

# Product Manager API Rest

## Table of Contents

<b>1.2. API REST</b>	<b>2</b>
<b>1.2.1. Create Order</b>	<b>2</b>
<b>1.2.1.1. Method</b>	<b>2</b>
1.2.1.2. Test Case 1	3
1.2.1.3. Test Case 2	4
1.2.1.4. Test Case 3	5
<b>1.2.2. Get orders by Customer</b>	<b>6</b>
<b>1.2.3. Method</b>	<b>6</b>
1.2.3.1. Test Case 1	8
1.2.3.2. Test Case 2	8
1.3. Diagrams	9
1.3.1. ER	9
1.3.2. Component	10
<b>1.2. Web Page</b>	<b>11</b>
1.2.1. Table order	11

## 1.2.API REST

### 1.2.1. Create Order

#### 1.2.1.1. Method

<b>Title</b>	<b>Create order</b>		
<b>URL</b>	<b>/v1/order</b>		
<b>Method</b>	The request type <b>POST</b>	<b>Version</b>	1.0
<b>URL Params</b>	None		
<b>Data Params</b>	<p>URL <a href="http://[host]:[port]/productManager-api-rest/v1/order">http://[host]:[port]/productManager-api-rest/v1/order</a></p> <p>Content</p> <pre>{   "order": {     "deliveryAddress": "15 Queens Park Road, W32 YY, UK",     "customer": {       "customerId": 1     }   },   "orderProducts": [     {       "productId": 1     },     {       "productId": 2     },     {       "productId": 2     },     {       "productId": 3     }   ] }</pre>		
<b>Success Response</b>	<p><b>The order was created.</b></p> <p><b>Code:</b> 200</p> <p><b>Content:</b></p> <pre>{   "description": "OK",   "code": 200 }</pre>		
<b>Error Response</b>	<p>- Validate the maximum products</p> <p><b>Example:</b></p> <p><b>Code:</b> 400 BAD_REQUEST</p> <p><b>Content:</b></p> <pre>{   "description": "The amount of products must be less or equal to 5 ",   "code": 400 }</pre> <p>-Not available products by Customer</p> <p><b>Example:</b></p> <p><b>Code:</b> 400 BAD_REQUEST</p>		

### Content:

```
{
  "description": "The product with id 4 is not available for 'Manny Bharmar'",
  "code": 400
}
```

#### 1.2.1.2. Test Case 1

The screenshot displays a REST client interface with a POST request to `http://ec2-52-14-66-207.us-east-2.compute.amazonaws.com:8080/productManager-api-rest/v1/order`. The request body is a JSON object with the following structure:

```
{
  "order": {
    "deliveryAddress": "Calle false 1234",
    "customer": {
      "customerId": 2
    }
  },
  "orderProducts": [
    {
      "productId": 2
    },
    {
      "productId": 2
    }
  ]
}
```

The response status is `200 OK`. The response body is a JSON object:

```
{
  "description": "OK",
  "code": 200
}
```

### 1.2.1.3. Test Case 2

The screenshot displays a REST client interface with a POST request to the endpoint `http://ec2-52-14-66-207.us-east-2.compute.amazonaws.com:8080/productManager-api-rest/v1/order`. The request body is a JSON object with the following structure:

```
1 {
2   "order": {
3     "deliveryAddress": "Calle false 1234",
4     "customer": {
5       "customerId": 2
6     }
7   },
8   "orderProducts": [
9     {
10      "productId": 2
11    },
12    {
13      "productId": 2
14    },
15    {
16      "productId": 2
17    },
18    {
19      "productId": 2
20    },
21    {
22      "productId": 2
23    },
24    {
25      "productId": 2
26    }
27  ]
28 }
```

The response status is **400 Bad Request**. The response body, shown in the 'Body' tab, is a JSON object:

```
1 {
2   "description": "The amount of products must be less or equal to 5",
3   "code": 400
4 }
```

### 1.2.1.4. Test Case 3

The screenshot displays a REST client interface with the following details:

- Method:** POST
- URL:** `http://ec2-52-14-66-207.us-east-2.compute.amazonaws.com:8080/productManager-api-rest/v1/order`
- Body Type:** raw (selected), with a content type of `JSON (application/json)`.
- Request Body:**

```
1 {  
2   "order": {  
3     "deliveryAddress": "15 Queens Park Road, W32 YYY, UK",  
4     "customer": {  
5       "customerId": 1  
6     },  
7   },  
8   "products": [  
9     {  
10      "productId": 1  
11    },  
12    {  
13      "productId": 2  
14    },  
15    {  
16      "productId": 2  
17    },  
18    {  
19      "productId": 4  
20    }  
21  ]  
22 }  
23 }
```
- Response Status:** 400 Bad Request (Time: 255 ms)
- Response Body:**

```
1 {  
2   "description": "The product with id 4 is not available for 'Manny Bharna'",  
3   "code": 400  
4 }
```

