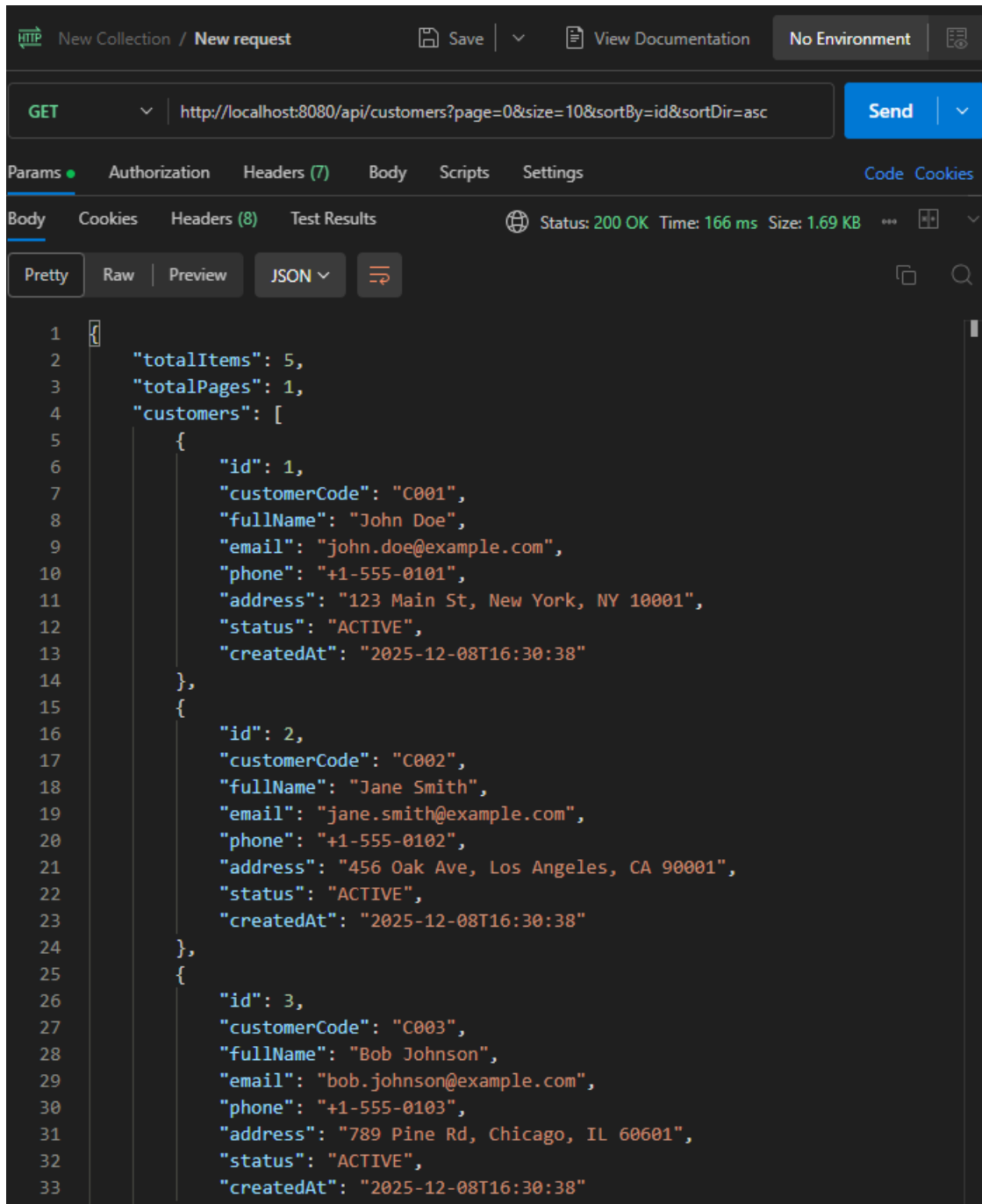


A1 — GET all customers (200) + Pagination/Sorting

This request retrieves the customer list using pagination and sorting parameters. The response returns the first page of customers sorted by id in ascending order, including metadata like totalItems and totalPages.



