Proyecto EduTech Innovators Spa.

Equipo: Tiare Acuña

Ricardo Rodríguez

Maximiliano Rubilar

Fecha: 27 de mayo de 2025.

Contenido

[1. Introducción 3](#_Toc199247729)

[2. Diagrama de arquitectura de microservicios 4](#_Toc199247730)

[3. Estructura del proyecto 6](#_Toc199247731)

[3.1 Dependencias 7](#_Toc199247732)

[3.2 Componentes implementados 8](#_Toc199247733)

[4. Base de datos 10](#_Toc199247734)

[5. Implementación de los servicios 12](#_Toc199247735)

[6. Implementación de vistas 15](#_Toc199247736)

[7. Git – Git Hub 16](#_Toc199247737)

[7.1 Comandos utilizados 16](#_Toc199247738)

[7.2 Archivo README.md 17](#_Toc199247739)

[8. Conclusión 18](#_Toc199247740)

# 1. Introducción

Bienvenidos a esta nueva presentación donde compartiremos los avances más recientes de nuestro proyecto **EduTech Innovators Spa**, una empresa comprometida con la transformación digital en el ámbito educativo. Desde su creación, nuestra misión ha sido proporcionar plataformas educativas accesibles, dinámicas e innovadoras, permitiendo que estudiantes de diversas regiones puedan acceder a herramientas de aprendizaje efectivas sin importar su ubicación geográfica.

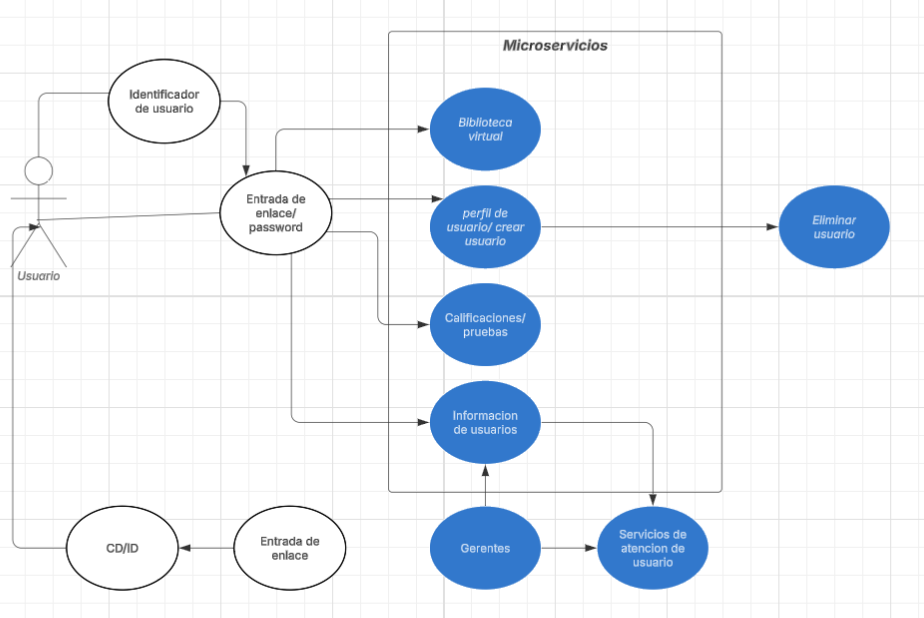
Nuestra historia comenzó con una oficina en **Providencia, Santiago**, donde dimos nuestros primeros pasos en la creación de cursos interactivos en línea. Gracias al respaldo de nuestros usuarios y la creciente demanda de soluciones digitales en educación, hemos logrado expandir nuestras operaciones y establecer nuevas oficinas en **Valparaíso y La Serena**, consolidando así nuestra presencia en el mercado y fortaleciendo nuestra infraestructura para ofrecer un servicio más sólido y escalable.

Como se discutió en nuestra presentación anterior, el sistema **EduTech** enfrenta diversos desafíos relacionados con el rendimiento y la disponibilidad. Identificamos y analizamos estos problemas con detalle, presentando propuestas y soluciones que han permitido optimizar la estabilidad del software y mejorar la experiencia del usuario. En esta sesión, nos enfocaremos en la **estructura del proyecto**, sus **componentes clave**, el **modelo de base de datos** y las **implementaciones de servicios** que han sido fundamentales para garantizar un funcionamiento eficiente y escalable.

