

API Testing with Postman



What is API?



What is API?

Application programming interface (API)

- API is a software interface that allows two applications to interact with each other without any user intervention.

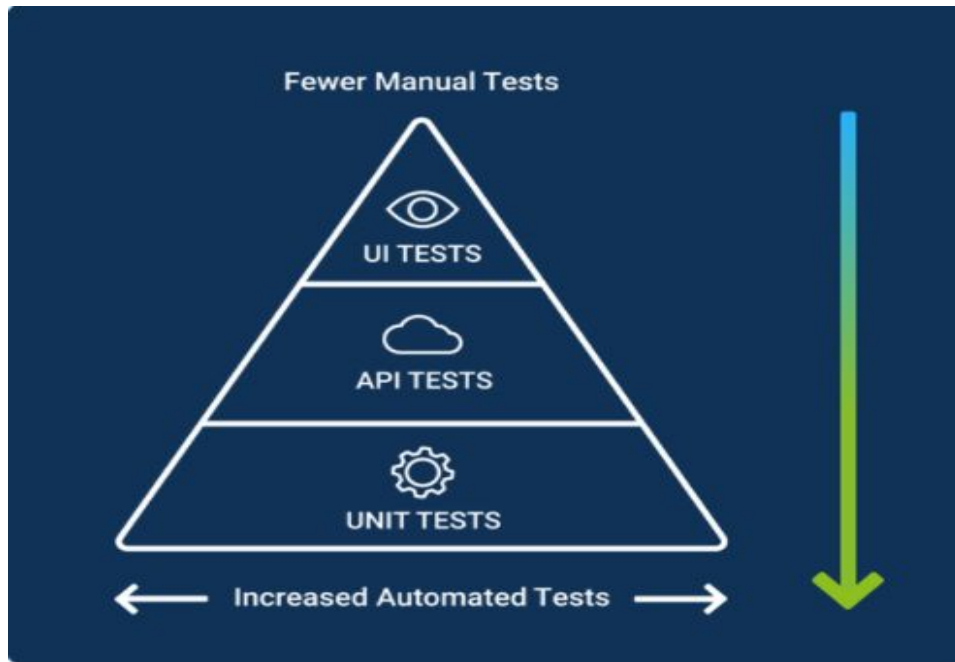


Why API Testing is required?



Why API testing is important

- Focuses on business rules of the application
- Early feedback/Time Effective
- Speed and Coverage of testing
- Faster time to resolution

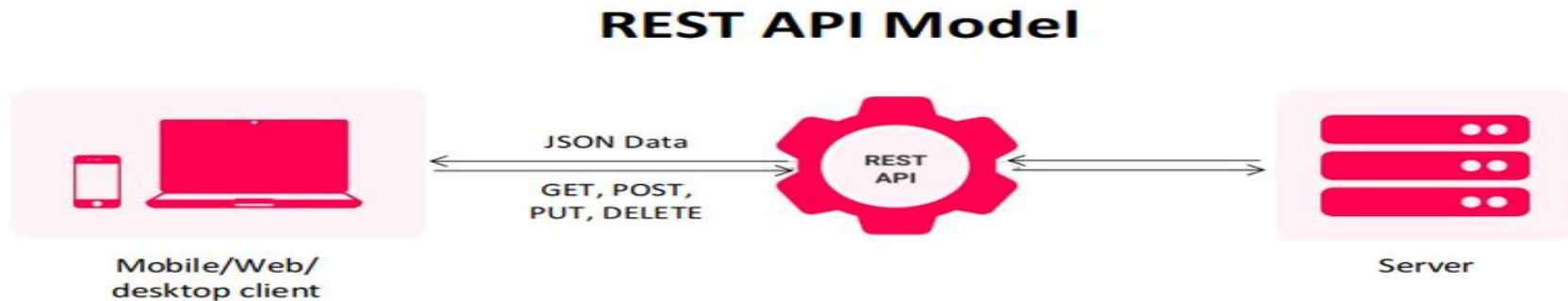


What is REST API?



What is REST?

