**Github**

For clone/checkout of project please use this link:

<https://github.com/vishwagitbit/TreezProject.git>

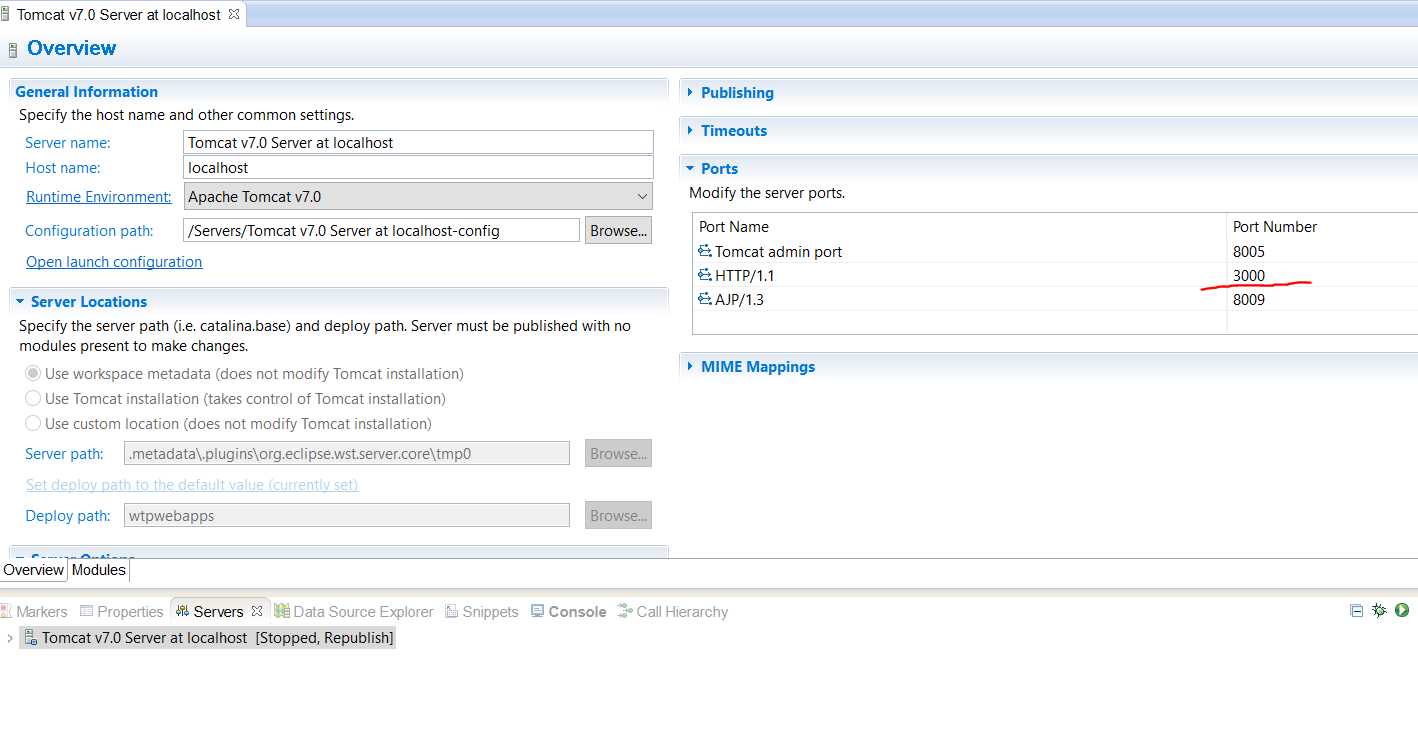
**Java project/code (Eclipse)**

In Eclipse (Kepler), I created a Dynamic web project and used JAX-RS Jersey 2.30 to implement REST APIs, also included the MySql connector jar. JRE is java 1.8.

**Apache Tomcat server**

Used Apache Tomcat server 7.0 for this project in Eclispe.

* Update the Tomcat default port as 3000 in server config file or in Eclipse.

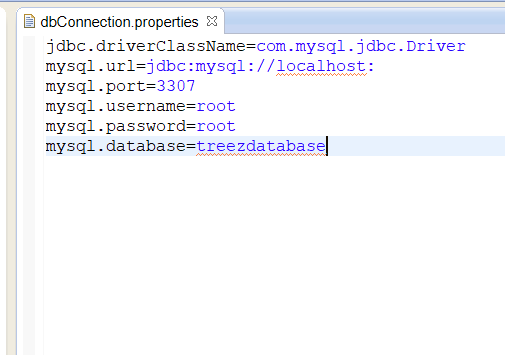


**MySQL DB**

Used MySQL 5.7 but the port number was set as 3307 instead of the default port 3306.

