Requerimientos funcionales

RF1. Registrar un restaurante. Para esto se necesita el nombre del restaurante, el NIT y el nombre del administrador. Si no existe otro restaurante con el NIT ingresado, entonces el restaurante se registra exitosamente.

RF2. Registrar un producto. Para esto se necesita el código, el nombre, una descripción, el costo y el NIT del restaurante que ofrece dicho producto. Hay dos casos en los que no se podrá registrar un producto. Primero, no existe un restaurante que tenga el NIT ingresado. Segundo, ya existe un producto con el código ingresado. Si ninguno de estos casos ocurre el producto es registrado exitosamente.

RF3. Registrar un cliente. Para esto se necesita el tipo de identificación, el número de identificación, el nombre, el apellido, el teléfono y la dirección del cliente. Si no existe otro cliente con el mismo número de identificación, entonces el cliente se registra exitosamente. Cuando se agrega un cliente este se inserta de forma ordenada teniendo en cuenta su apellido y nombre, pues la lista de clientes está organizada alfabéticamente de forma descendente por apellido y nombre.

RF4. Registrar un pedido. La información necesaria para registrar un pedido es el código de pedido (autogenerado), la fecha y hora en la que se realizó (tomada del sistema), el número de identificación del cliente que está haciendo el pedido, el NIT del restaurante y la lista de productos a pedir. De cada producto se conoce el código del producto y la cantidad que se pidió. Si todos los productos del pedido pertenecen al restaurante que el cliente eligió para hacer el pedido, entonces el pedido se registra exitosamente.

RF5. Actualizar los datos de un restaurante dado su NIT. El usuario debe indicar que dato va actualizar (puede ser el nombre del restaurante, el NIT o el nombre del administrador) y luego introducir la información para actualizar ese dato. Si existe un restaurante que tenga el NIT ingresado inicialmente entonces el dato se le actualizará a ese restaurante.

RF6. Actualizar los datos de un producto dado su código. El usuario debe indicar que dato va actualizar (puede ser el código, el nombre, la descripción, el costo, o el NIT del restaurante que ofrece dicho producto) y luego introducir la información para actualizar ese dato. Si existe un producto que tenga el código ingresado inicialmente entonces el dato se le actualizará a ese producto.

RF7. Actualizar los datos de un cliente dado su número de documento. El usuario debe indicar que dato va actualizar (puede ser el tipo de identificación, el número de identificación, el teléfono, la dirección del cliente o la lista de pedidos) y luego introducir la información para actualizar ese dato. Si existe un cliente que tenga el número de documento ingresado inicialmente entonces el dato se le actualizará a ese cliente.

RF8. Actualizar los datos de un pedido dado su código. El usuario debe indicar que dato va actualizar (puede ser el número de documento del cliente que realizo el pedido, el NIT del restaurante que ofrece los productos, la lista de productos o el estado de la orden) y luego introducir la información para actualizar ese dato. El estado de un pedido solo se puede actualizar hacia adelante (de solicitado a en proceso, de en proceso ha entregado, de enviado a entregado),

no hacía atrás. Si existe un pedido que tenga el código ingresado inicialmente entonces el dato se le actualizará a ese pedido.

RF9. Guardar la información en archivos serializados de forma automática cada vez que se actualice o se registre información.

RF10. Generar un archivo cvs de pedidos. Para cada pedido se conocen los datos del restaurante, del cliente y del producto pedido. El listado debe estar ordenado en este orden: NIT del restaurante ascendente, documento del cliente descendente, fecha del pedido ascendente y código del producto ascendente. El usuario debe ingresar cual es el separador que utilizará y el nombre que tendrá el archivo. La primera línea tendrá el nombre de las columnas separadas también por dicho separador. Si no existe un archivo con el nombre ingresado entonces el archivo se genera exitosamente.

RF11. Mostrar en pantalla todos los restaurantes en orden alfabético ascendente.

RF12. Mostrar en pantalla todos los clientes en orden de su número telefónico descendente.

RF13. Buscar eficientemente un cliente dado su nombre completo e indicar el tiempo que tardo la búsqueda. Si se encuentra el nombre los datos de ese cliente se imprimirán en pantalla. Si no se encuentra ningún cliente con ese nombre, entonces se le informará al usuario. En ambos casos se muestra el tiempo que tardó la búsqueda.