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

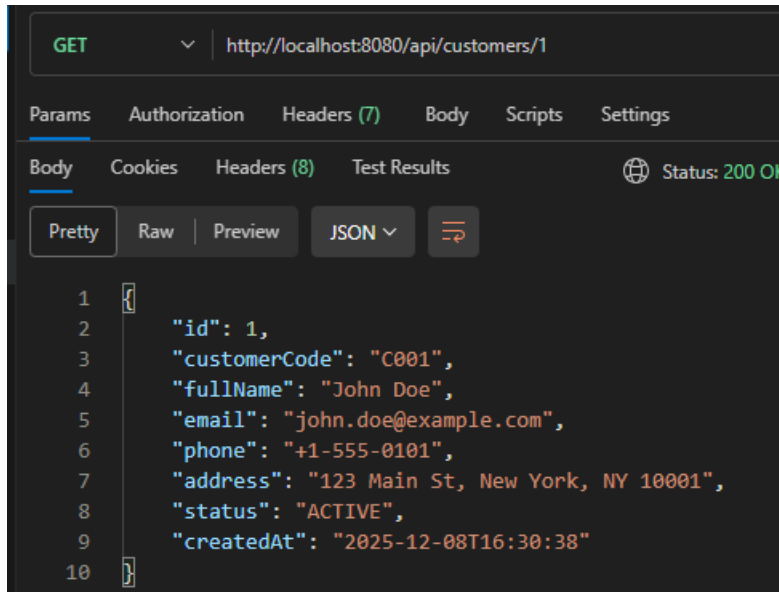
- Method:** GET
- URL:** `http://localhost:8080/api/customers?page=0&size=10&sortBy=id&sortDir=asc`
- Status:** 200 OK
- Time:** 166 ms
- Size:** 1.69 KB

The response body is displayed in JSON format, showing metadata and a list of customers:

```
1 {
2   "totalItems": 5,
3   "totalPages": 1,
4   "customers": [
5     {
6       "id": 1,
7       "customerCode": "C001",
8       "fullName": "John Doe",
9       "email": "john.doe@example.com",
10      "phone": "+1-555-0101",
11      "address": "123 Main St, New York, NY 10001",
12      "status": "ACTIVE",
13      "createdAt": "2025-12-08T16:30:38"
14    },
15    {
16      "id": 2,
17      "customerCode": "C002",
18      "fullName": "Jane Smith",
19      "email": "jane.smith@example.com",
20      "phone": "+1-555-0102",
21      "address": "456 Oak Ave, Los Angeles, CA 90001",
22      "status": "ACTIVE",
23      "createdAt": "2025-12-08T16:30:38"
24    },
25    {
26      "id": 3,
27      "customerCode": "C003",
28      "fullName": "Bob Johnson",
29      "email": "bob.johnson@example.com",
30      "phone": "+1-555-0103",
31      "address": "789 Pine Rd, Chicago, IL 60601",
32      "status": "ACTIVE",
33      "createdAt": "2025-12-08T16:30:38"
34    }
35  ]
36 }
```

B1 — GET customer by ID (200)

This request fetches a single customer by a specific ID. The response confirms the customer exists and returns that customer's details (e.g., customerCode C001).



C1 — Search customers by keyword (200)

This request searches customers using a keyword across name, email, and customer code. The response returns a list of matching customers that contain the keyword (e.g., “john”).

```
GET http://localhost:8080/api/customers/search?keyword=john
Status: 200 OK Time: 158 ms Size: 766 B

{
  "id": 1,
  "customerCode": "C001",
  "fullName": "John Doe",
  "email": "john.doe@example.com",
  "phone": "+1-555-0101",
  "address": "123 Main St, New York, NY 10001",
  "status": "ACTIVE",
  "createdAt": "2025-12-08T16:30:38"
},
{
  "id": 3,
  "customerCode": "C003",
  "fullName": "Bob Johnson",
  "email": "bob.johnson@example.com",
  "phone": "+1-555-0103",
  "address": "789 Pine Rd, Chicago, IL 60601",
  "status": "ACTIVE",
  "createdAt": "2025-12-08T16:30:38"
}
```

D1 — Filter customers by status ACTIVE (200)

This request filters customers by status ACTIVE. The response returns only customers whose status is ACTIVE.