* Username: root
* Password: root
* Port: 3307

This can be changed accordingly in dbConnection.properties file under ‘src’ folder

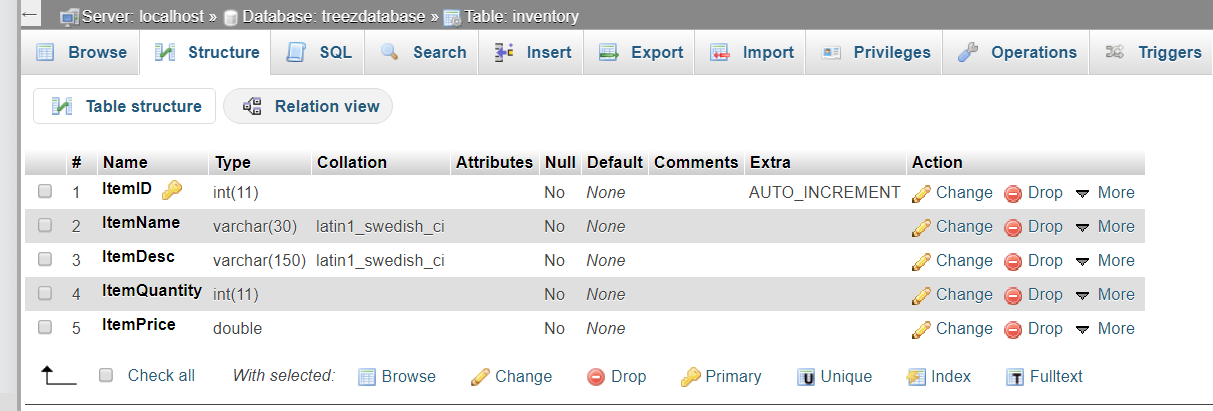


**PhpMyAdmin to create/manage DB tables**

Used PhPMyAdmin to create below table, below are the screenshots

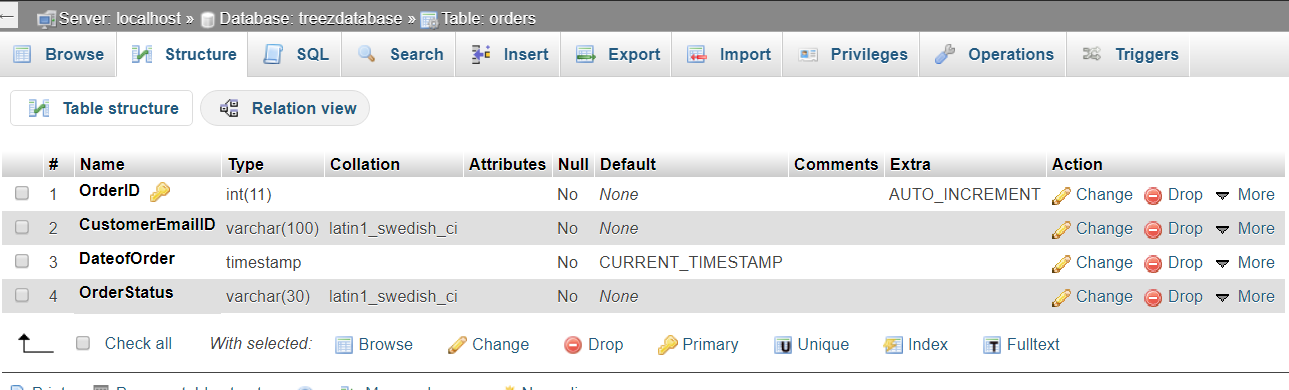
‘inventory’

ItemID is primary key and it is Auto Incremented from 1.



‘orders’

OrderID is primary key and it is Auto Incremented from 1.