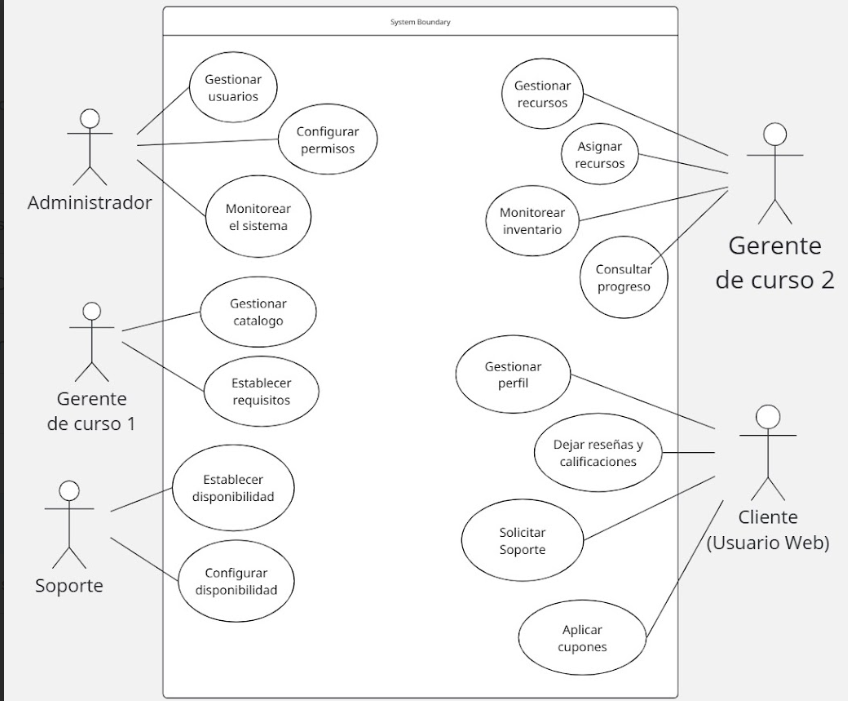
A lo largo de esta presentación, explicaremos en profundidad cómo hemos refinado la arquitectura del software, detallaremos las mejoras realizadas en la integración de nuestros servicios y mostraremos cómo cada uno de estos cambios contribuye a la misión de **EduTech Innovators Spa**: ofrecer educación accesible y de calidad a través de plataformas tecnológicas avanzadas.

Esperamos que esta actualización sea de gran utilidad para todos los presentes y estamos abiertos a cualquier consulta o sugerencia que ayude a seguir mejorando nuestro sistema.

# 2. Diagrama de arquitectura de microservicios



En este diagrama de arquitectura se ve los procesos que sigue cada usuario a la hora de acceder a los servicios los cuales se divide en diferentes secciones basados en los crud (crear, leer actualizar y eliminar) para que así se pueda mantener una base de datos funcional basada en los microservicios que requiere la empresa para su funcionalidad



También se debe agregar el diagrama de los usuarios ya que es un factor determinante a la hora de realizar un programa de desarrollo software ya que es la única forma de saber los requisitos y roles sobre los cuales se basarán para desarrollar una base de datos y programa que se pueda adaptar a las necesidades de los usuarios y personal encargado de la página web.

