

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE SOFTWARE

Tema:

MIGRACIÓN DEL SISTEMA "SISDE" A ENTORNO WEB PARA LA EMPRESA ELÉCTRICA AMBATO REGIONAL CENTRO NORTE S.A.

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Software

ÁREA: Desarrollo de software

LÍNEA DE INVESTIGACIÓN: Software, Tecnologías de la información y Ciencia de datos

AUTOR: Joan Sebastian Salán Tamami

TUTOR: Ing. Félix Oscar Fernández Peña, PhD.

Ambato - Ecuador

julio - 2025

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: MIGRACIÓN DEL SISTEMA "SISDE" A ENTORNO WEB PARA LA EMPRESA ELÉCTRICA AMBATO REGIONAL CENTRO NORTE S.A., desarrollado bajo la modalidad Proyecto de Investigación por el señor Joan Sebastian Salán Tamami, estudiante de la Carrera de Software, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, julio 2025.

Firmado electrónicamente por:
FELIX OSCAR FERNANDEZ PENA
Validar únicamente con FirmaEC

Ing. Félix Oscar Fernández Peña, PhD.
TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: MIGRACIÓN DEL SISTEMA "SISDE" A ENTORNO WEB PARA LA EMPRESA ELÉCTRICA AMBATO REGIONAL CENTRO NORTE S.A. es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, julio 2025.



Joan Sebastian Salán Tamami

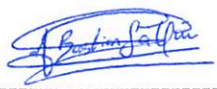
C.C. 1805465133

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, julio 2025.



Joan Sebastian Salán Tamami

C.C. 1805465133

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Joan Sebastian Salán Tamami, estudiante de la Carrera de Software, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado MIGRACIÓN DEL SISTEMA "SISDE" A ENTORNO WEB PARA LA EMPRESA ELÉCTRICA AMBATO REGIONAL CENTRO NORTE S.A., nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con el señor Presidente del Tribunal.

Ambato, julio 2025.

Firmado electrónicamente por:
FRANKLIN OSWALDO MAYORGA MAYORGA
Validar únicamente con FirmaEC

Ing. Franklin Oswaldo Mayorga Mayorga, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
DENNIS VINICIO CHICAIZA CASTILLO
Validar únicamente con FirmaEC

Ing. Dennis Chicaiza Castillo, Mg.
PROFESOR CALIFICADOR

Firmado electrónicamente por:
JULIO ENRIQUE BALAREZO LOPEZ
Validar únicamente con FirmaEC

Ing. Julio Balarezo López, PhD.
PROFESOR CALIFICADOR

DEDICATORIA

A mis padres, Fernanda y Marco, por el apoyo incondicional; Por el esfuerzo y el cariño que me han brindado a lo largo de este camino y ser mi soporte en todo momento.

A mi hermana, Sarahi, por su ayuda y compañía constante.

A mi amada, Angie, por animarme a cada instante, y ser la melodía que ilumina mis días y me impulsa a cumplir mis sueños.

A mis familiares y amigos, por sus palabras de aliento y por creer en mí cuando más lo necesitaba.

Joan Salán

AGRADECIMIENTO

A mis padres, Fernanda y Marco, por creer en mí y renunciar a sus sueños para permitir que los míos se hagan realidad, que ahora son los suyos también.

A mi amada, Angie, por su apoyo, compañía y motivación constante.

A mi tutor, Ing. Félix, por su guía académica, paciencia y dedicación en el desarrollo de este proyecto.

A mi tutor empresarial, Ing. Iván Vargas, por compartir sus conocimientos y experiencia, facilitando mi aprendizaje práctico.

A todos quienes contribuyeron de alguna manera a la realización de este proyecto.

Joan Salán

ÍNDICE GENERAL DE CONTENIDOS

PORTADA ..... **i**

APROBACIÓN DEL TUTOR ..... **ii**

AUTORÍA ..... **iii**

DERECHOS DE AUTOR ..... **iv**

APROBACIÓN DEL TRIBUNAL DE GRADO ..... **v**

DEDICATORIA ..... **vi**

AGRADECIMIENTO ..... **vii**

ÍNDICE GENERAL DE CONTENIDOS ..... **viii**

ÍNDICE DE TABLAS ..... **xi**

ÍNDICE DE FIGURAS ..... **xiii**

ÍNDICE DE ANEXOS ..... **xviii**

RESUMEN EJECUTIVO ..... **xix**

ABSTRACT ..... **xx**

CAPÍTULO I. MARCO TEÓRICO **1**

1.1 Tema de investigación 1

1.1.1 Planteamiento del problema 1

1.2 Antecedentes investigativos 2

1.3 Fundamentación teórica 6

1.3.1 Sistemas de Información Geográfica (GIS) 6

1.3.2 Tecnologías Frontend 10

1.3.3 Tecnologías Backend 14

1.3.4 Bases de Datos Geoespaciales 18

1.3.5 Seguridad en Sistemas Web GIS 22

1.3.6 Arquitecturas de Sistemas Web GIS 24

1.3.7 Metodologías de desarrollo 26

1.4 Objetivos 29

1.4.1 Objetivo general 29

1.4.2 Objetivos específicos 29

CAPÍTULO II. METODOLOGÍA 30

2.1 Materiales 30

2.2 Métodos 30

2.2.1 Modalidad de la investigación 30

2.2.2 Población y muestra 31

2.2.3 Recolección de información 32

2.2.4 Procesamiento y análisis de datos 35

CAPÍTULO III. RESULTADOS Y DISCUSIÓN 40

3.1 Análisis y Discusión de resultados 40

3.1.1 Análisis global de la solución propuesta 40

3.2 Desarrollo de la propuesta 44

3.2.1 Planificación de requisitos 44

3.2.2 Diseño y Feedback con el Usuario 53

3.2.3 Construcción del Backend 58

3.2.4 Construcción del Frontend 74

3.2.5 Transición 114

3.3 Resultados 118

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES 121

4.1 Conclusiones 121

4.2 Recomendaciones 122

REFERENCIAS BIBLIOGRÁFICAS 123

ANEXOS 129

ÍNDICE DE TABLAS

Tabla 1. Comparación de herramientas de Sistemas de Información Geográfica 9

Tabla 2. Comparación de tecnologías frontend..... 13

Tabla 3. Comparación de tecnologías backend..... 17

Tabla 4. Comparación de bases de datos geoespaciales 21

Tabla 5. Comparación de metodologías de desarrollo 26

Tabla 6. Escala de valores de puntuación SUS 32

Tabla 7. Resultados de la encuesta SUS en el sistema SISDE de escritorio..... 35

Tabla 8. Preguntas y respuestas de la entrevista aplicada al administrador del sistema SISDE de escritorio..... 36

Tabla 9. Responsables del desarrollo del proyecto de investigación 44

Tabla 10. Requerimientos técnicos de la EEASA..... 45

Tabla 11. Requerimientos funcionales SISDE Web 45

Tabla 12. Requerimientos no funcionales SISDE Web 46

Tabla 13. Plan de entrega por iteraciones 47

Tabla 14. Rediseño de tablas de la base de datos existente del sistema SISDE escritorio..... 50

Tabla 15. Propiedades del componente **`CustomAlert`** 82

Tabla 16. Propiedades de **`CustomTable`** 89

Tabla 17. Definición de las columnas en **`fieldDefinitions`** 90

Tabla 18. Propiedades de **`CustomComboBox`** 92

Tabla 19. Propiedades de **`CustomTabs`** 95

Tabla 20. Propiedades de **`tabsConfig`** 95

Tabla 21. Definición de las opciones del **`ActionBar`** 96

Tabla 22. Pruebas de aceptación iteración 1 114

Tabla 23. Pruebas de aceptación iteración 2 114

Tabla 24. Pruebas de aceptación iteración 3 115

Tabla 25. Pruebas de aceptación iteración 4 115

Tabla 26. Pruebas de aceptación iteración 5 116

Tabla 27. Resultados de la encuesta SUS del SISDE Web 119

Tabla 28. Comparación de resultados SUS en los sistemas SISDE de escritorio y web 120

ÍNDICE DE FIGURAS

Figura 1. Procesos del sistema SISDE de escritorio 41

Figura 2. Procesos propuestos del sistema SISDE Web 42

Figura 3. Comparación de las arquitecturas de los sistemas SISDE de escritorio y web 43

Figura 4. Arquitectura SISDE Web 49

Figura 5. Rediseño del extracto de tablas de la base de datos para la gestión de capas y elementos gráficos del mapa 52

Figura 6. Rediseño del extracto de tablas de la base de datos para la gestión de usuarios, roles y permisos 52

Figura 7. Interfaz pantalla inicial del sistema SISDE de escritorio 53

Figura 8. Prototipo funcional de Inicio de sesión SISDE Web 54

Figura 9. Prototipo funcional de pantalla de Inicio SISDE Web 54

Figura 10. Prototipo funcional de Gestión de Proyectos SISDE Web 55

Figura 11. Prototipo funcional de modal de edición SISDE Web 56

Figura 12. Prototipo funcional de pantalla de Actualización de Precios de mano de obra y materiales SISDE Web 57

Figura 13. Prototipo funcional del área de trabajo SISDE Web 57

Figura 14. Fragmento de código con la configuración de dependencias en **`pom.xml`** 60

Figura 15. Fragmento de código de la clase principal de la aplicación **`WSSisdeApplication.java`** 61

Figura 16. Fragmento de código del archivo de configuración **application.properties** 61

Figura 17. Fragmento de código con el bloqueo por múltiples intentos fallidos al iniciar sesión..... 62

Figura 18. Fragmento de código de la configuración de rutas privadas y públicas ... 63

Figura 19. Procedimiento Oracle **PROC_SD_FEATURE_CLASS_GET_LAYERS** en el paquete **PKG_SISTEMA** 64

Figura 20. Tabla **SD_FEATURE_CLASS** 64

Figura 21. Fragmento de código con el modelo **FeatureClass** 65

Figura 22. Fragmento de código para respuestas personalizadas con la clase **OperacionResponse** 65

Figura 23. Clase genérica **BaseStoredProcedureRepository** 66

Figura 24. Clase **FeatureClassRepository** en la ejecución del procedimiento **PROC_SD_FEATURE_CLASS_GET_LAYERS** sin parámetros 67

Figura 25. Clase **DisenioRepository** en la ejecución del procedimiento **PROC_SD_DISENIOS_BUSQUEDA** con parámetros 68

Figura 26. **RowMapper** personalizado en la captura de las propiedades de **DisenioEstado** 68

Figura 27. Fragmento de código de **DisenioService.java** 69

Figura 28. Fragmento de código del archivo **PermisoController.java** 71

Figura 29. Fragmento de código de **ResponseEntityUtil.java** 72

Figura 30. Endpoints de **permiso-controller** en Swagger 73

Figura 31. Respuesta del servidor backend al crear un permiso 74

Figura 32. Fragmento de código con las dependencias del frontend en el **`package.json`** 75

Figura 33. Archivo de configuración **`vite.config.js`** 76

Figura 34. Fragmento de código de **`index.jsx`** 77

Figura 35. Estructura de ficheros frontend 78

Figura 36. Archivo **`pageIcons.js`** 79

Figura 37. Fragmento de código **`api.js`** 79

Figura 38. Fragmento de código de **`layerService.js`** 80

Figura 39. Visualización de una alerta de estado correcto 81

Figura 40. Visualización de una alerta de advertencia 81

Figura 41. Visualización de una alerta de error 81

Figura 42. Modal de **`CustomTable`** para la creación de un nuevo registro 82

Figura 43. Fragmento de código para añadir una estructura 83

Figura 44. Modal de **`CustomTable`** para la edición de un registro 83

Figura 45. Fragmento de código para editar una estructura 84

Figura 46. Opción de eliminar en las opciones de las acciones 84

Figura 47. Fragmento de código para eliminar una estructura 85

Figura 48. Modal para crear una estructura 85

Figura 49. Fragmento de código **`structureFieldDefinitions`** 86

Figura 50. Verificación de campos obligatorios en la interfaz de usuario 86

Figura 51. Detalles de un registro 87

Figura 52. Barra de búsqueda en **`MaterialReactTable`** 87

Figura 53. Opciones de *"Materiales"* y *"Clonar"* 88

Figura 54. Definición de botones con acciones personalizadas 89

Figura 55. Filtrado de registros en **`CustomComboBox`** 91

Figura 56. Fragmento de código de **`CustomComboBox`** 92

Figura 57. **`CustomTabs`** en la pestaña de *Estructuras* 94

Figura 58. **`CustomTabs`** en la pestaña de *Clasificación* 94

Figura 59. Fragmento de código de **`CustomTabs`** 95

Figura 60. Barra lateral de **`ActionBar`** 97

Figura 61. Pantalla del mapa 97

Figura 62. Fragmento de código con las propiedades del componente **`Map`** 98

Figura 63. Fragmento de código de **`useMapInitialization.js`** 100

Figura 64. Capas generadas dinámicamente 101

Figura 65. Fragmento de código de **`useLayerManagement.js`** 101

Figura 66. Herramientas de dibujo geográfico 103

Figura 67. Fragmento de código de **`useSketchTools.js`** 104

Figura 68. Tooltip con las coordenadas UTM 105

Figura 69. Botones personalizados en el mapa 105

Figura 70. Ventana emergente con las propiedades de la capa seleccionada 106

Figura 71. Ventana emergente para buscar elementos por capas 107

Figura 72. Pantalla de inicio de sesión SISDE Web 108

Figura 73. Pantalla de recuperación de contraseña SISDE Web 108

Figura 74. Pantalla de Inicio SISDE Web..... 109

Figura 75. Pantalla de Gestión de Proyectos SISDE Web 109

Figura 76. Pantalla de Gestión de Materiales SISDE Web..... 110

Figura 77. Pantalla de Gestión de Estructuras SISDE Web..... 111

Figura 78. Pantalla de Actualización de Precios SISDE Web 111

Figura 79. Pantalla de Gestión de Usuarios SISDE Web..... 112

Figura 80. Pantalla de gestión de roles y permisos SISDE Web 113

Figura 81. Pantalla del área de trabajo SISDE Web 113

Figura 82. Base de datos en producción..... 117

Figura 83. Datasorce *"sisdeDS"* en WebLogic 117

Figura 84. Aplicaciones frontend y backend desplegadas en WebLogic..... 118

ÍNDICE DE ANEXOS

Anexo A. Tabulación de respuestas SUS del sistema SISDE de escritorio. 129

Anexo B. Tabulación de respuestas SUS del sistema SISDE Web. 130

RESUMEN EJECUTIVO

La migración del sistema "SISDE" de la Empresa Eléctrica Ambato Regional Centro Norte S.A. (EEASA) a un entorno web tiene por objetivo solucionar problemas críticos del modelo de escritorio. La investigación identifica que el software instalado localmente genera altos costos operativos, ineficiencias significativas y retrasos en proyectos debido a la necesidad de actualizaciones manuales que requieren desplazamientos físicos de los contratistas.

Mediante investigación de campo aplicando el System Usability Scale (SUS) a 60 usuarios y entrevistas estructuradas a administradores del sistema, se determinó una arquitectura tecnológica óptima basada en ArcGIS, React, Spring Boot y Oracle, combinando rendimiento, escalabilidad y compatibilidad con la infraestructura existente en la EEASA. La metodología RAD (Rapid Application Development) fue seleccionada por su enfoque en la alta participación de usuarios, reutilización de componentes y ciclos cortos de desarrollo.

La solución implementada demuestra resultados excepcionales con una mejora del 32.3% en usabilidad, incrementando el puntaje del SUS de 61.04 a 80.75 puntos, elevando el sistema de la categoría “D” a “A-”, según las categorías SUS. La migración del sistema SISDE eliminó completamente las experiencias de usuario problemáticas, transformando un escenario donde el 75% de usuarios se encontraba en categorías insatisfactorias a uno donde el 85% alcanza calificaciones excelentes. El entorno web permite actualizaciones 100% automáticas y remotas, eliminando la necesidad de desplazamientos físicos y tiempos de inactividad.

Palabras clave: GIS, migración web, aplicaciones geoespaciales, RAD, arquitecturas web, redes eléctricas.

ABSTRACT

The migration of the "SISDE" system from Empresa Eléctrica Ambato Regional Centro Norte S.A. (EEASA) to a web environment aims to address critical issues within the desktop model. Research identifies that locally installed software generates high operational costs, significant inefficiencies, and project delays due to the need for manual updates requiring physical deployment of users.

Through field research applying the System Usability Scale (SUS) to 60 users and structured interviews with system administrators, an optimal technological architecture was determined based on ArcGIS, React, Spring Boot, and Oracle, combining performance, scalability, and compatibility with existing EEASA's infrastructure. The RAD (Rapid Application Development) methodology was selected for its focus on high user participation, component reusability, and short development cycles.

The implemented solution demonstrates exceptional results with a 32.3% improvement in usability, increasing the SUS score from 61.04 to 80.75 points, elevating the system from category "D" to "A-" according to SUS categories. The SISDE system migration eliminated problematic user experiences, transforming a scenario where 75% of users were in unsatisfactory categories to one where 85% achieve excellent ratings. The web environment enables 100% automatic and remote updates, eliminates the need for physical deployments and downtime.

Keywords: GIS, web migration, geospatial applications, RAD, web architectures, electrical networks.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

MIGRACIÓN DEL SISTEMA "SISDE" A ENTORNO WEB PARA LA EMPRESA ELÉCTRICA AMBATO REGIONAL CENTRO NORTE S.A.

1.1.1 Planteamiento del problema

El costo asociado a la actualización del software es un desafío significativo, especialmente cuando las actualizaciones deben realizarse con frecuencia. En la actualidad, el software “SISDE”, utilizado en la Empresa Eléctrica de Ambato (EEASA) para la elaboración de proyectos de red eléctrica, está instalado localmente en los equipos de los contratistas que operan en Tungurahua, Napo y Pastaza. Este modelo de distribución genera problemas críticos relacionados con los costos y tiempos de actualización.

Cada vez que se lanza una nueva versión del software, que incluye cambios en la base de datos o en las funcionalidades, los contratistas deben desplazarse físicamente a las instalaciones de la empresa para reinstalar el programa. Este proceso no solo implica inactividad en los proyectos mientras se realiza la actualización, sino también un alto costo para la empresa, que debe invertir tiempo y recursos en generar instaladores y proporcionar soporte técnico personalizado para cada contratista. La situación se agrava debido a la dispersión geográfica de los usuarios, que a menudo están ubicados en provincias distintas, aumentando aún más los costos de desplazamiento y tiempo perdido.

El problema radica en que la infraestructura actual carece de un mecanismo centralizado y automatizado para distribuir actualizaciones, lo que resulta ineficiente tanto para la empresa como para los contratistas. Este modelo de actualización no solo incrementa los costos operativos, sino que también retrasa los proyectos al generar periodos de inactividad mientras los contratistas esperan obtener la versión más reciente del software.

Por lo tanto, la migración del software “SISDE” a un entorno web emerge como una solución efectiva para este problema. Un entorno web permitiría centralizar las actualizaciones, distribuyéndolas automáticamente a todos los usuarios en tiempo real, eliminando la necesidad de desplazamientos y reduciendo significativamente los costos asociados al proceso. Este cambio también mejoraría la productividad de los contratistas al garantizar que siempre trabajen con la versión más actualizada del software.

1.2 Antecedentes investigativos

Ding et al. implementaron una plataforma GIS serverless en la nube utilizando Mapbox GL JS para el frontend y servicios AWS Lambda con S3 en el backend. La base de datos PostgreSQL se integró con un modelo serverless para mejorar el rendimiento y reducir la latencia. El sistema web y de escritorio beneficia tanto a empresas como usuarios finales, logrando manejar grandes volúmenes de datos de manera eficiente y eliminando la necesidad de mantener infraestructura costosa. No se especificó la metodología utilizada ni las dificultades encontradas [1].

Kum et al. desarrollaron un paquete de visualización de datos en tiempo real usando Jupyter Notebook y bqplot para el frontend, con Python, Pandas y Xarray en el backend. El sistema utiliza Ipyleaflet y OpenStreetMap para visualización de mapas, con datos almacenados en JSON, CSV y NetCDF. Se implementó FTP en lugar de ERDDAP para mejorar los tiempos de carga en conexiones inestables. El sistema web y de escritorio beneficia a empresas y usuarios finales, con planes de integración futura con más datos de IOOS. No se especificaron las dificultades ni la metodología empleada [2].

Li et al. desarrollaron una arquitectura GIS integrada utilizando SuperMap GIS y Apache Spark Streaming, con Elasticsearch como base de datos. La migración a la nube usando contenedores Docker permitió una gestión más eficiente de recursos y mejor respuesta. La plataforma ofrece interfaces web, móvil y escritorio con visualización 2D y 3D, beneficiando tanto a empresas como usuarios finales. Se logró mejorar la rapidez en actualizaciones de datos y reducir costos operativos mediante DevOps y entrega continua [3].

Sugumaran et al. implementaron una metodología GIS Cloud Computing utilizando HTML5 y JavaScript tanto en frontend como backend. La solución demostró ser más rápida y fácil que las soluciones GIS tradicionales, con mejor escalabilidad al no estar limitada por el número de núcleos del servidor. El sistema web y móvil con visualización 2D beneficia a empresas y usuarios finales, logrando reducir los costos del servidor casi a la mitad. No se especificaron la metodología ni las dificultades encontradas [4].

Kiarie desarrolló una aplicación web GIS utilizando HTML/CSS/JavaScript en el frontend y Python con Apache Web Server en el backend, empleando GeoDjango para la visualización de mapas y base de datos. La migración a un servidor virtual en la nube mejoró el acceso y velocidad de los datos. El sistema web con visualización 2D beneficia a empresas y usuarios finales, mejorando la gestión de datos de infraestructura eléctrica. Se encontraron dificultades en el acceso a datos espaciales de otras agencias y la consolidación de requisitos de usuarios [5].

Barlocco implementó una herramienta de modelado georeferenciado usando ReactJS/Redux en el frontend y CherryPy/Python en el backend, con MapStore y GeoServer para visualización de mapas. La migración usando Docker mejoró la flexibilidad y rendimiento del sistema. La plataforma web y móvil con visualización 2D y 3D beneficia a empresas y usuarios finales, mejorando la escalabilidad y reduciendo costos operativos. Se encontraron dificultades en la conectividad entre componentes y la autenticación entre servicios [6].

Kumar et al. desarrollaron una plataforma Web-GIS integrada usando Apache y MapScript en el backend, con PostgreSQL/PostGIS como base de datos. El sistema permite consultas rápidas y precisas para la toma de decisiones, con una arquitectura de tres capas. La plataforma web con visualización 2D beneficia a empresas y usuarios finales, reduciendo tiempos de análisis y costos operativos. Se identificó la necesidad de mejorar el manejo de múltiples capas y optimizar el almacenamiento de datos [7].

Moura et al. implementaron una plataforma GISColab utilizando WebGIS con GeoServer, integrando widgets y capas dinámicas para un procesamiento más eficiente. El sistema web con visualización 2D beneficia a empresas y usuarios finales, permitiendo cálculos en tiempo real de métricas ambientales. Se utilizó una

metodología basada en co-creación y aprendizaje transformativo, encontrando dificultades en la actualización de infraestructuras de datos espaciales y la calidad de datos disponibles [8].

See et al. desarrollaron un software de código abierto para la validación y comparación de mapas educativos. El sistema de escritorio con visualización 2D beneficia principalmente a usuarios finales, enfocándose en la creación de materiales educativos accesibles. Se utilizó un enfoque basado en encuestas y talleres prácticos. No se especificaron detalles sobre tecnologías frontend/backend ni dificultades encontradas [9].

Chen et al. crearon un sistema de mapeo de funciones de edificios urbanos utilizando la API de Google Maps Static. El sistema web y móvil con visualización 2D beneficia a empresas y usuarios finales, optimizando el proceso mediante computación paralela y web crawlers. Se utilizó la metodología OneClassSVM, encontrando desafíos en la falta de datos en ciertas regiones, especialmente en países del sur global [10].

Ettefagh implementó una plataforma Web GIS usando C# y ASP.NET MVC en el backend, con SQL Server como base de datos y ArcMap/ArcGIS para visualización. El sistema web con visualización 2D beneficia a usuarios finales, utilizando procesamiento paralelo y arquitectura MVC. Se encontraron problemas con la precisión de datos GPS en áreas densamente construidas. No se especificaron resultados esperados [11].

Mandel desarrolló una plataforma de gestión de datos utilizando Collector for ArcGIS en el frontend y ArcGIS Online en el backend con una geodatabase. El sistema web, móvil y de escritorio con visualización 2D beneficia a empresas y usuarios finales, permitiendo la recolección de datos en tiempo real y mejorando la eficiencia en la gestión. No se especificaron la metodología ni las dificultades encontradas [12].

Minango implementó una herramienta GIS usando JavaScript/HTML/CSS/Leaflet en el frontend y PHP/PostgreSQL en el backend, con MapServer para visualización de mapas. El sistema web y de escritorio con visualización 2D beneficia a empresas y usuarios finales, facilitando la gestión de grandes volúmenes de datos geoespaciales.

Se utilizó la metodología Scrum y XP. No se especificaron las dificultades encontradas [13].

Bychkov et al. desarrollaron una infraestructura de procesamiento de datos geoespaciales utilizando servicios del Open Geospatial Consortium y PostgreSQL. El sistema web beneficia a usuarios finales, simplificando las actualizaciones mediante servidores remotos y servicios distribuidos, reduciendo costos operativos. Se encontraron dificultades en la integración de recursos de información distribuidos y la adopción de estándares. No se especificaron detalles sobre tecnologías frontend [14].

Assis et al. implementaron una infraestructura de análisis de datos espaciales usando GeoServer y microservicios SaaS, con PostgreSQL y Redis como bases de datos. El sistema web con visualización 2D beneficia a usuarios finales, permitiendo una implementación y actualización más rápidas mediante contenedores y microservicios. Se encontraron dificultades en garantizar la interoperabilidad con grandes volúmenes de datos espaciales. No se especificó la metodología utilizada [15].

Giglione et al. desarrollaron una plataforma web GIS usando TypeScript en el frontend y Java/Apache Tomcat en el backend, con MySQL como base de datos. El sistema web con visualización 2D beneficia a empresas, proporcionando acceso más rápido y directo a la información. Se encontraron dificultades en la migración manual de la base de datos desde Access a MySQL. No se especificaron resultados esperados en términos de escalabilidad y costos [16].

Solpieva creó una plataforma de e-learning utilizando HTML/CSS/JavaScript con múltiples APIs de mapeo (ArcGIS Maps SDK, Google Maps API y Leaflet). El sistema web con visualización 2D y 3D beneficia a usuarios finales. Se utilizó una matriz comparativa de APIs de mapeo como metodología, encontrando dificultades en la evaluación y selección de APIs debido a la falta de comparaciones detalladas. No se especificaron resultados esperados [17].

Cignetti et al. implementaron una plataforma web usando GeoNetwork en el frontend y Java en el backend, con GeoServer para visualización de mapas. El sistema web y de escritorio con visualización 2D beneficia a usuarios finales, mejorando la interoperabilidad y acceso a datos geoespaciales. Se encontraron dificultades en la

integración de datos multisensoriales y la migración del sistema. No se especificaron resultados esperados en términos de costos y escalabilidad [18].

White et al. desarrollaron una plataforma de modelado participativo usando React y OpenLayers en el frontend y Django con REST Framework en el backend, utilizando PostgreSQL con PostGIS como base de datos. El sistema web con visualización 2D y 3D beneficia a usuarios finales, mejorando la eficiencia en la recolección y análisis de datos. Se encontraron dificultades en la integración de datos geoespaciales complejos y el mantenimiento de la interoperabilidad. No se especificaron resultados esperados en términos de costos y escalabilidad [19].

1.3 Fundamentación teórica

A continuación, se proporciona una base conceptual para la migración del software "SISDE" a entorno web, abordando las tecnologías y enfoques clave en sistemas de información geográfica y plataformas de desarrollo web. Esta sección revisa conceptos fundamentales y estudios relevantes, permitiendo entender las herramientas seleccionadas y la justificación de su aplicación en este proyecto, así como el contexto técnico que sustenta la propuesta de migración.

1.3.1 Sistemas de Información Geográfica (GIS)

Los Sistemas de Información Geográfica (GIS) son herramientas fundamentales para la gestión y análisis de datos espaciales. Permiten integrar, almacenar, analizar, manipular y visualizar información geográfica, lo cual los convierte en soluciones clave para la planificación y administración de infraestructuras, como es el caso de las redes eléctricas como se menciona en [20], [21]. Los GIS han evolucionado de sistemas de escritorio a soluciones web, lo que ha permitido una mayor accesibilidad, colaboración y toma de decisiones basadas en datos espaciales. La migración de sistemas GIS a entornos web ha sido impulsada por la necesidad de mejorar la accesibilidad, escalabilidad y rendimiento de estas aplicaciones.

Los GIS funcionan mediante la superposición de capas de información georreferenciada, permitiendo realizar análisis espaciales complejos que integran

diferentes tipos de datos como topografía, infraestructura, demografía y recursos naturales. Estos sistemas utilizan estructuras de datos especializadas para representar la información espacial, como formatos vectoriales para entidades discretas (puntos, líneas y polígonos) y formatos ráster para fenómenos continuos (elevación, temperatura, etc.).

Las implementaciones modernas de GIS incluyen capacidades avanzadas como análisis de redes, modelado 3D, procesamiento de imágenes satelitales y soporte para sensores en tiempo real, lo que los convierte en plataformas versátiles para la toma de decisiones espaciales. Además, la integración con tecnologías emergentes como Internet of Things (IoT), Big Data y técnicas de inteligencia artificial ha expandido significativamente su aplicabilidad en diversos sectores.

Como se evidencia en [5], la migración de sistemas GIS a entornos web ha demostrado mejorar significativamente el acceso y la velocidad de procesamiento de datos en la gestión de infraestructura eléctrica. Algunas de las tecnologías relevantes en este contexto son las siguientes [22]:

a. ArcGIS

Una suite integral de productos de Esri para la gestión de datos geográficos, que incluye herramientas para mapeo, análisis y administración de GIS. ArcGIS ofrece soluciones tanto de escritorio como web, destacándose por su robustez y funcionalidades avanzadas. En [11] y [12] se demuestra su efectividad en la implementación de sistemas web GIS empresariales.

ArcGIS implementa un modelo de geodatabase multiusuario que permite la edición simultánea de datos espaciales, utilizando un sistema de versiones para mantener la integridad de los datos durante transacciones concurrentes. Su arquitectura orientada a servicios (SOA) facilita la distribución de capacidades geoespaciales a través de la red mediante servicios web interoperables basados en estándares OGC (Open Geospatial Consortium).

Entre sus características principales destaca el soporte para datos 3D y temporales, capacidades avanzadas de cartografía y visualización, herramientas de análisis

espacial y geoprocesamiento, y una infraestructura escalable para implementaciones empresariales [23]. Asimismo, ofrece integración con múltiples fuentes de datos, sistemas de coordenadas y proyecciones cartográficas, y posibilidad de desarrollo personalizado mediante APIs y SDKs.

b. OpenLayers

Una biblioteca de JavaScript de código abierto para la visualización de mapas en aplicaciones web. OpenLayers permite crear mapas interactivos y reutilizables, con soporte para una amplia variedad de formatos y servicios geoespaciales. Su uso exitoso se evidencia en [19], donde se implementó junto con React para crear una plataforma de modelado participativo.

OpenLayers utiliza Canvas y WebGL para renderizar grandes cantidades de datos vectoriales y ráster directamente en el navegador, implementando técnicas avanzadas como teselado adaptativo y simplificación geométrica para optimizar el rendimiento. Su arquitectura modular permite cargar sólo los componentes necesarios, reduciendo el tamaño de la aplicación final.

Proporciona controles interactivos para navegación, selección y edición de entidades geográficas, así como soporte para eventos táctiles en dispositivos móviles [24]. Incorpora capacidades de proyección cartográfica al vuelo, estilos vectoriales avanzados basados en expresiones, animaciones temporales y se integra fácilmente con frameworks modernos de JavaScript como React, Angular y Vue.

c. GeoServer y MapServer

Herramientas de código abierto que permiten la publicación y visualización básica de datos geográficos en 2D a través de la web. [13] y [8] demuestran su eficacia en la implementación de sistemas GIS web escalables.

GeoServer implementa una arquitectura RESTful que facilita la configuración y administración programática del servidor, mientras utiliza GeoTools como su motor de procesamiento geoespacial central. Soporta transacciones WFS-T para edición remota de datos vectoriales y servicios WMTS para distribución eficiente de datos ráster mediante caché.

Estas plataformas destacan por su implementación completa de estándares OGC, capacidades de transformación de coordenadas al vuelo, extensibilidad mediante plugins, y soporte para clustering y balanceo de carga [25]. Adicionalmente, incorporan seguridad granular basada en roles, servicios de catálogo para metadatos geoespaciales (CSW), y ofrecen herramientas avanzadas para styling cartográfico a través de SLD y CSS.

d. ArcGIS Enterprise

Una solución completa de Esri que ofrece visualización y análisis espacial avanzado en entornos web, integrando múltiples componentes como servidores de mapas, bases de datos geoespaciales y herramientas de gestión.

ArcGIS Enterprise implementa una arquitectura de microservicios distribuida que separa funcionalidades en componentes especializados como Portal, Server, Data Store y GeoEvent, permitiendo despliegues escalables y resilientes. Utiliza un sistema de federación para integrar múltiples implementaciones en entornos distribuidos geográficamente.

Se caracteriza por sus capacidades analíticas avanzadas incluyendo procesamiento distribuido mediante ArcGIS GeoAnalytics, herramientas de machine learning, análisis espacio-temporal, y procesamiento de big data espacial [26]. Implementa un modelo de seguridad empresarial con integración LDAP/Active Directory, autenticación multifactor, y encriptación end-to-end, complementado con un marco de colaboración que incluye dashboards, storytelling y compartición controlada de recursos.

Análisis comparativo de Sistemas de Información Geográfica (GIS)

Tabla 1. Comparación de herramientas de Sistemas de Información Geográfica

Característica	ArcGIS	OpenLayers	GeoServer/ MapServer
Compatibilidad con infraestructura existente	Alta (ya utilizado en EEASA)	Media	Media
Curva de aprendizaje para usuarios actuales	Baja (familiaridad existente)	Alta	Alta

Funcionalidades para gestión de redes eléctricas	Completos y especializadas	Básicas, requieren desarrollo adicional	Básicas, requieren personalización
Soporte para visualización 2D	Excelente	Bueno	Bueno
Capacidades de análisis espacial	Avanzados	Limitadas	Limitadas
Integración con bases de datos espaciales	Nativa con Oracle Spatial	Requiere configuración adicional	Buena mediante conectores
Escalabilidad	Alta	Media	Media
Soporte técnico profesional	Complete (Esrí)	Comunidad de código abierto	Comunidad de código abierto
Costo de implementación	Alto (licencias)	Bajo (código abierto)	Bajo (código abierto)
Actualizaciones y mantenimiento	Regulares y profesionales	Dependiente de la comunidad	Dependiente de la comunidad

ArcGIS ha sido seleccionado principalmente por la compatibilidad con la infraestructura existente en la EEASA, donde ya se utiliza ArcGIS para otras operaciones. Esto minimiza la curva de aprendizaje para los usuarios actuales y facilita la integración con los sistemas existentes. Sus capacidades avanzadas para gestión de redes eléctricas representan una ventaja significativa para las necesidades específicas de la empresa, adicionalmente la empresa cuenta con el licenciamiento.

