

# Algorithmic and advanced Programming in Python

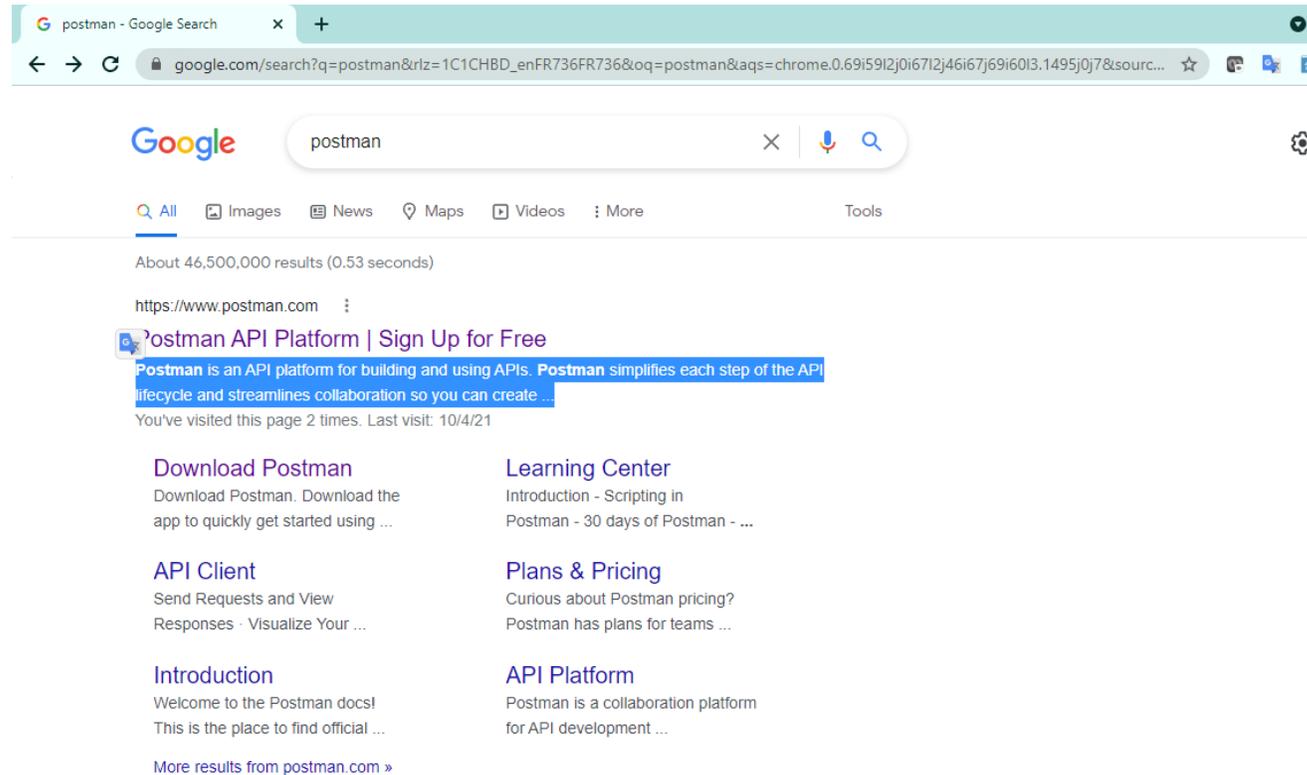
Eric Benhamou [eric.benhamou@dauphine.eu](mailto:eric.benhamou@dauphine.eu)  
Chien-Chung.Huang [chien-chung.huang@ens.fr](mailto:chien-chung.huang@ens.fr)  
Sofía Vázquez [sofia.vazquez@dauphine.eu](mailto:sofia.vazquez@dauphine.eu)



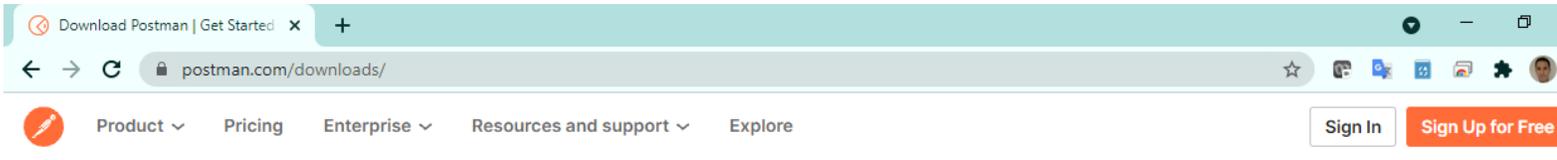
# Let us play manually with postman

**Postman** is an API platform for building and using APIs. **Postman** simplifies each step of the API lifecycle and streamlines collaboration so you can create easily REST request

# Download and install postman



# Download the app 1/2



## Download Postman

Download the app to quickly get started using the Postman API Platform. Or, if you prefer a browser experience, you can try the new web version of Postman.

