utilización de condiciones de evaluación uniformes permitió establecer una base de comparación entre las vistas seleccionadas. Asimismo, el registro de los parámetros técnicos y la aplicación de la mediana como medida representativa contribuyeron a reducir la influencia de las variaciones entre ejecuciones, favoreciendo una interpretación consistente de los resultados.

Finalmente, los resultados de las pruebas se contrastaron con los criterios de aceptación establecidos para el proyecto, permitiendo identificar fortalezas, limitaciones y oportunidades de mejora dentro del alcance del *frontend* desarrollado.

### 5.1.1 Criterios de Aprobación

Para evaluar el cumplimiento de los requisitos técnicos del frontend de Amukan, se establecieron criterios de aprobación asociados con la responsividad y las auditorías automatizadas mediante *Google Lighthouse*. Estos criterios se fundamentaron en recomendaciones técnicas y estándares de referencia, permitiendo identificar el comportamiento de las interfaces y las oportunidades de mejora de la implementación.

- **Criterios de aprobación de responsividad**

La evaluación de responsividad se fundamentó en los principios del diseño web responsivo, los cuales establecen que las interfaces deben adaptar la presentación y distribución de sus contenidos a las dimensiones disponibles de la pantalla [4].

Para este procedimiento se definieron criterios cualitativos orientados a verificar que las funcionalidades principales permanecieran accesibles y que los componentes conservaran una distribución adecuada en las tres resoluciones seleccionadas.

| Criterio | Condición de aprobación |
|---|---|
| Adaptación de componentes | Los elementos de la interfaz se ajustan al espacio disponible sin perder información esencial. |
| Ausencia de desbordamiento horizontal | No se presenta desplazamiento horizontal innecesario que dificulte la navegación. |
| Visibilidad de controles | Los botones, campos y menús principales permanecen visibles o accesibles. |
| Legibilidad | Los textos y etiquetas se presentan sin recortes ni superposiciones. |
| Navegación e interacción | Las funcionalidades principales pueden utilizarse en las resoluciones evaluadas. |
| Consistencia visual | La disposición de los elementos mantiene una organización comprensible. |

**Tabla 5**. Criterios de aprobación para la evaluación de responsividad.
Fuente: Elaboración propia, basada en [4]

Se consideró satisfactorio el comportamiento responsivo de una vista cuando cumplió los criterios establecidos en las tres resoluciones evaluadas. Los problemas detectados se registraron como limitaciones de la interfaz, permitiendo identificar los componentes que podrían requerir ajustes posteriores. Debido a que las evaluaciones se realizaron mediante emulación de pantalla, estos resultados no constituyeron una certificación de accesibilidad ni garantizaron el funcionamiento en todos los dispositivos físicos.

- **Criterios de aprobación de *Lighthouse***

Para las auditorías automatizadas se utilizaron como referencia los rangos de clasificación *de Google Lighthouse*. De acuerdo con su documentación oficial, los puntajes entre 0 y 49 se consideran deficientes, entre 50 y 89 requieren mejoras y entre 90 y 100 se clasifican como buenos [15].

| Categoría | Umbral de aprobación |
|---|---|
| Rendimiento (*Performance*) | ≥ 90 puntos |
| Accesibilidad (*Accessibility*) | ≥ 90 puntos |
| Buenas prácticas (*Best Practices*) | ≥ 90 puntos |
| Optimización para motores de búsqueda (*SEO*) | ≥ 90 puntos |

**Tabla 6**. Umbrales de aprobación establecidos para las auditorías de Google Lighthouse
Fuente: Elaboración Propia, basada en [15]

La adopción de estos umbrales permitió evaluar las vistas de Amukan utilizando una referencia uniforme de calidad técnica. Los puntajes iguales o superiores a 90 se consideraron satisfactorios, mientras que los resultados inferiores se interpretaron de acuerdo con los rangos oficiales de *Lighthouse*.

Se reconoció que las vistas con resultados inferiores al umbral establecido presentan oportunidades de mejora, especialmente en aquellas categorías afectadas por la carga de recursos, la complejidad de los componentes o las características particulares de cada módulo. No obstante, un puntaje bajo no implica necesariamente la ausencia de funcionalidad, sino que identifica aspectos técnicos que requieren revisión u optimización.

Finalmente, el cumplimiento de los criterios se determinó mediante la mediana de tres ejecuciones independientes por vista y categoría, con el propósito de reducir la influencia de las variaciones entre mediciones.

## 5.2 Evaluación de Responsividad

La evaluación de responsividad se orientó a verificar la adaptación de las principales vistas del *frontend* de Amukan a diferentes dimensiones de pantalla, considerando los criterios de aprobación establecidos en la sección 5.1.1. Este procedimiento permitió examinar el comportamiento de los componentes implementados y determinar si conservaron una distribución adecuada y sus elementos interactivos accesibles.

Para esta evaluación se seleccionaron las vistas registro, mapa, mi comercio y *dashboard*, debido a que presentan distintas estructuras y componentes dentro de la plataforma. Su selección permitió considerar formularios, elementos de navegación, paneles de información y componentes de visualización, proporcionando una perspectiva representativa de la implementación responsiva del *frontend*.

Las verificaciones consideraron tres dimensiones de pantalla: móvil, tablet y escritorio, de acuerdo con los parámetros definidos en la Tabla 3. En cada caso se examinó la organización de los elementos, la visibilidad de los controles, la legibilidad del contenido y la presencia de posibles desbordamientos horizontales o superposiciones. Asimismo, se consideró la disponibilidad de las interacciones principales de las interfaces evaluadas.

Los resultados se documentaron mediante capturas de pantalla y observaciones del comportamiento visual de cada vista. Estas evidencias permitieron contrastar la implementación con los criterios de aprobación establecidos, identificando tanto los aspectos satisfactorios como las posibles limitaciones de adaptación en las dimensiones evaluadas.

### 5.2.1 Vista Registro

La evaluación de responsividad de la vista registro se realizó considerando las tres dimensiones de pantalla mencionadas anteriormente. El propósito fue verificar la adaptación de los componentes del formulario y su correspondencia con los criterios de aprobación establecidos en la sección 5.1.1.

En las tres resoluciones se observó que el formulario mantuvo una disposición centrada, conservando la visibilidad de los campos de ingreso, los botones y los enlaces complementarios. Asimismo, no se identificaron superposiciones ni recortes evidentes de los componentes en las capturas obtenidas.

En la resolución móvil, se comprobó adicionalmente la navegación desde la primera etapa del registro hacia la segunda, verificando la presentación de los campos de correo electrónico, contraseña y confirmación, junto con sus respectivos controles de navegación.

