# Flask RESTful API

## 🚀 Get started here

This template guides you through CRUD operations (GET, POST, PUT, DELETE), variables, and tests.

### 🩹 How to use this template

#### Step 1: Send requests

RESTful APIs allow you to perform CRUD operations using the POST, GET, PUT, and DELETE HTTP methods.

This collection contains each of these request types. Open each request and click "Send" to see what happens.

#### Step 2: View responses

Observe the response tab for status code (200 OK), response time, and size.

#### Step 3: Send new Body data

Update or add new data in "Body" in the POST request. Typically, Body data is also used in PUT request.

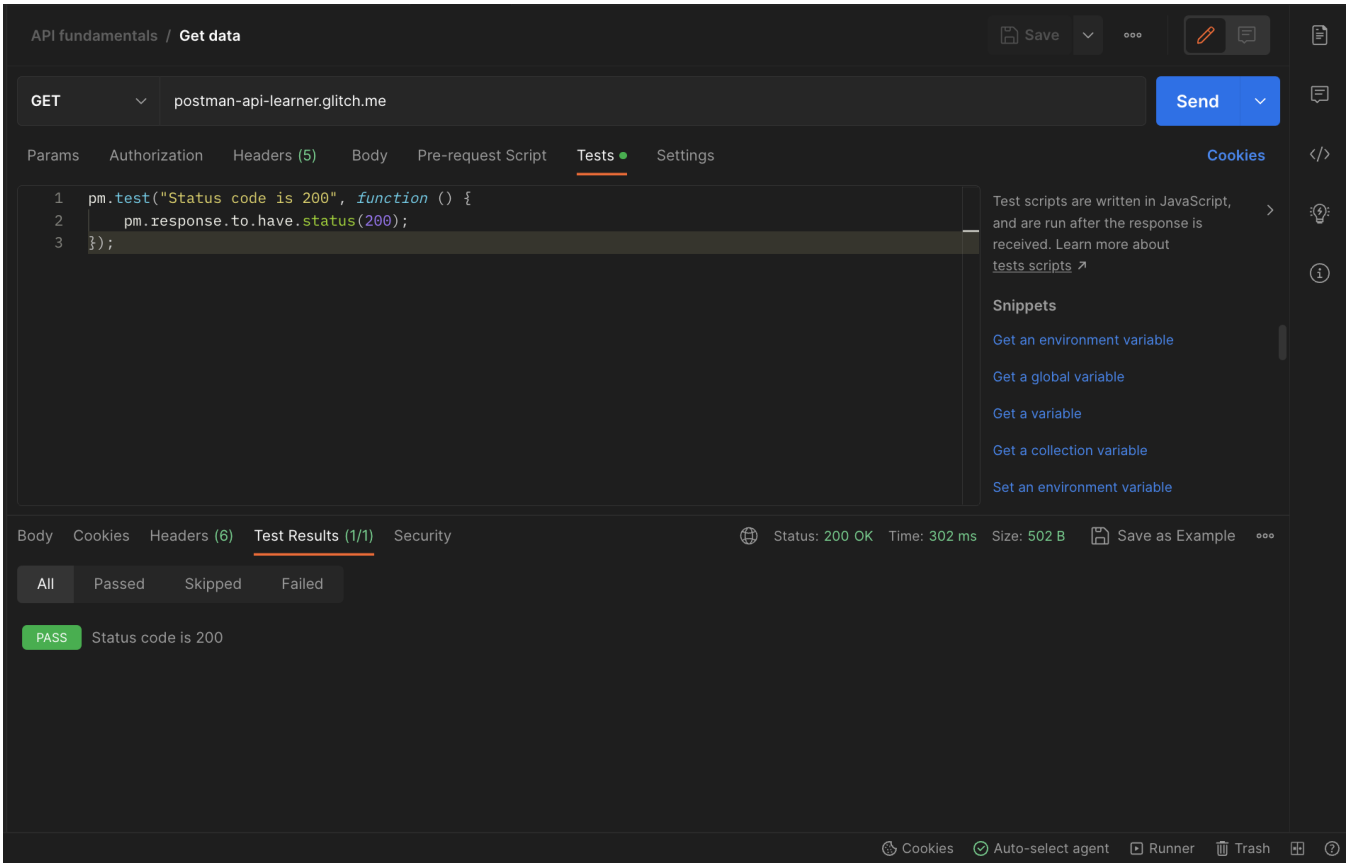
```
Plain Text

{
    "name": "Add your name in the body"
}
```

#### Step 4: Update the variable

Variables enable you to store and reuse values in Postman. We have created a variable called `base_url` with the sample request https://postman-api-learner.glitch.me. Replace it with your API endpoint to customize this collection.

#### Step 5: Add tests in the "Tests" tab

Tests help you confirm that your API is working as expected. You can write test scripts in JavaScript and view the output in the "Test Results" tab.




## 💪 Pro tips

- Use folders to group related requests and organize the collection.
- Add more scripts in "Tests" to verify if the API works as expected and execute flows.

## ℹ️ Resources

Building requests
Authorizing requests
Using variables
Managing environments
Writing scripts

### AUTHORIZATION Bearer Token

Token ••••••••

---

## GET Home Page

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1`).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection Flask RESTful API

---

## POST User Login

/login

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1`).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection Flask RESTful API

### Body raw (json)

**json**

```
{
    "username": "•••••••",
    "password": "•••••••"
}
```

---

## GET Tasks

/tasks

This is a GET request and it is used to "get" data from an endpoint. There is no request body for a GET request, but you can use query parameters to help specify the resource you want data on (e.g., in this request, we have `id=1`).

A successful GET response will have a `200 OK` status, and should include some kind of response body - for example, HTML web content or JSON data.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection Flask RESTful API

---

## POST Create task

/create-task

This is a POST request, submitting data to an API via the request body. This request submits JSON data, and the data is reflected in the response.

A successful POST request typically returns a `200 OK` or `201 Created` response code.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection Flask RESTful API

### HEADERS

| | |
|---|---|
| **Content-Type** | application/json |

### Body raw

```
{
    "title": "Task Title",
    "description": "Task Description",
    "due_date": "2023-12-31",
    "priority": "High",
    "status": "pending",
    "user_id": 1
}
```

---

## PUT Update task

/update-task/1

This is a PUT request and it is used to overwrite an existing piece of data. For instance, after you create an entity with a POST request, you may want to modify that later. You can do that using a PUT request. You typically identify the entity being updated by including an identifier in the URL (eg. `id=1`).

A successful PUT request typically returns a `200 OK`, `201 Created`, or `204 No Content` response code.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection Flask RESTful API

### Body raw (json)

**json**

```
{
    "title": "Updated Task Title",
    "description": "Updated Task Description",
```

```
    "due_date": "2023-12-31",
    "priority": "High",
    "status": "pending",

    "user_id": 1
}
```


---

## DELETE Delete task

/delete-task/1

This is a DELETE request, and it is used to delete data that was previously created via a POST request. You typically identify the entity being updated by including an identifier in the URL (eg. `id=1`).

A successful DELETE request typically returns a `200 OK`, `202 Accepted`, or `204 No Content` response code.

### AUTHORIZATION Bearer Token

This request is using Bearer Token from collection Flask RESTful API