1.3.2 Tecnologías Frontend

En el desarrollo de interfaces para aplicaciones GIS, los frameworks de frontend juegan un rol esencial en la visualización y manipulación de datos espaciales en tiempo real. A continuación, se presentan algunas de las tecnologías relevantes [27], [28]:

a. JavaScript/HTML/CSS

Tecnologías web fundamentales para la creación de interfaces interactivas y la representación de mapas y datos geográficos. [13] y [4] demuestran su efectividad en implementaciones GIS web, sin embargo, también se menciona el desafío de no contar elementos reutilizables, por lo que la duplicación de código es inminente.

En aplicaciones GIS web, JavaScript implementa el modelo de programación asíncrona mediante Promises y async/await para manejar operaciones como carga de datos espaciales y consultas a servicios web sin bloquear la interfaz de usuario. La manipulación del DOM y eventos se utiliza para crear interacciones complejas con los mapas, como selección de entidades y herramientas de medición.

Estas tecnologías se complementan con WebGL para renderizado 3D eficiente, Web Workers para procesamiento en segundo plano de datos geoespaciales voluminosos, y APIs como Geolocation para funcionalidades basadas en ubicación. El uso de LocalStorage y IndexedDB permite implementar capacidades offline en aplicaciones GIS móviles, mientras que CSS Grid y Flexbox facilitan diseños responsivos para diferentes dispositivos y orientaciones de pantalla.

b. ReactJS/Redux

Un popular framework de JavaScript para la construcción de interfaces de usuario reactivas y modulares. [6] implementó exitosamente esta tecnología para una herramienta de modelado georeferenciado.

En aplicaciones GIS, React implementa un sistema de componentes reutilizables para elementos de mapas como controles de navegación, leyendas y paneles de información, utilizando el Virtual DOM para optimizar actualizaciones en visualizaciones complejas. Redux proporciona un estado centralizado que mantiene sincronizados los datos geoespaciales a través de múltiples vistas y componentes.

React Hooks ha simplificado la integración con bibliotecas GIS como OpenLayers y Leaflet, permitiendo encapsular funcionalidades espaciales complejas en hooks personalizados [29]. Sus capacidades de código splitting y lazy loading son cruciales para aplicaciones GIS que cargan datos bajo demanda según el área y nivel de zoom visible, mientras que la renderización del lado del servidor (SSR) mejora el rendimiento inicial y la indexación de contenido espacial.

c. TypeScript

TypeScript implementa interfaces y tipos genéricos para modelar estructuras de datos geoespaciales complejas, mejorando la consistencia en el manejo de formatos como

GeoJSON, TopoJSON y KML. Su sistema de módulos facilita la organización de funcionalidades espaciales en unidades cohesivas y reutilizables. Un superconjunto de JavaScript que añade tipado estático, como se utilizó en [16] para mejorar la mantenibilidad del código.

Se caracteriza por herramientas avanzadas de refactorización y navegación de código que agilizan el desarrollo de aplicaciones GIS complejas, verificación estática de tipos para prevenir errores comunes al manipular datos espaciales, y decoradores para implementar patrones como observadores en entidades geográficas [30]. Además, su interoperabilidad con librerías JavaScript existentes permite adopción gradual en proyectos GIS establecidos, mientras que sus capacidades de análisis estático mejoran significativamente la documentación automática de APIs espaciales.

d. Mapbox GL JS

Mapbox GL JS utiliza WebGL para renderizar mapas vectoriales directamente en la GPU, permitiendo rotación 3D, inclinación y visualización fluida de grandes volúmenes de datos geoespaciales. Implementa un sistema de expresiones declarativas para definir estilos dinámicos basados en propiedades de los datos y condiciones de zoom. Utilizado en [1] para implementar una plataforma GIS serverless con excelentes resultados en rendimiento.

Destaca por su arquitectura de teselado vectorial que optimiza la transferencia de datos mediante compresión y descarga progresiva, animaciones y transiciones fluidas para cambios de estado del mapa, y soporte avanzado para fuentes de datos personalizadas mediante protocolos como Vector Tiles y GeoJSON [31]. Incorpora capacidades de terreno 3D y extrusión de edificios, herramientas para geocodificación y enrutamiento, y una API extensible mediante plugins para funcionalidades personalizadas.

e. ArcGIS Maps SDK

ArcGIS Maps SDK implementa un modelo de programación orientado a promesas para operaciones asíncronas como carga de capas y consultas espaciales, junto con un sistema de observación reactiva que mantiene sincronizados múltiples componentes de la aplicación con el estado del mapa. Utiliza workers en segundo plano para

procesar operaciones intensivas como análisis de visibilidad y cálculos de superficie. Implementado en [17] junto con otras APIs de mapeo para crear una plataforma educativa versátil.

Se caracteriza por capacidades 3D avanzadas incluyendo visualización de subsuperficie y análisis volumétrico, integración nativa con servicios de ArcGIS Online y Enterprise para autenticación y acceso a contenido, y widgets preconstruidos para funcionalidades comunes como búsqueda, medición y edición [32]. También ofrece herramientas para visualización avanzada de datos temporales, soporte para sistemas de coordenadas personalizados, y capacidades offline mediante un sistema sofisticado de caché y sincronización.

Análisis comparativo de Tecnologías Frontend

Tabla 2. Comparación de tecnologías frontend

Característica	React + JSX	Angular	Vue.js	JavaScript/ HTML/CSS puro
Integración con ArcGIS SDK	Excelente (componentes oficiales)	Buena	Buena	Posible pero manual
Rendimiento con datos geoespaciales	Alto	Alto	Alto	Medio
Componentes reutilizables	Excelente	Excelente	Bueno	Limitado
Mantenibilidad del código	Alta	Alta	Alta	Baja
Curva de aprendizaje para desarrolladores	Media	Alta	Media	Baja
Comunidad y soporte	Muy amplia	Amplia	Creciente	Universal
Sistemas de diseño disponibles	Múltiples (Material UI, Calcite)	Angular Material	Voetify	Limitados
Estado centralizado	Redux/Context API	NgRx	Vuex	Implementación manual
Renderizado eficiente	Virtual DOM	Change Detection	Virtual DOM	Manual
Actualizaciones y estabilidad	Frecuentes, API estable	Frecuentes, mayores cambios entre versiones	Frecuentes, API estable	Estándar web

React con JSX fue seleccionado por su excelente integración con ArcGIS SDK for JavaScript, que proporciona componentes oficiales específicamente diseñados para aplicaciones GIS. La arquitectura basada en componentes de React facilita la creación de interfaces modulares y reutilizables, aspecto crítico identificado en la entrevista con el administrador. La combinación de Calcite Design System (alineado con el ecosistema Esri) y Material UI proporciona un conjunto completo de componentes de interfaz que acelerarán el desarrollo y mejorarán la usabilidad del sistema, abordando directamente las quejas sobre la interfaz del sistema de escritorio.

1.3.3 Tecnologías Backend

Para respaldar la funcionalidad avanzada de un sistema GIS web, se requiere un backend escalable y robusto que permita gestionar grandes volúmenes de datos y facilitar su acceso en tiempo real. Algunas de las tecnologías relevantes incluyen [33], [34]:

a. Python/Django

Python implementa bibliotecas especializadas como GeoPandas, Shapely y Rasterio que proporcionan estructuras de datos optimizadas para operaciones espaciales vectoriales y ráster, con integración nativa con GDAL/OGR para acceso a múltiples formatos geoespaciales. Django integra GeoDjango como extensión que añade tipos de datos espaciales, indexación espacial y funciones de consulta geográfica a su ORM. Utilizado exitosamente en [2] y [19] para procesamiento de datos geoespaciales.

Este stack tecnológico destaca por su ecosistema de análisis científico (NumPy, SciPy, pandas) que facilita procesamiento y análisis espacial avanzado, integración con servicios de procesamiento remoto como Earth Engine, y frameworks de machine learning como scikit-learn para análisis predictivo espacial [35]. Además, incorpora herramientas como Celery para procesamiento asíncrono de tareas geoespaciales intensivas, soporta despliegue como contenedores Docker para escalabilidad horizontal, y proporciona serializadores especializados para datos geográficos en APIs REST.

b. Java/Apache Tomcat

Java implementa la Java Topology Suite (JTS) para operaciones geométricas avanzadas como análisis de topología, simplificación y generalización, mientras que frameworks como GeoTools proporcionan una plataforma completa para desarrollo de aplicaciones GIS en conformidad con estándares OGC. Apache Tomcat sirve como contenedor de servlets que ejecuta aplicaciones GIS web con capacidad de clustering para alta disponibilidad. Implementado en [16] y [18] para crear sistemas GIS web robustos.

Este stack tecnológico se caracteriza por su máquina virtual (JVM) que proporciona portabilidad a través de diferentes plataformas, gestión de memoria automática crucial para operaciones geoespaciales intensivas, e integración con bibliotecas empresariales como Spring para inyección de dependencias y gestión transaccional [36]. Adicionalmente, ofrece herramientas para monitoreo y profiling de rendimiento, soporte para multi-threading que aprovecha procesadores multinúcleo en operaciones espaciales paralelas, y frameworks como JAX-RS para implementación de servicios web RESTful geoespaciales.

c. PHP

PHP implementa extensiones como PHP-OGR y PHP-GDAL que proporcionan bindings nativos a bibliotecas geoespaciales de alto rendimiento, mientras que frameworks como Laravel y Symfony facilitan el desarrollo estructurado de aplicaciones GIS siguiendo patrones MVC. Su integración con Leaflet y OpenLayers mediante generación dinámica de JavaScript permite crear interfaces de mapas interactivas. Demostrado efectivo en [13] para la gestión de datos geoespaciales en entornos web.

Se caracteriza por su facilidad de despliegue en diversos entornos de hosting, integración directa con bases de datos espaciales mediante PDO y extensiones específicas para PostGIS, y procesamiento eficiente de archivos shapefile y otros formatos geoespaciales mediante bibliotecas especializadas [37]. También ofrece capacidades de cache avanzadas mediante Redis o Memcached para mejorar el rendimiento de consultas espaciales frecuentes, soporte para servicios web SOAP y REST para interoperabilidad con otros sistemas GIS, y herramientas de templating para generar mapas estáticos y dinámicos.

d. C#/.NET

C# implementa el framework .NET Feature Service que proporciona una API unificada para acceso a datos geoespaciales de diversas fuentes, mientras que Entity Framework con extensiones espaciales facilita operaciones CRUD sobre bases de datos geográficas utilizando LINQ. ASP.NET Core MVC estructura las aplicaciones GIS web en componentes modulares siguiendo el patrón Modelo-Vista-Controlador. Utilizado en [11] para implementar una plataforma Web GIS con arquitectura MVC.

C# implementa el framework .NET Feature Service que proporciona una API unificada para acceso a datos geoespaciales de diversas fuentes, mientras que Entity Framework con extensiones espaciales facilita operaciones CRUD sobre bases de datos geográficas utilizando LINQ. ASP.NET Core MVC estructura las aplicaciones GIS web en componentes modulares siguiendo el patrón Modelo-Vista-Controlador.

Este stack tecnológico destaca por su rendimiento optimizado mediante compilación JIT y AOT para operaciones geoespaciales intensivas, soporte para múltiples plataformas a través de .NET Core permitiendo despliegue en Windows, Linux y macOS, y capacidades asíncronas mediante async/await para operaciones espaciales sin bloqueo [38]. Incluye bibliotecas como NetTopologySuite para operaciones geométricas complejas, integración con Azure Maps y otros servicios cloud geoespaciales, y herramientas avanzadas de depuración y profiling para identificar cuellos de botella en procesamiento espacial.

e. AWS Lambda

AWS Lambda implementa funciones sin estado que procesan datos geoespaciales en respuesta a eventos como subidas de archivos a S3 o peticiones HTTP a API Gateway, escalando automáticamente según la demanda sin necesidad de aprovisionar servidores. Utiliza capas Lambda para compartir bibliotecas geoespaciales comunes como GDAL y PROJ entre múltiples funciones. Implementado en [1] como parte de una arquitectura serverless para mejorar la escalabilidad.

Se caracteriza por su modelo de facturación basado en tiempo de ejecución que optimiza costos para cargas de trabajo geoespaciales variables, integración nativa con

servicios de almacenamiento como S3 para datos geoespaciales de gran volumen, y capacidades de logging y monitoreo a través de CloudWatch [39]. Adicionalmente, ofrece procesamiento en paralelo mediante invocaciones asíncronas para análisis distribuido de datos espaciales, tiempos de arranque en milisegundos mediante provisioned concurrency para aplicaciones GIS en tiempo real, y acceso a aceleradores de hardware como GPU para procesamiento de imágenes satelitales.

f. Spring Boot

Un framework de Java que simplifica el desarrollo de aplicaciones web al proporcionar una configuración predeterminada y herramientas de inyección de dependencias. Spring Boot implementa configuración automática para componentes GIS comunes como repositorios espaciales y servicios de geocodificación, reduciendo significativamente el código boilerplate mediante anotaciones como @GeoJsonView para serialización de entidades espaciales. Su arquitectura basada en aspectos permite separar claramente las preocupaciones espaciales de la lógica de negocio principal.

Este framework destaca por su sistema de perfiles que facilita diferentes configuraciones para entornos de desarrollo, prueba y producción, actuadores para monitoreo en tiempo real de aplicaciones GIS, y soporte para microservicios geoespaciales mediante Spring Cloud [40]. Incorpora Spring Data Spatial para abstracción consistente sobre diferentes bases de datos geoespaciales, Spring Security para protección de APIs y datos espaciales sensibles, y Spring Batch para procesamiento eficiente de grandes volúmenes de datos geoespaciales.

Análisis comparativo de Tecnologías Backend

Tabla 3. Comparación de tecnologías backend

Característica	Spring Boot	Python/ Django	Java/Apache Tomcat	PHP	AWS Lambda
Compatibilidad con WebLogic	Excelente (WAR)	Limitada	Buena	Limitada	No aplicable
Integración con Oracle Spatial	Nativa	Mediante bibliotecas	Nativa	Mediante extensiones	Requiere configuración

Rendimiento	Alto	Bueno	Alto	Medio	Alto (con latencia inicial)
Escalabilidad	Excelente	Bueno	Bueno	Medio	Excelente
Seguridad	Robusta	Bueno	Robusta	Requiere configuración adicional	Buena
Ecosistema Spring	Completo (Security, Data, etc.)	No aplicable	Parcial	No aplicable	No aplicable
Facilidad de desarrollo	Alta	Alta	Media	Alta	Media
Soporte para servicios RESTful	Nativo	Django REST Framework	JAX-RS	Requiere frameworks	API Gateway
Documentación API	Swagger/ OpenAPI integrado	Django REST Docs	Requiere configuración	Requiere herramientas adicionales	Swagger/ OpenAPI
Compatibilidad con microservicios	Excelente	Bueno	Medio	Limitada	Excelente

Spring Boot fue seleccionado principalmente por su capacidad para generar archivos WAR compatibles con WebLogic, el servidor de aplicaciones existente en EEASA. Esta característica facilita la integración con la infraestructura actual sin requerir cambios significativos en el entorno de despliegue. Adicionalmente, Spring Boot ofrece una integración nativa con Oracle, la base de datos geoespacial seleccionada, y proporciona un ecosistema completo de módulos para seguridad, acceso a datos y servicios REST, que son esenciales para la implementación de las funcionalidades requeridas. Su rendimiento y escalabilidad también fueron factores decisivos, considerando el requisito de tiempos de respuesta rápidos expresado en las entrevistas.

1.3.4 Bases de Datos Geoespaciales

El soporte para datos geoespaciales en sistemas de gestión de bases de datos es un componente crítico para el desarrollo de aplicaciones GIS efectivas. Las bases de datos geoespaciales extienden los sistemas de bases de datos tradicionales con tipos de datos especializados, índices espaciales, y operadores geométricos que permiten almacenar,

consultar y analizar información geográfica de manera eficiente [41]. A continuación, se presentan las principales opciones disponibles en el mercado:

a. PostgreSQL y PostGIS

PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto, que en combinación con la extensión PostGIS, se ha convertido en una opción popular y líder para el almacenamiento y gestión de datos geoespaciales. Como se demuestra en [19] y [13], PostGIS proporciona un soporte robusto para datos espaciales y funciones de análisis geoespacial avanzadas, siendo especialmente efectivo en aplicaciones GIS que manejan grandes volúmenes de información geográfica.

PostGIS implementa el estándar Simple Features for SQL del OGC con extensiones avanzadas, utilizando índices espaciales R-tree sobre GiST (Generalized Search Tree) para optimizar consultas geográficas complejas. Su motor MVCC (Multi-Version Concurrency Control) permite lecturas consistentes sin bloqueos, crucial para aplicaciones GIS multiusuario.

Se caracteriza por su soporte para topología espacial manteniendo relaciones entre elementos geográficos, capacidades 3D y 4D (tiempo) para modelado espacio-temporal, y procesamiento ráster para análisis de imágenes satelitales y MDE [42]. Incluye más de 1,000 funciones espaciales para operaciones como enrutamiento, geocodificación y análisis de clusters, escalabilidad horizontal mediante particionamiento y sharding geográfico, y soporte para datos LiDAR mediante extensión pointcloud.

b. Oracle

Oracle es un sistema de gestión de bases de datos ampliamente utilizado en entornos empresariales. Se destaca por su rendimiento en consultas complejas y su robustez, proporcionando características avanzadas de particionamiento y soporte para datos geoespaciales. Esto lo convierte en una elección ideal para aplicaciones GIS que requieren alta disponibilidad, escalabilidad y precisión en el manejo de grandes volúmenes de datos.

Oracle Spatial implementa un sistema de referencia espacial unificado que soporta miles de sistemas de coordenadas y transformaciones, optimizando proyecciones cartográficas al vuelo mediante algoritmos altamente precisos. Utiliza índices espaciales R-tree y Quadtree adaptados a diferentes patrones de consulta geoespacial.

Este sistema destaca por su arquitectura multipropósito que integra tipos de datos vectoriales, ráster, topología y redes en un modelo unificado, capacidades de procesamiento en memoria para análisis espacial de alto rendimiento, y soporte nativo para modelado jerárquico de redes multimodales [43]. Además, incorpora funcionalidades avanzadas de particionamiento espacial para gestión eficiente de datos históricos y actuales, replicación multisitio para distribución geográfica de datos críticos, y algoritmos especializados para análisis de proximidad, rutas óptimas y áreas de servicio.

c. MySQL y SQL Server

Aunque MySQL y SQL Server son opciones confiables de bases de datos relacionales, su soporte para datos geoespaciales es más limitado. Sin embargo, [16] demuestra la viabilidad de MySQL en sistemas GIS más pequeños, mientras que [11] utilizó SQL Server exitosamente en una implementación GIS web.

MySQL implementa tipos de datos espaciales siguiendo el estándar OGC, con soporte para índices R-tree para mejorar el rendimiento de consultas espaciales. Sus características espaciales se complementan con procedimientos almacenados para encapsular lógica geoespacial compleja.

SQL Server Spatial proporciona dos tipos de datos espaciales: geometry para datos planos y geography para datos geodésicos en coordenadas elipsoidales, implementando aproximadamente 70 métodos para operaciones espaciales y soportando múltiples formatos de serialización [44]. Estas plataformas se caracterizan por su facilidad de implementación y administración, amplio soporte empresarial con herramientas gráficas para gestión de datos espaciales, e integración nativa con sus respectivos ecosistemas de desarrollo (PHP/MySQL y .NET/SQL Server).

d. Elasticsearch

Elasticsearch implementa índices espaciales basados en árboles cuadráticos adaptados a su arquitectura distribuida, permitiendo búsquedas geoespaciales de alta velocidad en conjuntos de datos masivos. Su modelo de documentos JSON flexible facilita el almacenamiento de geometrías complejas junto con metadatos asociados. Como se evidencia en [3], Elasticsearch es efectivo para búsquedas y análisis en tiempo real de datos geográficos, especialmente cuando se combina con tecnologías como Apache Spark Streaming.

Este sistema destaca por su escalabilidad horizontal mediante fragmentación geoespacial que distribuye datos según ubicación, capacidades avanzadas de agregación geoespacial para análisis en tiempo real, y arquitectura sin esquema que permite evolución flexible de modelos de datos espaciales [45]. Incorpora consultas complejas combinando criterios espaciales y textuales, visualizaciones geoespaciales mediante integración con Kibana, y capacidad de procesamiento de streams geoespaciales en tiempo real mediante pipeline de ingestión.

Análisis comparativo de Bases de Datos Geoespaciales

Tabla 4. Comparación de bases de datos geoespaciales

Característica	Oracle	PostgreSQL/ PostGIS	MySQL Spatial	SQL Server Spatial	Elasticsearch
Infraestructura existente en EEASA	Ya implementado	Requiere nueva instalación	Requiere nueva instalación	Requiere nueva instalación	Requiere nueva instalación
Rendimiento con datos geoespaciales complejos	Excelente	Excelente	Bueno	Bueno	Muy bueno para búsquedas
Funciones de análisis espacial	Completas	Completas	Básicas	Intermedias	Limitadas
Soporte para topología	Avanzado	Avanzado	Limitado	Intermedio	No disponible
Escalabilidad	Alta	Alta	Media	Alta	Muy alta
Integración con ArcGIS	Nativa	Buena	Limitada	Buena	Requiere configuración

Seguridad	Robusta	Buena	Básica	Robusta	Requiere configuración
Facilidad de migración desde sistema de escritorio	Directa	Requiere conversión	Requiere conversión	Requiere conversión	Requiere reformulación
Soporte empresarial	Completo (Oracle)	Comunidad y proveedores comerciales	Comunidad y MySQL Enterprise	Microsoft	Elastic
Costo	Alto	Bajo (código abierto)	Variable (versión)	Alto	Variable (versión)

Oracle fue seleccionado principalmente por ser la solución ya implementada en la infraestructura de EEASA, lo que elimina la necesidad de migración de datos y facilita la integración con sistemas existentes. Sus capacidades avanzadas para análisis espacial y topología son particularmente relevantes para la gestión de redes eléctricas, requisito fundamental del sistema SISDE. La integración nativa con ArcGIS, la plataforma GIS seleccionada, también representa una ventaja significativa que simplificará el desarrollo y garantizará la coherencia en el manejo de datos geoespaciales. Aunque el costo de licenciamiento es alto, el hecho de que la empresa ya cuenta con esta infraestructura mitiga este factor.

1.3.5 Seguridad en Sistemas Web GIS

La seguridad es un aspecto fundamental cuando se gestionan aplicaciones web GIS, ya que estos sistemas suelen manejar información crítica y confidencial, como datos geoespaciales de infraestructura energética. Las amenazas a la seguridad incluyen el acceso no autorizado a los datos, ataques de denegación de servicio (DoS), y la alteración de datos. [6] destaca la importancia de abordar los desafíos de autenticación entre servicios. Algunas de las medidas de seguridad más comunes se listan a continuación:

a. Autenticación y autorización

Los sistemas modernos de autenticación para aplicaciones GIS web implementan protocolos como OAuth 2.0 y OpenID Connect para delegar la autenticación a proveedores de identidad seguros, mientras mantienen un control granular sobre

recursos geoespaciales sensibles. La federación de identidades permite integraciones seguras entre sistemas GIS distribuidos en diferentes organizaciones. Se recomienda el uso de métodos como la autenticación multifactor (MFA) para proteger el acceso a sistemas críticos. [3] implementó exitosamente estas medidas en su arquitectura GIS integrada.

Se caracterizan por implementar control de acceso basado en atributos (ABAC) que considera múltiples factores como ubicación del usuario, hora del día y sensibilidad de los datos geoespaciales para decisiones de autorización dinámica, sistemas de Single Sign-On (SSO) para experiencia de usuario fluida a través de múltiples aplicaciones GIS, y auditoría completa de actividades de usuarios para cumplimiento regulatorio [46]. También incorporan gestión de ciclo de vida de identidades con aprovisionamiento y desaprovisionamiento automatizado, integración con directorio corporativo (LDAP/Active Directory), y políticas de caducidad de contraseñas y sesiones.

b. Encriptación de datos

La encriptación de datos geoespaciales implementa esquemas avanzados como encriptación homomórfica que permite realizar ciertas operaciones espaciales directamente sobre datos cifrados sin necesidad de desencriptación previa. La segmentación de datos según nivel de sensibilidad permite aplicar diferentes niveles de protección criptográfica. Tanto en tránsito como en reposo, utilizando protocolos como HTTPS y cifrado de bases de datos. [1] demostró la importancia de la encriptación en entornos cloud.

Se caracteriza por utilizar encriptación TLS 1.3 con Perfect Forward Secrecy para proteger datos geoespaciales en tránsito, cifrado transparente a nivel de base de datos (TDE) para datos en reposo sin modificar aplicaciones, y gestión robusta de claves mediante HSM (Hardware Security Modules) [47]. Adicionalmente, implementa tokenización para sustituir identificadores personales en datasets geoespaciales, cifrado de metadatos sensibles, y cifrado selectivo por campo que permite búsquedas y análisis espacial mientras protege atributos confidenciales.

c. Auditoría y monitoreo

Los sistemas modernos de monitoreo para aplicaciones GIS implementan análisis de comportamiento (UEBA) que establece líneas base de actividad normal para detectar anomalías en patrones de acceso a datos geoespaciales mediante algoritmos de machine learning. La correlación de eventos a través de múltiples componentes del sistema GIS permite identificar ataques sofisticados. Implementación de registros detallados de acceso y uso del sistema para detectar actividades sospechosas. [15] implementó estas prácticas en su infraestructura de análisis de datos espaciales.

Se caracterizan por recolección centralizada de logs con parsing y normalización automatizada, dashboards en tiempo real sobre actividad del sistema con alertas configurables, y retención de logs a largo plazo para análisis forense e investigaciones [48]. También incluyen monitoreo de integridad de datos espaciales para detectar modificaciones no autorizadas, detección de anomalías en consultas espaciales (como acceso inusual a zonas geográficas restringidas), y escaneo automatizado de vulnerabilidades específicas para componentes GIS.

1.3.6 Arquitecturas de Sistemas Web GIS

Las arquitecturas de sistemas web GIS son estructuras esenciales que determinan cómo se organizan y se comunican los componentes del sistema. Las experiencias documentadas en varios estudios han demostrado la efectividad de diferentes enfoques:

a. Arquitectura de 2 capas

La arquitectura de dos capas implementa procesamiento en el cliente mediante JavaScript y WebGL para operaciones espaciales básicas como cálculo de distancias, áreas y visualización, reduciendo la carga en el servidor y mejorando la experiencia interactiva. La transferencia de datos optimizada utiliza formatos compactos como GeoJSON y TopoJSON para comunicación eficiente. En este modelo, el cliente interactúa directamente con la base de datos a través de una interfaz web. [4] implementó esta arquitectura con éxito en una solución cloud.

Se caracteriza por su simplicidad que reduce latencia al eliminar capas intermedias, implementación directa de APIs espaciales en bases de datos modernas que exponen

funcionalidades geoespaciales mediante interfaces RESTful, y gestión eficiente de conexiones mediante pooling y balanceo de carga [49]. Sin embargo, presenta limitaciones en separación de responsabilidades y encapsulamiento de lógica compleja, requiriendo complementos como servicios de procesamiento dedicados para análisis avanzado.

b. Arquitectura de 3 capas

La arquitectura de tres capas implementa un nivel intermedio de servicios que encapsula lógica de negocio geoespacial compleja como geocodificación, enrutamiento y análisis espacial, aislando la presentación (cliente) del almacenamiento (base de datos). Este diseño permite implementar políticas de seguridad y reglas de negocio centralizadas para datos geoespaciales sensibles. [7] demostró la efectividad de esta arquitectura para consultas rápidas y precisas en la toma de decisiones.

Se caracteriza por clara separación de responsabilidades que facilita mantenimiento y evolución independiente de componentes, reutilización de servicios geoespaciales comunes entre múltiples aplicaciones, y flexibilidad para escalar componentes específicos según cuellos de botella identificados [50]. Implementa patrones como Command Query Responsibility Segregation (CQRS) para optimizar separadamente operaciones de lectura y escritura en datos geoespaciales, caché inteligente en nivel intermedio, y transformación de datos para adaptarse a diferentes clientes y dispositivos.

c. Arquitectura de microservicios

La arquitectura de microservicios para GIS implementa servicios especializados y autónomos para funcionalidades geoespaciales específicas como geocodificación, análisis de rutas, procesamiento de imágenes satelitales y visualización cartográfica [51]. Cada servicio mantiene su propio almacenamiento de datos optimizado para su dominio específico, mientras que APIs bien definidas facilitan la integración. [15] implementó exitosamente una arquitectura de microservicios usando contenedores Docker y servicios SaaS.

Se caracteriza por despliegue independiente que permite actualizar componentes individuales sin afectar al sistema completo, aislamiento de fallos que evita que problemas en un servicio geoespacial afecten a otros, y poliglotismo tecnológico que permite seleccionar las herramientas óptimas para cada funcionalidad espacial [52]. También implementa descubrimiento dinámico de servicios, circuit breakers para manejar fallos en servicios dependientes, y orquestación de flujos de trabajo geoespaciales complejos mediante coreografía basada en eventos.

d. Arquitectura cliente-servidor

La arquitectura cliente-servidor para GIS web implementa capacidades diferenciadas donde el servidor se enfoca en almacenamiento seguro, consultas complejas y análisis espacial intensivo, mientras el cliente maneja visualización interactiva y operaciones básicas. La comunicación se estructura mediante APIs RESTful siguiendo principios HATEOAS para autodescubrimiento de recursos geoespaciales. [5] utilizó esta arquitectura con Python y Apache Web Server, demostrando su efectividad en la gestión de datos de infraestructura eléctrica.

Se caracteriza por su robustez y amplia adopción en sistemas empresariales, clara delimitación de responsabilidades entre frontend y backend, y compatibilidad con estándares GIS establecidos como WMS, WFS y WCS [53]. Implementa autenticación centralizada, control de transacciones espaciales, y ofrece buena eficiencia para aplicaciones con requerimientos moderados de escalabilidad, aunque puede enfrentar limitaciones ante cargas extremadamente variables que requerirían arquitecturas más distribuidas.

1.3.7 Metodologías de desarrollo

Tabla 5. Comparación de metodologías de desarrollo

Característica	Scrum	XP	Lean	RAD
Enfoque principal	Gestión de proyectos iterativa e incremental	Calidad de código y prácticas de ingeniería	Eliminación de desperdicios y optimización	Desarrollo rápido con prototipos funcionales
Duración de iteraciones	2-4 semanas (Sprints)	1-2 semanas	Variable, flujo continuo	Ciclos cortos de 2-3 semanas por fase

Roles definidos	Product Owner, Scrum Master, Equipo de desarrollo	Coach, Cliente, Programadores, Testers	No roles específicos, enfoque en equipos multifuncionales	Usuarios finales, Especialistas, Desarrolladores, Patrocinadores
Ceremonias/Reuniones	Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective	Planning Game, Stand-up meetings, Retrospectivas	Daily stand-ups, Kanban meetings, Retrospectivas	JAD (Joint Application Development) workshops
Documentación	Mínima, enfocada en el producto	Mínima, código como documentación	Mínima, solo lo necesario	Moderada, modelado y documentación inicial
Participación del cliente	A través del Product Owner	Cliente presente y disponible	Feedback constante del cliente	Alta participación en las fases de diseño
Adaptabilidad a cambios	Alta	Muy alta	Alta	Moderada una vez iniciada la construcción
Escalabilidad a equipos grandes	Requiere adaptaciones (Scrum of Scrums)	Difícil de escalar sin modificaciones	Escalable con Kanban	Escalable con equipos paralelos
Énfasis técnico	Entregables funcionales	Programación en parejas, TDD, Integración continua	Automatización y mejora continua	Herramientas CASE, componentes reutilizables
Adecuado para proyectos	Complejos con requisitos cambiantes	Donde la calidad es crítica	Donde la eficiencia es prioritaria	Con alcance bien definido y plazos cortos
Complejidad de implementación	Media	Alta	Media	Media-Baja
Tiempo de entrega de valor	Incremental cada Sprint	Continuo	Continuo	Rápido al final del ciclo

La metodología RAD (Rapid Application Development) ha sido seleccionada como la más adecuada para el proyecto de migración del sistema SISDE a entorno web, principalmente por las siguientes razones:

- **Alcance bien definido**: El proyecto tiene un alcance claramente establecido: migrar un sistema existente a entorno web manteniendo sus funcionalidades principales, lo que se alinea perfectamente con RAD que funciona mejor cuando los requisitos están bien comprendidos [54].

- **Alta participación de usuarios**: Las entrevistas y el análisis SUS revelaron la importancia de involucrar activamente a los usuarios en el desarrollo para

asegurar una mejor usabilidad, aspecto que RAD enfatiza mediante sesiones JAD (Joint Application Development) [55].

- **Reutilización de componentes**: La migración puede aprovechar componentes existentes del sistema de escritorio y de las bibliotecas seleccionadas (ArcGIS, React, Spring Boot), alineándose con el principio de RAD de maximizar la reutilización [56].

a. Fases de la metodología RAD

La metodología RAD consta de cuatro fases cíclicas, que se aplican para implantar el modelo de desarrollo rápido de aplicaciones, y se detallan a continuación:

- ***Planificación de requisitos.*** Se combina la planificación estratégica con el análisis detallado de requisitos. Los usuarios, desarrolladores y miembros del equipo de TI trabajan juntos para identificar los objetivos de la aplicación y las especificaciones funcionales [56].

- ***Diseño y feedback con el usuario.*** Se caracteriza por la interacción continua con los usuarios para desarrollar modelos y prototipos que representan todos los procesos, entradas y salidas del sistema. Los prototipos evolucionan a través de ciclos iterativos hasta que cumplen con los requisitos establecidos [57].

- ***Construcción.*** Se realiza la construcción del sistema, combinando la programación y pruebas con la continua revisión de los usuarios. RAD enfatiza el desarrollo component-based, donde los elementos reutilizables aceleran el proceso de construcción [56].

- ***Transición.*** La última fase, también conocida como “cutover”, es la transición entre el desarrollo y la implementación en producción, se realizan todas las pruebas que sean necesarias, concluye con la entrega final [57].

1.4 Objetivos

1.4.1 Objetivo general

Migrar el sistema SISDE a entorno web para la Empresa Eléctrica Ambato Regional Centro Norte S.A.

1.4.2 Objetivos específicos

- Definir la estrategia de migración del sistema SISDE a entorno web.
- Implementar el sistema SISDE utilizando tecnologías web.
- Validar los beneficios de la migración del sistema SISDE al entorno web en la EEASA.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para la ejecución de este proyecto de investigación se emplearon diversos recursos tecnológicos y metodológicos que facilitaron la recolección y análisis de información. Se utilizó el System Usability Scale (SUS), aplicado a 60 usuarios del sistema SISDE (contratistas y personal técnico) para obtener métricas cuantificables sobre la experiencia de usuario e identificar deficiencias específicas en términos de usabilidad que debían abordarse en la migración a entorno web.

