FORMATO N° 04
INFORME TÉCNICO DE PRÁCTICAS PRE PROFESIONALES QUE PRESENTA EL ESTUDIANTE[1]

El informe deberá iniciar con una carátula como la que se presenta a continuación:

1. **PORTADA**

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACION

CARRERA DE INGENIERIA DE SOFTWARE

INFORME DE:

[ ]	Pasantía	[ ]	Ayudante de Cátedra
[ ]	Práctica Pre Profesional No Remunerada	[ ]	Ayudante de Investigación
[X]	Servicio a la comunidad		

NOMBRE DE LA EMPRESA/ INSTITUCIÓN/ COMUNIDAD DONDE REALIZÓ LA PRÁCTICA PRE PROFESIONAL

NOMBRES Y APELLIDOS DEL ESTUDIANTE: JOHN JAIRO LIMONES GAVILANES

NOMBRES Y APELLIDOS DEL TUTOR ACADÉMICO: JENNY ALEXANDRA RUIZ ROBALINO
NOMBRES Y APELLIDOS DEL TUTOR EMPRESARIAL: CARLOS ROBERTO CEPEDA VACACELA

CALIFICACIÓN DEL INFORME 14

Firmado electrónicamente por:
JENNY ALEXANDRA RUIZ ROBALINO
Validar únicamente con FirmaEC

FIRMA DE TUTOR(A) ACADÉMICO(A)
JENNY ALEXANDRA RUIZ ROBALINO

FIRMA DEL ESTUDIANTE
JOHN JAIRO LIMONES GAVILANES

FIRMA DEL TUTOR EMPRESARIAL
CARLOS ROBERTO CEPEDA VACACELA

[1] **El informe será realizado y firmado por el estudiante y presentado a los tutores académico y empresarial, luego al coordinador de prácticas pre profesionales de la carrera y/o departamento.**

2. INTRODUCCIÓN

El presente informe tiene como propósito detallar las actividades desarrolladas durante la práctica pre profesional no remunerada realizada en la Universidad de las Fuerzas Armadas – ESPE, en el área de tecnologías de la información y desarrollo de software del Departamento de UTIC.

El trabajo se centró en dos ejes:

1. La investigación y análisis de factibilidad para implementar un módulo de códigos QR destinado al timbrado de asistencia, que quedó en etapa inicial sin llegar a consolidarse en un sistema funcional.

2. El desarrollo de un sistema de registro de reservas calendarizado, que avanzó hasta una fase operativa con pruebas de usabilidad y mejoras en la lógica de negocio.

Las actividades fueron ejecutadas por el estudiante en calidad de practicante, demostrando pertinencia con el perfil de la carrera, ya que implicaron el análisis de tecnologías emergentes.

Las prácticas se llevaron a cabo en la Universidad de las Fuerzas Armadas – ESPE, institución de educación superior que dispone de infraestructura tecnológica. Se identificaron procesos manuales como el control de asistencia y la reserva de espacios físicos, lo cual motivó la propuesta de soluciones digitales.

El estudiante se desenvolvió en las áreas de investigación de nuevas tecnologías, diseño de prototipos y desarrollo de aplicaciones web. Respecto al módulo QR, se investigaron bibliotecas y APIs en Spring Boot y Flutter, se diseñaron prototipos básicos y se exploraron opciones de integración cliente-servidor, pero el trabajo quedó en fase inicial sin llegar a consolidar un producto final. En cuanto al sistema calendarizado de reservas, se implementaron funcionalidades como gestión de roles con Spatie en Laravel, adaptación de la lógica de negocio del sistema MRBS, interfaces diferenciadas por rol y optimización de la base de datos.

Las prácticas se desarrollaron entre el 20 de junio del 2025 y el 15 de agosto del 2025, de acuerdo con lo estipulado en el convenio aprobado por la universidad.

Las actividades realizadas se justifican porque fortalecen competencias técnicas y profesionales alineadas al perfil de egreso, tales como: investigación aplicada, análisis de requerimientos, diseño de prototipos, administración de bases de datos y desarrollo de aplicaciones orientadas a la gestión académica.

El objetivo principal fue mejorar la gestión de asistencia y reservas dentro de la universidad mediante propuestas innovadoras. Aunque el módulo QR en inicial, se generaron aprendizajes relevantes en exploración de tecnologías de la información. El sistema calendarizado sí alcanzó una versión funcional, optimizando la administración de recursos. Entre los resultados de aprendizaje se destacan el fortalecimiento en el uso de frameworks modernos, la integración de roles y permisos, y la capacidad de proponer soluciones escalables.

3. DESARROLLO

Durante el período de prácticas, el estudiante participó en actividades de investigación y diseño.

3.1 Actividades relacionadas al módulo QR (etapa inicial):

3.1.1 Investigación de bibliotecas y APIs en Spring Boot y Flutter

Durante la práctica, se llevó a cabo una investigación exhaustiva sobre las diferentes bibliotecas y APIs disponibles para la implementación de un módulo de reconocimiento de códigos QR. En Spring Boot, se exploraron opciones para el procesamiento y validación de códigos QR mediante endpoints, mientras que en Flutter se analizaron paquetes que permiten la lectura y decodificación de QR desde dispositivos móviles. Esta fase de investigación permitió identificar las herramientas adecuadas para el proyecto, evaluando compatibilidad, facilidad de integración y mantenimiento, aunque no se avanzó hacia la implementación funcional.

3.1.2 Diseño conceptual de un prototipo básico de reconocimiento de QR con endpoints en Spring Boot

A partir de la investigación, se elaboró un diseño conceptual de un prototipo que permitiera recibir, procesar y validar códigos QR a través de endpoints en Spring Boot. El prototipo fue pensado como un modelo inicial para probar la lógica de comunicación entre el cliente y el servidor, así como la funcionalidad básica de lectura y respuesta ante códigos válidos o inválidos. Este diseño constituyó la base para un futuro desarrollo, pero no llegó a ejecutarse completamente ni a integrarse con sistemas existentes.

