Sistemas transaccionales – ISIS2304 – Entrega 3
Leidy J. Lozano Flórez
Laura M. Restrepo Palomino
Karen T. Vera Hernández

Documentación de los Requerimientos Funcionales

Requerimientos del CRUD

Requerimiento funcional 1

Nombre	RF1 - Registrar / actualizar / borrar / consultar tipos de habitación
Resumen	Se implementan las operaciones CRUD para la entidad tipos de habitación. Usa los métodos del repositorio Tipo_habitacionRepository - buscar() - buscarPorId(@Query("{_id: ?0}")) - eliminar (@DeleteQuery("{'_id' : ?0}") int id) - actualizar (@Query("{_id: ?0}"), @Update("{$set:{nombre: ?1, capacidad: ?2}}") int id, String nombre, int capacidad) - insertar (Tipo_habitacion tipoHabitacion)
Usuario	Usuario de la base de datos del hotel
Entradas	
- [REGISTRAR] Recibe un documento de tipo habitación. - [ACTUALIZAR] Recibe un ID del tipo de habitación que se desea actualizar junto con los nuevos atributos correspondientes al documento. - [ELIMINAR] Recibe el ID del tipo de habitación que se va a eliminar. - [CONSULTAR] Recibe el ID del tipo de uhabitación que se va a consultar.	
Resultados	
- Se registra un nuevo tipo de habitación en la base de datos del hotel, el cual puede sersometido a las demás operaciones CRUD. - Se actualizan los atributos de un tipo de habitación en la base de datos del hotel. - Se elimina un tipo de habitación permanentemente de la base del hotel. - Se obtiene el tipo de habitación correspondiente al ID ingresado.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Requerimiento funcional 2

Nombre	RF2 - Registrar / actualizar / borrar / consultar habitación
Resumen	Se implementan las operaciones CRUD para la entidad habitación. Usa los métodos del repositorio HabitaciónRepository - buscar() - buscarPorId(@Query("{_id: ?0}") int id) - eliminar (@DeleteQuery("{'_id' : ?0}") int id) - actualizar (@Query("{_id: ?0}"), @Update("{$set:{costo_noche: ?1, tipo: ?2}}") int id, Double costo_noche, int tipo) - insertar (Habitacion habitacion)

Usuario	Usuario de la base de datos del hotel.
Entradas	
- [REGISTRAR] Recibe un documento de habitación. - [ACTUALIZAR] Recibe un ID de la habitación que se desea actualizar junto con los nuevos atributos correspondientes al documento. - [ELIMINAR] Recibe el número de documento de la habitación que se va a eliminar. - [CONSULTAR] Recibe el número de documento de la habitación que se va a consultar.	
Resultados	
- Se registra una nueva habitación en la base de datos del hotel, el cual puede ser sometido alas demás operaciones CRUD. - Se actualiza la información de una habitación en la base de datos del hotel. - Se elimina una habitación permanentemente de la base del hotel. - Se obtiene la información vinculada a una habitación dado su número de identificación.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Requerimiento funcional 3

Nombre	RF3 - Registrar / actualizar / borrar / consultar servicio del hotel.
Resumen	Se implementan las operaciones CRUD para la entidad servicio. Usa losmétodos del repositorio ServicioRepository: - buscar() - buscarPorId(@Query("{_id: ?0}") int id) - buscarPorNombre(@Query("{nombre: ?0}"), String nombre) - eliminar (@DeleteQuery("{'_id' : ?0}") int id) - actualizar @Update("{$set:{nombre: ?1, tipo_cobro: ?2, precio: ?3, capacidad: ?4, estilo: ?5, costo_adicional: ?6}}") int id, String nombre, String tipo_cobro, Double precio, int capacidad, String estilo, Double costo_adicional) insertar (Habitacion habitacion)
Usuario	Usuario de la base de datos del hotel.
Entradas	
- [REGISTRAR] Recibe un documento de servicio. - [ACTUALIZAR] Recibe un ID del servicio que se desea actualizar junto con los nuevos atributos correspondientes al documento. - [ELIMINAR] Recibe el ID del servicio que se va a eliminar. - [CONSULTAR] Recibe el ID del servicio que se va a consultar.	

