HTTP Responses and Troubleshooting

HTTP Response Codes & API Troubleshooting Guide

Common HTTP Response Codes

Code	Meaning	What It Usually Means	Common Cause
200	Success	Request completed successfully	API working correctly
201	Created	Resource successfully created	New user, session, or object created
400	Bad Request	API could not process request	Invalid JSON, missing fields, bad formatting
401	Unauthorized	Authentication failed	Invalid token, expired token, missing credentials
403	Forbidden	Access denied	User authenticated but lacks permissions
404	Not Found	Resource or endpoint not found	Wrong URL or API endpoint
405	Method Not Allowed	Wrong HTTP method used	Using GET instead of POST
408	Request Timeout	Request took too long	Slow network or backend delay

Code	Meaning	What It Usually Means	Common Cause
429	Too Many Requests	Rate limit exceeded	Too many API calls
500	Internal Server Error	Backend/server issue	Application crash or server-side failure
502	Bad Gateway	Invalid upstream response	Proxy/load balancer/backend issue
503	Service Unavailable	Service temporarily unavailable	Maintenance or overloaded server

---

Quick API Troubleshooting Flow

Step 1 — Identify the Error Code

Always start with:

- HTTP response code
- error message
- timestamp
- affected endpoint

Example:

```
HTTP/1.1 401 Unauthorized
```

Step 2 — Validate Authentication

Most API issues are:

authentication-related

Check:

- bearer token valid?
- token expired?
- API key correct?
- OAuth issue?
- permissions assigned?

---

401 Unauthorized

Meaning

Authentication failed.

Common Causes

- expired token
- invalid credentials
- missing Authorization header

Example

```
Authorization: Bearer invalid_token
```

Troubleshooting

- regenerate token
- verify OAuth flow
- confirm credentials
- validate headers

---

403 Forbidden

Meaning

Authenticated BUT not authorized.

Common Causes

- missing permissions
- RBAC restrictions
- blocked API access

Troubleshooting

- validate user roles
- confirm API permissions
- verify account access

---

400 Bad Request

Meaning

API request invalid.

Common Causes

- malformed JSON
- missing required fields
- invalid parameters

Example Bad JSON

```
{
"name": "Cesar"
"role": "admin"
}
```

Missing comma causes failure.

Troubleshooting

- validate JSON syntax
- review API documentation
- check required fields
- verify content-type headers

---

404 Not Found

Meaning

Endpoint/resource does not exist.

Common Causes

- incorrect URL
- typo in endpoint
- resource deleted

Troubleshooting

- verify endpoint path
- check API version
- confirm resource exists

---

405 Method Not Allowed

Meaning

Wrong HTTP method used.

Example

Using:

```
GET /api/create-user
```

when API expects:

```
POST /api/create-user
```

Troubleshooting

- verify REST method
- review API documentation

---

429 Too Many Requests

Meaning

API rate limit exceeded.

Common Causes

- excessive API calls
- automation overload

Troubleshooting

- reduce request frequency
- implement retry timers
- review API rate limits

---

500 Internal Server Error

Meaning

Backend application/server failed.

Common Causes

- application crash
- database issue
- backend exception

Troubleshooting

- check backend logs
- identify failed service
- escalate to engineering

---

502 Bad Gateway

Meaning

Gateway/proxy received invalid response.

Common Causes

- load balancer issue
- backend unavailable
- reverse proxy failure

Troubleshooting

- validate backend health
- check proxy/load balancer logs
- verify upstream connectivity

---

503 Service Unavailable

Meaning

Service temporarily unavailable.

Common Causes

- maintenance window
- overloaded system
- service outage

Troubleshooting

- verify service health
- check maintenance alerts
- retry later
- escalate if persistent

Structured Troubleshooting Methodology

1. Reproduce the Issue

Questions:

- Can issue be repeated?
- Is it intermittent?

- Does it affect all users?

---

2. Validate Authentication

Check:

- OAuth flow
- bearer token
- permissions
- API keys

---

3. Validate Request

Check:

- endpoint URL
- HTTP method
- headers
- JSON payload

---

4. Review Response Codes

Use HTTP response code to isolate:

- auth issue
- formatting issue
- backend issue
- permissions issue

---

5. Review Logs

Look for:

- timestamps
- transaction IDs
- correlation IDs
- stack traces

---

6. Validate Connectivity

Check:

- DNS
- firewall
- HTTPS/TLS
- proxies
- load balancers
- ports

---

7. Escalate Properly

Gather:

- screenshots
- logs
- timestamps
- request examples
- reproduction steps

before escalating.

Good Interview Answer

“How do you troubleshoot API issues?”

> “I typically start by identifying the HTTP response code and validating whether the issue is related to authentication, request formatting, permissions, networking, or backend failures. I review request headers, payloads, logs, connectivity, and timestamps to isolate the issue before escalating if necessary.”

Common Interview Tip

NEVER immediately blame the backend.

Good engineers:

- isolate the issue methodically
- validate layers step-by-step
- gather evidence before escalation

That’s what interviewers want to hear.

Revision #1
Created 21 May 2026 00:40:31 by Cesar Gzz
Updated 21 May 2026 00:45:22 by Cesar Gzz