QA RESOURCE GUIDE

SK

API Testing
The Definitive Guide

REST APIs · HTTP Methods · Postman · Authentication

A complete reference covering API fundamentals, REST constraints, HTTP methods, authentication types, and how to test APIs using Postman. Curated for QA engineers and SDETs.

15+	10	REST
TOPICS	PAGES	FOCUS

Created by QA Pulse by SK · skakarh.com

01 · API FUNDAMENTALS

What is API?

API stands for Application Programming Interface — a collection of functions and procedures that allows two applications or libraries to communicate with each other.

In one line: An API is an interface between different software programs or services.

Analogy

The Restaurant Waiter

API is like a waiter. You (client) give your order to the waiter (API) who takes it to the kitchen (server/system) and returns with exactly what you asked for (response).

Architecture

Frontend ↔ Backend

API sits between your Frontend (HTML, CSS, JS) and the Backend (database). The frontend makes requests through the API and the backend sends data back through it.

02 · TYPES OF API

Types of APIs

We primarily focus on **Web APIs** in API Testing.

Web API

SOAP

Simple Object Access Protocol. Uses XML + WSDL for describing network services.

Web API

RPC

Remote Procedure Call. Executes a procedure (subroutine) on another computer over a network.

Primary Focus

REST

Representational State Transfer. The most widely used Web API architecture today.

03 · API TESTING

What is API Testing?

Testing APIs and their integration with services. One of the most challenging types of testing — missing test cases at the API level can cause severe production issues that are hard to debug.

In this guide, we focus on **REST API Testing** — validating REST APIs for correctness, error codes, and load behaviour.

04 · REST API

What is REST API?

REST = REpresentational State Transfer. An API is considered REST if it follows 6 architectural constraints defined by Roy Fielding in his doctoral dissertation.

01

Uniform Interface

Client and server must agree to communicate using the same resources (JSON, XML, HTML) with consistent encoding like UTF-8 and self-descriptive messages (same MIME types).

02

Stateless

APIs in REST are stateless — neither client nor server stores information about the state of the previous request or response.

03

Cacheable

Clients can cache responses. Responses must explicitly define themselves as cacheable or non-cacheable. Server controls cache expiry.

04

Client-Server

Client and Server are separate entities. They can be developed and replaced independently, as long as the interface between them remains unchanged.

05

Layered System

Any number of intermediary layers (load balancers, caches, gateways) can exist between client and server — neither side needs to know about them.

06

Code on Demand

Server can transfer executable code or logic to the client when needed, rather than requiring client-side logic to be pre-installed. (Optional constraint)

Rule: If any API fulfills ALL of the above constraints, it can be called a REST API. Real-world examples: Twitter API, LinkedIn API, Slack API, YouTube API.

05 · REST VS SOAP

REST vs SOAP

REST API	SOAP API
Uses Resource URI (e.g. /users/1)	Uses Endpoint URI
Supports JSON, XML, HTML, plain text	Strictly XML only
Methods: GET, POST, PUT, DELETE	Methods: HTTP POST + SMTP + MQ
Lightweight, stateless, faster	More complex, uses WSDL descriptor files
Most modern web APIs use REST	Common in enterprise / legacy systems

06 · WHAT TO TEST

What to Test in API Testing?

01 Min & Max Range Validation

Validate keys with minimum and maximum range of APIs (e.g. maximum and minimum field length).

02 XML / JSON Schema Validation

Have a test case to validate the XML or JSON schema structure and ensure it matches the expected contract.

03 Keys Verification

If we have JSON or XML APIs, verify that ALL expected keys are present in the response.

04 Error Code Handling

Verify how the API handles error codes — correct HTTP status codes should be returned for invalid inputs, missing auth, and server errors.

07 · WHY API TESTING

Why Perform API Testing?

Risk

Cascading Failures

Many services rely on hundreds of interconnected APIs. If any single one fails, the entire service can go down.

Scale

Millions of APIs Online

The internet runs on millions of APIs. All of them should be tested thoroughly before going live to production.

Quality

Developers Make Mistakes

Developers create buggy APIs. API testing catches defects at the integration layer — before they reach the UI or production.

Validation

Pre-Production Validation

Validating APIs before they go live to production is critical to ensuring a stable, secure, and functional product.

08 · HTTP

HTTP Fundamentals

HTTP (HyperText Transfer Protocol) is an application layer protocol designed within the framework of the Internet protocol suite. A Client sends an HTTP Request to a Server, and the Server returns an HTTP Response.