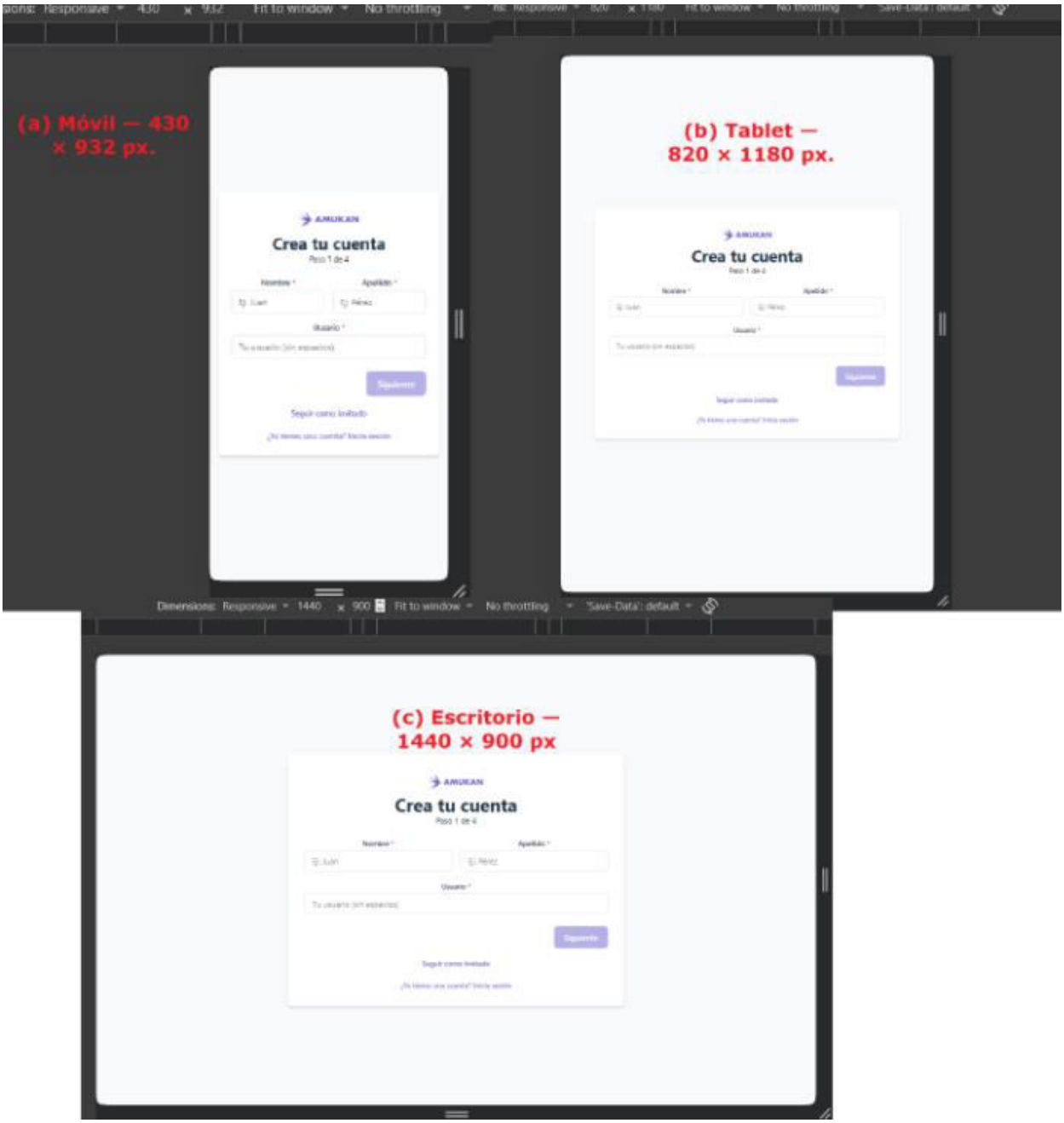



**Figura 28.** Evaluación de Responsividad de la Vista Registro
Fuente: Elaboración propia

En la resolución móvil, los campos del formulario conservaron una distribución horizontal en determinados grupos de entrada. Esta disposición se evaluó considerando el espacio disponible, la legibilidad de las etiquetas y la ausencia de desbordamientos o superposiciones, sin establecer la organización vertical como un requisito obligatorio de responsividad.

| Criterio | Resultado observado |
|---|---|
| Adaptación de componentes | El formulario conserva su distribución sin superposiciones evidentes en las tres resoluciones. |
| Ausencia de desbordamiento horizontal | No se aprecia desbordamiento horizontal en las capturas. |
| Visibilidad de controles | Los campos, botones y enlaces principales permanecen visibles. |
| Legibilidad | No se observan textos cortados ni superpuestos. |
| Navegación e interacción | Se comprobó el avance del primer al segundo paso en móvil. Las demás interacciones no fueron evaluadas en estas capturas. |
| Consistencia de la interfaz | Se mantiene una organización similar en las tres dimensiones. |

**Tabla 7**. Resultados de la evaluación de la vista registro
Fuente: Elaboración propia

Los resultados obtenidos evidenciaron una adaptación visual satisfactoria de la vista registro en las tres dimensiones evaluadas, sin identificarse problemas evidentes de distribución, visibilidad o superposición de componentes. La navegación comprobada entre las dos primeras etapas del formulario complementó estos resultados, aunque la evaluación no contempló la totalidad del proceso de registro.

### 5.2.2 Vista Mapa

La evaluación de responsividad de la vista mapa se realizó en las resoluciones antes descritas, considerando la adaptación de sus componentes principales y la disponibilidad de los controles de navegación.

En las tres resoluciones evaluadas se observó la visualización del componente cartográfico y de los elementos principales de la interfaz, sin identificarse superposiciones o recortes evidentes que dificultaran su presentación.

En las resoluciones, la interfaz incorporó un mecanismo de ocultación y despliegue del panel lateral mediante un control identificado por una flecha morada en escritorio y tablet, mientras que en móvil, el control se presenta mediante una barra gris. Este comportamiento permitió modificar la visibilidad del panel y disponer de mayor espacio

para la visualización del mapa. En escritorio, la distribución aprovechó el ancho disponible para presentar simultáneamente el mapa y los elementos complementarios.

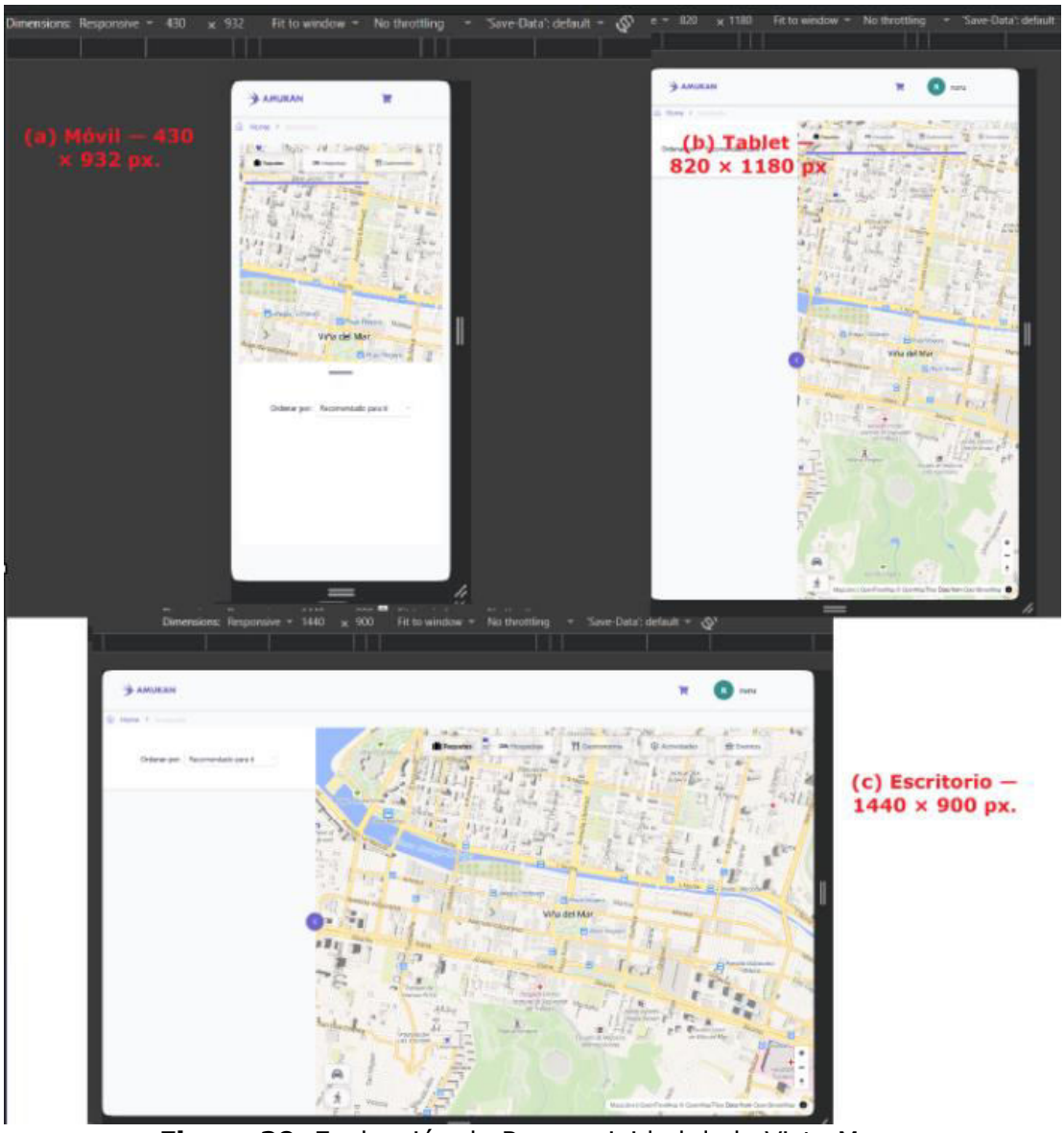



**Figura 29.** Evaluación de Responsividad de la Vista Mapa
Fuente: Elaboración Propia

| Criterio | Resultado observado |
|---|---|
| Adaptación de componentes | El mapa y los elementos principales se presentan en las tres resoluciones evaluadas. |
| Ausencia de desbordamiento horizontal | No se aprecian desbordamientos horizontales generales en las capturas. |
| Visibilidad de controles | Los controles principales permanecen visibles y se dispone de un mecanismo para desplegar el panel lateral en móvil y tablet. |
| Legibilidad | Los elementos visibles conservan una presentación legible en las resoluciones evaluadas. |
| Navegación e interacción | El panel lateral dispone de un control para alternar entre sus estados visible y oculto en móvil y tablet. |
| Consistencia visual | La distribución se adapta a las dimensiones de pantalla y conserva los elementos principales de la interfaz. |

**Tabla 8.** Resultados de la Evaluación de Responsividad de la Vista Mapa
Fuente: Elaboración propia

Los resultados de la evaluación evidenciaron una adaptación responsiva satisfactoria de la vista mapa en las tres resoluciones consideradas. Se destacó la implementación de un panel lateral desplegable, cuyo comportamiento permitió ajustar la distribución de los componentes según el espacio disponible. No se identificaron problemas evidentes de superposición o desbordamiento en las capturas analizadas.

### 5.2.3 Vista Mi Comercio

La evaluación de responsividad de la vista mi comercio se centró en la adaptación de sus componentes principales, considerando la distribución del contenido, la navegación lateral y la visibilidad de las opciones de gestión.

En la versión móvil, se observa que la barra lateral permaneció oculta, incorporando un control de acceso al menú para aprovechar el espacio disponible. En tablet y escritorio, la navegación lateral se presentó de forma permanente, manteniendo visible el área principal destinada a las publicaciones.

Asimismo, las opciones para crear eventos, actividades y lugares permanecieron disponibles en las tres resoluciones evaluadas, mediante un menú asociado al botón de creación. No se identificaron superposiciones ni desbordamientos horizontales evidentes en las capturas obtenidas.