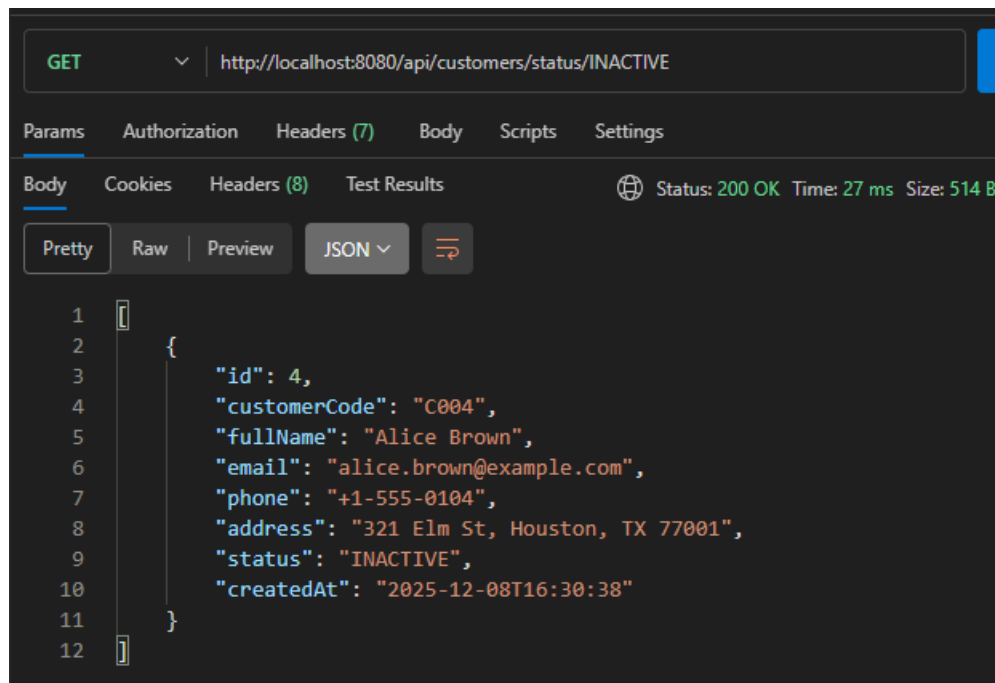
```
GET http://localhost:8080/api/customers/status/ACTIVE
Send

Params Authorization Headers (7) Body Scripts Settings Code Cookies
Body Cookies Headers (8) Test Results Status: 200 OK Time: 214 ms Size: 1.26 KB
Pretty Raw Preview JSON

[
  {
    "email": "john.doe@example.com",
    "phone": "+1-555-0101",
    "address": "123 Main St, New York, NY 10001",
    "status": "ACTIVE",
    "createdAt": "2025-12-08T16:30:38"
  },
  {
    "id": 2,
    "customerCode": "C002",
    "fullName": "Jane Smith",
    "email": "jane.smith@example.com",
    "phone": "+1-555-0102",
    "address": "456 Oak Ave, Los Angeles, CA 90001",
    "status": "ACTIVE",
    "createdAt": "2025-12-08T16:30:38"
  },
  {
    "id": 3,
    "customerCode": "C003",
    "fullName": "Bob Johnson",
    "email": "bob.johnson@example.com",
    "phone": "+1-555-0103",
    "address": "789 Pine Rd, Chicago, IL 60601",
    "status": "ACTIVE",
    "createdAt": "2025-12-08T16:30:38"
  },
  {
    "id": 5,
    "customerCode": "C005",
    "fullName": "Charlie Wilson",
    "email": "charlie.wilson@example.com",
    "phone": "+1-555-0105",
    "address": "654 Maple Dr, Phoenix, AZ 85001",
    "status": "ACTIVE",
    "createdAt": "2025-12-08T16:30:38"
  }
]
```