### The Postman app

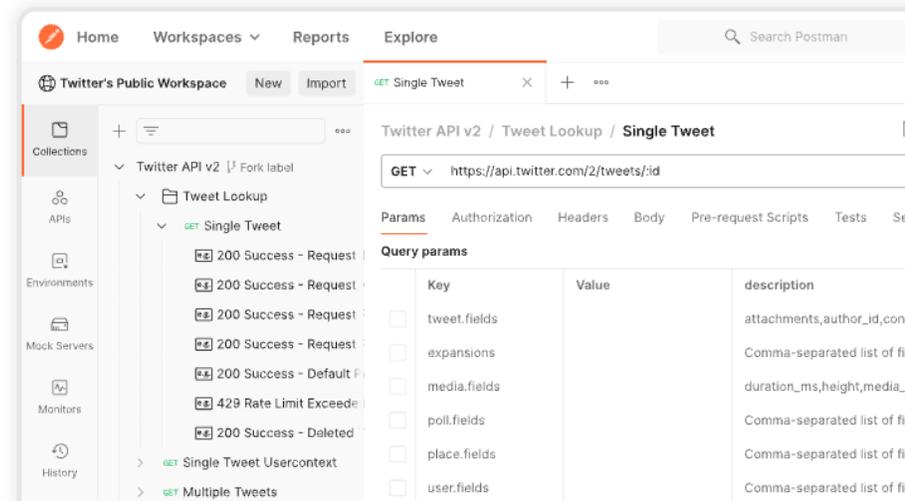
The ever-improving Postman app (a new release every two weeks) gives you a full-featured Postman experience.

[Download the App](#)

By downloading and using Postman, I agree to the [Privacy Policy](#) and [Terms](#).

Version 8.12.4 · [Release Notes](#) · [Product Roadmap](#)

Not your OS? Download for Mac ([macOS](#)) or Linux ([x64](#))



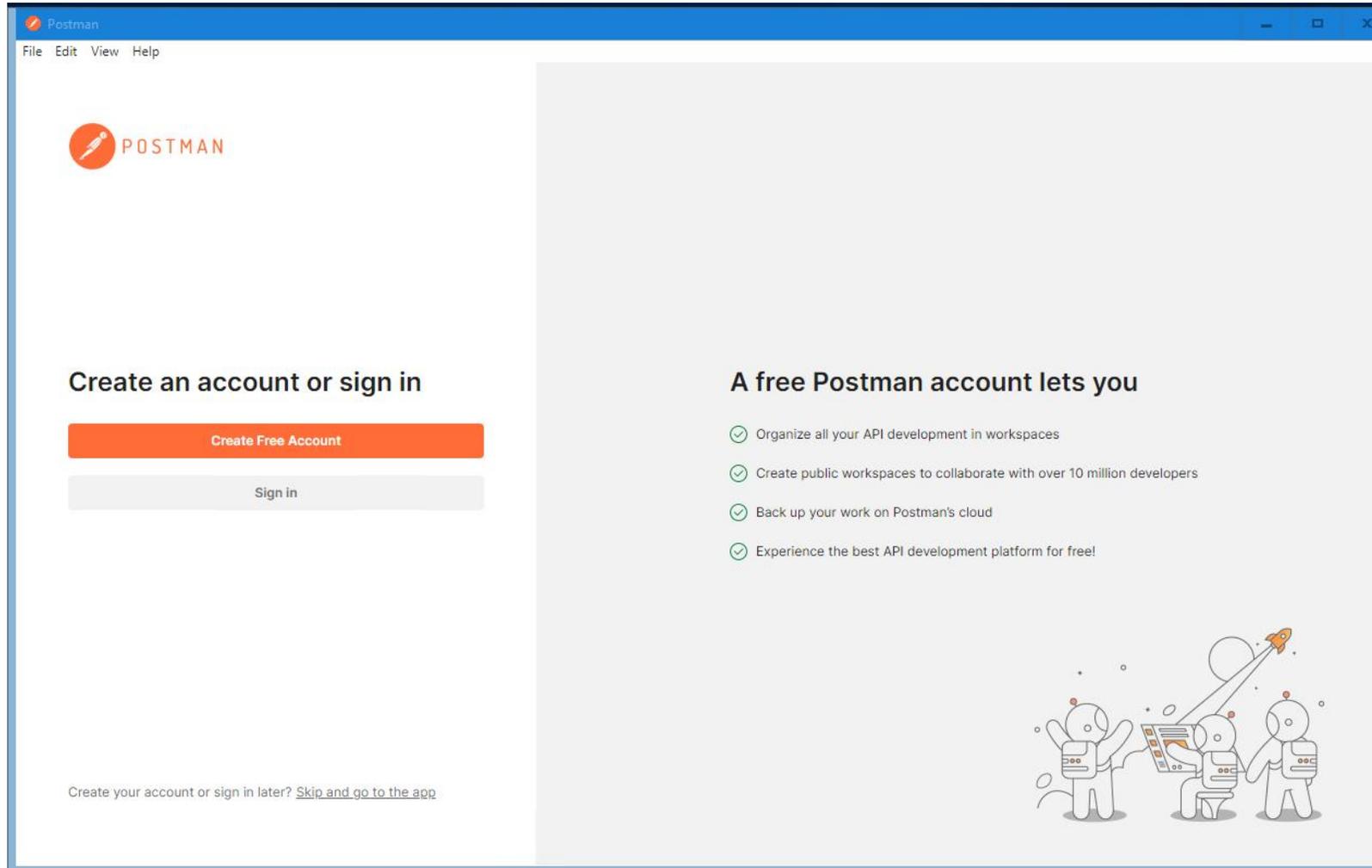
# Download the app 2/2

The screenshot shows the Postman website's download page. The main heading is "The Postman app" with a sub-heading "The ever-improving Postman app (a new release every two weeks) gives you a full-featured Postman experience." Below this is a prominent orange "Download the App" button. Underneath, it says "Select the version you want to download" and lists "Windows 32-bit" and "Windows 64-bit". A blue arrow points from the "Download the App" button to the right, where a vertical banner for "DB Browser" is visible. The background of the screenshot shows a Postman workspace with a "Single Tweet" collection and a "GET" request to the Twitter API.

