**Requisitos Técnicos y Funcionales**

Cada equipo debe cumplir con los siguientes requisitos:

1. **Desarrollo de la API RESTful**:
   * **Framework**: Utilicen **Java** junto con el framework **Spring Boot** para la creación de la API.
   * **Estructura de la API**: Asegúrense de implementar las operaciones CRUD (Create, Read, Update, Delete).
   * **Patrón de Diseño MVC**: Estructuren la aplicación siguiendo el **patrón MVC** (Modelo-Vista-Controlador):
     + **Modelo (M)**: Define las clases que representan las entidades de la base de datos.
     + **Vista (V)**: En el caso de una API RESTful, la vista estará representada por las respuestas JSON.
     + **Controlador (C)**: Los controladores manejarán las solicitudes HTTP y llamarán a los servicios y repositorios necesarios.
2. **Operaciones CRUD**:
   * **Create**: Implementen una ruta para crear nuevos registros en la base de datos.
   * **Read**: Incluyan rutas para obtener registros individuales o listas de registros, con la posibilidad de filtrar los resultados según parámetros específicos.
   * **Update**: Definan una ruta para actualizar registros existentes en la base de datos.
   * **Delete**: Implementen una ruta para eliminar registros.
3. **Integración con la Base de Datos**:
   * **Repositorio**: Utilicen la **Interfaz de Repositorio de Spring Data JPA** para gestionar la persistencia de los datos. Esto facilitará las operaciones de CRUD sin necesidad de escribir consultas SQL manualmente.
   * **Configuración de Base de Datos**: Configuren una base de datos en memoria (H2) o una base de datos local (por ejemplo, MySQL o PostgreSQL) y asegúrense de que las tablas se generen automáticamente a partir de las entidades definidas.
4. **Documentación de la API**:
   * Documenten de forma detallada los endpoints de la API, incluyendo:
     + **Ruta (Endpoint)**: Especificar la ruta de cada recurso (por ejemplo, /api/productos, /api/usuarios).
     + **Método HTTP**: Indiquen el método HTTP asociado a cada endpoint (GET, POST, PUT, DELETE).
     + **Parámetros**: Documenten los parámetros requeridos y opcionales de cada endpoint (por ejemplo, parámetros de ruta, de consulta, o del cuerpo de la solicitud).
     + **Códigos de Respuesta**: Especifiquen los códigos de respuesta HTTP que la API devolverá (por ejemplo, 200 OK, 201 Created, 404 Not Found, 500 Internal Server Error).
     + **Ejemplos de Solicitudes y Respuestas**: Incluir ejemplos de solicitudes (cuerpo JSON cuando aplique) y respuestas esperadas.
   * **Herramientas de Documentación**: Se recomienda el uso de herramientas como **Swagger** o **Postman** para facilitar la creación de la documentación.

**Estructura de la Entrega**

El archivo a entregar debe contener los siguientes apartados en orden:

1. **Introducción**:
   * Breve descripción de la API desarrollada, el problema que resuelve y el dominio o sector al que está orientada.
2. **Descripción del Diseño**:
   * **Diagrama de Arquitectura**: Incluyan un diagrama sencillo que muestre la estructura del proyecto siguiendo el patrón MVC, destacando las interacciones entre Controladores, Servicios, y Repositorios.
   * **Modelo de Datos**: Incluir un diagrama entidad-relación o una tabla que describa cada entidad, sus atributos y las relaciones entre ellas.
3. **Documentación de Endpoints**:
   * Lista de todos los endpoints de la API con el siguiente formato para cada uno:
     + **Endpoint**: URL del recurso (ejemplo: /api/usuarios/{id}).
     + **Método HTTP**: (GET, POST, PUT, DELETE).
     + **Descripción**: Explicación breve de la función del endpoint.
     + **Parámetros**:
       - Parámetros de ruta: especificar si es requerido.
       - Parámetros de consulta: especificar si es opcional o requerido.
       - Cuerpo de la solicitud (si aplica): formato y estructura esperada en JSON.
     + **Códigos de Respuesta**: Indiquen los códigos de estado HTTP que puede devolver cada endpoint y su significado.
     + **Ejemplo de Solicitud y Respuesta**: Incluyan ejemplos de la solicitud en formato JSON (para POST y PUT) y la respuesta esperada.
4. **Instrucciones de Configuración y Ejecución**:
   * Explicar los pasos para ejecutar la API en un entorno de desarrollo, incluyendo:
     + **Requisitos previos**: software necesario, como JDK, Maven, y cualquier base de datos adicional.
     + **Pasos para compilar y ejecutar el proyecto**: Comandos necesarios para compilar (mvn clean install o similar) y ejecutar la aplicación (mvn spring-boot:run).
     + **Configuración de la Base de Datos**: Detallar cualquier archivo de configuración (application.properties o application.yml) que deba modificarse.