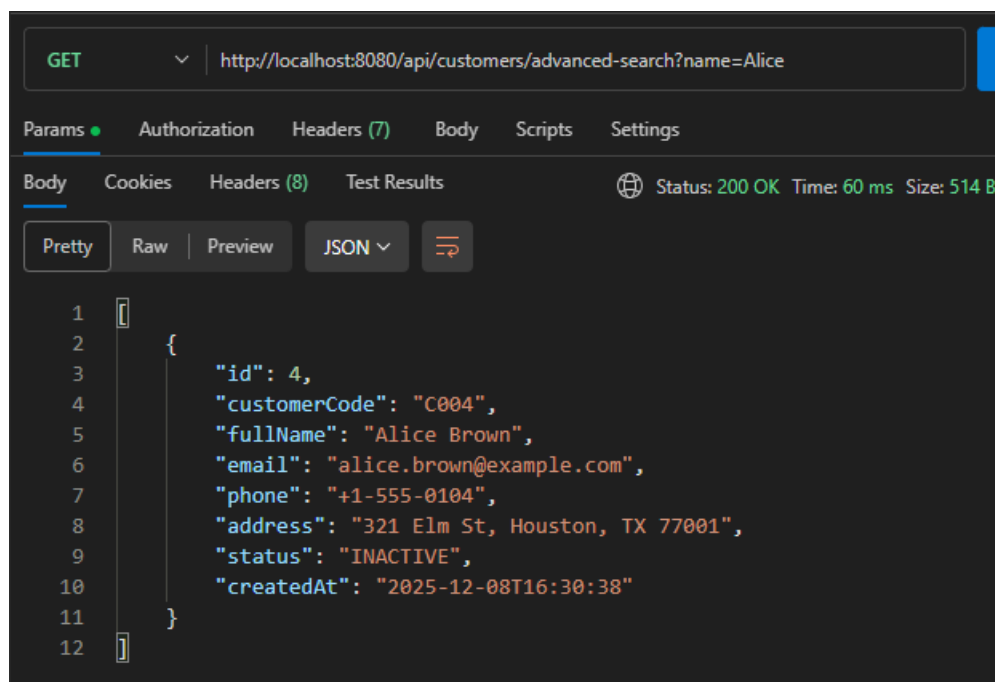
D2 — Filter customers by status INACTIVE (200)

This request filters customers by status INACTIVE. The response returns only customers whose status is INACTIVE.



