API Session – Chapter 12

- Authentication Types -

# Authorization

Authorization is a process that verifies whether a user or application has the necessary permissions to access a resource. In API testing, it's crucial because it ensures that only authenticated and authorized users can interact with the API. This helps protect sensitive data and functionalities from unauthorized access or misuse. Properly testing authorization verifies that the API correctly enforces security policies and handles access controls as intended.

Postman supports various types of authorization methods for securing API requests. Here’s a rundown of the most common types you can use and how to configure them in Postman:

# Auth Types

## **1. No Auth**

* **Description**: No authentication is used; the request is sent as-is.
* **Usage**: When the API does not require authentication.
* **Configuration**: Select No Auth from the Authorization tab.

## **2. Basic Auth**

* **Description**: Authentication using a username and password encoded in Base64.
* **Usage**: Simple and often used for internal or less secure APIs.
* **Configuration**:
  + **Type**: Basic Auth
  + **Username**: Enter your username.
  + **Password**: Enter your password.
  + Postman will automatically encode these details and add them to the Authorization header.

### Example

Here's a quick example configuration for basic auth:

* **URL**: <https://postman-echo.com/basic-auth>
* **Method**: GET
* **Authorization:** Basic Auth [ Enter username and password]
* **postman / password**

Another Example - <https://the-internet.herokuapp.com/basic_auth>

## **3. Digest Auth**

* **Description**: A more secure method than Basic Auth, where authentication is done via a digest algorithm.
* **Usage**: Often used in APIs requiring additional security over Basic Auth.
* **Configuration**:
  + **Type**: Digest Auth
  + **Username**: Enter your username.
  + **Password**: Enter your password.
  + **Realm**: (Optional) The authentication realm, if required by the server.
  + **Algorithm**: Select the appropriate algorithm (usually MD5).

### Example

Here's a quick example configuration for basic auth:

* **URL**: <https://postman-echo.com/digest-auth>
* **Method**: GET
* **Authorization:** Digest Auth [ Enter username and password which will be encrypted and sent]

Another Example - <https://the-internet.herokuapp.com/digest_auth>

## **4. API Key**

* **Description**: A key is sent as a part of the request header or query parameters.
* **Usage**: Common in APIs where the key is used to identify the client.
* **Configuration**:
  + **Type**: API Key
  + **Key**: Enter the key name as specified by the API (e.g., Authorization or x-api-key).
  + **Value**: Enter the API key value.
  + **Add to**: Choose Header or Query Params depending on where the API expects the key.

**Steps to generate API Key**

**Reference:** [**https://www.nettoolkit.com/docs/overview/test-api-keys#get-a-test-api-key**](https://www.nettoolkit.com/docs/overview/test-api-keys#get-a-test-api-key)

**API Endpoint(To generate API Key):** <https://api.nettoolkit.com/v1/account/test-api-keys>

### Example

Here's a quick example configuration for API key auth:

* **URL**: <https://api.nettoolkit.com/v1/geo/tiles/12/1171/1566>
* **Method**: GET
* **Authorization:** API Key [Key: X-NTK-KEY], [Value: Key generated from previous request]

## **5. Bearer Token**

* **Description**: A token is used for authentication, usually in OAuth2.0.
* **Usage**: Common in modern APIs and for services using OAuth2.0.
* **Configuration**:
  + **Type**: Bearer Token
  + **Token**: Enter your bearer token value.
  + Postman will automatically add the token to the Authorization header as Bearer <token>.

### Example

Here's a quick example configuration for bearer token:

* **URL**: <https://gorest.co.in/public/v2/users/>
* **Method**: POST
* **Authorization:** Bearer <Token>

Another Example – <https://api.github.com/user/repos>

## **6. OAuth 1.0**

* **Description**: Authentication using OAuth 1.0a protocol with token-based authorization.
* **Usage**: Common in APIs that use OAuth 1.0 for authentication.
* **Configuration**:
  + **Type**: OAuth 1.0
  + **Consumer Key**: Enter your consumer key.
  + **Consumer Secret**: Enter your consumer secret.
  + **Token**: Enter your access token.
  + **Token Secret**: Enter your token secret.
  + **Signature Method**: Choose the signature method (e.g., HMAC-SHA1).

## **7. OAuth 2.0**

* **Description**: A widely used authorization framework that provides a variety of flows (e.g., Authorization Code, Client Credentials).
* **Usage**: Common in modern applications using OAuth2 for user and app authorization.
* **Configuration**:
  + **Type**: OAuth 2.0
  + **Add auth data to**: Choose Request Headers or Request URL.
  + **Grant Type**: Select the appropriate grant type (e.g., Authorization Code, Client Credentials).
  + **Callback URL**: Enter the redirect URI used in the authorization process.
  + **Auth URL**: The authorization URL where users authenticate.
  + **Access Token URL**: The URL to obtain the access token.
  + **Client ID**: Enter your client ID.
  + **Client Secret**: Enter your client secret.
  + **Scope**: Enter the scope of access required.
  + **State**: (Optional) A state parameter for CSRF protection.
  + Click on Get New Access Token to fetch and use the token.

# **Generating a Personal Access Token on GitHub**

1. **Log In to GitHub:**

* Go to [GitHub](https://github.com) and sign in with your credentials.

1. **Navigate to Developer Settings:**

* Click on your profile picture in the upper-right corner of the GitHub page.
* Select **Settings** from the dropdown menu.
* In the left sidebar, click on **Developer settings**.

1. **Access Personal Access Tokens:**

* In the Developer settings menu, select Personal access tokens.

1. **Generate a New Token:**

* Click the Generate new token button.

1. **Configure the Token:**

* **Note**: Give your token a descriptive name in the Note field (e.g., "Postman API Access").
* **Expiration**: Choose an expiration date for your token, or select No expiration for a permanent token (though it’s generally safer to set an expiration).
* **Select Scopes**: Check the boxes for the scopes/permissions you need. For example:
* repo (for full control of private repositories)
* user (for accessing user profile information)
* gist (for creating and managing gists)
* workflow (for interacting with workflows)

You can customize the scopes according to your needs, but make sure to select only those that are necessary for your tasks.

1. **Generate and Copy the Token:**

* After configuring the token, click the Generate token button at the bottom of the page.
* **Important**: Copy the token immediately and store it securely. You won’t be able to see it again once you navigate away from the page.

1. **Use the Token:**

* You can use this token in your API requests by setting it as a Bearer token in the Authorization header or by including it as a parameter in your API requests.

# OAuth WorkFlow