3.1.3 Exploración de comunicación entre aplicaciones móviles (Flutter) y servidor (Spring Boot)

Se realizaron análisis preliminares sobre la comunicación entre aplicaciones móviles desarrolladas en Flutter y un servidor backend en Spring Boot, con el objetivo de establecer cómo se podría gestionar la lectura de códigos QR y su posterior validación. Se estudiaron protocolos de comunicación, estructuras de datos y posibles flujos de intercambio de información, con especial atención a la seguridad y consistencia de los datos. Esta exploración proporcionó conocimiento técnico valioso, aunque permaneció en un nivel conceptual y experimental.

3.1.4 Análisis preliminar de integración de QR con sistemas de gestión académica

Finalmente, se llevó a cabo un análisis preliminar sobre cómo el módulo de QR podría integrarse con los sistemas de gestión académica de la universidad. Se evaluaron posibles puntos de conexión, estructuras de datos necesarias y la compatibilidad con las funciones existentes de registro de asistencia. El estudio permitió identificar desafíos potenciales y oportunidades de mejora, sirviendo como guía para futuras implementaciones.

Nota: Todo el proyecto QR se mantuvo en fase de investigación y prototipo inicial, sin llegar a una implementación funcional dentro del sistema.

3.2 Actividades relacionadas al sistema de reservas calendarizado (etapa funcional):

3.2.1 Definición de estructura de base de datos para roles y permisos

Durante la práctica se diseñó y estructuró la base de datos en MySQL, que permitió establecer un sistema de control de roles y permisos en la aplicación. Para ello, se identificaron las entidades principales y sus relaciones, definiendo tablas que gestionan usuarios, roles, permisos y la asignación de estos últimos de manera

dinámica. Esta labor fue fundamental para garantizar que la aplicación contara con un modelo de seguridad robusto y escalable, permitiendo posteriormente la integración de librerías especializadas de control de acceso. Además, se cuidó la normalización de las tablas y la consistencia de las claves foráneas con el fin de mantener la integridad de los datos en todo momento.

3.2.2 Implementación de Spatie en Laravel para control de accesos

Con la estructura de base de datos definida, se procedió a la implementación del paquete Spatie Laravel Permission, ampliamente utilizado para la gestión de roles y permisos en proyectos modernos. La integración de este paquete permitió asignar permisos específicos a cada rol y usuario, garantizando que las acciones dentro del sistema se alinearan con el principio de mínimo privilegio. De esta forma, el administrador contaba con acceso total a la configuración y gestión de la plataforma, mientras que los usuarios finales tenían permisos restringidos según su perfil. Esta configuración fortaleció la seguridad del sistema y facilitó la gestión centralizada de accesos.

3.2.3 Adaptación del sistema MRBS a las necesidades de la ESPE como un calendario

Una parte esencial del proyecto consistió en adaptar el sistema MRBS (Meeting Room Booking System), una herramienta de código abierto, a las necesidades particulares de la Universidad de las Fuerzas Armadas – ESPE. El sistema fue ajustado para que funcionara como un calendario institucional de reservas, gestionando aulas, laboratorios y otros espacios físicos de manera más organizada. La adaptación incluyó la modificación de la lógica de negocio, la traducción de la interfaz al contexto académico de la universidad, así como la personalización de estilos y funciones de acuerdo con la identidad visual de la institución. Esta fase representó un avance significativo hacia la optimización de la gestión de recursos universitarios.

3.2.4 Desarrollo de interfaces diferenciadas para administrador y usuario

El proyecto también contempló el desarrollo de interfaces gráficas adaptadas a los diferentes roles de la plataforma. El administrador disponía de una vista completa para gestionar usuarios, roles, permisos y reservas, mientras que el usuario regular accedía

a una interfaz simplificada enfocada en la consulta y creación de reservas. Este enfoque de diseño permitió que cada tipo de usuario tuviera acceso únicamente a las funciones pertinentes a sus responsabilidades, lo cual mejoró la experiencia de uso y redujo la posibilidad de errores en la interacción con el sistema.

3.2.5 Optimización de consultas y mejoras en la experiencia de usuario

Durante la implementación se identificaron consultas a la base de datos que podían generar lentitud en la plataforma, especialmente en operaciones de búsqueda y filtrado de reservas. Por esta razón, se aplicaron técnicas de optimización mediante índices, reducción de redundancia y mejoras en la construcción de sentencias SQL. Paralelamente, se introdujeron ajustes en la experiencia de usuario, tales como la incorporación de filtros dinámicos, validaciones en tiempo real y tiempos de carga más eficientes. Estos cambios mejoraron tanto el rendimiento del sistema como la satisfacción del usuario final.

3.2.6 Implementación de notificaciones y componentes reutilizables en frontend

Para incrementar la interacción y comunicación dentro del sistema, se desarrolló un módulo de notificaciones que informaba a los usuarios sobre eventos importantes, como la confirmación de reservas, cancelaciones o cambios en los horarios. Asimismo, se implementaron componentes reutilizables en el frontend, construidos bajo principios de modularidad y consistencia visual. Estos componentes permitieron mantener una interfaz uniforme, reducir tiempos de desarrollo y facilitar futuras ampliaciones de la plataforma.

3.2.7 Pruebas unitarias, de integración y despliegue en entorno de desarrollo

Una vez implementadas las funcionalidades principales, se realizaron pruebas unitarias con el fin de verificar el correcto funcionamiento de los módulos desarrollados de manera aislada. Posteriormente, se llevaron a cabo pruebas de integración para asegurar la interoperabilidad entre los diferentes componentes del sistema. Finalmente, el proyecto fue desplegado en un entorno de desarrollo controlado, lo cual permitió simular escenarios de uso real, identificar errores y aplicar correcciones antes de su futura puesta en producción. Estas pruebas garantizaron la estabilidad, confiabilidad y consistencia del sistema.