Adicionalmente, se diseñó y aplicó una entrevista estructurada dirigida al administrador del sistema SISDE en la EEASA, obteniendo información detallada sobre las problemáticas técnicas, expectativas de la migración y requerimientos específicos desde la perspectiva administrativa. Estas herramientas permitieron establecer una base sólida para la selección de tecnologías y el diseño de la solución web propuesta.

2.2 Métodos

2.2.1 Modalidad de la investigación

a. Investigación de Campo

Se realizará una investigación de campo mediante encuestas y entrevistas dirigidas a los usuarios y responsables del sistema SISDE de escritorio. Esto permitirá recopilar información clave sobre sus necesidades y problemáticas, que servirá como base para la propuesta de migración hacia un entorno web.

b. Investigación Bibliográfica - Documental

La investigación bibliográfica-documental se enfocará en la recopilación de información de libros, artículos científicos y estudios previos relacionados con el

desarrollo o la migración de sistemas de escritorio a plataformas web, siendo de utilidad en la fundamentación del presente proyecto de investigación.

2.2.2 Población y muestra

a. Población

La población de este proyecto está conformada por los usuarios del software SISDE de la Empresa Eléctrica de Ambato, que incluye:

- Contratistas: 50 usuarios.
- Desarrolladores y administrativos: 10 usuarios.

Por lo tanto, la población total es:

$$\boldsymbol{N = 50 + 10 = 60}$$

b. Muestra

Dado que la población es pequeña y finita (N = 60), se utilizó la fórmula para calcular el tamaño de la muestra en una población finita:

$$\boldsymbol{n = \frac{N \cdot Z^2 \cdot p \cdot (1 - p)}{e^2 \cdot (N - 1) + Z^2 \cdot p \cdot (1 - p)}}$$

Donde:

- $\boldsymbol{n}$: tamaño de la muestra.
- $\boldsymbol{N}$: tamaño de la población (60).
- $\boldsymbol{Z}$: nivel de confianza (1,96 para un 95 % de confianza).
- $\boldsymbol{p}$: proporción esperada (se asume p = 0,5, que maximiza la varianza).
- $\boldsymbol{e}$: margen de error (5%, es decir, 0,05).

Desarrollando:

$$n = \frac{60 \cdot (1.96)^2 \cdot 0.5 \cdot (1 - 0.5)}{(0.05)^2 \cdot (60 - 1) + (1.96)^2 \cdot 0.5 \cdot (1 - 0.5)}$$

$$n \approx 52.01$$

$$\therefore n = 52$$

El tamaño de la muestra calculada es $n = 52$. Sin embargo, dado que la población total es de solo 60 usuarios y la muestra requerida es cercana al total, incluir toda la población garantiza resultados más precisos y fiables, y en este caso es una opción viable porque el grupo total no es demasiado grande para gestionarlo.

2.2.3 Recolección de información

a. System Usability Scale (SUS)

El System Usability Scale (SUS) es una herramienta validada ampliamente aceptada para medir la usabilidad de sistemas, productos o servicios. Consta de 10 afirmaciones que los usuarios califican en una escala de 1 (totalmente en desacuerdo) a 5 (totalmente de acuerdo) como se observa en la Tabla 6:

Tabla 6. Escala de valores de puntuación SUS

Escala	Valor
Totalmente en desacuerdo	1
En desacuerdo	2
Neutral	3
De acuerdo	4
Totalmente de acuerdo	5

Estas afirmaciones alternan entre formulaciones positivas y negativas para minimizar sesgos, las cuales son las siguientes:

- Creo que me gustaría usar este sistema con frecuencia.
- Encontré el sistema innecesariamente complejo.

- Pensé que el sistema era fácil de usar.
- Creo que necesitaría la ayuda de un técnico para poder utilizar el sistema.
- Descubrí que las diversas funciones de este sistema estaban bien integradas.
- Pensé que había demasiada inconsistencia en este sistema.
- Imagino que la mayoría de las personas aprenderían a utilizar este sistema muy rápidamente.
- Encontré el sistema muy complicado de usar.
- Me sentí muy confiado usando el sistema.
- Necesité aprender muchas cosas antes de poder empezar a usar este sistema.

El puntaje se calcula convirtiendo las respuestas en una escala de 0 a 100: para las afirmaciones positivas, se resta 1 al valor dado por el usuario, y para las negativas, se resta la respuesta de 5; la suma de los valores ajustados se multiplica por 2.5 para obtener el puntaje final. Un puntaje promedio de 68 es considerado aceptable, mientras que valores superiores reflejan una mejor usabilidad [58].

En este contexto, el SUS será utilizado para comparar la usabilidad del sistema SISDE de escritorio en su versión de escritorio con la nueva versión propuesta en entorno web. Esto permitirá medir de manera objetiva las mejoras en términos de facilidad de uso, accesibilidad y satisfacción del usuario, validando la efectividad de la migración tecnológica hacia un entorno web.

b. Entrevista para Administrativos

La entrevista tiene como objetivo identificar requerimientos técnicos y funcionales para la implementación del nuevo software basado en la web. Por lo que se han propuesto las siguientes preguntas:

- ¿Cuáles son los principales problemas que los usuarios reportan sobre el software actual?

- ¿Qué tipo de funcionalidades consideran esenciales para el nuevo software basado en la web?
- ¿Qué tan importante es para la empresa la capacidad de trabajar de forma remota con el software?
- ¿Qué datos o procesos del sistema SISDE de escritorio son críticos y no deben perderse en la migración?
- ¿Cuáles son las expectativas en términos de tiempo de respuesta y procesamiento del nuevo sistema?
- ¿Existen preocupaciones específicas de seguridad para la gestión de datos geoespaciales en el nuevo entorno?
- ¿Cómo se espera que el nuevo software impacte en los tiempos de actualización de proyectos?
- ¿Cuáles serían las métricas clave para evaluar el éxito del nuevo sistema?
- ¿Qué tan importante es la compatibilidad con dispositivos móviles o tabletas?
- ¿Qué nivel de capacitación se planea ofrecer a los usuarios para la transición al nuevo sistema?
- ¿Qué funcionalidades del software actual consideran obsoletas y no deberían incluirse en la nueva versión?
- ¿Qué tecnologías o plataformas están considerando para el soporte técnico del nuevo software?
- ¿Qué tan frecuentemente necesitarían que el sistema reciba actualizaciones o mejoras una vez implementado?
- ¿Cómo visualizan la interacción del nuevo sistema con otras herramientas o sistemas existentes en la empresa?

- ¿Existen recomendaciones específicas que les gustaría añadir para garantizar el éxito del proyecto?

2.2.4 Procesamiento y análisis de datos

a. Análisis SUS

Tras aplicar la encuesta a los usuarios se obtienen los datos en el Anexo A y estos resultados se agrupan en la Tabla 7, donde los puntajes reflejan la facilidad de uso y satisfacción de los usuarios.

Tabla 7. Resultados de la encuesta SUS en el sistema SISDE de escritorio

Categoría SUS	Rango de Puntaje	Número de Participantes	Porcentaje
F	0 - 51.6	9	15%
D	51.7 - 62.6	18	30%
C-	62.7 - 64.9	8	13.33%
C	65 - 71	24	40%
B	74.1 - 77.1	1	1.67%
TOTAL	-	60	100%

El análisis de las respuestas del SUS revela aspectos críticos sobre la usabilidad del sistema SISDE de escritorio que demandan atención inmediata. Los resultados muestran que el 45% de los usuarios califican el sistema con puntajes problemáticos (categorías F y D), indicando serias deficiencias en la experiencia de usuario. Esta distribución refleja que una proporción significativa de usuarios enfrenta dificultades sustanciales al interactuar con la plataforma actual, lo que impacta directamente en su productividad y satisfacción laboral.

La concentración del 40% de usuarios en la categoría C sugiere que, aunque el sistema es funcionalmente utilizable, presenta limitaciones importantes que impiden una experiencia óptima. Esta categoría representa usuarios que pueden completar sus tareas, pero experimentan fricciones, inconsistencias o complejidades innecesarias que dificultan su trabajo diario. El hecho de que solo el 1.67% de usuarios califique el sistema como "bueno" (categoría B) y ninguno lo considere excelente, evidencia la necesidad urgente de mejoras sustanciales en la usabilidad.

La ausencia de usuarios en las categorías superiores (B+, A-, A, A+) es particularmente reveladora, ya que indica que incluso los usuarios más experimentados o tolerantes no consideran que el sistema actual proporcione una experiencia satisfactoria. Esta situación sugiere problemas fundamentales en el diseño de la interfaz, la organización de funcionalidades y la eficiencia de los flujos de trabajo, que van más allá de simples ajustes superficiales.

El puntaje promedio de 61.04 puntos sitúa al sistema SISDE de escritorio en la categoría D, clasificándolo como "por debajo del promedio" según los estándares internacionales de usabilidad. Este resultado indica que el sistema presenta deficiencias significativas que afectan la experiencia del usuario y justifican plenamente la decisión de migrar hacia una solución web moderna. La distribución de puntajes revela una experiencia de usuario consistentemente problemática que requiere una transformación integral.

b. Análisis de la entrevista

Tras aplicar la entrevista al principal administrador del software SISDE, Ing. Iván Vargas, se obtienen las siguientes respuestas y el análisis de cada pregunta en la Tabla 8.

Tabla 8. Preguntas y respuestas de la entrevista aplicada al administrador del sistema SISDE de escritorio

Pregunta	Respuesta	Análisis
¿Cuáles son los principales problemas que los usuarios reportan sobre el software actual?	Problemas con actualización (desplazamientos físicos y reinstalaciones manuales), sistema poco intuitivo, tiempos de carga elevados e inconsistencias en procesamiento de datos geoespaciales.	Los problemas identificados afectan directamente la operatividad y eficiencia. Las actualizaciones manuales representan un costo significativo en tiempo y recursos, mientras que la interfaz poco intuitiva impacta negativamente en la experiencia del usuario. Los tiempos de carga extensos disminuyen la productividad diaria.
¿Qué tipo de funcionalidades consideran esenciales para el nuevo software basado en la web?	Actualizaciones automáticas, acceso remoto, interfaz intuitiva, herramientas avanzadas de visualización 2D, gestión eficiente de materiales y colaboración en tiempo real.	Las funcionalidades solicitadas apuntan directamente a resolver los problemas actuales, con énfasis en automatización, accesibilidad y colaboración. La visualización 2D y gestión de materiales sugieren necesidades específicas del sector. La colaboración en tiempo real indica un enfoque hacia la eficiencia operativa colectiva.

¿Qué tan importante es para la empresa la capacidad de trabajar de forma remota con el software?	Fundamental, ya que los contratistas están distribuidos en varias provincias y necesitan acceso sin depender de desplazamientos a la empresa.	La dispersión geográfica de los contratistas convierte el acceso remoto en una prioridad crítica para la operatividad. Esto sugiere que la arquitectura web debe optimizarse para conexiones estables incluso en áreas con conectividad variable, considerando la dispersión provincial.
¿Qué datos o procesos del sistema SISDE de escritorio son críticos y no deben perderse en la migración?	Ubicación geoespacial de proyectos, inventarios de materiales, historial de proyectos, procesos de exportación y generación de reportes.	Los datos mencionados representan el núcleo operativo del sistema. La preservación de información geoespacial e inventarios es crucial para la continuidad operativa. El historial de proyectos indica la necesidad de mantener trazabilidad, mientras que los procesos de exportación y reportes sugieren funcionalidades de análisis y documentación esenciales.
¿Cuáles son las expectativas en términos de tiempo de respuesta y procesamiento del nuevo sistema?	Reducción significativa: operaciones básicas no deberían superar unos segundos.	La expectativa de respuesta rápida revela frustración con los tiempos actuales. Esta métrica de "segundos" establece un umbral concreto para el rendimiento que debe cumplir el nuevo sistema, lo que impactará decisiones sobre arquitectura, capacidad de servidores y optimización de bases de datos.
¿Existen preocupaciones específicas de seguridad para la gestión de datos geoespaciales en el nuevo entorno?	Protección de datos sensibles, control de acceso basado en roles, encriptación de datos en tránsito y reposo.	Las preocupaciones de seguridad reflejan un entendimiento sofisticado de los riesgos. El control basado en roles sugiere una estructura organizacional jerárquica que debe reflejarse en el diseño de permisos. La mención de encriptación tanto en tránsito como en reposo indica conciencia sobre vulnerabilidades en múltiples estados de los datos.
¿Cómo se espera que el nuevo software impacte en los tiempos de actualización de proyectos?	Actualizaciones inmediatas y transparentes, eliminando retrasos del modelo de distribución manual.	La expectativa de inmediatez sugiere que los usuarios experimentan actualmente retrasos significativos que afectan su trabajo. La transparencia mencionada indica necesidad de visibilidad en el proceso, posiblemente con notificaciones o registros de cambios accesibles.
¿Cuáles serían las métricas clave para evaluar el éxito del nuevo sistema?	Reducción de tiempo de actualización, disminución de costos operativos (soporte y desplazamientos), capacidad de escalar para más usuarios.	Las métricas identificadas son cuantificables y orientadas a resultados empresariales tangibles. La reducción de tiempos y costos sugiere un enfoque en ROI, mientras que la escalabilidad indica visión de crecimiento futuro y preocupación por la sostenibilidad de la solución.
¿Qué tan importante es la compatibilidad con dispositivos móviles o tabletas?	No es prioritario. Enfoque en plataforma web, aunque debe ser responsiva para acceso ocasional desde dispositivos móviles.	La baja prioridad de compatibilidad móvil contrasta con tendencias actuales, sugiriendo un entorno de trabajo principalmente de escritorio. Sin embargo, la mención de responsividad para acceso ocasional reconoce cambios graduales en patrones de uso y prevé necesidades futuras sin comprometer recursos inmediatos.
¿Qué nivel de capacitación se planea ofrecer a los usuarios para	Talleres presenciales y capacitación virtual para todos los usuarios.	El enfoque dual de capacitación (presencial y virtual) demuestra comprensión de diferentes necesidades de aprendizaje y reconoce la distribución geográfica

la transición al nuevo sistema?		mencionada anteriormente. La planificación proactiva de capacitación sugiere conciencia sobre la resistencia al cambio y desafíos de adopción.
¿Qué funcionalidades del software actual consideran obsoletas y no deberían incluirse en la nueva versión?	Herramientas redundantes de exportación de datos y procesos manuales que pueden automatizarse.	La identificación de elementos obsoletos muestra disposición a optimizar y no solo replicar el sistema de escritorio. La mención específica de exportaciones redundantes y procesos manuales sugiere áreas concretas donde la automatización podría generar ganancias inmediatas de eficiencia.
¿Qué tecnologías o plataformas están considerando para el soporte técnico del nuevo software?	ArcGIS Online para alojamiento y soporte técnico, más herramientas de monitoreo en tiempo real.	La mención de ArcGIS Online revela familiaridad con soluciones específicas del sector geoespacial. El interés en monitoreo en tiempo real sugiere necesidad de visibilidad operativa continua, posiblemente para identificar y resolver problemas rápidamente.
¿Qué tan frecuentemente necesitarían que el sistema reciba actualizaciones o mejoras una vez implementado?	Actualizaciones trimestrales para mejoras funcionales y corrección de errores, con revisiones mayores anuales.	El ciclo de actualización propuesto es estructurado y muestra planificación a largo plazo. La diferenciación entre actualizaciones trimestrales y revisiones anuales sugiere comprensión de diferentes niveles de cambio y su impacto operativo, facilitando la planificación de recursos técnicos.
¿Cómo visualizan la interacción del nuevo sistema con otras herramientas o sistemas existentes en la empresa?	Integración con software de gestión de inventarios y herramientas de planificación de proyectos mediante APIs.	La necesidad de integración revela un ecosistema tecnológico más amplio que debe considerarse. La mención específica de APIs sugiere preferencia por enfoques modernos de integración, con énfasis en inventarios y planificación como puntos críticos de conexión entre sistemas.
¿Existen recomendaciones específicas que les gustaría añadir para garantizar el éxito del proyecto?	Comunicación constante con usuarios durante el desarrollo, pruebas piloto previas al lanzamiento y diseño escalable para adaptarse a futuras necesidades.	Las recomendaciones reflejan enfoque en factores críticos de éxito más allá de la tecnología. La comunicación con usuarios y pruebas piloto sugieren valoración del feedback y reducción de riesgos. La escalabilidad mencionada refuerza la visión a largo plazo y preocupación por sostenibilidad de la inversión tecnológica.

El análisis de las entrevistas revela desafíos significativos con el sistema SISDE de escritorio, principalmente relacionados con su arquitectura no web que requiere actualizaciones manuales e impone limitaciones de acceso remoto. Los usuarios expresan frustración con la interfaz poco intuitiva y los tiempos de respuesta lentos, lo que impacta directamente en la productividad operativa. La distribución geográfica de los contratistas en diferentes provincias intensifica estos problemas, creando ineficiencias logísticas y costos innecesarios.

Los requerimientos para el nuevo sistema se centran en tres pilares fundamentales: accesibilidad remota, automatización de procesos y una experiencia de usuario mejorada. Existe claridad sobre la información crítica que debe preservarse durante la migración, particularmente datos geoespaciales, inventarios de materiales e historiales de proyectos. Las expectativas de rendimiento son específicas, con una demanda clara de respuestas en "segundos" para operaciones básicas.

En términos de implementación, existe un claro reconocimiento de la importancia de la capacitación y comunicación constante con los usuarios para facilitar la adopción. La preferencia por una implementación con pruebas piloto refleja un enfoque prudente orientado a minimizar riesgos y recopilar retroalimentación antes de un despliegue completo.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Análisis y Discusión de resultados

Las tecnologías seleccionadas para la migración del sistema SISDE a entorno web fueron determinadas mediante un análisis comparativo exhaustivo presentado anteriormente. Como resultado de este análisis, se definió un stack tecnológico compuesto por React con JSX para el frontend debido a su excelente integración con ArcGIS SDK y capacidades de componentización, Spring Boot para el backend por su compatibilidad nativa con WebLogic y Oracle, ArcGIS como plataforma GIS aprovechando la familiaridad existente en EEASA, y Oracle como base de datos geoespacial eliminando la necesidad de migración de datos.

3.1.1 Análisis global de la solución propuesta

Una vez definida la estrategia de migración del sistema SISDE a entorno web mediante la selección del stack tecnológico óptimo, es fundamental realizar un análisis comparativo detallado entre los procesos operativos del sistema SISDE de escritorio y los procesos que se implementaron con la nueva solución web, que se detallan a continuación:

a. Análisis de los procesos del sistema SISDE

- ***Análisis de los procesos del sistema SISDE de escritorio.*** El proceso actual de actualización del sistema SISDE, mostrado en la Figura 1, se caracteriza por ser manual y presencial, generando importantes ineficiencias operativas. El mismo requiere que los contratistas se desplacen físicamente hasta las instalaciones de la EEASA para recibir actualizaciones, lo que ocasiona costos de transporte y pérdida de tiempo productivo. La instalación individualizada consume recursos técnicos valiosos y crea tiempos de inactividad prolongados en los proyectos. Este modelo también puede generar inconsistencias entre versiones y complica la resolución de problemas, que frecuentemente requiere asistencia presencial, extendiendo aún más las interrupciones laborales.

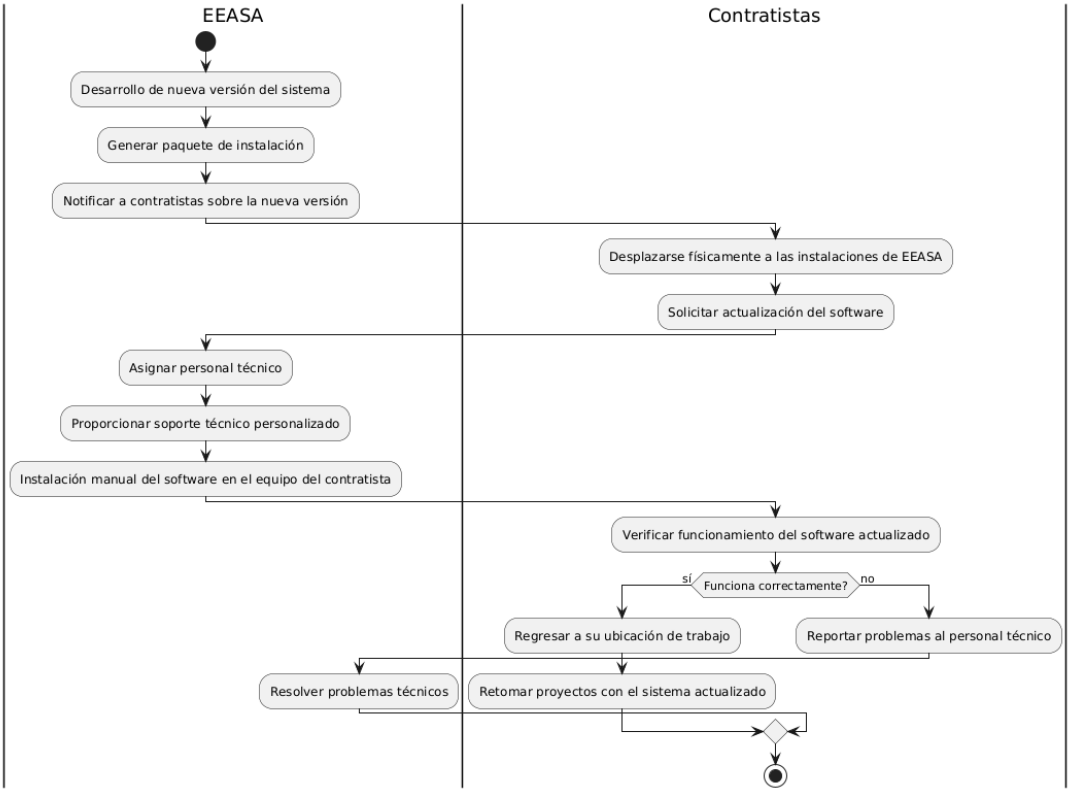


Figura 1. Procesos del sistema SISDE de escritorio

- ***Análisis de los procesos propuestos del sistema SISDE Web.*** El proceso propuesto con la migración a entorno web, ilustrado en la Figura 2, transforma radicalmente la experiencia de actualización mediante un enfoque centralizado y automatizado. Las actualizaciones se realizan una sola vez en el servidor central y se distribuyen instantáneamente a todos los usuarios, quienes pueden acceder al sistema desde cualquier ubicación con internet. La actualización es transparente y requiere solo segundos para completarse, minimizando el tiempo de inactividad. Todos los usuarios trabajan siempre con la misma versión, eliminando problemas de compatibilidad. La resolución de problemas ocurre remotamente, sin necesidad de interacciones presenciales, y las soluciones se implementan para todos los usuarios simultáneamente. Esta transformación optimiza recursos, reduce costos y mejora sustancialmente la satisfacción de los usuarios.

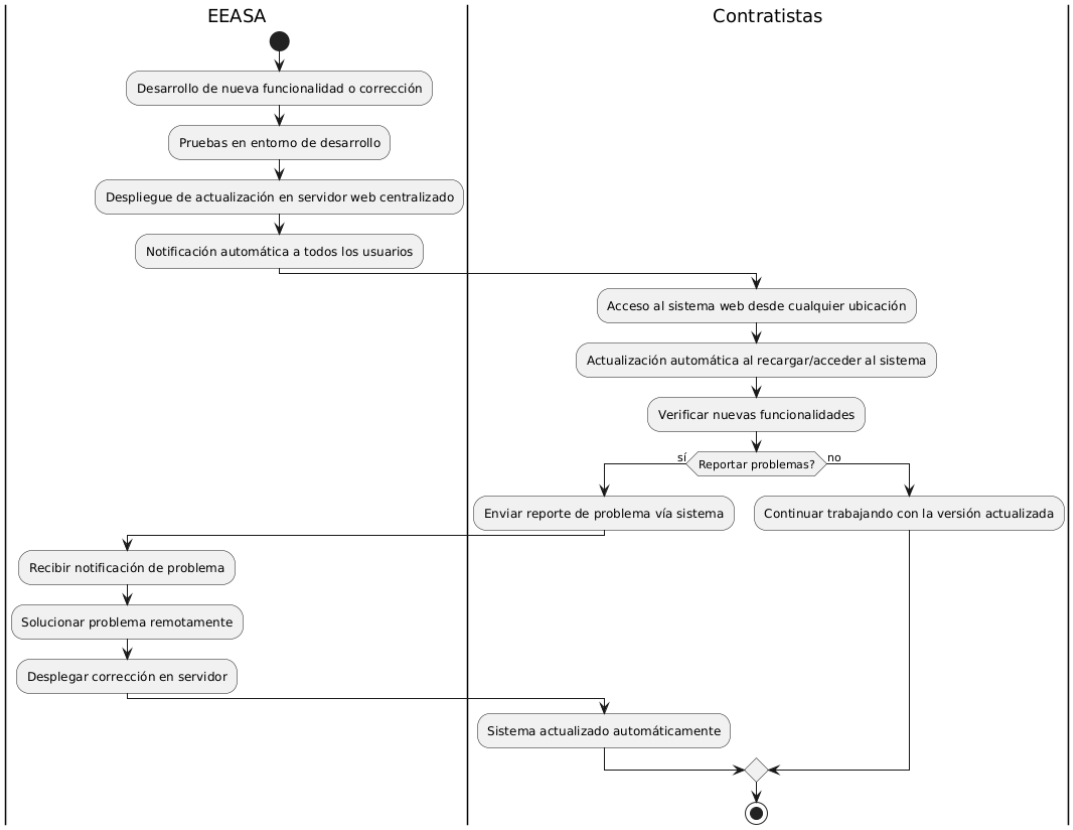


Figura 2. Procesos propuestos del sistema SISDE Web

b. Análisis de la solución tecnológica

La combinación de tecnologías seleccionada representa una solución integral que aborda directamente los problemas identificados en el sistema SISDE de escritorio:

- **Actualizaciones remotas y automáticas**: La arquitectura web permitirá que las actualizaciones se realicen de forma centralizada, eliminando la necesidad de desplazamientos físicos de los contratistas, tal como se identificó en la entrevista con los usuarios.

- **Mejora de usabilidad**: Los resultados del System Usability Scale (SUS) revelaron problemas significativos con la interfaz actual. La implementación con React y los sistemas de diseño Calcite y Material UI proporcionará una interfaz más intuitiva y consistente.

- **Rendimiento optimizado**: Las quejas sobre tiempos de carga fueron recurrentes en la evaluación del sistema SISDE de escritorio. La arquitectura

propuesta, con Spring Boot en el backend y React en el frontend, está optimizada para tiempos de respuesta rápidos, cumpliendo con la expectativa expresada de operaciones completadas en "segundos".

- **Compatibilidad con infraestructura existente**: La selección de tecnologías que son compatibles con la infraestructura ya implementada en EEASA (ArcGIS, Oracle, WebLogic) minimiza los riesgos de implementación y reduce los costos asociados con cambios en la infraestructura.

La propuesta de migración no solo aborda los problemas actuales, sino que también establece una base tecnológica moderna y escalable, se puede identificar la comparación en la arquitectura actual del sistema SISDE de escritorio en la Figura 3 con la arquitectura web propuesta, que podrá adaptarse a las necesidades futuras de la empresa, tal como se solicitó en las recomendaciones específicas durante la entrevista.

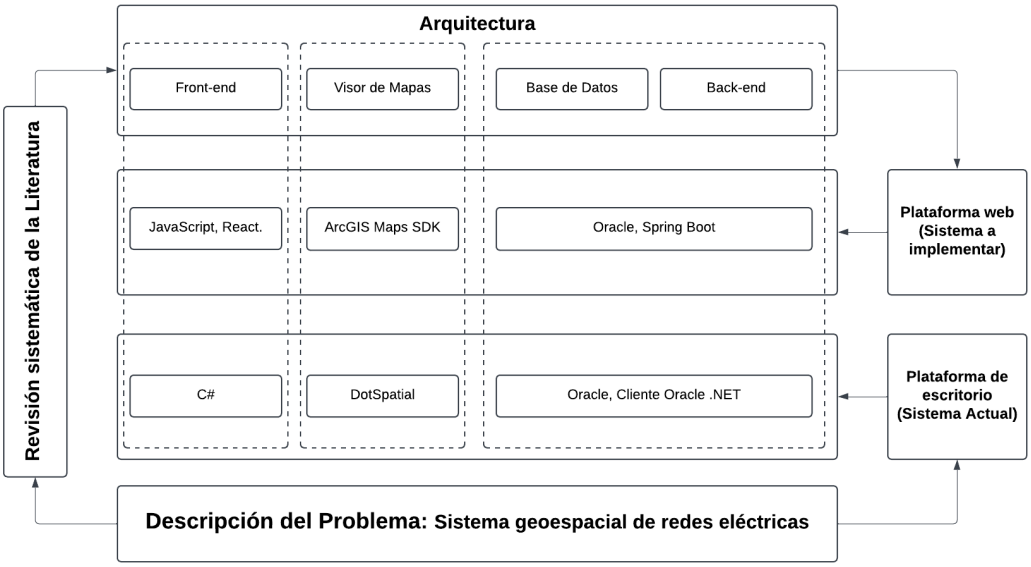


Figura 3. Comparación de las arquitecturas de los sistemas SISDE de escritorio y web

3.2 Desarrollo de la propuesta

3.2.1 Planificación de requisitos

La fase de Planificación de requisitos en la metodología RAD constituye una etapa fundamental que establecerá las bases para el desarrollo exitoso del sistema SISDE en entorno web. Durante esta fase, se llevará a cabo un análisis exhaustivo de las necesidades de la Empresa Eléctrica Ambato en relación con los inconvenientes que presenta el sistema SISDE de escritorio, involucrando activamente a usuarios clave y personal técnico. A continuación, se presenta el detalle de los elementos esenciales identificados durante esta fase:

a. Roles del Proyecto

Se han establecido la intervención de tres participantes en el desarrollo del proyecto, sus roles y funciones específicas se detallan en la Tabla 9.

Tabla 9. Responsables del desarrollo del proyecto de investigación

Responsable	Rol	Función
Joan Salán	Desarrollador	Responsable del diseño, desarrollo, pruebas y despliegue del sistema web SISDE
Ing. Félix Fernández	Supervisor	Proporciona orientación metodológica y técnica al proyecto, supervisa el cumplimiento de tareas y revisa los entregables en cada fase del ciclo RAD.
Ing. Iván Vargas	Cliente	Administrador del sistema SISDE de escritorio y principal representante de la EEASA. Define los requerimientos, proporciona información detallada sobre la operativa actual y valida los prototipos y entregables durante las sesiones JAD. Facilita el acceso a la infraestructura existente y es responsable final de la aceptación del sistema migrado a entorno web.

b. Requerimientos técnicos

Los requerimientos técnicos se han definido considerando la infraestructura tecnológica ya existente en la EEASA. A continuación, en la Tabla 10 se detallan las tecnologías base que soportarán el sistema SISDE en su versión web:

Tabla 10. Requerimientos técnicos de la EEASA

Tecnología	Versión	Descripción
Oracle Database	19c	Gestor de Base de Datos
Oracle WebLogic Server	12.2.1.4	Servidor web.
Java	1.8.0	Lenguaje de programación servidor.

c. Requerimientos funcionales y no funcionales

El listado de requerimientos para la migración del sistema SISDE a entorno web fue recolectado mediante una entrevista con el administrador del sistema (gestión técnica y mantenimiento de la plataforma), encuestas SUS aplicadas a los usuarios actuales (quienes utilizan el software para la elaboración de proyectos de red eléctrica), y un análisis detallado del sistema existente. Esta metodología se alinea con el enfoque RAD para involucrar activamente a los usuarios finales y obtener un entendimiento claro de las necesidades y expectativas.

Se ha definido la siguiente estructura para la enumeración de los requisitos descritas en la Tabla 11 y

Tabla 12:

- **Código:** Identificador del requerimiento, con la siguiente estructura:

  RF-XX: Requerimientos Funcionales

  RNF-XX: Requerimientos No Funcionales

- **Descripción:** Explicación detallada del requerimiento.
- **Prioridad:** Se rige a la escala: alta, media y baja.
- **Riesgo:** Se rige a la escala: alto, medio y bajo.

Tabla 11. Requerimientos funcionales SISDE Web

Código	Descripción	Prioridad	Riesgo
RF-01	Permitir la gestión de proyectos de red eléctrica con datos geoespaciales.	Alta	Medio

RF-02	Implementar un inventario de materiales (y clasificación).	Alta	Medio
RF-03	Implementar un inventario de estructuras (y clasificación).	Alta	Medio
RF-04	Permitir asociar materiales específicos a estructuras de proyectos.	Alta	Bajo
RF-05	Permitir la clonación de una estructura con sus materiales asociados.	Media	Bajo
RF-06	Implementar drag and drop para cargar ficheros excel con estructura determinada para la actualización de precios de mano de obra y materiales.	Media	Medio
RF-07	Implementar un sistema de autenticación seguro para acceso al sistema web.	Alta	Alto
RF-08	Permitir la gestión de usuarios con diferentes niveles de permisos a través de diferentes tipos de roles.	Alta	Alto
RF-09	Proporcionar control de acceso basado en roles con diferentes niveles de permisos.	Alta	Alto
RF-10	Permitir la gestión de datos personales permitidos de los usuarios.	Media	Medio
RF-11	Implementar visualización 2D de elementos de red eléctrica en el mapa (postes, transformadores, tramos, etc.).	Alta	Medio
RF-12	Permitir visualizar características de los elementos gráficos.	Alta	Bajo
RF-13	Proporcionar herramientas de medición espacial para elementos de red eléctrica.	Media	Medio
RF-14	Implementar herramientas de búsqueda y filtrado de elementos geoespaciales.	Media	Bajo
RF-15	Implementar funciones de zoom y navegación en el mapa.	Alta	Bajo
RF-16	Mostrar las capas de elementos geoespaciales en el mapa.	Alta	Medio
RF-17	Proporcionar herramientas para dibujar y editar elementos geoespaciales (puntos y líneas) con la selección de características propias del elemento a graficar.	Alta	Medio
RF-18	Verificar al graficar los elementos que requieren de otros elementos.	Media	Medio
RF-19	Permitir asociar estructuras específicas a elementos de red eléctrica en el mapa.	Alta	Medio
RF-20	Permitir la selección y edición de características de los elementos gráficos.	Alta	Medio
RF-21	Selección del tipo de red: Red existente o proyectada.	Alta	Bajo
RF-22	Generar reportes de los proyectos.	Alta	Bajo
RF-23	Permitir la carga y visualización de capas geoespaciales externas (archivos SHP).	Media	Medio
RF-24	Permitir la exportación del proyecto en formatos PDF, JPG y PNG para su posterior impresión.	Media	Medio

Tabla 12. Requerimientos no funcionales SISDE Web

Código	Descripción	Prioridad	Riesgo
RNF-01	El sistema debe permitir actualizaciones automáticas y centralizadas sin requerir desplazamiento físico de los usuarios.	Alta	Alto
RNF-02	La interfaz de usuario debe ser intuitiva y seguir los principios de usabilidad, mejorando la experiencia del sistema de escritorio.	Alta	Medio
RNF-03	El tiempo de respuesta para operaciones básicas no debe exceder los 3 segundos.	Alta	Alto

RNF-04	El sistema debe ser responsivo para permitir acceso ocasional desde dispositivos móviles, aunque no es prioridad central.	Media	Bajo
RNF-05	La aplicación debe soportar los navegadores web modernos (Chrome, Firefox, Edge).	Alta	Bajo
RNF-06	El sistema debe implementar validación de entradas para prevenir ataques como SQL Injection.	Alta	Alto
RNF-07	La aplicación debe utilizar carga diferida (lazy loading) para optimizar los tiempos de carga de capas y datos geoespaciales.	Media	Medio
RNF-08	El sistema debe gestionar sesiones de usuario de forma segura, con tiempo de inactividad configurable.	Alta	Medio
RNF-09	El sistema debe mostrar mensajes de error claros y específicos para facilitar la solución de problemas.	Media	Bajo

d. Plan de entrega

En base a los requerimientos funcionales y no funcionales recolectados previamente, se propone un plan de entrega como se observa en la Tabla 13, considerando las dependencias entre requerimientos, complejidad técnica y distribución equilibrada entre iteraciones.

