

The screenshot shows the Postman application interface. On the left, the sidebar displays 'sana786q's Workspace' with collections like 'Collection-1' and 'Testing\_Flask\_App'. Under 'Testing\_Flask\_App', several requests are listed: 'GET1' (selected), 'GET\_Users', 'GET\_UsersByID', 'POST\_POST\_N', 'PUT\_PUT\_Data', 'PATCH\_Patch\_Data', and 'DELETE\_Delete\_Data'. The main panel shows a request for 'GET1' with the URL 'http://127.0.0.1:5000/'. The 'Body' tab shows the response: 'Welcome to User API'. The status bar at the bottom indicates '200 OK', '93 ms', and '193 B'.

The screenshot shows the Postman application interface again. The sidebar is identical. The main panel now shows a request for 'GET\_Users' with the URL 'http://127.0.0.1:5000/users'. The 'Body' tab shows the response as JSON: '[{"id": 1, "name": "Sana"}, {"id": 2, "name": "Rashmi"}, {"id": 3, "name": "Rupali"}, {"id": 4, "name": "Rani"}]'. The status bar at the bottom indicates '200 OK', '8 ms', and '338 B'.

This screenshot is identical to the one above, showing the 'GET\_Users' request and its JSON response. The status bar at the bottom indicates '200 OK', '8 ms', and '338 B'.

The screenshot shows the Postman application interface. On the left, the sidebar displays 'sana786q's Workspace' with collections like 'Collection-1' and 'Testing\_Flask\_App'. Under 'Testing\_Flask\_App', several API endpoints are listed: 'GET1', 'GET\_Users', 'GET\_UsersByID', 'POST\_NewUser' (which is selected), 'PUT\_PUT\_Data', 'PATCH\_Patch\_Data', and 'DEL\_Delete\_Data'. The main workspace shows a 'Testing\_Flask\_App / GET\_UsersByID' request. The method is set to 'GET' and the URL is 'http://127.0.0.1:5000/users/1'. The 'Params' tab is active, showing a single parameter 'Key' with the value 'Value'. Below the request, the 'Body' tab shows the response: a JSON object with 'id': 1 and 'name': 'Sana'. The status bar at the bottom indicates '200 OK' with a response time of 8 ms and a size of 198 B.

The screenshot shows the Postman application interface. The sidebar and workspace are identical to the first one. A new request is being prepared under 'Testing\_Flask\_App / POST\_NewUser'. The method is set to 'POST' and the URL is 'http://127.0.0.1:5000/users'. The 'Body' tab is active, showing a JSON payload with 'name': 'Rani'. The status bar at the bottom indicates '201 CREATED' with a response time of 196 ms and a size of 203 B.

This screenshot is identical to the previous one, showing the same workspace and request details. It also shows the successful creation of a new user 'Rani' with a response of '201 CREATED'.

The image consists of three vertically stacked screenshots of the Postman application interface, demonstrating the use of the PUT and PATCH methods on a Flask API.

**Screenshot 1: PUT Request**

In this screenshot, a PUT request is being made to `http://127.0.0.1:5000/users/4`. The request body is set to raw JSON:

```
1 {  
2   "name": "Rani Updated"  
3 }
```

The response status is 200 OK, with a response time of 20 ms, a response size of 206 B, and a timestamp of 18:21 21-01-2026. The response body is:

```
1 {  
2   "id": 4,  
3   "name": "Rani Updated"  
4 }
```

**Screenshot 2: PATCH Request**

In this screenshot, a PATCH request is being made to `http://127.0.0.1:5000/users/4`. The request body is set to raw JSON:

```
1 {  
2   "name": "Rani Updated twice"  
3 }
```

The response status is 200 OK, with a response time of 13 ms, a response size of 212 B, and a timestamp of 18:21 21-01-2026. The response body is:

```
1 {  
2   "id": 4,  
3   "name": "Rani Updated twice"  
4 }
```

**Screenshot 3: Application Taskbar**

The bottom of the image shows a taskbar with several application icons, including Microsoft Edge, Google Chrome, Mozilla Firefox, and others. The system tray shows the date and time as 18:21 21-01-2026.

Home Workspaces API Network

sana786q's Workspace

New Import

Overview GET GET1 GET GET\_User POST POST\_N GET GET\_User PUT PUT\_Data PATCH Patch\_F DEL Delete\_Dr + - No environment X

Collections

+ Q Search collections

> Collection-1

> Testing\_Flask\_App

GET GET1

GET GET\_Users

GET GET\_UsersByID

POST POST\_NewUser

PUT PUT\_Data

PATCH Patch\_Data

DEL Delete\_Data

ENV Environment

History

Flows

Cloud View Find and replace Console Terminal

Upcoming Earnings

Search

Ctrl K

invite

Upgrade

Save Share

Send

DELETE http://127.0.0.1:5000/users/4

Docs Params Authorization Headers (8) Body Scripts Tests Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Body Cookies Headers (5) Test Results

200 OK 7 ms 211 B Save Response

{ } JSON Preview Visualize

```
1: {  
2:   "message": "User deleted successfully"  
3: }
```

Runner Start Proxy Cookies Vault Trash ENG IN 21-01-2026 18:21

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Collections' (Collection-1, Testing\_Flask\_App), 'Environments', 'History', and 'Flows'. The main workspace shows a 'Testing\_Flask\_App' collection with various API endpoints like GET1, GET\_Users, etc. A specific 'Delete\_Data' endpoint is selected. The 'Params' tab is active, showing a single parameter 'Key' with value 'Value'. Below it, the 'Body' tab shows a JSON response: { "message": "User deleted successfully" }. The bottom status bar indicates a 200 OK response with 7 ms latency and 211 B size.