Tabla 1: Asignación de tareas

FECHA	ACTIVIDAD	TAREA	HORAS DE TRABAJO
2025-06-20	Capacitación y orientación de funciones, familiarización con tecnologías (Spring Boot y Flutter)	Se investigan y analizan las tecnologías clave, como Spring Boot para el backend y Flutter para el frontend, para comprender sus capacidades, fortalezas y cómo se integran con otras herramientas.	2
		Se define el proceso de desarrollo, incluyendo la metodología previa a la investigación del proyecto	2
		Se estudian y adoptan estándares de codificación	1
		Se instalan y configuran las herramientas necesarias, como JDK, IDEs (como IntelliJ o VS Code), y SDKs (Flutter), para crear un entorno de trabajo funcional y eficiente.	1
2025-06-23	Análisis de Spring Boot, Flutter con manejo de bases de datos en ejemplos de API	Revisión de funcionalidades clave en Spring Boot y Flutter.	2
		Configuración de entornos y compatibilidad de componentes.	2
		Práctica con ejemplos básicos de bases de datos y APIs.	2
2025-06-24		Conexión y gestión de base de datos MySQL en XAMPP.	2
		Diseño de APIs simples en Spring Boot para pruebas iniciales.	2
		Aplicación de principios SOLID en el diseño del código.	2
2025-06-25	Proyecto CRUD Usuarios (Spring Boot/IntelliJ) con xampp	Diseño del esquema de la base de datos para usuarios (tablas, campos, etc.).	3
		Configuración de la base de datos MySQL en XAMPP.	2
		Creación del proyecto de Spring Boot y configuración de dependencias.	1

2025-06-26		Desarrollo de la API REST para usuarios en Spring Boot (CRUD básico).	6
2025-06-27		Desarrollo de la interfaz de usuario en Flutter para el CRUD de usuarios.	6
2025-06-30		Integración completa y refinamiento del proyecto CRUD de usuarios	2
		Sincronización de Spring Boot y Flutter	2
		Aplicación de conceptos de Docker vistos previamente. Validación final de funcionalidad	2
2025-07-01	Proyecto Microservicios "Encuentro" con Eureka Server	Inicio del proyecto Encuentro: configuración inicial de microservicios y preparación del entorno.	3
		Implementación de Eureka Server para el descubrimiento de servicios.	3
2025-07-02		Desarrollo del primer microservicio (usuarios) y registro en Eureka Server.	3
		Desarrollo del segundo microservicio (productos/servicios) y conexión con el de usuarios.	3
2025-07-03		Configuración de la comunicación entre microservicios mediante Eureka.	3
		Implementación de pruebas básicas de interacción entre los servicios desarrollados.	3
2025-07-04		Integración de un sistema de monitoreo básico para supervisar el estado de los microservicios.	3
		Diseño de un API Gateway para centralizar el acceso y mejorar la seguridad del proyecto Encuentro.	3
2025-07-07	Estudio e Investigación: Módulo Reconocimiento QR	Investigación inicial de bibliotecas y APIs para QR en Spring Boot, sin implementación completa.	3
		Revisión de alternativas para reconocimiento QR en proyectos Java, aún en análisis.	3
2025-07-08		Desarrollo parcial de un prototipo básico de reconocimiento QR en Spring Boot.	3

		Pruebas preliminares de funcionamiento del prototipo con end points, sin consolidación.	3
2025-07-09		Análisis de integración de módulos QR en aplicaciones Flutter, en etapa inicial.	3
		Exploración de paquetes y dependencias de Flutter para lectura de QR, sin aplicación práctica.	3
2025-07-10		Estudio de la comunicación entre Flutter y Spring Boot para gestión de QR, con avance limitado.	6
2025-07-11		Investigación de integración de QR en Flutter con end points, sin pruebas concluyentes.	6
2025-07-14	Preparación para el Cambio de Proyecto y análisis para registros	Definir la estructura de la base de datos para roles y permisos.	3
		Configurar el entorno de Laravel para usar Spatie.	3
2025-07-15		Implementar la asignación de roles a usuarios.	3
		Crear la interfaz de usuario para la gestión de roles.	3
2025-07-16		Desarrollar Request Validators para formularios de registro y reserva.	3
		Integrar los validadores en los formularios correspondientes.	3
2025-07-17		Realizar pruebas unitarias de la asignación de roles.	3
		Realizar pruebas de integración de los validadores.	3
2025-07-18		Ajustar la configuración de permisos de Spatie.	3
		Generar pantallas iniciales por rol.	3
2025-07-21	Diseño y Desarrollo de Pantallas Iniciales	Diseñar wireframes y maquetas de las pantallas principales.	3
		Desarrollar la interfaz de usuario para el administrador.	3
2025-07-22		Desarrollar la interfaz de usuario para el usuario regular.	3
		Desplegar la plataforma en un entorno de	3

		desarrollo.	
2025-07-23		Analizar el código de MRBS para la adaptación.	3
		Modificar el código de MRBS para la lógica de negocio de Espe Reg.	3
2025-07-24		Desarrollar la funcionalidad para gestionar el calendario.	3
		Implementar el módulo de gestión de ubicaciones.	3
2025-07-25		Optimizar el diseño de la interfaz de usuario de MRBS.	3
		Adaptar la interfaz de MRBS a la identidad visual de Espe Reg.	3
2025-07-28	Refinamiento de Módulos y Experiencia de Usuario	Realizar ajustes menores en el despliegue de la plataforma.	3
		Optimizar la configuración del servidor para la producción.	3
2025-07-29		Refinar las funcionalidades de registro y reserva.	3
		Revisar la lógica de negocio del proyecto.	3
2025-07-30		Optimizar la eficiencia de las consultas a la base de datos.	3
		Mejorar la interactividad y fluidez de la interfaz de usuario.	3
2025-07-31		Implementar animaciones y transiciones sutiles.	3
		Implementar el diseño de un sistema de notificaciones para usuarios.	3
2025-08-01		Desarrollar componentes reutilizables para la interfaz de usuario (frontend).	3
		Mejorar la experiencia general del usuario.	3
2025-08-04	Preparación del Despliegue y Presentación	Explorar e implementar integraciones con sistemas de calendarios externos.	3
		Optimizar el rendimiento de las consultas de la base de datos.	3
2025-08-05		Revisar las configuraciones de seguridad de la aplicación.	3
		Preparar la base de datos final y los scripts de despliegue.	3

