

Reading Material

Melakukan API Testing - API Testing menggunakan Postman



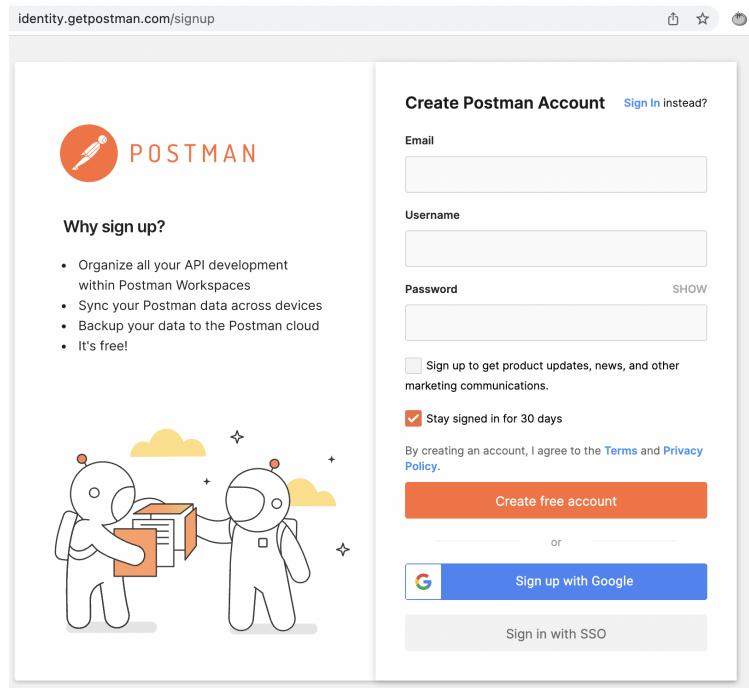
READING

Ketika membuat sebuah aplikasi berupa REST API atau GraphQL. Terdapat tools atau alat yang dapat digunakan untuk membantu dalam proses pengujian aplikasi yang telah dibuat. Salah satu tools tersebut adalah **Postman**. Postman merupakan salah satu tools yang paling populer untuk melakukan pengujian API. Postman muncul pada tahun 2012 sebagai proyek sampingan oleh **Abhinav Asthana** untuk menyederhanakan alur kerja API dalam pengujian dan pengembangan. Postman dirancang untuk memudahkan pekerjaan developer dengan membangun, menguji, dan memodifikasi API. Postman mengirim permintaan API ke server web dan menerima respons, dapat mengirimkan permintaan HTTP seperti GET, POST, PUT, dan DELETE ke endpoint API yang ditentukan dan bisa juga untuk mengirim dan menerima data dalam berbagai format seperti JSON dan XML. Postman dapat dijalankan melalui aplikasi atau melalui web.