Tabla 13. Plan de entrega por iteraciones

Iteración	Código	Requerimientos	Tiempo estimado	
			Horas	Días
Iteración 1	RF-11, RF-15, RF-16	Visualización 2D de elementos y navegación básica del mapa	24	3
	RF-17, RF-18	Herramientas para dibujar y editar elementos geoespaciales	20	2.5
	RNF-02, RNF-05	Interfaz intuitiva y soporte para navegadores modernos	12	1.5
Iteración 2	RF-12, RF-13, RF-14	Visualización de características y herramientas de medición/búsqueda	20	2.5
	RF-20, RF-21	Edición de características y selección de tipo de red	16	2
	RF-23, RNF-07	Carga de capas externas y optimización con lazy loading	16	2
Iteración 3	RF-01, RF-22, RF-24	Gestión de proyectos y generación de reportes/exportación	24	3
	RNF-01, RNF-03	Actualizaciones centralizadas y optimización de tiempos de respuesta	16	2
	RNF-04, RNF-09	Diseño responsivo y mensajes de error claros	12	1.5
Iteración 4	RF-02, RF-03	Inventario de materiales y estructuras	20	2.5
	RF-04, RF-05	Asociación de materiales a estructuras y clonación	16	2
	RF-06, RF-19	Actualización de precios y asociación a elementos de red	16	2
Iteración 5	RF-07, RF-08	Sistema de autenticación y gestión de usuarios	16	2

	RF-09, RF-10	Control de acceso basado en roles y gestión de datos personales	16	2
	RNF-06, RNF-08	Validación de entradas y gestión segura de sesiones	16	2

e. Arquitectura del sistema

La arquitectura del sistema SISDE para entorno web se ha diseñado siguiendo un enfoque de múltiples capas que optimiza la experiencia del usuario mientras aprovecha la infraestructura existente de la EEASA. A continuación, se detalla cada componente de la arquitectura en la Figura 4:

- ***Cliente.*** Interfaz de usuario desarrollada con React y JSX, que permite a los usuarios interactuar con las funcionalidades del sistema desde cualquier navegador web moderno. Utiliza componentes de ArcGIS Maps SDK para JavaScript para la visualización y manipulación de datos geoespaciales, y frameworks de diseño como Calcite Components y Material UI para proporcionar una experiencia coherente y usable.

- ***Servidor.*** Implementado con Spring Boot para gestionar la lógica de negocio, procesamiento de datos y seguridad. Este componente centraliza todas las operaciones del sistema y gestiona las comunicaciones entre la interfaz de usuario y los sistemas de almacenamiento.

- ***Comunicación.*** Se utilizan servicios RESTful implementados en Spring Boot para facilitar el intercambio de datos entre el cliente React y el servidor. Estos servicios siguen principios REST para garantizar operaciones eficientes, seguras y escalables.

- ***Servidor de Aplicaciones.*** Oracle WebLogic Server actúa como entorno de ejecución para los componentes del servidor, proporcionando alta disponibilidad, balanceo de carga y gestión eficiente de recursos. Los componentes Spring Boot se despliegan como archivos WAR dentro de WebLogic.

- ***Base de Datos.*** Oracle Database almacena todos los datos del sistema, incluyendo información geoespacial, inventarios de materiales, datos de usuarios y proyectos.

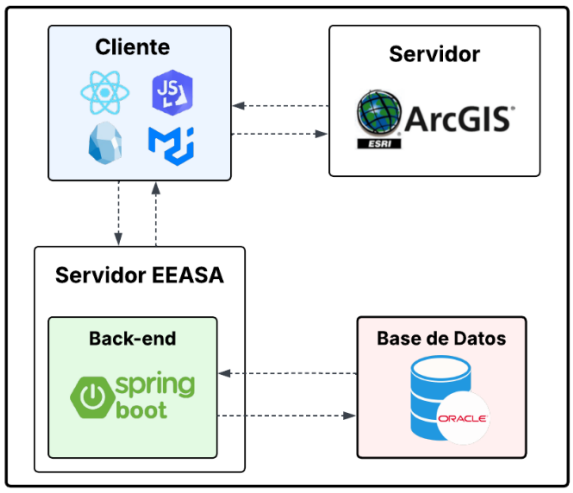


Figura 4. Arquitectura SISDE Web

f. Diseño de la base de datos

El sistema SISDE utiliza una base de datos Oracle robusta que ha sido diseñada específicamente para la gestión de redes eléctricas. Esta base de datos contiene numerosas tablas relacionadas con diferentes aspectos del diseño y gestión de infraestructura eléctrica, estructuradas en varios grupos funcionales.

Estructura general de la base de datos

La base de datos del sistema SISDE contiene más de 60 tablas organizadas en diferentes grupos funcionales:

- **Administración de usuarios y seguridad**: Tablas para gestión de usuarios, roles, permisos y autenticación.
- **Diseños de redes eléctricas**: Tablas para almacenar información sobre proyectos de diseño, estados y aprobaciones.
- **Geoespacial**: Tablas para la gestión de elementos geográficos, incluyendo features, subtipos y dominios.
- **Infraestructura eléctrica**: Múltiples tablas especializadas para diferentes componentes de la red (postes, transformadores, tensores, etc.).

- **Materiales y estructuras**: Tablas para la gestión de inventarios de materiales, clasificaciones y precios.

- **Divisiones administrativas**: Tablas para provincias, cantones y parroquias.

La base de datos también incorpora una lógica de negocio robusta implementada a través de paquetes de procedimientos almacenados, procedimientos individuales, funciones y triggers que manejan operaciones CRUD y validaciones específicas del dominio eléctrico, por tanto, se ha optado por utilizar la base de datos existente.

Sin embargo, se han realizado importantes modificaciones en la estructura de la misma para mejorar su diseño, normalización y seguridad, principalmente en los aspectos detallados en la Tabla 14:

Tabla 14. Rediseño de tablas de la base de datos existente del sistema SISDE escritorio

Tipo	Tabla	Descripción	Detalle
Gestión de Usuarios	SD_USUARIOS	Rediseño completo de la tabla	• Se añadieron campos para gestión de credenciales (USERNAME, PASSWORD) • Mejora en almacenamiento de datos personales (NOMBRE1, NOMBRE2, APELLIDO1, APELLIDO2, EMAIL) • Incorporación de campos para control de seguridad (ESTADO, FECHA_CAMBIO_ESTADO, INTENTOS_FALLIDOS, FECHA_MAX_ACCESO)
	SD_USUARIOS_ESTADOS	Nueva tabla para estados de cuenta	• Formaliza los posibles estados: Activo (usuario habilitado), Bloqueado (por intentos fallidos), Eliminado (usuario eliminado del sistema)
	SD_USUARIOS_ROLES	Nueva tabla para relación usuarios-roles	• Implementa relación muchos a muchos con campos: ID_USUARIO, ID_ROL, FECHA_ASIGNACION
	SD_PERMISOS	Nueva tabla para gestión de permisos	• Contiene: ID_PERMISO (identificador único), NOMBRE_PERMISO (nombre descriptivo), DESCRIPCION (detalle),

			ACCION (acción específica permitida)
	SD_ROLES	Reestructuración de tabla (antes SD_PERFILES)	• Implementación de control de acceso basado en roles; definición de roles predeterminados: Administrador, Supervisor, Fiscalizador y Diseñador
	SD_ROLES_PERMISOS	Nueva tabla para relación roles-permisos	• Establece la relación muchos a muchos entre roles y permisos
Gestión de diseños	SD_DISENOS	Modificación para mejor gestión de proyectos	• Eliminación de tablas redundantes (SD_USUARIOS y SD_DISENADORES) • Adición de campos para fiscalizadores y diseñadores mediante relaciones con SD_USUARIOS.
	SD_DISENOS_PERMISOS	Rediseño para permisos específicos a nivel de diseño	• Incluye: ID_USUARIO (usuario con permiso), TIPO_PERMISO (tipo otorgado), ID_USUARIO_ASIGNADOR (usuario que otorgó el permiso)
Sistema Geoespacial	SD_FEATURE_CLASS	Enriquecimiento con nuevos campos	• Adición de: DISPLAY_NAME (nombre amigable para interfaz), GEOMETRY_TYPE (tipo de geometría: PUNTO o LINEA), FEATURE_DEPENDENCY (referencia a dependencias entre features)
	SD_FEATURE_FIELDS	Nueva tabla para unificación de gestión de campos	• Reemplaza las tablas SD_CAMPOS y SD_CAMPOS_SUBTIPOS • Permite gestionar campos fijos y variables según subtipo • Incorpora validaciones y tipos de datos
	SD_TIPOS_ESTILO	Nueva tabla para gestión de representación visual	• Tipos: simple-line (elementos lineales), simple-marker (elementos puntuales), picture-marker (elementos con representación gráfica personalizada)
	SD_SUBTIPOS_ESTILOS	Nueva tabla para configuración visual de subtipos	• Gestiona colores, anchos y desplazamientos según tipo de elemento • Proporciona control visual para evitar solapamiento gráfico

Estos cambios en la base de datos establecen una estructura más robusta, normalizada y segura para soportar la migración del sistema SISDE a entorno web, como se ilustra en las figuras: Figura 5 y Figura 6. Estos diagramas representan visualmente las relaciones entre las tablas principales, facilitando la implementación de nuevas funcionalidades mientras se mantiene la compatibilidad con los datos existentes.

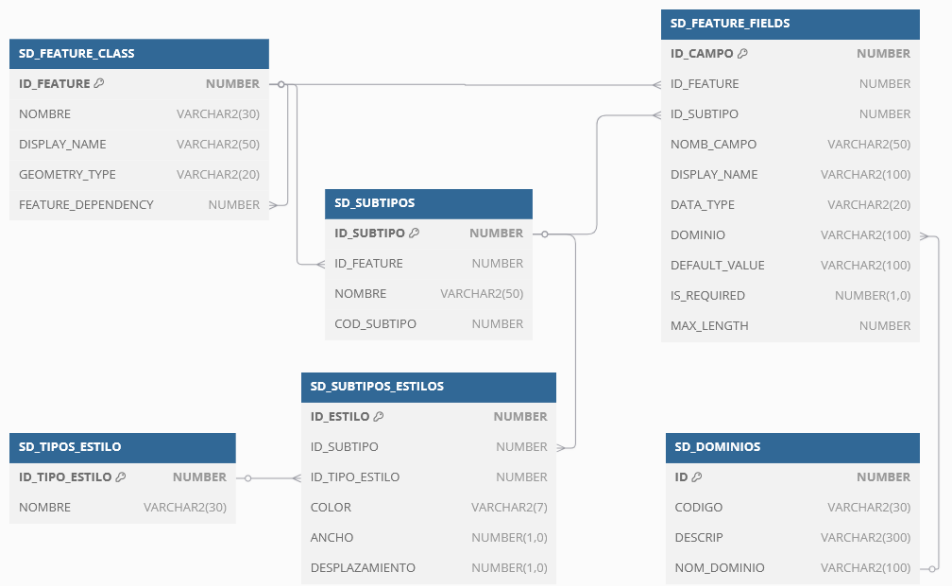


Figura 5. Rediseño del extracto de tablas de la base de datos para la gestión de capas y elementos gráficos del mapa

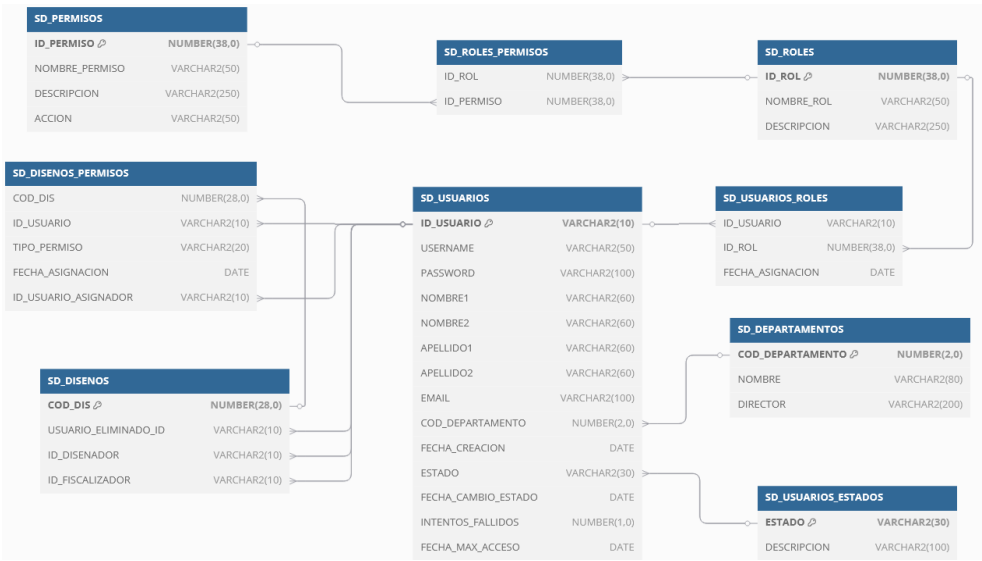


Figura 6. Rediseño del extracto de tablas de la base de datos para la gestión de usuarios, roles y permisos

3.2.2 Diseño y Feedback con el Usuario

Durante la fase de Diseño y Feedback con el Usuario, siguiendo la metodología RAD, se desarrollaron múltiples prototipos funcionales que fueron presentados al cliente para obtener retroalimentación temprana. Esta iteración continua permitió refinar la interfaz y la experiencia de usuario, garantizando que el sistema web final respondiera adecuadamente a las necesidades y expectativas de los usuarios de la EEASA.

En la Figura 7 se muestra la interfaz del sistema SISDE anterior (versión de escritorio). La aplicación presenta una interfaz datada con múltiples barras de herramientas y una visualización del mapa que ocupa la mayor parte de la interfaz. Se puede observar que el sistema tiene un diseño visual propio de aplicaciones de escritorio más antiguas, con interfaces densas y menos intuitivas que las soluciones web modernas.

El sistema SISDE de escritorio, aunque funcional, presenta problemas de usabilidad significativos. Los usuarios encuentran difícil localizar las herramientas necesarias entre las barras y opciones. El cliente recomienda utilizar las interfaces existentes como fuente de inspiración para el diseño de las nuevas interfaces, teniendo en cuenta las mejoras en usabilidad.

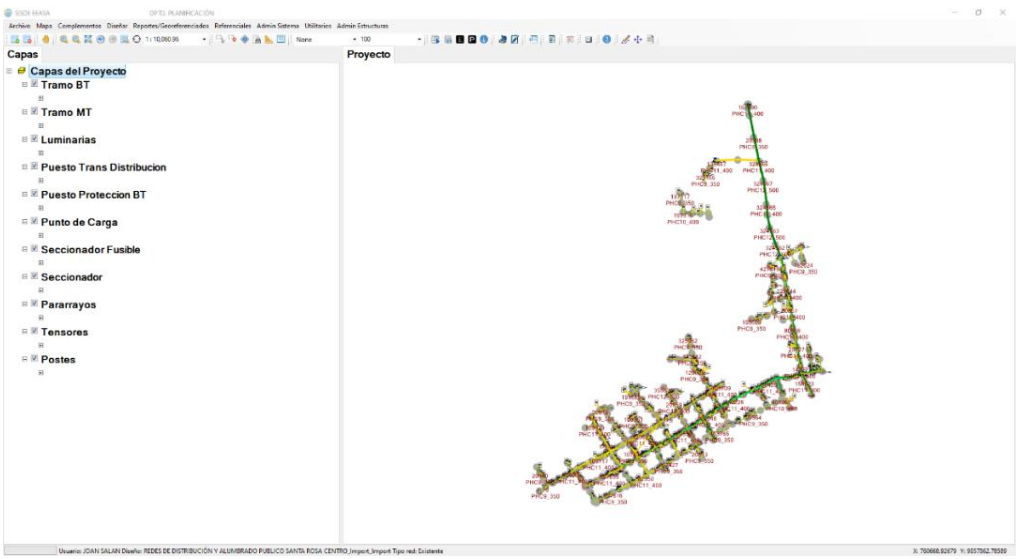


Figura 7. Interfaz pantalla inicial del sistema SISDE de escritorio

A continuación, se presentan los prototipos del sistema SISDE Web, junto con el análisis y la retroalimentación de los mismos:

a. Pantalla de Inicio de Sesión



Figura 8. Prototipo funcional de Inicio de sesión SISDE Web

- **Contenido y funcionalidades:** La Figura 8 muestra la pantalla de inicio de sesión del nuevo sistema web SISDE. Presenta un diseño limpio y minimalista con el logo de SISDE, campos para usuario y contraseña, y opciones para recuperar contraseña. La interfaz utiliza el tema de colores corporativo de la EEASA.

- **Retroalimentación del cliente:** La pantalla de login es simple y directa, debe incluir validación de campos para mejorar la experiencia de usuario. También se sugiere verificar que los mensajes de error sean claros y específicos cuando ocurran problemas de autenticación.

b. Pantalla Inicial

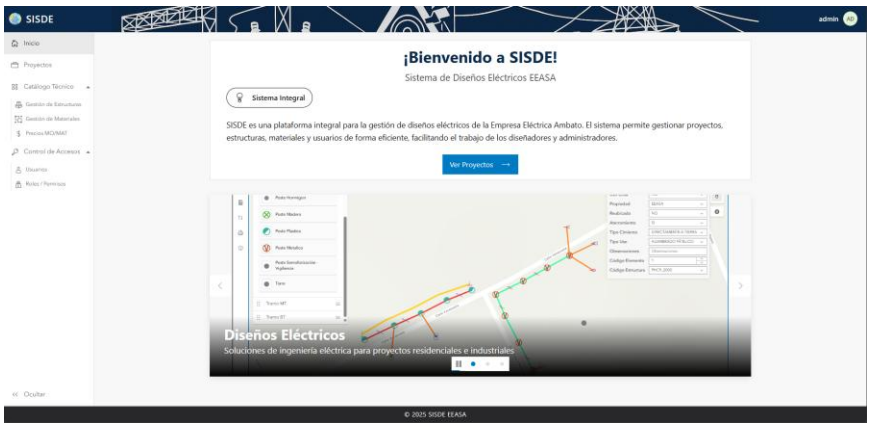


Figura 9. Prototipo funcional de pantalla de Inicio SISDE Web

- **Contenido y funcionalidades:** En la Figura 9 se muestra la pantalla principal al ingresar al sistema. Incluye una barra de navegación lateral con íconos para acceder a diferentes módulos (proyectos, materiales, usuarios, etc.), un encabezado con información del usuario actual, y un panel central con información de bienvenida y un carrusel que muestra características importantes del sistema.

- **Retroalimentación del cliente:** El diseño es intuitivo y la organización de módulos en la barra lateral es clara. Se recomienda incluir botones con redirección hacia las opciones más frecuentes en el sistema, no mostrarlas en el carrusel, como por ejemplo Proyectos, Estructuras, Materiales o Usuarios.

c. Pantalla de Gestión general

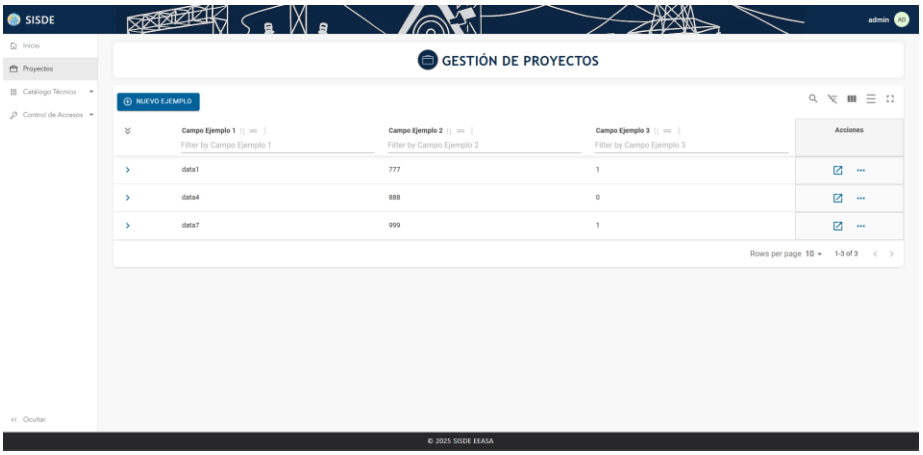


Figura 10. Prototipo funcional de Gestión de Proyectos SISDE Web

- **Contenido y funcionalidades:** El prototipo de la Figura 10 presenta la interfaz de gestión de proyectos, sin embargo, se generaliza para la gestión de los demás elementos como estructuras, materiales y usuarios. Incluye una tabla interactiva con información de los elementos a gestionar (campos específicos), opciones de filtrado y búsqueda, y botones para agregar, editar y eliminar.

- **Retroalimentación del cliente:** La interfaz de gestión de elementos es clara y funcional. Sin embargo, debe permitir ver rápidamente todas las características de un registro, dado el caso de entidades que cuenten con un número

considerable de columnas. Debe permitir agregar botones con acciones personalizadas según sea necesario.

d. Pantalla de Operaciones CRUD

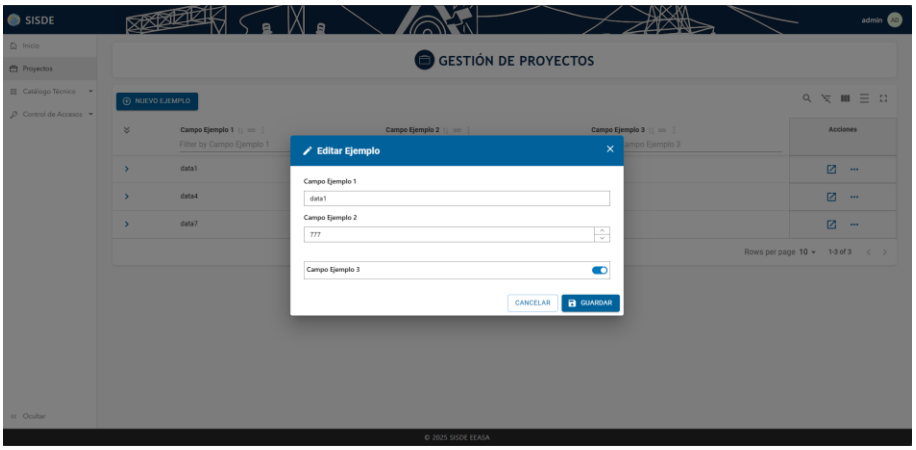


Figura 11. Prototipo funcional de modal de edición SISDE Web

- **Contenido y funcionalidades:** La Figura 11 muestra la interfaz para operaciones CRUD en forma de un modal de edición de registro. Presenta campos de formulario organizados sistemáticamente para modificar las propiedades de un elemento de ejemplo. Incluye campos de texto para ingresar información como códigos y nombres, campos desplegables para seleccionar opciones predefinidas, y controles tipo switch para opciones booleanas. El formulario tiene botones de cancelar y guardar claramente visibles en la parte inferior.

- **Retroalimentación del cliente:** La interfaz de edición es clara y bien estructurada. Se recomienda implementar validación de campos para mejorar la experiencia de usuario y prevenir errores de entrada. Los campos obligatorios deben estar claramente marcados si no han sido llenados. El modal debe ser el mismo tanto para crear como para editar, también que sea responsivo para adaptar formularios más extensos y debe mantener consistencia visual con el resto de la aplicación.

e. Pantalla para subir archivos



Figura 12. Prototipo funcional de pantalla de Actualización de Precios de mano de obra y materiales SISDE Web

- **Contenido y funcionalidades:** En la pantalla de la Figura 12 muestra la funcionalidad para actualizar precios mediante carga de archivos. Incluye una zona para arrastrar y soltar archivos Excel, siendo una instrucción clara sobre el formato esperado. El diseño implementa la funcionalidad de drag and drop solicitada en los requerimientos.

- **Retroalimentación del cliente:** La interfaz para actualización de precios es intuitiva y el área de drag and drop debe detectar el formato esperado, ya sea, para mano de obra o materiales. Se recomienda añadir un enlace para descargar una plantilla de ejemplo que muestre el formato exacto esperado.

f. Pantalla del Mapa

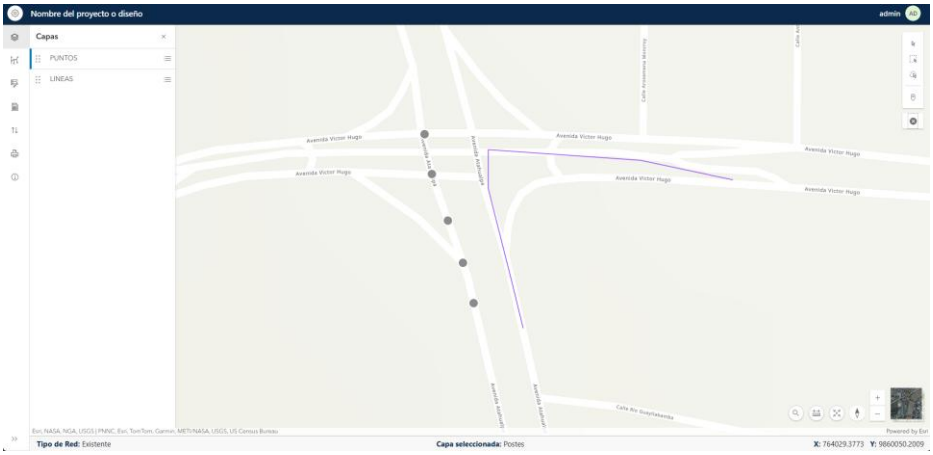


Figura 13. Prototipo funcional del área de trabajo SISDE Web

- **Contenido y funcionalidades:** En la Figura 13 el prototipo muestra el entorno principal de trabajo con mapas, el corazón funcional del sistema. Se han abstraído las funcionalidades del SISDE de escritorio, para disponer las opciones por características similares. Presenta un mapa interactivo con herramientas de dibujo, navegación y medición. La interfaz incluye un panel lateral para gestionar capas de información geográfica, un conjunto de herramientas de edición en la parte superior, y controles de navegación en el mapa.

- **Retroalimentación del cliente:** La interfaz del mapa es limpia e intuitiva, la organización de las herramientas es clara y la mayor área visible para el mapa. Se recomienda mostrar una previsualización de subtipos por capas. Agregar snap (conexión entre elementos) entre todos los elementos de las capas para mayor precisión al dibujar. Tener un control del tamaño de visualización de elementos según el zoom en el mapa. Al seleccionar la herramienta de "Dibujar" se debe mostrar el cuadro de características para la selección de campos, y controlar aquellos que son obligatorios y únicos. Mostrar la información de un elemento seleccionado y añadir un botón que permita editarlo.

El proceso de diseño permitió una evolución significativa desde la versión de escritorio hacia una solución web moderna. Los prototipos funcionales desarrollados reflejan una transformación integral que mantiene las capacidades técnicas del sistema original mientras incorpora principios avanzados de experiencia de usuario y diseño responsivo. La retroalimentación constante del cliente ha sido fundamental para refinar cada interfaz, logrando un equilibrio entre la familiaridad para los usuarios existentes y las mejoras en usabilidad.

3.2.3 Construcción del Backend

El backend del sistema SISDE Web constituye el núcleo de la aplicación, gestionando la lógica de negocio, el acceso a datos y la seguridad del sistema. Se ha implementado utilizando Spring Boot, lo que permite un desarrollo ágil y modular mientras se mantiene la compatibilidad con la infraestructura existente de la EEASA.

A partir de los requerimientos y la revisión del SISDE de escritorio se plantearon los siguientes módulos principales en el backend:

- **Control de acceso**: Gestión de usuarios, roles y permisos.
- **Proyectos**: Gestión del ciclo de vida de diseños eléctricos, estados, historial y workflows de aprobación.
- **Geoespacial**: Gestión de capas, campos y elementos geográficos.
- **Estructuras**: Gestión de estructuras eléctricas y sus materiales asociados.
- **Materiales**: Gestión del inventario de materiales y sus clasificaciones.
- **Reportes**: Generación de reportes técnicos, exportación a diferentes formatos (Excel, PDF, Word).

La implementación de estos módulos sigue la arquitectura en capas descrita anteriormente, aprovechando los procedimientos almacenados existentes en la base de datos Oracle. El sistema utiliza un total de 55 procedimientos y funciones organizados en paquetes especializados: *PKG_SISTEMA* (31 procedimientos) para la lógica de negocio principal incluyendo gestión de diseños, elementos GIS y estructuras; *PKG_UTIL* (8 procedimientos) para operaciones utilitarias; *PKG_ADMIN* (6 procedimientos) para gestión administrativa de usuarios; *PKG_SISTEMA5* (4 procedimientos) para clasificaciones de materiales y estructuras; *PKG_SISTEMA4* (2 procedimientos) para actualización de precios; y *PKG_SISTEMA2* (3 funciones) para validaciones específicas. Esta arquitectura de procedimientos almacenados garantiza una clara separación de responsabilidades y optimiza el rendimiento de las operaciones de base de datos.

a. Inicialización

El backend se ha construido sobre Spring Boot 2.7.18, seleccionado específicamente por su compatibilidad con Java 8 y la capacidad de generar archivos WAR desplegables en el servidor Oracle WebLogic existente en la EEASA. A continuación, se muestran las dependencias principales en la Figura 14:

```xml
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-tomcat</artifactId>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>com.oracle.database.jdbc</groupId>
  <artifactId>ojdbc8</artifactId>
  <version>21.9.0.0</version>
</dependency>
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.springdoc</groupId>
  <artifactId>springdoc-openapi-ui</artifactId>
  <version>1.8.0</version>
</dependency>
```

Figura 14. Fragmento de código con la configuración de dependencias en **`pom.xml`**

La clase principal de la aplicación extiende **`SpringBootServletInitializer`** como se muestra en la Figura 15 para permitir el despliegue en el servidor de aplicaciones WebLogic, que será necesario en la fase de transición más adelante.

```java
@SpringBootApplication
public class WSSisdeApplication extends
SpringBootServletInitializer {

  @Override
  protected SpringApplicationBuilder
configure(SpringApplicationBuilder application) {
    return application.sources(WSSisdeApplication.class);
  }

  public static void main(String[] args) {
    SpringApplication.run(WSSisdeApplication.class, args);
```

```
13      }
14  }
```

Figura 15. Fragmento de código de la clase principal de la aplicación **`WSSisdeApplication.java`**

La configuración de propiedades incluye la conexión a la base de datos Oracle y el contexto de la aplicación. Como requerimiento del cliente, en entorno de producción se debe conectar a través de un datasource en WebLogic. Sin embargo, para el ambiente de desarrollo se configura la conexión con las credenciales de la base de datos, tal y como se muestra en la Figura 16.

```
1  # JNDI DataSource Configuration: Uncomment this for production
2  # spring.datasource.jndi-name=dataSourceName
3  server.servlet.context-path=/WSSisde
4  # JPA Configuration
5  spring.jpa.hibernate.ddl-auto=none
6  spring.jpa.show-sql=true
7  spring.jpa.properties.hibernate.format_sql=true
8  spring.jpa.properties.hibernate.dialect=
9  org.hibernate.dialect.Oracle12cDialect
10 # Development Database Configuration
11 spring.datasource.url=jdbc:oracle:thin:@<dbhost>:<dbport>:<sid>
12 spring.datasource.username=USERNAME
13 spring.datasource.password=PASSWORD
14 spring.datasource.driver-class-name=oracle.jdbc.OracleDriver
```

Figura 16. Fragmento de código del archivo de configuración **`application.properties`**

b. Seguridad

El sistema SISDE Web implementa un robusto sistema de seguridad multicapa que incluye autenticación JWT, control de acceso basado en roles y gestión de recuperación de contraseñas. La seguridad se ha diseñado para garantizar que solo usuarios autorizados puedan acceder a los recursos del sistema, manteniendo la integridad y confidencialidad de la información.

Autenticación JWT

El sistema utiliza JSON Web Tokens (JWT) para la autenticación de usuarios, proporcionando un mecanismo *stateless* que permite escalabilidad y facilita la

integración con otros sistemas. Los tokens se generan durante el proceso de login exitoso y contienen información del usuario como ID, nombre completo y email, con una duración configurable de 24 horas por defecto.

La validación de tokens se realiza a través de un filtro personalizado que intercepta todas las peticiones HTTP, verificando la presencia y validez del token JWT en el header *Authorization*. Esto asegura que solo usuarios autenticados puedan acceder a los endpoints protegidos del sistema.

Control de Acceso y Gestión de Usuarios

El sistema implementa un modelo de seguridad basado en usuarios, roles y permisos que permite una granular administración de accesos. Los usuarios pueden tener asignado un rol, y cada rol puede tener diversos permisos específicos que determinan las operaciones que pueden realizar en el sistema.

Para prevenir ataques de fuerza bruta, el sistema incluye un mecanismo de bloqueo automático después de tres intentos fallidos de login. Cuando un usuario supera este límite, su cuenta se bloquea automáticamente, requiriendo un proceso de recuperación de contraseña para reactivarla, como se muestra en la Figura 17:

```
1  if (!passwordMatches) {
2   int nuevosIntentos = (usuario.getIntentosFallidos() != null ?
3              usuario.getIntentosFallidos() : 0) + 1;
4    if (nuevosIntentos >= 3) {
5      usuarioService.bloquearUsuarioPorIntentos(
6       usuario.getIdUsuario());
7    } else {
8      usuarioService.registrarIntentoFallido(
9       usuario.getIdUsuario(), nuevosIntentos);
10     int intentosRestantes = 3 - nuevosIntentos;
11   }
12 }
```

Figura 17. Fragmento de código con el bloqueo por múltiples intentos fallidos al iniciar sesión

El sistema implementa un robusto esquema de seguridad que utiliza **`BCrypt`** con factor de costo 12 para cifrar contraseñas con *salt* único, nunca almacenándolas en texto plano. Para la recuperación de credenciales, genera tokens criptográficamente seguros

con expiración de 1 hora que se envían por correo electrónico y pueden reactivar cuentas bloqueadas automáticamente.