### 1.2.2. Get orders by Customer

#### 1.2.3. Method

Title	Get orders by Customer and dates		
URL	/v1/customer/:id/order		
Method	The request type <b>GET</b>	Version	1.0
URL Params	/v1/customer/{customerId}/order? startDate=[ startDate]& endDate=[ startDate]		
Data Params	<p>{customerId}: Id customer [ startDate] : From date format "yyyy-MM-dd" [ endDate] : End date format "yyyy-MM-dd"</p> <p><b>Example:</b></p> <p><a href="http://[host]:[port]/productManager-api-rest/v1/customer/1/order?startDate=2017-09-27&amp;endDate=2017-10-27">http://[host]:[port]/productManager-api-rest/v1/customer/1/order?startDate=2017-09-27&amp;endDate=2017-10-27</a></p>		
Success Response	<p><b>The order was found.</b></p> <p><b>Code:</b> 200</p> <p><b>Content:</b></p> <pre>{   "customerId": 1,   "email": "manny@Bharma.com",   "name": "Manny Bharma",   "availableProducts": [     {       "productId": 1,       "name": "Product A",       "price": 100.05     },     {       "productId": 2,       "name": "Product B",       "price": 200.1     },     {       "productId": 3,       "name": "Product C",       "price": 300.05     }   ],   "orders": [     {       "orderId": 1,       "deliveryAddress": "15 Queens Park Road, W32 YYY, UK",       "orderTime": "2017-10-26",       "orderDetails": [         {           "orderDetailId": 1,           "price": 200.1,           "productDescription": "2 X Product A"         },         {           "orderDetailId": 2,           "price": 200.1,           "productDescription": "1 X Product B"         }       ]     }   ] }</pre>		

	<pre>    },     {       "orderDetailId": 3,       "price": 300.05,       "productDescription": "1 X Product C"     }   ],   "totalPrice": 700.25 }, {   "orderId": 2,   "deliveryAddress": "15 Queens Park Road, W32 YYY, UK",   "orderTime": "2017-10-26",   "orderDetails": [     {       "orderDetailId": 4,       "price": 100.05,       "productDescription": "1 X Product A"     },     {       "orderDetailId": 5,       "price": 200.1,       "productDescription": "1 X Product B"     },     {       "orderDetailId": 6,       "price": 600.1,       "productDescription": "2 X Product C"     }   ],   "totalPrice": 900.25 } ]</pre>
Error Response	<p>- Order not found</p> <p><b>Example:</b></p> <p><b>Code:</b> 404 Not Found</p> <p><b>Content:</b></p> <pre>{   "customerId": 0,   "email": null,   "name": null,   "availableProducts": null,   "orders": [] }</pre>

### 1.2.3.1. Test Case 1

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** `http://ec2-52-14-66-207.us-east-2.compute.amazonaws.com:8080/productManager-api-rest/v1/customer/1/order?startDate=2017-09-27&endDate=2017-10-27`
- Authorization:** No Auth
- Status:** 200 OK
- Body:** JSON (Pretty view)

```
1 {
2   "customerId": 1,
3   "email": "nanny@bharma.com",
4   "name": "Nanny Bharma",
5   "availableProducts": [{}],
22  "orders": [
23    {
46      {
69        {
87          "orderId": 6,
88          "deliveryAddress": "15 Queens Park Road, W32 YYY, UK",
89          "orderLine": "2017-10-27",
90          "orderDetails": [
91            {
92              "orderDetailId": 11,
93              "price": 100.05,
94              "productDescription": "1 X Product A"
95            },
96            {
97              "orderDetailId": 12,
98              "price": 400.2,
99              "productDescription": "2 X Product B"
100            },
101            {
102              "orderDetailId": 13,
103              "price": 300.05,
104              "productDescription": "1 X Product C"
105            }
106          ],
107          "totalPrice": 800.3
108        }
109      ]
110    }
111  }
```

### 1.2.3.2. Test Case 2

The screenshot shows a REST client interface with the following details:

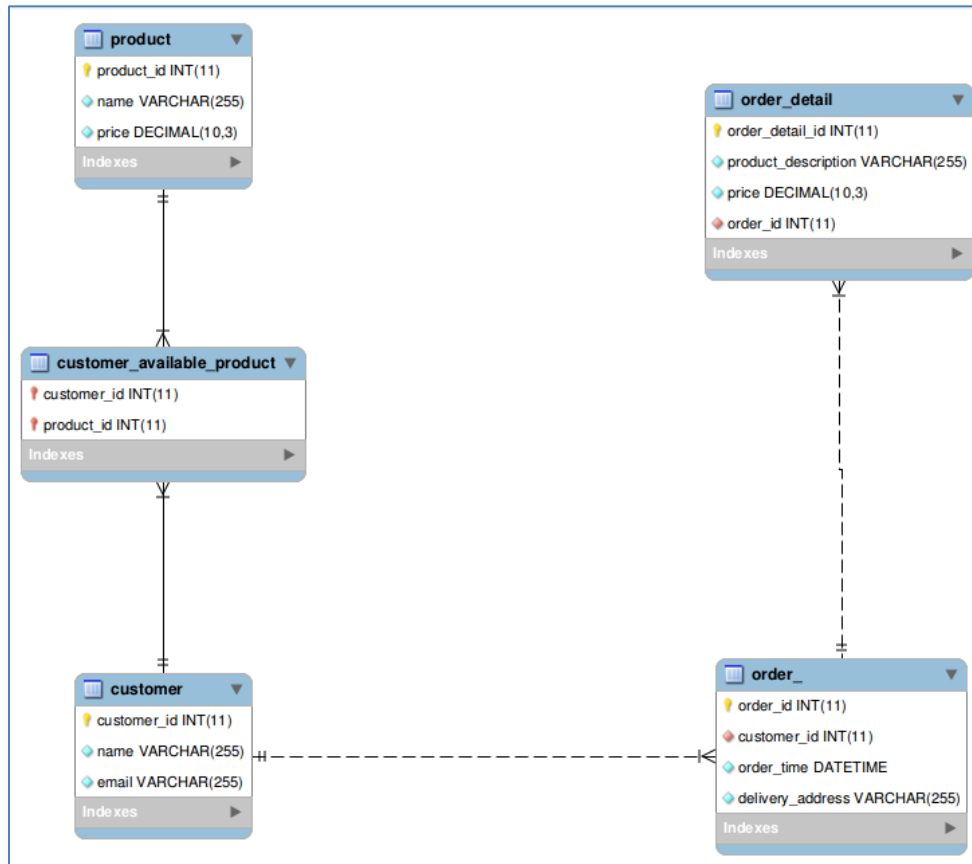
- Method:** GET
- URL:** `http://ec2-52-14-66-207.us-east-2.compute.amazonaws.com:8080/productManager-api-rest/v1/customer/4/order?startDate=2017-09-27&endDate=2017-10-27`
- Authorization:** No Auth
- Status:** 404 Not Found
- Body:** JSON (Pretty view)

```
1 {
2   "customerId": 0,
3   "email": null,
4   "name": null,
5   "availableProducts": null,
6   "orders": []
7 }
```