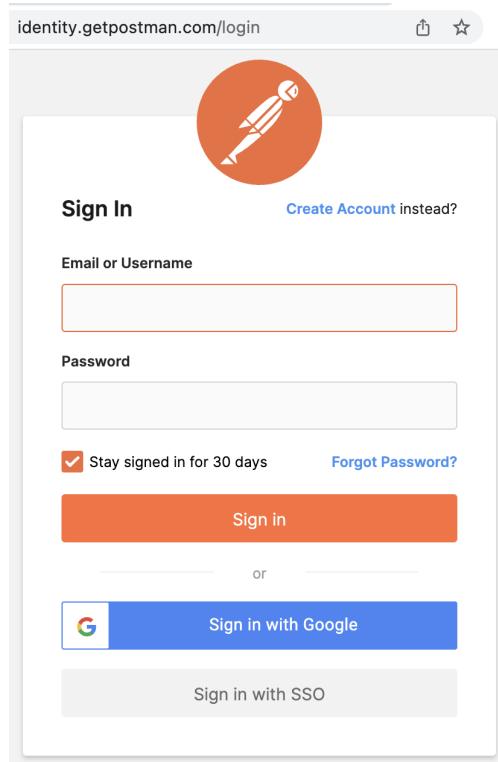
Registrasi dan Login

Jika belum memiliki akun, daftar melalui link berikut

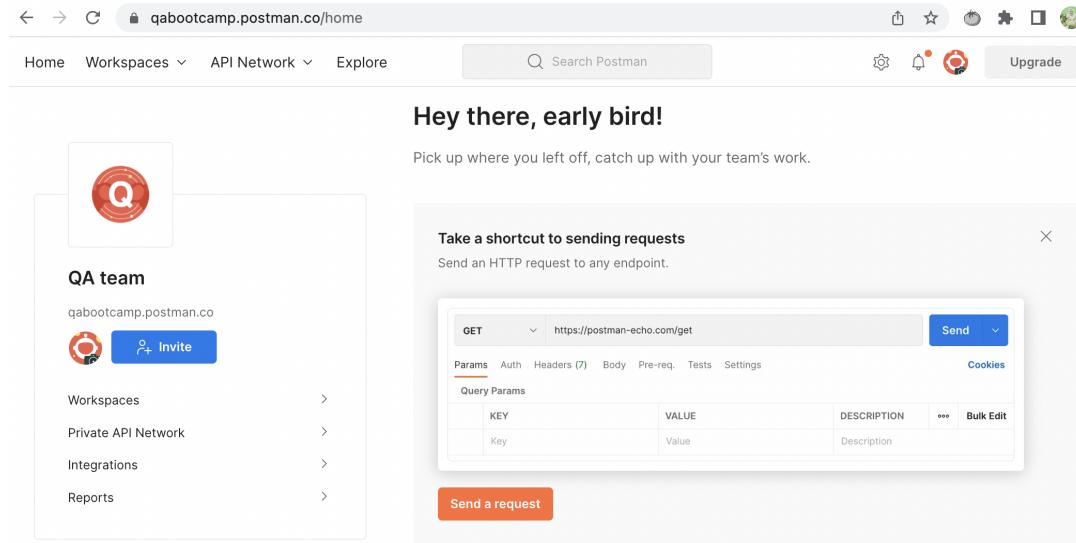
<https://identity.getpostman.com/signup>



Setelah melakukan registrasi, silahkan langsung login menggunakan data yang sudah diregistrasikan



Setelah Login, Team Name dan Team Domain dapat diganti jika diperlukan.



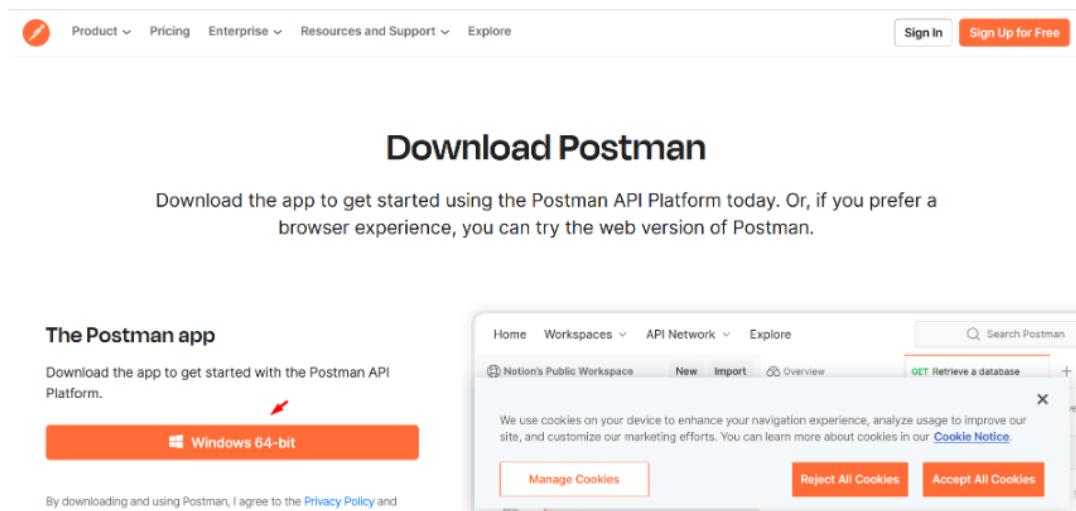
The screenshot shows the Postman web interface. On the left, there's a sidebar for the 'QA team' workspace, which includes links for Workspaces, Private API Network, Integrations, and Reports. A 'Send an invite' button is visible. The main area displays a 'Hey there, early bird!' message and a 'Take a shortcut to sending requests' modal. The modal shows a request configuration for a GET method to 'https://postman-echo.com/get'. It includes tabs for Params, Auth, Headers, Body, Pre-req., Tests, and Settings, along with a 'Cookies' tab. Below the configuration is a 'Send a request' button.

