Universidad Don Bosco

Asignatura: Desarrollo de Aplicaciones con Web Frameworks

Grupo: G02T

Actividad: Proyecto final

Docente: Emerson Francisco Cartagena Candelario

Carnet	Integrantes de trabajo
FN242644	Abner Leví Funes Navarro
LT222760	Cristian Alexis Lopez Tamayo
HM242655	Francisco Josué Hernández Meléndez
GF242647	Jasmín Azucena García Flores

Soyapango, 24 de octubre de 2025

MANUAL TÉCNICO

1. Descripción General

El proyecto Punto Evento API es una aplicación web desarrollada con Spring Boot, diseñada para gestionar información relacionada con clientes, asignaciones y autenticación de usuarios dentro de un entorno empresarial.

2. Arquitectura del Sistema

El sistema utiliza una arquitectura basada en capas (Controller, Service, Repository, Entity y Config). Cada módulo gestiona una parte específica del sistema para mantener una estructura limpia y modular.

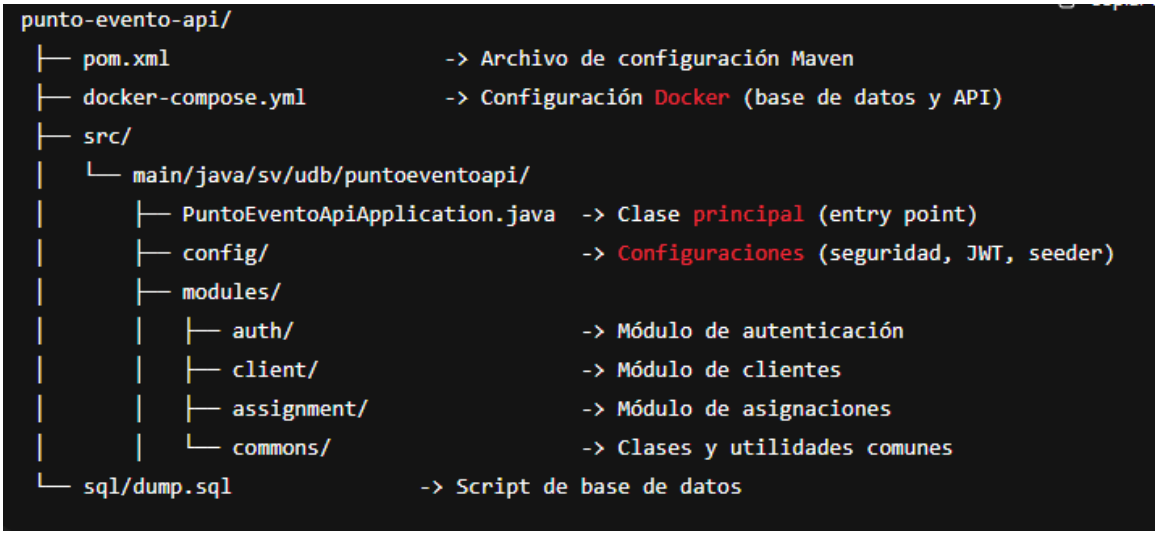
- Capa de Controladores (Controller): Gestiona las solicitudes HTTP y define los endpoints de la API.
- Capa de Servicios (Service): Contiene la lógica de negocio del sistema.
- Capa de Repositorios (Repository): Interactúa con la base de datos utilizando Spring Data JPA.
- Capa de Entidades (Entity): Define las tablas de la base de datos mediante anotaciones JPA.
- Capa de Configuración (Config): Contiene la configuración de seguridad, JWT, CORS, y seeding inicial de datos.

3. Tecnologías Utilizadas

Java 17, Spring Boot, Spring Data JPA, Spring Security con JWT, MySQL, Maven, Docker e IntelliJ IDEA.

4. Estructura del Proyecto

El proyecto se organiza en módulos principales: auth, client, assignment y commons. Cada módulo incluye controladores, servicios, entidades y repositorios.



5. Instalación y Configuración

Requisitos previos

- Java JDK 17
- Maven
- MySQL Server (o Docker)
- IDE (IntelliJ, Eclipse o VS Code)

Pasos de instalación

- Clonar o descomprimir el proyecto:
- Abrir el proyecto en IntelliJ IDEA.
- Verificar que las dependencias se descarguen correctamente con Maven.
- Configurar las credenciales de la base de datos en application.properties:
  spring.datasource.url=jdbc:mysql://localhost:3306/puntoevento
  spring.datasource.username=root
  spring.datasource.password=tu_password
- Iniciar el proyecto y acceder a la API Localhost:8080

6. Seguridad y Autenticación

La seguridad se gestiona con JWT. Las contraseñas se encriptan con BCrypt. Las rutas seguras requieren autenticación.

- El proyecto utiliza JWT (JSON Web Token) para la autenticación de usuarios.
- La clase SecurityConfig.java configura los filtros de seguridad.
- Las rutas públicas y privadas se manejan mediante anotaciones como @PreAuthorize.
- Las contraseñas se encriptan con BCrypt (PasswordEncoderConfig.java).

7. Base de Datos

MySQL es el motor utilizado. Las entidades están mapeadas mediante JPA y el script dump.sql genera la estructura inicial.

- El proyecto utiliza MySQL como motor de base de datos relacional.
- Las entidades están definidas en los paquetes entity/.
- Las relaciones se implementan con anotaciones JPA (@OneToMany, @ManyToOne, etc.).
- El archivo dump.sql contiene las estructuras y datos iniciales.

8. Principales Endpoints

Auth: /api/auth/login, /api/auth/register | Clients: /api/clients | Assignments: /api/assignments

Módulo	Método	Endpoint	Descripción
Auth	POST	/api/auth/register	Registrar nuevo usuario
Auth	POST	/api/auth/login	Iniciar sesión y obtener token JWT
Client	GET	/api/clients	Listar clientes
Assignment	GET	/api/assignments	Consultar asignaciones
Assignment	POST	/api/assignments	Crear nueva asignación

9. Ejecución en Docker

Se puede usar docker compose up -d' para ejecutar la API y la base de datos en contenedores.

10. Mantenimiento y Actualización

Actualizar dependencias con 'mvn clean install' y reiniciar el servicio para aplicar cambios.

- Actualizar dependencias con mvn clean install.
- Revisar logs en target/logs/ o consola.
- Reiniciar el servicio con mvn spring-boot:run tras aplicar cambios.