Key Concept

Stateless Protocol

HTTP is stateless — the current request has no knowledge of what was done in any previous request. Each request is independent.

Flow

Request → Response

Client sends an HTTP Request for a resource (HTML page, file, data). Server processes it and returns an HTTP Response using the same protocol.

URL Structure: `http://www.domain.com:1234/path/to/resource?a=b&x=y`
Components: Protocol · Host · Port · Resource Path · Query Params

09 · COOKIES

What are Cookies?

Cookies are small text files with ID tags stored on the user's browser directory or program data subfolders.

What Cookies Store

Record user's browsing activity, which pages were visited, session tokens, and user preferences. Contain the domain name and a lifetime/expiry.

Cookie in HTTP Header

```
GET /spec.html HTTP/1.1
Host: www.example.org
Cookie: theme=light; sessionToken=abc123
```

10 · AUTHENTICATION

What is Authentication?

Authentication is the process of presenting your credentials (username, password, or secret key) to a system to verify your identity. In API terms, it protects API endpoints — only valid users with valid credentials can access them.

01 Basic Authentication

Credentials are encoded using Base64 and sent in the Authorization header.

```
curl --header "Authorization: Basic am9objpzZWNyZXQ=" my-website.com
```

02 Digest Authentication

Authentication is performed by transmitting the password in an encrypted form (with salt). More secure than Basic Auth.

03 OAuth (1 & 2)

Authentication protocol that allows one application to interact with another on your behalf — without sharing your actual password. e.g. "Login with Google".

Key Point: Authentication answers "Who are you?" — it verifies identity. Authorization (a separate concept) answers "What can you do?" — it grants permissions after identity is confirmed.

11 · HTTP METHODS OVERVIEW

HTTP Methods

METHOD	OPERATION	DESCRIPTION
GET	Read	Request to read/retrieve a resource or web page
POST	Create	Append/create a new resource in a collection
PUT	Replace	Request to store / fully replace a resource at a URI
PATCH	Update	Partially update a resource (one field only)
DELETE	Delete	Remove the specified resource
HEAD	Headers	Request to read only a web page's header information
TRACE	Debug	Echo the incoming request back to the sender
OPTIONS	Query	Return available HTTP methods and other options

12 · GET & POST

GET & POST Methods

GET

HTTP GET Request

```
GET /test?name=value1 HTTP/1.1
Host: scrolltest.com
```

Retrieve all resources in a collection — can be cached. GET requests remain in browser history and can be bookmarked. Should **never** be used with sensitive data. Has length restrictions. Used only to retrieve data — never to modify it.

POST

HTTP POST Request

```
POST /test/t.php HTTP/1.1
Host: scrolltest.com
name1=value1&name2=value2
```

Create a new resource in a collection. POST requests are **never cached**, do not remain in browser history, cannot be bookmarked. No restrictions on data length.

13 · PUT, PATCH & DELETE

PUT, PATCH & DELETE

PUT

HTTP PUT Request

```
PUT /boo/foo.txt HTTP/1.1
Host: www.foo.com
Content-Type: plain/text
```

Puts a file or resource at a specific URI. **Replaces** the entire file or resource. PUT responses are not cacheable. Used to fully update a resource.

PATCH

HTTP PATCH Request

```
PATCH /groups/api/v1/groups/{id}
{action:activate|deactivate}
```

Updates a **partial resource** — only used when you need to update a single field of a resource, not the entire record.

DELETE

HTTP DELETE Request

Deletes the specified resource from the server. Permanently removes the resource at the given URI.

HEAD / TRACE

HEAD & TRACE

HEAD — Returns only the header information, not the response body.
TRACE — Returns traces of the request, echoing what was received. Useful for debugging.

```
curl -I http://google.com
```

14 · METHODS COMPARISON

PUT vs PATCH vs POST

Feature	POST	PUT	PATCH
Operation	Create new resource	Replace entire resource	Update partial resource
Idempotent?	No	Yes	No
Cacheable?	No	No	No
Body required?	Yes	Yes	Yes
Use case	Submit form, create user	Update entire user record	Update user's email only

15 · CRUD MAPPING

CRUD → HTTP Method Mapping

CREATE

POST

Create a new resource in the collection.

```
POST /api/users
```

READ

GET

Retrieve a resource or list of resources.

```
GET /api/users/1
```

UPDATE

PUT / PATCH

PUT = full replace. PATCH = partial update.

```
PUT /api/users/1
```

DELETE

DELETE

Remove a specific resource by its identifier. Use with caution — often irreversible.

```
DELETE /api/users/1
```

OPTIONS — Returns available HTTP methods and other options for a specific endpoint. Commonly used in CORS (Cross-Origin Resource Sharing) preflight checks.