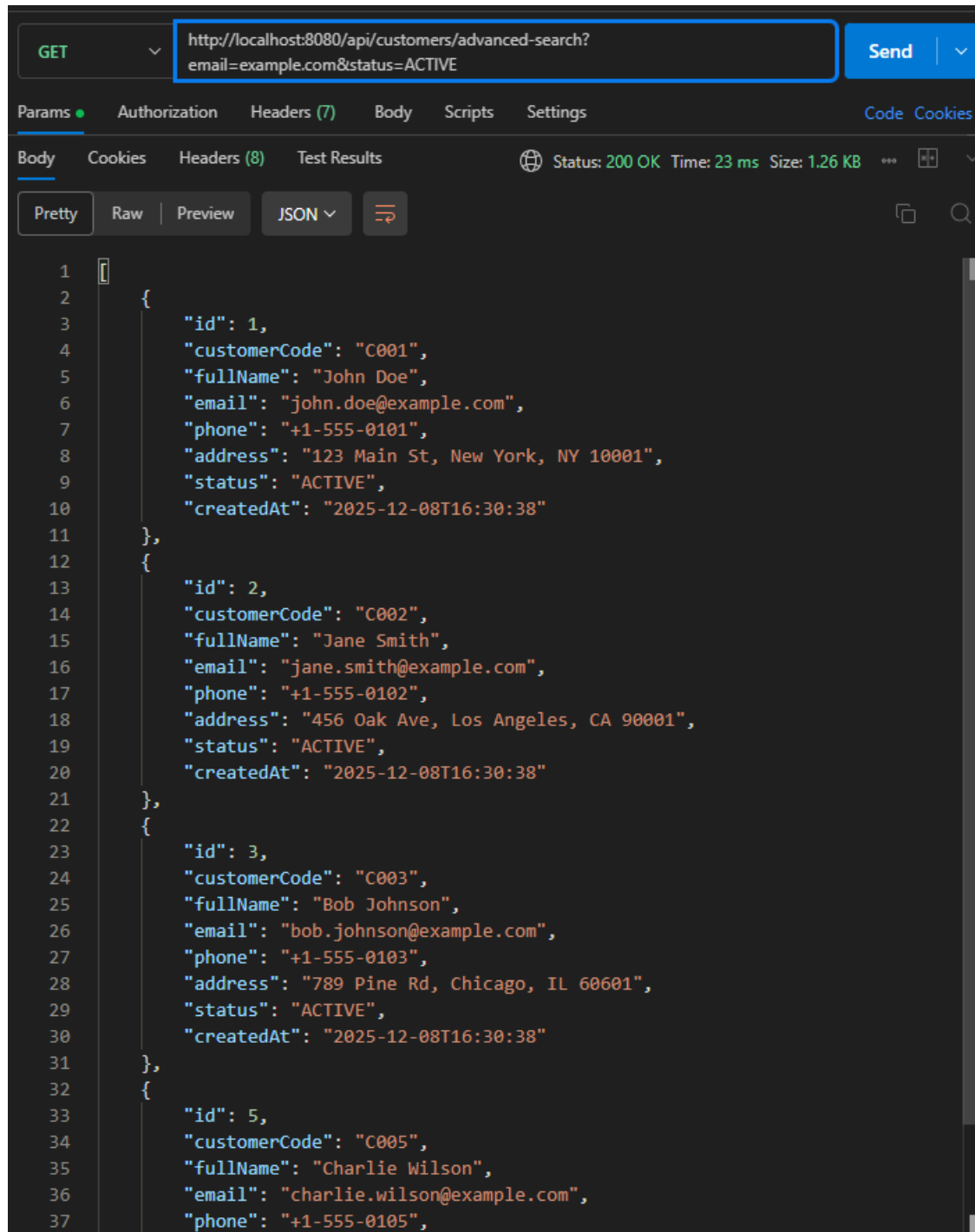
E1 — Advanced search by name (200)

This request performs an advanced search using an optional name parameter. The response returns customers whose full name matches the provided keyword (e.g., “Alice”).



E2 — Advanced search by email + status (200)

This request performs an advanced search using multiple criteria (email and status). The response returns customers that match all provided conditions (e.g., email contains “example.com” and status is ACTIVE).