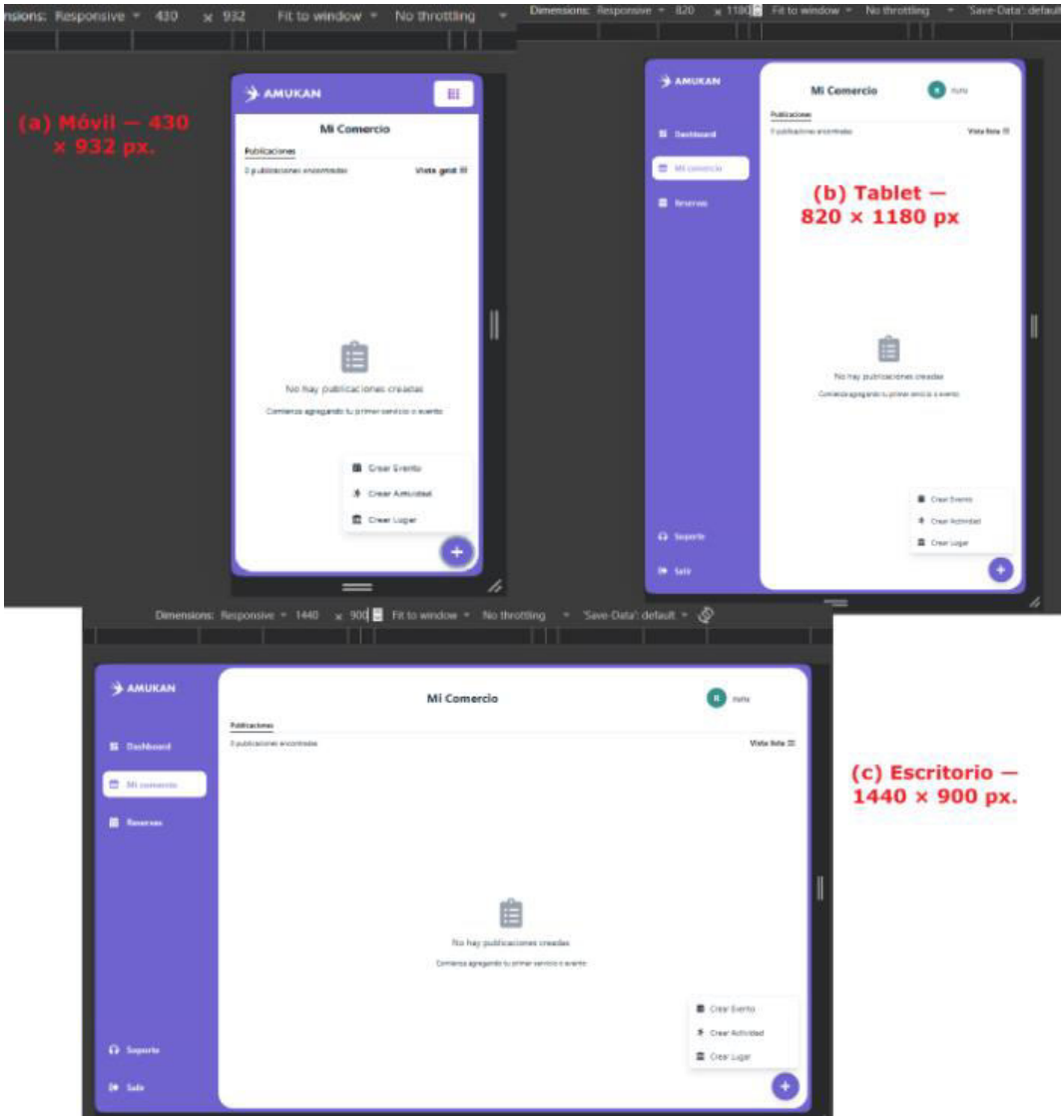



**Figura 30.** Evaluación de Responsividad de la Vista Mi Comercio
Fuente: Elaboración Propia

| Criterio | Resultado observado |
|---|---|
| Adaptación de componentes | El contenido principal se ajusta al espacio disponible en las tres resoluciones. |
| Ausencia de desbordamiento horizontal | No se observan desbordamientos horizontales evidentes. |
| Visibilidad de controles | Las opciones de gestión y creación permanecen visibles. |
| Legibilidad | Los textos y controles principales no presentan recortes evidentes. |
| Navegación e interacción | Se observa una navegación lateral adaptable y el menú de creación desplegado. |
| Consistencia visual | Se mantiene la estructura general de la interfaz, con variaciones en la disposición del menú lateral. |

**Tabla 9.** Resultados de la Evaluación de Responsividad de la Vista Mi Comercio
Fuente: Elaboración propia

Los resultados evidenciaron una adaptación responsiva satisfactoria de la vista mi comercio, destacando el comportamiento de la navegación lateral y la disponibilidad de los controles de gestión en las dimensiones evaluadas. No obstante, debido a que la vista no contaba con publicaciones registradas durante las pruebas, la evaluación se limitó a los componentes visibles y no permitió verificar la distribución de las tarjetas de contenido.

### 5.2.4 Panel Institucional

La evaluación de responsividad del panel institucional se centró en la adaptación de los componentes principales del *frontend*, considerando la navegación lateral, el encabezado y la distribución del contenedor destinado a la visualización de indicadores.

En la versión móvil, la barra lateral se presentó de forma colapsada, permitiendo aprovechar el espacio disponible para el contenido principal. En tablet y escritorio, la navegación lateral permaneció visible y el contenedor de información se ajustó a las dimensiones correspondientes.

En las tres resoluciones evaluadas se observó una distribución adecuada de la estructura general de la interfaz, sin identificarse superposiciones ni desbordamientos horizontales evidentes. Asimismo, los elementos principales de navegación conservaron su visibilidad o acceso según la disposición de cada pantalla.

La evaluación se concentró en la estructura responsiva implementada en el frontend, considerando el espacio destinado a la integración de las visualizaciones institucionales.

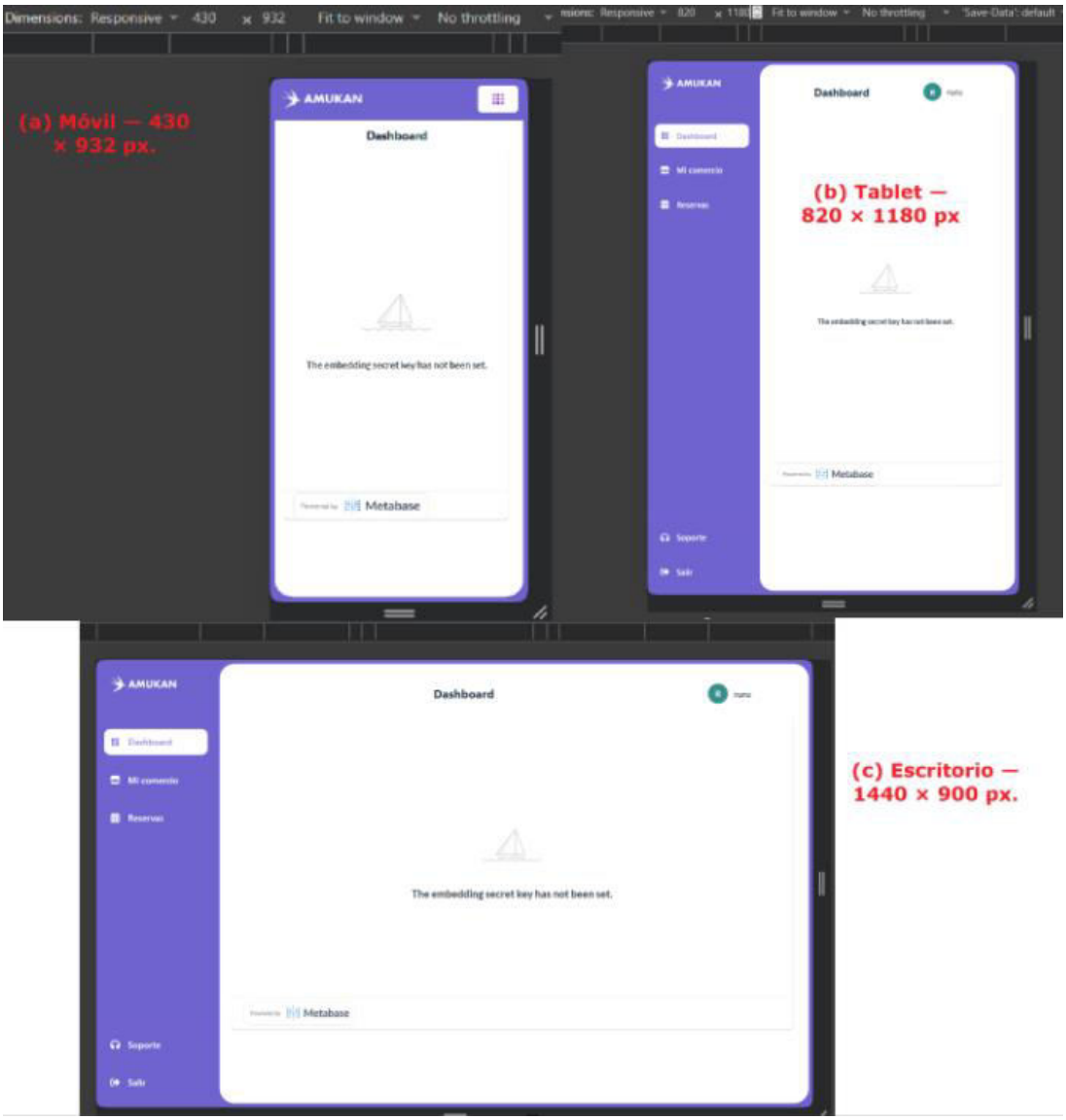



**Figura 31.** Evaluación de Responsividad de la Vista Institucional/*Dashboard*
Fuente: Elaboración Propia

| Criterio | Resultado observado |
|---|---|
| Adaptación de componentes | El encabezado, la navegación lateral y el contenedor principal se ajustan a las dimensiones evaluadas. |
| Ausencia de desbordamiento horizontal | No se observan desbordamientos horizontales evidentes en la estructura del *frontend*. |
| Visibilidad de controles | Los controles principales de navegación permanecen visibles o accesibles. |
| Legibilidad | Los textos y elementos de navegación visibles no presentan recortes evidentes. |
| Navegación e interacción | La navegación lateral presenta una disposición colapsada en móvil y permanece visible en tablet y escritorio. |
| Consistencia visual | La estructura general conserva una distribución coherente entre las tres resoluciones. |

