**SpringBoot-Modelo de trabajo con APIs:**

1. **Entidad (Entity):**
   * **Objetivo:** Representa una tabla en la base de datos.
   * **Uso:** Se mapea directamente a una tabla en la base de datos con la ayuda de JPA (Java Persistence API).
   * **Ejemplo:** La clase **Oracion** que has proporcionado representa una tabla en la base de datos que guarda oraciones.
2. **DTO (Data Transfer Object):**
   * **Objetivo:** Representar o transportar datos entre procesos o entre capas de la aplicación. Facilita el intercambio de datos y puede excluir detalles innecesarios que tiene la entidad.
   * **Uso:** En operaciones de API, en lugar de enviar/recibir directamente una entidad, se utiliza un DTO.
   * **Ejemplo:** **OracionDTO** podría tener solo los campos que deseas enviar/recibir a través de tus endpoints.
3. **Service:**
   * **Objetivo:** Contiene la lógica empresarial y opera entre el controlador y el repositorio. Puede manejar transacciones, transformaciones de datos, etc.
   * **Uso:** Llamar a métodos del repositorio, realizar operaciones sobre los datos y devolver resultados al controlador.
   * **Ejemplo:** **OracionService** puede tener métodos para guardar una oración, buscarla, actualizarla, etc.
4. **Repositorio (Repository):**
   * **Objetivo:** Abstracción de la capa de datos. Proporciona operaciones CRUD (Crear, Leer, Actualizar, Eliminar) sin que se tenga que implementar la lógica detrás.
   * **Uso:** Se utiliza para interactuar con la base de datos directamente.
   * **Ejemplo:** **OracionRepository** podría extender **JpaRepository**, lo que proporciona automáticamente métodos para operaciones CRUD en la tabla **Oracion**.
5. **Controlador (Controller):**
   * **Objetivo:** Escuchar las solicitudes HTTP, gestionar la lógica de la solicitud y devolver una respuesta al cliente.
   * **Uso:** Define los endpoints de la API y maneja las solicitudes HTTP. Llama a la capa de servicio para obtener o manipular datos.
   * **Ejemplo:** **OracionController** tiene endpoints como **GET /api/oraciones/{id}** para obtener una oración por su ID.

La interacción típica en un flujo de trabajo sería la siguiente:

* Un cliente (como un navegador o una aplicación móvil) hace una solicitud HTTP.
* El **Controlador** recoge esta solicitud.
* Llama a un método en la capa de **Servicio**.
* El **Servicio** realiza la lógica empresarial y/o llama al **Repositorio** para interactuar con la base de datos.
* El **Repositorio** interactúa con la base de datos y devuelve los datos a la capa de **Servicio**.
* El **Servicio** podría convertir la **Entidad** en **DTO** (si es necesario) y devolverlo al **Controlador**.
* El **Controlador** devuelve el **DTO** como respuesta HTTP al cliente.