La configuración de Spring Security opera en modo stateless con JWT, protegiendo todos los recursos excepto endpoints públicos específicos como autenticación, health checks y documentación Swagger, como se muestra en la Figura 18:

```
@Bean
public SecurityFilterChain filterChain(HttpSecurity http)
throws Exception {
  http.csrf().disable()
    .cors().and()
    .sessionManagement()
    .sessionCreationPolicy(SessionCreationPolicy.STATELESS)
    .and()
    .authorizeHttpRequests(authz -> authz
      .requestMatchers(new AntPathRequestMatcher(
                        "/api/auth/**")
    ).permitAll()
      .requestMatchers(new AntPathRequestMatcher(
                       "/api/health/**")
    ).permitAll()
      .anyRequest().authenticated());

  http.addFilterBefore(jwtAuthenticationFilter,
         UsernamePasswordAuthenticationFilter.class);
  return http.build();
}
```

Figura 18. Fragmento de código de la configuración de rutas privadas y públicas

c. ***Modelo.***

Los modelos en el sistema SISDE Web se implementan como clases POJO (Plain Old Java Objects) que representan entidades de negocio. Utilizan Lombok para reducir el código boilerplate, lo que mejora significativamente la legibilidad y mantenibilidad.

Como se mencionó anteriormente, el sistema SISDE de escritorio utiliza procedimientos almacenados en la base de datos para gestionar la lógica en sentencias SQL. Por tanto, para aprovechar funcionalidad existente, los modelos no solo deben representar las entidades del negocio sino también adaptarse a la estructura de resultados que devuelven estos procedimientos.

Por ejemplo, **PROC_SD_FEATURE_CLASS_GET_LAYERS** es un procedimiento almacenado que recupera las capas de elementos como se muestra en la Figura 19:

```
1  PROCEDURE PROC_SD_FEATURE_CLASS_GET_LAYERS(
2    RESPUESTA OUT SYS_REFCURSOR
3  ) IS
4  BEGIN
5    OPEN RESPUESTA FOR
6    SELECT *
7    FROM SD_FEATURE_CLASS WHERE GEOMETRY_TYPE IS NOT NULL
8    ORDER BY ID_FEATURE DESC;
9    EXCEPTION
10 WHEN OTHERS THEN
11   IF RESPUESTA%ISOPEN THEN
12     CLOSE RESPUESTA;
13   END IF;
14   OPEN RESPUESTA FOR
15   SELECT 'ERROR' AS RESULTADO FROM DUAL;
16 END PROC_SD_FEATURE_CLASS_GET_LAYERS;
```

Figura 19. Procedimiento Oracle **PROC_SD_FEATURE_CLASS_GET_LAYERS** en el paquete **PKG_SISTEMA**

La respuesta está basada en los campos de la tabla **SD_FEATURE_CLASS** como se muestra en la Figura 20:

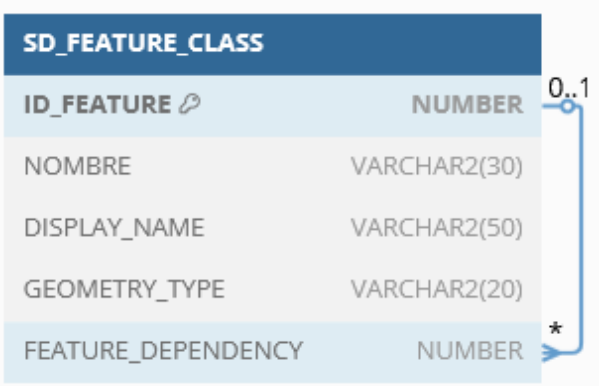


Figura 20. Tabla **SD_FEATURE_CLASS**

Por tanto, se ha definido un modelo correspondiente que captura exactamente los campos que retorna el procedimiento en **FeatureClass.java** como se muestra en la Figura 21:

```java
package com.sisde.model.gis;

import lombok.Data;

@Data
public class FeatureClass {
  private Long idFeature;
  private String nombre;
  private String displayName;
  private String geometryType;
  private Long featureDependency;
}
```

Figura 21. Fragmento de código con el modelo **`FeatureClass`**

En este modelo, la anotación **`@Data`** de Lombok genera automáticamente los métodos **`getter`**, **`setter`**, **`equals`**, **`hashCode`** y **`toString`**, reduciendo significativamente el código boilerplate. Los nombres de los atributos se han definido siguiendo una convención camelCase, mientras mantienen correspondencia con los nombres de columnas en el resultado del procedimiento almacenado (generalmente en mayúsculas con guiones bajos).

```java
@Data
@AllArgsConstructor
@NoArgsConstructor
public class OperacionResponse {
  private String mensaje;
  private Object data;
  private String estado; // "success", "warning", "danger"
  private boolean exito;
}
```

Figura 22. Fragmento de código para respuestas personalizadas con la clase **`OperacionResponse`**

Además, se ha implementado una clase genérica para proporcionar respuestas consistentes a los clientes, estableciendo un estándar unificado para todas las comunicaciones API del sistema SISDE Web como se muestra en la Figura 22. Esta clase encapsula no solo los datos solicitados sino también metadatos cruciales sobre el resultado de la operación, incluyendo mensajes descriptivos, códigos de estado y valores booleanos que indican el éxito o fracaso. Este enfoque permite al frontend

procesar las respuestas de manera uniforme, independientemente del tipo de operación realizada, mejorando así la robustez de la integración entre capas:

De manera similar a **`FeatureClass`** y **`OperationResponse`**, los demás modelos de los módulos del backend siguen el mismo patrón: representan entidades de negocio y se adaptan a la estructura de resultados que devuelven los procedimientos almacenados de la base de datos.

d. Procedimientos Almacenados (PA).

La capa de acceso de datos en SISDE Web tiene como característica principal la integración con procedimientos almacenados Oracle, lo que permite aprovechar la lógica de negocio ya implementada en la base de datos.

Se ha desarrollado una estructura base que facilita la ejecución de procedimientos con o sin parámetros, como adaptadores a través de la clase genérica **`BaseStoredProcedureRepository`** para la ejecución de procedimientos como se muestra en la Figura 23.

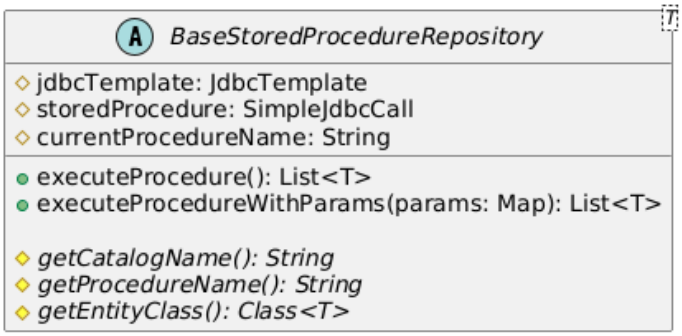


Figura 23. Clase genérica **`BaseStoredProcedureRepository`**

Este repositorio base sirve como adaptador para trabajar con procedimientos almacenados de Oracle. La clase principal utiliza Spring JDBC (específicamente **`SimpleJdbcCall`**) para establecer un método estandarizado de ejecución de estos procedimientos. Esta clase base se inicializa con la anotación **`@PostConstruct`** y ofrece dos métodos principales: **`executeProcedure`** (sin parámetros) y **`executeProcedureWithParams`** (con múltiples parámetros).

El repositorio facilita el mapeo automático de los resultados del procedimiento a objetos Java mediante **`BeanPropertyRowMapper`**. Se presentan dos ejemplos de implementación:

- **`FeatureClassRepository`**, un caso simple que extiende la clase base para conectar con el procedimiento **`PROC_SD_FEATURE_CLASS_GET_LAYERS`** sin necesidad de parámetros como se muestra en la Figura 24:

```
1  @Repository
2  public class FeatureClassRepository extends
3  BaseStoredProcedureRepository<FeatureClass> {
4    public FeatureClassRepository(JdbcTemplate jdbcTemplate) {
5    super(jdbcTemplate); }
6
7    @Override protected String getCatalogName() {
8    return "PKG_SISTEMA"; }
9    @Override protected String getProcedureName() {
10   return "PROC_SD_FEATURE_CLASS_GET_LAYERS"; }
11   @Override protected Class<FeatureClass> getEntityClass() {
12   return FeatureClass.class; }
13
14   public List<FeatureClass> getLayers() {
15   return executeProcedure(); }
16 }
```

Figura 24. Clase **`FeatureClassRepository`** en la ejecución del procedimiento **`PROC_SD_FEATURE_CLASS_GET_LAYERS`** sin parámetros

- **`DisenioRepository`**, un caso más complejo que demuestra cómo pasar parámetros como se muestra en la Figura 25:

```
1  @Repository
2  public class DisenioRepository extends
3  BaseStoredProcedureRepository<Disenio> {
4    public DisenioRepository(JdbcTemplate jdbcTemplate) {
5    super(jdbcTemplate); }
6
7    @Override protected String getCatalogName() {
8    return "PKG_SISTEMA"; }
9    @Override protected String getProcedureName() {
10   return "PROC_SD_DISENIOS_BUSQUEDA"; }
11   @Override protected Class<Disenio> getEntityClass() {
12   return Disenio.class; }
13
```

```
14    public List<Disenio> getDisenioByCodigo(Integer codDis) {
15      var params = new HashMap<String, Object>() {{
16       put("IN_COD_DIS", codDis); }};
17      return executeProcedureWithParams(params);
18    }
19  }
```

Figura 25. Clase **DisenioRepository** en la ejecución del procedimiento **PROC_SD_DISENIOS_BUSQUEDA** con parámetros

El diseño demuestra gran flexibilidad, permitiendo desde simples llamadas sin parámetros hasta escenarios complejos con múltiples procedimientos, parámetros personalizados y mapeos específicos. Cuando el mapeo automático no es suficiente, se pueden implementar **RowMapper** personalizados como se muestra en la Figura 26 con **DisenioEstadoRowMapper**, que transforma los resultados de la consulta en objetos de tipo **DisenioEstado** con lógica específica de mapeo:

```
1   class DisenioEstadoRowMapper implements
2   RowMapper<DisenioEstado> {
3     public DisenioEstado mapRow(ResultSet rs, int rowNum)
4     throws SQLException {
5       var estado = new DisenioEstado();
6       estado.setCodEstado(rs.getInt("COD_ESTADO"));
7       estado.setEstado(rs.getString("ESTADO"));
8       estado.setDescripcion(rs.getString("DESCRIPCION"));
9       return estado;
10    }
11  }
```

Figura 26. **RowMapper** personalizado en la captura de las propiedades de **DisenioEstado**

De manera similar a **FeatureClassRepository** y **DisenioRepository**, los demás repositorios en de los módulos del backend siguen el mismo patrón arquitectónico: todos extienden de **BaseStoredProcedureRepository** para heredar la funcionalidad común de acceso a procedimientos almacenados, implementan los métodos abstractos requeridos y proporcionan métodos específicos para ejecutar operaciones de negocio adaptadas a las necesidades particulares de cada entidad. Para casos que requieren mapeos especializados entre los resultados de la base de datos y los objetos de dominio, el sistema utiliza RowMappers personalizados.

e. ***Servicio***

La capa de servicio implementa la lógica de negocio y actúa como intermediaria entre los controladores y los adaptadores de PA. Siguiendo el ejemplo, se obtienen los métodos de **`DisenioRepository`** para implementar su servicio como se muestra en la Figura 27:

```
@Autowired
private DisenioRepository disenioRepository;

public List<Disenio> getDisenios() {
  return disenioRepository.getDisenios();
}

public OperacionResponse crearDisenio(Disenio disenio,
 String idUsuario) {
  String result = (String) disenioRepository.
    crearDisenio(disenio, idUsuario);
  return new OperacionResponse((String)
    result.get("MENSAJE"), disenio);
}

public OperacionResponse eliminarDisenio(Long codDis,
 String idUsuario) {
  return new OperacionResponse((String) disenioRepository.
   eliminarDisenio(codDis, idUsuario).get("MENSAJE"));
}
```

Figura 27. Fragmento de código de **`DisenioService.java`**

El repositorio implementa métodos como **`getDisenios()`** para recuperar diseños, **`crearDisenio()`** para crear nuevos, **`actualizarDisenio()`** para modificar existentes y **`eliminarDisenio()`** para eliminar registros. Cada servicio sigue un patrón común: delega las operaciones al adaptador PA, maneja el resultado y lo encapsula en un objeto **`OperacionResponse`** estandarizado, proporcionando así una estructura de respuesta uniforme para los controladores.

Un aspecto destacable de esta implementación es el manejo centralizado de excepciones; todos los métodos que modifican datos están envueltos en bloques try-catch que capturan cualquier excepción, la transforman en un mensaje legible mediante **`ErrorUtil.obtenerMensajeError()`** y la devuelven como parte de un

OperacionResponse con estado de error. Este enfoque asegura que todas las operaciones, exitosas o fallidas, retornan una respuesta consistente a la capa de controlador, facilitando así el manejo de errores y la presentación de mensajes informativos al usuario final.

De manera similar a **DisenioService**, los demás servicios de los módulos del backend siguen el mismo patrón arquitectónico: todos actúan como intermediarios entre controladores y los adaptadores de PA, implementan la lógica de negocio específica para cada entidad, gestionan las operaciones CRUD delegando las llamadas a sus respectivos adaptadores PA y encapsulan uniformemente los resultados en objetos **OperacionResponse**.

Todos los servicios incorporan un manejo centralizado de excepciones mediante bloques try-catch que capturan errores, los transforman en mensajes comprensibles utilizando utilidades como **ErrorUtil**, y los devuelven en respuestas estandarizadas, garantizando así que la capa de presentación reciba siempre información estructurada de manera consistente, independientemente del resultado de la operación.

f. ***Controlador***

Los controladores en el sistema SISDE Web se han implementado siguiendo los principios REST, exponiendo endpoints claros y consistentes para cada funcionalidad del sistema. Los controladores se especializan en los módulos del backend.

El controlador **PermisoController**, como se muestra en la Figura 28, implementa una API RESTful para la gestión de permisos en el sistema SISDE Web. Anotado con **@RestController** y mapeado a la ruta "/api/permisos", expone cinco endpoints principales que siguen las convenciones HTTP estándar: **GET** para obtener todos los permisos o uno específico por ID, **POST** para crear nuevos permisos, **PUT** para actualizar permisos existentes y **DELETE** para eliminarlos. Cada método delega la lógica de negocio al servicio correspondiente (**permisoService**) y utiliza **ResponseEntity** para encapsular las respuestas HTTP con los códigos de estado apropiados, apoyándose en una clase utilitaria (**ResponseEntityUtil**) para determinar el código de estado basado en el resultado de la operación.

```java
@RestController
@RequestMapping("/api/permisos")
public class PermisoController {
  @Autowired private PermisoService permisoService;

  @GetMapping
  public ResponseEntity<List<Permiso>> getAllPermisos() {
    return ResponseEntity.ok(
    permisoService.getAllPermisos());
  }

  @PostMapping
  public ResponseEntity<OperacionResponse>
  crearPermiso(@RequestBody Permiso permiso) {
    return ResponseEntityUtil.determinarResponseEntity(
     permisoService.crearPermiso(permiso));
  }
}
```

Figura 28. Fragmento de código del archivo **`PermisoController.java`**

De manera similar a **`PermisoController`**, los demás controladores de los módulos admin, comun, estructuras, gis y materiales siguen el mismo patrón arquitectónico RESTful. Cada controlador se especializa en un dominio específico de la aplicación, está anotado con **`@RestController`**, define una ruta base mediante **`@RequestMapping`** y expone endpoints para operaciones CRUD que mapean directamente a los métodos HTTP correspondientes mediante anotaciones como **`@GetMapping`**, **`@PostMapping`**, **`@PutMapping`** y **`@DeleteMapping`**. Todos los controladores delegan la lógica de negocio a sus respectivos servicios y utilizan **`ResponseEntity`** para devolver respuestas HTTP bien formadas, manteniendo así una estructura consistente a través de toda la capa de presentación y facilitando la integración con clientes frontend.

g. Utililatio

El paquete de utilidades contiene clases que proporcionan funcionalidades comunes utilizadas en todo el sistema, desde el manejo de errores hasta el procesamiento de respuestas HTTP.

```java
public class ResponseEntityUtil {
  public static ResponseEntity<OperacionResponse>
```

```
    determinarResponseEntity(OperacionResponse response) {
    if (response == null) {
      return ResponseEntity.status(
        HttpStatus.INTERNAL_SERVER_ERROR
      ).body(new OperacionResponse(
          "Error: respuesta nula").
          fallido("Error: respuesta nula")
      );
    }
    switch (response.getEstado()) {
      case "danger":
        return ResponseEntity.status(
         HttpStatus.INTERNAL_SERVER_ERROR).
          body(response);
      default:
        return ResponseEntity.status(HttpStatus.OK).
          body(response);
    }
  }
}
```

Figura 29. Fragmento de código de **`ResponseEntityUtil.java`**

En la Figura 29, **`ResponseEntityUtil.java`** actúa como una clase utilitaria que facilita la creación de respuestas HTTP estandarizadas a partir de objetos **`OperacionResponse`**. Su método principal, **`determinarResponseEntity()`**, evalúa el estado de la respuesta ("danger", "warning", "success") y determina automáticamente el código HTTP apropiado a devolver, asegurando consistencia en las respuestas de la API. Además, proporciona métodos auxiliares como **`created()`**, **`error()`**, **`ok()`**, **`badRequest()`** y **`notFound()`** que simplifican la creación de **`ResponseEntity`** con códigos de estado específicos, mejorando así la claridad y mantenibilidad del código en los controladores.

`ErrorUtil.java` es una clase utilitaria especializada en traducir excepciones técnicas de base de datos en mensajes de error comprensibles para el usuario final. Su método principal, **`obtenerMensajeError()`**, navega a través de la cadena de excepciones para encontrar la causa raíz, particularmente **`SQLException`**, y analiza los códigos de error de Oracle (como **`ORA-12899`**, **`ORA-00001`**, **`ORA-02292`**, **`ORA-01400`**) utilizando expresiones regulares para extraer información relevante y generar mensajes descriptivos. Esta clase transforma errores técnicos crípticos en mensajes

claros y contextualizados como "El campo 'X' no puede estar vacío" o "Ya existe un registro con este identificador", mejorando significativamente la experiencia del usuario.

h. Pruebas

OpenAPI con Swagger proporciona una interfaz intuitiva para desarrolladores y usuarios técnicos. La misma facilita la exploración y validación de los endpoints en tiempo real, asegurando su correcto funcionamiento antes de la integración con el frontend. Esta herramienta se ha utilizado para la documentación y prueba interactiva de la API del sistema SISDE Web.

Siguiendo el ejemplo de **`PermisoController`**, en la Figura 30 La interfaz de Swagger muestra los endpoints disponibles para la gestión de permisos, siguiendo principios REST con métodos HTTP estándar. Se observan las operaciones **`GET`** para obtener permisos, **`POST`** para crear, **`PUT`** para actualizar y **`DELETE`** para eliminar, cada una asociada a sus respectivas rutas. El panel inferior permite configurar parámetros y cuerpos de solicitud necesarios para ejecutar las operaciones.

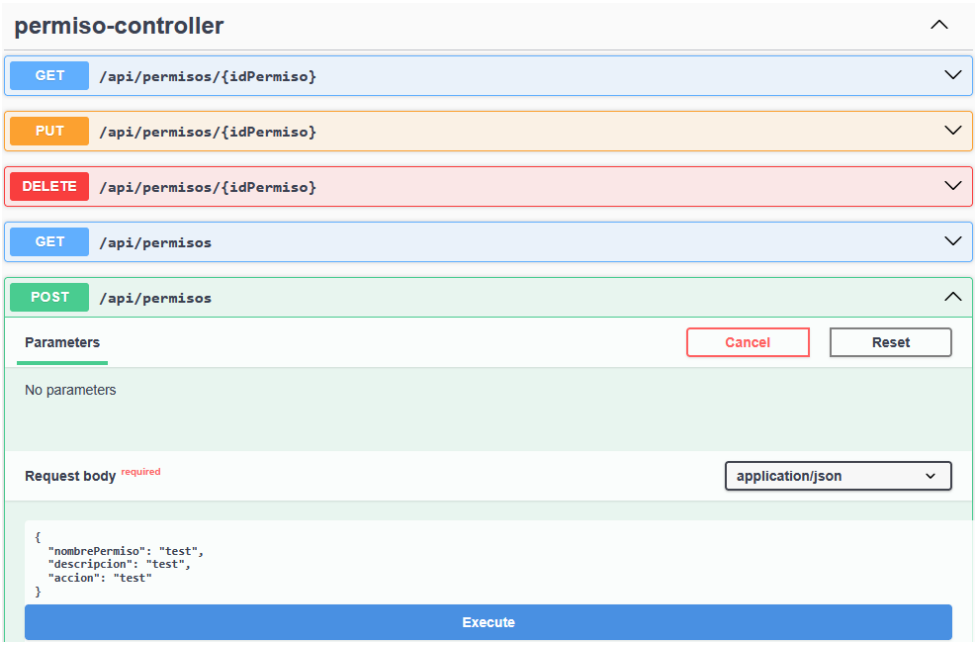


Figura 30. Endpoints de **`permiso-controller`** en Swagger

La Figura 31 muestra la respuesta generada por el servidor tras la creación exitosa de un permiso. Se visualiza claramente el código de estado HTTP 200 indicando éxito, junto con el cuerpo de respuesta JSON que implementa la estructura estandarizada **`OperacionResponse`**. La respuesta incluye el mensaje de confirmación "Permiso creado correctamente", los datos del permiso creado, y los campos de estado "success" y éxito "true" que facilitan el manejo del resultado.

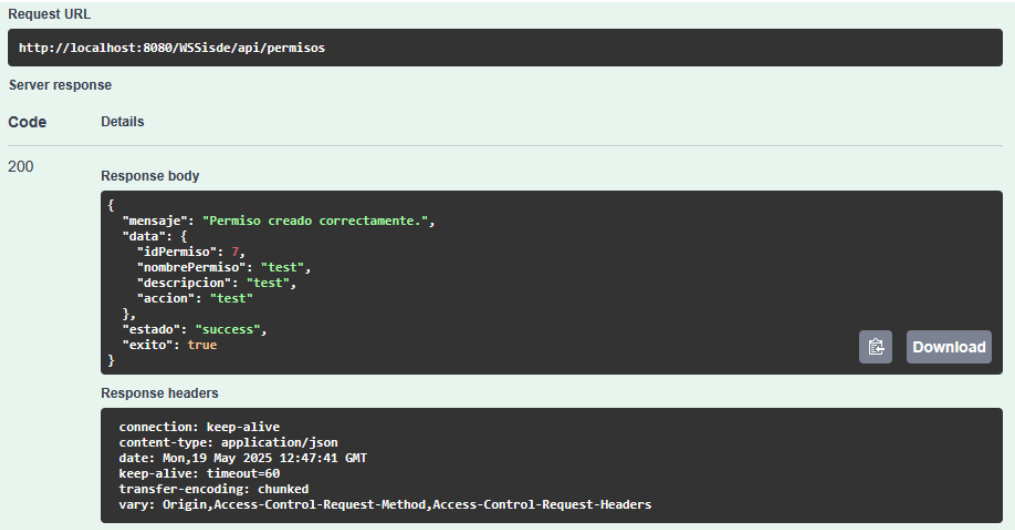


Figura 31. Respuesta del servidor backend al crear un permiso

Los demás endpoints del sistema SISDE Web siguen la misma estructura consistente con los demás módulos del backend. Esta uniformidad asegura que los desarrolladores puedan predecir fácilmente el comportamiento de cualquier endpoint, reduciendo significativamente la curva de aprendizaje y acelerando la integración. Cada endpoint está debidamente documentado con descripciones claras de su propósito, parámetros esperados y posibles respuestas, facilitando así el desarrollo colaborativo y el mantenimiento a largo plazo del sistema.

3.2.4 Construcción del Frontend

El frontend del sistema SISDE Web se desarrolló utilizando React junto con librerías complementarias para crear una interfaz de usuario intuitiva y responsiva. La arquitectura se estructuró de manera modular, separando claramente los componentes, servicios y páginas, para facilitar su mantenimiento y escalabilidad.

a. Cuestiones fundamentales de implementación

- ***Inicialización.*** El sistema SISDE utiliza una serie de dependencias modernas para construir una aplicación robusta y funcional como se muestra en la Figura 32:

```
{
 "dependencies": {
  "@arcgis/core": "~4.31.0",
  "@arcgis/map-components": "~4.31.0",
  "@arcgis/map-components-react": "~4.31.0",
  "@esri/calcite-components": "^2.13.2",
  "@esri/calcite-components-react": "^2.13.0",
  "@mui/material": "^7.0.2",
  "axios": "^1.8.4",
  "jspdf": "^3.0.1",
  "material-react-table": "^3.2.1",
  "proj4": "^2.15.0",
  "react": "^18.0.0",
  "react-dom": "^18.0.0",
  "react-dropzone": "^14.2.3",
  "react-router-dom": "^7.5.1"
 }
}
```

Figura 32. Fragmento de código con las dependencias del frontend en el **`package.json`**

Esta configuración de dependencias proporciona un sólido conjunto de herramientas para el desarrollo. React sirve como el framework base para la interfaz de usuario, complementado por React Router DOM para la navegación entre páginas. Para la visualización geoespacial, se implementaron los componentes de ArcGIS, permitiendo la integración de mapas interactivos y herramientas de análisis espacial. La interfaz gráfica se enriquece con componentes de Material UI y Calcite de Esri, que proporcionan controles visuales modernos y consistentes. Axios facilita las comunicaciones HTTP con el backend, mientras que librerías como jsPDF permiten la generación de documentos. Para la manipulación de datos georreferenciados, se utiliza proj4 para transformaciones de coordenadas, fundamental en una aplicación orientada a la gestión de redes eléctricas distribuidas geográficamente.

Configuración inicial

La configuración de la aplicación SISDE se realiza mediante el archivo **`vite.config.js`**, que define los parámetros esenciales para el entorno de desarrollo y construcción como se muestra en la Figura 33:

```
1  import { defineConfig } from "vite";
2  import react from "@vitejs/plugin-react";
3
4  export default defineConfig({
5   plugins: [react()],
6   server: {
7    open: true,
8   },
9   base: "/sisde/",
10  build: {
11   outDir: "dist",
12  },
13 });
```

Figura 33. Archivo de configuración **`vite.config.js`**

La configuración inicial del sistema SISDE se implementa a través de este archivo, donde se establece Vite como herramienta de construcción junto con el plugin de React para características como JSX y Hot Module Replacement. Se define el parámetro **`base: "/sisde/"`**, indicando que la aplicación será accesible desde la ruta **`http://[dominio]/sisde/`** en lugar de la raíz del servidor, para el despliegue en entornos de producción más adelante. Adicionalmente, se configura que los archivos compilados se generarán en el directorio **`dist`**, y que el servidor de desarrollo se abrirá automáticamente al iniciar el proyecto, mejorando así la experiencia del desarrollador.

Configuración de ArcGIS y Calcite Components

```
1  // Importación y configuración de Calcite Components
2  import { setAssetPath }
3  from '@esri/calcite-components/dist/components';
4  setAssetPath(window.location.href);
5
6  import { defineCustomElements }
7  from "@esri/calcite-components/dist/loader";
8  defineCustomElements();
9
10 // Importación y configuración de los Map Components de ArcGIS
11 import { defineCustomElements as defineMapElements }
12 from "@arcgis/mapcomponents/dist/loader";
```

```
13 defineMapElements(window, {
14  resourcesUrl: "https://js.arcgis.com/map-
15  components/4.31/assets",
16 });
```

Figura 34. Fragmento de código de **`index.jsx`**

Se establece la configuración fundamental para dos bibliotecas clave del sistema SISDE como se muestra en la Figura 34. En primer lugar, configura Calcite Components, un sistema de diseño desarrollado por Esri que proporciona componentes web reutilizables con una estética moderna y coherente. El código importa la función **`setAssetPath`** para definir la ruta base de los recursos de Calcite utilizando la URL actual, y luego emplea **`defineCustomElements`** para registrar todos los componentes Calcite como elementos personalizados en el DOM, haciéndolos disponibles en toda la aplicación.

En segundo lugar, el código configura los Map Components de ArcGIS, una biblioteca especializada para integrar mapas interactivos en aplicaciones web. Mediante la función **`defineMapElements`**, se registran todos los componentes cartográficos de ArcGIS como elementos web personalizados, especificando explícitamente la URL donde se encuentran los recursos necesarios (iconos, estilos, etc.) en el CDN de Esri. Esta configuración es esencial para el funcionamiento adecuado de los mapas y herramientas geoespaciales que forman parte integral del sistema SISDE para la gestión de redes eléctricas.

- ***Estructura.*** La estructura del proyecto SISDE Web en el cliente se organizó siguiendo principios de arquitectura modular y separación de responsabilidades, donde cada directorio cumple un propósito específico. La organización que se muestra en la Figura 35 se eligió por sus múltiples ventajas: mejora la mantenibilidad al separar claramente las responsabilidades de cada segmento de código, facilita la escalabilidad al permitir la incorporación sencilla de nuevas funcionalidades sin afectar las existentes, potencia la reutilización de componentes independientes en diferentes partes de la aplicación.

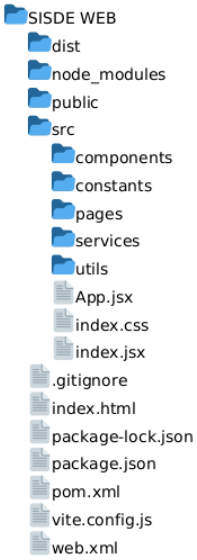


Figura 35. Estructura de ficheros frontend

- ***Constantes.*** El archivo ***`pageIcons.js`***, que se muestra en la Figura 36,cumple un papel fundamental en la arquitectura del sistema SISDE Web al centralizar todos los iconos utilizados a través de la aplicación mediante un objeto ***`PAGE_ICONS`*** que mapea identificadores semánticos a nombres de iconos específicos. Esta estrategia de centralización garantiza la consistencia visual en toda la interfaz, simplifica considerablemente el mantenimiento al permitir cambiar cualquier icono de forma global con una sola modificación, potencia la productividad del desarrollador aprovechando el autocompletado en IDEs, y reduce significativamente la probabilidad de errores tipográficos en los nombres de iconos, lo que resultaría en incongruencias visuales o iconos faltantes en la interfaz de usuario.

```
1  export const PAGE_ICONS = {
2    HOME: 'home',
3    PROJECTS: 'projects',
4    CATALOG: 'miscellaneous-collection',
5    ACCESS: 'key',
6    USERS: 'user',
7    STRUCTURES: '3d-building',
8    MATERIALS: 'group-items',
9    PRICE_UPDATE: 'money',
10   ROLES: 'lock',
11   REPORTS: 'reports',
12   STRUCTURE_TAB: 'package',
13   CLASSIFICATION_TAB: 'ungroup-layout-elements',
```

```
  MATERIAL_TAB: 'auto-layout',
  ROLES_TAB: 'user-key',
  PERMISSIONS_TAB: 'user-to-device',
};
```

Figura 36. Archivo **pageIcons.js**

- ***Servicios.*** Los servicios en la aplicación SISDE Web actúan como una capa de abstracción que encapsula toda la lógica de comunicación con el backend, proporcionando métodos estructurados para realizar operaciones CRUD, transformar datos y manejar errores. Esta capa aísla los componentes de la interfaz de usuario de la complejidad de las llamadas HTTP, permitiendo mantener un código más limpio, testeable y fácil de mantener cuando los endpoints de la API cambian.

El archivo **api.js** configura una instancia central de Axios que establece la URL base **http://localhost:8080/WSSisde/api/** para todas las solicitudes HTTP, define los encabezados predeterminados, e implementa interceptores para el procesamiento uniforme de solicitudes y respuestas como se muestra en la Figura 37:

```
import axios from 'axios';

const api = axios.create({
  baseURL: 'http://localhost:8080/WSSisde/api/',
  headers: {
    'Content-Type': 'application/json',
  },
});
```

Figura 37. Fragmento de código **api.js**

El servicio **layerService** de la Figura 38 ejemplifica el patrón utilizado en todos los servicios de la aplicación, importando la instancia base de API y definiendo métodos específicos para interactuar con los endpoints relacionados con las capas del mapa. Cada método encapsula una operación específica (**getLayers**, **getSubtipos**, **getCampos**), maneja la construcción de URL y parámetros, procesa las respuestas exitosas extrayendo directamente los datos (**response.data**), y propaga los errores de forma controlada.

```
import api from './api';

```

```
export const layerService = {
  getLayers: async () => {
    try {
      const response = await api.get('/layers');
      return response.data;
    } catch (error) {
      throw error;
    }
  },
  // Demás funciones…
};
```

Figura 38. Fragmento de código de **`layerService.js`**

Este mismo patrón, una interfaz consistente y predecible, se replica en todos los demás servicios mencionados en la construcción del backend.

- ***Componentes.*** El sistema SISDE utiliza una arquitectura basada en componentes reutilizables que facilitan el desarrollo y mantenimiento de la aplicación. Estos componentes están diseñados para proporcionar una experiencia de usuario consistente a través de diferentes módulos del sistema, reduciendo la duplicación de código y mejorando la escalabilidad. Los componentes están construidos sobre una base de React y Calcite Components de ESRI, complementados con Material UI para funcionalidades avanzadas de interfaz.

CustomAlert

`CustomAlert` es un componente de notificación que muestra mensajes al usuario con diferentes estilos según el tipo de mensaje: en la Figura 39 cuando el proceso fue correcto, en la Figura 40 cuando el proceso presenta un error leve y en la Figura 41 cuando se ha producido un error. Utiliza CalciteAlert de Esri para proporcionar una experiencia visual consistente con el diseño del sistema. Entre las principales características de este componente se encuentran:

- **Mensajes contextuales:** Diferentes estilos según el tipo de mensaje.
- **Posicionamiento flexible:** Puede aparecer en diferentes áreas de la pantalla.
- **Cierre automático:** Se cierra solo después de un tiempo configurable.

- **Personalizable:** Permite personalizar título, mensaje e icono.

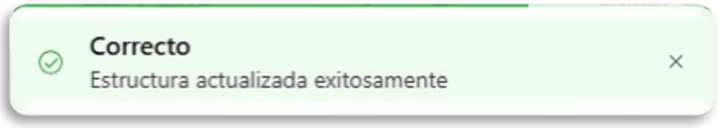


Figura 39. Visualización de una alerta de estado correcto

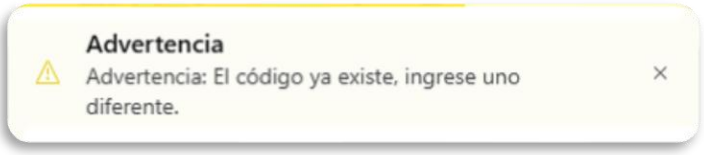


Figura 40. Visualización de una alerta de advertencia

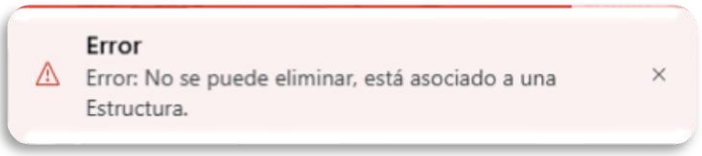


Figura 41. Visualización de una alerta de error

La función **`getBackgroundColor`** implementa un sistema de feedback visual diferenciado que asocia colores específicos a distintos tipos de alertas, utilizando tonalidades verdes suaves para mensajes de éxito, amarillos pálidos para advertencias, y rosas tenues para errores, mejorando así la interpretación inmediata del mensaje por parte del usuario. Complementariamente, la función **`getTitle`** refuerza esta comunicación visual mediante la asignación automática de textos descriptivos según el tipo de alerta, determinando si debe mostrar "Correcto", "Advertencia" o "Error" cuando no se proporciona un título personalizado.

