API Testing Tutorial for Beginners

Introduction

Confused by technical terms such as JSON, an endpoint, and a status code? The concepts of how APIs work and how to perform manual and tool-based testing baffle many beginners.

This tutorial simplifies the complex world of API Testing by showing you exactly how data flows and what to check for. We focus on practical, hands-on steps so that you have full confidence in your software's reliability and performance.

Ready to master essential testing skills? Have a look at our complete **API Testing Course Syllabus** and start building your expertise!

Why Students or Freshers Learn API Testing?

The following are the reasons for students or freshers learning API Testing:

- **High Market Demand:** Modern software relies heavily on APIs. Proficiency in API testing is a highly sought-after skill that makes you a critical asset in any development team.
- **Career Foundation:** API testing lays the foundation for automation and shift-left testing practices that will set you up for a career in DevOps, QA Automation, and Backend Development.
- **Faster, Cheaper Testing:** It is faster and more reliable to test the API layer than the UI. Learning this saves time and catches bugs earlier on in the development cycle.
- **Better Understanding of Software Architecture:** This provides a deep insight into how the client, i.e., front-end, and server, or back-end, communicate, which becomes important for full-stack knowledge.

Get your dream job! Gear up with our selected list of **API testing interview questions and answers** today.

Step-by-Step API Testing Tutorial for Beginners

Testing an API-which stands for Application Programming Interface-is a vital activity in software systems to validate the layer of communication between different services. This API Testing tutorial will walk you through setting up and performing manual testing using Postman, the industry-standard tool.

Part 1: Installation and Setup (The Tool)

To start testing APIs, we need a dedicated tool to send HTTP requests and inspect responses with ease. Postman is the best place to begin.

Step 1: Install Postman

- **Download Postman:** From the official Postman website, download the application corresponding to your operating system: either Windows, macOS, or Linux.

- **Install and Open:** Follow through on the installation prompts. Once installed, launch Postman.
- **Create an Account:** You can log in to sync your work, or you can select “Skip and go to the app” for local use.

Step 2: Set Up a Workspace and Collection

In Postman, a Workspace is where you organize your projects, and a Collection groups related API requests.

- **Create a New Workspace:** Click on Workspaces in the top left, then click on Create Workspace. Give it a descriptive name such as “Beginner API Testing”.
- **Create a New Collection:** Click on New in the sidebar, select Collection, and name it, such as “Public API Practice”.

Step 3: Find a Practice API

For this API Testing tutorial, we’ll use a public, dummy API that allows you to perform basic operations without requiring any authentication or touching real data. We are going to use the JSONPlaceholder API.

- **Base URL or Endpoint:** *https://jsonplaceholder.typicode.com*

Part 2: Understanding HTTP Methods and Status Codes

Before you can test, you need to understand the basic language APIs use: HTTP methods and status codes.

Step 4: Learn the Core HTTP Methods

APIs use standard verbs or methods to tell the server what action to perform on a resource, like a user or a post.

HTTP Method	Action	Purpose	Analogy

GET	Retrieve	Fetches data from a server (Read).	Asking a library for a specific book.
POST	Create	Sends data to a server to create a new resource.	Submitting a form to create a new user profile.
PUT/PATCH	Update	Sends data to completely replace (PUT) or partially modify (PATCH) an existing resource.	Correcting or changing details on an existing user profile.
DELETE	Remove	Requests the removal of a specified resource.	Throwing away an unwanted item.

Step 5: Identify Important HTTP Status Codes

A response from an API always includes a three-digit status code indicating the outcome of the request.

Range	**Meaning**	**Common Codes**	**Description**
2xx	Success	200 OK, 201 Created, 204 No Content	The request was successfully received, understood, and accepted.

4xx	Client Error	400 Bad Request, 401 Unauthorized, 404 Not Found	The client (you) sent a request that the server cannot process.
5xx	Server Error	500 Internal Server Error	The server failed to fulfill a valid request due to its own error.

Part 3: Manual API Testing with Postman

We will now perform the four core CRUD operations (Create, Read, Update, Delete) against our practice API.

Step 6: Test the GET Method for Reading Data

The objective is to fetch a list of all posts.

1. **Create a New Request:** In your collection, click the three dots (.) and select Add Request. Name it GET All Posts.
2. **Set Method:** The method dropdown should be set to GET.
3. **Enter Endpoint:** In the URL bar, paste the complete endpoint: *https://jsonplaceholder.typicode.com/posts*
4. **Send Request:** Click the **Send** button.
5. **Validate Response:**
    - **Status Code:** The status should be 200 OK.
    - **Body:** Click the Body tab, and verify that it contains a JSON array (starts with [) of post objects.
    - **Latency:** Note the time taken, such as time: 150ms.