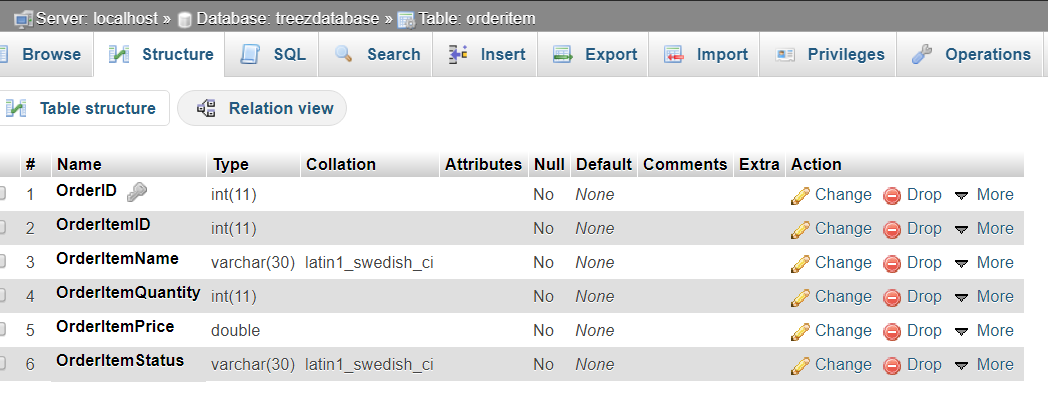


‘orderitem’

There is no primary key but the OrderID is a foreign key from ‘orders’ table.

[ALTER](http://localhost:125/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/alter-table.html) [TABLE](http://localhost:125/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/alter-table.html) `orderitem` ADD FOREIGN KEY (`OrderID`) REFERENCES `orders`(`OrderID`) ON [DELETE](http://localhost:125/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/delete.html) RESTRICT ON [UPDATE](http://localhost:125/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/update.html) RESTRICT;

ItemID and other ‘inventory’ columns are referred by this table during input through queries in code.



Postman – as REST client

To test (consume) the REST call, used Postman with JSON input and XML (or text) output.

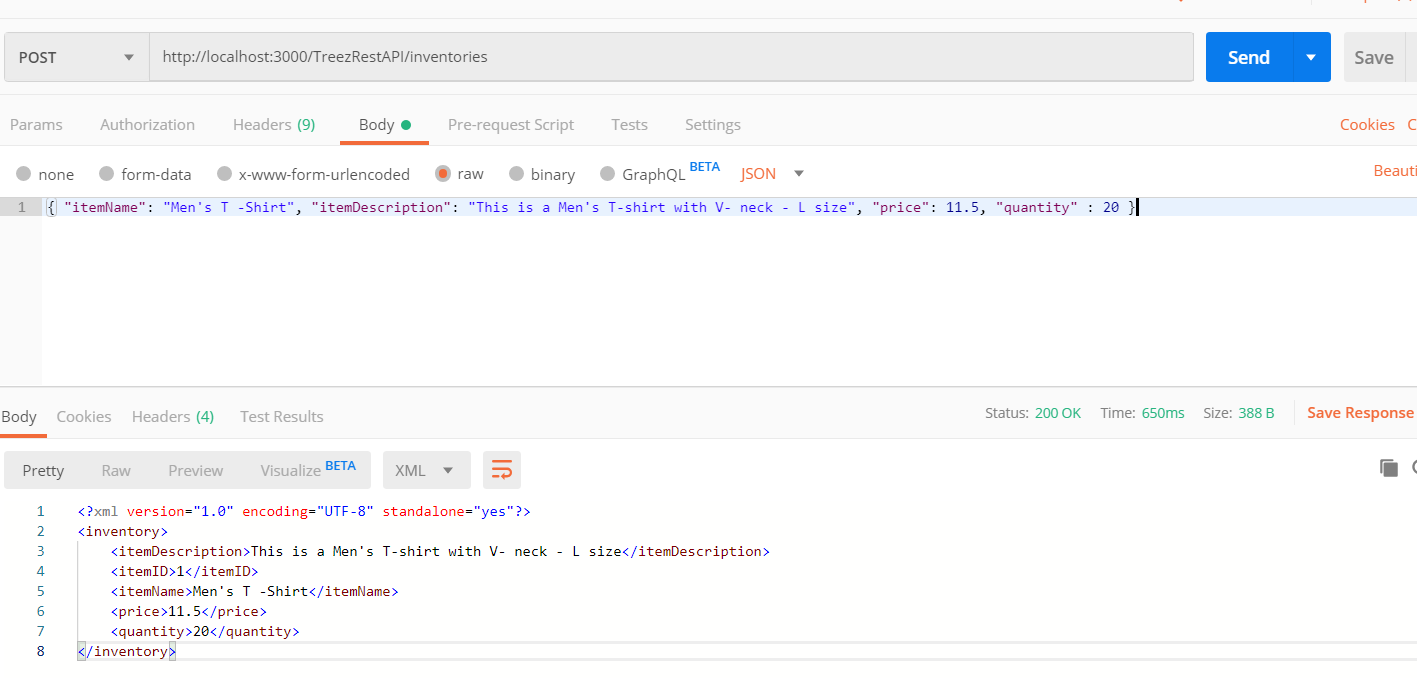
1. Create inventory item

POST <http://localhost:3000/TreezRestAPI/inventories>

**Sample Input format :**

{ "itemName": "Men's T -Shirt", "itemDescription": "This is a Men's T-shirt with V- neck - L size", "price": 11.5, "quantity" : 20 }

