**Informe Transformación Digital: EduTech Innovators SPA.**

**Martín Cofré - Dantte García - Joaquín Molina**

***DESARROLLO FULLSTACK I\_001D***

**Definición del Caso**

EduTech Innovators SPA es una empresa emergente chilena especializada en plataformas educativas en línea. Su rápido crecimiento y expansión a distintas regiones han evidenciado las limitaciones de su sistema monolítico actual, el cual presenta problemas de rendimiento y disponibilidad. Esto afecta la operatividad diaria, la satisfacción del cliente y la continuidad del negocio. Para resolver esta problemática, se propone la transición hacia una arquitectura de microservicios con MySQL como motor de base de datos, lo que permitirá una mejor escalabilidad, modularidad y disponibilidad del sistema.

**Estrategias de Microservicios a Utilizar**

Cada microservicio representará una funcionalidad clave del negocio, como gestión de usuarios, cursos, evaluaciones y soporte. Los servicios se comunicarán mediante eventos asíncronos, reduciendo la dependencia directa entre ellos. Se implementará un API Gateway para gestionar todas las solicitudes desde el frontend hacia los microservicios, mejorando la seguridad y la eficiencia.Cada microservicio tendrá su propia base de datos, utilizando MySQL y asegurando independencia entre ellos.

**1. Análisis de Requerimientos**

**Requerimientos Funcionales:**

* Gestionar usuarios (creación, actualización, desactivación y eliminación de cuentas).
* Configurar permisos de acceso a los módulos del sistema.
* Gestionar cursos (creación, edición, eliminación, inscripciones y evaluaciones).
* Brindar soporte técnico (gestión de incidencias y optimización de recursos).
* Interacción con estudiantes (foro, chat, calificaciones, y progreso en cursos).

**Requerimientos No Funcionales:**

* Alta disponibilidad y escalabilidad del sistema.
* Seguridad en la autenticación y protección de datos.
* Bajo tiempo de respuesta y eficiencia en consultas.
* Capacidad de integración con servicios externos.

**Entrevistas a Perfiles de Usuarios**

Administrador del Sistema:

¿Qué es lo que se requiere para administrar el sistema? *Herramientas para gestión de usuarios y permisos.*

Gerente de Cursos:

¿Cuáles son las funcionalidades esenciales para administrar los cursos? *Un panel para gestionar la creación, edición y eliminación de cursos, así como reportes de inscripciones y rendimiento.*

Logística de Soporte:

¿Qué herramientas se necesitan para optimizar el soporte técnico? *Un sistema de tickets con prioridad y asignación automática, integración con chat en vivo y un panel de estadísticas de incidencias.*

Cliente:

¿Qué aspectos mejorarían tu experiencia en la plataforma? *Una interfaz intuitiva, tiempos de carga más rápidos y notificaciones en tiempo real sobre cambios en cursos y evaluaciones.*

**2. Análisis del Sistema Actual**

Actualmente, EduTech Innovators SPA opera con un sistema monolítico que centraliza todas sus funcionalidades en una sola aplicación. Este sistema presenta problemas de:

* Escalabilidad*:* No soporta el crecimiento de usuarios y cursos sin afectar el rendimiento.
* Disponibilidad*:* Fallas en una parte del sistema afectan a toda la plataforma.
* Mantenimiento*:* Los cambios y actualizaciones requieren modificar toda la aplicación.

**Principales Causas de Fallos y Sobrecargas:**

1. Carga Excesiva: Un solo servidor maneja todas las solicitudes, causando demoras.
2. Dependencias Internas: Un error en un módulo puede inhabilitar todo el sistema.
3. Dificultad para escalar: No se pueden agregar más recursos específicos sin afectar la estructura general.

**3. Diseño de la Nueva Arquitectura**

La arquitectura propuesta sigue el enfoque de microservicios, donde cada módulo del sistema representa un servicio independiente que se comunica con los demás a través de APIs REST y, en algunos casos, mediante mensajes asíncronos.