Step 7: Testing the POST Method (Creating Data)

The goal is to create a new post on the server.

1. **Create a New Request:** Add a new request, naming it *POST Create Post*.
2. **Set Method:** In the dropdown for method, select POST.
3. **Enter Endpoint:** Use the same URL: https://jsonplaceholder.typicode.com/posts
4. **Prepare Body Data:**
    - Click the Body tab below the URL.
    - Select the raw radio button.
    - Select JSON from the dropdown menu, if not already selected.
    - Paste the following JSON data into the text area:

{

“title”: “My First API Post”,

“body”: “This is content created via Postman API testing.”,

“userId”: 1

}

5. **Send Request:** Click Send.
6. **Validate Response:**
    - **Status Code:** Verify that the status is 201 Created.
    - **Body:** The response body should include the data you submitted with a new unique identifier, for example, “id”: 101, confirming creation.

Step 8: Test the PUT Method (Updating Data)

The idea here is to edit an existing post – we’ll edit the one with ID 1.

1. **Create a New Request:** Add a new request, naming it *PUT Update Post 1*.
2. **Set Method:** In the method dropdown, select PUT.

3. **Endpoint:** Change the URL to point to a specific resource: *https://jsonplaceholder.typicode.com/posts/1*
4. **Prepare Body Data:**
    - Go to the Body tab, select raw and JSON.
    - Paste the following, making sure the title and body have changed:

{

“id”: 1,

“title”: “Updated Title via PUT”,

“body”: “The entire content has been replaced.”,

“userId”: 1

}

5. **Send Request:** Click Send.
6. **Validate Response:**
    - **Status Code:** Verify the status is 200 OK.
    - **Body:** The response body should include the updated title and body.

Step 9: Test the DELETE Method (Deleting Data)

The objective is to remove the post whose ID is 1.

1. **Create a New Request:** Add a new request, naming it *DELETE Post 1*.
2. **Set Method:** In the method dropdown, select *DELETE*.
3. **Enter Endpoint:** Use the following target URL: *https://jsonplaceholder.typicode.com/posts/1*
4. **Body/Headers:** Normally there is no body or special headers required for a simple DELETE.

5. **Send Request:** Click Send.
6. **Validate Response:**
    - **Status Code:** Verify the status is 200 OK or 204 No Content. The 204 status is frequently used to confirm deletion was successful without returning a body.
    - **Body:** The body should be empty or contain an empty JSON object {}.

Part 4: Beyond Basic Testing

Once you get comfortable with performing CRUD operations, it's time to learn how to make your testing efficient and repeatable.

Step 10: Using Parameters and Headers

Parameters and Headers are used to refine requests.

Query Parameters (Filtering):

- Change your *GET All Posts* request URL to:

https://jsonplaceholder.typicode.com/posts?userId=1

- Note that as you type the parameter (userId=1), Postman automatically populates the Params tab below the URL bar.
- Send the request. The response body now should contain only posts where the userId is 1.

Headers (Authentication/Data Type):

- Click the Headers tab. You'll often want to send headers either for authentication,
    - For example, ***Authorization: Bearer <token>***, or for specifying the format, such as ***Content-Type: application/json.***
- For this API, no specific header is needed, but this is where you would configure security tokens.

Step 11: Introduce Postman Tests (Automation Start)

The real power of Postman comes from the writing of Tests, which are pieces of JavaScript that automatically validate the response.

1. **Open an Existing Request:** Open your *GET All Posts* request.
2. **Go to the Tests Tab:** Click the Tests tab next to the Body tab.
3. **Add a Simple Test:** Copy the code below into the editor. This code simply checks if the returned HTTP status code is 200 and if the response time is fast enough.

```
// Test 1: Check if the status code is 200

pm.test("Status code is 200 OK", function () {

    pm.response.to.have.status(200);

});

// Test 2: Check the response time (Latency)

pm.test("Response time is under 500ms", function () {

    pm.expect(pm.response.responseTime).to.be.below(500);

});

// Test 3: Check if the response body is an array (Basic structure validation)

pm.test("Response body is a JSON array", function () {

    const responseData = pm.response.json();

    pm.expect(responseData).to.be.an('array');
```

});

4. **Send and View Results:** Click Send. Now click on the Test Results tab beside the Status code. You should see three passing tests, marking your first step into API Test Automation!

Ready to take on the testing puzzles in the real world? Download our guide to **API Testing Challenges and Solutions** to debug complex issues, handle authentication, and master asynchronous operations.