- REST stands for Representational State Transfer
- A RESTful API is an architectural style for an application program interface that uses HTTP requests to access and use data
- RESTful web services are stateless



URL - Uniform Resource Locator

- A *Uniform Resource Locator (URL)* that defines the location of the request
- URL is composed of different parts, some mandatory and others optional

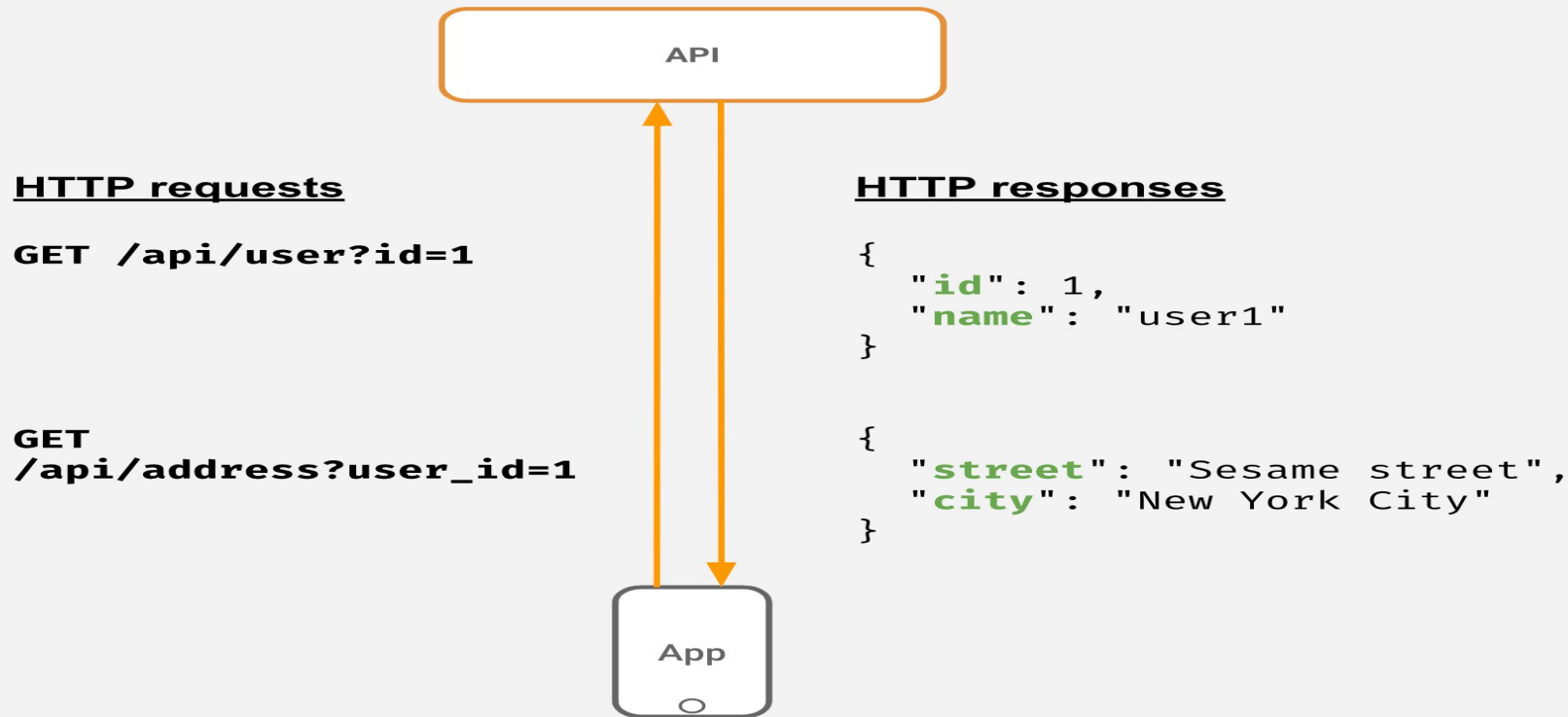
<https://spree-vapasi-prod.herokuapp.com/t/bags>

Protocol: **https**

Domain name: spree-vapasi-prod.herokuapp.com

Resource : [/t/bags](https://spree-vapasi-prod.herokuapp.com/t/bags)

