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: 19525, 17461, 18089 – Periodo 202320

Examen No. 3 – Valor 20%
Patrón Repositorio – Implementación de API

Fecha de Entrega: viernes 13 de octubre de 2023, 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.

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
- Sábados de 8am a 12m: 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.
- Viernes 13 de octubre: **TODO EL DIA.**

Sobre el dominio del problema y los datos registrados allí

- **Este es un dominio de problema ficticio y excesivamente simplificado** sobre una situación real que ocurre en el distrito de Medellín.
- **Las reglas del dominio del problema son ficticias** y no reflejan la realidad del proceso de abastecimiento de autobuses y cargadores para la electrificación del transporte público del distrito.
- **Los datos son igualmente ficticios** y no representan limitaciones ni ventajas de fabricantes de autobuses y cargadores.

Sobre herramientas disponibles para la implementación

Motor de base de datos:

PostgreSQL. Opcionalmente se podrá hacer Oracle, SQL Server, MySQL. **No se puede hacer en SQLite/MS ACCESS, MS EXCEL.**

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.**

Entregable:

- Código fuente del proyecto publicado **en GitHub** en un repositorio **privado** denominado **"tadb_202320_ex03"**. 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.

Gestión de Programación de Transporte Público Eléctrico

La Secretaría de Movilidad del Municipio de Medellín en conjunto con la Empresa de Transporte Masivo del Valle de Aburrá están realizando un estudio sobre la programación de la utilización y mantenimiento de los buses eléctricos de tal manera que se pueda realizar una sustitución progresiva cumpliendo los requerimientos mínimos de operación del servicio.

En las etapas iniciales, se requiere una aplicación en la que se pueda registrar **de manera individual y manual**, la utilización de autobuses y cargadores por hora del día, teniendo presente algunas restricciones manifestadas como reglas del dominio, expresadas de esta manera:

Horario:

- El servicio de transporte público deberá operar las 24 horas del día.
- Para simplificar el análisis, solo se agendarán utilización de buses y cargadores de lunes a viernes de manera consistente, es decir, todos los días de la misma forma.
- Se define como horario pico el ubicado en las franjas horarias entre las **5 am y las 9 am** para la mañana y entre las **4 pm y 8 pm** para la tarde noche.
- Se debe garantizar que en el horario declarado pico, **ningún cargador** está siendo utilizado y **todos los autobuses** están en operación.

Cargador

- El cargador puede operar de manera continua las **24 horas del día**, pero como todos los **autobuses** deben estar operando en horario pico, **ningún cargador** debe estar siendo utilizado en esas franjas horarias.
- Un cargador solo puede atender un **autobús** a la vez.
- Un cargador estará utilizado por lapsos de **2 horas**, para un máximo de 12 ciclos de carga por día. SON 8 CICLOS
- Un cargador puede estar desocupado, es decir, puede no estar cargando a ningún autobús.

Autobús

- Un autobús puede trabajar continuamente durante **4 horas** y debe **inmediatamente** detenerse por **2 horas** para actividades de limpieza, mantenimiento, cambio de conductor y cargue de batería.
- Un autobús **tiene que estar en operación** en las franjas horarias definidas como horario pico.
- Un autobús **no puede estar utilizando un cargador** en las franjas horarias definidas como horario pico.
- **Un autobús puede estar parqueado** sin estar cargando ni estar operando, siempre y cuando no sea en las franjas horarias definidas como horario pico.

Disponibilidad del servicio

- La franja horaria declarada como horario pico, debe contar con **disponibilidad de buses cada 5 minutos**. Esto quiere decir que cada hora en dicha franja, **12 autobuses** deben estar siempre operando.
- Fuera del horario pico, un autobús puede estar en operación, en mantenimiento o parqueado, pero **siempre debe estar en operación en horario pico**.

Con estas reglas del dominio, se desea construir una API 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

Cargador

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

Autobús

6. Obtener autobuses registrados
7. Obtener autobús por Id
8. Crear un nuevo autobús
9. Actualizar un autobús existente
10. Eliminar un autobús existente
11. Registrar utilización de cargador por hora del día
12. Actualizar utilización de cargador por hora del día
13. Eliminar utilización de cargador por hora del día
14. Registrar operación por hora del día
15. Actualizar operación por hora del día
16. Eliminar operación por hora del día

Horario

17. Obtener horas registradas
18. Obtener informe de hora por Id **(*)**
19. Obtener informe de utilización de cargadores por hora
20. Obtener informe de utilización de buses por hora

(*): Para cada hora, cuando se consulta en el correspondiente controlador, se debe indicar el % de utilización de cargadores (Cargadores usados / Cargadores Disponibles) y % de buses en operación (Buses en operación / Buses Disponibles).

Se considerará que la API tiene correcta operación cuando al consultar las horas del horario pico, **todas** aparecen con utilización de cargadores al 0% y buses en operación al 100%

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 con las especificaciones indicadas
- Publicar el proyecto en el repositorio
- 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 relacional que implemente las restricciones (*constraints*) correspondientes: PK, FK, UK
- La cantidad de tablas, atributos y relaciones hacen parte de su diseño pero deben tener consistencia en la nomenclatura.

Entregable: El diagrama relacional como imagen JPG publicada en el repositorio creado previamente.

Implementación de peticiones para cada entidad: 3% por cada petición, 20 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)
- Para las peticiones de creación, actualización y eliminación, debe utilizar invocaciones a procedimientos almacenados.

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