Resultados
- Se registra un nuevo servicio en la base de datos del hotel, el cual puede sersometido a las demás operaciones CRUD. - Se actualiza la información de un servicio en la base de datos del hotel. - Se elimina un servicio permanentemente de la base del hotel. - Se obtiene la información del servicio vinculado al ID ingresado.
RNF asociados
RNF1 – Persistencia y RNF2 – Distribución

Requisito funcional 4

Nombre	RF4 - Registrar / actualizar / borrar / consultar reserva de alojamiento
Resumen	Se implementan las operaciones CRUD para la entidad reserva. Usa losmétodos del repositorio HabitacionRepository - buscarReservas(@Aggregation […]) - buscarReservasPor habitación (@Aggregation […] int id) - buscarReservaPorCodigo(@Aggregation […] int codigo) - eliminar (@DeleteQuery("{'_id' : ?0}") int id) - actualizar (@Query("{_id: ?0}"), @Update("{$set:{nombre: ?1, capacidad: ?2}}") int id, String nombre, int capacidad) - insertarReserva (int id, int codigo, String fecha_entrada, String fecha_salida, String estado, int num_huespedes, String num_documento, String tipo_documento, String nombre, String correo) - verificarDisponibilidad(@Aggregation […] int id, String fecha_entrada, String fecha_salida)
Usuario	Usuario de la base de datos del hotel
Entradas	
- [REGISTRAR] Recibe los atributos necesarios para crear un documento de tipo reserva. - [ACTUALIZAR] Recibe un ID de la reserva que se desea actualizar junto con los nuevos atributos correspondientes al documento. - [ELIMINAR] Recibe el ID de la reserva que se va a eliminar. - [CONSULTAR] Recibe el ID de la reserva que se va a consultar.	
Resultados	
- Se registra una nueva reserva en la base de datos del hotel, el cual puede ser sometido a las demás operaciones CRUD. - Se actualizan los atributos de una reserva en la base de datos del hotel. - Se elimina una reserva permanentemente de la base del hotel. - Se obtiene la reserva correspondiente al ID ingresado.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Leidy J. Lozano Flórez
Laura M. Restrepo Palomino
Karen T. Vera Hernández

Requisito funcional 5

Nombre	RF5 - Registrar / actualizar / borrar / consultar llegada de un cliente al hotel.
Resumen	Se implementan las operaciones CRUD para la entidad reserva (Que contiene la llegada de un cliente). Usa losmétodos del repositorio HabitacionRepository - buscarReservaPorCodigo(@Aggregation […] int codigo) - actualizar (@Query("{_id: ?0}"), @Update("{$set:{nombre: ?1, capacidad: ?2}}") int id, String nombre, int capacidad)
Usuario	Usuario de la base de datos del hotel
Entradas	
- [REGISTRAR] Recibe un ID de la reserva asociada, - [ACTUALIZAR] Recibe un ID de la reserva a la cual se le va a actualizar el estado de la reserva. - [ELIMINAR] Recibe el ID de la reserva, que contiene la llegada del hotel que se va a eliminar. - [CONSULTAR] Recibe el ID de una reserva, que contiene la llegada al hotel que se va a consultar.	
Resultados	
- Se registra la llegada de un cliente al hotel en una reserva, la cual se puede someter a operaciones CRUD. - Se actualiza el estado de una reserva de la base del hotel. - Se elimina la llegada de un cliente permanentemente de la base del hotel. - Se obtiene la entrada al hotel de la reserva consultada.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Requisito funcional 6

Nombre	RF6 - Registrar / actualizar / borrar / consultar consumo de un servicio
Resumen	Se implementan las operaciones CRUD para la entidad consumo. Usa losmétodos del repositorio HabitacionRepository - buscarConsumos(@Aggregation […]) - buscarConsumosHabitación (@Aggregation […] int id) - buscarConsumoPorId(@Aggregation […] int id_consumo) - eliminar (@DeleteQuery("{'_id' : ?0}") int id) - actualizar @Query("{_id: ?0, 'consumos._id': ?1}"), @Update([…]) int id, int id_consumo, int servicio, String cliente, String fecha, String hora) - insertarConsumo (int id, int id_consumo, int servicio, String cliente, String fecha, String hora) - verificarDisponibilidad(@Aggregation […] int id, String fecha_entrada, String fecha_salida)
Usuario	Usuario de la base de datos del hotel
Entradas	
- [REGISTRAR] Recibe los atributos necesarios para crear un documento de tipo consumo. - [ACTUALIZAR] Recibe un ID del consumo que se desea actualizar junto con los nuevos atributos correspondientes al documento. - [ELIMINAR] Recibe el ID del consumo que se va a eliminar. - [CONSULTAR] Recibe el ID del consumo que se va a consultar.	