Instalasi

Selain web version, Postman juga menyediakan versi aplikasi yang dapat diunduh dan diinstal di dalam komputer. Aplikasi Postman dapat diunduh pada link berikut

<https://www.postman.com/downloads/>

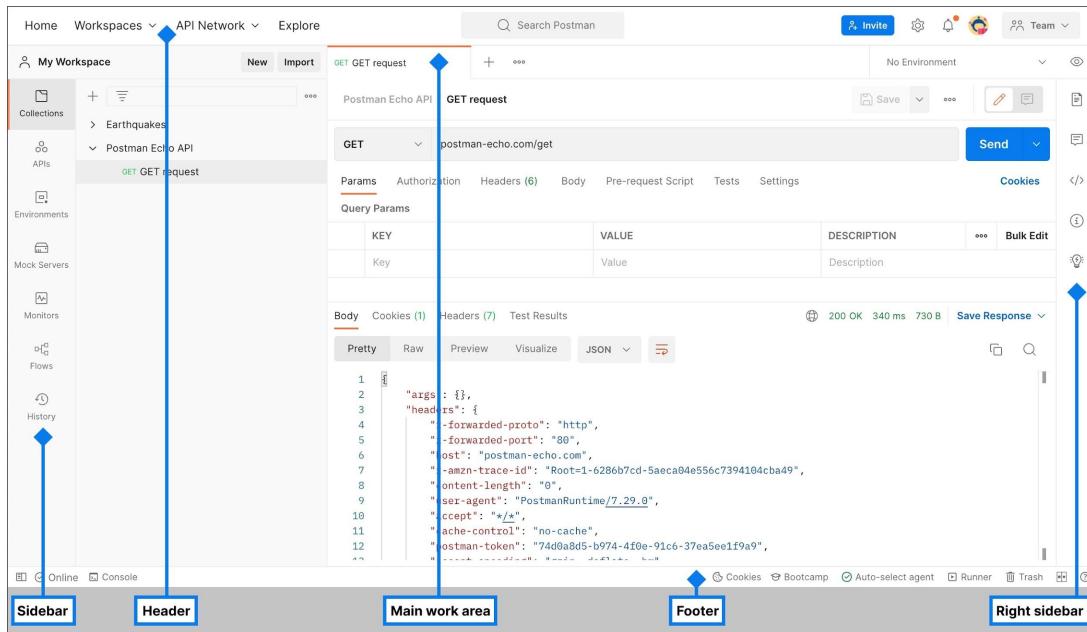
Pilih OS yang sesuai dengan komputer Anda.



The screenshot shows the Postman download page. At the top, there are navigation links for Product, Pricing, Enterprise, Resources and Support, and Explore, along with Sign In and Sign Up for Free buttons. The main heading is 'Download Postman'. Below it, text says 'Download the app to get started using the Postman API Platform today. Or, if you prefer a browser experience, you can try the web version of Postman.' A section titled 'The Postman app' provides a download link for the Windows 64-bit version. To the right, a preview of the Postman web interface is shown, featuring a cookie consent banner at the bottom with options to Manage Cookies, Reject All Cookies, or Accept All Cookies.