Real Time Examples for API Testing Tutorial for Beginners

Each of these examples shows how a different kind of API is tested within a live, production environment, offering practical insight into real-world QA.

E-Commerce Shopping Cart Microservices

- **The Scenario:** When a user adds an item to the cart, the front-end triggers a POST request to the /cart/add API endpoint.
- **Real-Time Testing:**
    - Testers ensure the API instantly returns the 201 Created status code and verifies that the response body correctly shows the added item, its updated quantity, and the new total price.
    - This requires immediate validation of the business logic at the API layer.
- **Key Skill:** Testing transactional consistency and data integrity across multiple microservices, such as Inventory, Pricing, and Cart services.

Payment Gateway Integration (Webhooks)

- **Scenario:** Once the payment is successfully made, the provider initiates an automated, asynchronous notification-a webhook-to the /payment/status endpoint of your server.
- **Real-time Testing:**

    - Testers will simulate this external POST request to ensure that the server receives and processes the data as expected through an immediate update in their internal database (e.g., changing the order status from ‘Pending’ to ‘Paid’).
    - They are also supposed to test for error handling on failure of that webhook to arrive.
- **Key Skill:** Testing asynchronous communication, security headers, and guaranteed system state changes.

Live Stock Market Price Feed

- **Scenario:** A trading platform constantly retrieves real-time stock prices through a GET request on a dedicated, fast /market/live API endpoint.
- **Real-Time Testing:**
    - The focus is shifting from simple correctness to performance and reliability.
    - Testers, by using tools that send thousands of requests per second (load testing), make sure the API maintains a sub-50ms response time and doesn’t return stale or incorrect data, which is critical for trading applications.
- **Key Skill:** Load Testing, response latency verification, assurance of data type/format correctness in case of prices being floats.

Put theory into practice! Read through our list of practical **API Testing project ideas** to build a strong portfolio and master advanced automation techniques.

FAQs About API Testing Tutorial for Beginners

1. How to start learning API Testing?

Understand the HTTP methods (GET, POST), status codes (200, 404), and JSON data format. Set up a tool like Postman to do some manual testing in public APIs, such as JSONPlaceholder. Move on to writing simple validation scripts in the same tool.

2, What are the 4 types of API?

The four main types based on architecture are: SOAP, using XML; REST, most common, uses JSON/XML over HTTP; RPC, or Remote Procedure Call; and GraphQL, which allows clients to only request the data they need.

3. What is the basic of API Testing?

The main objective here is the testing of the business logic and communication between systems. This includes sending requests-like GET/POST-to API endpoints and verifying the response status code-e.g., 200-and the data structure and content in the response body.

4. Can I learn testing in 3 months?

Yes, you can learn the basics of manual and basic automated testing, like API and UI basics, in three months. Continuous practice, building a portfolio, and understanding advanced concepts, such as performance testing, take more time to be proficient and job-ready.

5. What are the 5 methods of API?

Here are the five most common HTTP methods, mapping to the CRUD operations: GET for Read, POST for Create, PUT for Update/Replace, PATCH for Update/Modify, and DELETE for Remove.

6. Which tool is best for API testing?

Postman is generally regarded as the most popular tool for doing manual and exploratory API tests, based on its ease of use, rich features, and collaboration capabilities. For pure automation and integration into CI/CD pipelines, Rest Assured in Java or Pytest/Requests in Python can serve very well.

7. Is API Testing easy to learn?

API testing is generally much easier to start compared to complex UI testing, since it deals directly with data and logic and not with the complexities of web browsers and graphical elements. Once you understand JSON and HTTP, the learning curve isn’t steep at all.

8. What are the three types of testing in API?

The three main types are Functional Testing, which checks for correct business logic and data; Security Testing, which checks authentication and authorization; and Performance Testing, which assesses speed and stability under load.

9. Can I learn API without coding?

You can start learning manual API testing using tools like Postman or SoapUI without deep coding knowledge. To advance into API test automation, which is essential for most jobs, you will be required to learn a scripting language in either Python or JavaScript.

10. What is API salary?

In countries like the US, the average **API Tester Salary for Freshers** ranges are from $80,000 to $120,000 annually, depending on experience, location, and specific automation skills. Eg., Rest Assured and advanced tooling.

Conclusion

Congratulations! You have gone through the basics of API testing, starting with setting up Postman to performing CRUD operations and writing automated tests. You now understand how to validate the core communication layer of modern applications. Mastering this skill is the way to be a valuable member in any development team. Advanced automation and integration are where the real power is; don't just learn the basics. Ready to become an expert? Enroll in our comprehensive **API testing course in Chennai** today to unlock advanced techniques and deployment strategies!