

**Por favor lea
y entienda el
enunciado
antes de
comenzar su
ejecución.**

Universidad Pontificia Bolivariana
Facultad Ingeniería en Energía, Computación y TIC
Programas de Ingeniería en Sistemas e Informática – Ingeniería en Ciencia de Datos
Curso de Tópicos Avanzados de Base de Datos
NRC: 20407, 20602 – Periodo 202510

Examen No. 4 – Valor 20%
Patrón Repositorio - Bases de datos NoSQL - Implementación de API

Fecha de Entrega: miércoles 14 de mayo de 2025, 6:00 pm (GMT -5)
Cualquier entrega después de fecha, su calificación será de **0.00 - sin excepciones.**

Solo se responden dudas sobre el examen hasta el lunes 12 de mayo, 6:00 pm (GMT-5)

Tiempo estimado de ejecución: 10 horas

Tipo de Evaluación:

- Grupal, máximo 2 integrantes por equipo. Opcionalmente, puede presentarse individualmente.
- Los miembros del equipo de trabajo pueden ser de diferentes grupos.

Nota importante:

Por favor responda a la confianza que se le brinda con este examen, trabajando honestamente. Si tiene dificultades para cumplir con el proyecto, por favor realice una valoración de la situación para que trabajemos un plan de mejoramiento específico a sus necesidades. **¡No haga trampa!**

Sobre supletorios

No se otorgarán plazos adicionales. Recuerde que por indicaciones de la Escuela de Ingeniería, cualquier evaluación superior al 10% de la definitiva del curso que no sea presentado oportunamente, deberá tramitar autorización para ejecución de supletorio.

Como dará bonificación este examen

Si elabora este examen en un **lenguaje distinto de C# .NET pero con MongoDB**, este examen lo exime del examen No. 3 – API REST con base de datos Relacional.

Si elabora este examen **en cualquier lenguaje pero con base de datos NoSQL orientada a documento diferente de MongoDB**, este examen lo exime del examen No. 3 – API REST con base de datos Relacional.

No aplica restricciones de bonificaciones previas. En otras palabras, se puede acumular con otras promociones! 😊

Sobre herramientas disponibles para la implementación

Motor de base de datos:

MongoDB. Para bonificación puede considerar MS CosmosDB, AWS DynamoDB, Google FireStore, Oracle NoSQL.

Lenguaje de programación:

C# .NET framework 8.x. Opcionalmente se podrá hacer en Java, PHP, Python, Javascript. **Por favor hable previamente con el docente si lo quiere hacer en otro lenguaje diferente a los mencionados.**

Entregable:

- Código fuente del proyecto publicado **en GitHub** en un repositorio **privado** denominado “**tadb_202510_ex04**”. Debe incluir al docente como colaborador del repositorio (usuario: jdrodas).
- Se debe evidenciar trabajo colaborativo. **Si se está haciendo en parejas, TODOS los estudiantes del equipo deben tener actualizaciones en el repositorio.**
- Instructivo de compilación y ejecución de la solución, publicado en el repositorio como un archivo “**README.md**”.

Notificar vía correo electrónico al docente (juand.rodasm@upb.edu.co) el URL del repositorio y los estudiantes que participaron en el proyecto.

Por favor no compartir ningún entregable a través de Microsoft OneDrive o Microsoft Teams.

Compuestos de Medicamentos

A raíz de las dificultades evidenciadas recientemente en nuestro país, se hace necesario tener una herramienta que permita identificar alternativas de medicamentos que puedan tener una composición similar, fabricadas por diferentes laboratorios.

Para ello, se implementará una API que permita realizar operaciones CRUD básicas sobre los medicamentos y sus respectivos compuestos que lo constituyen.

El modelo de datos tiene tres colecciones con la siguiente información:

Compuesto

- Id
- Nombre

Medicamento

- Id
- Nombre
- Fabricante

Compuestos por medicamento

- Id de compuesto
- Id de medicamento
- Concentración
- Unidad de medida de la concentración.