Ambas funciones trabajan en conjunto para crear una experiencia de usuario coherente y accesible, donde tanto el color como el texto transmiten de manera uniforme la naturaleza del mensaje, siguiendo principios de diseño de interfaces que facilitan la interpretación rápida de información crítica. Para su utilización se disponen las siguientes propiedades descritas en la Tabla 15:

Tabla 15. Propiedades del componente **`CustomAlert`**

Propiedad	Descripción	Valor por defecto	Posibles valores
`title`	Título de la alerta	`null`	`String`
`message`	Mensaje principal	`''`	`String`
`type`	Tipo de alerta (afecta color y estilo)	`'success'`	`'success'`, `'warning'`, `'danger'`
`position`	Posición en la pantalla	`'top-end'`	`'top'`, `'top-end'`, `'top-start'`, `'bottom'`, `'bottom-end'`, `'bottom-start'`
`autoCloseDuration`	Velocidad de cierre automático	`'fast'`	`'fast'`, `'medium'`, `'slow'`
`onClose`	Función llamada al cerrar	`null`	`Function`

CustomTable

Es un componente complejo y altamente configurable para mostrar, editar y gestionar datos tabulares. Está construido sobre **`MaterialReactTable`** para proporcionar funcionalidades avanzadas como ordenación, filtrado y paginación, pero añade capacidades de CRUD completas. Entre las principales características de este componente se encuentran:

- **Operaciones CRUD:** Proporciona interfaces para crear, leer, actualizar y eliminar registros.



Figura 42. Modal de **`CustomTable`** para la creación de un nuevo registro

CustomTable dispone un botón para agregar elementos, al presionarlo aparecerá un modal con los campos previamente definidos y que sean visible al crear, como se muestra en la Figura 42. Se implementa la lógica de operación a través de una función, la cual ejecuta las funciones de los servicios según sea necesario como en la Figura 43, en este caso el servicio **structureService** a través de su método **createStructure**. Es importante devolver el resultado de la ejecución del servicio "**return newStructure;**", ya que, el **CustomAlert** se mostrará de acuerdo con el resultado de dicha operación.

```
1  const handleAddStructure = async (structureData) => {
2    try {
3     const newStructure = await
4      structureService.createStructure(structureData);
5     return newStructure;
6    } catch (error) {
7     console.error("Error al crear estructura:", error);
8     return error;
9    }
10  };
```

Figura 43. Fragmento de código para añadir una estructura

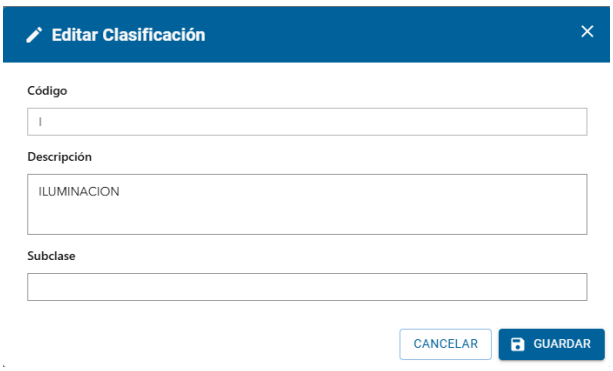


Figura 44. Modal de **CustomTable** para la edición de un registro

Para editar un registro se proporcionan dos formas de acceder al modal de edición como se ilustra en la Figura 44: al hacer doble click en el registro o al desplegar las opciones del registro y presionar en **Editar**. De forma similar al proceso de crear, se dispone una función de editar, como se muestra en la Figura 45, la cual realizará la operación a través de los servicios correspondientes:

```
const handleUpdateStructure = async (structureData) => {
  try {
   const updatedStructure = await
    structureService.updateStructure(
    structureData.codigoEstructura,
    structureData
    );
   return updatedStructure;
  } catch (error) {
   console.error("Error al actualizar estructura:", error);
   return error;
  }
 };
```

Figura 45. Fragmento de código para editar una estructura

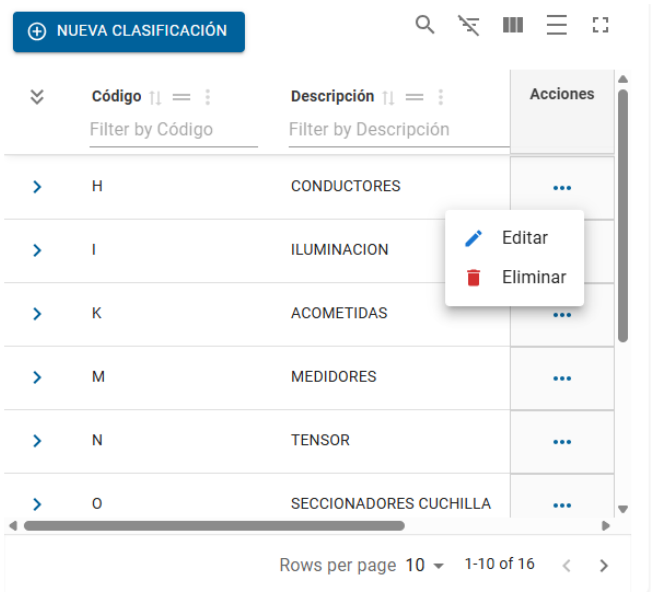


Figura 46. Opción de eliminar en las opciones de las acciones

Finalmente, para abarcar todas las operaciones CRUD, se tiene el proceso de **`Eliminar`**, el cual se encuentra en las opciones de las **`Acciones`** del registro de interés, como se muestra en la Figura 46. La siguiente función ejecuta la operación de eliminar dado el **`structureId`** del registro como se muestra en la Figura 47:

```
Const handleDeleteStructure = async (structureId) => {
  try {
   const deletedStructure = await
    structureService.deleteStructure(structureId);
   return deletedStructure;
  } catch (error) {
```

```
7     console.error("Error al eliminar estructura:", error);
8     return error;
9    }
10  };
```

Figura 47. Fragmento de código para eliminar una estructura

- **Formularios dinámicos:** Genera formularios basados en la configuración de campos.

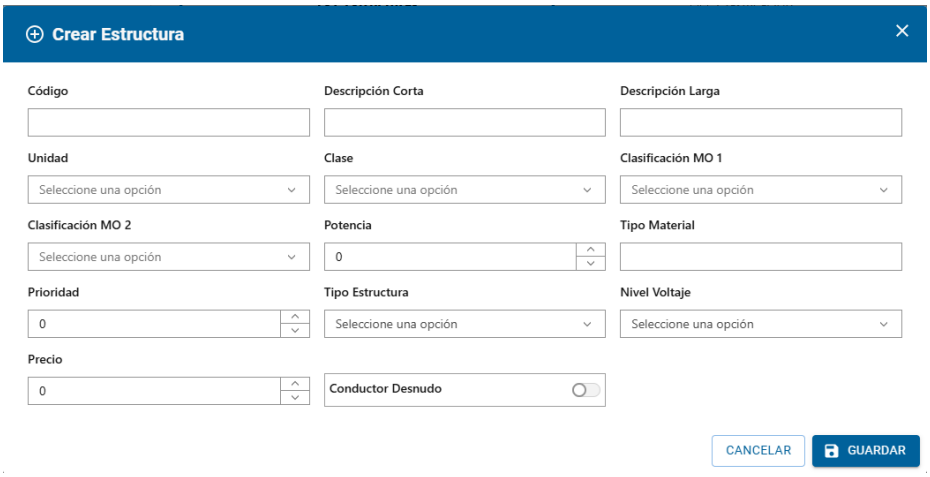


Figura 48. Modal para crear una estructura

El sistema emplea un enfoque declarativo para la generación de formularios dinámicos mediante una definición estructurada de campos, como se aprecia en la Figura 48. Esta implementación utiliza **`useMemo`** para optimizar el rendimiento, evitando recálculos innecesarios de la configuración de campos al renderizar el componente.

Cada campo se define con propiedades específicas que controlan su comportamiento: **`fieldName`** establece el identificador único, **`header`** define la etiqueta visible, **`size`** controla el ancho visual, **`required`** determina la obligatoriedad, **`type`** especifica el tipo de entrada (text, select, etc.), **`disabledInEdit`** permite deshabilitar campos durante la edición, y **`maxLength`** limita la longitud del texto. Esta arquitectura facilita la adaptación de formularios a diferentes contextos y requisitos de datos, proporcionando una experiencia consistente y personalizada como se observa en el formulario de la Figura 49.

```
const structureFieldDefinitions = useMemo(() => [
  {
   fieldName: 'codigoEstructura',
   header: 'Código',
   size: 100,
   required: true,
   type: 'text',
   disabledInEdit: true,
   maxLength: 15,
  },
  // Otros campos de estructura
], []);
```

Figura 49. Fragmento de código **`structureFieldDefinitions`**

- **Validación de datos:** Verifica requerimientos y formatos antes de enviar.

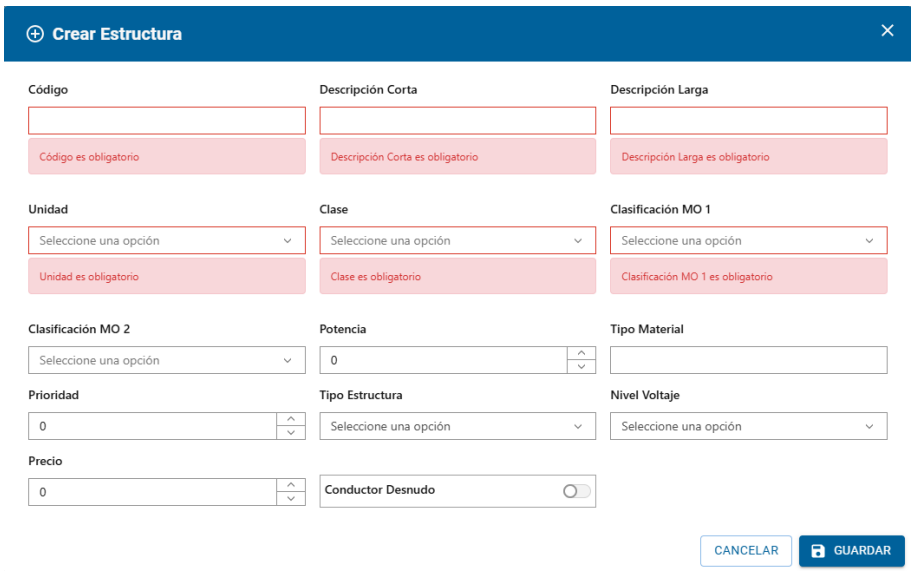


Figura 50. Verificación de campos obligatorios en la interfaz de usuario

El sistema implementa un mecanismo de validación de datos que verifica el cumplimiento de requisitos para cada campo del formulario. Como se observa en la Figura 50, cuando el usuario intenta enviar información incompleta o incorrecta, el sistema identifica los campos problemáticos y muestra mensajes de error específicos y contextuales (como "Código es obligatorio", "Descripción Corta es obligatorio", etc.) con un distintivo fondo rojo que destaca visualmente las áreas que requieren atención. Esta validación inmediata proporciona retroalimentación clara al usuario, reduciendo la frustración y agilizando el

proceso de completar formularios correctamente, al tiempo que garantiza la integridad de los datos antes de su envío al servidor.

- **Visualización detallada:** Modo expandido para ver todos los detalles de un registro.

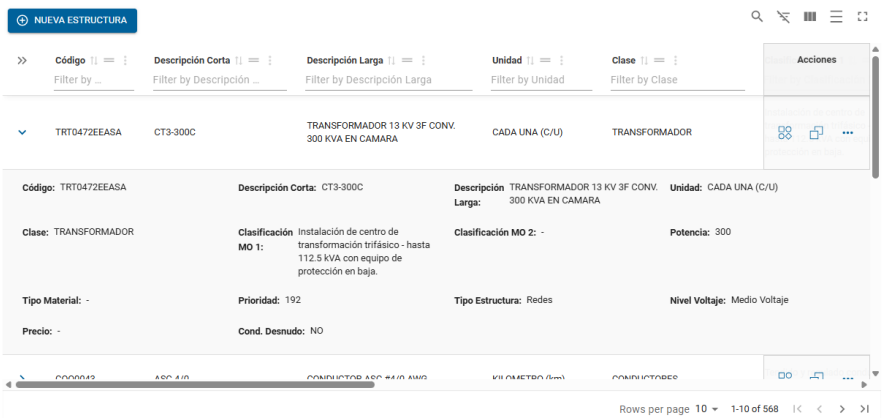


Figura 51. Detalles de un registro

Dado el caso que el formulario contenga numerosos campos como en la Figura 51 se facilita así la visualización sin tener la necesidad de desplazarse horizontalmente para ver todos los campos, a través del renderizado del detalle del registro con **`renderDetailPanel`**.

- **Filtrado avanzado:** Soporta filtrado por múltiples criterios.

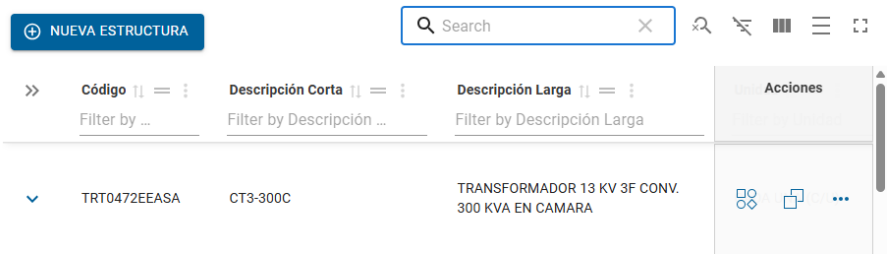


Figura 52. Barra de búsqueda en **`MaterialReactTable`**

La interfaz implementa capacidades de filtrado avanzado mediante el componente MaterialReactTable, que proporciona controles intuitivos para búsqueda y filtrado en cada columna de datos. Como se observa en la Figura 52, el sistema permite al usuario refinar la visualización de estructuras mediante un campo de búsqueda

global en la parte superior y filtros individuales para cada columna, facilitando así la localización rápida de elementos específicos en conjuntos de datos extensos.

- **Acciones personalizadas:** Permite añadir botones de acción específicos para cada fila.

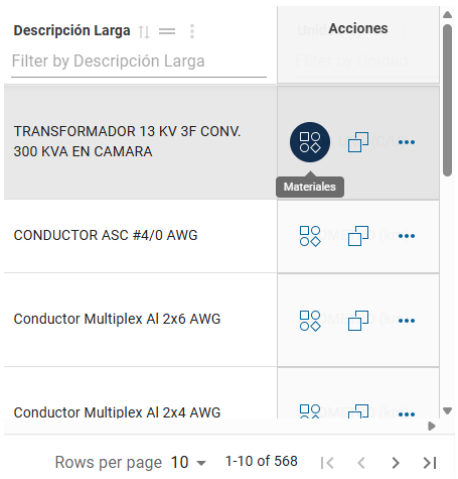


Figura 53. Opciones de *"Materiales"* y *"Clonar"*

El sistema implementa una funcionalidad de acciones personalizadas para cada registro de la tabla mediante la función **`customStructureActions`**, que enriquece la interfaz con botones interactivos específicos por fila. Como se observa en la Figura 53, cada estructura muestra un conjunto de iconos de acción en la columna "Acciones", incluyendo botones para acceder a los materiales asociados (representado por un icono de gestión de materiales) y para clonar la estructura completa (representado por un icono de duplicación). Estos botones utilizan componentes IconButton de Material UI y CalciteIcon de Esri, complementados con Tooltip para mejorar la usabilidad mediante descripciones emergentes como se muestra en la Figura 54.

```
1  const customStructureActions = (structure, sx) => {
2    return (
3     <>
4      <Tooltip title="Materiales" arrow>
5       <IconButton
6        color="success"
7        size="small"
8        onClick={(e) => {
9         e.stopPropagation();
10        handleViewStructure(structure);
```

```
11          }}
12          sx={sx}
13        >
14          <CalciteIcon icon={PAGE_ICONS.MATERIAL_TAB} />
15        </IconButton>
16      </Tooltip>
17      // Más botones…
18    </>
19   );
20  };
```

Figura 54. Definición de botones con acciones personalizadas

La función **`validateForm`** implementa un sistema de validación de formularios que se adapta dinámicamente al contexto. Primero verifica si la validación es necesaria según el tipo de diálogo activo, luego recorre cada campo del formulario aplicando reglas específicas según su configuración. La función maneja inteligentemente los campos ocultos en diferentes modos (creación o edición), comprueba si los campos son obligatorios (mediante valores booleanos o funciones dinámicas), verifica restricciones de longitud mínima y máxima, y aplica validaciones especializadas como la verificación de formato de email mediante expresiones regulares. Finalmente, almacena los mensajes de error para su visualización y determina si el formulario es válido basándose en la ausencia de errores, proporcionando así una experiencia de usuario informativa y previniendo el envío de datos incorrectos.

Para utilizar correctamente el componente **`CustomTable`** correctamente se describen a continuación las propiedades, tanto, para el componente propiamente dicho en la Tabla 16, como para la definición de campos en la Tabla 17:

Tabla 16. Propiedades de **`CustomTable`**

Propiedad	Descripción	Valor por defecto	Posibles valores
`data`	Array de datos a mostrar en la tabla	`[]`	Array de objetos
`fieldDefinitions`	Definición de campos para la tabla y formularios	`[]`	Array de objetos que definen cada campo
`title`	Título usado en encabezados y botones	`''`	String
`handleView`	Función para ver detalles de un elemento	`null`	Función
`uniqueIdField`	Campo que sirve como identificador único	`'id'`	String (nombre de campo)

`addButtonText`	Texto para el botón de añadir	`'Nuevo Item'`	String
`customActions`	Acciones personalizadas para cada fila	`null`	Función que retorna JSX
`withPadding`	Si se debe aplicar padding al contenedor	`false`	Boolean
`onAdd`	Función para añadir un elemento	`null`	Función
`onUpdate`	Función para actualizar un elemento	`null`	Función
`onDelete`	Función para eliminar un elemento	`null`	Función
`onFormChange`	Función para manejar cambios en el formulario	`null`	Función
`isInTab`	Si el componente está dentro de pestañas	`true`	Boolean
`topButtons`	Botones adicionales para la barra superior	`[]`	Array de objetos
`customDialogContents`	Contenidos para diálogos personalizados	`{}`	Objeto con configuraciones
`onDialogClose`	Función llamada al cerrar un diálogo	`null`	Función

Tabla 17. Definición de las columnas en `fieldDefinitions`

Propiedad	Descripción	Valor por defecto	Posibles valores
`fieldName`	Nombre del campo en los datos	`''`	`String`
`header`	Texto a mostrar como cabecera	`''`	`String`
`size`	Ancho aproximado de la columna	`150`	`Number`
`type`	Tipo de campo para el formulario	`'text'`	`'text', 'number', 'textarea', 'select', 'comboBox', 'switch', 'password', 'email', 'date'`
`required`	Si el campo es obligatorio	`false`	`Boolean, Function`
`disabledInEdit`	Si está deshabilitado en modo edición	`false`	`Boolean, Function`
`disabledInAdd`	Si está deshabilitado al añadir	`false`	`Boolean`
`hideInAdd`	Si se oculta al añadir	`false`	`Boolean`
`hideInEdit`	Si se oculta al editar	`false`	`Boolean, Function`
`maxLength`	Longitud máxima para texto	`null`	`Number`
`minLength`	Longitud mínima para texto	`null`	`Number`
`options`	Opciones para select/ combobox	`[]`	Array de objetos: `{value, label}`
`selectionMode`	Modo de selección para combobox	`'single'`	`'single', 'multiple'`
`readOnly`	Si el campo es de solo lectura	`false`	`Boolean`
`fullWidth`	Si el campo ocupa ancho completo	`false`	`Boolean`

`step`	Paso para campos numéricos	`1`	`Number`
`rows`	Filas para textarea	`2`	`Number`
`min`	Valor mínimo para campos numéricos	`null`	`Number`
`max`	Valor máximo para campos numéricos	`null`	`Number`

CustomComboBox

`CustomComboBox` es un componente de selección avanzado que maneja eficientemente grandes conjuntos de datos. Implementa optimizaciones como carga perezosa, búsqueda en tiempo real y visualización limitada de opciones para mejorar el rendimiento, como se muestra en la Figura 55. Entre las principales características de este componente se encuentran:

- **Manejo optimizado de grandes datasets:** Limita inicialmente las opciones mostradas.
- **Búsqueda en tiempo real:** Filtra las opciones mientras el usuario escribe.
- **Selección múltiple:** Permite seleccionar varios valores.
- **Persistencia de selecciones:** Mantiene las opciones seleccionadas visibles.
- **Interfaz de usuario amigable:** Muestra chips para valores múltiples seleccionados.



Figura 55. Filtrado de registros en **`CustomComboBox`**

```
const getInitialOptions = useCallback(() => {
 if (options.length > MAX_OPTIONS) {
  const initialOptions = options.filter(option =>
   existingValuesSet.has(option.value)
  );
  if (selectedOptions.length > 0) {
   selectedOptions.forEach(selectedOpt => {
    if (selectedOpt && !initialOptions.some(opt =>
              opt.value === selectedOpt.value)) {
     initialOptions.push(selectedOpt);
    }
   });
  }
  return initialOptions.length > 0 ?
   initialOptions : options.slice(0, 50);
 }
 return options;
}, [options, existingValuesSet, selectedOptions]);
```

Figura 56. Fragmento de código de **`CustomComboBox`**

La función **`getInitialOptions`** en la Figura 56 implementa una estrategia de optimización para manejar conjuntos grandes de datos en elementos de selección, abordando el problema de rendimiento cuando el número de opciones supera un umbral máximo predefinido. En estos casos, en lugar de cargar todas las opciones disponibles (lo que podría degradar significativamente el rendimiento), la función aplica un filtrado inteligente para mostrar únicamente las opciones que ya están siendo utilizadas en la tabla existente, identificadas mediante un conjunto de valores precalculado.

Además, garantiza que cualquier opción actualmente seleccionada siempre esté disponible en la lista mostrada, incluso si no forma parte del conjunto de valores existentes. Como medida de respaldo, si este filtrado resulta en una lista vacía, la función devuelve las primeras 50 opciones, asegurando así un balance óptimo entre usabilidad y rendimiento para conjuntos de datos extensos. Para su utilización se describen las propiedades existentes en la Tabla 18:

Tabla 18. Propiedades de **`CustomComboBox`**

Propiedad	Descripción	Valor por defecto	Posibles valores
`name`	Nombre del campo	**`null`**	**`String`**

`options`	Opciones disponibles para selección	`[]`	Array de objetos: `{value, label}`
`currentValue`	Valor(es) actualmente seleccionado(s)	`{}`	`String, Number, Array`
`required`	Si el campo es requerido	`false`	`Boolean`
`disabled`	Si el campo está deshabilitado	`false`	`Boolean`
`status`	Estado visual del campo	`undefined`	`'invalid', 'valid', undefined`
`onComboboxChange`	Función llamada al cambiar la selección	`null`	`Function`
`placeholder`	Texto cuando no hay selección	`Automático`	`String`
`errorMessage`	Mensaje de error a mostrar	`null`	`String`
`data`	Conjunto completo de datos para optimización	`[]`	`Array`
`selectionMode`	Modo de selección	`'single'`	`'single', 'multiple'`

CustomTabs

`CustomTabs` es un componente para organizar contenido en pestañas, basado en CalciteTabs de Esri. Proporciona una interfaz de usuario clara y consistente para navegar entre diferentes secciones de contenido, como en la Figura 57 y en la Figura 58. Entre las principales características de este componente se encuentran:

- **Organización por pestañas:** Agrupa contenido relacionado.
- **Estilo personalizable:** Diferentes disposiciones y tamaños.
- **Soporte para iconos:** Cada pestaña puede tener un icono representativo.
- **Estado persistente:** Mantiene la pestaña activa.
- **Callbacks:** Notifica cuando se cambia de pestaña.

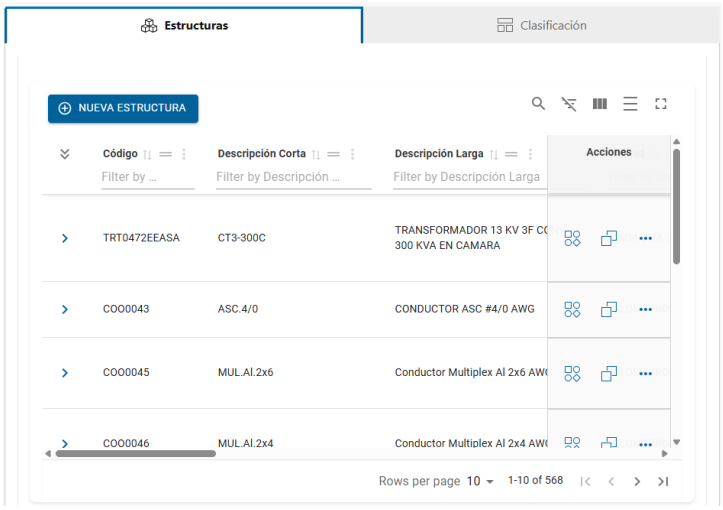


Figura 57. **`CustomTabs`** en la pestaña de *Estructuras*

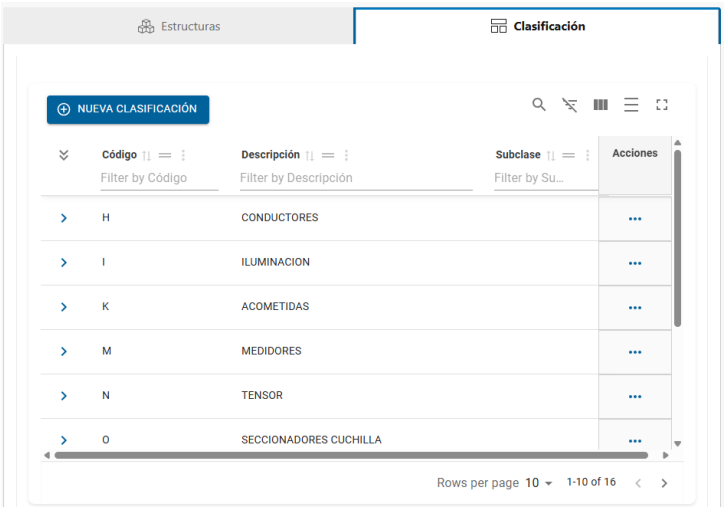


Figura 58. **`CustomTabs`** en la pestaña de *Clasificación*

```
<CalciteTabTitle
 key={`tab-title-${index}`}
 iconStart={tab.icon}
 {...(activeTab === index ? { selected: true } : {})}
 onCalciteTabsActivate={() => handleTabChange(index)}
 style={activeTab === index ? {
  fontWeight: 'bold',
  borderColor: 'var(--primary-color-variant)',
  borderWidth: '1px 3px 0 3px',
  borderStyle: 'solid',
  '--calcite-tab-icon-color-start': 'red',
 } : {
  backgroundColor: '#ededed',
 }}
```

```
15  >
16    {tab.label}
17  </CalciteTabTitle>
```

Figura 59. Fragmento de código de **`CustomTabs`**

El sistema de navegación por pestañas implementa un manejo elegante de la interacción del usuario mediante la función **`handleTabChange`** como se muestra en la Figura 59, que actualiza el estado interno y propaga el evento a componentes padres a través de un callback opcional, permitiendo una comunicación flexible entre componentes. Las propiedades disponibles se describen en la Tabla 19.

Tabla 19. Propiedades de **`CustomTabs`**

Propiedad	Descripción	Valor por defecto	Posibles valores
`tabsConfig`	Configuración de las pestañas	**`[]`**	Array de objetos con: **`label`, `icon` y `content`**
`layout`	Disposición de las pestañas	**`'center'`**	**`'center'`, `'inline'`**
`scale`	Tamaño de las pestañas	**`'l'`**	**`'s'`, `'m'`, `'l'`**
`bordered`	Si las pestañas tienen borde	**`true`**	**`Boolean`**
`defaultActiveTab`	Índice de la pestaña activa inicial	**`0`**	**`Number`**
`onTabChange`	Función llamada al cambiar de pestaña	**`null`**	**`Function`**

Complementariamente, el componente **`CalciteTabTitle`** aplica estilos condicionales sofisticados basados en el estado activo de cada pestaña, destacando visualmente la selección actual mediante una combinación de bordes acentuados, tipografía en negrita y colores distintivos definidos en variables CSS personalizadas, mientras mantiene un aspecto más sutil para las pestañas inactivas con un fondo gris claro. Para su utilización se presentan las propiedades existentes de **`tabsConfig`** en la Tabla 20:

Tabla 20. Propiedades de **`tabsConfig`**

Propiedad	Descripción	Valor por defecto	Posibles valores
`label`	Texto de la pestaña	**`null`**	**`String`**
`icon`	Icono de la pestaña	**`null`**	**`String`** (nombre del icono Calcite)
`content`	Contenido a mostrar	**`null`**	**`JSX`**

ActionBar

El **`ActionBar`** es el componente principal de navegación y control para el sistema de mapeo. Proporciona un panel lateral con herramientas específicas para interactuar con el mapa y administrar los recursos del sistema. Entre las principales características de este componente se encuentran:

- Panel lateral con barra de acciones para controlar el mapa
- Integración con paneles de herramientas específicas
- Comunicación bidireccional con el componente Map
- Gestión de estado de capas y acciones del usuario

Contiene varios componentes de panel que se muestran según la acción seleccionada, que se muestran en la barra lateral de la Figura 60 y la descripción detallada en la Tabla 21:

- **NetworkTypePanel:** Selección de tipo de red (existente/proyectada)
- **ManageElementsPanel:** Herramientas para crear y editar estructuras
- **ReportsPanel:** Generación de informes y reportes
- **UtilitiesPanel:** Importación de shapefiles y otras utilidades
- **PrintPanel:** Herramientas para imprimir y exportar el mapa
- **InfoPanel:** Información del sistema y atajos de teclado

Tabla 21. Definición de las opciones del **`ActionBar`**

Opción	ID	Icono	Descripción
Capas	action-layers	layers	Muestra/oculta el panel de capas del mapa
Tipo de Red	action-net-type	utility-network	Configura el tipo de red eléctrica a visualizar
Gestionar elementos	action-manager	system-management	Herramientas para gestionar elementos en el mapa
Reportes	action-reports	file-report	Genera reportes sobre elementos del mapa
Utilitarios	action-utils	arrow-up-down	Herramientas adicionales (importer shapes)

Imprimir	action-print	print	Herramientas para exportar e imprimir el mapa
Información	action-about	information	Información del sistema y ayuda

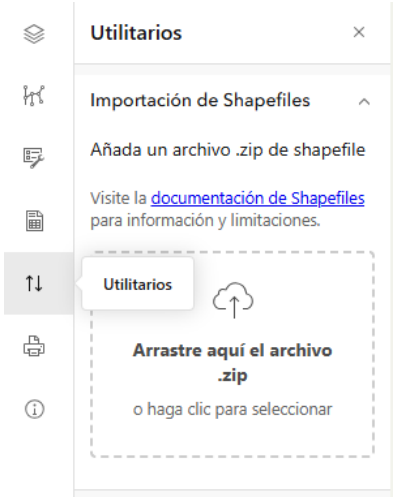


Figura 60. Barra lateral de **`ActionBar`**

Map

El componente **`Map`** es el núcleo del sistema de visualización geográfica, encargado de renderizar y gestionar todas las operaciones relacionadas con el mapa interactivo utilizando ArcGIS Maps SDK, como se muestra en la Figura 61.

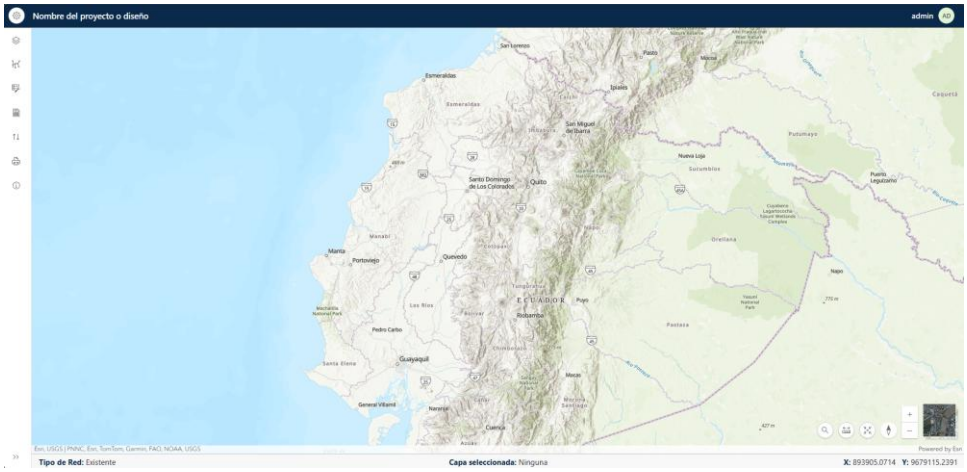


Figura 61. Pantalla del mapa

En términos de gestión de capas, el sistema permite cargar y visualizar múltiples capas gráficas configurables, con un panel de control que facilita mostrar u ocultar capas específicas según sea necesario. Cada capa puede ser personalizada con estilos únicos

aplicados automáticamente según el tipo de elemento y sus atributos específicos, proporcionando una representación visual clara y diferenciada de la información geográfica.

Las herramientas de edición constituyen una funcionalidad central del componente, ofreciendo capacidades completas para crear y modificar elementos geográficos. El sistema soporta la edición tanto de puntos como de líneas, incorporando validación de geometría para asegurar la integridad de los datos. Además, incluye un sistema de edición de atributos a través de diálogos de propiedades intuitivos que facilitan la actualización de información asociada a cada elemento.

Entre las funcionalidades adicionales, el componente incorpora un sistema de búsqueda que permite localizar elementos específicos basándose en sus atributos, una herramienta de medición de distancias para análisis espacial, y un *tooltip* informativo que muestra coordenadas UTM en tiempo real. La gestión de *popups* informativos se activa automáticamente al hacer clic en cualquier elemento del mapa, proporcionando acceso inmediato a información relevante.

Props

Las propiedades en la Figura 62 se detallan a continuación:

- **`onLayerChange`**: Función llamada cuando cambia la capa activa
- **`onCoordinateChange`**: Función para actualizar coordenadas al mover el cursor
- **`onViewCreated`**: Callback cuando la vista del mapa está lista
- **`layerListVisible`**: Controla la visibilidad del panel de capas

```
<Map
 onLayerChange={handleLayerChange}
 onCoordinateChange={handleCoordinateChange}
 onViewCreated={setMapView}
 layerListVisible={layerListVisible}
/>
```

Figura 62. Fragmento de código con las propiedades del componente **`Map`**

Configuración de capas

El archivo **`layerConfig.js`** es un módulo central para la configuración y gestión de capas en el sistema de mapas, proporcionando funciones y configuraciones esenciales para definir cómo se visualizan y comportan las diferentes capas geográficas.

