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: 20635, 21333 – Periodo 202410

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

Fecha de Entrega: miércoles 8 de mayo de 2024, 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 martes 7 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!**

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

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.

Sobre atención de docente:

Para garantizar una ejecución oportuna y con una utilización racional del tiempo asignado, por favor tenga presente los siguientes **horarios de indisponibilidad del docente**:

- Martes a viernes de 8 a 10 pm: Horario de seguimiento de bonificaciones de Kanban
- Martes a viernes de 6 a 8 am: Horario de clase
- Sábado de 12m a lunes 6 am: Horario de descanso y actividades familiares.
- Todos los días de 10 pm a 6 am del día siguiente: Horario de descanso.
- martes 7 de mayo: **TODO EL DIA.**

Origen de los datos

Los datos que se utilizarán en este examen fueron obtenidos de la página del **Organismo Internacional de Energía Atómica** (https://www.iaea.org/). Allí se encuentra la base de datos de reactores nucleares de investigación, la cual puede accederse desde esta dirección:

https://nucleus.iaea.org/rrdb/#/home

De allí, solo las siguientes columnas fueron tenidas en cuenta:

- Nombre del Reactor
- País donde está ubicado
- Ciudad más cercana donde está ubicado
- Tipo de Reactor
- Potencia Térmica expresada en KW
- Estado del reactor
- Fecha Primera reacción crítica.

Luego de este proceso, se tienen 840 reactores para realizar las actividades de esta evaluación. El archivo con los datos se denomina **IAEA_Reactores_Nucleares_Investigacion.zip**

Sobre herramientas disponibles para la implementación

Motor de base de datos:

MongoDB. Según lo acordado en la formulación de proyectos avanzados notificada a comienzo de semestre, se podrá realizar en otras bases de datos NoSQL orientadas al documento como **CosmoDB y Oracle NoSQL**.

De implementar el examen correctamente, **se otorgará bonificación del 20%** de acuerdo con lo propuesto. Esta bonificación es excluyente con otras bonificaciones especificadas en dicha formulación.

Lenguaje de programación:

C# .NET framework 7.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.**

Prioridad y Promesa

Si para el examen No. 3 no terminó completamente las implementaciones de las capas de Controladores y Servicios, recuerde que si los termina correctamente y actualiza el repositorio del examen No. 3, **su calificación de ese examen puede mejorar hasta en un 50% por cada ruta pendiente.**

Recuerde publicar su contenido luego del commit realizado por el docente.

Entregable:

- Código fuente del proyecto publicado **en GitHub** en un repositorio **privado** denominado "**tadb_202410_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.

Reactores Nucleares para Investigación

En el año 2024, se ha comenzado en Colombia la gestión de una ley marco de regulación para la utilización de la energía nuclear en todas sus aplicaciones comerciales, académicas y de investigación.

Dentro de los primeros pasos para poder tener el aval y acompañamiento del **Organismo Internacional de Energía Atómica,** es tener una legislación actualizada, vigente y socializada. La Red Nuclear Colombiana es una iniciativa apoyada por la Corporación Ruta N de Medellín con el fin de educar al público en general en temas relacionados con la ciencia y la tecnología nuclear.

Para apoyar esta iniciativa, se propone la creación de aplicaciones que ayuden al proceso de divulgación de los temas relacionados con la ciencia nuclear.

Se desea construir entonces una API que permita consultar y actualizar la base de datos de los reactores nucleares de investigación, la cual se puede descargar de los enlaces previamente indicados.

Esta API tendrá una ruta para realizar las acciones CRUD asociadas al **reactor** (insertar, consultar, actualizar, eliminar) y un par de rutas para consultar por **tipo** y por **ubicación**.

Propiedades del Reactor:

- ID
- Nombre,
- Ubicación
- Tipo
- Estado
- Fecha Primera Reacción.

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

Controlador	Manejo de peticiones y respuestas asociadas a verbos HTTP
Servicio	Implementación de las validaciones de las reglas del negocio
Repositorio	Ejecución de acciones CRUD asociadas a cada operación
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

Reactor

1. Obtener reactores registrados
2. Obtener un reactor por Id
3. Crear un nuevo reactor
4. Actualizar un reactor existente
5. Eliminar un reactor existente

Tipo

6. Obtener tipos de reactores registrados
7. Obtener tipo de reactor por Id. **Respuesta incluye todos los reactores asociados al tipo.**

Ubicaciones

8. Obtener Ubicaciones Registradas
9. Obtener Ubicación por Id.
10. Obtener Reactores registrados por Ubicación

Para validar el funcionamiento del API, debe utilizarse para registrar 5 reactores ubicados en alguno de los países de Latinoamérica: Argentina, Brasil, Chile, Colombia, Perú, Venezuela, Uruguay.

Bonificaciones adicionales:

Eliminación de la obligatoriedad del examen No. 5.

La elaboración de los exámenes 3 y 4, en plataformas diferentes a C# - Web API, le permitirá eximirse del examen No. 5. Solo aplica si no ha recibido bonificación de los exámenes 1 y 2.

Es necesario que ambos sean implementados de esta manera para optar por la bonificación ofrecida. Entregar solo uno de los dos no será válido para optar por esta bonificación.

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.
- 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: 6% por cada petición, 10 peticiones, 60% en total

- Para cada petición, evidencie la implementación del "*stack*" requerido: Método en el *Controller*, *Service* y *Repository*.
- Todos los controladores y servicios deben coincidir con los creados en el examen previo, aunque pueden cambiar de acuerdo a necesidades de referenciación de ObjectIds y validaciones.

Entregable: El "stack" completo para cada entidad publicado en el repositorio.