**Inventory Management System - Put and Delete**

**Grade settings**: Maximum Grade: 100  
**Disable external file upload, paste and drop external content**: Yes  
**Based on**: [Inventory Management System - Put and Delete](https://cognizant.tekstac.com/mod/vpl/view.php?id=9509)  
**Run**: Yes **Save**: Yes **Evaluate**: Yes  
**Automatic grade**: Yes **Maximum execution time**: 120 s **Maximum memory used**: 1 GiB

**Objective:**

To work with @RestController, http methods - PUT and DELETE

**Concept Explanation:**

1. **HTTP PUT** method is like web's handy "overwrite" button. It's perfect for times when you need to give something a complete makeover. Often used for completely replacing an existing resource with the data provided in the request body.
2. The**HTTP PATCH** method is akin to a "quick fix" tool on the web. It's perfect for making partial updates to existing records without needing to send the entire object.
3. The **HTTP DELETE**method acts as the web's "erase" button. It's used to remove data from the server, ensuring that unwanted or obsolete information can be cleaned up from your application.

**Concept Implementation:**

1. In the REST Controller, named **OrderController** for Inventory Management, we should map the endpoint**/BHO/updateBookQuantity/101/10**to the annotation corresponding to the **HTTP PUT** or **HTTP PATCH** request, as it is designed for updating the quantity of the book.
2. We should map the endpoint**/BHO/deleteOrder/101**to the annotation corresponding to the**HTTP DELETE**request, which is used for removing an order from the system based on the given id.

**Inventory Management System**

[**Click here to download the code skeleton**](https://cognizant.tekstac.com/mod/vpl/viewfile.php/21051/mod_vpl/intro/InventoryManagementSystem%20%282%29.zip)

A local bookstore, Book Haven, aims to streamline its inventory management system by incorporating a web service for handling book orders, updating book availability, cancelling orders, and retrieving order details. Help Zee-tech to automate the above process by developing Rest Service using Maven to modify and delete an order.

As an enhancement to the previous requirement, **OrderController** which is the RestController should be created the following services

**Request URL -->/BHO/updateBookQuantity/101/10** :  This service should invoke the **updateBookQuantity**() method of the OrderService class and update the quantity of the book for the given order id.

**Request URL-->/BHO/deleteOrder/101** : This service should invoke the **deleteOrder()**method of theOrderService classand delete the order for the given order id from the list

**Note:**

* The **OrderService** class is provided with two sample orders and added to the list as part of the skeleton. Do not alter the same.
* Partial maven solution for the Same is provided. Do not alter the className/packageName/MethodName.
* You can add new methods/attributes/classes if required.
* The provided **OrderController** should render the above mentioned services.
* In the **OrderService** class , include the required business logic.
* The data returned from the Controller should be a JSON.
* Inject the **OrderService** inside the **OrderController** and invoke the appropriate methods.

[Next activity](https://cognizant.tekstac.com/mod/vpl/view.php?id=20587&forceview=1)**[Online Recipe Management - Exception Handling](https://cognizant.tekstac.com/mod/vpl/view.php?id=20587&forceview=1)**

**Kudos!**You have earned some XP points.