2025-08-06		Realizar las migraciones y Seed de la base de datos.	3
		Cambiar el entorno del proyecto de test a despliegue.	3
2025-08-07		Refinar la interfaz de usuario y la experiencia general.	3
		Preparar la presentación del proyecto como vistas del usuario y administrador.	3
2025-08-08		Incluir demostraciones en vivo en la presentación.	3
		Realizar una primera recolección de posibles modificaciones.	3
2025-08-12	Modificaciones Post-Presentación y Validación(Post-Presentación)	Recopilar y analizar el feedback de la presentación.	3
		Identificar las áreas prioritarias para mejoras y correcciones.	3
2025-08-13		Priorizar la implementación de las primeras modificaciones.	3
		Implementar las correcciones solicitadas.	3
2025-08-14		Implementar las primeras mejoras de interfaz	3
		Realizar pruebas unitarias de los módulos modificados.	3
2025-08-15		Realizar pruebas de integración para asegurar que no haya regresiones.	2
		Validar las modificaciones con los stakeholders.	2
		Generar un informe de las modificaciones implementadas. Reade.me	2
Total			240

.

4. CONCLUSIONES

La práctica permitió al estudiante fortalecer competencias en investigación tecnológica y desarrollo de software aplicado. Aunque el módulo QR no se consolidó, el análisis realizado aportó una base para futuros proyectos.

En contraste, el sistema de reservas calendarizado alcanzó una versión funcional que puede ser integrada en procesos institucionales.

Los objetivos planteados se cumplieron parcialmente en el caso del módulo QR y plenamente en el caso del sistema de reservas.

5. RECOMENDACIONES

Se recomienda continuar con la investigación y desarrollo del módulo QR hasta lograr una implementación estable, ya que esta funcionalidad puede optimizar de manera significativa el control de asistencia.

Asimismo, se sugiere ampliar el sistema de reservas calendarizado para integrarlo con plataformas móviles y servicios en la nube.

Finalmente, se recomienda documentar las lecciones aprendidas para que futuras prácticas puedan dar continuidad a los proyectos iniciados.

6. ANEXOS

Anexo 1 – Figura 1

Listado de rutas GET de la aplicación de reservas

A continuación, se presenta el listado de las rutas GET configuradas en la aplicación de reservas, las cuales permiten acceder a las diferentes funcionalidades del sistema desde el frontend o mediante pruebas en el entorno de desarrollo. Este listado facilita la comprensión de la estructura de navegación y los endpoints disponibles para la consulta de información dentro de la plataforma:

Tabla 2: Lista de rutas GET

Ruta	Descripción	Método	Observaciones
/reservas	Obtiene todas las reservas registradas	GET	Endpoint principal de consulta
/reservas/{id}	Obtiene los detalles de una reserva específica	GET	Requiere ID de la reserva
/usuarios	Lista todos los usuarios registrados	GET	Endpoint protegido por permisos
/usuarios/{id}	Obtiene información de un usuario específico	GET	Requiere ID de usuario
/aulas	Lista todas las aulas disponibles para reserva	GET	Permite filtrado por ubicación
/calendario	Muestra el calendario de reservas por día/mes	GET	Integrado con la interfaz del calendario