16 · API TESTING APPROACH

How to Test an API?

API testing can be done manually or using tools. It is always recommended to use dedicated API testing tools for efficiency and repeatability.

Example Public API: `https://api.chucknorris.io/jokes/random`
Returns JSON with keys: category, icon_url, id, url, value — all as String or Number types.

Step 1

Understand the API Contract

Review the API documentation. Know the expected request format (method, headers, body) and the expected response format (status code, JSON keys, data types).

Step 2

Validate Response Structure

Confirm all expected JSON/XML keys are present in the response. Validate data types (string, number, boolean, null) match the API spec.

Step 3

Verify Status Codes

Test all expected HTTP status codes: 200 OK, 201 Created, 400 Bad Request, 401 Unauthorized, 404 Not Found, 500 Server Error.

Step 4

Test Edge Cases

Test with invalid inputs, missing required fields, boundary values, empty payloads, and special characters to verify error handling.

17 · API TESTING TOOLS

API Testing Tools

01 **Postman (Recommended)**
The most popular API testing tool. Supports global variables, mock requests, environments, CI/CD integration, and API monitoring with test case support.

02 **Rest Assured**
Java-based library for REST API testing. Great for automation and CI/CD pipelines.

03 **SoapUI**
Supports both REST and SOAP API testing. Popular in enterprise environments for functional, security, and load testing of APIs.

04 **Katalon Studio**
All-in-one test automation tool supporting API, web, mobile, and desktop testing with CI/CD integration.

05 **Runscope**
Cloud-based API monitoring and testing tool — great for monitoring API health in production environments.

18 · POSTMAN

API Testing Using Postman

Postman is the industry-standard API testing tool used by developers and testers alike. Available as a native app on macOS, Windows, and Linux.

Download: https://www.getpostman.com

Feature

Collections

An executable API description format. Run requests, test & debug, create automated tests, and mock and document APIs — all organized in collections.

Feature

Workspaces

Powerful collaboration spaces for teams of any size. Share collections, set permissions, and manage participation across multiple workspaces.

Feature

Built-in Tools

Everything a developer needs to work with APIs — design & mock, debug, test automation, documentation, monitoring, and publish.

19 · POSTMAN KEY FEATURES

Postman Key Features

01

Global Variables & Environments

Define variables (base URLs, tokens) once and reuse them across all requests. Switch between Dev, Staging, and Production environments instantly.

02

Test Scripts (Pre/Post)

Write JavaScript test scripts to automatically validate response status codes, response time, JSON keys, and data values after every request.

03

Mock Requests

Create mock servers to simulate API responses before the real backend is ready. Enables frontend and QA teams to work in parallel.

04

CI/CD Integration

Run Postman collections from the command line using Newman. Integrate API tests into your CI/CD pipeline (Jenkins, GitHub Actions, GitLab CI).

05

API Monitoring

Schedule collection runs to monitor API health and performance over time. Get alerts when APIs fail or degrade.

20 · HTTP STATUS CODES

HTTP Status Codes

Every API response includes an HTTP status code. Knowing what each code means is essential for effective API testing.

2xx – Success

Success Codes

200 OK — Request succeeded.
201 Created — Resource was created (POST).
204 No Content — Success but no body (DELETE).

3xx – Redirect

Redirect Codes

301 Moved Permanently — Resource URL changed permanently.
302 Found — Temporary redirect.
304 Not Modified — Use cached version.

4xx – Client Error

Client Error Codes

400 Bad Request — Invalid request format.
401 Unauthorized — Authentication required.
403 Forbidden — No permission.
404 Not Found — Resource doesn't exist.
429 Too Many Requests — Rate limited.

5xx – Server Error

Server Error Codes

500 Internal Server Error — Generic server failure.
502 Bad Gateway — Invalid response from upstream.
503 Service Unavailable — Server overloaded or down.

21 · QUICK REFERENCE

API Testing Quick Reference

METHOD	CRUD	CACHEABLE?	IDEMPOTENT?	BODY?
GET	Read	Yes	Yes	No
POST	Create	No	No	Yes
PUT	Replace	No	Yes	Yes
PATCH	Update	No	No	Yes
DELETE	Delete	No	Yes	Optional

QA Pulse by SK
Your weekly signal for QA, Test Automation & AI in Software Engineering
skakarh.com
Subscribe Free