Entre los componentes principales se encuentran:

- **`SPATIAL_REFERENCE_MAP`**: Define la referencia espacial utilizada por el mapa (EPSG:102100)
- **`LAYER_TYPES`**: Enumera los tipos de capas soportados (POINT, LINE)
- **`SVG_DEFAULT_SIZE`**, **`SVG_MIN_SIZE`**, **`SVG_MAX_SIZE`**: Constantes para control de tamaño de símbolos SVG
- **`dynamicLayerConfigs`**: Almacena configuraciones de capas cargadas desde el servidor
- **`layerStyles`**: Mantiene estilos específicos para cada tipo de elemento
- **`defaultLayerAttributes`**: Atributos predeterminados para nuevos elementos
- **`initialLayers`**: Definición inicial de capas a cargar

Este componente representa el núcleo de la integración visual del mapa, facilitando el acceso a configuraciones mediante funciones **`getter`** como **`getLayerConfigs`**, **`getLayerStyles`**, **`getDefaultLayerAttributes`** y **`getInitialLayers`**. Se utiliza principalmente durante la inicialización de la aplicación para establecer las configuraciones base, proporcionar representaciones visuales consistentes a los elementos geográficos, definir comportamientos específicos según el tipo de elemento, y gestionar la creación de nuevos elementos con atributos predefinidos.

Su diseño permite una estrecha integración con componentes clave del sistema como el componente **`Map`** para visualización, **`useLayerManagement`** para la gestión

dinámica de capas, **`FeatureDialog`** para la edición de atributos, y los sistemas de conversión de coordenadas para una representación espacial precisa.

Hooks

- **`useMapInitialization`**: Este hook gestiona la inicialización fundamental del mapa, estableciendo la configuración básica necesaria para su correcto funcionamiento y navegación. Se encarga de registrar eventos para el seguimiento de coordenadas del cursor, implementa funcionalidades de centrado automático, y maneja el estado de carga del mapa. Su papel es crucial durante la fase inicial de renderizado, ya que prepara el entorno de visualización con referencias espaciales adecuadas, habilita capacidades de navegación como zoom y desplazamiento, y configura limitaciones de vista para optimizar la experiencia del usuario. Mediante un sistema de eventos bien definido, este hook proporciona información en tiempo real sobre la posición del cursor, permitiendo mostrar coordenadas precisas y facilitar las operaciones de edición geográfica como se muestra en la Figura 63:

```
useEffect(() => {
 if (!mapReady || !viewRef.current) return;
 const pointerMoveHandler =
 viewRef.current.on("pointer-move", (evt) => {
  const point = viewRef.current.toMap(
   { x: evt.x, y: evt.y });
  if (point) {
   const [utmX, utmY] =
    CoordinateUtils.transformToUTM(point.x, point.y);
   updateCoordinates(utmX.toFixed(4), utmY.toFixed(4));
  }
 });
}, [mapReady, updateCoordinates]);
```

Figura 63. Fragmento de código de **`useMapInitialization.js`**

- **`useLayerManagement`**: Este hook proporciona un sistema de gestión para las capas de mapa, funcionando como intermediario entre la configuración de capas y su visualización efectiva como se muestra en la Figura 64. Administra el ciclo de vida completo de las capas, desde su creación y estilizado hasta su selección y actualización, manteniendo un registro centralizado de capas

activas, estilos aplicados y propiedades visuales. Incorpora funcionalidades sofisticadas como la gestión de subtipos de elementos, generación dinámica de leyendas, y opciones de ajuste ("snapping") para facilitar la edición precisa. El hook también implementa un sistema visual de selección de capas mediante un componente LayerList personalizado, permitiendo al usuario alternar entre diferentes capas temáticas y controlar su visibilidad de manera intuitiva, mejorando significativamente la experiencia de usuario al trabajar con datos espaciales complejos, como se muestra en la Figura 65:



Figura 64. Capas generadas dinámicamente

```
import LayerList from "@arcgis/core/widgets/LayerList";
const initializeLayers = useCallback((map, view) => {
  if (isLoading || layers.length === 0) return [];

  const layerList = new LayerList({
    view: view,
    selectionMode: "single",
    container: document.createElement('div')
  });

  view.ui.add(layerList.container, "top-left");
  setLayerListWidget(layerList);

  return updatedLayers;
}, [layers, layerStyles, isLoading]);
```

Figura 65. Fragmento de código de **`useLayerManagement.js`**

- **`useMapData`**: Especializado en la gestión del flujo de datos entre la aplicación y las capas del mapa, este hook maneja todas las operaciones de carga, transformación y persistencia de información geográfica. Implementa funciones para recuperar datos desde la API, transformarlos al formato requerido por el mapa, y aplicar manipulaciones geométricas necesarias como desplazamientos de líneas o ajustes de coordenadas. Incluye capacidades para generar contenido dinámico de popups basado en los atributos de los elementos, aplicar simbología condicional según los atributos, y preparar datos para su envío a servicios externos. Este componente es crucial para mantener la integridad de los datos durante las operaciones de edición y asegurar que los cambios realizados en el mapa se reflejen correctamente en el sistema de almacenamiento subyacente.

- **`useSketchTools`**: Este hook proporciona todas las funcionalidades relacionadas con las herramientas de dibujo y edición geométrica en el mapa como se ilustra en la Figura 66. Gestiona la creación, modificación y eliminación de elementos geográficos como puntos y líneas, implementando validaciones geométricas avanzadas como verificación de dependencias entre elementos o restricciones de conectividad. Coordina el flujo de trabajo de edición mediante la integración con diálogos de atributos, permitiendo una experiencia fluida donde los usuarios pueden editar tanto la geometría como las propiedades de los elementos. El hook, como se muestra en la Figura 67, incorpora mecanismos de confirmación para operaciones críticas, gestión de estados de múltiples selecciones, y funcionalidades de cancelación que preservan la integridad de los datos. Su diseño facilita la personalización de popups interactivos con opciones de edición directa, mejorando significativamente la usabilidad en tareas de diseño y mantenimiento de redes.

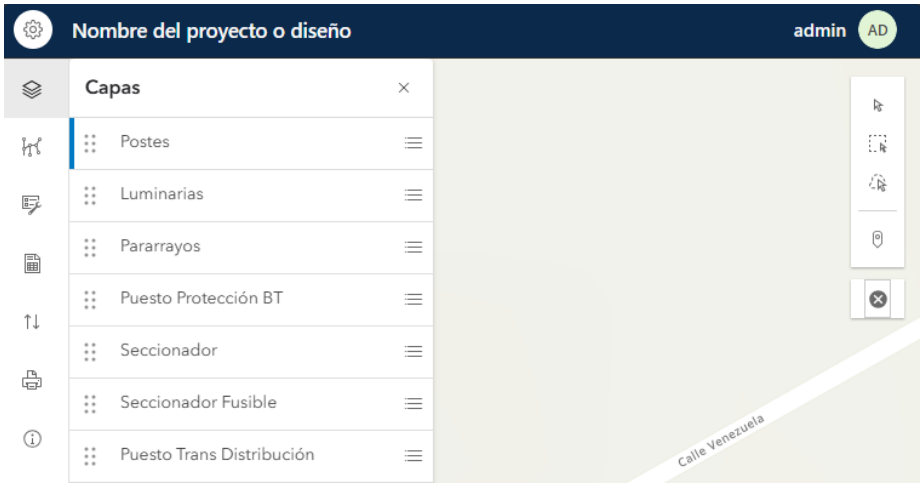


Figura 66. Herramientas de dibujo geográfico

```
// Creación de elementos
const handleSketchCreate = (event) => {
  const { graphic, state, tool } = event.detail;
  if (state !== "complete") return;

  // Validar y crear según tipo (punto/línea)
  if (tool === "point") {
    const { x, y } = graphic.geometry;
    const [utmX, utmY] =
     CoordinateUtils.transformToUTM(x, y);

    graphic.attributes =
     { ...currentAttributes, utmX, utmY };
    graphic.symbol = getLayerStyle(activeLayerId, "point",
     graphic.attributes);
  }
};

// Actualización de elementos
const handleUpdateGraphic = () => {
  selectedGraphics.forEach(graphic => {
    // Actualizar atributos
    graphic.attributes =
     { ...graphic.attributes, ...currentAttributes };

    // Actualizar estilo si cambió subtipo
    if (currentAttributes.SUBTIPO !==
      graphic.attributes.SUBTIPO) {
      const geometryType = graphic.geometry.type;
      graphic.symbol = getLayerStyle(
       graphic.attributes.layerId,
       geometryType,
```

```
33          graphic.attributes);
34        }
35      });
36
37      setDialogOpen(false);
38      setEditingMode(false);
39      sketchRef.current?.cancel();
40  };
```

Figura 67. Fragmento de código de **`useSketchTools.js`**

- **`useCoordinateTooltip`**: Especializado en proporcionar retroalimentación visual sobre coordenadas, este hook implementa un sistema de tooltip que muestra las coordenadas UTM en tiempo real durante operaciones de navegación y edición. Monitoriza el estado de las herramientas de dibujo para determinar cuándo debe mostrarse, implementa técnicas de optimización como debouncing para evitar actualizaciones excesivas, y maneja la transformación de coordenadas entre diferentes sistemas de referencia espacial. Su implementación va más allá de una simple visualización de coordenadas, ofreciendo una integración profunda con el estado de la aplicación para mostrar información contextual relevante solo cuando es necesaria, mejorando así la precisión durante tareas de edición y reduciendo la sobrecarga visual cuando no se requiere esta información.

Herramientas

- **`CoordinateTooltip`**: proporciona retroalimentación visual en tiempo real de las coordenadas UTM bajo el cursor mientras se edita el mapa como se muestra en la Figura 68. Diseñado con un enfoque minimalista, muestra un recuadro flotante que sigue al cursor con las coordenadas X e Y transformadas al sistema UTM, actualizado dinámicamente conforme el usuario se mueve por el mapa.



Figura 68. Tooltip con las coordenadas UTM

- **`CustomMapButtons`**: implementa una interfaz unificada para botones de acción personalizados en el mapa, permitiendo extender las funcionalidades estándar de ArcGIS con operaciones específicas de la aplicación como se ilustra en la Figura 69. Cada botón se configura mediante un esquema declarativo que define su apariencia, posición, comportamiento y retroalimentación visual. El sistema soporta estados dinámicos como activación, desactivación y visibilidad condicional. La implementación utiliza componentes **`CalciteButton`** junto con tooltips informativos para mejorar la accesibilidad.

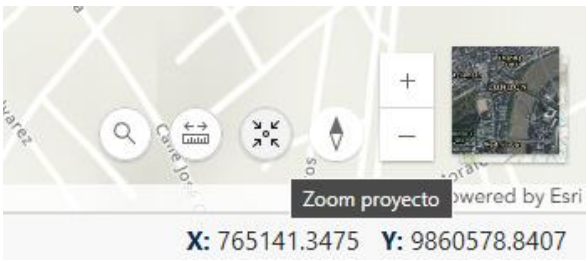


Figura 69. Botones personalizados en el mapa

- **`FeatureDialog`**: proporciona una interfaz dinámica para la edición y visualización de atributos de elementos geográficos. Implementado como un diálogo emergente, presenta formularios contextuales que se adaptan al tipo de elemento seleccionado, permitiendo modificar propiedades específicas como subtipo, valores técnicos y metadatos como se muestra en la Figura 70. Su diseño modular carga campos diferentes según la capa activa, conectándose con servicios de backend para obtener opciones válidas para cada atributo. El

componente soporta modos de creación y edición, maneja múltiples selecciones, incluye validación de datos, y ofrece una vista resumida cuando está colapsado. Proporciona retroalimentación visual sobre cambios pendientes y previene pérdida de datos mediante confirmaciones antes de cerrar, mejorando significativamente la experiencia de usuario en tareas de edición.



Figura 70. Ventana emergente con las propiedades de la capa seleccionada

- **`LayerSearch`**: implementa un sistema de búsqueda avanzado que permite localizar elementos específicos dentro de las capas del mapa según múltiples criterios como se ilustra en la Figura 71. Presenta una interfaz de usuario intuitiva con un panel deslizable donde los usuarios pueden seleccionar capas, definir filtros complejos basados en atributos, y visualizar los resultados en tiempo real. Su implementación incluye búsqueda incremental, filtrado por múltiples campos, y soporte para diferentes tipos de operadores de comparación (contiene, igual a, comienza con, etc.). El componente gestiona dinámicamente las opciones de filtrado según los tipos de datos de cada campo, optimiza el rendimiento con técnicas de búsqueda eficiente para grandes conjuntos de datos, y proporciona navegación directa al seleccionar un resultado, centrando automáticamente el mapa en el elemento encontrado.

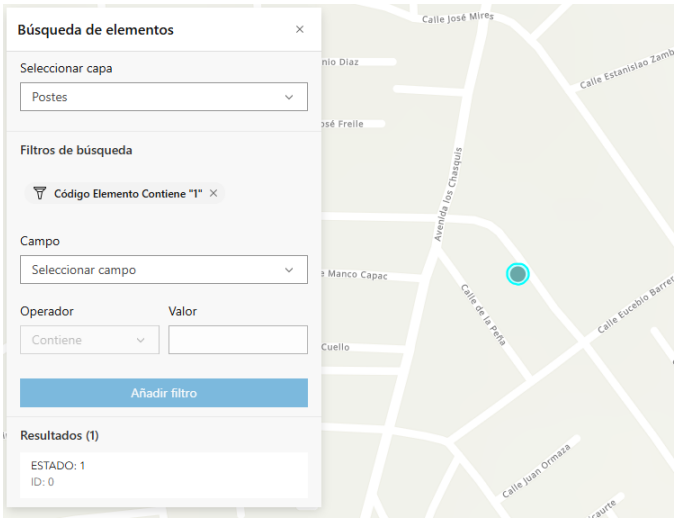


Figura 71. Ventana emergente para buscar elementos por capas

b. Uso de la Aplicación

- ***Páginas.*** El sistema utiliza los componentes reutilizables que se describieron anteriormente, manteniendo así la consistencia visual y comportamental a través de toda la aplicación. A cada página se le asigna una ruta específica, la cual va a estar protegida por el inicio de sesión y a la vez por los permisos de acceso del rol del usuario que intenta acceder. Esta organización permite a los usuarios navegar eficientemente entre diferentes secciones que se describen a continuación:

  - **Inicio de sesión `/login`:** implementa el punto de acceso principal al sistema, utiliza componentes **`CalciteCard`**, **`CalciteInput`**, **`CalciteButton`** y **`CalciteLoader`** para crear una interfaz atractiva y responsiva que verifica las credenciales del usuario antes de permitir el acceso. La página maneja estados para controlar la visualización de errores, la validación de campos y el proceso de carga durante la autenticación, proporcionando retroalimentación clara al usuario sobre el estado de su intento de inicio de sesión.



Figura 72. Pantalla de inicio de sesión SISDE Web

- **Recuperación de contraseña `/recover-password`:** proporciona un mecanismo para que los usuarios restablezcan su contraseña cuando las han olvidado. Implementa un formulario con validación utilizando componentes **`CalciteCard`**, **`CalciteInput`** y **`CalciteNotice`**, verificando el formato correcto del correo electrónico antes de procesar la solicitud.

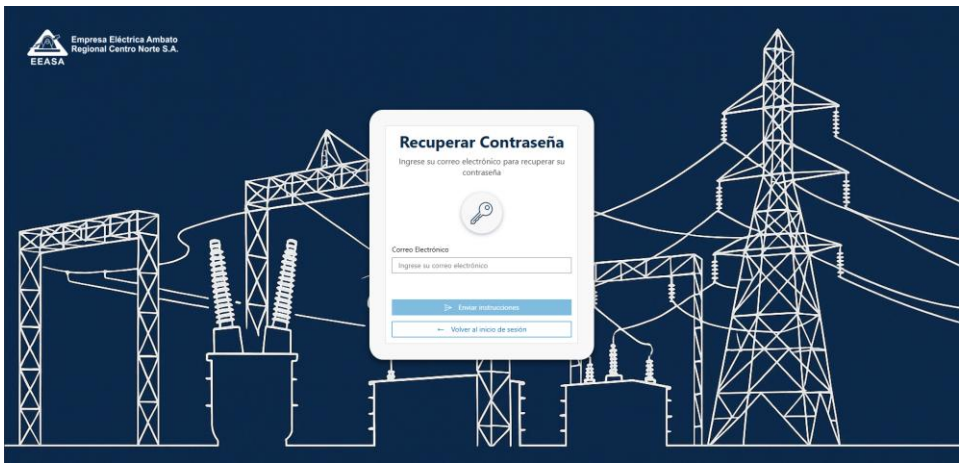


Figura 73. Pantalla de recuperación de contraseña SISDE Web

- **Inicio `/home`:** sirve como punto central de navegación tras el inicio de sesión, presentando una visión general del sistema SISDE y sus principales funcionalidades. Emplea componentes como **`CalciteCard`**, **`CalciteCarousel`**, **`CalciteCarouselItem`** y **`CalciteChip`** para mostrar información relevante, características del sistema y noticias importantes. La

interfaz incluye secciones bien definidas con opciones de acceso rápido a las diferentes secciones de la aplicación.

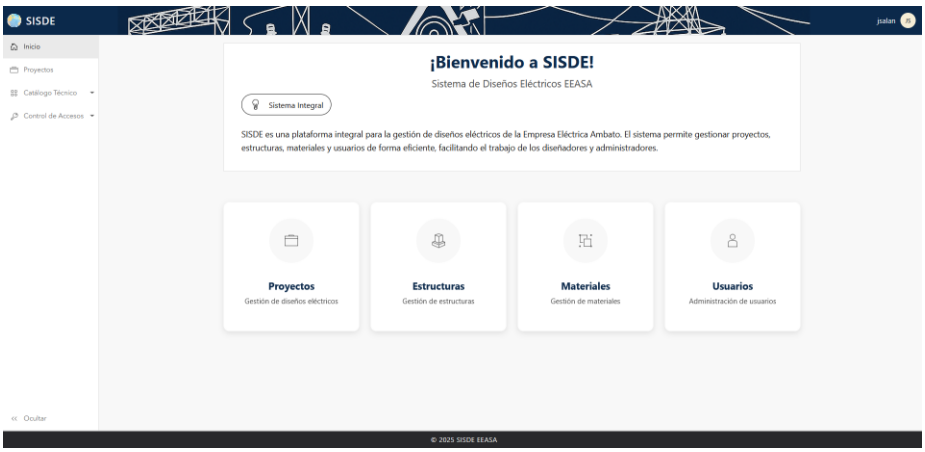


Figura 74. Pantalla de Inicio SISDE Web

- **Gestión de Proyectos `/projects`:** centraliza la gestión de diseños eléctricos en todas sus fases. Implementa un **`CustomTable`** avanzado con filtrado multicriterio, visualización paginada y acciones contextuales para cada proyecto. Integra componentes de formulario dinámicos con validación y menús en cascada para seleccionar ubicaciones geográficas (provincias, cantones, parroquias). La página permite la creación de nuevos proyectos, la edición de existentes, el seguimiento de su estado de aprobación, y proporciona acceso directo a la estación de trabajo (workstation) para la edición geoespacial de cada diseño, constituyendo así el núcleo operativo del sistema SISDE.

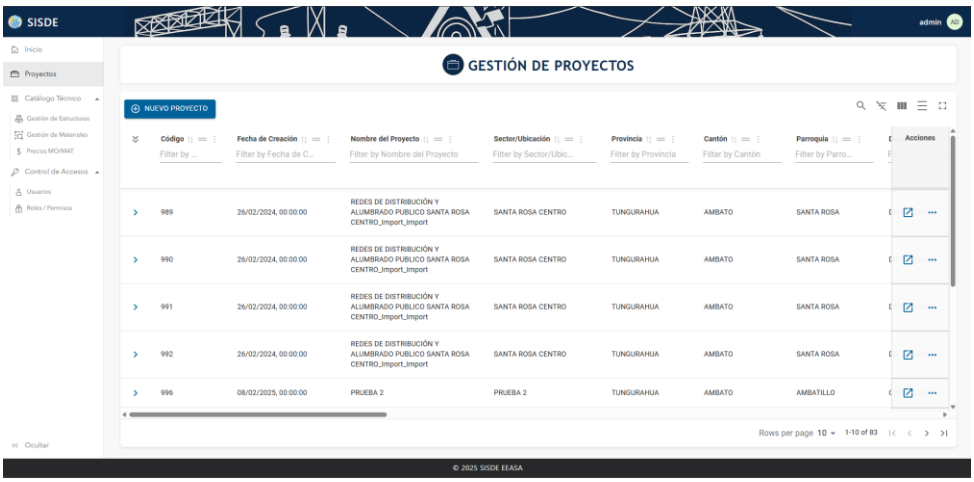


Figura 75. Pantalla de Gestión de Proyectos SISDE Web

- **Gestión de Materiales `/materials`:** permite la administración completa del catálogo de materiales utilizados en los diseños eléctricos. Implementa un sistema de pestañas mediante **`CustomTabs`** para separar la gestión de materiales y sus clasificaciones, utilizando componentes **`CustomTable`** para presentar listados con capacidades CRUD completas. La interfaz integra validaciones, filtros avanzados y acciones contextuales, facilitando a los usuarios la creación, edición, búsqueda y eliminación de registros de materiales con sus atributos específicos como código, descripción, clasificación y unidad de medida.

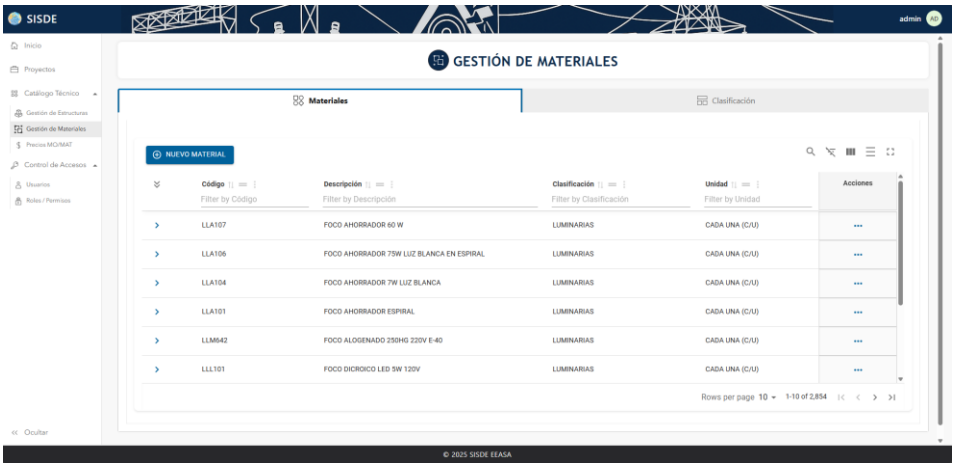


Figura 76. Pantalla de Gestión de Materiales SISDE Web

- **Gestión de Estructuras `/structures`:** gestiona los componentes estructurales de las redes eléctricas, presentando una interfaz organizada en pestañas mediante **`CustomTabs`**. Utiliza componentes **`CustomTable`** para listar las estructuras existentes y sus clasificaciones, y **`StructureComposition`** para visualizar los materiales asociados a cada estructura. La interfaz ofrece funcionalidades CRUD completas, incluyendo la capacidad de clonar estructuras existentes, y proporciona validación de formularios. La página implementa un sistema de navegación contextual que permite al usuario profundizar en los detalles específicos de cada estructura y administrar sus componentes asociados.

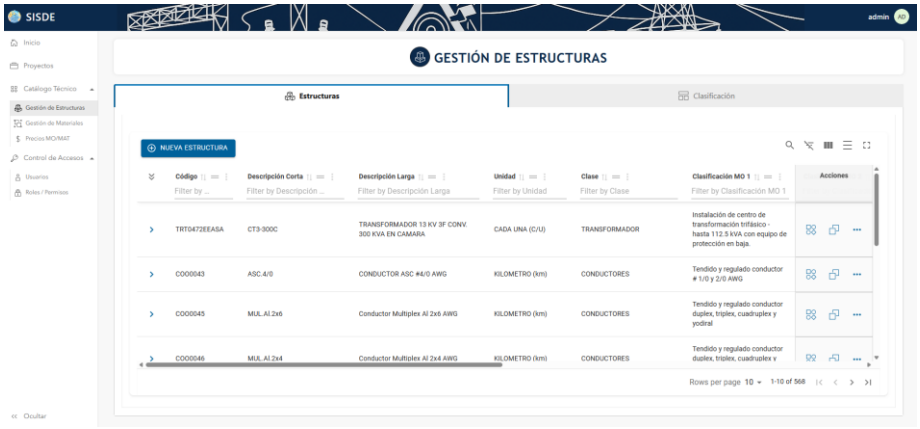


Figura 77. Pantalla de Gestión de Estructuras SISDE Web

- **Actualización de Precios Mano de Obra y Materiales `/price-update`:** facilita la importación y actualización masiva de costos de materiales y mano de obra en el sistema. Implementa componentes **`DropZone`** para la carga intuitiva de archivos mediante arrastre, combinados con **`CalciteNotice`** para proporcionar instrucciones claras y retroalimentación sobre el proceso. La interfaz detecta automáticamente el tipo de archivo importado (materiales o mano de obra) basándose en su estructura, y proporciona indicadores visuales sobre el estado del procesamiento. Esta página centraliza un proceso crítico para mantener actualizados los costos utilizados en los presupuestos de los proyectos eléctricos, garantizando que los diseños reflejen valores económicos precisos y actuales.

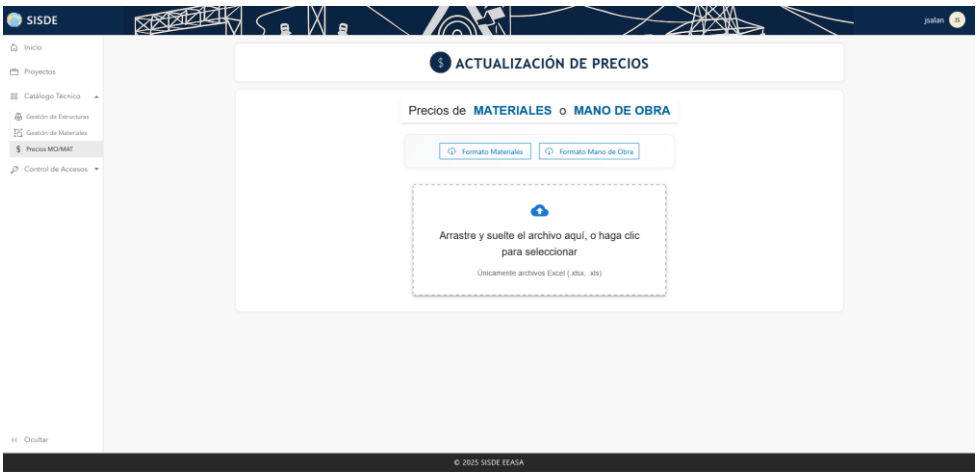


Figura 78. Pantalla de Actualización de Precios SISDE Web

- **Gestión de Usuarios `/users`:** proporciona herramientas administrativas para el control de acceso al sistema. Utiliza **`CustomTable`** con formularios dinámicos para gestionar usuarios, incluyendo información personal, departamentos asociados y control de estado (activo/inactivo/eliminado). Incorpora acciones personalizadas como el restablecimiento de contraseñas, validación de formatos de correo electrónico y cédula, y manejo seguro de credenciales. La interfaz permite asignar usuarios a departamentos específicos y gestionar sus fechas máximas de acceso, proporcionando un control granular sobre quién puede acceder al sistema y qué acciones puede realizar.

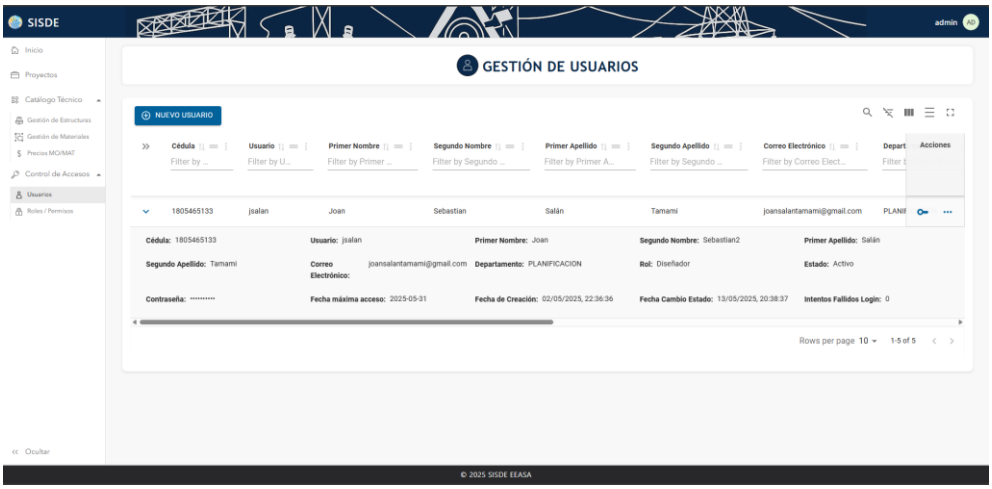


Figura 79. Pantalla de Gestión de Usuarios SISDE Web

- **Gestión de Roles y Permisos `/roles`:** implementa un sistema de control de acceso basado en roles que define las capacidades de los usuarios en el sistema. Utiliza **`CustomTabs`** para separar la gestión de roles y permisos, con formularios dinámicos que permiten asignar múltiples permisos a cada rol mediante selección múltiple. La interfaz facilita la creación y modificación de permisos individuales con descripciones detalladas y acciones específicas, así como la asignación de estos permisos a roles predefinidos o personalizados. Este diseño modular permite una administración flexible de los derechos de acceso, crucial para mantener la seguridad del sistema mientras se garantiza la funcionalidad para cada tipo de usuario.

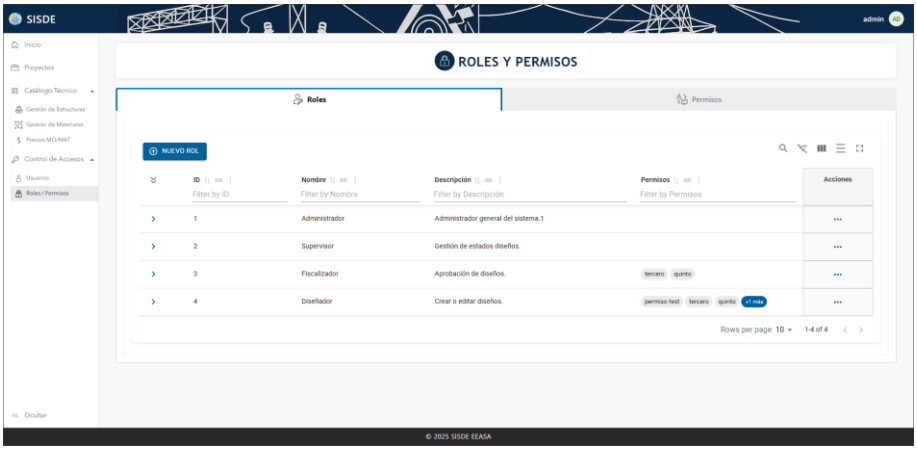


Figura 80. Pantalla de gestión de roles y permisos SISDE Web

- **Área de trabajo `/workstation`:** constituye el entorno principal de trabajo geoespacial del sistema, proporcionando una interfaz avanzada para la edición y visualización de redes eléctricas sobre mapas interactivos. Integra componentes ArcGIS como **`ArcgisMap`**, **`ArcgisSketch`**, **`ArcgisBasemapToggle`** y **`ArcgisCompass`** para crear un entorno GIS completo. Incluye paneles laterales personalizados para la gestión de capas, edición de atributos y configuración del proyecto, así como herramientas especializadas para medición, búsqueda y exportación de mapas. La interfaz permite a los diseñadores crear y editar elementos georreferenciados como postes, transformadores y líneas, asignándoles atributos específicos según los estándares eléctricos, constituyendo así el motor operativo para el diseño de redes eléctricas.

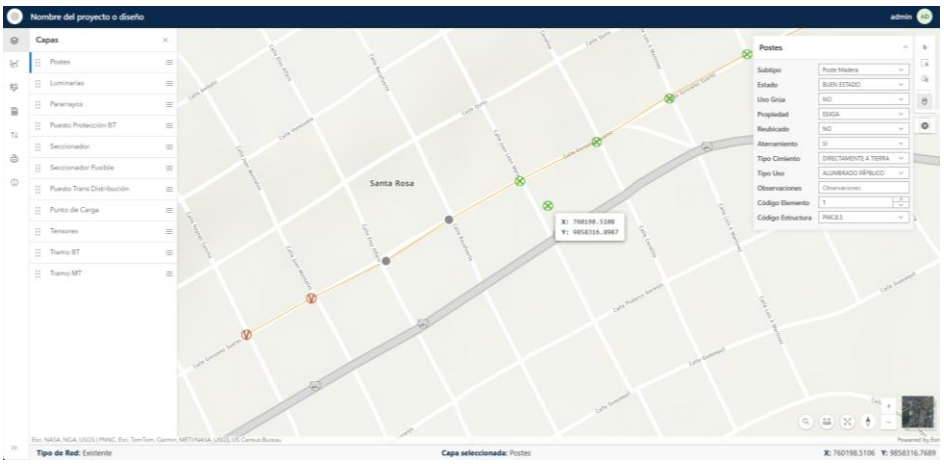


Figura 81. Pantalla del área de trabajo SISDE Web

3.2.5 Transición

La fase de Transición en la metodología RAD constituye la etapa final del desarrollo, donde se valida que el sistema SISDE Web cumple exitosamente con el objetivo planteado de implementar una solución basada en tecnologías web modernas. Por tanto, como etapas finales se disponen las pruebas de aceptación del sistema implementado, así como, su implantación en el servidor de WebLogic.

Pruebas de aceptación

Las pruebas de aceptación se realizaron siguiendo el plan de entrega establecido, validando cada requerimiento funcional y no funcional en su iteración correspondiente.

