**Iteración 3**

Kevin Babativa

Julian Arenas

1. **Análisis de impacto:** La introducción de los nuevos requerimientos de consulta y de registro implica que el programa de RotondAndes ocupe más espacio en disco. Así mismo, el volumen ocupado en general por el programa, incluyendo, la base de datos como la estructura del programa se verá aumentado.

Por otra parte, el tamaño del código, como su complejidad, aumenta de manera considerable, esto se debe a que los nuevos requerimientos se obtienen con la combinación de requerimientos anteriores o la consulta de varias tablas distintas en una sola sentencia de la base de datos.

Para concluir, esta iteración a pesar de ser más corta que la iteración 2, presenta un mayor nivel de complejidad para poder hacerla por las razones que se mencionaron arriba.

1. Efectivamente, las tablas se encuentran en BCNF.
2. **Atomicidad:** La atomicidad se garantiza a través del uso del *COMMIT* indicar como finalizada la transacción, si la ejecución del programa termina de manera inesperada en medio de una transacción no se llegará a ese punto.

**Consistencia:** Para garantizar la consistencia se parte desde el *CRUD* (Create, Retrieve, Update, Delete), en primer lugar, las operaciones del *CRUD* deben ser consistentes. Si estas no alcanzan los resultados esperados no se ejecuta nada con ellas, por lo cual, para garantizar la consistencia se parte desde garantizar la consistencia del *CRUD*, a medida que se va especializando cada función del programa, se prueba de tal forma que, de lo esperado, de lo contrario, se vuelve a programar.

**Aislamiento:** Se usa el nivel *serializable* para garantizar el aislamiento, este nivel es el más restrictivo, sin embargo, los recursos no se verán bloqueados por mucho tiempo, lo que permite que los recursos se bloqueen por un momento pequeño de tiempo y siga la siguiente transacción.

**Durabilidad:** La durabilidad se conforma gracias al cumplimiento de las tres características anteriores junto a la característica principal de las bases de datos que es la persistencia, la persistencia junto a estas tres características conforman unos datos que son consistentes y duraran hasta el fin de los tiempos.

1. ¿Qué diferencias hay en el manejo transaccional por parte de un contenedor de aplicaciones con respecto al manejo dado por parte del programador de la aplicación? Explique claramente las ventajas y desventajas de cada uno de ellos y cómo se refleja en su arquitectura de software y diseño detallado de su RotondAndes.

El uso de un contenedor de aplicaciones para desarrollar la iteración ayuda principalmente en la conexión con la web, el contenedor hace de intermediario entre la capa REST y la base de datos, así mismo el contenedor es el que ejecuta la lógica de negocio, ganando así menos complejidad a la hora de ejecutarlo. Así mismo, se maneja como mayor facilidad la entrada de datos del usuario como el manejo de estos parámetros, el contenedor de aplicaciones ayuda al manejo de datos JSON para poder interpretarlos y ejecutar las diversas acciones con ellos. Finalmente, el manejo de memoria se lleva a cabo por el contenedor, dejando así un problema menos por el que preocuparse.

Sin embargo, una de las principales desventajas que se encuentran al usar el contenedor de aplicaciones es el uso de SQL embebido, al hacer uso de este, el manejo de las sentencias al momento de escribirlas se vuelve más complicado, en el momento no es tan claro para el programador. Así mismo, se hacen varias tareas adicionales como lo son establecer la conexión con la base de datos, agregar los recursos, cerrar la conexión y cerrar los recursos. Finalmente, al trabajar con capas se vuelve más complejo por el hecho de que es imposible utilizar funciones de capas del mismo nivel o de un nivel más alto, haciendo así, un código más ilegible.