**Tabla 10**. Resultados de la Evaluación de Responsividad del Panel Institucional
Fuente: Elaboración propia

Los resultados evidenciaron una adaptación responsiva satisfactoria de la estructura del frontend correspondiente al panel institucional. La disposición de la navegación lateral y del contenedor principal permitió mantener una organización adecuada en las dimensiones evaluadas, sin identificarse problemas evidentes de superposición o desbordamiento horizontal.

### 5.2.5 Síntesis de los Resultados de Responsividad

| Vista | Resultado observado | Alcance de la evaluación |
|---|---|---|
| Registro | Adaptación adecuada de formulario y controles. | Se comprobó adicionalmente el avance entre etapas en móvil. |
| Mapa | Visualización del mapa y adaptación del panel lateral. | Componentes principales visibles. |
| Mi Comercio | Navegación adaptable y controles de gestión disponibles. | Vista evaluada sin publicaciones registradas. |
| Panel Institucional | Adaptación del encabezado, menú y contenedor principal. | Estructura frontend visible |

**Tabla 11**. Resumen de la Evaluación Responsiva
Fuente: Elaboración propia

Los resultados de las pruebas de responsividad permitieron evaluar la adaptación de las cuatro vistas seleccionadas, considerando los criterios definidos en la sección 5.1.1. En las dimensiones examinadas, las interfaces presentaron una distribución adecuada de sus componentes principales, sin identificarse desbordamientos horizontales ni superposiciones evidentes en las capturas obtenidas.

Entre los comportamientos observados destacaron la adaptación de los formularios y los mecanismos de navegación lateral colapsable, que permitieron ajustar la presentación de los contenidos según el espacio disponible.

Las evaluaciones de mi comercio y del panel institucional se realizaron considerando los componentes visibles durante las pruebas, por lo que los resultados no abarcaron todos los posibles estados de contenido de estas interfaces.

En conjunto, las evidencias obtenidas mostraron una adaptación responsiva satisfactoria de los componentes evaluados, dentro de las resoluciones y condiciones establecidas para las pruebas.

## 5.3 Evaluación Técnica Mediante Google Lighthouse

La evaluación técnica del *frontend* de Amukan se planteó mediante auditorías automatizadas con *Google Lighthouse*, con el propósito de analizar el comportamiento de las principales vistas de la plataforma en términos de rendimiento (*Performance*), accesibilidad (*Accessibility*), buenas prácticas (*Best Practices*) y optimización para motores de búsqueda (*SEO*).

Para esta evaluación se consideraron las vistas registro, mapa, mi comercio y panel institucional, las cuales representan diferentes estructuras y componentes implementados en el *frontend*. Las auditorías se definieron sobre una compilación optimizada de la aplicación, ejecutada en un entorno local mediante la función de previsualización de *Vite*, bajo los parámetros establecidos en la sección 5.1.

Con el fin de reducir la influencia de las variaciones entre mediciones, se establecieron tres ejecuciones independientes por vista. Para cada categoría se utilizó la mediana de los puntajes obtenidos, permitiendo disponer de un valor representativo para su comparación con los criterios de aprobación definidos en la sección 5.1.1.

Los resultados se analizarán considerando el umbral de 90 puntos establecido para las cuatro categorías evaluadas, identificando aquellas que alcanzaron el criterio definido y aquellas que presentaron oportunidades de mejora. Asimismo, se considerarán las limitaciones observadas durante las auditorías para interpretar adecuadamente el comportamiento técnico de las interfaces.

### 5.3.1 Resultados de la vista Registro con Lighthouse

Se realizaron tres auditorías de la vista Registro mediante Google Lighthouse, utilizando las condiciones definidas en la sección 5.1. En la siguiente figura se presentan los puntajes obtenidos en cada ejecución, mientras que la Tabla 12 resume los resultados de las cuatro categorías evaluadas y sus respectivas medianas.

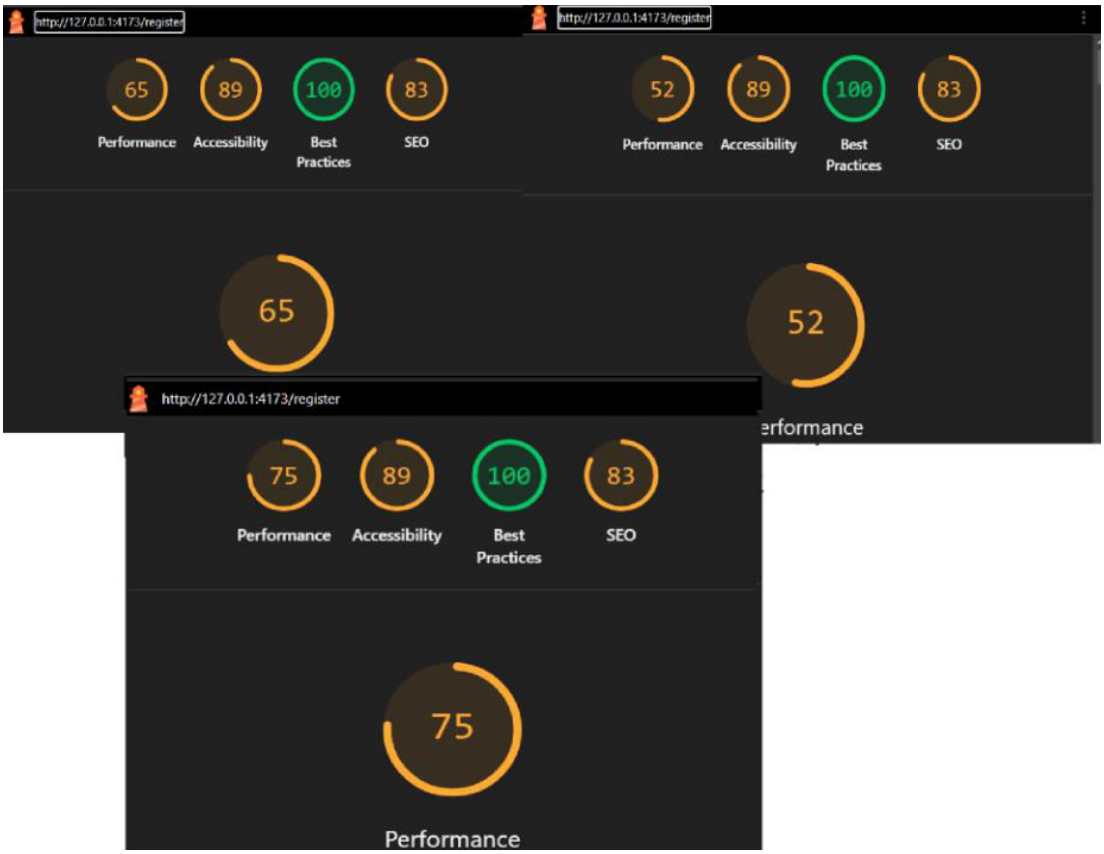



**Figura 32.** Resultados de las Tres Auditorías *Lighthouse* de la Vista Registro
Fuente: Elaboración Propia

| Categoría | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana | Cumplimiento |
|---|---|---|---|---|---|
| Performance | 52 | 65 | 75 | **65** | No cumple |
| Accessibility | 89 | 89 | 89 | **89** | No cumple |
| Best Practices | 100 | 100 | 100 | **100** | Cumple |
| SEO | 83 | 83 | 83 | **83** | No cumple |

**Tabla 12**. Resultados Generales de *Lighthouse* en la Vista Registro
Fuente: Elaboración propia

Para complementar la evaluación del rendimiento, se registraron las métricas *First Contentful Paint* (FCP), *Largest Contentful Paint* (LCP), *Total Blocking Time* (TBT), *Cumulative Layout Shift* (CLS) y *Speed Index* (SI), cuyos resultados se presentan en la siguiente tabla.

| Métrica | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana |
|---|---|---|---|---|
| FCP | 3,1 s | 2,9 s | 2,2 s | **2,9 s** |
| LCP | 3,8 s | 3,4 s | 2,5 s | **3,4 s** |
| TBT | 2100 ms | 820 ms | 850 ms | **850 ms** |
| CLS | 0,001 | 0 | 0 | **0** |
| Speed Index | 4,8 s | 4,0 s | 2,6 s | **4,0 s** |

**Tabla 13**. Métricas de Rendimiento de *Lighthouse* en la Vista Registro
Fuente: Elaboración propia

Los resultados evidenciaron variaciones en la categoría de rendimiento, cuya mediana alcanzó 65 puntos. En contraste, las categorías de accesibilidad, buenas prácticas y *SEO* mantuvieron puntuaciones constantes entre las ejecuciones, con medianas de 89, 100 y 83 puntos, respectivamente.

