

POST (crear producto)

The screenshot shows the Postman application interface. On the left, there's a sidebar with collections, environments, history, flows, and files. The main area displays a POST request for 'createProduct' under the 'API_NodeJS_TP3' collection. The request URL is `http://localhost:3000/api/products/`. The 'Body' tab is selected, showing a JSON payload:

```
1 {
2   "name": "Smart TV Android 70",
3   "price": 850,
4   "stock": 4,
5   "description": "Impresionante Smart TV con Android."
6 }
```

Below the request, the response is shown in a JSON format:

```
1 {
2   "ok": true,
3   "status": 201,
4   "message": "PRODUCTO CREADO",
5   "data": {
6     "name": "Smart TV Android 70",
7     "price": 850,
8     "stock": 4,
9     "description": "Impresionante Smart TV con Android.",
10    "_id": "697elba0a739cf8569408800",
11    "__v": 0
12  }
13 }
```

GET (obtener productos)

The screenshot shows the Postman application interface. On the left, there's a sidebar with collections, environments, history, flows, and files. The main area displays a GET request for 'createProduct' under the 'API_NodeJS_TP3' collection. The request URL is `http://localhost:3000/api/products/`. The 'Body' tab is selected, showing a JSON payload:

```
1 {
2   "ok": true,
3   "status": 201,
4   "message": "PRODUCTO CREADO",
5   "data": {
6     "name": "Smart TV Android 70",
7     "price": 850,
8     "stock": 4,
9     "description": "Impresionante Smart TV con Android.",
10    "_id": "697elba0a739cf8569408800",
11    "__v": 0
12  }
13 }
```

Below the request, the response is shown in a JSON format:

```
1 [
2   {
3     "_id": "697elc35a739cf8569408804",
4     "name": "Teléfono celular BlackBerry 8520",
5     "price": 255,
6     "stock": 8,
7     "description": "En caja cerrada con todos sus accesorios.",
8     "__v": 0
9   },
10  {
11    "_id": "697elc35a739cf8569408804",
12    "name": "iPhone 16 Pro 256GB Silver",
13    "price": 890,
14    "stock": 3,
15    "description": "Reacondicionado, con 85% de batería.",
16    "__v": 0
17  },
18  {
19    "_id": "697elc5da739cf8569408806",
20    "name": "Notebook Lenovo P16s Gen 4 AMD",
21    "price": 1450,
22    "stock": 2,
23    "description": "Producto nuevo, en caja cerrada directo de fábrica.",
24    "__v": 0
25  }
26 ]
27 
```

PUT (actualizar producto)

The screenshot shows the Postman application interface. On the left, there's a sidebar with collections, environments, and flows. The main area shows a collection named 'NodeJS_TP2' with a sub-collection 'API_NodeJS_TP3'. Under 'API_NodeJS_TP3', there is a GET endpoint named 'listarProductos'. A PUT request is being made to the URL `http://localhost:3000/api/products/697e1c07a739cf8569408804`. The 'Body' tab is selected, showing the following JSON payload:

```
1 {
2   "name": "iPhone 16 Pro 256GB COLOR GRIS",
3   "price": 885,
4   "stock": 3,
5   "description": "Reacondicionado, con 85% de bateria."
6 }
```

Below the body, the response is shown in JSON format:

```
1 {
2   "ok": true,
3   "status": 200,
4   "message": "PRODUCTO ACTUALIZADO",
5   "data": [
6     {
7       "_id": "697e1c07a739cf8569408804",
8       "name": "iPhone 16 Pro 256GB COLOR GRIS",
9       "price": 885,
10      "stock": 3,
11      "description": "Reacondicionado, con 85% de bateria.",
12      "__v": 0
13   }
14 }
```

DELETE (eliminar producto)

The screenshot shows the Postman application interface. The sidebar and collection structure are identical to the previous screenshot. A DELETE request is being made to the URL `http://localhost:3000/api/products/_id`. The 'Params' tab is selected, showing a key 'id' with the value '697e1c07a739cf8569408802'. Below the body, the response is shown in JSON format:

```
1 {
2   "ok": true,
3   "status": 200,
4   "message": "PRODUCTO ELIMINADO",
5   "data": [
6     {
7       "_id": "697e1c07a739cf8569408802",
8       "name": "Teléfono celular BlackBerry 8520",
9       "price": 255,
10      "stock": 8,
11      "description": "En caja cerrada con todos sus accesorios.",
12      "__v": 0
13   }
14 }
```