RF14. Importar datos de un archivo cvs con información de restaurantes. Si un restaurante a importar tiene el mismo NIT que el de un restaurante guardado, entonces ese restaurante no se importa.

RF15. Importar datos de un archivo cvs con información de clientes. Si un cliente a importar tiene el mismo número de documento que el de un cliente guardado, entonces ese cliente no se importa.

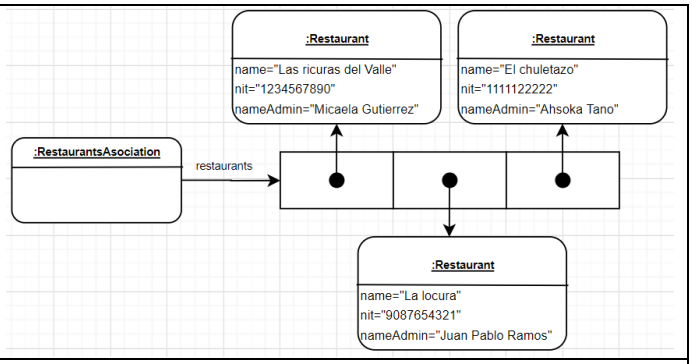
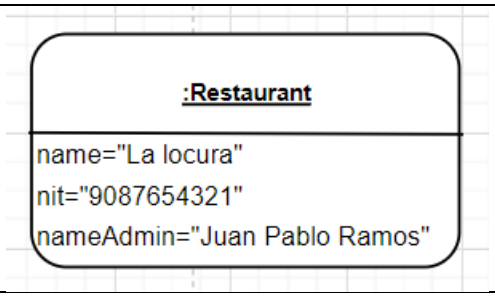
RF16. Importar datos de un archivo cvs con información de productos. Si un producto a importar tiene el mismo código que el de un producto guardado o no existe un restaurante con el NIT que tiene el producto, entonces ese producto no se importa.

RF17. Importar datos de un archivo cvs con información de pedidos. Si una orden a importar tiene el mismo código que el de una orden guardada, o el NIT del restaurante al que pertenece la orden no existe, o la lista de productos esta vacía, o no todos los productos pertenecen al mismo restaurante, o alguno de los productos no existe, o el cliente al que pertenece esa orden no existe, entonces esa orden no se importa.

Pruebas Unitarias

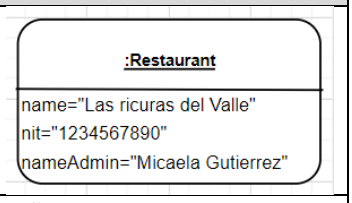
Escenarios

Nombre	**Clase**	**Escenario**

setupScenary1	RestaurantsAsociationsTest	
setupScenary1	RestaurantTest	Vacío
setupScenary2	RestaurantTest	

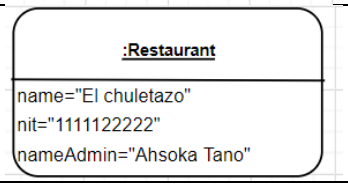
Casos de prueba

Objetivo de la Prueba: Verificar que se actualicen correctamente los datos de un restaurante				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
RestaurantsAsociation	updateDataRestaurant	setupScenary1	nit=”1234567890” option=1 data=”La sopaza”	

Objetivo de la Prueba: Verificar que la búsqueda en la lista de restaurantes retorne el restaurante buscado si existe y null si no existe.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
RestaurantsAsociation	findRestaurant	setupScenary2	nit=”1234567890”	
RestaurantsAsociation	findRestaurant	setupScenary2	Nit=”1111111111”	null

Objetivo de la Prueba: Verificar que se crea exitosamente un restaurante y que todos sus atributos sean asignados correctamente.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
Restaurant	Restaurant	setupScenary2	name=”La locura” nit=”1234567891”	Se crea un nuevo restaurante

			nameAdmin=”Juan Pablo Ramos”	exitosamente. Cada uno de los atributos del nuevo restaurante tiene asignado correctamente la información pasada por parámetro.

Objetivo de la Prueba: Verificar que se comparan correctamente dos restaurantes teniendo en cuenta sus nombres. Si el restaurante es menor entonces el método retorna un valor negativo, si es mayor un valor positivo, si son iguales 0.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
Restaurant	compareTo	setupScenary2		Un número negativo
Restaurant	compareTo	setupScenary2		Un número positivo