In your desktop



# Create an account



# Create your first post request

The screenshot shows the Postman application interface. At the top, there is a menu bar with 'Postman', 'File', 'Edit', 'View', and 'Help'. Below the menu bar, there are navigation tabs: 'Home', 'Workspaces', 'Reports', and 'Explore'. A search bar labeled 'Search Postman' is located to the right of these tabs. In the top right corner, there are several utility icons, including a green refresh icon, a blue 'Invite' button, a share icon, a settings gear, a notification bell, and a red circular icon. Below the navigation bar, the main content area is titled 'My Workspace'. On the left side of this area, there are two buttons: 'New' and 'Import', both of which are circled in red. The 'New' button is highlighted with a red circle. The main content area displays a summary of the workspace, including a description, a list of collections and APIs, and an activity log. The activity log shows recent actions such as 'Eric edited the Flask API collection' and 'Eric added the Flask API collection'. On the right side of the workspace, there is a 'Get started' section with links to 'Create a request', 'Create a collection', 'Create an API', and 'Create an environment'. Below this, there is a 'Sharing' section with a 'Visibility' dropdown menu set to 'Personal'.

Postman  
File Edit View Help

Home Workspaces Reports Explore Search Postman Invite

My Workspace New Import Over... X GET G. GET G. R. GET G. [CONFL No Environment

My Workspace

Add summary to briefly explain what this workspace is all about...

This is your personal, private workspace to play around in. Only you can see the collections and APIs you create here - unless you share them with your team.

In this workspace

4	0	0	0	0
Collections	APIs	Environments	Mock Servers	Monitors

Activity

User Entity Refresh

Today

- Eric edited the Flask API collection. View Changelog  
8 mins ago
- Eric shared the Flask API collection to this workspace  
11 mins ago
- Eric added the Flask API collection  
11 mins ago

Get started

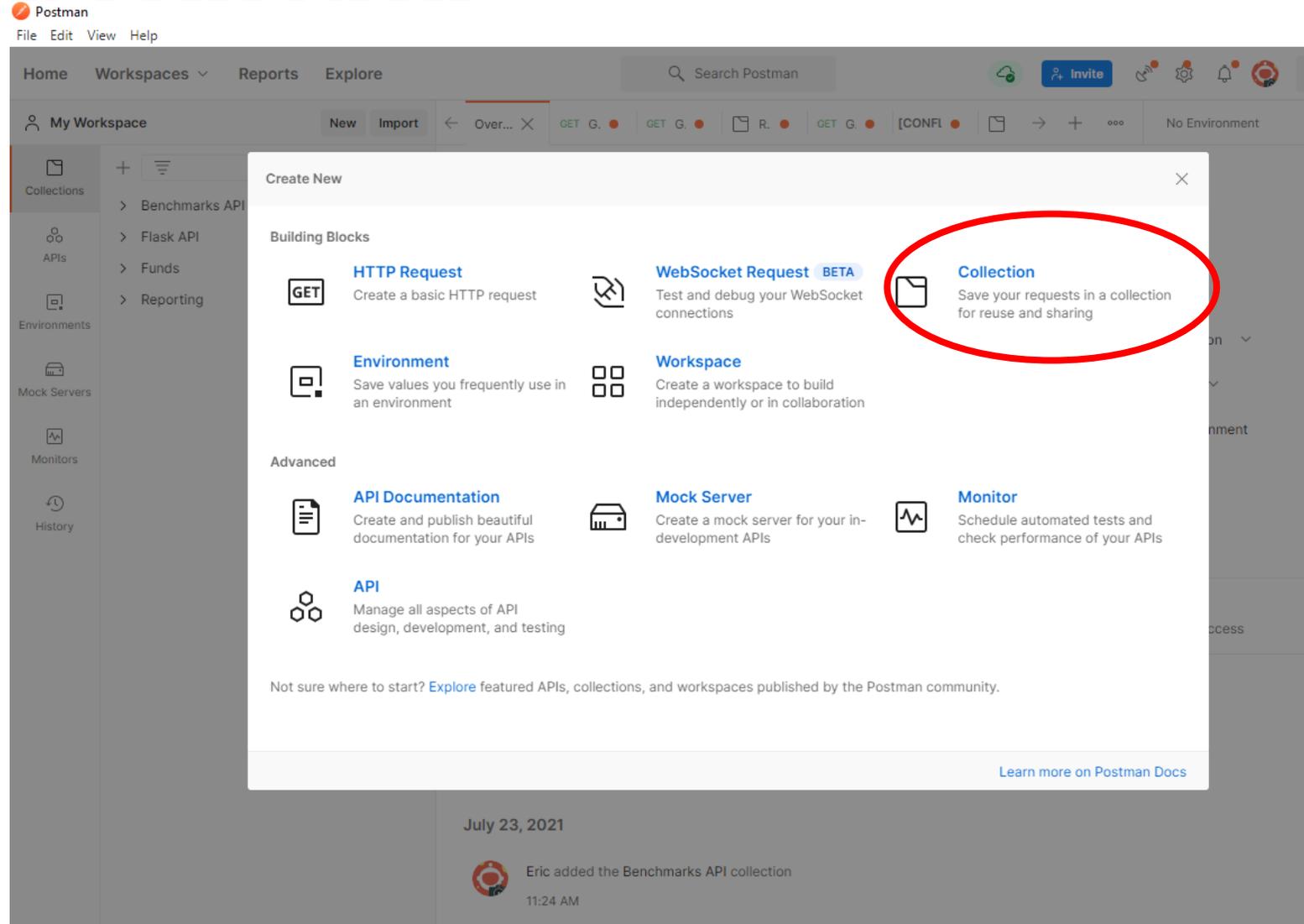
- Create a request
- Create a collection
- Create an API
- Create an environment
- View More