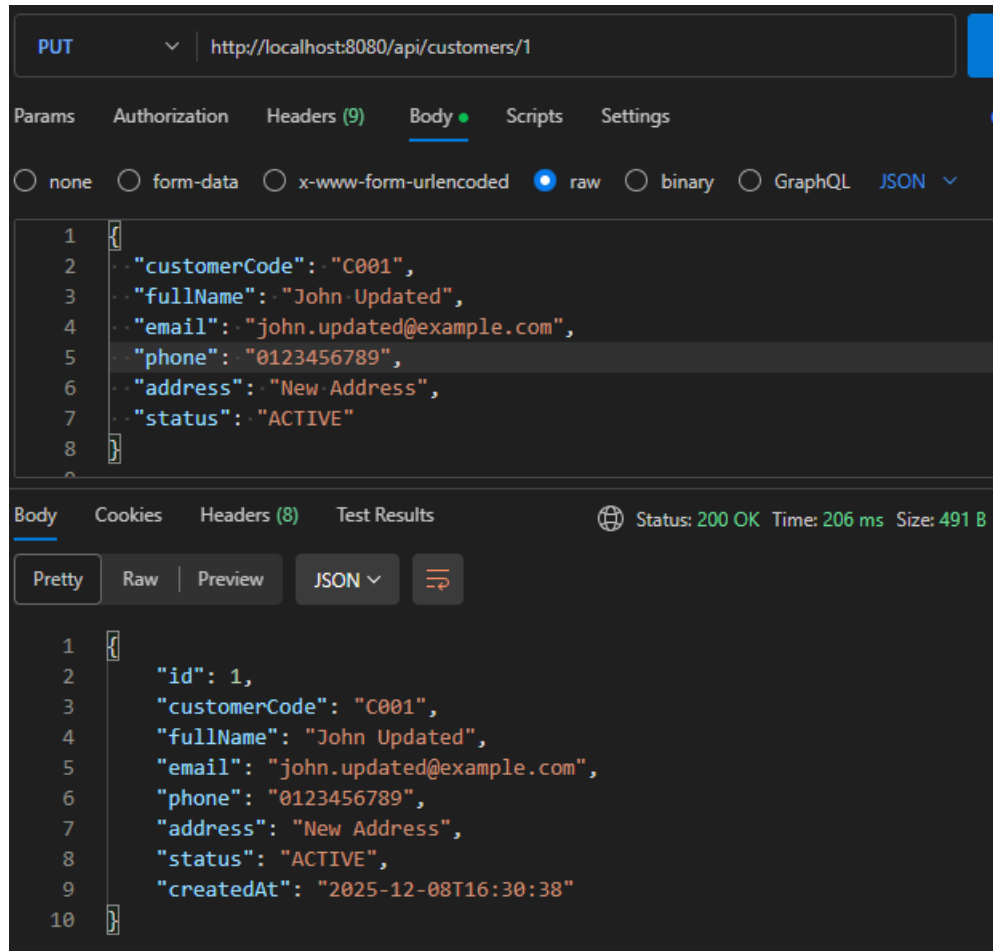
```
GET http://localhost:8080/api/customers/advanced-search?email=example.com&status=ACTIVE Send

Params Authorization Headers (7) Body Scripts Settings Code Cookies
Body Cookies Headers (8) Test Results Status: 200 OK Time: 23 ms Size: 1.26 KB
Pretty Raw Preview JSON

1 {
2   {
3     "id": 1,
4     "customerCode": "C001",
5     "fullName": "John Doe",
6     "email": "john.doe@example.com",
7     "phone": "+1-555-0101",
8     "address": "123 Main St, New York, NY 10001",
9     "status": "ACTIVE",
10    "createdAt": "2025-12-08T16:30:38"
11  },
12  {
13    "id": 2,
14    "customerCode": "C002",
15    "fullName": "Jane Smith",
16    "email": "jane.smith@example.com",
17    "phone": "+1-555-0102",
18    "address": "456 Oak Ave, Los Angeles, CA 90001",
19    "status": "ACTIVE",
20    "createdAt": "2025-12-08T16:30:38"
21  },
22  {
23    "id": 3,
24    "customerCode": "C003",
25    "fullName": "Bob Johnson",
26    "email": "bob.johnson@example.com",
27    "phone": "+1-555-0103",
28    "address": "789 Pine Rd, Chicago, IL 60601",
29    "status": "ACTIVE",
30    "createdAt": "2025-12-08T16:30:38"
31  },
32  {
33    "id": 5,
34    "customerCode": "C005",
35    "fullName": "Charlie Wilson",
36    "email": "charlie.wilson@example.com",
37    "phone": "+1-555-0105",
```

