ACTA DE CONSTITUCIÓN DE PROYECTO

Proyecto: Transformación Digital de EduTech Innovators SPA

Fecha de inicio: Abril 2025

Fecha estimada de término: Julio 2025

Cliente: EduTech Innovators SPA

Docente: Viviana Poblete Lopez

Equipo: Rodrigo Abraham Toro Gonzalez

# ÍNDICE DE CONTENIDOS

1. 1. Información del proyecto
2. 2. Datos generales
3. 2. Patrocinadores y responsables
4. 2. Lista de interesados
5. 3. Descripción del proyecto
6. 3. Objetivos de negocio
7. 4. Justificación del proyecto – Contexto
8. 4. Problema y necesidad
9. 5. Requerimientos de alto nivel del proyecto
10. 5. Análisis de propuesta inicial
11. 5. Redefinición de entrada
12. 6. Objetivo del proyecto
13. 7. Alcances
14. 7. Objetivos del desarrollo
15. 7,8. Descripción de la solución
16. 9. Alcances del producto – Premisas y restricciones
17. 10. Especificaciones técnicas de las herramientas
18. 10. Infraestructura y sistemas de implementación
19. 11. Organización del equipo
20. 12. Hitos principales
21. 13. Requisitos de aprobación
22. 13. Control de cambios
23. 14. Análisis de tiempo y costos

# 1. Información del proyecto

Empresa / Organización: EduTech Innovators SPA  
Nombre del Proyecto: Transformación Digital y Migración a Microservicios  
Cliente: Gerencia Técnica y Gerencia de Plataforma Digital – EduTech  
Jefe de Proyecto: Rodrigo Toro

# 2. Patrocinadores y responsables

| Nombre | Cargo | Área |
| --- | --- | --- |
| María Rivas | CEO | Dirección General |
| Cristian Toledo | Gerente de TI | Departamento Técnico |
| Pablo Zamora | Product Owner (Cliente) | Plataforma Académica |

# 3. Lista de interesados (Stakeholders)

| Nombre | Tipo | Cargo | Área |  
|----------------- - --|------------ |--------------------------- |-------------------------------------|  
| María Rivas | Cliente | CEO | Dirección General |  
| Cristian Toledo | Cliente | Gerente de TI | Infraestructura Digital |  
| Gerente de Cursos | Interno | Coordinador Académico | Producción de Contenido |  
| Logística Soporte | Interno | Encargado de Incidencias | Soporte Técnico |  
| Estudiantes | Externo | Usuario final | Web |

# 4. Descripción del proyecto

El proyecto de transformación digital de EduTech Innovators SPA consiste en una reestructuración integral de su plataforma tecnológica, migrando de una arquitectura monolítica a una arquitectura de microservicios. Esta decisión estratégica se fundamenta en la necesidad de mejorar el rendimiento, la escalabilidad, la disponibilidad y la mantenibilidad del sistema educativo en línea que ofrece la empresa. El rediseño busca implementar una solución moderna, modular, distribuida y alineada considerando las buenas prácticas del desarrollo de software actual. Este cambio permitirá que cada módulo funcione como un servicio independiente, reduciendo la dependencia entre componentes y facilitando su actualización, despliegue y monitoreo.

# 5. Objetivos de negocio

* **Aumentar la disponibilidad:** Asegurar un tiempo de actividad superior al 99,9%, permitiendo el acceso continuo a estudiantes, docentes y personal administrativo.
* **Mejorar la experiencia del usuario:** Minimizar los tiempos de espera y garantizar una navegación fluida, adaptada a Plataformas WEB,teniendo a su vez en cuenta responsive para los usos que se puedan dar en algún dispositivo móvil.
* **Reducir costos operacionales:** Disminuir el esfuerzo técnico requerido para mantener y actualizar el sistema, reduciendo los tiempos de soporte y parches correctivos.
* **Elevar la eficiencia modular:** Permitir el desarrollo simultáneo de múltiples funcionalidades en paralelo, mejorando la productividad del equipo técnico.
* **Proceso de migración**: En el proceso de migración se busca no tener un impacto significativo a nivel de usuario, Y que este proceso sea lo más rápido posible.

# 6. Justificación del proyecto – Contexto

La arquitectura monolítica Actualmente ha generado cuellos de botella ante el crecimiento de usuarios y contenidos. La migración permitirá una estructura modular, escalable y segura, compatible con las exigencias actuales del mercado educativo digital.

La empresa ha crecido aceleradamente en los últimos años, posicionándose como Una startup sólida tanto como en el desarrollo y comercialización de cursos en línea en Chile. La plataforma actual fue construida bajo un enfoque monolítico, adecuado en su etapa inicial, pero que ahora presenta limitaciones importantes frente al aumento masivo de usuarios y contenido. Los procesos son rígidos, difíciles de escalar y complican la evolución funcional del sistema. Las caídas del sistema durante eventos de alta demanda afectan directamente la percepción de calidad del servicio y la retención de clientes. Esta modernización representa un paso fundamental hacia la consolidación tecnológica y operativa de la empresa.