- **Iteración 1:** Funcionalidades Base del Mapa

Tabla 22. Pruebas de aceptación iteración 1

Pruebas de aceptación: Funcionalidades Base del Mapa			
ID	**Criterio de Prueba**	**Estado**	**Responsable**
1	Los elementos de red eléctrica se visualizan correctamente en el mapa con iconografía diferenciada	Aceptado	Usuario
2	Las funciones de zoom permiten acercar y alejar el mapa sin pérdida de calidad	Aceptado	Usuario
3	La navegación por el mapa es fluida y responde inmediatamente a las interacciones	Aceptado	Usuario
4	El panel de capas permite mostrar y ocultar elementos de forma independiente	Aceptado	Usuario
5	Las herramientas de dibujo crean puntos y líneas con precisión geométrica	Aceptado	Usuario
6	El sistema valida automáticamente las conexiones entre elementos dibujados	Aceptado	Programador
7	La interfaz web muestra mejoras significativas en usabilidad respecto al sistema de escritorio	Aceptado	Usuario
8	El sistema funciona correctamente en Chrome, Firefox y Edge sin degradación	Aceptado	Programador

- **Iteración 2:** Funcionalidades Avanzadas del Mapa

Tabla 23. Pruebas de aceptación iteración 2

Pruebas de aceptación: Funcionalidades Avanzadas del Mapa			
ID	**Criterio de Prueba**	**Estado**	**Responsable**

1	Los popups muestran información completa y detallada de cada elemento seleccionado	Aceptado	Usuario
2	La herramienta de medición calcula distancias con precisión métrica verificable	Aceptado	Usuario
3	El motor de búsqueda localiza elementos en tiempo real según múltiples criterios	Aceptado	Usuario
4	El editor de atributos permite modificar propiedades sin errores de validación	Aceptado	Usuario
5	El toggle de tipo de red cambia instantáneamente entre vista existente y proyectada	Aceptado	Usuario
6	Los archivos shapefile se cargan completamente sin pérdida de geometría o atributos	Aceptado	Programador
7	El lazy loading reduce significativamente los tiempos de carga inicial del mapa	Aceptado	Programador

- **Iteración 3:** Gestión de Proyectos y Reportes

Tabla 24. Pruebas de aceptación iteración 3

Pruebas de aceptación: Gestión de Proyectos y Reportes			
ID	**Criterio de Prueba**	**Estado**	**Responsable**
1	Los proyectos se crean, editan y eliminan correctamente con información geoespacial	Aceptado	Usuario
2	Los reportes se generan automáticamente con toda la información técnica y económica	Aceptado	Usuario
3	La exportación produce archivos PDF, JPG y PNG	Aceptado	Usuario
4	Las actualizaciones del sistema se aplican automáticamente sin intervención del usuario	Aceptado	Programador
5	Las operaciones básicas se completan consistentemente en menos de 3 segundos	Aceptado	Programador
6	La interfaz se adapta correctamente a dispositivos móviles y tablets para accesos ocacionales	Aceptado	Programador
7	Los mensajes de error proporcionan información clara para resolver problemas	Aceptado	Usuario

- **Iteración 4:** Gestión de Inventarios

Tabla 25. Pruebas de aceptación iteración 4

Pruebas de aceptación: Gestión de Inventarios			
ID	**Criterio de Prueba**	**Estado**	**Responsable**
1	El catálogo de materiales se organiza correctamente con sus clasificaciones	Aceptado	Usuario
2	Las estructuras eléctricas se clasifican con tipologías y propiedades técnicas completas	Aceptado	Usuario
3	La asociación entre materiales y estructuras mantiene control preciso de cantidades	Aceptado	Usuario
4	La clonación de estructuras replica completamente materiales y configuraciones	Aceptado	Usuario
5	Los archivos Excel con precios se procesan automáticamente con validación de formato	Aceptado	Programador

6	La vinculación entre elementos del mapa y estructuras del catálogo es bidireccional	Aceptado	Programador

- **Iteración 5:** Seguridad y Control de Acceso

Tabla 26. Pruebas de aceptación iteración 5

Pruebas de aceptación: Seguridad y Control de Acceso			
ID	**Criterio de Prueba**	**Estado**	**Responsable**
1	La autenticación JWT funciona correctamente con tokens de 24 horas de duración	Aceptado	Programador
2	Los usuarios se gestionan correctamente con roles diferenciados y permisos específicos	Aceptado	Usuario
3	El control de acceso restringe efectivamente funcionalidades según el rol asignado	Aceptado	Programador
4	Los datos personales se manejan de forma segura cumpliendo estándares de protección	Aceptado	Programador
5	El sistema resiste ataques de inyección SQL mediante procedimientos almacenados	Aceptado	Programador

Implantación del sistema SISDE Web

La fase de implantación del sistema SISDE Web constituye el proceso final para poner en funcionamiento la solución en el entorno de producción de la EEASA. Esta etapa crítica requiere una planificación meticulosa y la ejecución de procedimientos específicos para garantizar una transición exitosa desde el entorno de desarrollo hacia la infraestructura de producción.

a. Migración y Actualización de la Base de Datos

La migración de la base de datos representa uno de los aspectos más críticos del proceso de implantación. El sistema parte de la base de datos existente del SISDE de escritorio, la cual ha sido sometida a importantes modificaciones estructurales durante el desarrollo para mejorar la normalización, seguridad y funcionalidades del sistema web. Estas modificaciones incluyen la reestructuración completa del sistema de usuarios y permisos, la implementación de nuevas tablas para el control de acceso basado en roles, y el enriquecimiento de las configuraciones geoespaciales.

El proceso de migración se ejecuta mediante la exportación completa del esquema y datos de la base de datos modificada en el entorno de desarrollo, utilizando *SQL Developer*. Una vez generado el archivo de exportación, este se transfiere al servidor

de producción donde se procede con la importación, reemplazando el esquema anterior con la nueva estructura optimizada para el entorno web.

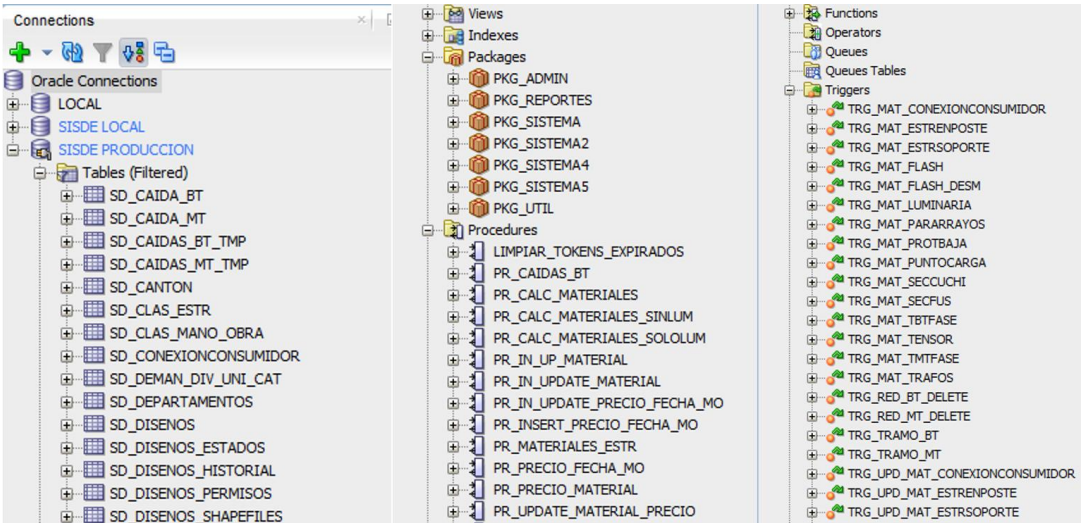


Figura 82. Base de datos en producción

b. Preparación del Entorno de Producción

A continuación, en el proceso de implantación se involucra la preparación del servidor de producción *Oracle WebLogic Server 12.2.1.4* existente en la infraestructura de EEASA. Se debe verificar que todas las dependencias necesarias estén correctamente instaladas, incluyendo *Java 1.8* y los drivers de *Oracle Database 19c*. La configuración del servidor debe incluir la creación de un datasource JNDI denominado *"sisdeDS"* que apunte a la base de datos Oracle de producción, asegurando que las credenciales y parámetros de conexión estén correctamente establecidos.



Figura 83. Datasorce *"sisdeDS"* en WebLogic

c. Despliegue de la Aplicación Backend

El despliegue del backend implica la generación del archivo WAR (Web Application Archive) del proyecto *Spring Boot* mediante el comando **`mvn clean install`**, asegurando que el perfil de producción esté activo para utilizar las configuraciones apropiadas. El archivo WAR resultante se despliega en el servidor WebLogic a través de la consola de administración, configurando la ruta de contexto como *"/WSSisde"* para mantener consistencia con la configuración de desarrollo.

d. Despliegue de la Aplicación Frontend

El frontend requiere una configuración específica para el entorno de producción, modificando las URLs base de los servicios para apuntar al servidor de producción en lugar del entorno de desarrollo. Se debe ejecutar el comando **`mvn clean install`** para generar la versión optimizada y minificada de la aplicación React, configurando las variables de entorno de producción. Los archivos resultantes del directorio *dist* se copian al directorio de recursos estáticos del servidor web y se integran directamente en la estructura de despliegue de WebLogic.

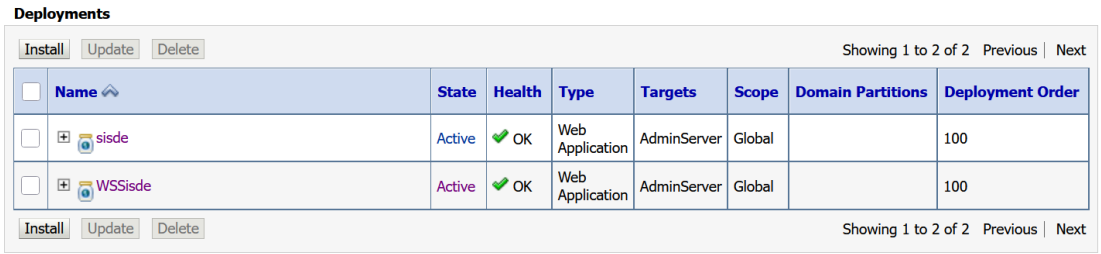


Figura 84. Aplicaciones frontend y backend desplegadas en WebLogic

3.3 Resultados

Una vez implantada la aplicación, se exponen las funcionalidades implementadas a los usuarios del sistema a quienes se les aplica nuevamente la encuesta SUS obteniendo los resultados en el Anexo B, los cuales demuestran una transformación radical en la experiencia de usuario, evidenciando el éxito de la migración hacia tecnologías web modernas. El puntaje promedio de 80.75 puntos sitúa al nuevo sistema en la categoría A-, clasificándolo como "excelente" según los estándares internacionales de

usabilidad. Esta mejora representa un salto cualitativo extraordinario respecto al sistema de escritorio, que obtuvo únicamente 61.04 puntos (categoría D), generando un incremento de 19.71 puntos que equivale a una mejora del 32.3% en la percepción de usabilidad.

Tabla 27. Resultados de la encuesta SUS del SISDE Web

Categoría SUS	Rango de Puntaje	Número de Participantes	Porcentaje
A+	84.1-100	7	11.67%
A	80.8-84	6	10%
A-	78.9-80.7	38	63.33%
B+	77.2-78.8	9	15%
B	74.1-77.1	0	0%
Categorías inferiores	<74	0	0%
TOTAL	-	60	100%

La distribución de usuarios por categorías en la Tabla 27 revela una concentración excepcional en los niveles superiores de usabilidad, con el 85% de usuarios ubicados en las categorías de grado A (A+, A, A-), indicando que la gran mayoría considera el sistema web como excelente. Particularmente notable es que el 63.33% de usuarios se encuentra en la categoría A-, reflejando una experiencia consistentemente positiva a través de la base de usuarios. Esta homogeneidad en las altas calificaciones contrasta dramáticamente con la dispersión problemática observada en el sistema de escritorio, donde solo el 1.67% alcanzó la categoría B y ningún usuario logró categorías superiores.

El hecho de que el 100% de usuarios califique el sistema web en categorías buenas o excelentes (B+ y superiores) representa un logro excepcional en términos de usabilidad. Esta distribución elimina completamente las categorías problemáticas (F, D, C-, C, C+, B-, B) que afectaban al 98.33% de usuarios en el sistema de escritorio. La ausencia total de calificaciones negativas o marginales indica que la migración web ha resuelto efectivamente los problemas fundamentales de usabilidad que aquejaban al sistema anterior.

Comparación y mejoras implementadas

La comparación directa entre ambos sistemas revela mejoras sustanciales en todas las dimensiones evaluadas. El incremento de 19.71 puntos en el puntaje SUS promedio representa una de las mejoras más significativas documentadas en proyectos de migración de sistemas GIS empresariales. Mientras el sistema de escritorio mantenía al 58.33% de usuarios en un estado de frustración constante (categorías D y inferiores), el sistema web ha logrado elevar al 85% de usuarios a niveles de excelencia, generando un incremento neto del 83.33% en la satisfacción del usuario.

La arquitectura web ha eliminado completamente las inconsistencias que caracterizaban al sistema de escritorio, como lo evidencia la concentración de usuarios en categorías superiores. Las mejoras en accesibilidad, reflejadas en la capacidad de acceso remoto sin restricciones geográficas, han transformado la experiencia operativa de los contratistas distribuidos en las tres provincias de operación. La implementación de actualizaciones automáticas y centralizadas ha resuelto definitivamente el problema más crítico identificado en las entrevistas: la necesidad de desplazamientos físicos y reinstalaciones manuales. Todos estos beneficios que se han obtenido se muestran en la Tabla 28.

Tabla 28. Comparación de resultados SUS en los sistemas SISDE de escritorio y web

Aspecto	Sistema de Escritorio	Sistema Web	Mejora Obtenida
Puntaje SUS Promedio	61.04 puntos (Categoría D)	80.75 puntos (Categoría A-)	+19.71 puntos (+32.3%)
Usuarios en Categorías Problemáticas (F, D, C-, C)	45 usuarios (75%)	0 usuarios (0%)	Reducción de usuarios insatisfechos
Usuarios en Categorías Excelentes (A+, A, A-)	1 usuario (1.67%)	51 usuarios (85%)	Incremento significativo en la satisfacción de los usuarios
Método de Actualización	Manual, presencial	Automático, remoto	100% eliminación de desplazamientos
Tiempo de Inactividad por Actualizaciones	Horas/días por usuario	Segundos para todos	>95% reducción de tiempo perdido
Consistencia de Versiones	Múltiples versiones activas	Una versión centralizada	100% sincronización
Costos Operativos de Soporte	Altos (desplazamientos + soporte individual)	Bajos (soporte centralizado)	Reducción significativa de costos

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- La migración del sistema SISDE a entorno web ha demostrado ser altamente exitosa, evidenciado por el incremento del puntaje SUS de 61.04 a 80.75 puntos, representando una mejora del 32.3% en la percepción de usabilidad. Esta transformación sitúa al sistema en la categoría A- (excelente), eliminando completamente los problemas de usabilidad que afectaban al 75% de los usuarios en el sistema de escritorio.

- El stack tecnológico seleccionado (React + JSX, Spring Boot, ArcGIS y Oracle) ha demostrado ser óptimo para las necesidades de EEASA, permitiendo aprovechar la infraestructura existente mientras se implementan tecnologías web modernas. La compatibilidad con WebLogic Server y Oracle Database facilitó la migración sin requerir cambios costosos en la infraestructura, mientras que la integración nativa de ArcGIS garantizó la continuidad de las capacidades geoespaciales especializadas.

- La arquitectura web centralizada ha resuelto definitivamente el problema crítico de las actualizaciones manuales, eliminando al 100% la necesidad de desplazamientos físicos de contratistas y reduciendo los tiempos de inactividad de horas/días a segundos. Esta mejora operativa impacta directamente en la productividad de los 50 contratistas distribuidos en las provincias de Tungurahua, Napo y Pastaza, transformando un proceso costoso e ineficiente en una operación automática y transparente.

- La metodología RAD (Rapid Application Development) fue fundamental para el éxito del proyecto, permitiendo la entrega incremental de funcionalidades y la validación temprana del usuario administrador del sistema SISDE de escritorio. La alta participación de usuarios durante las fases de diseño y feedback resultó en una solución que supera las expectativas, como lo demuestra que el 85% de usuarios califique el sistema en categorías excelentes (A+, A, A-).

4.2 Recomendaciones

- Se recomienda realizar una revisión exhaustiva y sistemática de la documentación oficial de ArcGIS SDK for JavaScript, ya que su complejidad inherente requiere un dominio profundo de su arquitectura y patrones de diseño. Aunque la comunidad de desarrolladores ofrece valiosas soluciones en general, se presentan requerimientos únicos que trascienden las implementaciones estándar, por lo que se debe optar la creatividad técnica sin preocupaciones para satisfacer las necesidades específicas del proyecto.

- Establecer un programa de actualizaciones trimestrales con revisiones anuales mayores, siguiendo las recomendaciones del administrador del sistema. Este ciclo permitirá incorporar mejoras funcionales basadas en el feedback continuo de usuarios, mantener la compatibilidad con nuevas versiones de ArcGIS y navegadores web, y añadir nuevas capacidades que respondan a la evolución de las necesidades operativas de EEASA.

- Considerar la expansión gradual hacia capacidades móviles nativas una vez consolidado el uso del sistema web responsivo actual. Aunque la compatibilidad móvil no fue identificada como prioritaria, el éxito de la migración web crea una base sólida para futuras mejoras que podrían incluir aplicaciones móviles especializadas para inspecciones de campo y capturas de datos geoespaciales in situ.

- Implementar un sistema de métricas y analytics para monitorear continuamente el rendimiento y uso del sistema, estableciendo indicadores clave como tiempos de respuesta, patrones de uso por módulo, y satisfacción de usuarios. Esto permitirá identificar oportunidades de optimización temprana y mantener los altos estándares de usabilidad alcanzados, facilitando la toma de decisiones informadas para futuras mejoras del sistema.

REFERENCIAS BIBLIOGRÁFICAS

[1] L. Ding, B. Tian, K. Chen y X. Li, «Implementation of serverless cloud GIS platform for land valuation,» *International Journal of Digital Earth,* vol. 14, pp. 1279-1297, 2021.

[2] D. H. Kum, T. W. Kim y I. J. Moon, «Near real-time interactive data visualization and dashboard package for oceanic gliders and modeling systems,» de *IEEE Conference on Technologies for Sustainability*, 2020.

[3] S. Li, S. Dragicevic, F. A. Castro, M. Sester, S. Winter, A. Coltekin, C. Pettit, B. Jiang, J. Haworth, A. Stein y T. Cheng, «An integrated GIS platform architecture for spatiotemporal big data,» *Future Generation Computer Systems,* vol. 89, pp. 381-392, 2018.

[4] R. Sugumaran y J. Burnett, «GIS Cloud Computing Methodology,» de *International Conference on Computing, Information Technology and Security*, 2018.

[5] P. K. Kiarie, «Developing a web-based Gis application for mapping the electrical grid network in Kenya case study: Nakuru County,» 2017.

[6] F. Barlocco, «A web-based georeferenced modelling tool for the Turin tramway network electrification system,» 2023.

[7] P. Kumar, R. Dasgupta, B. A. Johnson, C. Saraswat, M. Basu, M. Kefi y B. K. Mishra, «Web-GIS integrated open source mashup technology as a cue for integrated management in coastal megacities,» *Journal of Coastal Conservation,* vol. 24, pp. 1-15, 2020.

[8] A. C. M. Moura, B. C. Teixeira, S. A. Santana y N. C. B. Moura, «Scalability in the Application of Geodesign in Brazil: Expanding the Use of the Brazilian Geodesign Platform to Metropolitan Regions in Transformative-Learning Planning,» *Sustainability,* vol. 13, p. 6508, 2021.

[9] L. See, J. C. Laso Bayas, D. Schepaschenko, C. Perger, C. Dresel, V. Maus, C. Salk, J. Weichselbaum, M. Lesiv, I. McCallum y others, «Open source software and open educational material on land cover maps intercomparison and validation,» *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* vol. 42, pp. 61-66, 2018.

[10] W. Chen, Y. Zhou, E. C. Stokes y X. Zhang, «Large-scale urban building function mapping by integrating multi-source web-based geospatial data,» *Geo-spatial Information Science,* 2023.

[11] T. Ettefagh, «Designing and implementing a Web GIS platform for visitors' event experiences,» 2023.

[12] L. H. Mandel, «Managing In-Library Use Data: Putting a Web Geographic Information Systems Platform through its Paces,» *Information Technology and Libraries,* vol. 37, pp. 34-49, 2018.

[13] T. Minango, «Geographic information system management tool for underground electrical and telecommunication networks: case of the University of Zambia,» 2019.

[14] I. V. Bychkov, G. M. Ruzhnikov, V. V. Paramonov, A. S. Shumilov, R. K. Fedorov, K. G. Levi y S. Demberel, «Infrastructural approach and geospatial data processing services in the tasks of territorial development management,» *IOP Conference Series: Earth and Environmental Science,* vol. 190, p. 012048, 2018.

[15] L. F. Assis, K. R. Ferreira, L. Vinhas, L. Maurano, C. Almeida, A. Carvalho, J. Rodrigues, A. Maciel y C. Camargo, «TerraBrasilis: A Spatial Data Analytics Infrastructure for Large-Scale Thematic Mapping,» *ISPRS International Journal of Geo-Information,* vol. 8, p. 513, 2019.

[16] G. Giglione, A. Annibaldi, A. Iaccarino, R. Capancioni, G. Borghini, F. Ciabattoni, S. Illuminati, G. Pace, F. Memmola y G. Giantomassi, «An Integrated Web-Based GIS Platform for the Environmental Monitoring of

Industrial Emissions: Preliminary Results of the Project,» *Applied Sciences,* vol. 12, p. 3369, 2022.

[17] Z. Solpieva, «E-learning platform for web cartography,» 2023.

[18] M. Cignetti, D. Guenzi, F. Ardizzone, P. Allasia y D. Giordan, «An Open-Source Web Platform to Share Multisource, Multisensor Geospatial Data and Measurements of Ground Deformation in Mountain Areas,» *ISPRS International Journal of Geo-Information,* vol. 9, p. 4, 2020.

[19] C. T. White, A. Petrasova, V. Petras, L. G. Tateosian, J. Vukomanovic, H. Mitasova y R. K. Meentemeyer, «An open-source platform for geospatial participatory modeling in the cloud,» *Environmental Modelling & Software,* vol. 163, p. 105767, 2023.

[20] Esri, What is GIS?, 2022.

[21] P. A. Longley, M. F. Goodchild, D. J. Maguire y D. W. Rhind, Geographic Information Science and Systems, John Wiley & Sons, 2015.

[22] M. N. K. Boulos, «Web GIS in practice III: creating a simple interactive map of England's Strategic Health Authorities using Google Maps API, Google Earth KML, and MSN Virtual Earth Map Control,» *International Journal of Health Geographics,* vol. 4, p. 22, 2005.

[23] S. Quinn y U. Thakkar, «Enabling geospatial information in support of disaster management with an enterprise GIS approach,» *ISPRS International Journal of Geo-Information,* vol. 7, p. 25, 2018.

[24] T. Gratier, P. Spencer y E. Hazzard, OpenLayers 3: Beginner's Guide, Packt Publishing, 2015.

[25] S. Iacovella y B. Youngblood, GeoServer Beginner's Guide: Share geospatial data using Open Source standards, 2 ed., Packt Publishing, 2017.

[26] M. Kumar y J. Kerski, Spatial Analytics with ArcGIS Enterprise: Build powerful geospatial analysis workflows from the ground up, Packt Publishing, 2022.

[27] A. Charland y B. Leroux, «Mobile application development: web vs. native,» *Communications of the ACM,* vol. 54, pp. 49-53, 2011.

[28] A. I. Wasserman, «Software engineering issues for mobile application development,» *Proceedings of the FSE/SDP workshop on Future of software engineering research,* pp. 397-400, 2010.

[29] R. Wieruch, The Road to React: Your journey to master plain yet pragmatic React.js, Self-published, 2019.

[30] Y. Fain y A. Moiseev, TypeScript Quickly: Modern web development with TypeScript and Angular, Manning Publications, 2018.

[31] M. P. Peterson, Mapping in the Cloud: Web and Mobile Maps Development with Mapbox, Guilford Press, 2018.

[32] B. Szukalski y P. Thakkar, Advanced ArcGIS Developer Guide: Building professional real-world web applications, Packt Publishing, 2019.

[33] H. Nadkarni, The Complete Reference Java 2, McGraw-Hill, 1999.

[34] A. S. Tanenbaum, Modern Operating Systems, Pearson Prentice Hall, 2007.

[35] S. Rey, D. Arribas-Bel y L. J. Wolf, Geographic Data Science with Python, Chapman and Hall/CRC, 2019.

[36] C. W. Hall y R. Breidenbach, Spring in Action, 5 ed., Manning Publications, 2018.

[37] K. McArthur, PHP: Object-Oriented Solutions, 2 ed., Apress, 2018.

[38] A. Freeman, Pro ASP.NET Core 3: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages, 8 ed., Apress, 2020.

[39] M. Roberts y J. Chapin, Programming AWS Lambda: Build and Deploy Serverless Applications with Java, O'Reilly Media, 2018.

[40] F. Gutierrez, Spring Boot 2.0 Cookbook: Configure, test, extend, deploy, and monitor your Spring Boot application, 2 ed., Packt Publishing, 2019.

[41] C. Murray y R. Kothuri, Oracle Spatial and Graph Developer's Guide, 21c ed., Oracle Corporation, 2021.

[42] A. Martinez, PostGIS Essentials: Learn how to build powerful spatial database solutions with PostGIS quickly and efficiently, Packt Publishing, 2020.

[43] R. Kothuri, A. Godfrind y E. Beinat, Pro Oracle Spatial for Oracle Database, 2 ed., Apress, 2015.

[44] A. Machanic y J. DeWitt, Expert Performance Indexing in SQL Server, 2 ed., Apress, 2016.

[45] T. Cholakian y Z. Rogozinski, Elasticsearch in Action, 2 ed., Manning Publications, 2018.

[46] D. Ferraiolo, R. Chandramouli, R. Kuhn y V. Hu, Attribute Based Access Control, 2 ed., Artech House, 2022.

[47] W. Stallings, Cryptography and Network Security: Principles and Practice, 7 ed., Pearson, 2017.

[48] A. Chuvakin, K. Schmidt y C. Phillips, Logging and Log Management: The Authoritative Guide to Understanding the Concepts Surrounding Logging and Log Management, 2 ed., Syngress, 2019.

[49] Z.-R. Peng, D. Yu y E. Cai, «The state-of-the-art of WebGIS,» *Environment and Planning B: Urban Analytics and City Science,* vol. 45, p. 489–501, 2018.

[50] M. Benet, F. Golay y D. Rochat, «The three-tier architecture for geospatial web applications: concepts and performance implications,» *International Journal of Geographical Information Science,* vol. 33, p. 1682–1701, 2019.

[51] I. Nadareishvili, R. Mitra, M. McLarty y M. Amundsen, Microservice Architecture: Aligning Principles, Practices, and Culture, O'Reilly Media, 2016.

[52] M. Fowler y J. Lewis, «Microservices: a definition of this new architectural term,» *martinFowler.com,* vol. 25, 2014.

[53] J. L. Harrington, Relational Database Design and Implementation, 4 ed., Morgan Kaufmann, 2017.

[54] J. Martin, Rapid Application Development, Macmillan, 1991.

[55] P. Beynon-Davies, C. Carne, H. Mackay y D. Tudhope, «Rapid application development (RAD): an empirical review,» *European Journal of Information Systems,* vol. 8, nº 3, pp. 211-223, 1999.

[56] J. L. Whitten y L. D. Bentley, Systems Analysis and Design Methods, McGraw-Hill/Irwin, 2007.

[57] M. L. Despa, «Comparative study on software development methodologies,» *Database Systems Journal,* vol. 5, nº 3, pp. 37-56, 2014.

[58] J. Brooke, «SUS: A "quick and dirty" usability scale,» de *Usability Evaluation in Industry*, P. W. Jordan, B. Thomas, B. A. Weerdmeester y I. L. McClelland, Edits., Taylor & Francis, 1996, p. 189–194.

ANEXOS

Anexo A. Tabulación de respuestas SUS del sistema SISDE de escritorio.

Usuario	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Puntaje SUS	Grado SUS
1	3	4	4	2	2	1	2	3	4	3	55	F
2	1	4	4	2	2	1	3	2	3	2	55	F
3	3	1	3	2	3	3	2	2	3	1	62.5	C-
4	3	1	2	4	5	3	3	1	4	2	65	C
5	2	2	2	4	3	2	3	2	3	1	55	F
6	5	3	5	4	3	3	2	1	2	2	60	D
7	3	2	4	2	3	2	2	3	2	3	55	F
8	4	1	2	4	4	3	4	1	3	1	67.5	C
9	4	3	3	1	3	2	4	2	4	2	70	C
10	2	3	4	2	4	2	3	2	2	1	62.5	C-
11	1	3	3	2	5	2	3	3	3	1	60	D
12	2	1	5	4	4	3	4	1	4	2	70	C
13	4	4	3	1	3	2	4	3	2	2	60	D
14	3	2	2	2	5	2	3	3	3	3	60	D
15	5	2	3	2	3	1	4	2	2	2	70	C
16	5	2	2	2	5	2	4	3	3	3	67.5	C
17	3	3	4	2	2	3	3	1	2	2	57.5	D
18	3	2	3	3	4	2	3	3	4	2	75	B
19	3	1	5	4	3	2	4	2	3	2	67.5	C
20	1	4	5	3	3	2	4	3	4	3	55	F
21	3	1	2	4	4	1	2	2	2	1	60	D
22	3	3	4	2	4	2	3	3	4	3	62.5	C-
23	3	2	2	2	3	2	5	2	4	2	67.5	C
24	4	2	3	2	2	3	3	1	4	1	67.5	C
25	5	4	5	2	2	3	3	5	3	2	62.5	C-
26	2	2	4	2	2	2	5	2	3	1	67.5	C
27	3	4	2	2	4	3	4	3	3	2	55	F
28	4	4	4	4	3	2	4	2	4	3	60	D
29	4	2	3	1	3	1	3	2	3	2	70	C
30	1	1	4	2	2	2	4	2	4	1	67.5	C
31	1	2	3	3	4	3	3	2	4	2	57.5	D
32	3	1	2	4	4	2	3	1	2	2	60	D
33	3	3	4	2	5	3	3	2	3	2	65	C
34	3	3	2	1	2	1	4	3	3	1	62.5	C-
35	3	2	2	2	3	2	4	1	3	2	65	C
36	3	4	4	1	2	2	2	2	5	2	62.5	C-
37	3	3	3	2	5	2	5	3	2	1	67.5	C
38	2	1	2	3	2	2	4	1	4	2	62.5	C-
39	2	3	4	4	4	3	3	1	4	3	57.5	D
40	5	2	3	3	3	3	4	2	4	3	65	C
41	2	4	3	4	3	3	3	1	5	1	57.5	D
42	3	4	5	2	4	2	3	2	4	3	65	C
43	3	3	3	3	3	2	3	2	3	1	57.5	D
44	2	3	4	1	2	3	2	3	4	1	57.5	D
45	1	4	3	2	4	3	5	3	3	1	57.5	D
46	4	2	3	4	2	2	2	2	2	1	55	F
47	2	3	4	2	5	2	3	3	3	3	60	D
48	1	4	2	4	3	3	2	2	5	2	45	F
49	3	3	3	3	3	2	3	2	5	3	55	F
50	3	3	3	1	3	2	3	1	5	3	67.5	C
51	3	2	3	4	4	1	5	3	5	3	67.5	C
52	3	2	5	4	2	1	3	3	5	1	67.5	C
53	3	2	5	4	4	2	3	2	2	3	60	D
54	4	4	5	2	4	3	4	2	3	1	70	C
55	5	3	3	3	4	2	2	2	3	3	60	D
56	3	4	3	2	4	2	2	1	4	2	62.5	C-
57	4	2	2	1	3	2	3	2	5	2	70	C
58	3	3	3	3	4	3	2	3	3	2	52.5	D
59	3	4	3	2	4	2	2	1	2	1	60	D
60	2	3	2	2	4	2	4	1	2	1	62.5	C-

Anexo B. Tabulación de respuestas SUS del sistema SISDE Web.

Usuario	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	Puntaje SUS	Grado SUS
1	5	2	5	2	5	2	5	2	5	2	90	A+
2	4	2	4	2	4	2	4	2	4	2	80	A-
3	5	1	4	2	5	1	4	2	5	1	87.5	A+
4	4	2	5	1	4	2	5	1	4	2	85	A+
5	4	2	4	2	4	2	4	2	4	2	80	A-
6	5	1	5	1	5	1	5	1	5	1	95	A+
7	4	2	4	2	4	2	4	2	4	2	80	A-
8	5	2	4	2	4	2	5	1	4	2	82.5	A
9	4	2	4	2	4	2	4	2	4	2	80	A-
10	5	1	5	1	4	2	4	2	5	1	87.5	A+
11	4	2	4	2	4	2	4	2	4	2	80	A-
12	5	2	4	1	5	1	4	2	4	2	85	A+
13	4	2	4	2	4	2	4	2	4	2	80	A-
14	4	2	4	2	4	2	4	2	4	2	80	A-
15	5	1	4	2	4	2	4	2	4	1	82.5	A
16	4	2	4	2	4	2	4	2	4	2	80	A-
17	4	2	4	2	4	2	4	2	4	2	80	A-
18	5	1	5	1	4	2	4	1	5	1	90	A+
19	4	2	4	2	4	2	4	2	4	2	80	A-
20	4	2	4	2	4	2	4	2	4	2	80	A-
21	5	2	4	1	4	2	4	2	4	2	82.5	A
22	4	2	4	2	4	2	4	2	4	2	80	A-
23	4	2	4	2	4	2	4	2	4	2	80	A-
24	5	1	4	2	4	2	4	1	4	2	82.5	A
25	4	2	4	2	4	2	4	2	4	2	80	A-
26	4	2	4	2	4	2	4	2	4	2	80	A-
27	5	1	4	2	4	2	4	1	4	2	82.5	A
28	4	2	4	2	4	2	4	2	4	2	80	A-
29	4	2	4	2	4	2	4	2	4	2	80	A-
30	4	2	4	2	4	2	4	2	4	2	80	A-
31	4	2	4	2	4	2	4	2	4	2	80	A-
32	4	2	4	2	4	2	4	2	4	2	80	A-
33	5	2	4	1	4	2	4	2	4	1	82.5	A
34	4	2	4	2	4	2	4	2	4	2	80	A-
35	4	2	4	2	4	2	4	2	4	2	80	A-
36	4	3	4	2	4	2	4	2	4	2	77.5	B+
37	4	2	4	2	4	2	4	2	4	2	80	A-
38	4	2	4	2	4	2	4	2	4	2	80	A-
39	4	3	4	2	4	2	4	2	4	2	77.5	B+
40	4	2	4	2	4	2	4	2	4	2	80	A-
41	4	2	4	2	4	3	4	2	4	2	77.5	B+
42	4	2	4	2	4	2	4	2	4	2	80	A-
43	4	2	4	2	4	2	4	2	4	2	80	A-
44	4	3	4	2	4	2	4	2	4	2	77.5	B+
45	4	2	4	2	4	2	4	2	4	2	80	A-
46	4	2	4	2	4	3	3	2	4	2	75	B
47	4	2	4	2	4	2	4	2	4	2	80	A-
48	4	3	4	2	4	2	4	2	4	2	77.5	B+
49	4	2	4	2	4	2	4	2	3	2	77.5	B+
50	4	2	4	2	4	2	4	2	4	2	80	A-
51	4	2	4	3	4	2	4	2	4	2	77.5	B+
52	4	2	4	2	4	2	4	2	4	2	80	A-
53	4	2	4	2	4	2	4	2	4	2	80	A-
54	4	2	4	2	4	2	4	2	4	3	77.5	B+
55	4	2	4	2	4	2	4	2	4	2	80	A-
56	4	2	4	2	4	2	4	2	4	2	80	A-
57	4	2	4	2	4	3	4	2	4	2	77.5	B+
58	4	2	4	2	4	2	4	2	4	2	80	A-
59	4	2	4	3	4	2	4	2	4	2	77.5	B+
60	3	2	4	2	4	2	4	2	4	2	75	B