Request and Response



REST API calls

HTTP Headers

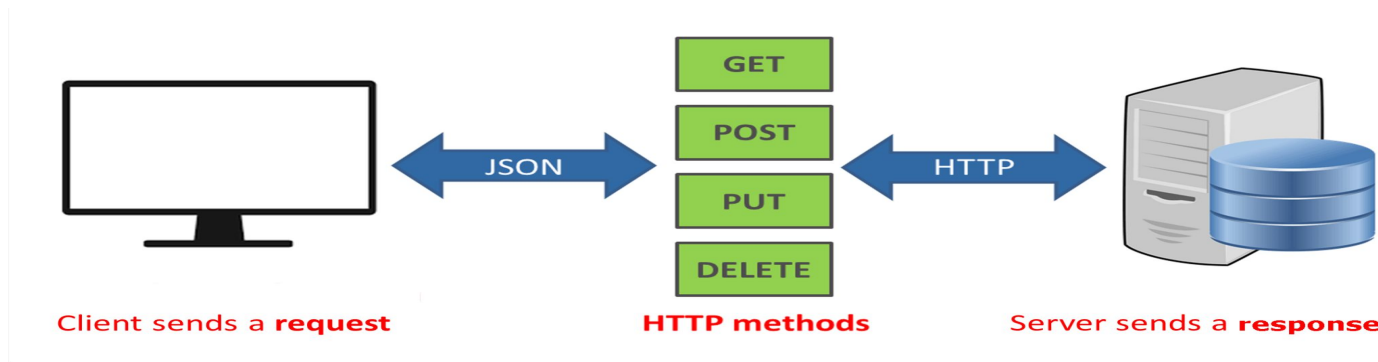
- **HTTP headers** that provide information to the server about the request
- A **request body** that provides further details for the request

Header name	Description	Examples
Content-Type	Indicates the content type that is used in the body of the request. The supported content type is XML.	application/xml
Accept	Specifies the content types that are valid in the response message.	application/xml application/json
Authorization	Contains the authentication credentials for HTTP authentication.	
Cache-Control :	The cache policy defined by the server for this response, a cached response can be stored by the client and re-used till the time defined by the Cache-Control header.	

If the server cannot respond with the requested content type, the 406 Not Acceptable HTTP status will be returned.

HTTP Methods

- A **GET** request fetches a record from a database
- A **POST** request adds a new record to a database
- A **PUT** request replaces a record with a new one
- A **DELETE** request removes a record from a database



HTTP Response/Status Codes

- 1xx: Informational
- 2xx: Success
- 3xx: Redirection
- 4xx: Client Error
- 5xx: Server Error

Response codes - <https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>

CAT Codes - <https://http.cat/>

DEMO URL - <https://openweathermap.org/current> <https://reqres.in/>

API Testing - How?



Postman



Postman - Introduction

Postman is an application which makes API testing easy and provides:

1. User-friendly interface for interacting with an API
2. Ability to setup test data, construct an HTTP request with custom payload and assert for values in the response
3. Run and configure a collection of requests using a built in runner or via command line

How to install Postman?

<https://www.postman.com/downloads/>

Making a Request



Making a request

Home

Workspaces

Reports

Explore

Search Postman

Sign In

Create Account

Scratch Pad

New

Import

GET New Request

+

...

No Environment

▼

👁

New Request

Save

▼

...

✎

💬

📄

GET

▼

Enter request URL

Send

▼

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Cookies


</>

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

Response

▼



Hit Send to get a response

📄

Find and Replace

Console

Runner

🔍

?

18

GET Request



GET Request

URL: <https://test-api.k6.io/public/crocodiles/>

Let's try this out...

- Install postman using <https://www.postman.com/downloads/>
- Hit the below GET Request in postman and check the response

<https://test-api.k6.io/public/crocodiles/>

Sample App



Sample Application Flow

- **User Registration**
- **Login as Registered User**
- **As a logged in user, You should be able to**
 - **Create Crocodiles**
 - **Update your Crocodiles**
 - **See your own Crocodiles**
 - **Delete a Crocodile**

POST Request



POST Request

URL: <https://test-api.k6.io/user/register/>

Request Body:

```
{  
  "username": "loges_r",  
  "first_name": "Loges",  
  "last_name": "R",  
  "email": "logeslgs@gmail.com",  
  "password": "Loges123"  
}
```

POST Request to Login

URL: <https://test-api.k6.io/auth/cookie/login/>

Request Body:

```
{  
  "username": "loges_r",  
  "password": "Loges123"  
}
```

POST Request after Login

URL: <https://test-api.k6.io/my/crocodiles/>

Request Body:

```
{  
  "name": "Croc 3",  
  "sex": "F",  
  "date_of_birth": "2020-02-01"  
}
```

PUT Request



PUT Request

URL: `https://test-api.k6.io/my/crocodiles/{id}/`

Request Body:

```
{  
  "name": "Croc 3",  
  "sex": "F",  
  "date_of_birth": "2020-02-01"  
}
```

DELETE Request



DELETE Request

URL: <https://test-api.k6.io/my/crocodiles/{id}/>

Exercise

Let's try this out...