Sharing

Visibility

Personal Only you can access

# Create a collection



# Give a name to your collection

The screenshot displays the Postman application interface. At the top, there is a menu bar with 'File', 'Edit', 'View', and 'Help'. Below the menu bar, there are navigation tabs for 'Home', 'Workspaces', 'Reports', and 'Explore', along with a search bar labeled 'Search Postman'. The main workspace area is titled 'My Workspace' and contains a sidebar on the left with categories: 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. Under 'Collections', there is a sub-section 'Benchmarks API' with several endpoints listed. The 'Flask API' collection is highlighted with a red circle. The main content area shows the details for the 'Flask API' collection, including tabs for 'Auth', 'Pre-req.', 'Tests', and 'Variables'. The 'Auth' tab is selected, showing a message: 'This authorization method will be used for every request in this collection. You can override this by specifying one in the request.' Below this, there is a 'Type' dropdown menu set to 'No Auth'. At the bottom, there is a note: 'This collection does not use any authorization. [Learn more about authorization](#)'.

# Add a request

The screenshot displays the Postman application interface. At the top, the 'Postman' logo and menu items (File, Edit, View, Help) are visible. Below the menu, there are navigation tabs for 'Home', 'Workspaces', 'Reports', and 'Explore', along with a search bar labeled 'Search Postman'. The main workspace is titled 'My Workspace' and contains a sidebar with navigation options: Collections, APIs, Environments, Mock Servers, Monitors, and History. The 'Collections' sidebar is expanded to show a tree view with 'Benchmarks API' and 'Flask API'. The 'Flask API' collection is selected and highlighted, and its content is displayed in the main panel. The content shows a list of endpoints: 'PUT Set benchmark composition', 'GET Get benchmark', 'GET Get all benchmarks', and 'GET Get benchmark composition'. Below this list, a message states 'This collection is empty' and a blue link 'Add a request to start working.' is circled in red. The right-hand panel shows the 'Auth' tab selected, with a dropdown menu set to 'No Auth' and a note stating 'This collection does not use any authorization' with a link to 'Learn more about authorization'.

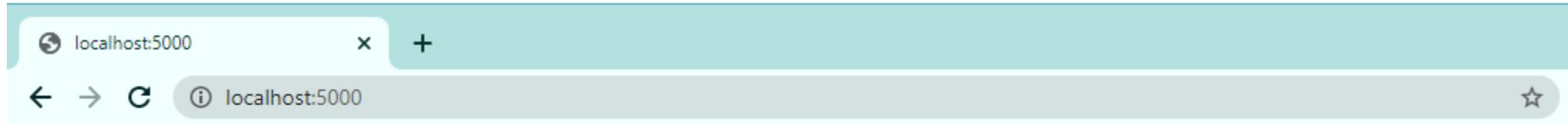
# Run your server

- Run 3.4 create routes.py in python or in spyder or in pycharm
- Ensure that you have added a function to see if this is running like this

```
@app.route("/")  
def home():  
    return "<h1>API Running</h1>"
```

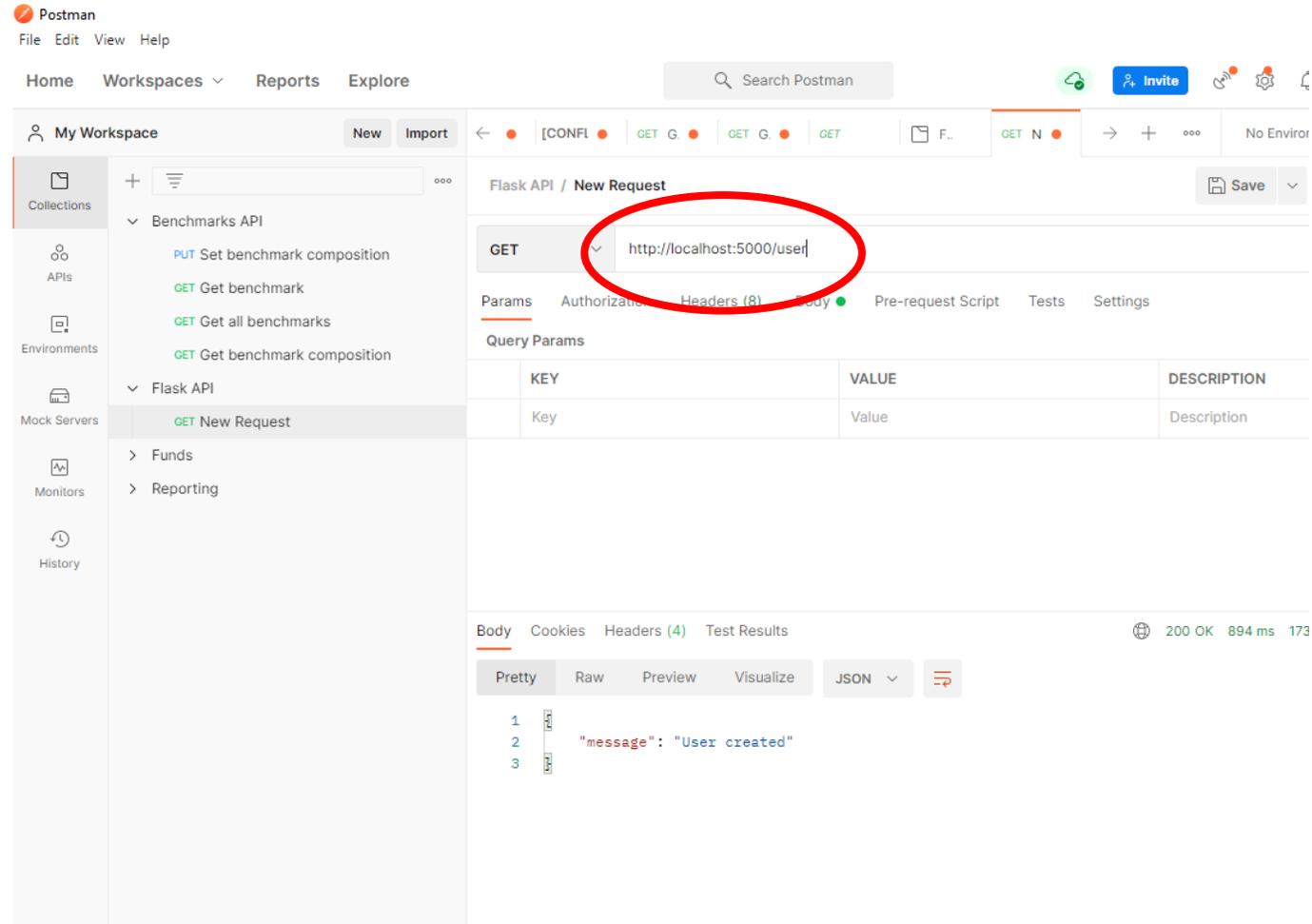
- You should see that the local host is working and get something like