# 3. Estructura del proyecto

├── usuarios/ ← Microservicio 1: gestión de usuarios

│ ├── src/

│ │ └── main/

│ │ ├── java/com/edutech/usuarios/

│ │ │ ├── controller/ ← UsuarioController.java

│ │ │ ├── model/ ← Usuario.java

│ │ │ ├── repository/ ← UsuarioRepository.java

│ │ │ └── service/ ← UsuarioService.java

│ │ └── resources/

│ │ ├── templates/ ← detalles.html

│ │ └── application.properties

│ └── pom.xml

│

├── cursos/ ← Microservicio 2: gestión de cursos

│ ├── src/

│ │ └── main/

│ │ ├── java/com/edutech/cursos/

│ │ │ ├── controller/ ← CursoController.java

│ │ │ ├── model/ ← Curso.java

│ │ │ ├── repository/ ← CursoRepository.java

│ │ │ └── service/ ← CursoService.java

│ │ └── resources/

│ │ ├── templates/ ← cursos.html

│ │ └── application.properties

│ └── pom.xml

│

├── inscripciones/ ← Microservicio 3: inscripción de usuarios a cursos

│ ├── src/

│ │ └── main/

│ │ ├── java/com/edutech/inscripciones/

│ │ │ ├── controller/ ← InscripcionController.java

│ │ │ ├── model/ ← Inscripcion.java

│ │ │ ├── repository/ ← InscripcionRepository.java

│ │ │ └── service/ ← InscripcionService.java

│ │ └── resources/

│ │ ├── templates/ ← index.html o principal.html

│ │ └── application.properties

│ └── pom.xml

│

├── .gitignore ← Ignora carpetas como target/, .vscode/, etc.

└── README.md ← Documentación del proyecto (opcional)

## 3.1 Dependencias

Las dependencias utilizadas en el proyecto permiten el desarrollo completo de una aplicación basada en Spring Boot y arquitectura de microservicios. Se incluyen librerías esenciales como spring-boot-starter-web para construir APIs REST, spring-boot-starter-data-jpa para persistencia de datos con JPA, y mysql-connector-java como driver para la base de datos. En los servicios que presentan vistas, se añade spring-boot-starter-thymeleaf para renderizar contenido HTML dinámico. Finalmente, spring-boot-devtools se incorpora como una herramienta de desarrollo que permite reinicios automáticos y mayor productividad durante la implementación.

<!-- Spring Boot -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- Spring Data JPA -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<!-- MySQL Connector -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<!-- Thymeleaf (para vistas) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<!-- Devtools (opcional para recarga en caliente) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

## 3.2 Componentes implementados

Se aplicaron muchos componentes en este programa software los cuales cada uno cumplió su propósito. Estos componentes son los siguientes:

* **@Request controller y @RequestMapping**: El @Request controller significa que una clase es un **controlador REST**, lo que significa que manejará solicitudes HTTP y devolverá respuestas en formato JSON o XML. Básicamente, convierte la clase en un punto de acceso para la API. Mientras que el @RequestMapping sirve para definir la base URL que el controlador maneja al especificar los métodos HTTP
* **@PostMapping:** Ayuda a registrar una nueva inscripción, recibiendo los datos en el cuerpo de la solicitud
* **@RequestBody:** Sirve para extraer los datos del cuerpo de una solicitud HTTP y convertirlos en un objeto de java. Bastante útil al recibir datos en formato de JSON o XML y necesitarás procesarlos en la aplicación
* **@DeleteMapping (“/{id}))** : Este componente sirve para borrar inscripciones específicas identificada como id
* **@RestTemplate:** Es para llamar a los servicios externos y así poder obtener datos de usuarios y cursos
* **(model.addAttribute(“inscripciones”, detalles)):** Este componente se usa para poder pasar datos desde un controlador a la vista de la aplicación web
* **@RequestParam:** Sirve para poder obtener parámetros de solicitud de HTTP, generalmente está relacionado con consultar URL o formularios. Útil a la hora capturar valores específicos como los de un cuerpo JSON
* **(LocalDate.now()):** Este componente se usa para crear una nueva inscripción con su fecha actual
* **(inscripcionService.guardar(nueva))** y **(redirect:/):** El primero sirve para guardar las inscripciones usando el servicio y la segunda para redirigir al usuario a la página principal
* **Encapsulación de información:** Es útil a la hora de restringir los datos específicos de un objeto con métodos específicos para de esta forma poder mejorar la seguridad en los códigos
* **@Entity:** Este componente se usa para poder marcar una clase como una entidad de base datos dentro de un JPA(Java Persistence Api) por ende este funciona solo dentro de una API en un programa software
* **@Id:** Sirve para definir un atributo en este caso el de un **id** como clave primaria
* **@GeneratedValue**(**strategy** **= GenerationType.IDENTITY):** Este componente logra que cada base de datos pueda generar de manera automática un id
* **listar():** Este componente sirve para poder obtener todas las inscripciones de los usuarios
* **FindAll():** Sirve para obtener todos los registros de una entidad en la base de datos. Este componente es prácticamente el la operación más común de un Crud(Crear, leer, actualizar, eliminar) y también permite recuperar una lista completa de objetos sin la necesidad de tener que escribir consultas desde la base de datos agilizando el programa software
* **Guardar(inscripcion i):** Este componente sirve para guardar la nueva inscripción de los usuarios en la base de datos mediante un save**(i)**
* **Eliminar(Long id):** Como se ve en el nombre de este componente. Este sirve para eliminar las inscripciones de los usuarios designados usando su ID como registro
* **deleteById(id):** Es el componente o mejor dicho código, el cual se encarga de eliminar al usuario utilizando el Id de este mismo al igual que el código anterior
* **@SpringBootAplication:** Es el componente más usado y más importante ya que este está encargado de configurar, habilitar modificaciones y configuraciones, también permite que el Spring escanee los paquetes y registre los componentes del objeto
* **SpringApplication.run(InscripcionesApplication.class, args);** Este componente es para iniciar la aplicación, creando el contexto de spring y levantando los servicios necesarios para el sistema software
* **Main(String[ ] args):** Es el método principal que se ejecuta cuando la aplicación arranca ya que sin este el programa no compilaria ni sería funcional