Al comparar estos valores con el umbral establecido en la sección 5.1.1, se determinó que únicamente la categoría de buenas prácticas cumplió el criterio de aprobación. Las restantes categorías presentaron valores inferiores a 90 puntos.

Respecto de las métricas de rendimiento, destacó la variación del *TBT* entre las ejecuciones, con valores comprendidos entre 820 y 2100 milisegundos. Asimismo, la mediana de *LCP* alcanzó 3,4 segundos, evidenciando oportunidades de optimización en el proceso de carga y procesamiento de la vista. Por otra parte, el *CLS* presentó una mediana de 0, lo que indicó estabilidad visual durante las mediciones.

### 5.3.2 Resultado Vista Mapa con Lighthouse

Se realizaron tres auditorías de la vista mapa mediante *Google Lighthouse*, utilizando las condiciones definidas en la sección 5.1. La Figura 33 presenta los puntajes obtenidos en cada ejecución, mientras que la Tabla 14 resume los resultados de las cuatro categorías evaluadas y sus respectivas medianas.

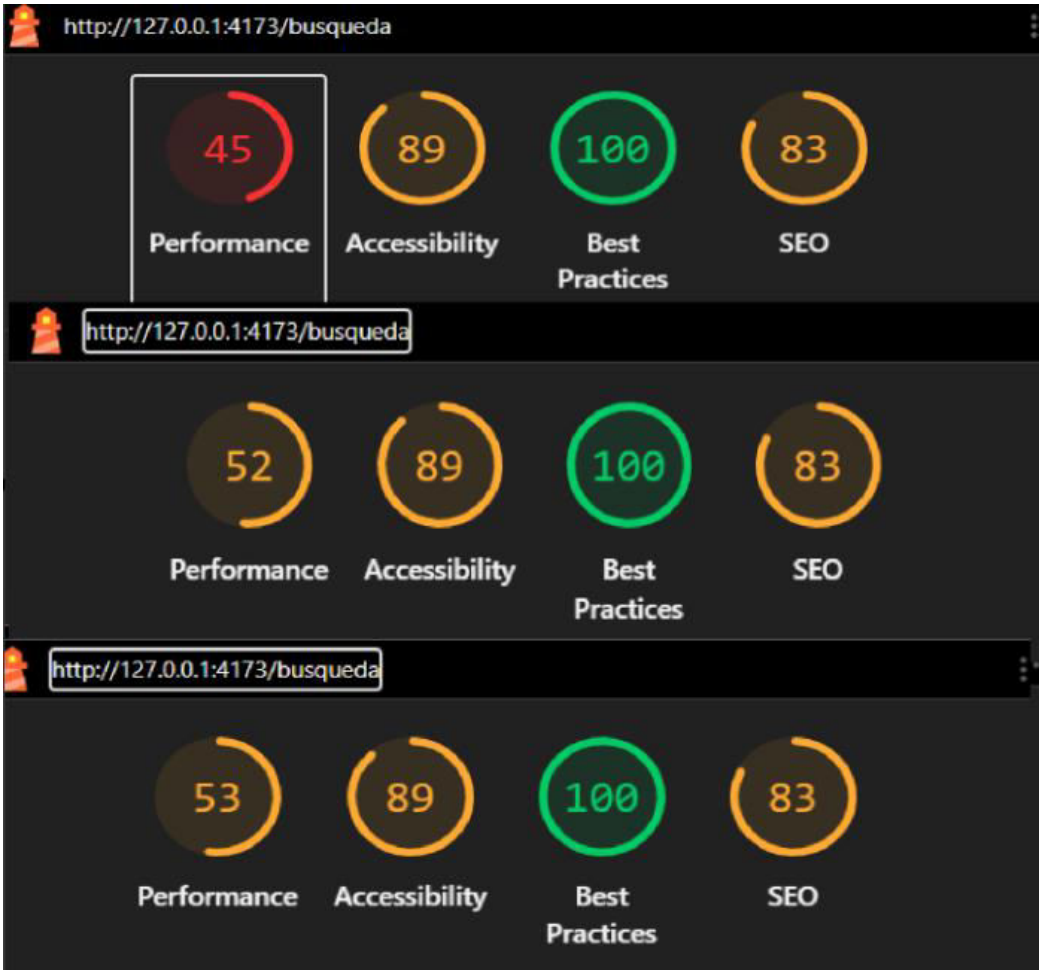



**Figura 33.** Resultados de las Tres Auditorías *Lighthouse* de la Vista Mapa
Fuente: Elaboración Propia

| **Categoría** | **Ejecución 1** | **Ejecución 2** | **Ejecución 3** | **Mediana** | **Cumplimiento** |
|---|---|---|---|---|---|
| Performance | 45 | 52 | 53 | **52** | No cumple |
| Accessibility | 89 | 89 | 89 | **89** | No cumple |
| Best Practices | 100 | 100 | 100 | **100** | Cumple |
| SEO | 83 | 83 | 83 | **83** | No cumple |

**Tabla 14**. Resultados Generales de *Lighthouse* en la Vista Mapa
Fuente: Elaboración propia

Para complementar la evaluación del rendimiento, se registraron las métricas *First Contentful Paint* (FCP), *Largest Contentful Paint* (LCP), *Total Blocking Time* (TBT), *Cumulative Layout Shift* (CLS) y *Speed Index* (SI), cuyos resultados se presentan en la Tabla 15.

| Métrica | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana |
|---|---|---|---|---|
| FCP | 3,4 s | 2,7 s | 2,6 s | **2,7 s** |
| LCP | 3,7 s | 3,0 s | 3,1 s | **3,1 s** |
| TBT | 6370 ms | 5670 ms | 5330 ms | **5670 ms** |
| CLS | 0,064 | 0,064 | 0,063 | **0,064** |
| Speed Index | 8,3 s | 7,9 s | 6,9 s | **7,9 s** |

**Tabla 15**. Métricas de Rendimiento de *Lighthouse* en la Vista Mapa
Fuente: Elaboración propia

Los resultados evidenciaron que la vista Mapa presentó una mediana de 52 puntos en la categoría de rendimiento, constituyendo el valor más bajo entre las categorías evaluadas. En contraste, las categorías de accesibilidad, buenas prácticas y *SEO* alcanzaron medianas de 89, 100 y 83 puntos, respectivamente.

Al contrastar estos resultados con el umbral de aprobación definido en la sección 5.1.1, se determinó que únicamente la categoría de buenas prácticas cumplió el criterio establecido. Las restantes categorías se mantuvieron bajo los 90 puntos requeridos.

Respecto de las métricas de rendimiento, destacó especialmente el *Total Blocking Time* (*TBT*), cuya mediana alcanzó 5670 milisegundos. Asimismo, el *Largest Contentful Paint* (*LCP*) presentó una mediana de 3,1 segundos y el *Speed Index* una mediana de 7,9 segundos. Estos valores evidenciaron oportunidades de mejora en el tiempo de carga y en el procesamiento de recursos de la vista.

Por otra parte, el *Cumulative Layout Shif*t (*CLS*) presentó una mediana de 0,064, lo que indicó una variación visual reducida durante la carga de la interfaz. En conjunto, estos resultados mostraron que la vista mapa mantuvo una adecuada estabilidad visual, aunque con limitaciones de rendimiento más marcadas que las observadas en la vista registro.

### 5.3.3 Resultados de la Vista Mi Comercio con Lighthouse

Se realizaron tres auditorías de la vista mi comercio mediante *Google Lighthouse*, utilizando las condiciones establecidas en la sección 5.1. La Figura 34 presenta los puntajes obtenidos en cada ejecución, mientras que la Tabla 16 resume los resultados de las cuatro categorías evaluadas y sus respectivas medianas.

