

## Post create:

The screenshot shows the Postman application interface. On the left, the sidebar displays collections like 'test-api', 'Lab Web App Dev', and 'Lab08'. Under 'Lab08', several requests are listed: 'GET all customers', 'GET by ID', 'POST POST create' (which is selected), 'PUT PUT update', 'DELETE DELETE', 'POST Validation error', 'GET Not found error', 'POST Duplicate error', 'GET Search results', and 'GET Paginated results'. The main workspace shows a POST request to 'http://localhost:8080/api/customers'. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2   "customerCode": "C001",  
3   "fullName": "John Doe",  
4   "email": "john@example.com",  
5   "phone": "+1555123499",  
6   "address": "123 Street",  
7   "status": "ACTIVE"  
8 }  
9  
10 }
```

Below the body, the response is shown as a 201 Created status with a timestamp: '2025-12-08T20:56:37.0784123'. The bottom navigation bar includes icons for Cloud View, Find and replace, Console, Terminal, Runner, Start Proxy, Cookies, Vault, Trash, and Help.

## GET all

The screenshot shows the Postman application interface. The left sidebar displays collections, environments, and flows. The main workspace shows a collection named "test-api" under "Lab Web App Dev". A specific API endpoint, "GET all customers", is selected. The request URL is set to "http://localhost:8080/api/customers". The "Body" tab is selected, showing the response body as JSON. The response status is "200 OK" with a duration of "15 ms" and a size of "502 B". The response content is as follows:

```
1 {  
2   "totalItems": 1,  
3   "totalPages": 1,  
4   "customers": [  
5     {  
6       "id": 4,  
7       "customerCode": "C001",  
8       "fullName": "John Doe",  
9       "email": "john@example.com",  
10      "phone": "+1555123499",  
11      "address": "123 Street",  
12      "status": "ACTIVE",  
13      "createdAt": "2025-12-08T20:56:37",  
14      "links": []  
15    },  
16  ],  
17  "currentPage": 0  
18 }
```

## GET by ID

The screenshot shows the API Network tool interface. On the left, the sidebar displays the project 'test-api' and collections like 'Lab Web App Dev' and 'Lab08'. In the main area, a collection named 'Lab08' is selected, and a specific endpoint 'GET by ID' is highlighted. The URL is set to 'http://localhost:8080/api/customers/4'. The response status is '200 OK' with a response time of 89 ms and a size of 616 B. The response body is a JSON object representing a customer:

```
1 {  
2   "id": 4,  
3   "customerCode": "C001",  
4   "fullName": "John Doe",  
5   "email": "john@example.com",  
6   "phone": "+1555123499",  
7   "address": "123 Street",  
8   "status": "ACTIVE",  
9   "createdAt": "2025-12-08T20:56:37",  
10  "_links": {  
11    "self": {  
12      "href": "http://localhost:8080/api/customers/4"  
13    },  
14    "all-customers": {  
15      "href": "http://localhost:8080/api/customers?page=0&size=10&sortDir=asc&sortBy=id",  
16      "templated": true  
17    }  
18  }  
19 }
```

Below the body, there are tabs for 'Docs', 'Params', 'Auth', 'Headers (6)', 'Body', 'Scripts', 'Settings', and 'Cookies'. The 'Params' tab is active, showing an empty table for query parameters. The 'Body' tab shows the JSON response. At the bottom, there are various navigation and utility buttons.

## PUT update

The screenshot shows the Postman application interface. The left sidebar displays a workspace named "test-api" containing collections like "Lab Web App Dev" and "Lab08". Under "Lab08", there are several requests: "GET all customers", "GET by ID", "POST POST create", "PUT PUT update" (which is selected), "DEL DELETE", "POST Validation error", "GET Not found error", "POST Duplicate error", "GET Search results", and "GET Paginated results". The main panel shows a "PUT" request to "http://localhost:8080/api/customers/4". The "Body" tab is selected, showing a JSON payload:

```
1 {  
2     "customerCode": "C001",  
3     "fullName": "John Updated",  
4     "email": "john.updated@example.com",  
5     "phone": "+1555999966",  
6     "address": "New Address",  
7     "status": "ACTIVE"  
8 }  
9  
10 }
```

Below the request, the response is shown as a 200 OK status with a timestamp of "2025-12-08T20:56:37". The response body is identical to the request body.

## GET all (paginated)

The screenshot shows the Postman application interface. The left sidebar displays collections like 'test-api', 'Lab Web App Dev', and 'Lab08'. Under 'Lab08', several endpoints are listed: 'GET all customers', 'GET by ID', 'POST POST create', 'PUT PUT update', 'DEL DELETE', 'POST Validation error', 'GET Not found error', 'POST Duplicate error', 'GET Search results', and 'GET Paginated results'. The 'GET Paginated results' endpoint is selected. The main panel shows an 'HTTP' request to 'http://localhost:8080/api/customers?page=0&size=2&sortBy=fullName'. The 'Params' tab is active, showing query parameters: page (0), size (2), sortBy (fullName), and sortDir (asc). The 'Body' tab shows a JSON response with 18 numbered lines. The response object has properties: totalItems (1), totalPages (1), and customers (an array containing one customer object). The customer object includes fields: id (4), customerCode (C001), fullName (John Updated), email (john.updated@example.com), phone (+1555999966), address (New Address), status (ACTIVE), createdAt (2025-12-08T20:56:37), and links (empty array). The 'currentPage' is 0.