El sistema contará con los siguientes componentes principales:

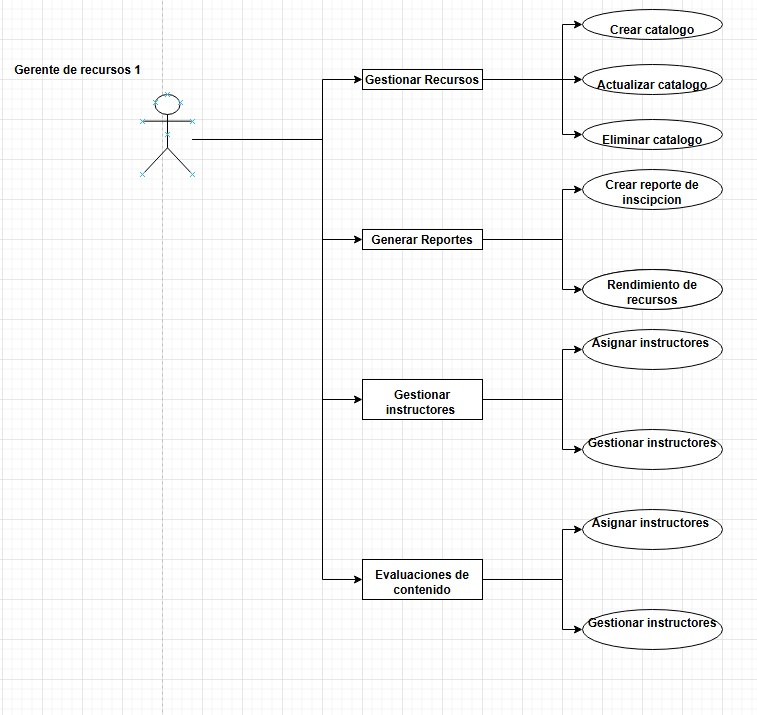
* Servicios principales: Gestión de usuarios, gestión de cursos, gestión de pagos, gestión de notificaciones y gestión de contenidos.
* API Gateway: Encargado de recibir todas las solicitudes y distribuirlas a los microservicios correspondientes.
* Base de datos: Se utilizará un modelo de base de datos centralizada con PostgreSQL, con la opción de segmentar datos en el futuro.
* Comunicación entre servicios: Uso de APIs REST para la mayoría de las interacciones y WebSockets para notificaciones en tiempo real.
* Despliegue y monitoreo: Uso de Docker para la administración de servicios y herramientas básicas de monitoreo.

**Componentes y Tecnologías**

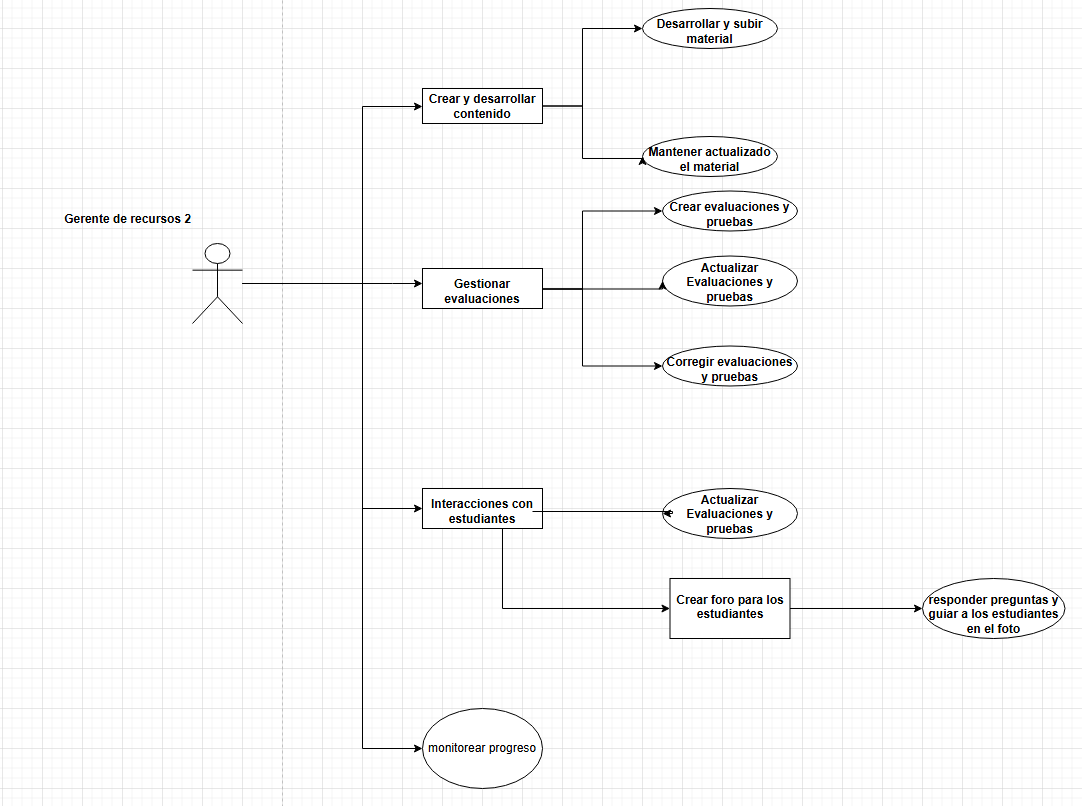
* Microservicios Spring Boot: Cada servicio será desarrollado como una aplicación independiente.
* API Gateway Spring Cloud Gateway: Actúa como un proxy inverso que centraliza las peticiones de los clientes.
* Descubrimiento de Servicios Configuración estática o Nginx: Se define una configuración manual de rutas o se usa Nginx para dirigir las solicitudes.
* Base de Datos PostgreSQL: Almacena los datos de los distintos servicios.
* Mensajería Asíncrona WebSockets (o RabbitMQ): Permite notificaciones en tiempo real y comunicación entre servicios.
* Autenticación JWT (JSON Web Tokens): Seguridad y autenticación de usuarios en cada solicitud.
* Despliegue Docker + Docker Compose: Facilita la administración y ejecución de los servicios en contenedores.
* Monitoreo Logs en archivos + Kibana (opcional): Registro de eventos y visualización de logs para diagnóstico.