# Ensure the API server is running

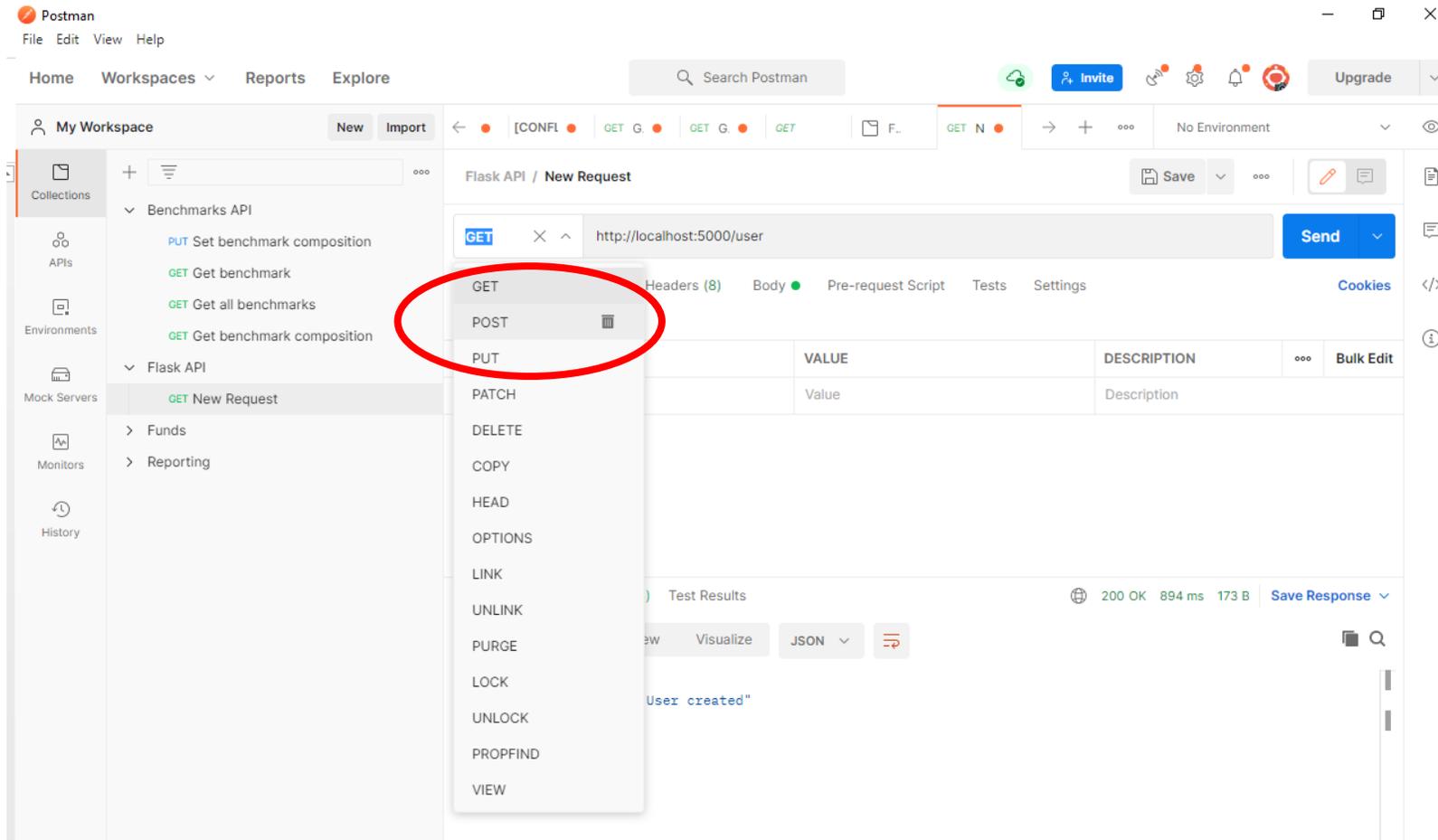


**API Running**

# Type the correct url in postman



# Change the GET to POST request



# Say post

The screenshot shows the Postman application interface. The main workspace is titled "Flask API / New Request". The URL bar contains "http://localhost:5000/user". The HTTP method is set to "POST", which is circled in red. The interface includes a sidebar with "Collections", "APIs", "Environments", "Mock Servers", "Monitors", and "History". The top navigation bar includes "Home", "Workspaces", "Reports", and "Explore". The bottom section shows tabs for "Params", "Authorization", "Headers (8)", "Body", "Pre-request Script", "Tests", and "Settings". A "Query Params" table is visible below the tabs.