# 7. Problema y necesidad

* Caídas frecuentes del sistema ante alta concurrencia.
* Tiempos de carga elevados, especialmente en búsqueda e inscripción de cursos.
* Dificultad para agregar nuevas funcionalidades sin comprometer el sistema completo.
* Complejidad para implementar pruebas automatizadas y mejoras de seguridad.

Necesidades:

* Implementar una arquitectura flexible y desacoplada.
* Facilitar la integración con servicios externos (pagos, reportes, notificaciones).
* Garantizar la continuidad del negocio con alta disponibilidad.
* Potenciar el desarrollo ágil y seguro del sistema.

# 8. Requerimientos de alto nivel del proyecto

* Diseñar e implementar una arquitectura de microservicios.
* Separar los dominios funcionales: usuarios, cursos, evaluaciones, reportes, soporte, notificaciones.
* Establecer un API Gateway para la orquestación del tráfico.
  + Redirigir correctamente las peticiones del usuario hacia el microservicio correspondiente.
  + Aplicar reglas de seguridad, balanceo de carga, autenticación, etc.
  + Monitorizar el flujo de datos para detectar errores, cuellos de botella o caídas.
* Usar JWT para autenticación y autorización.
* Monitorear el sistema usando CloudWatch
* Implementar pipelines CI/CD para cada servicio.
  + CI: Automatiza pruebas y validaciones cada vez que subes código nuevo
  + CD : automatiza el despliegue de ese código a producción o Staging
* Usar Docker y Kubernetes para contenerización y despliegue distribuido.(Por la poca experiencia se debe utilizar ECS+ Fargate)
* Utilizar una base de datos por microservicio (MySQL).

# 9. Análisis de propuesta inicial

En un inicio, se planteó una migración total y simultánea del sistema, lo que implicaba riesgos operacionales muy altos. También se evaluó utilizar una única tecnología backend para todos los microservicios, lo cual habría limitado la capacidad de seleccionar el stack adecuado para cada dominio. La propuesta se centró luego en:

* Un desarrollo progresivo por fases.
* La elección de tecnologías según complejidad: Spring Boot para servicios complejos y FastAPI (Framework de python ) para servicios rápidos.
* Mantenimiento del sistema actual como respaldo durante la migración(Se puede utilizar con ECS y fargate).

# 10. Redefinición de entrada

La estrategia adoptada redefine la implementación hacia un enfoque incremental. Cada módulo será desarrollado, probado e integrado de forma independiente. Esta metodología permite validar tempranamente el funcionamiento y garantizar la calidad de cada servicio antes de su paso a producción. Se priorizan funcionalidades críticas como autenticación, cursos y evaluaciones. El despliegue será automatizado en un entorno de staging y posteriormente en producción. Se realizan pruebas de carga, seguridad y regresión por cada iteración.

# 11. Objetivo del proyecto

Desarrollar una solución tecnológica moderna, modular y escalable mediante microservicios, que permita resolver los problemas actuales de EduTech Innovators SPA, garantizando una operación estable, eficiente y adaptable a futuro. El sistema permitirá ofrecer una mejor experiencia de usuario, con mayor capacidad de crecimiento y adaptación a nuevas necesidades tecnológicas y educativas.

# 12. Alcances

* Reemplazo progresivo del sistema monolítico por microservicios.
* Desarrollo de seis servicios principales: Usuarios, Cursos, Evaluaciones, Reportes, Soporte y Notificaciones.
* Implementación de API Gateway y herramientas de monitoreo.
* Configuración de entornos de staging y producción en la nube.
  + Con esto vamos a evitar que los cambios lleguen a los usuarios reales, Antes de pasar a produccion.
* Establecimiento de políticas de seguridad, auditoría y trazabilidad.
* Capacitación al personal interno en el uso y administración del nuevo sistema.
* Periodo de prueba de 2 semanas con usuarios reales
* **Presupuesto estimado:** Enfocado en horas de desarrollo, plataformas cloud educativas y licencias open-source.
* **Tiempo de implementación:** 10 a 12 semanas (incluyendo pruebas y validaciones).
* **Infraestructura:** Servidores cloud públicos y privados según fases.
* **Personal:** 1 Scrum Master, 2 Desarrolladores Backend, 1 Frontend, 1 QA, 1 Documentador técnico.

.En el punto de Configuracion y stating, Se realizara de la siguiente manera :

Infraestructura duplicada

Tendrás dos versiones de todo:

Instancia ECS (para contenedores)  
Base de datos (RDS o instancia aislada)  
API Gateway  
Variables de entorno distintas  
CloudWatch Logs separados

# 13. Objetivos del desarrollo

Separar los servicios en componentes autónomos.

Documentar cada funcionalidad y endpoint.

Desplegar servicios con configuración desacoplada.

Usar control de versiones y ramas con integración continua.

Establecer pruebas unitarias, de integración y de carga.

Garantizar la escalabilidad horizontal por microservicio.