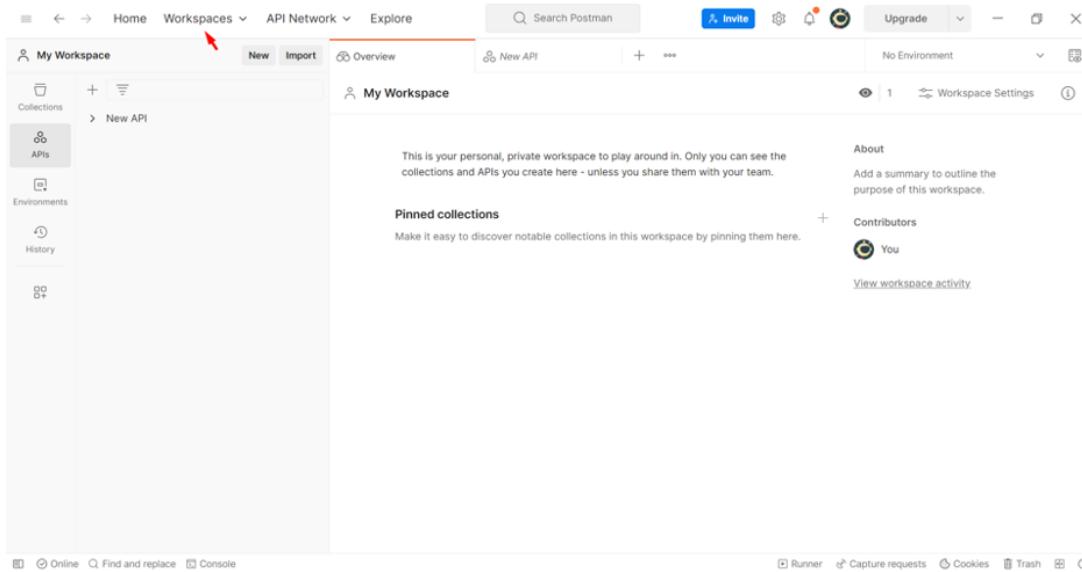
Setelah aplikasi diunduh, lakukan instalasi lalu buat akun atau Login Postman di dalam aplikasi Postman.