KEY	VALUE	DESCRIPTION	...	Bulk E
Key	Value	Description		

# Goto body

The screenshot displays the Postman application interface. At the top, there's a menu bar with 'File', 'Edit', 'View', and 'Help'. Below it, navigation tabs include 'Home', 'Workspaces', 'Reports', and 'Explore'. A search bar labeled 'Search Postman' is present. The main workspace is titled 'My Workspace' and contains a sidebar with 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The 'APIs' section is expanded to show 'Benchmarks API' and 'Flask API'. The 'Flask API' collection is selected, showing a 'New Request' endpoint. The request configuration is set to 'POST' with the URL 'http://localhost:5000/user'. The 'Body' tab is selected and circled in red, showing a JSON body: 

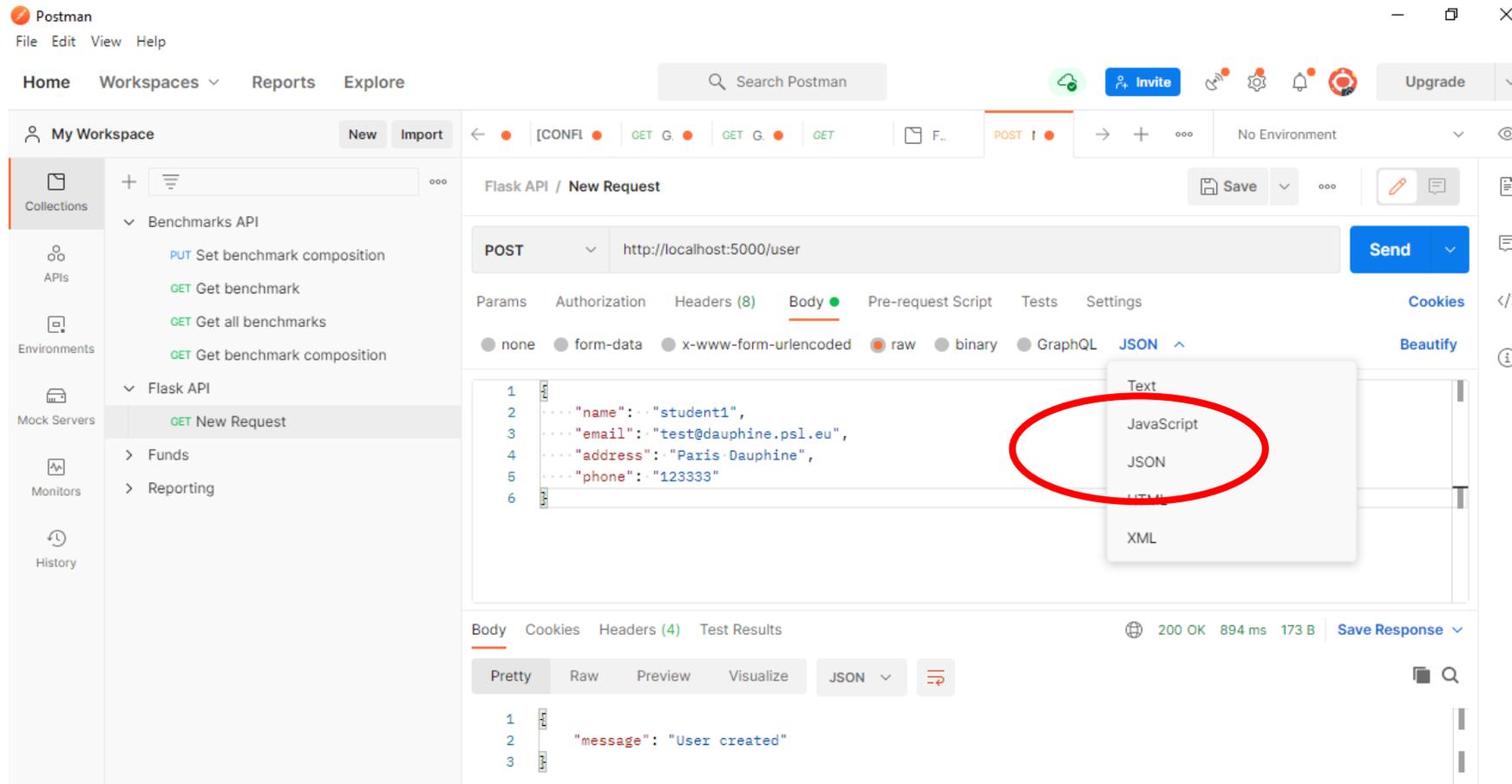
```
1 {
2   "name": "student1",
3   "email": "test@dauphine.psl.eu",
4   "address": "Paris Dauphine",
5   "phone": "123333"
6 }
```

 Below the request configuration, the response is shown in the 'Body' tab, displaying a JSON object: 

```
1 {
2   "message": "User created"
3 }
```

 The status bar at the bottom indicates a '200 OK' response with a time of '894 ms' and a size of '173 B'.

# Select Json



# Enter the following input

```
{  
  "name": "student1",  
  "email": "test@dauphine.psl.eu",  
  "address": "Paris Dauphine",  
  "phone": "123333"  
}
```

The screenshot shows the Postman application interface. The main workspace displays a 'New Request' for a 'POST' method to the URL 'http://localhost:5000/user'. The 'Body' tab is selected, and the request body is a JSON object: 

```
{ "name": "student1", "email": "test@dauphine.psl.eu", "address": "Paris Dauphine", "phone": "123333" }
```

. This JSON body is circled in red. A dropdown menu is open over the body, showing options: Text, JavaScript, JSON, HTML, and XML. The left sidebar shows a collection named 'Flask API' with a 'New Request' item selected. The top navigation bar includes 'Home', 'Workspaces', 'Reports', 'Explore', and a search bar.

