# **Requisitos Arquitectónicos del Sistema InnoSistemas**

## **1. Introducción**

Los requisitos arquitectónicos definen las propiedades fundamentales que debe cumplir la estructura interna del sistema InnoSistemas para garantizar su seguridad, rendimiento, escalabilidad, facilidad de mantenimiento y calidad general. Estas condiciones son esenciales para que el sistema cumpla con sus objetivos funcionales dentro del contexto educativo en el que será implementado.

## **2. Requisitos Arquitectónicos**

### **2.1 Principales Requisitos No Funcionales**

A continuación, se detallan los principales requisitos no funcionales que deben ser considerados para la arquitectura del sistema InnoSistemas. Estos requisitos buscan asegurar que la plataforma sea segura, escalable, eficiente y adecuada al contexto institucional:

#### **2.1.1 Seguridad**

* **RA-01**: El sistema debe restringir el acceso únicamente a usuarios con direcciones de correo electrónico institucionales que terminen en @udea.edu.co. La autenticación se realizará mediante un sistema propio (por ejemplo, login con JWT), validando que el campo de email contenga el dominio autorizado.
* **RA-02**: Toda la información sensible, como contraseñas, debe almacenarse utilizando algoritmos de cifrado seguros (AES-256 o HTTPS/TLS 1.3).
* **RA-03**: Debe existir un sistema de autorización basado en roles, que restrinja el acceso a funcionalidades según el perfil del usuario (estudiante, administrador, docente).
* **RA-04**: Se debe implementar un sistema de registro de eventos de seguridad (LOG), que almacene información sobre intentos de inicio de sesión fallidos y otras acciones relevantes para auditoría.
* **RA-05:** El sistema debe mitigar riesgos de seguridad mediante validación de entradas, uso de tokens CSRF, y protección contra ataques como XSS e inyecciones SQL.

#### **2.1.2 Escalabilidad y Desempeño**

* **RA-06**: La arquitectura debe ser compatible con un entorno escalable, permitiendo el despliegue en contenedores o en infraestructura cloud si fuera necesario.
* **RA-07**: Las operaciones críticas (inicio de sesión, consultas de equipo, reportes) deben ejecutarse en un tiempo de respuesta menor a un 1 segundo bajo condiciones normales.
* **RA-08**: La base de datos debe diseñarse de forma normalizada, permitiendo el crecimiento en número de usuarios, equipos y cursos sin afectar significativamente el rendimiento.
* **RA-09:** Debe existir un sistema de colas para gestionar tareas en segundo plano, como el envío de notificaciones y generación de reportes.

#### **2.1.3 Modularidad y Mantenibilidad**

* **RA-10**: El sistema debe dividirse en módulos funcionales independientes (usuarios, equipos, cursos, seguridad), para facilitar el mantenimiento, pruebas y futuras ampliaciones.
* **RA-11**: Se debe aplicar un patrón arquitectónico estructurado, preferiblemente el patrón Modelo-Vista-Controlador (MVC), o en su defecto una arquitectura basada en servicios o microservicios si el alcance crece.

#### **2.1.4 Interoperabilidad y Extensibilidad**

* **RA-12**: El sistema debe exponer interfaces API RESTful que permitan su integración con otros sistemas académicos o plataformas institucionales.
* **RA-13**: La arquitectura debe ser extensible, permitiendo incorporar nuevas funcionalidades como seguimiento de tareas, revisión por pares, o métricas de desempeño sin reestructurar el sistema base.

#### **2.1.5 Usabilidad y Disponibilidad**

* **RA-14**: La interfaz de usuario debe ser responsiva, accesible desde distintos dispositivos (computadores, tablets, celulares), y cumplir con principios de diseño centrado en el usuario.
* **RA-15**: El sistema debe estar disponible al menos el 99% del tiempo durante horarios académicos, considerando alertas y mecanismos de recuperación ante fallos.
* **RA-16**: El sistema debe manejar errores de forma segura, sin exponer información sensible ni comprometer la integridad del sistema ante fallas operativas.

#### **2.1.6 Documentación y Soporte Educativo**

* **RA-17**: La arquitectura y el código fuente deben estar documentados adecuadamente para facilitar su estudio, uso y adaptación por parte de los estudiantes y docentes involucrados en los cursos de Ingeniería de Sistemas.

### **2.2 Restricciones Técnicas**

La arquitectura del sistema InnoSistemas estará sujeta a las siguientes restricciones tecnológicas, definidas en función de criterios institucionales, de escalabilidad y de mantenibilidad a largo plazo:

* **Lenguaje de programación:** El backend del sistema será desarrollado en **Java**, por su robustez, amplio ecosistema y compatibilidad con arquitecturas empresariales.
* **Framework backend:** Se empleará **Spring Boot** como framework principal para el desarrollo del backend. Este proporciona una estructura modular, inyección de dependencias, manejo de seguridad, integración con bases de datos y facilidad para construir APIs RESTful.
* **Base de datos:** Se utilizará **PostgreSQL** como sistema gestor de bases de datos relacional, por su potencia, compatibilidad con herramientas modernas y cumplimiento con estándares SQL.
* **Despliegue e infraestructura:** El sistema será empaquetado en contenedores **Docker** y gestionado mediante **Kubernetes**, lo que permitirá el escalado automático, monitoreo, balanceo de carga y alta disponibilidad.
* **Framework frontend:** El frontend se construirá utilizando el stack compuesto por:  
  + **React** para la construcción de interfaces dinámicas,
  + **Next.js** para el manejo del enrutamiento y renderizado híbrido (SSR/CSR),
  + **Tailwind CSS** para la definición rápida y consistente del diseño visual,
  + **TypeScript** para la verificación estática de tipos, mejorando la mantenibilidad del código.
* **Compatibilidad con navegadores:** La plataforma debe ser totalmente funcional en las dos versiones más recientes de los navegadores Chrome, Firefox, Edge y Safari.
* **Estándares de integración:** Las APIs expuestas seguirán principios REST, documentadas mediante herramientas como Swagger u OpenAPI.

## **3. Conclusión**

La arquitectura del sistema InnoSistemas debe garantizar no solo el cumplimiento de los requerimientos funcionales, sino también mantener un estándar de calidad y sostenibilidad en su diseño técnico. El cumplimiento de estos requisitos permitirá que la plataforma sea segura, escalable, accesible y útil en el contexto académico para el cual ha sido concebida.