Refer the APIs in <https://test-api.k6.io/>

- Register as a new user
- Login as a registered user
- Create 3 crocodiles "Croc 1", "Croc 2" and "Croc 3"
- Update "Croc 1"
- Delete "Croc 3"

Points to remember

- *Keep an eye on the HTTP method and status code in response for the API that you are using*
- *Check if you can produce different status codes*

API Parameters...

Query Parameter:-

`https://myserver.com/resource-name?param1=value1¶m2=value2`

Path Parameter:-

`http://myserver.com/some-path/parameter/path-continued/parameter2`

Request Body Parameter:-

```
{  
  "dMarketValueBeforeTrade": 20432666,  
  "price": "950.000000"  
}
```

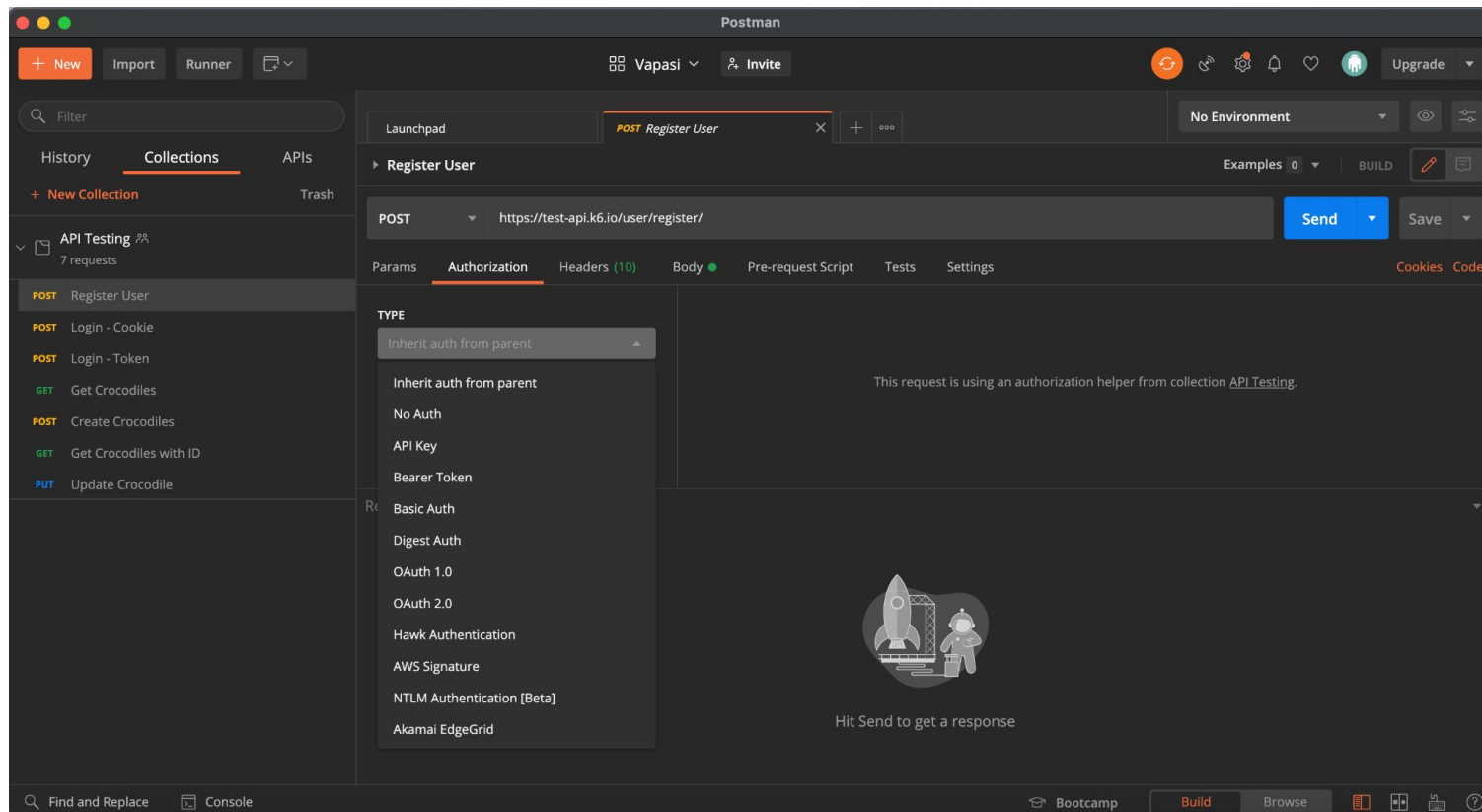
Authorizing a request



Authorizing a request

- APIs use authorization to ensure that client requests access data securely. This can involve authenticating the sender of a request and verifying that they have permission to access or manipulate the relevant data.
- Auth data can be included in the header, body, or as parameters to a request. If you enter your auth details in the Authorization tab, Postman will automatically populate the relevant parts of the request for your chosen auth type.

Authorizing a request



Requests with Auth



To fetch Auth token

URL: <https://test-api.k6.io/auth/token/login/>

Request Body:

```
{  
  "username": "loges_r",  
  "password": "Loges123"  
}
```

POST Request with Auth

URL: <https://test-api.k6.io/my/crocodiles/>

Request Body:

```
{  
  "name": "Croc 3",  
  "sex": "F",  
  "date_of_birth": "2020-02-01"  
}
```


Appendix

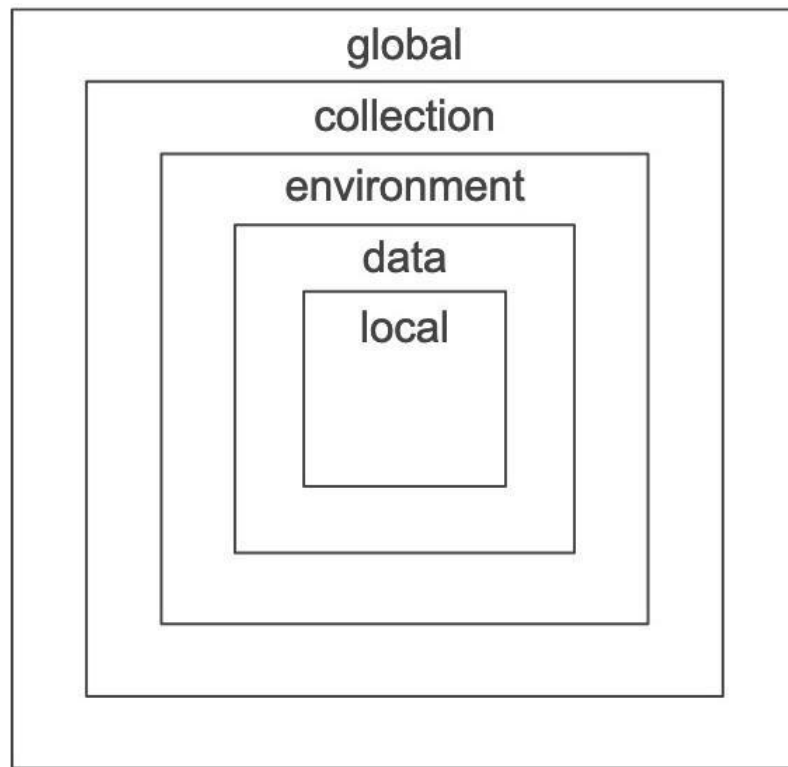
Managing Variables and Environments



Using Variables

- Variables enable you to store and reuse values in your requests and scripts.
- By storing a value in a variable, you can reference it throughout your collections, environments and requests.
- If you need to update the value, you only have to change it in one place.

Variable Scope



Managing Environments

- An environment is a set of variables you can use in your Postman requests.
- You can toggle between environments when running your requests to test them against either the development or the production environment.

Let's try this out...

- Create an environment “Test”
- Add the login credentials “username” and “password” as variables
- Add the placeholders in the login request [/auth/token/login/](#) that we used in the last exercise
- Check if you are able to login successfully

Commonly used HTTP codes

200 OK success status response code indicates that the request has succeeded.

400 Bad Request response status code indicates that the server cannot or will not process the request due to something that is perceived to be a client error

401 Unauthorized client error status response code indicates that the client request has not been completed because it lacks valid authentication credentials for the requested resource.

Commonly used HTTP codes

403 Forbidden client error status response code indicates that the server understands the request but refuses to authorize it.

404 Not Found client error response code indicates that the server can't find the requested resource.

500 Internal Server Error server error response code indicates that the server encountered an unexpected condition that prevented it from fulfilling the request.

Thank you

