Document your APIs in Postman

You can add documentation to any Postman Collection and its API requests. You can also use Postman to create API documentation and share it with your API's consumers, so they know what endpoints are available and how to interact with them.

Postman automatically generates documentation for your collections and APIs.

Write Documentation

https://learning.postman.com/docs/publishing-your-api/authoring-your-documentation/

Publish Documentation

https://learning.postman.com/docs/publishing-your-api/publishing-your-docs/

- Publishing your documentation makes it publicly available to anyone with the link to the documentation.
- Publish your documentation to help people around the world learn how to use your collection or interact with your public API.
- Public documentation automatically includes details for each request or endpoint in the published collection, along with sample code in various client languages.
- As you update your collection, the published documentation automatically stays in sync with your latest changes. There's no need to publish the documentation again after making changes.

To publish the documentation for a collection, do the following:

1. Select **Collections** in the sidebar.
2. Select the more actions icon ⁰⁰⁰ next to a collection, and then select **View Documentation**.
3. Select **Publish** to navigate to the **Publish Documentation** interface.

After publishing get the link.

Example: https://galactic-shuttle-999073.postman.co/workspace/2c3a7fbd-3f73-4526-96f0-4ce5c86df0ee/overview

Swagger

Swagger is an open-source framework used for **designing**, **building**, **documenting**, and **consuming RESTful APIs**. It **simplifies the process of creating APIs** by providing a standardized and **interactive interface for API documentation**.

Swagger helps developers and testers **understand the capabilities, endpoints, and behavior of an API**.

Key Components of Swagger:

1. **Swagger UI:** An interactive and visual web interface for exploring and testing APIs directly in the browser.
2. **Swagger Editor:** A web-based editor for designing API specifications using YAML or JSON.
3. **Swagger Codegen:** A tool to generate client libraries, server stubs, and API documentation in various programming languages.
4. **OpenAPI Specification (OAS):** A standard format for describing RESTful APIs. Swagger uses OAS to document APIs.

Why is Swagger Needed for API Testers?

Swagger is particularly valuable for API testers for the following reasons:

1. Interactive Documentation:

- Swagger UI provides a detailed and visual representation of API endpoints, including request and response details, headers, and parameters. Testers can use this interface **to explore and understand the API**.

2. Easy Testing Without External Tools:

- Testers can send API requests directly from Swagger UI, making it easy to validate endpoints without relying on additional tools like Postman or Curl.

3. Improves Collaboration:

- Swagger serves as a single source of truth for developers, testers, and stakeholders, enabling effective communication about API functionality.

4. Error Identification:

- By testing endpoints through Swagger UI, testers can quickly identify issues like incorrect HTTP methods, invalid parameters, or unexpected responses.

8. Saves Time:

- Detailed and structured documentation saves time by eliminating the need to manually figure out API details. This allows testers to focus on test case design and validation.

Example Use Case:

Suppose you are testing an e-commerce API with endpoints for creating orders, retrieving products, and managing users. Using Swagger, you can:

1. View all endpoints and their respective request/response details in one place.
2. Test the API in real time to validate functionalities like product retrieval, order creation, or user login.
3. Use the Swagger Editor to analyze any updates to the API and adjust test cases accordingly.

Swagger documents

https://fakerestapi.azurewebsites.net/index.html

https://petstore.swagger.io/

https://httpbin.org/#/

Swagger Vs Postman

Feature	Swagger	Postman
Testing Features	Manual testing only	Automated and manual testing
Assertions	No built-in support for assertions	Supports assertions with JavaScript
Dynamic Data	Limited dynamic data handling	Supports dynamic variables and workflows
Reporting	No built-in reporting	Provides detailed test reports and logs
Mock APIs	No built-in API mocking	Built-in mock server support
User Interface	Developer-focused	More user-friendly, suitable for testers
Authentication	Basic support for authentication	Advanced support for complex authentication flows
Testing Complexity	Limited to simple requests	Supports complex test scenarios
Best For	API documentation and exploration	Comprehensive API testing, automation

cURL - Client URL

cURL (Client URL) is a **command-line tool** and library for transferring data with URLs. It supports a wide range of protocols, including **HTTP**, **HTTPS**, **FTP**, and more.

cURL allows you to send requests to APIs and receive responses, making it a useful tool for testing and interacting with APIs.

cURL allows you to easily send **GET**, **POST**, **PUT**, **DELETE**, and other types of HTTP requests to APIs directly from the command line.

Simple cURL commands for API testing:

1. GET Request – Fetch Data from an API

To retrieve data from an API (e.g., getting user information from a REST API):

```
curl https://api.example.com/users/1
```

This sends a **GET** request to the API endpoint https://api.example.com/users/1 and fetches the user with ID 1.

2. POST Request – Send Data to an API

To send data (like creating a new user):

```
curl -X POST https://api.example.com/users \
-H "Content-Type: application/json" \
-d '{"name": "John Doe", "email": "john@example.com"}'
```

- -X POST: Specifies the POST request method.
- -H: Adds a header, in this case, setting the content type to JSON.
- -d: Sends the JSON data in the request body.

3. PUT Request – Update Data

To update a user's information (e.g., updating user 1):

```
curl -X PUT https://api.example.com/users/1 \
-H "Content-Type: application/json" \
-d '{"name": "John Smith", "email": "john.smith@example.com"}'
```

This sends a **PUT** request to update user 1's details.

4. DELETE Request – Remove Data

To delete a user from the system:

```
curl -X DELETE https://api.example.com/users/1
```

This sends a **DELETE** request to remove user 1.

5. Including Headers for Authentication

If the API requires an authentication token:

```
curl -X GET https://api.example.com/users/1 \
-H "Authorization: Bearer YOUR_ACCESS_TOKEN"
```

This includes an **Authorization header** with a Bearer token to authenticate the request.

6. View Response Headers

To view both the response body and headers:

```
curl -i https://api.example.com/users/1
```

The -i flag includes the response headers in the output.

Benefits for API Testers:

- **Lightweight**: It doesn't require installing heavy tools or setting up complex environments.
- **Fast**: Quickly test endpoints with minimal overhead.
- **Portable**: Available on most operating systems (Windows, Linux, macOS) and can be used in CI/CD pipelines.

In summary, **cURL** is a versatile tool that helps API testers quickly and effectively test API endpoints through the command line. It's great for **simple tests**, **automation**, and **debugging**.

Example:

```
curl -X GET "https://fakerestapi.azurewebsites.net/api/v1/Books" -H
"accept: text/plain; v=1.0"
```

We can send http request in command prompt using Curl.