```
1 {  
2   "totalItems": 1,  
3   "totalPages": 1,  
4   "customers": [  
5     {  
6       "id": 4,  
7       "customerCode": "C001",  
8       "fullName": "John Updated",  
9       "email": "john.updated@example.com",  
10      "phone": "+1555999966",  
11      "address": "New Address",  
12      "status": "ACTIVE",  
13      "createdAt": "2025-12-08T20:56:37",  
14      "links": []  
15     }  
16   ],  
17   "currentPage": 0  
18 }
```

## Search Keyword

The screenshot shows the Postman application interface. On the left, the sidebar displays collections, environments, history, and flows. The main workspace shows a collection named "test-api" under "Lab Web App Dev". A specific request titled "GET Search results" is selected. The request URL is `http://localhost:8080/api/customers/search?keyword=john`. The "Params" tab is active, showing a parameter named "keyword" with the value "john". The "Body" tab shows a JSON response with the following content:

```
1 [  
2 {  
3     "id": 4,  
4     "customerCode": "C001",  
5     "fullName": "John Updated",  
6     "email": "john.updated@example.com",  
7     "phone": "+15559999966",  
8     "address": "New Address",  
9     "status": "ACTIVE",  
10    "createdAt": "2025-12-08T20:56:37",  
11    "links": []  
12 }]  
13 ]
```

The response status is 200 OK with a duration of 20 ms and a size of 455 B. Below the body, there are tabs for Preview and Visualize, and a toolbar with various icons.

## Validation error

The screenshot shows the Postman application interface. On the left, the sidebar displays collections, environments, and flows. The main area shows a collection named "test-api" with a sub-collection "Lab08". Under "Lab08", a POST request named "Validation error" is selected. The request URL is "http://localhost:8080/api/customers". The request body is set to JSON and contains the following data:

```
1 {  
2     "customerCode": "C002",  
3     "fullName": "Bad Email",  
4     "email": "not-an-email",  
5     "phone": "+15550000000",  
6     "address": "Somewhere",  
7     "status": "ACTIVE"  
8 }  
9
```

Below the request, the response is shown as a 400 Bad Request. The response body is also JSON and includes the following details:

```
1 {  
2     "timestamp": "2025-12-08T21:11:16.9558747Z",  
3     "status": 400,  
4     "error": "Bad Request",  
5     "message": "Validation failed",  
6     "path": "/api/customers",  
7     "details": [  
8         "email: Email must be a valid email address"  
9     ]  
10 }
```

## Duplicate error

The screenshot shows the Postman application interface. The left sidebar displays the 'test-api' workspace with collections like 'Lab Web App Dev' and 'Lab08'. Under 'Lab08', several requests are listed, including 'POST Duplicate error' which is currently selected. The main panel shows a POST request to `http://localhost:8080/api/customers`. The request body is set to JSON and contains the following data:

```
1 {  
2   "customerCode": "C001",  
3   "fullName": "Dup Code",  
4   "email": "john@example.com",  
5   "phone": "+1555222222",  
6   "address": "Dup Addr",  
7   "status": "ACTIVE"  
8 }
```

The response status is **409 Conflict**, with a timestamp of `2025-12-08T21:11:52.7934758`, status `409`, error `Conflict`, message `Customer code already exists`, path `/api/customers`, and details `null`.

## Not Found

The screenshot shows the API Network tool interface. On the left, the sidebar displays collections like 'test-api' and environments like 'Lab Web App Dev' and 'Lab08'. In the main area, under 'Lab08', a 'GET Not found error' endpoint is selected. The URL is set to `http://localhost:8080/api/customers/9999999`. The 'Params' tab is active, showing a single parameter 'Key' with a value of 'Value'. The 'Body' tab shows a JSON response:

```
1 {  
2   "timestamp": "2025-12-08T21:12:17.3898199",  
3   "status": 404,  
4   "error": "Not Found",  
5   "message": "Customer not found with id: 9999999",  
6   "path": "/api/customers/9999999",  
7   "details": null  
8 }
```

The status bar at the bottom includes links for 'Cloud View', 'Find and replace', 'Console', 'Terminal', 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and a help icon.

## DELETE

The screenshot shows the Postman application interface. On the left, the sidebar displays collections like 'test-api' and environments. The main area shows a collection named 'Lab Web App Dev' with a sub-collection 'Lab08'. Under 'Lab08', there are several requests: 'GET all customers', 'GET by ID', 'POST POST create', 'PUT PUT update', and the selected 'DEL DELETE'. Below these are error handlers: 'POST Validation error', 'GET Not found error', 'POST Duplicate error', 'GET Search results', and 'GET Paginated results'. The 'DELETE' request is selected, and its details are shown in the center panel. The URL is set to `http://localhost:8080/api/customers/4`. The 'Params' tab is active, showing a table with one row: 'Key' and 'Value'. The 'Body' tab shows a JSON response with the message "Customer deleted successfully". The status bar at the bottom indicates a 200 OK response with 15 ms latency and 296 B size.

HTTP Lab08 / **DELETE**

**DELETE** `http://localhost:8080/api/customers/4`

Docs Params Auth Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body

200 OK • 15 ms • 296 B • Save Response

```
{ } JSON
```

```
1 {  
2   "message": "Customer deleted successfully"  
3 }
```

Body Preview Visualize

Cloud View Find and replace Console Terminal Runner Start Proxy Cookies Vault Trash ?