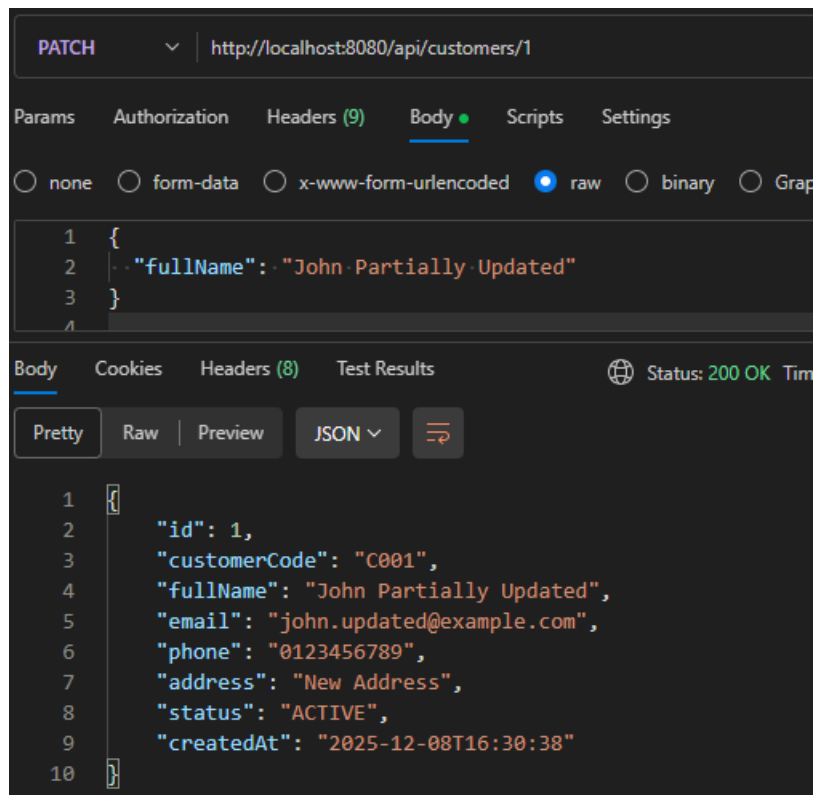
PUT1 — PUT full update customer (200)

This request fully updates a customer record using PUT. The response confirms the update succeeded and returns the updated customer data.



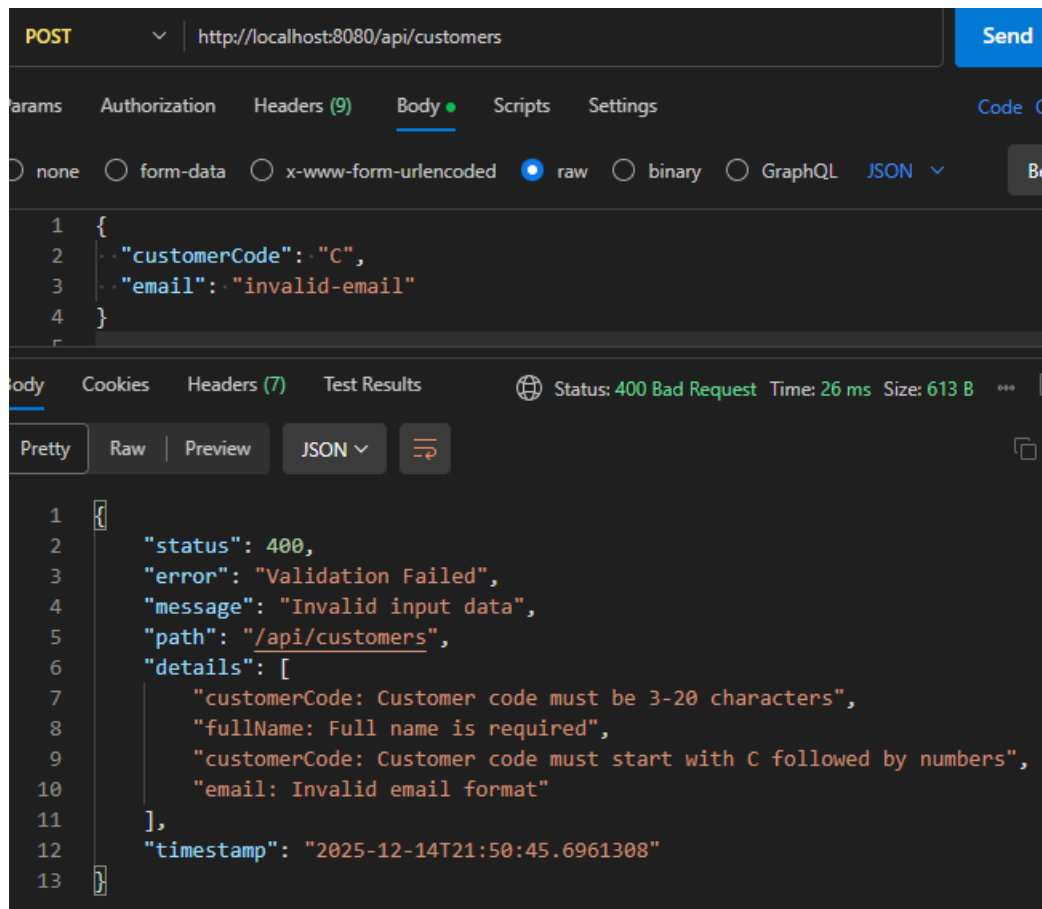
PATCH1 — PATCH partial update customer (200)