Setelah Login akun Postman, kita akan mendapatkan tampilan seperti berikut

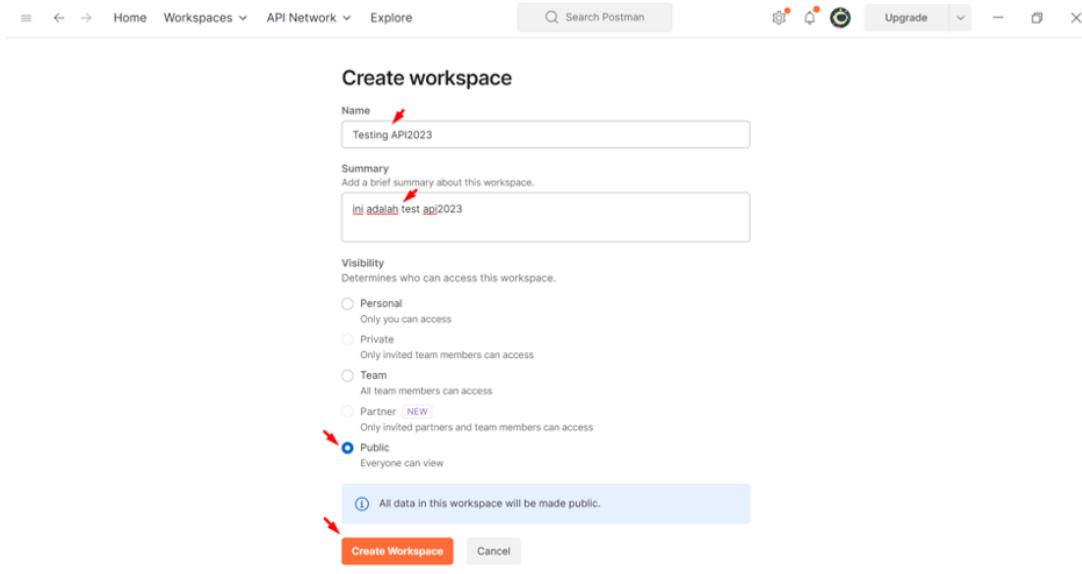


Membuat Workspace

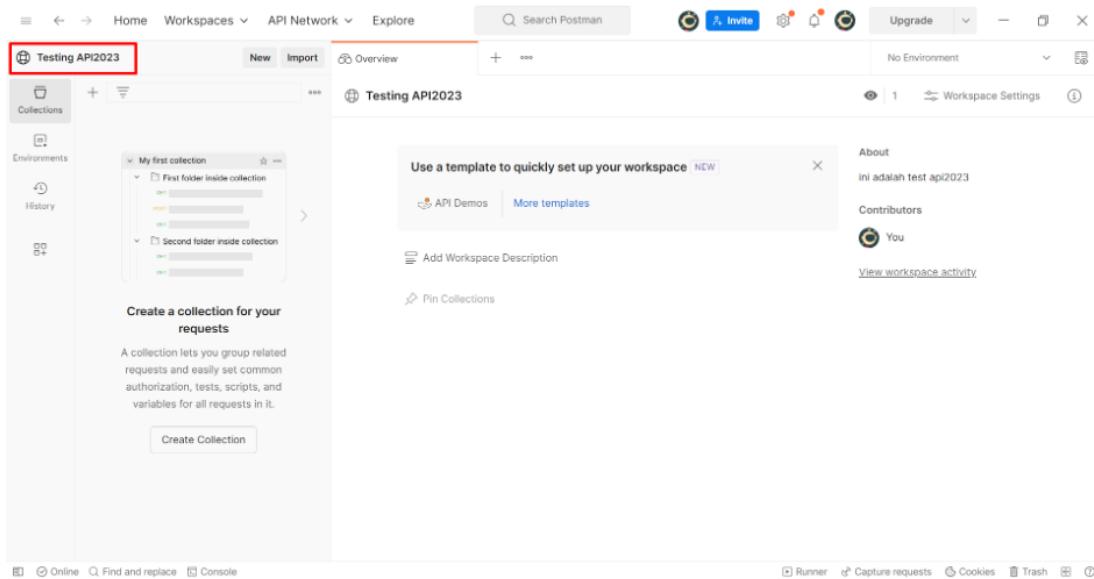
Workspace mengacu pada ruang kerja yang terorganisir untuk mengelompokkan, mengatur, dan mengelola koleksi, lingkungan, skrip, dan sumber daya lainnya terkait proyek atau tim.. Penggunaan workspaces mempermudah kolaborasi dan pengelolaan proyek yang kompleks dengan berbagai orang di tim yang bekerja pada API yang sama atau terkait.



Input field Name, Summary, dan pilih opsi Visibility. Lalu klik tombol Create Workspace.



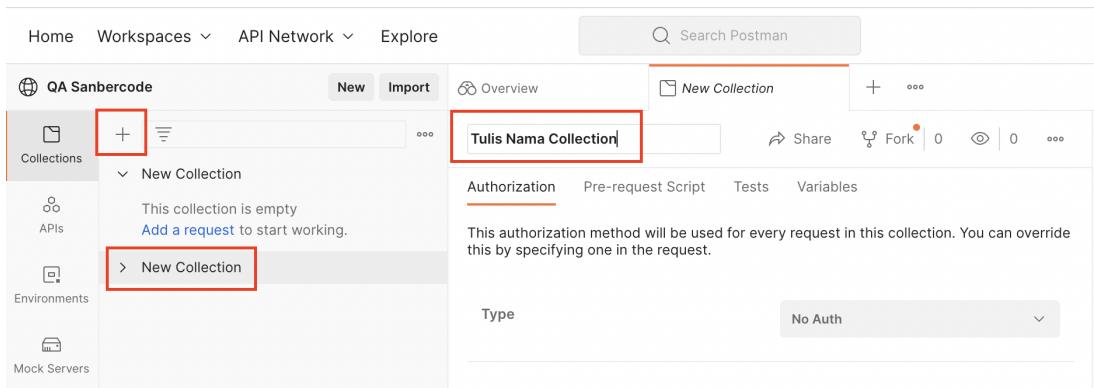
Setelah berhasil membuat Workspace, masuk ke dalam workspace yang baru saja dibuat, maka akan muncul tampilan seperti berikut



Membuat Collection

Pada aplikasi Postman, sebuah collection dapat dibuat untuk menampung berbagai request yang dilakukan pada aplikasi API yang telah dibuat. Selain itu, dengan menggunakan collection sebuah spesifikasi pada sebuah request dapat ditentukan seperti jenis autentikasi yang digunakan dan variabel yang digunakan untuk request tertentu.