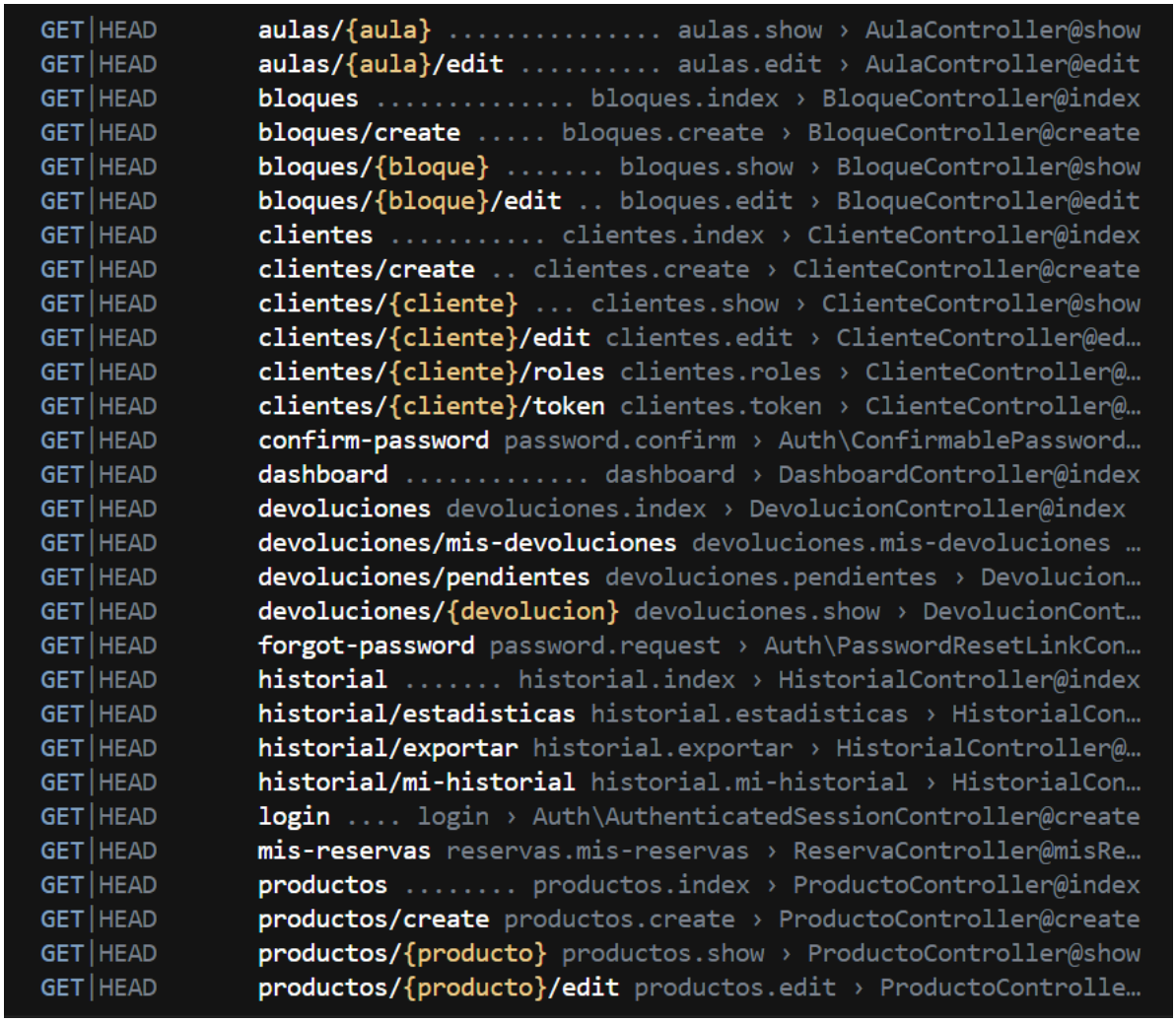

Figura 1. Listado de rutas GET de la aplicación de reservas, utilizado para pruebas, consulta de datos y desarrollo de interfaces.

Anexo 2 – Figura 2

Listado de rutas PATCH de la aplicación de reservas

En la siguiente tabla se presentan las rutas PATCH configuradas en la aplicación de reservas. Estas rutas permiten realizar actualizaciones parciales sobre recursos específicos, tales como cambiar estados, marcar reservas completadas o restaurar usuarios. La documentación de estas rutas contribuye a comprender la estructura de la API y las funcionalidades disponibles para la gestión dinámica de la información.

Tabla 3: PATCH de la aplicación de reservas

Ruta	Nombre del endpoint	Controlador / Acción	Descripción
/aulas/{aula}/toggle-status	aulas.toggleStatus	AulaController@toggleStatus	Cambia el estado activo/inactivo de un aula
/bloques/{bloque}/toggle-status	bloques.toggleStatus	BloqueController@toggleStatus	Cambia el estado activo/inactivo de un bloque
/clientes/{cliente}/toggle-status	clientes.toggle-status	ClienteController@toggleStatus	Activa o desactiva un cliente en el sistema
/devoluciones/{devolucion}/marcar-devuelto	devoluciones.marcar-devuelto	DevolucionesController@marcarDevuelto	Marca un artículo como devuelto
/productos/{producto}/toggle-status	productos.toggle-status	ProductoController@toggleStatus	Cambia el estado activo/inactivo de un producto
/profile	profile.update	ProfileController@update	Actualiza los datos del perfil del usuario
/reservas/{reserva}/completada	reservas.marcar-completada	ReservaController@marcarCompletada	Marca una reserva como completada
/reservas/{reserva}/estado	reservas.cambiar-estado	ReservaController@cambiarEstado	Cambia el estado general de la reserva
/reservas/{reserva}/expirada	reservas.marcar-expirada	ReservaController@marcarExpirada	Marca una reserva como expirada
/users/{id}/restore	users.restore	UserController@restore	Restaura un usuario previamente eliminado

/users/{user}/toggle-status	users.toggle-status	UserController@toggleStatus	Activa o desactiva un usuario en el sistema

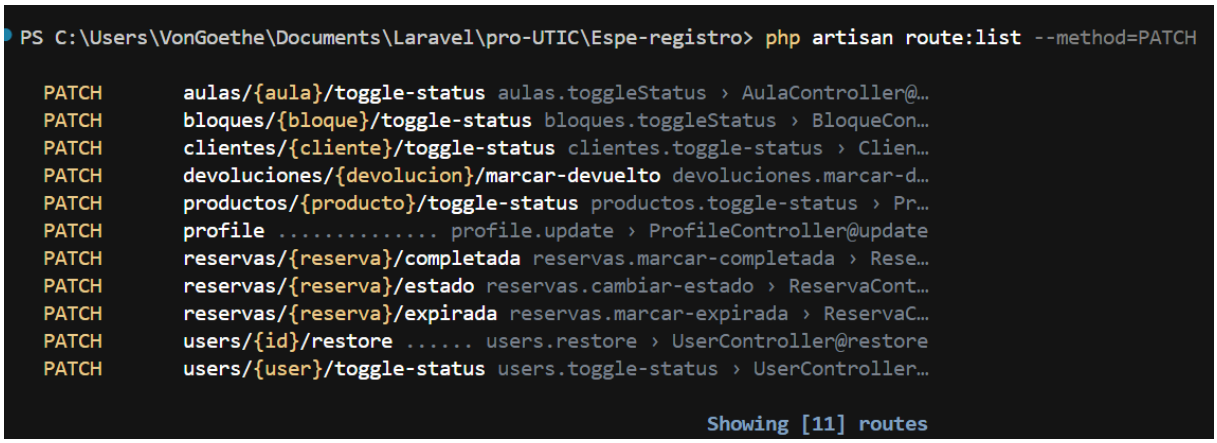

Figura 2. Listado de rutas PATCH de la aplicación de reservas, utilizado para la actualización de estados y gestión dinámica de recursos.

Anexo 3 – Figura 3

Procedimiento para la creación y gestión de productos en el inventario

A continuación se detalla el procedimiento utilizado para crear y gestionar productos dentro del sistema de reservas, incluyendo rutas, acciones y buenas prácticas. Esta documentación se adjunta como referencia visual y técnica para el correcto uso del módulo de inventario:

Tabla 4: Gestión de productos (inventario)

Acción	Ruta / Ubicación	Descripción	Observaciones / Buenas prácticas
Crear producto	productos/create	Permite registrar un nuevo producto en el inventario ingresando su nombre, descripción, stock disponible y estado (activo/inactivo).	Mantener descripciones claras y precisas; indicar unidades de medida; usar “inactivo” para productos sin stock o fuera de uso.
Listar productos	productos	Muestra todos los productos	Permite verificar el estado y stock

		registrados en el sistema.	de cada producto.
Ver/Editar producto	productos/{producto}, productos/{producto}/edit	Permite consultar la información de un producto específico o modificar sus datos.	Revisar consistencia de los datos antes de guardar cambios.
Activar/Desactivar producto	PATCH productos/{producto}/toggle-status	Cambia el estado de un producto entre activo e inactivo según disponibilidad o uso.	Útil para mantener actualizado el inventario sin eliminar registros.