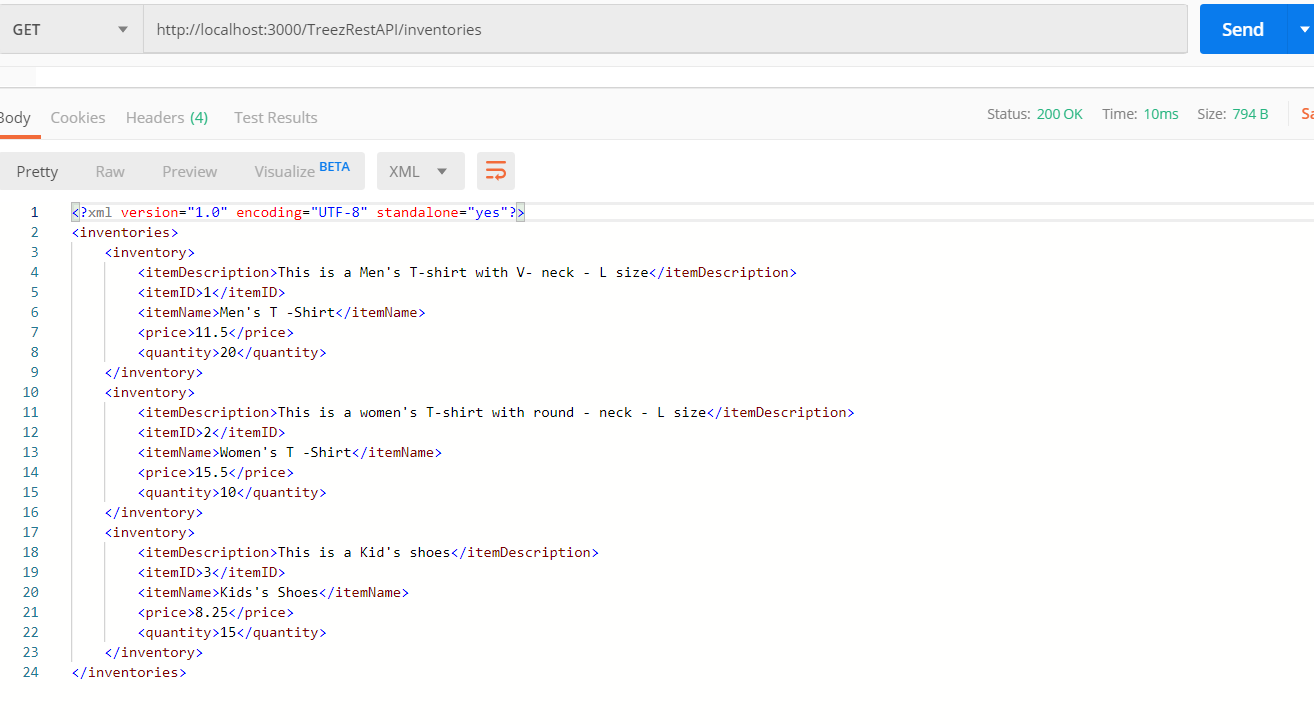
Output:



1. Read all inventory items

GET <http://localhost:3000/TreezRestAPI/inventories>

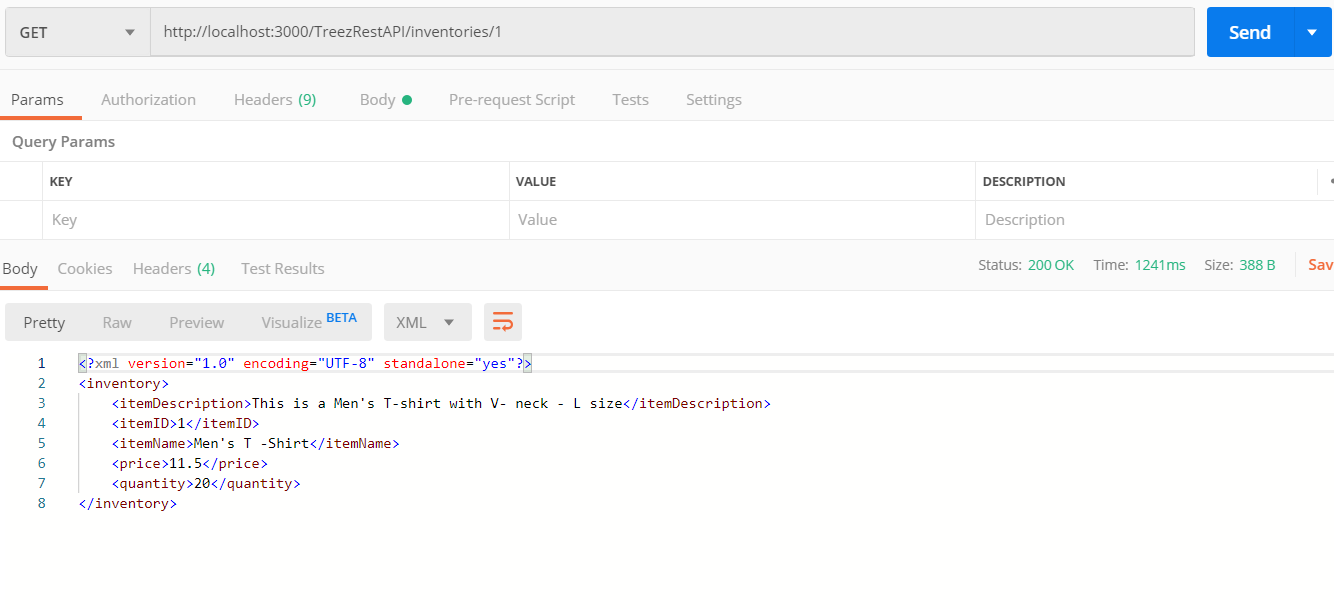
Output:



1. Read single inventory item

GET <http://localhost:3000/TreezRestAPI/inventories/1>

Output:

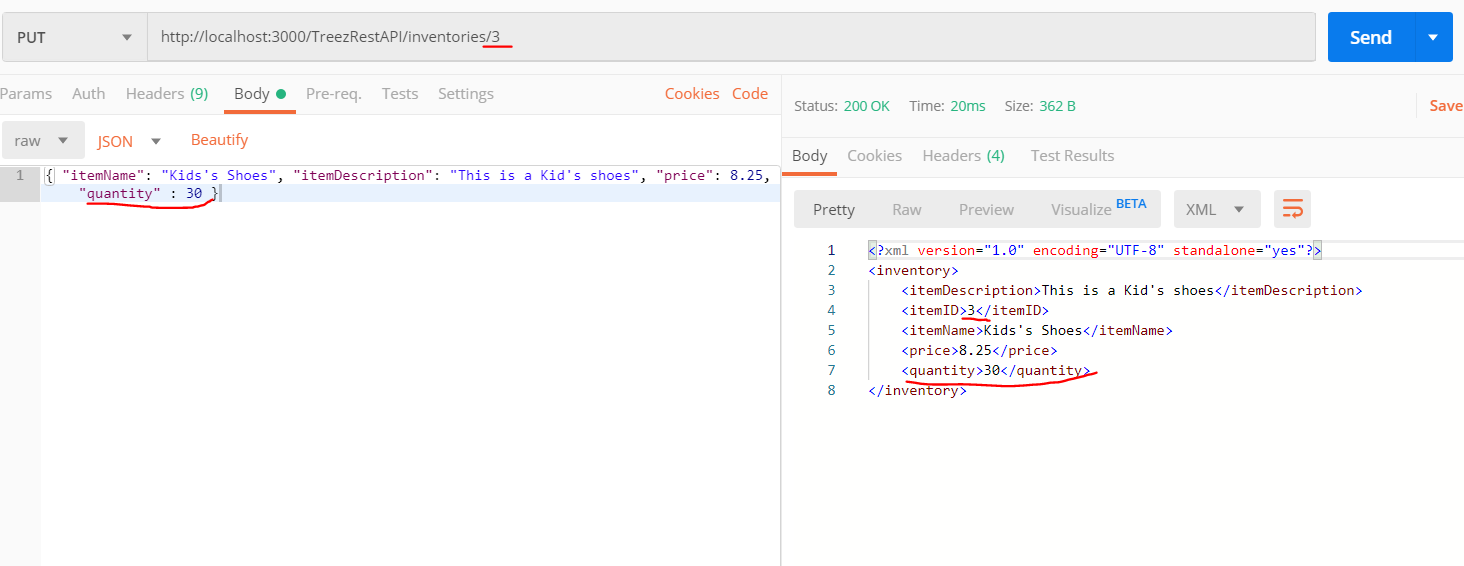


1. Update inventory item

PUT <http://localhost:3000/TreezRestAPI/inventories/1>

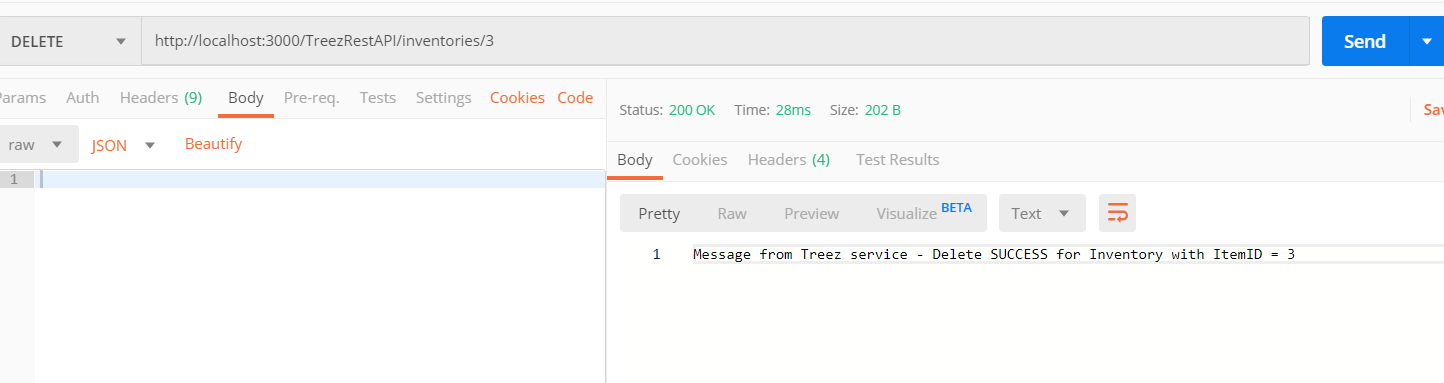
**Input**: can be same as POST with different values in quantity

Output:



1. Delete inventory item

DELETE <http://localhost:3000/TreezRestAPI/inventories/3>



1. Create order

POST <http://localhost:3000/TreezRestAPI/orders>

**Sample Input:**

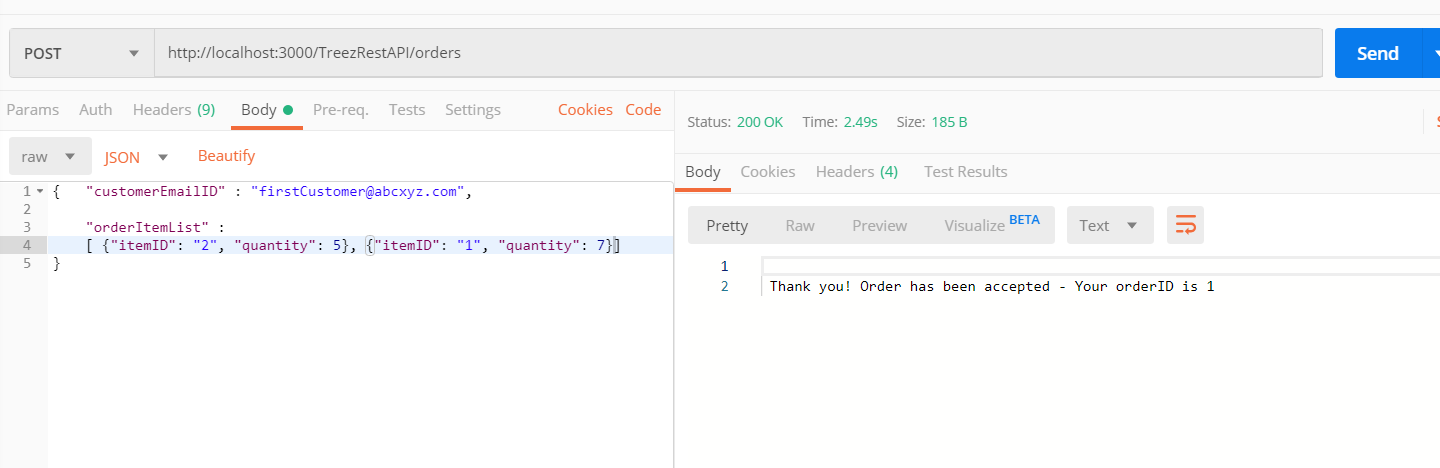
{ "customerEmailID" : "firstCustomer@abcxyz.com",

"orderItemList" :

[ {"itemID": "2", "quantity": 5}, {"itemID": "1", "quantity": 7}]