# 4. Base de datos

En la implementación de la base de datos logramos implementar mediante visualcode estudios y la base de datos se creó en MySql Workbench. esto es posible gracias a las apis y implementos de instancias en base al modelo de tipo crud (Crear, Leer, Actualizar, Eliminar)

**Motor utilizado:**

* **MySQL 8** con MySQL Workbench

Cada microservicio utiliza su propia base:

* bd\_usuarios con tabla usuario
* bd\_cursos con tabla curso
* bd\_inscripciones con tabla inscripcion

**Estructura de tablas:**

**usuario**

* id (PK)
* nombre
* correo
* contrasena

**curso**

* id (PK)
* titulo
* descripcion
* duracion

**inscripcion**

* id (PK)
* usuario\_id (FK)
* curso\_id (FK)
* fecha

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

# 5. Implementación de los servicios

**Usuarios (localhost:8081/usuarios)**

* GET: listar usuarios
* POST: agregar usuario
* DELETE: eliminar usuario

**Cursos (localhost:8082/cursos)**

* GET: listar cursos
* POST: agregar curso
* DELETE: eliminar curso

**Inscripciones (localhost:8083/inscripciones)**

* GET: listar inscripciones (con datos de usuario y curso)
* POST: inscribir un usuario a un curso

Texto

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

# 6. Implementación de vistas

Se utilizó Thymeleaf para desplegar datos obtenidos desde cada servicio REST:

**/usuarios → tabla HTML de usuarios registrados**

**/cursos → tabla HTML de cursos + formulario agregar curso**

**/ → inscripciones mostradas con nombres y cursos asociados**

Cada vista realiza una petición GET que carga datos desde el Service, conectándose a la base.

Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

# 7. Git – Git Hub

## 7.1 Comandos utilizados

git init

git config --global user.name "Tiare Acuña"

git config --global user.email "tiar.acuna@duocuc.cl"

git add .

git commit -m "Subida inicial del proyecto"

git remote add origin https://github.com/tiaaa/Evaluacion2\_FullStack.git

git push -u origin main

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

## 7.2 Archivo README.md

Texto

El contenido generado por IA puede ser incorrecto.

# 8. Conclusión

En esta presentación hemos recorrido los avances más recientes del proyecto **EduTech Innovators Spa**, destacando cómo hemos evolucionado para responder a los desafíos de rendimiento y disponibilidad que enfrentaba nuestro sistema. A través de un enfoque estratégico, la optimización de la arquitectura del software y la implementación de soluciones innovadoras, hemos logrado fortalecer la plataforma y mejorar la experiencia del usuario.

Nuestra expansión a nuevas regiones y la consolidación de nuestra infraestructura demuestran el compromiso de **EduTech Innovators Spa** con la transformación digital en la educación. Gracias al respaldo de nuestra comunidad y al trabajo constante de nuestro equipo, seguimos perfeccionando cada aspecto del sistema, asegurando que las herramientas que ofrecemos sean accesibles, eficientes y escalables.

A medida que avanzamos, nos enfocamos en seguir mejorando nuestras tecnologías y adaptándonos a las necesidades del entorno educativo en línea. La evolución de nuestro proyecto no termina aquí; continuaremos explorando nuevas oportunidades de crecimiento, innovación y mejora continua.

Agradecemos su atención y participación, y estamos abiertos a sus consultas, sugerencias y aportes para seguir construyendo juntos una plataforma educativa que marque la diferencia.