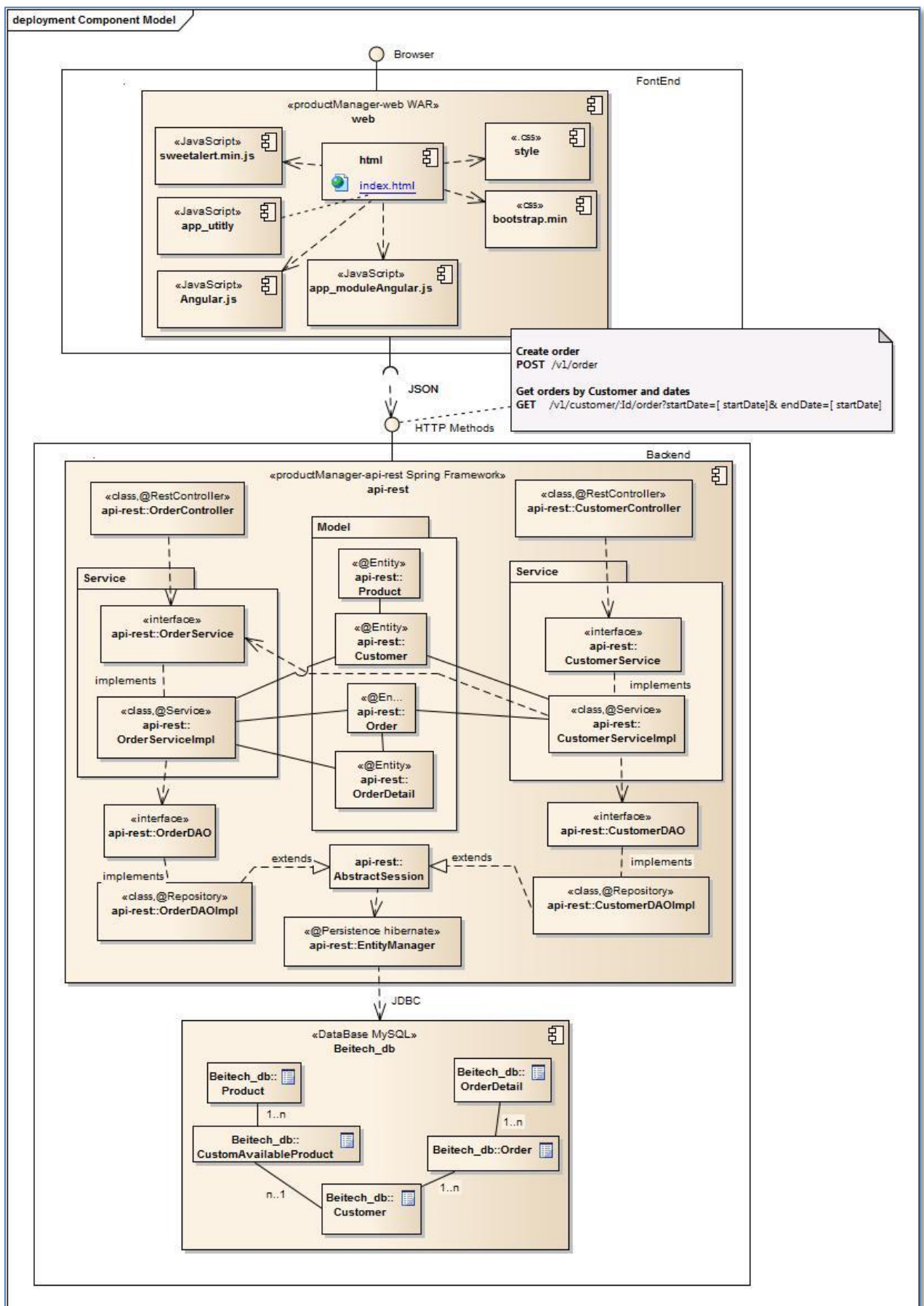


### 1.3. Diagrams

#### 1.3.1. ER




### 1.3.2. Component




## 1.2. Web Page

### 1.2.1. Table order



## Product Manager

Customer: --Select one customer--



**Oops, 'Customer name is required**

'Insufficient Data! Please provide values for Customer name!

OK

Creation Date				Products
2017-10-26				1 X Product A : \$ 200.1 1 X Product B : \$ 200.1 1 X Product C : \$ 300.05
2017-10-26				1 X Product A : \$ 100.05 1 X Product B : \$ 200.1 1 X Product C : \$ 600.1
2017-10-27	5	\$ 300.15	15 Queens Park Road, W32 YYY, UK	1 X Product A : \$ 100.05 1 X Product B : \$ 200.1
2017-10-27	6	\$ 800.3	15 Queens Park Road, W32 YYY, UK	1 X Product A : \$ 100.05 2 X Product B : \$ 400.2 1 X Product C : \$ 300.05

## Product Manager

Customer: Mike Simm



Oops, 'Data not found

'Orders not found!

OK

Creation Date

Products

## Product Manager

Customer: Manny Bharna

Search

### Order List

2017-09-27 - 2017-10-27

Creation Date	Order Id	Total \$	Delivery Address	Products
2017-10-26	1	\$ 700.25	15 Queens Park Road, W32 YYY, UK	2 X Product A : \$ 200.1 1 X Product B : \$ 200.1 1 X Product C : \$ 300.05
2017-10-26	2	\$ 900.25	15 Queens Park Road, W32 YYY, UK	1 X Product A : \$ 100.05 1 X Product B : \$ 200.1 2 X Product C : \$ 600.1
2017-10-27	5	\$ 300.15	15 Queens Park Road, W32 YYY, UK	1 X Product A : \$ 100.05 1 X Product B : \$ 200.1
2017-10-27	6	\$ 800.3	15 Queens Park Road, W32 YYY, UK	1 X Product A : \$ 100.05 2 X Product B : \$ 400.2 1 X Product C : \$ 300.05