Resultados
- Se registra un nuevo consumo en la base de datos del hotel, el cual puede ser sometido a las demás operaciones CRUD. - Se actualizan los atributos de un consumo en la base de datos del hotel. - Se elimina un consumo permanentemente de la base del hotel. - Se obtiene el consumo correspondiente al ID ingresado.
RNF asociados
RNF1 – Persistencia y RNF2 – Distribución

Requisito funcional 7

Nombre	RF5 - Registrar / actualizar / borrar / consultar salida de un cliente al hotel.
Resumen	Se implementan las operaciones CRUD para la entidad reserva (que incluye salida del hotel). Usa losmétodos del repositorio HabitacionRepository - buscarReservaPorCodigo(@Aggregation […] int codigo) - actualizar (@Query("{_id: ?0}"), @Update("{$set:{nombre: ?1, capacidad: ?2}}") int id, String nombre, int capacidad)
Usuario	Usuario de la base de datos del hotel
Entradas	
- [REGISTRAR] Recibe un ID de la reserva asociada. - [ACTUALIZAR] Recibe un ID de la reserva a la cual se le va a actualizar el estado de la reserva. - [ELIMINAR] Recibe el ID de la reserva, que contiene la salida del hotel que se va a eliminar. - [CONSULTAR] Recibe el ID de una reserva, que contiene la salida al hotel que se va a consultar.	
Resultados	
- Se registra la salida de un cliente al hotel en una reserva, la cual se puede someter a operaciones CRUD. - Se actualiza el estado de una reserva de la base del hotel. - Se elimina la salida de un cliente permanentemente de la base del hotel. - Se obtiene la salida al hotel de la reserva consultada.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Sistemas transaccionales – ISIS2304 – Entrega 3
Leidy J. Lozano Flórez
Laura M. Restrepo Palomino
Karen T. Vera Hernández

Requisitos funcionales de consultas básica

Requerimiento funcional de consulta 1

Nombre	RFC1 - Mostrar el dinero recolectado por servicios en cada habitación en el último año corrido.
Resumen	Se implementan las operaciones necesarias para obtener el dinero que recolectan todas las habitaciones gracias a sus servicios consumidos. Implementa los métodos del repositorio habitacionRepository - @Aggregation […] darDineroRecolectado(String fecha_inicio, String fecha_fin)
Usuario	Usuario de la base de datos del hotel
Entradas	
- Recibe las fechas del año corrido.	
Resultados	
- Se obtiene un valor que refleja el dinero recolectado por cada habitación con base a los consumos cargados a ella.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Requerimiento funcional de consulta 2

Nombre	RFC2 - Mostrar el índice de ocupación de cada una de las habitaciones del hotel en el último año corrido
Resumen	Se implementan las operaciones necesarias para obtener el % de ocupación de cada habitación en el último año. Implementa los métodos del repositorio habitacionRepository - @Aggregation […] darIndice_ocupacion(String fecha_inicio, String fecha_fin, long dias)
Usuario	Usuario de la base de datos del hotel
Entradas	
- Recibe las fechas del año corrido y la cantidad de días total en un año.	
Resultados	
- Se obtiene un porcentaje que refleja el índice de ocupación para cada una de las habitaciones del hotel.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Sistemas transaccionales – ISIS2304 – Entrega 3
Leidy J. Lozano Flórez
Laura M. Restrepo Palomino
Karen T. Vera Hernández

Requerimiento funcional de consulta 3

Nombre	RFC3 - Mostrar el consumo en hotelandes por un cliente, en un rango de fechas indicado
Resumen	Se implementan las operaciones necesarias para obtener los consumos de un cliente específico en un rango de fechas indicado. - @Aggregation […]darConsumosCliente(String cliente, String fecha_inicio, String fecha_fin)
Usuario	Usuario de la base de datos del hotel
Entradas	
- Recibe las fechas de interés y el número de documento del cliente que se desea consultar.	
Resultados	
- Se obtiene una lista de los consumos realizados por el cliente en el rango de fechas.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	

Requisitos funcionales de consultas avanzada

Nombre	RFC4 - Consultar consumo en hotelandes
Resumen	Se implementan las operaciones necesarias para conocer la información de los clientes que consumieron al menos una vez un determinado servicio del hotel, en un rango de fechas. Los resultados deben son clasificados según un criterio deseado por quien realiza la consulta. En la clasificación debe ofrecerse la posibilidad de agrupamiento y ordenamiento de las respuestas según los intereses del usuario que consulta como, por ejemplo, por los datos del cliente, por fecha y número de veces que se utilizó el servicio. Implementa los métodos de habitacionRepository. - @Aggregation […] darClientesConsumo(Integer servicio, String fecha, Integer veces, String num_documento, String tipo_documento, String nombre, String correo)
Usuario	Usuario de la base de datos del hotel
Entradas	
- Recibe el servicio de interés y los distintos filtros que se pueden tomar para la consulta, ya sea una fecha específica, la cantidad de veces o los distintos datos del cliente.	
Resultados	
- Se obtiene la información deseada según el criterio de filtro seleccionado.	
RNF asociados	
RNF1 – Persistencia y RNF2 – Distribución	