Datos Ejemplo:

Compuestos

Nombre
Acetaminofén
Cetirizina
Cafeína
Fenilefrina Clorhidrato

Medicamentos

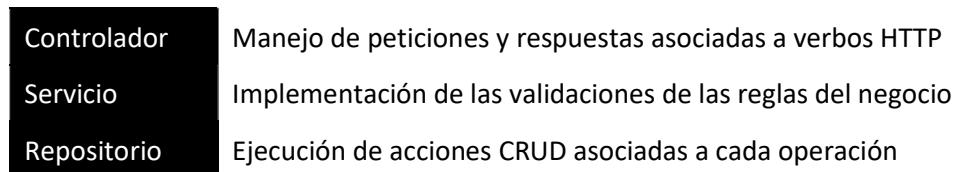
Nombre	Fabricante
Noxpirin	Laboratorios Siegried

Compuestos por Medicamento

Medicamento	Elemento	concentración	Unidad de medida
Noxpirin	Acetaminofén	500	mg
Noxpirin	Cetirizina	5	mg
Noxpirin	Cafeína	30	mg
Noxpirin	Fenilefrina Clorhidrato	10	mg

Los datos de ID son generados como ObjectIDs. En la tabla de compuestos por medicamentos **deben estar referenciados los Id de los compuestos y de los medicamentos**

Con estas especificaciones, se desea construir una API REST que implemente el patrón repositorio, con una distribución por capas de la siguiente manera:



Modelo	Clases que definen el estado y comportamiento de las entidades
Contexto	Conexión a la base de datos.

Inventario de peticiones que se espera implemente la API

Compuestos

- Listar todos los compuestos
(GET /api/compuestos)
- Listar un compuesto por Id
(GET /api/compuestos/{compuesto_id})
- Listar Medicamentos que tienen un compuesto por Id
(GET /api/compuestos/{compuesto_id}/medicamentos)
- Agregar un compuesto
(POST /api/compuestos)
- Actualizar un compuesto
(PUT /api/compuestos)
- Eliminar un compuesto
(DEL /api/compuestos/{compuesto_id})

Medicamentos

- Listar todos los medicamentos
(GET /api/medicamentos)
- Listar un medicamento por Id.
(GET /api/medicamentos/{medicamento_id})
- Listar compuestos de un medicamento por Id
(GET /api/medicamentos/{medicamento_id}/compuestos)
- Agregar un medicamento
(POST /api/medicamentos)
- Actualizar un medicamento
(PUT /api/medicamentos)

- Eliminar un medicamento
(DEL /api/medicamentos/{medicamento_id})

Para validar el funcionamiento del API, debe utilizarse para registrar 3 medicamentos con sus respectivos compuestos. En lo posible, seleccione medicamentos que tengan compuestos comunes para que pueda demostrar las consultas cruzadas entre entidades.

Rúbrica de la evaluación

Creación del repositorio en GitHub y publicación del proyecto: 20%

- Crear usuario en GitHub si no lo tiene
- Crear repositorio **privado** con las especificaciones indicadas.
- Invitar al docente al repositorio (usuario GitHub: jdrodas)
- Publicar el proyecto en el repositorio **privado**
- Evidenciar actualizaciones progresivas (*commits*) por parte de cada uno de los estudiantes que participan en el equipo.

Entregable: El repositorio en GitHub que evidencie las acciones descritas previamente

Implementación del modelo de datos: 20%

- Implementar un modelo No Relacional orientado a documentos que permita almacenar la información de las entidades referenciadas en el enunciado. Crear usuarios con privilegios limitados.
- La cantidad de colecciones, atributos y documentos hacen parte de su diseño pero deben tener consistencia en la nomenclatura.

Entregables:

- El script de creación del modelo de datos incluyendo el esquema JSON de validación para cada una de las colecciones resultantes.
- Los archivos de carga en formato JSON con la totalidad de la información migrada para todas las colecciones.
- Estos archivos deben estar almacenados en una carpeta llamada “Datos” dentro de su repositorio.

Implementación de peticiones para cada entidad: 5% por cada petición, 12 peticiones, 60% en total

- Para cada petición, evidencie la implementación del “stack” requerido: Método en el *Controller, Service y Repository*.
- Para las peticiones de consulta, los métodos en la capa de repositorio pueden utilizar sentencias SQL de consulta (SELECT)

Entregable: El “stack” completo para cada entidad, publicado en el repositorio.