# Send the command

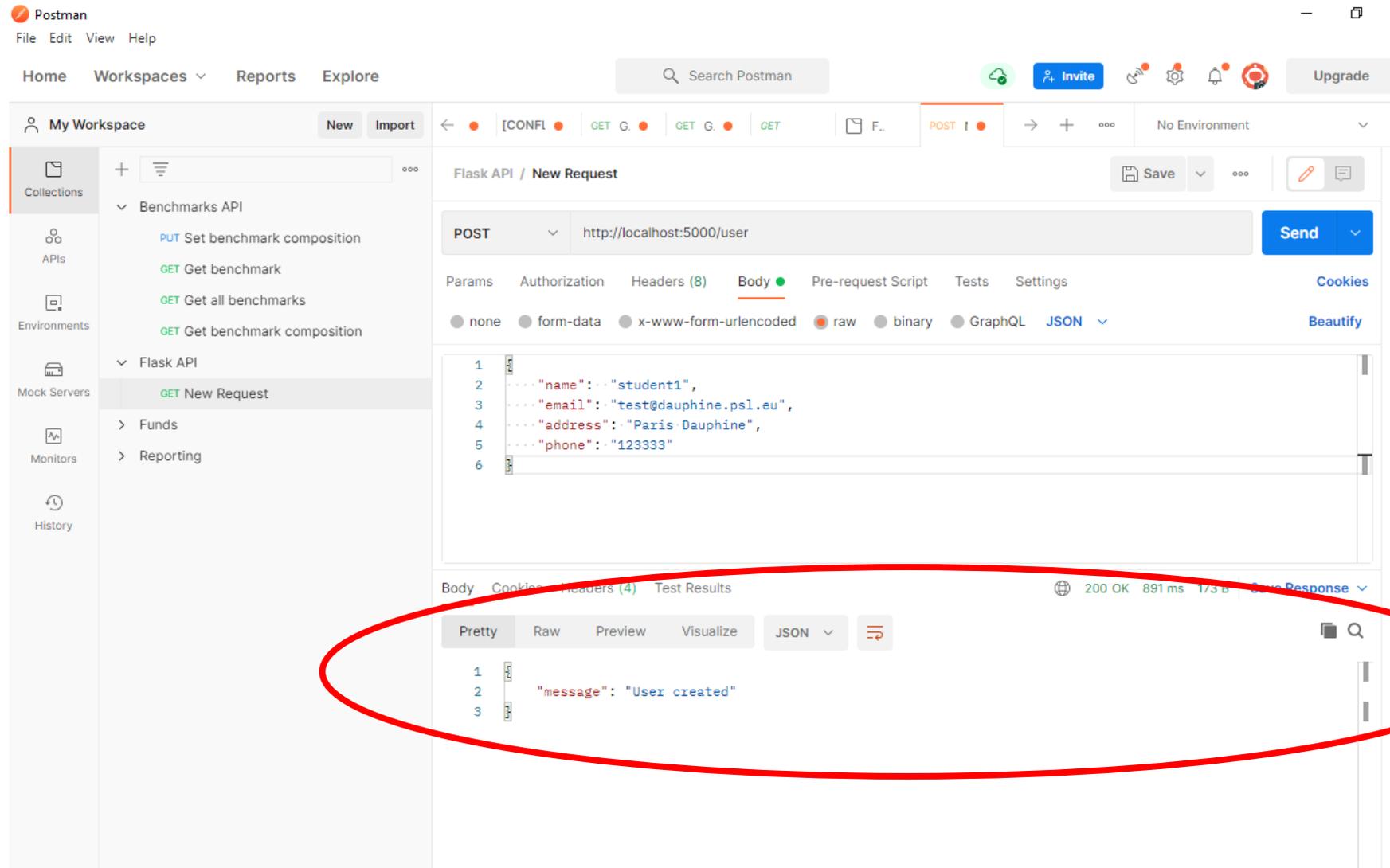
The screenshot shows the Postman application interface. At the top, there's a menu bar with 'File', 'Edit', 'View', and 'Help'. Below it, navigation tabs include 'Home', 'Workspaces', 'Reports', and 'Explore'. A search bar labeled 'Search Postman' is present. The main workspace is titled 'My Workspace' and contains a sidebar with 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The central area displays a 'New Request' for 'Flask API' with a 'POST' method and URL 'http://localhost:5000/user'. The request body is a JSON object: 

```
{  "name": "student1",  "email": "test@dauphine.psl.eu",  "address": "Paris Dauphine",  "phone": "123333"}
```

. The 'Send' button is circled in red. Below the request, the response is shown in 'Pretty' view: 