This request partially updates a customer by sending only the fields that need to change. The response confirms only the provided fields (e.g., `fullName`) were updated while other fields remain unchanged.



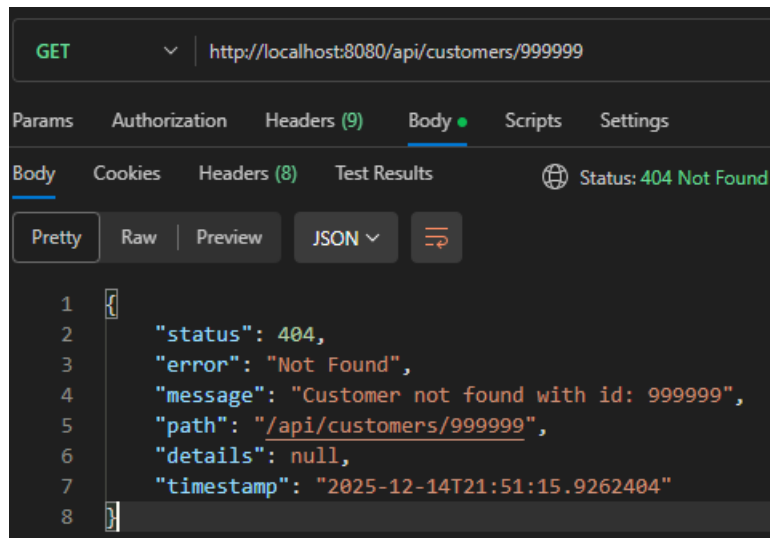
ERR400 — POST validation error (400)

This request sends invalid input data to test validation rules. The server responds with 400 Bad Request and returns validation error details for the incorrect/missing fields.



ERR404 — GET not found (404)

This request tries to fetch a customer using a non-existent ID. The server responds with 404 Not Found, confirming the resource does not exist.



ERR409 — POST duplicate resource (409)

This request attempts to create a customer with an email that already exists in the database. The server responds with 409 Conflict to indicate a duplicate resource constraint violation.

POST ▼ | http://localhost:8080/api/customers

Params Authorization Headers (9) **Body ●** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL

```
1 {
2   "customerCode": "C999",
3   "fullName": "Duplicate Email",
4   "email": "john.doe@example.com",
5   "phone": "+1555010000",
6   "address": "Test",
7   "status": "ACTIVE"
8 }
9
```

Body Cookies Headers (8) Test Results 🌐 Status: 409 Conflict T

Pretty Raw Preview JSON ▼ ≡

```
1 {
2   "status" : 409,
3   "error" : "Conflict",
4   "message" : "Customer code already exists: C999",
5   "path" : "/api/customers",
6   "details" : null,
7   "timestamp" : "2025-12-14T21:54:44.908827"
8 }
```