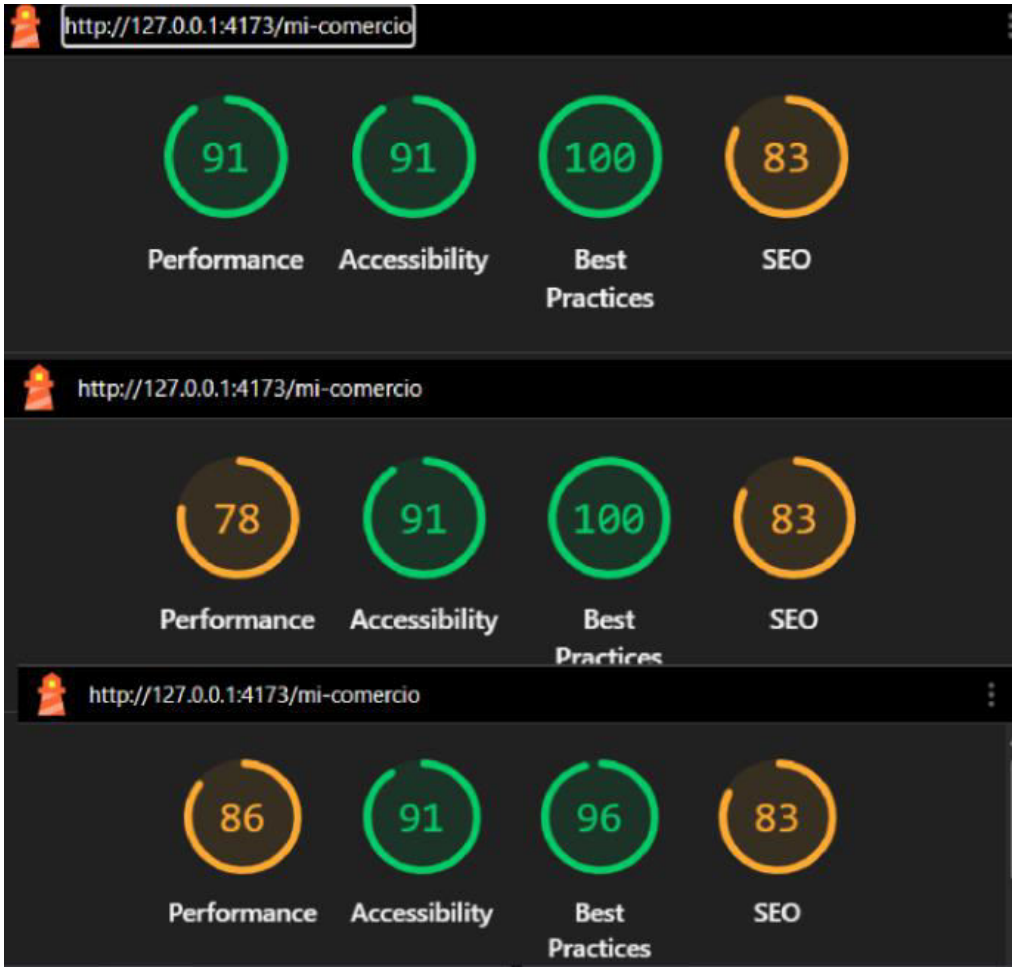



**Figura 34.** Resultados de las Tres Auditorías *Lighthouse* de la Vista Mi Comercio
Fuente: Elaboración propia

| Categoría | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana | Cumplimiento |
|---|---|---|---|---|---|
| Performance | 91 | 78 | 86 | **86** | No cumple |
| Accessibility | 91 | 91 | 91 | **91** | Cumple |
| Best Practices | 100 | 100 | 96 | **100** | Cumple |
| SEO | 83 | 83 | 83 | **83** | No cumple |

**Tabla 16.** Resultados Generales de *Lighthouse* en la Vista Mi Comercio
Fuente: Elaboración propia

Para complementar la evaluación del rendimiento, se registraron las métricas *First Contentful Paint* (*FCP*), *Largest Contentful Paint* (*LCP*), *Total Blocking Time* (*TBT*), *Cumulative Layout Shift* (*CLS*) y *Speed Index* (*SI*), cuyos resultados se presentan en la Tabla 17.

| Métrica | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana |
|---|---|---|---|---|
| FCP | 2,3 s | 2,8 s | 2,6 s | **2,6 s** |
| LCP | 2,8 s | 3,1 s | 3,0 s | **3,0 s** |
| TBT | 130 ms | 390 ms | 230 ms | **230 ms** |
| CLS | 0,049 | 0,049 | 0,049 | **0,049** |
| Speed Index | 2,3 s | 3,6 s | 3,0 s | **3,0 s** |

**Tabla 17.** Métricas de Rendimiento de *Lighthouse* en la Vista Mi Comercio
Fuente: Elaboración propia

Los resultados evidenciaron que la vista mi comercio obtuvo una mediana de 86 puntos en rendimiento, 91 en accesibilidad, 100 en buenas prácticas y 83 en *SEO*. Al contrastar estos valores con los criterios establecidos en la sección 5.1.1, se determinó que las categorías de accesibilidad y buenas prácticas alcanzaron el umbral de aprobación, mientras que rendimiento y *SEO* presentaron puntuaciones inferiores a 90 puntos.

En relación con las métricas de rendimiento, el *TBT* presentó una mediana de 230 milisegundos, mientras que el *FCP* y el *LCP* alcanzaron 2,6 y 3,0 segundos, respectivamente. Por su parte, el *Speed Index* registró una mediana de 3,0 segundos y el *CLS* se mantuvo constante en 0,049 durante las tres ejecuciones, evidenciando una baja variación en la distribución visual de los elementos durante la carga.

En conjunto, estos resultados permitieron identificar un comportamiento técnico favorable en accesibilidad y buenas prácticas, junto con oportunidades de optimización en rendimiento y *SEO*. Cabe señalar que las auditorías se realizaron con la vista sin publicaciones registradas, por lo que los resultados representan el comportamiento de la interfaz en dicho estado.

### 5.3.4 Resultados del Panel Institucional con Lighthouse

Se realizaron tres auditorías del panel institucional mediante *Google Lighthouse*, utilizando las condiciones establecidas en la sección 5.1. La Figura 35 presenta los puntajes obtenidos en cada ejecución, mientras que la Tabla 18 resume los resultados de las cuatro categorías evaluadas y sus respectivas medianas.

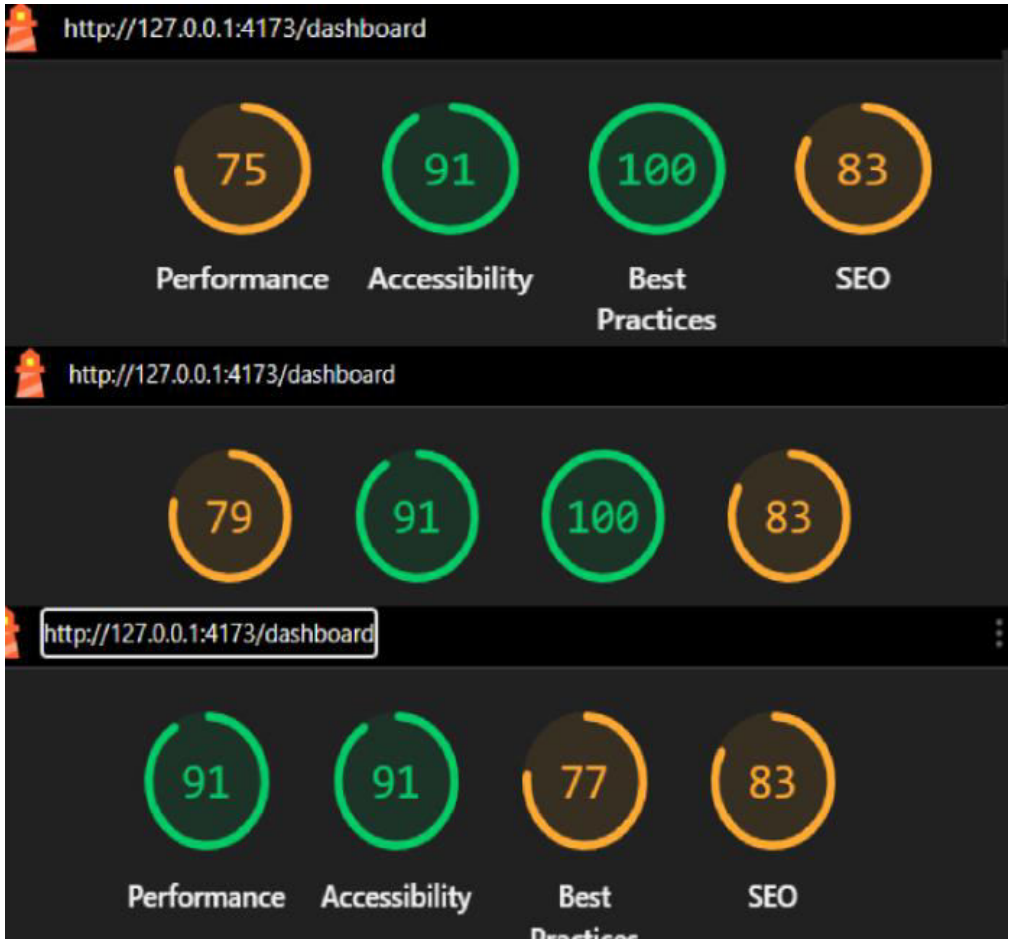



**Figura 35.** Resultados de las Tres Auditorías *Lighthouse* del Panel Institucional
Fuente: Elaboración propia

| Categoría | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana | Cumplimiento |
|---|---|---|---|---|---|
| Performance | 75 | 79 | 91 | **79** | No cumple |
| Accessibility | 91 | 91 | 91 | **91** | Cumple |
| Best Practices | 100 | 100 | 77 | **100** | Cumple |
| SEO | 83 | 83 | 83 | **83** | No cumple |

**Tabla 18.** Resultados Generales de *Lighthouse* en el Panel Institucional
Fuente: Elaboración propia

Para complementar la evaluación de rendimiento, se registraron las métricas First Contentful Paint (FCP), Largest Contentful Paint (LCP), Total Blocking Time (TBT), Cumulative Layout Shift (CLS) y Speed Index (SI), cuyos resultados se presentan en la Tabla 19.

| Métrica | Ejecución 1 | Ejecución 2 | Ejecución 3 | Mediana |
|---|---|---|---|---|
| FCP | 2,7 s | 2,6 s | 2,3 s | **2,6 s** |
| LCP | 2,9 s | 3,1 s | 2,7 s | **2,9 s** |
| TBT | 600 ms | 370 ms | 180 ms | **370 ms** |
| CLS | 0,001 | 0,001 | 0,001 | **0,001** |
| Speed Index | 4,0 s | 4,2 s | 2,3 s | **4,0 s** |

**Tabla 19.** Métricas de rendimiento de Lighthouse en el Panel Institucional.
Fuente: Elaboración propia

Los resultados evidenciaron que el panel institucional obtuvo una mediana de 79 puntos en rendimiento, 91 en accesibilidad, 100 en buenas prácticas y 83 en *SEO*. Al comparar estos valores con los criterios establecidos en la sección 5.1.1, se determinó que las categorías de accesibilidad y buenas prácticas alcanzaron el umbral de aprobación, mientras que rendimiento y *SEO* presentaron puntuaciones inferiores a 90 puntos.