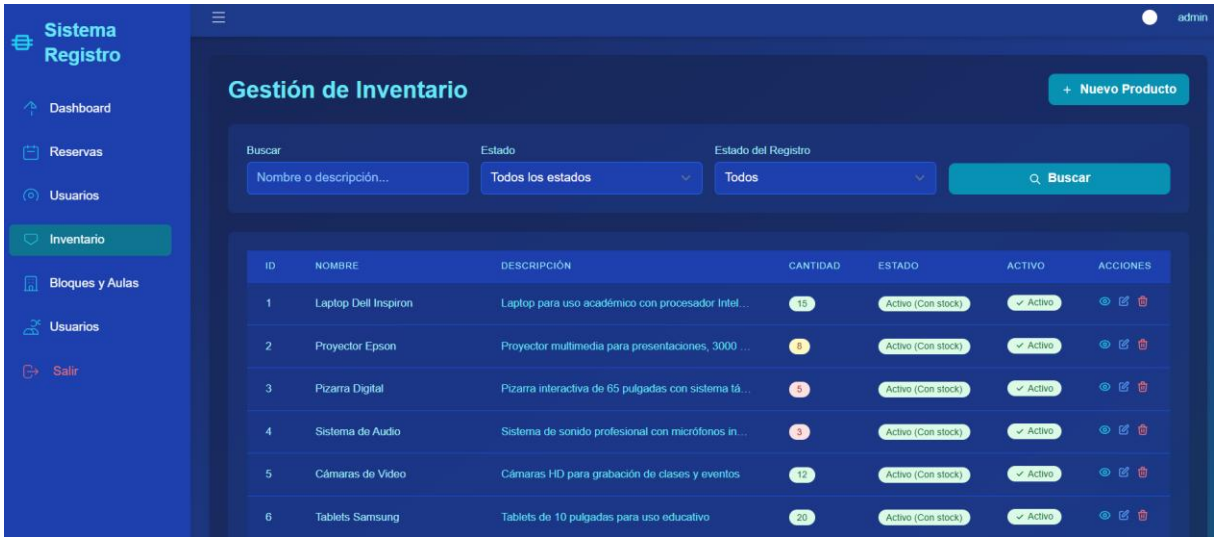

Figura 3. Interfaz y rutas utilizadas para la creación y gestión de productos en el inventario, incluyendo acciones de activación/desactivación y edición de información.

Anexo 4 – Figura 4

Procedimiento para la creación y gestión de bloques (franjas horarias)

El siguiente anexo documenta el procedimiento para crear y administrar bloques de tiempo dentro del sistema de reservas, incluyendo rutas, acciones rápidas y recomendaciones de buenas prácticas para la gestión de horarios:

Tabla 5: Bloques

Acción	Ruta / Ubicación	Descripción	Observaciones / Buenas prácticas
Crear bloque	bloques/create	Permite registrar un nuevo bloque de	Validar que los bloques no se

		tiempo especificando nombre o código, hora de inicio, hora de fin y estado (activo/inactivo).	solapen con otros existentes. Mantener consistencia en la nomenclatura.
Listar/Editar bloque	bloques, bloques/{bloque}/edit	Permite visualizar todos los bloques registrados y modificar los datos de bloques específicos.	Revisar horarios antes de guardar cambios; asegurar que los bloques sean coherentes con la planificación académica.
Activar/Desactivar bloque	PATCH bloques/{bloque}/toggle-status	Cambia el estado del bloque entre activo e inactivo según su uso actual.	Mantener activos solo los bloques realmente en uso; estandarizar horarios (p. ej., 07:00–09:00, 09:00–11:00) para facilitar la planificación.

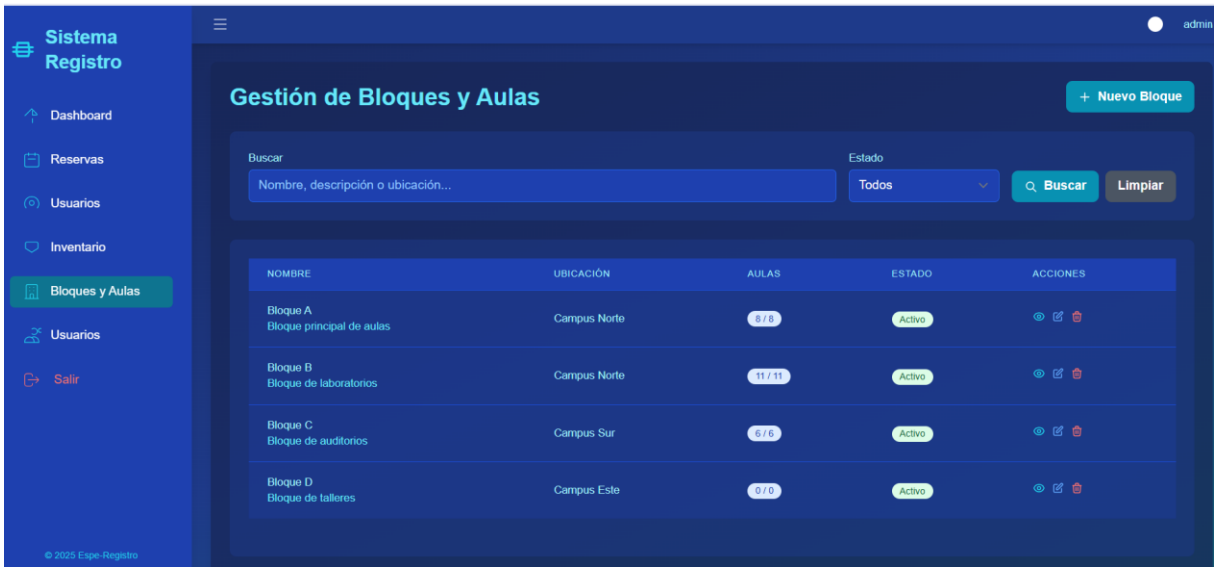

Figura 6. Interfaz y rutas utilizadas para la creación y gestión de aulas, incluyendo acciones de activación/desactivación y consultas por bloque.