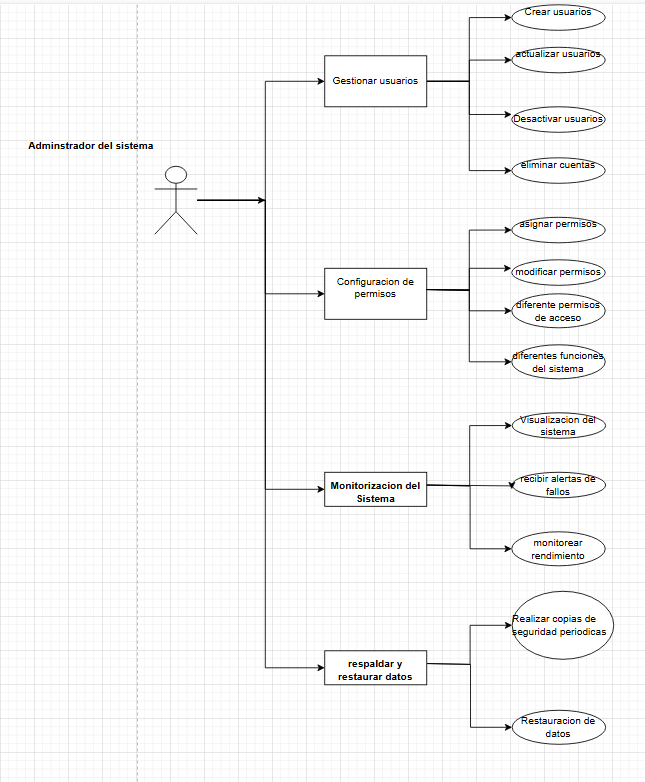
**Diagramas de Arquitectura**

**Gerente de Cursos 1:**

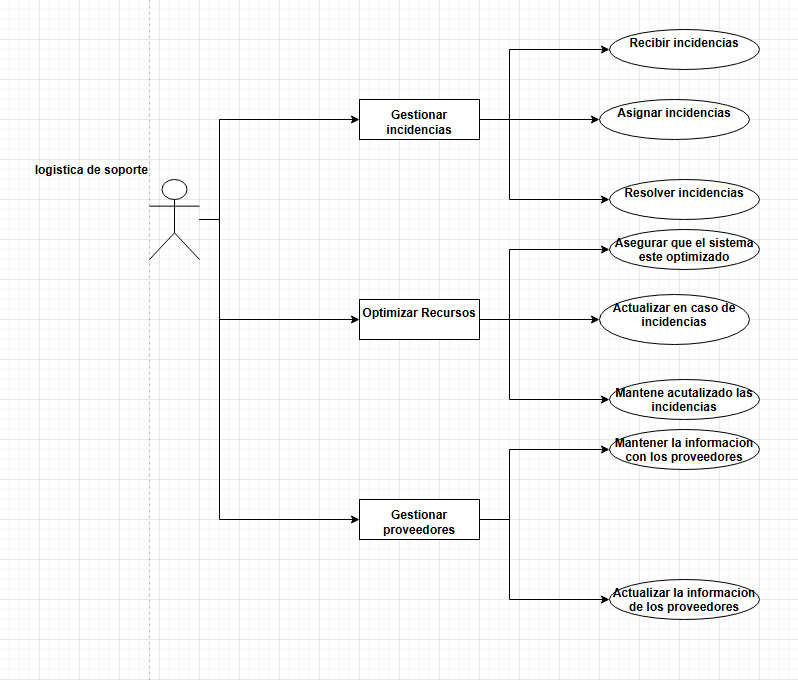
****

**Gerente de Cursos 2:**

****

**Administración del Sistema**

**Logística de Soporte**

****

**4. Planificación de la Migración a Microservicios**

### **Plan de Migración del Sistema Monolítico**

La transformación del sistema monolítico de EduTech Innovators SPA a una arquitectura basada en microservicios debe seguir un proceso estructurado para minimizar riesgos y garantizar la continuidad del negocio. A continuación, se presenta el plan de migración:

#### **1. Análisis del Sistema Actual**

* Evaluación del rendimiento del sistema monolítico actual.
* Identificación de los módulos más críticos y con mayor carga de trabajo.
* Definición de los puntos de integración con la nueva arquitectura.

#### **2. Diseño de la Nueva Arquitectura**

* Definir los microservicios basados en la funcionalidad del sistema actual.
* Establecer un modelo de comunicación eficiente entre microservicios (REST, gRPC, mensajería asíncrona).
* Seleccionar herramientas y tecnologías adecuadas, incluyendo MySQL como motor de base de datos.

#### **3. Estrategia de Implementación**

* Descomposición del sistema monolítico en fases.
* Despliegue progresivo de los microservicios con pruebas unitarias y de integración.
* Monitoreo y ajustes en tiempo real durante la transición.

### **Identificación de Riesgos y Plan de Mitigación**

Algunos de los riesgos asociados con la migración incluyen:

* **Riesgo de interrupción del servicio** → Mitigación: Implementación de un entorno de pruebas robusto y despliegue progresivo.
* **Fallas de comunicación entre microservicios** → Mitigación: Uso de herramientas de monitoreo y logging para detectar errores tempranamente.
* **Problemas de seguridad** → Mitigación: Implementación de autenticación y cifrado de datos.

### **Entregables**

1. **Documento de requisitos del sistema**: Definir las necesidades funcionales y técnicas.
2. **Análisis del sistema actual**: Evaluación de la arquitectura monolítica existente.
3. **Diseño detallado de la nueva arquitectura**: Plan de los microservicios y sus interacciones.
4. **Plan de migración**: Estrategia paso a paso para realizar la transición sin afectar la operatividad del sistema.

**Utilización de Herramientas de Trabajo Colectivo en la Planificación de Microservicios**

### **Herramientas Utilizadas**

A continuación, se presentan algunas herramientas clave para la colaboración y organización en proyectos de arquitectura de microservicios:

#### **1. Trello**

* Permite la gestión de tareas mediante tableros organizados por columnas To-Do, In Progress, Done.
* Facilita la asignación de tareas a los miembros del equipo.
* Ofrece integraciones con herramientas de desarrollo como GitHub y Slack.

#### **2. Jira**

* Especialmente diseñado para metodologías ágiles, como Scrum y Kanban.
* Permite la creación y seguimiento de historias de usuario y sprints.
* Ofrece generación de reportes sobre el rendimiento del equipo y el avance del proyecto.

#### **3. Slack**

* Plataforma de comunicación en tiempo real.
* Integraciones con otras herramientas como Trello y Jira.
* Facilita la creación de canales específicos para equipos de desarrollo, QA y despliegue.

### **Aplicación en el Proyecto**

Para la gestión de la migración de EduTech Innovators SPA a una arquitectura de microservicios, se propone el siguiente flujo de trabajo:

1. **Planificación en Jira**: Creación de historias de usuario y sprints.
2. **Organización de tareas en Trello**: Seguimiento del desarrollo de microservicios.
3. **Uso de Trello**: Creación de diagramas de arquitectura y flujos de interacción entre servicios.
4. **Comunicación en Slack**: Resolución de dudas y coordinación en tiempo real.

## **Evaluación de Enfoques Éticos en la Implementación de Microservicios**

### **Privacidad de Datos**

Uno de los principales desafíos en una arquitectura de microservicios es garantizar la privacidad de los datos de los usuarios. Para ello, se deben implementar:

* **Cifrado de datos en reposo y en tránsito**.
* **Autenticación y autorización segura** mediante protocolos como OAuth 2.0.
* **Minimización de datos**: almacenar sólo la información estrictamente necesaria.

El uso de herramientas de trabajo colectivo es fundamental para garantizar una buena organización y colaboración en la implementación de arquitecturas de microservicios. Asimismo, se deben considerar los principios éticos en el diseño y despliegue de estas arquitecturas para asegurar la privacidad de datos, la responsabilidad en su implementación y minimizar el impacto en el empleo. Aplicando estos principios, EduTech Innovators SPA podrá realizar una transición ética y eficiente hacia una arquitectura de microservicios.

## 