En las métricas de rendimiento, el *TBT* registró una mediana de 370 milisegundos, mientras que el *FCP* y el *LCP* alcanzaron 2,6 y 2,9 segundos, respectivamente. El *Speed Index* presentó una mediana de 4,0 segundos y el *CLS* se mantuvo constante en 0,001, indicando una baja variación en la distribución visual de los elementos durante la carga.

Se observó una variación en la categoría de buenas prácticas, que registró 77 puntos en la tercera ejecución, frente a los 100 puntos obtenidos en las dos anteriores. Aunque la mediana cumplió el criterio establecido, esta diferencia evidenció la necesidad de considerar la variabilidad de los resultados en su interpretación.

Las auditorías se realizaron sobre la estructura visible del panel institucional, sin contar con las visualizaciones embebidas de indicadores durante las mediciones. Por consiguiente, los resultados caracterizan el comportamiento técnico de la vista bajo las condiciones evaluadas y no necesariamente el que presentaría con todos sus contenidos cargados.

### 5.3.5 Síntesis de los Resultados de Lighthouse

La comparación de las medianas obtenidas mediante *Google Lighthouse* permitió identificar diferencias en el comportamiento técnico de las cuatro vistas evaluadas. La Tabla 20 presenta los resultados consolidados de rendimiento, accesibilidad, buenas prácticas y *SEO*, considerando el criterio de aprobación establecido en la sección 5.1.1.

| Vista | Performance | Accessibility | Best Practices | SEO |
|---|---|---|---|---|
| Registro | 65 | 89 | 100 | 83 |
| Mapa | 52 | 89 | 100 | 83 |
| Mi Comercio | 86 | 91 | 100 | 83 |
| Panel Institucional | 79 | 91 | 100 | 83 |

**Tabla 20**. Resultados Consolidados de *Lighthouse*
Fuente: Elaboración propia

En la categoría de rendimiento, ninguna de las vistas alcanzó el umbral de 90 puntos. mi comercio obtuvo la mayor mediana, con 86 puntos, seguida del panel institucional con 79, registro con 65 y mapa con 52. Este último presentó además el mayor tiempo

de bloqueo del hilo principal (*TBT*), con una mediana de 5670 milisegundos, constituyendo una de las principales oportunidades de optimización identificadas.

Respecto de la accesibilidad, mi comercio y el panel institucional alcanzaron 91 puntos, cumpliendo el criterio establecido. registro y mapa obtuvieron 89 puntos, situándose ligeramente por debajo del umbral definido. Estos resultados permitieron identificar un comportamiento relativamente uniforme en esta categoría, aunque con aspectos pendientes de mejora.

En buenas prácticas, las cuatro vistas registraron una mediana de 100 puntos, alcanzando el criterio de aprobación. Sin embargo, se observaron variaciones entre algunas ejecuciones individuales, lo que reafirmó la importancia de utilizar una medida representativa y mantener condiciones uniformes de evaluación.

En cuanto al *SEO*, todas las vistas obtuvieron una mediana de 83 puntos, sin alcanzar los 90 puntos establecidos. La coincidencia de los resultados señaló la conveniencia de revisar los hallazgos específicos de las auditorías para identificar posibles oportunidades de mejora compartidas.

En términos generales, las auditorías evidenciaron resultados favorables en buenas prácticas y un cumplimiento parcial de los criterios de accesibilidad. Por otra parte, rendimiento y *SEO* presentaron oportunidades de mejora en las cuatro vistas evaluadas. Estos resultados corresponden a las condiciones de carga observadas durante las pruebas y constituyen antecedentes para la evaluación del cumplimiento de los objetivos del proyecto.

## 5.4 Evaluación del Cumplimiento de Objetivos

La evaluación del cumplimiento de los objetivos se realizó considerando las evidencias de implementación presentadas en los capítulos 3 y 4, junto con los resultados de las pruebas de responsividad y las auditorías de *Google Lighthouse* desarrolladas en el presente capítulo. Este análisis permitió contrastar los resultados obtenidos con los objetivos específicos establecidos en la sección 1.4.

Para los objetivos asociados con la implementación del *frontend* (**OE1** al **OE4**), se consideró como criterio la existencia de los módulos, componentes y mecanismos de interacción definidos en cada objetivo, respaldados por la arquitectura, los fragmentos de código y las interfaces documentadas. En el caso del **OE5**, se consideró la ejecución de las evaluaciones técnicas bajo las condiciones establecidas y el análisis de sus resultados frente a los criterios de aprobación definidos en la sección 5.1.1.

La siguiente tabla presenta la relación entre los objetivos específicos, sus criterios de evaluación, las evidencias documentadas y el cumplimiento determinado.

| Objetivo | Criterio de evaluación | Evidencia | Cumplimiento |
|---|---|---|---|
| **OE1** | Implementación de una arquitectura modular, gestión de estados, comunicación con la *API* y control de acceso por roles. | Arquitectura de componentes del capítulo 3 y mecanismos de autenticación e integración descritos en la sección 4.2. | Cumplido a nivel de implementación. |
| **OE2** | Implementación de interfaces de búsqueda y exploración turística, presentación de itinerarios y exportación a PDF. | Componentes y funcionalidades documentados en las secciones 4.6 y 4.7, incluyendo las evidencias de exportación del documento. | Cumplido a nivel de implementación. |
| **OE3** | Implementación de las interfaces de gestión comercial, incluyendo formularios para crear y editar servicios. | Componentes *CrearServicio.jsx* y *EditarServicio.jsx*, renderizado condicional y mecanismos de comunicación documentados en la sección 4.6. | Cumplido a nivel de implementación; sin verificación integral del flujo comercial. |
| **OE4** | Implementación del panel institucional y de los componentes destinados a integrar y presentar indicadores. | Estructura, componentes e integración documentados en la sección 4.5, junto con las evaluaciones técnicas de la interfaz. | Implementación documentada; visualización operativa no corroborada durante las pruebas. |
| **OE5** | Realización de pruebas de responsividad y auditorías técnicas, con resultados contrastados frente a criterios previamente definidos. | Resultados de responsividad de la sección 5.2 y doce auditorías Lighthouse *presentadas* en la sección 5.3. | Validación realizada; cumplimiento parcial de los criterios técnicos de calidad. |

**Tabla 21**. Evaluación del Cumplimiento de los Objetivos Específicos
Fuente: Elaboración propia

Respecto del **OE1**, la arquitectura desarrollada permitió organizar el *frontend* mediante componentes reutilizables y una estructura modular, incorporando mecanismos de gestión de estados, navegación, autenticación y comunicación con los servicios de la

plataforma. Las implementaciones descritas en los capítulos 3 y 4 respaldaron el cumplimiento de este objetivo desde la perspectiva de la construcción del *frontend*.

En relación con el **OE2**, se implementaron los módulos de exploración y planificación turística mediante interfaces destinadas a presentar servicios, actividades e itinerarios. Asimismo, se incorporó la funcionalidad de exportación a PDF, cuya implementación y resultado generado se documentaron en la sección 4.7. Estas evidencias permitieron acreditar el desarrollo de los componentes contemplados en el objetivo.

En relación con el **OE3**, se implementó el módulo de gestión para negocios locales, incorporando una interfaz de visualización de publicaciones y formularios destinados a la creación y edición de servicios turísticos. Las evidencias presentadas en el capítulo 4 permitieron constatar la implementación de tarjetas informativas, controles de gestión y campos condicionales según la categoría del servicio, respaldando el desarrollo de las interfaces contempladas en este objetivo.

En cuanto al **OE4**, se implementó el panel institucional, incorporando componentes y espacios destinados a integrar las visualizaciones de indicadores. La estructura y los mecanismos de integración se documentaron en el capítulo 4. Durante las evaluaciones del capítulo 5 se verificó la adaptación responsiva de la interfaz y se obtuvo información sobre su comportamiento técnico, aunque no fue posible corroborar la visualización efectiva de los indicadores embebidos durante dichas pruebas.