Anexo 5 – Figura 5

Procedimiento para la creación y gestión de aulas

El siguiente anexo documenta el procedimiento para agregar y administrar aulas dentro del sistema de reservas, incluyendo rutas, acciones rápidas, consultas y recomendaciones de buenas prácticas:

Tabla 6: Agregar aulas a los bloques

Acción	Ruta / Ubicación	Descripción	Observaciones / Buenas prácticas
Agregar aula	aulas/create	Permite registrar un nuevo aula ingresando su nombre, capacidad, ubicación y estado (activo/inactivo).	Confirmar que el aula se haya guardado correctamente en el listado. Usar nombres únicos y cortos (p. ej., A-101, Lab-Redes).
Listar/Editar aula	aulas, aulas/{aula}/edit	Permite visualizar todas las aulas registradas y modificar datos específicos de cada aula.	Mantener la capacidad actualizada para evitar sobrecupos y garantizar una correcta asignación de espacios.
Activar/Desactivar aula	PATCH aulas/{aula}/toggle-status	Cambia el estado de un aula entre activo e inactivo según su disponibilidad.	Mantener inactivas las aulas que no estén en uso para reflejar la disponibilidad real.
Consultas por bloque	api/aulas/by-bloque	Permite obtener aulas disponibles filtradas por bloques de tiempo específicos mediante la API.	Útil para integraciones con sistemas de reservas y planificación automática.

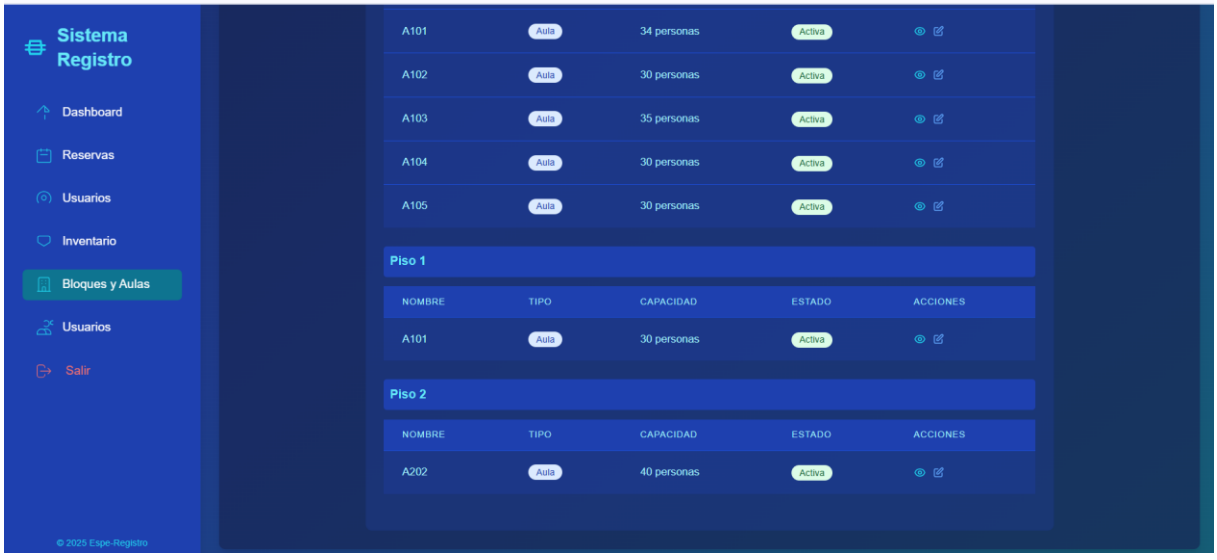

Figura 5. Interfaz y rutas utilizadas para la creación y gestión de aulas, incluyendo acciones de activación/desactivación y consultas por bloque.

Anexo 6 – Figura 6

Procedimiento para generar y gestionar reservas

El siguiente anexo documenta el procedimiento para la creación y seguimiento de reservas dentro del sistema, incluyendo rutas, opciones de interfaz, pasos y buenas prácticas:

Tabla 7: Gestión de Reservas

Acción	Ruta / Ubicación	Descripción	Observaciones / Buenas prácticas
Crear reserva vía formulario	reservas/create	Permite registrar una reserva directamente completando los campos solicitados, como fecha, bloque, aula, productos y observaciones.	Usar información precisa sobre la disponibilidad de aulas y productos; revisar conflictos de horario antes de enviar.
Crear reserva vía calendario	reservas/calendario	Permite seleccionar visualmente la fecha y el bloque horario, facilitando la planificación de reservas.	Garantizar que los bloques y aulas estén previamente configurados y activos.
Consultar mis reservas	mis-reservas	Permite al usuario revisar las reservas creadas por él, incluyendo estado y detalles de cada una.	Mantener actualizado el seguimiento de reservas propias para control personal.
Reservas pendientes de aprobación	reservas/pendientes-aprobacion	Permite al administrador o encargado gestionar las reservas que requieren autorización.	Facilita la revisión rápida de solicitudes y evita conflictos de disponibilidad.
Ver detalle/historial de reservas	reservas/{reserva}, reservas/{reserva}/historial	Permite consultar información detallada de una reserva específica, así como su historial de modificaciones y cambios de estado.	Útil para auditorías internas y seguimiento de cambios en la reserva.