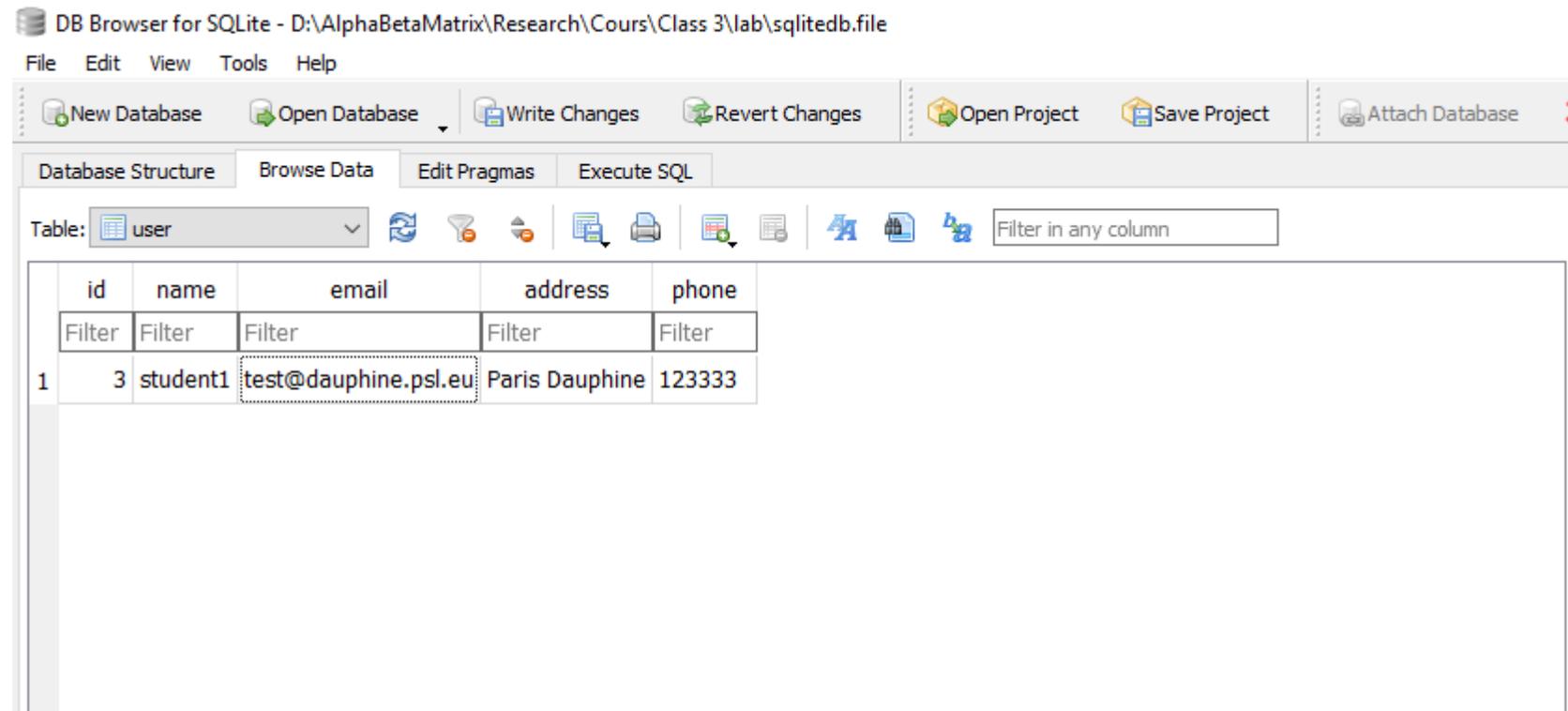
```
{  "message": "User created"}
```

# You should get something like



# Check that there is a record in the database with DB Browser

- Open DB Browser



# You are done!

# Add a few users!

The screenshot displays the Postman interface for a REST client. The main workspace shows a collection named "Flask API" with a sub-collection "Flask API" containing a "GET post user" endpoint. The selected endpoint is a POST request to "http://localhost:5000/user". The request body is a JSON object:

```
1 {
2   "name": "student3",
3   "email": "test3@dauphine.psl.eu",
4   "address": "Paris Dauphine",
5   "phone": "123333"
6 }
```

The response is displayed in the "Body" tab, showing a 200 OK status with a response time of 623 ms and a body of:

```
1 {
2   "message": "User created"
3 }
```

The interface includes a sidebar with navigation options like Collections, APIs, Environments, Mock Servers, Monitors, and History. The top navigation bar includes Home, Workspaces, Reports, and Explore. The bottom of the interface shows the response details, including status, time, and size, along with options to save the response and view it in different formats (Pretty, Raw, Preview, Visualize).

# Add a few users!

The screenshot shows the Postman interface with a REST client request configured. The request is a POST to `http://localhost:5000/user` with a JSON body:

```
1 {
2   "name": "student2",
3   "email": "test2@dauphine.psl.eu",
4   "address": "Paris Dauphine",
5   "phone": "123333"
6 }
```

The response is displayed in the bottom panel, showing a 200 OK status with a response time of 675 ms and a body of:

```
1 {
2   "message": "User created"
3 }
```

# Add a few users!

The screenshot displays the Postman interface for a REST client. The main workspace shows a collection named "Flask API" with a sub-collection "Flask API" containing a "GET post user" endpoint. The selected endpoint is a POST request to "http://localhost:5000/user". The request body is a JSON object: 

```
{ "name": "student3", "email": "test3@dauphine.psl.eu", "address": "Paris Dauphine", "phone": "123333" }
```

. The response is a 200 OK status with a response time of 623 ms and a body of 

```
{ "message": "User created" }
```

. The interface includes a sidebar with navigation options like Collections, APIs, Environments, and Mock Servers. The top navigation bar includes "Home", "Workspaces", "Reports", and "Explore".

# Call the linked list to get descending and ascending users

The screenshot shows the Postman interface with a REST client request configured. The request is a GET call to the endpoint `http://localhost:5000/user/descending_id`. The response body is a JSON array of three user objects, ordered by descending ID (5, 4, 3).

```
1 {
2   {
3     "address": "Paris Dauphine",
4     "email": "test3@dauphine.psl.eu",
5     "id": 5,
6     "name": "student3",
7     "phone": "123333"
8   },
9   {
10    "address": "Paris Dauphine",
11    "email": "test2@dauphine.psl.eu",
12    "id": 4,
13    "name": "student2",
14    "phone": "123333"
15  },
16  {
17    "address": "Paris Dauphine",
18    "email": "test@dauphine.psl.eu",
19    "id": 3,
20    "name": "student1",
21    "phone": "123333"
22  }
23 }
```