Respecto del **OE5**, se completó la evaluación de responsividad de las cuatro vistas seleccionadas y se realizaron doce auditorías mediante *Google Lighthouse.* Las pruebas de responsividad evidenciaron una adaptación satisfactoria de los componentes observados en las dimensiones consideradas. Por su parte, las auditorías mostraron que las cuatro vistas alcanzaron el criterio de buenas prácticas y dos cumplieron el umbral de accesibilidad, mientras que ninguna obtuvo una mediana igual o superior a 90 puntos en rendimiento y *SEO*. Por consiguiente, se cumplió el procedimiento de validación técnica establecido, aunque los resultados evidenciaron un cumplimiento parcial de los criterios de calidad definidos.

En conjunto, las evidencias permitieron constatar el desarrollo de una plataforma frontend modular y responsiva, con interfaces destinadas a la exploración turística, planificación de itinerarios, gestión comercial y visualización institucional. De esta manera, se alcanzó el propósito de implementación establecido en el objetivo general, dentro del alcance del trabajo desarrollado. No obstante, las evaluaciones técnicas identificaron limitaciones de rendimiento, accesibilidad y *SEO*, así como aspectos de integración que requieren verificaciones complementarias. Estos resultados constituyen una base para las conclusiones y oportunidades de mejora presentadas en el capítulo siguiente.

# 6 Conclusiones y Trabajo Futuro

El desarrollo del *frontend* responsivo de Amukan permitió implementar una interfaz web orientada a la presentación centralizada de información turística de Viña del Mar, incorporando módulos de exploración, planificación de itinerarios y gestión de servicios. Estas funcionalidades fueron desarrolladas mediante una arquitectura modular basada en *React, Vite y Tailwind CSS*, cuya implementación se documentó en los capítulos anteriores. Asimismo, las pruebas de responsividad permitieron verificar la adaptación de los componentes principales a diferentes dimensiones de pantalla, en concordancia con los objetivos técnicos establecidos para el proyecto. De esta manera, se alcanzó el propósito de implementación establecido en el objetivo general, considerando las limitaciones identificadas durante la validación.

La evaluación de responsividad permitió verificar la adaptación de las vistas registro, mapa, mi comercio y panel institucional en resoluciones representativas de dispositivos móviles, tablets y computadores. Los resultados evidenciaron una distribución adecuada de los componentes observados, sin identificarse desbordamientos horizontales ni superposiciones evidentes. Asimismo, se comprobó la presencia de mecanismos de adaptación, como la navegación lateral colapsable, que contribuyeron a mantener disponibles los elementos principales de las interfaces en diferentes dimensiones de pantalla.

Por otra parte, las auditorías realizadas mediante *Google Lighthouse* permitieron caracterizar el comportamiento técnico del frontend en términos de rendimiento, accesibilidad, buenas prácticas y *SEO*. En rendimiento, las vistas registro, mapa, mi comercio y panel institucional obtuvieron medianas de 65, 52, 86 y 79 puntos, respectivamente, sin alcanzar el umbral de 90 puntos establecido. En accesibilidad, mi comercio y el panel institucional cumplieron el criterio definido, mientras registro y mapa alcanzaron 89 puntos. Por su parte, las cuatro vistas obtuvieron una mediana de 100 puntos en buenas prácticas y 83 puntos en *SEO*. Estos resultados evidenciaron fortalezas en buenas prácticas de desarrollo, junto con oportunidades de mejora principalmente en rendimiento y optimización para motores de búsqueda.

Entre las contribuciones de la implementación destacan la construcción de interfaces de exploración turística, la presentación y organización de itinerarios, la incorporación de una funcionalidad de exportación a PDF, las herramientas de gestión para negocios locales y el panel institucional. Estas funcionalidades establecen una base tecnológica para facilitar el acceso a información turística y apoyar la planificación de actividades. En este sentido, se espera que la plataforma pueda contribuir a disminuir la dispersión de información y favorecer la visibilidad digital de los servicios turísticos locales; sin embargo, estos beneficios deberán evaluarse mediante estudios de uso y mediciones posteriores.

Como líneas de trabajo futuro, se propone continuar con la optimización del rendimiento del frontend, especialmente en la vista mapa, que presentó los mayores tiempos de bloqueo durante las auditorías. Asimismo, se plantea mejorar los aspectos de accesibilidad y *SEO* identificados por *Lighthouse*, además de ampliar las pruebas a dispositivos físicos y diferentes condiciones de uso. También se considera pertinente completar las verificaciones de integración y funcionamiento de los contenidos institucionales en un entorno de despliegue.

Adicionalmente, se propone evaluar la evolución de Amukan hacia *una Progressive Web App* (*PWA*), incorporando capacidades de funcionamiento sin conexión y

almacenamiento de recursos. Esta posibilidad constituye una ampliación respecto de la funcionalidad actual de exportación a PDF, la cual permite conservar un archivo del itinerario para su consulta posterior, pero no proporciona funcionamiento sin conexión a la plataforma. Como proyección adicional, se contempla explorar mecanismos de intercambio seguro y automatizado de información con instituciones municipales, así como la posibilidad de adaptar la arquitectura del sistema a otros destinos turísticos.

Finalmente, el desarrollo de este proyecto permitió aplicar conocimientos de Ingeniería en Informática relacionados con arquitectura de componentes, integración de servicios, desarrollo responsivo y optimización de aplicaciones web. La implementación iterativa y la evaluación mediante criterios técnicos proporcionaron una experiencia de desarrollo que permitió identificar tanto los resultados alcanzados como los aspectos que requieren mejoras. En conjunto, Amukan constituye una base de implementación frontend susceptible de optimización y ampliación en futuras etapas.

**Agradecimientos.** Deseo expresar mi más profundo agradecimiento a Dios por sostenerme y guiarme a lo largo de la carrera, especialmente durante este último año. A mi mamá, papá y hermano, por su motivación constante y apoyo incondicional. A mi pareja, por ser un gran apoyo académico y emocional durante estos dos años, brindando comprensión y confianza en mis momentos más difíciles. A todo el equipo Amukan, quienes hicieron posible este proyecto; todo el trayecto fue lo mejor de este año. A mis pastores, Cristian Valenzuela Ramírez y Johana Valdés Piucol, por su guía, el apoyo a mi familia y el cariño brindado. A mi tía Rossana Pérez Cornejo, por el apoyo incondicional. Y un agradecimiento especial a Benjamín Ramírez Ibarra, exalumno y amigo, cuyo apoyo académico comenzó en mi primer semestre y se convirtió en una guía invaluable.

# 7 Referencias

[1] Priceline Press Center. (2024, febrero 13). *New Priceline research finds average traveler spends two full work days to plan and book trips Priceline*. https://press.priceline.com/new-priceline-research-finds-average-traveler-spends-two-full-work-days-to-plan-and-book-trips/

[2] Municipalidad de Viña del Mar. (2025, junio). *Plan de desarrollo turístico de Viña del Mar 2025–2033 (PLADETUR)*. https://www.munivina.cl/wp-content/uploads/2025/10/Pladetur-Vina-del-Mar-2025-2033.pdf

[3] Amazon Web Services. (s. f.). *¿Cuál es la diferencia entre el front end y back end en el desarrollo de aplicaciones?* https://aws.amazon.com/es/compare/the-difference-between-frontend-and-backend/

[4] LePage, P., & Andrew, R. (2019, 12 de febrero). *Conceptos básicos del diseño web responsivo*. web.dev. https://web.dev/articles/responsive-web-design-basics?hl=es

[5] Kurtuldu, M. (s. f.). *Basics of UX. web.dev.* https://web.dev/articles/ux-basics?hl=es-419

[6] StatCounter Global Stats. (s. f.). *Desktop vs mobile vs tablet market share worldwide.* https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet

[7] MDN Web Docs. (s. f.). What is JavaScript? https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/What_is_JavaScript

[8] React. (s. f.). *Your First Component*. https://react.dev/learn/your-first-component

[9] Vite Team. (s. f.). *Why Vite.* https://vitejs.dev/guide/why

[10] Vite PWA. (s. f.). Getting Started. vite-plugin-pwa. https://vite-pwa-org.netlify.app/guide/

[11] Tailwind Labs. (s. f.). Utility-First Fundamentals. Tailwind CSS. https://tailwindcss.com/docs/utility-first

[12] MDN Web Docs. (s. f.). *Progressive web apps*. https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

[13] Google. (s. f.). Lighthouse overview. *Chrome for Developers.* https://developer.chrome.com/docs/lighthouse/overview

[14] van Riet, J., Malavolta, I., & Ghaleb, T. A. (2023). *Optimize along the way: An industrial case study on web performance. Journal of Systems and Software, 198, Article 111593.* https://www.sciencedirect.com/science/article/pii/S0164121222002692?via%3Dihub

[15] Google. (s. f.). *Lighthouse performance scoring*. Chrome for Developers. https://developer.chrome.com/docs/lighthouse/performance/performance-scoring

[16] web.dev. (s. f.). *Largest Contentful Paint (LCP)*. https://web.dev/articles/lcp?hl=es-419

[17] Chrome for Developers. (s. f.). *Speed Index*. https://developer.chrome.com/docs/lighthouse/performance/speed-index

[18] Chrome for Developers. (s. f.). *Total Blocking Time (TBT)*. https://developer.chrome.com/docs/lighthouse/performance/lighthouse-total-blocking-time

[19] web.dev. (s. f.). *Cumulative Layout Shift (CLS)*. https://web.dev/articles/cls