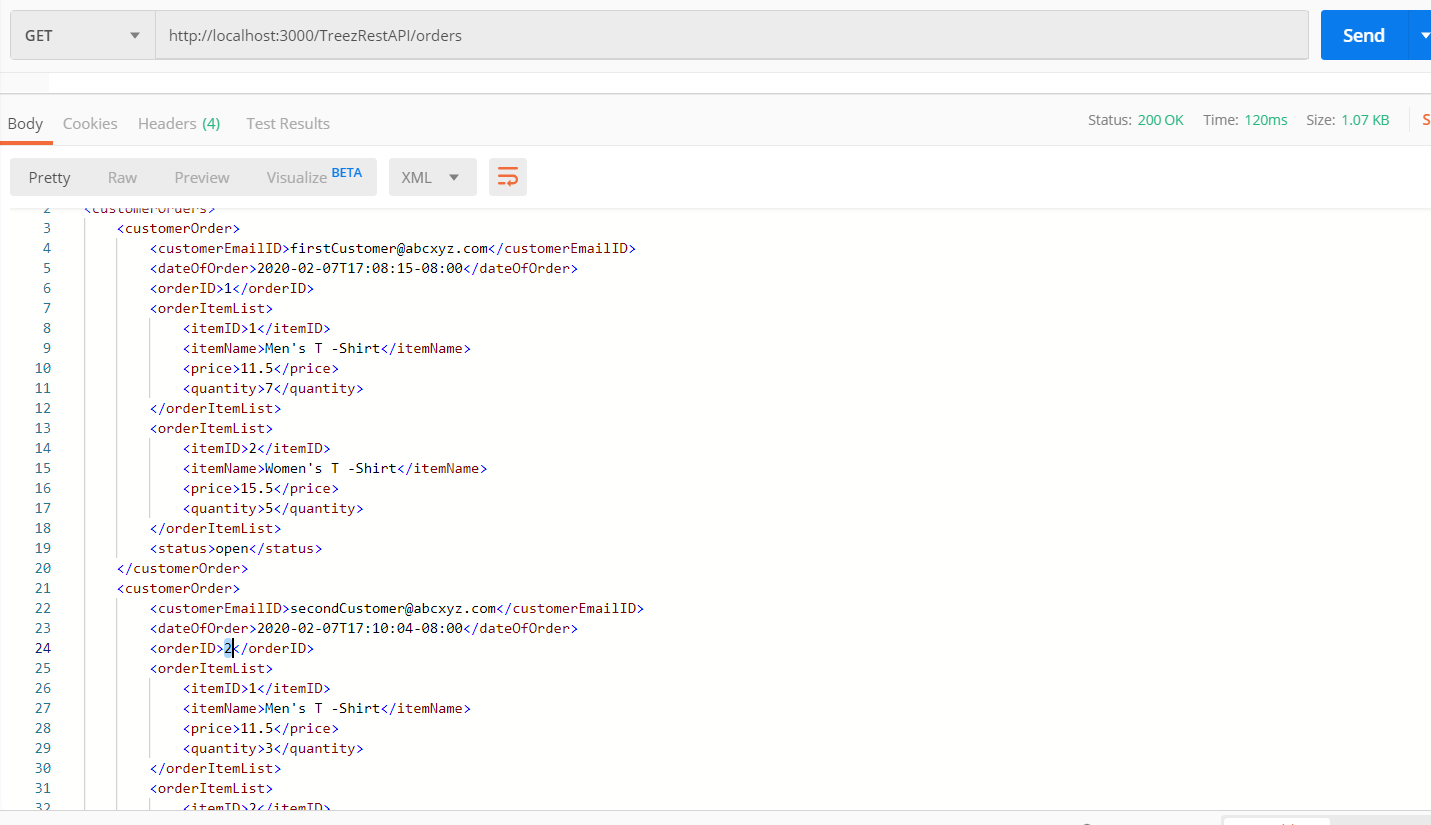
}

Output:



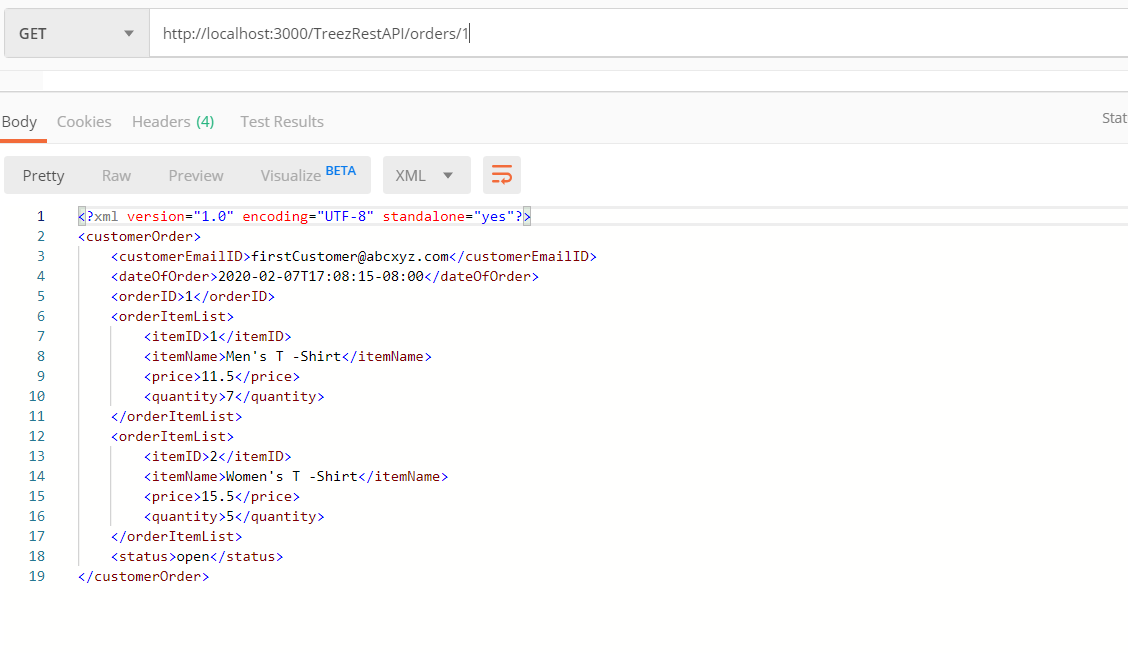
1. Read all orders

GET <http://localhost:3000/TreezRestAPI/orders>



1. Read single order

GET <http://localhost:3000/TreezRestAPI/orders/1>



1. Update order

PUT <http://localhost:3000/TreezRestAPI/orders/1>

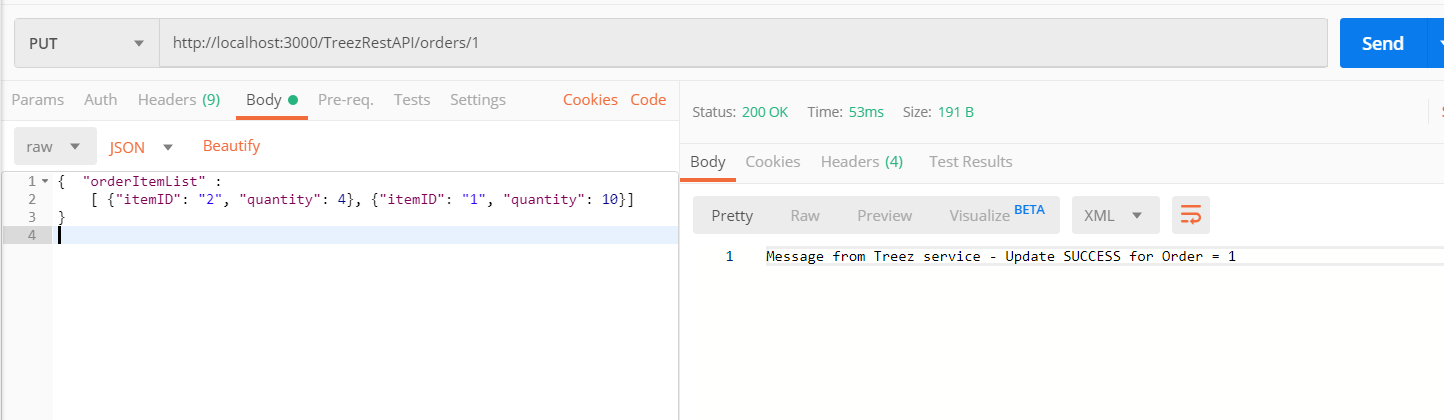
**Sample Input:** It could be same format as CREATE/POST orders input or simply the list of orders (itemID-quantity pair).

{ "orderItemList" :

[ {"itemID": "2", "quantity": 4}, {"itemID": "1", "quantity": 10}]

}

The ‘itemQuantity’ value mentioned in input is total consolidated updated quantity seeking for the existing order. So, the value will remain same if no update in required – this quantity input value agreement is to adhere to PUT idempotent behaviour.



1. Delete order

DELETE <http://localhost:3000/TreezRestAPI/orders/1>