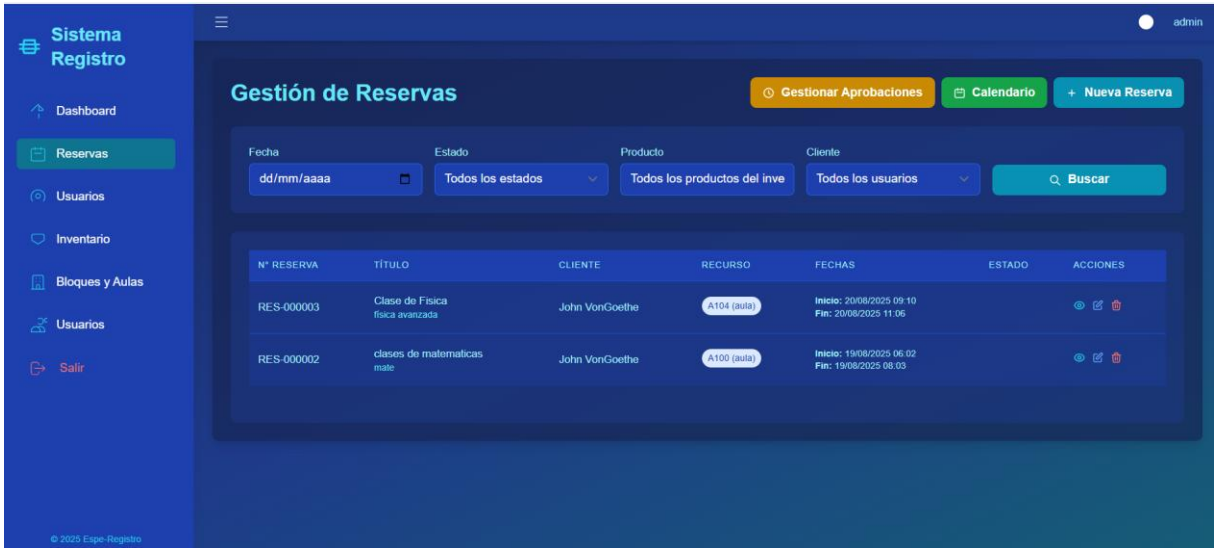

Figura 6. Interfaz y rutas utilizadas para la creación, seguimiento y gestión de reservas en el sistema, mostrando opciones de formulario y calendario, así como consultas de historial y estado.

Anexo 6 – Figura 7

Vista principal del calendario de reservas

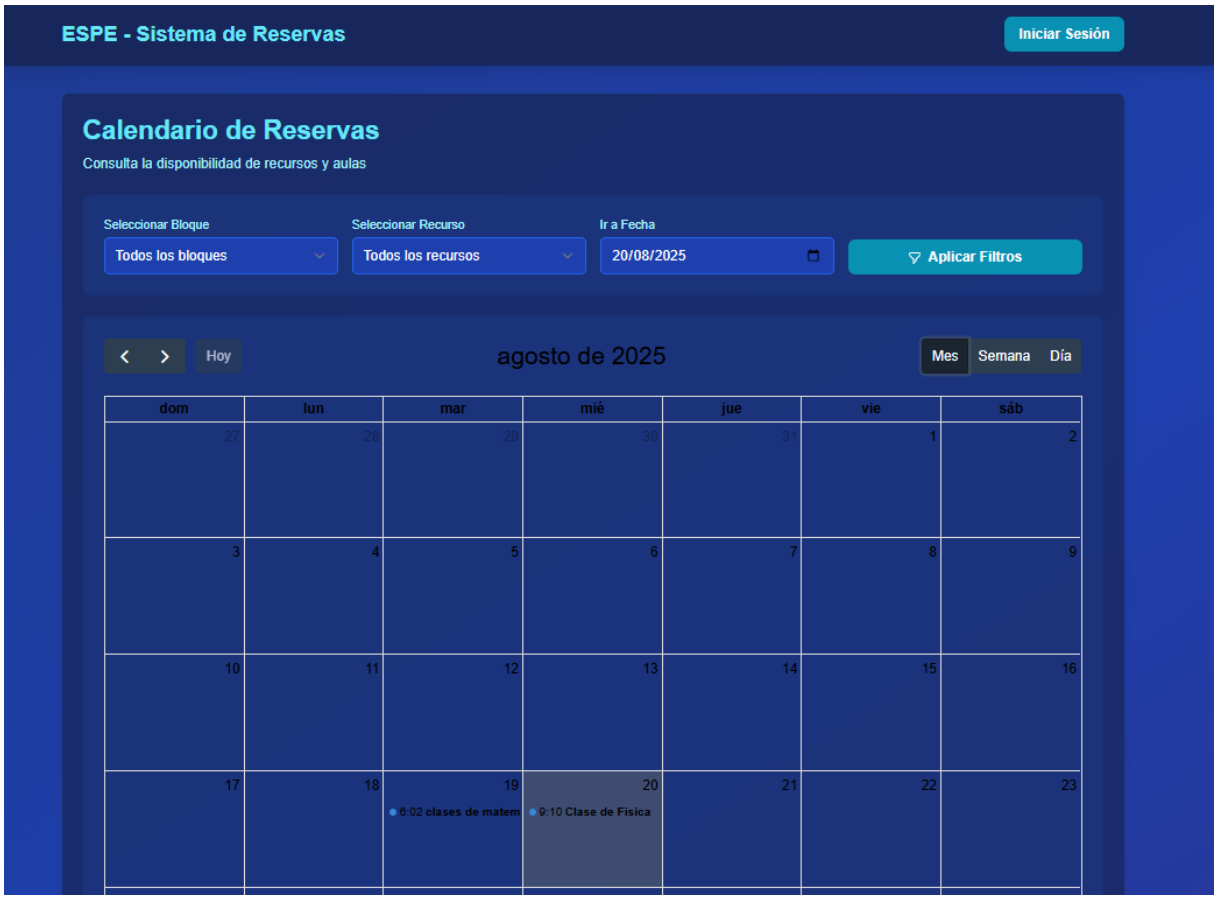

Figura 7. Vista principal del sistema de reservas "ESPE-Registro"desarrollo de software en la Universidad de las Fuerzas Armadas – ESPE.

.