# 14. Descripción de la solución

La solución estará compuesta por microservicios especializados y desacoplados, diseñados bajo principios de responsabilidad única y baja dependencia. Cada microservicio será responsable de un dominio del sistema (ej. cursos, usuarios, evaluaciones), se comunicará mediante HTTP REST o colas de eventos y dispondrá de una base de datos propia. Los servicios se desplegarán en contenedores Docker sobre Kubernetes, permitiendo su escalado horizontal. La interfaz gráfica (frontend) estará desacoplada y consumirá APIs mediante un API Gateway. Las pruebas automatizadas y el monitoreo centralizado permitirán una operación estable y de alto rendimiento.

# 15. Alcances del producto – Premisas y restricciones

### Premisas:

* El equipo de desarrollo tiene conocimientos previos en Java, Python y Basico en React.
* La plataforma cloud será provista con AWS Con modalidad escalable
* El sistema será documentado y versionado desde el inicio.

### Restricciones:

* El motor de base de datos será únicamente MySQL.
* Cumplimiento con la Ley N°19.628 sobre protección de datos personales.
* La migración debe realizarse sin afectar el sistema en producción actual.
* Recursos humanos limitados por calendario académico.

# 16. Especificaciones técnicas de las herramientas

- Backend: Java (Spring Boot), Python (FastAPI)

- Frontend: React + Tailwind CSS

- Base de Datos: MySQL 8.x (una instancia separada p or microservicio, gestionada con Amazon RDS)

- Contenedores: Docker (contenedores empaquetados por cada microservicio)

- Orquestación y Despliegue: AWS ECS (Elastic Container Service) + AWS Fargate (sin servidor)

- CI/CD: GitHub Actions (automatiza pruebas y despliegues por rama), integrado con Amazon ECR y ECS

-Monitoreo y Logs: AWS CloudWatch (métricas, alertas y trazabilidad de servicios)

- Repositorio de código: GitHub con control de ramas (main, dev, feature/x) y documentación integrada

- Autenticación: JWT validado desde el API Gateway

- Pasarela de entrada: Amazon API Gateway (control de tráfico, autorización y seguridad)

# 17. Infraestructura y sistemas de implementación

- Se utilizarán servicios gestionados de AWS para garantizar estabilidad, escalabilidad y seguridad.

- Cada microservicio se desplegará como contenedor Docker en AWS Fargate bajo un clúster ECS.

- API Gateway centralizará el acceso a los servicios, aplicando autenticación JWT y control de tráfico.

- Cada servicio tendrá su propia base de datos MySQL en Amazon RDS, aislada y escalable.

- Ambientes separados para desarrollo (staging) y producción (production), configurados con variables de entorno diferenciadas.

- CloudWatch manejará los registros de errores, métricas de uso, rendimiento y trazabilidad por microservicio.

- Backups automáticos configurados por RDS. Seguridad controlada con IAM, WAF y grupos de seguridad.

# 18. Organización del equipo

Scrum Master, Devs Backend/Frontend, QA Tester, Documentación. Roles rotativos según sprint.

# 19. Hitos principales

1. Semana 1: Levantamiento de requerimientos, análisis funcional y planificación.
2. Semana 2-3: Diseño de arquitectura por microservicios y configuración inicial en AWS.
3. Semana 4-7: Desarrollo progresivo de servicios: autenticación, cursos, evaluaciones, notificaciones.
4. Semana 8-9: Pruebas integrales en entornos staging, carga, seguridad y revisión de errores.
5. Semana 10: Despliegue en producción, entrega final de documentación, presentación del proyecto

# 20. Requisitos de aprobación

- Todos los microservicios deben estar funcionales y desplegados en producción (AWS).

- El sistema debe pasar pruebas de carga, integración, autenticación y validación en staging.

- Código documentado en GitHub, con historia de commits, control de cambios y scripts de despliegue.

- Evaluación docente positiva sobre la arquitectura, solución y presentación individual.

# 21. Control de cambios

Todo cambio deberá ser:

- Cada cambio en el proyecto será trackeado mediante commits descriptivos en GitHub.

- Se utilizará un sistema de issues por microservicio y una estrategia de ramas (dev → staging → main).

- Las revisiones se aprobarán mediante pull requests con revisión por parte de otro integrante del equipo.

- La bitácora de cambios se mantendrá dentro del repositorio y en cada entregable formal.

# 22. Análisis de tiempo y costos

Duración total estimada: 10 semanas académicas

Costos estimados:

- Uso de servicios AWS gratuitos (Free Tier o Educate) durante el desarrollo

- Licencias open-source para frameworks y herramientas

- Costos proyectados en base a consumo: instancias Fargate, RDS y tráfico en API Gateway

- Costos principales: horas de desarrollo del equipo, capacitación y configuración inicial

Este análisis redefine la base técnica del proyecto en torno al uso de AWS como proveedor cloud, los precios pueden ir variando según las necesidades, En primera opcion se buscara la capa gratuita pero en la medida que se vayan implementando se dara un costo real incluyendo el descuento que entregarán por asegurando compatibilidad con entornos reales y buenas prácticas de arquitectura moderna.