Untuk membuat collection baru, pada menu collection, tekan tombol tanda tambah (+) lalu isi nama collection.



Melakukan Request

Sekarang kita akan menambahkan request. Sebagai bahan belajar, ada banyak sekali dummy API yang dapat kita gunakan, seperti:

- <https://reqres.in/>
- <https://dummyjson.com/>
- <https://dummy.restapiexample.com/>
- <https://petstore.swagger.io/>

Untuk membuat request baru, pilih **Add a request**. Kemudian isi nama request, tentukan method, dan input URL. Kemudian tekan tombol Send atau gunakan keyboard Enter untuk mengirim request.

Request menyertakan URL dari API endpoint serta metode permintaan HTTP. Method ini menunjukkan jenis tindakan yang diminta untuk dilakukan oleh API. Ada beberapa method yang sering digunakan untuk melakukan request API

- **GET**

Untuk mengambil dan menampilkan data

- **POST**

Untuk menambahkan data baru, cara kerjanya seperti create.

- **PUT dan PATCH**

Untuk mengedit existing data

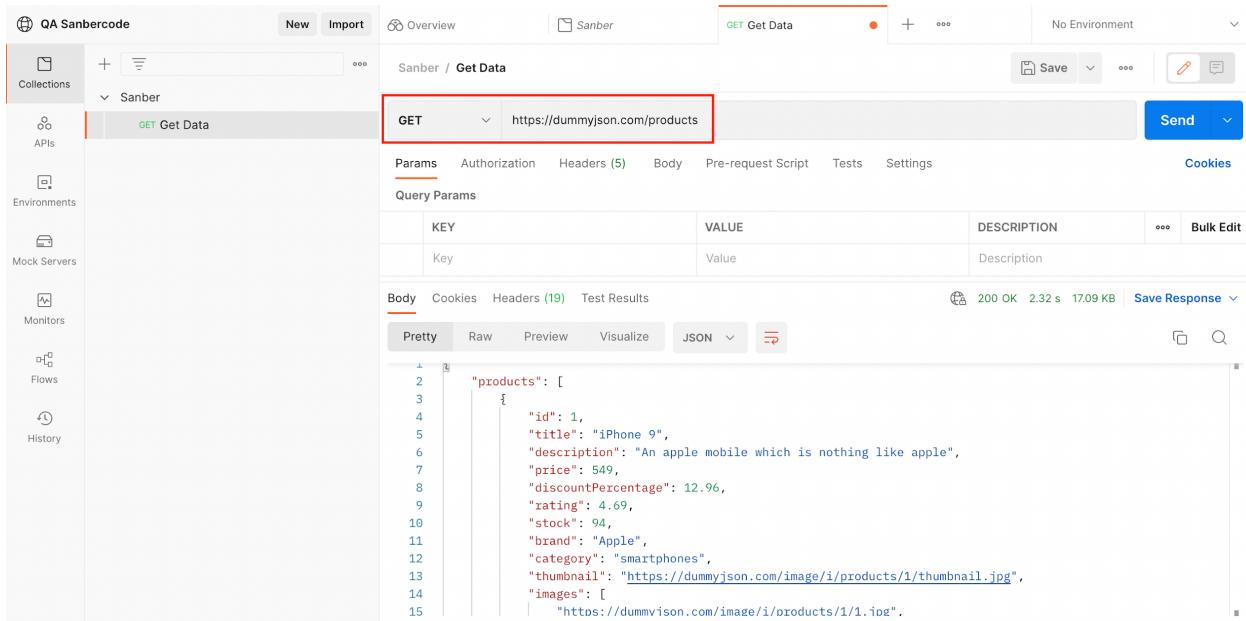
- **DELETE**

Untuk menghapus existing data

Sekarang kita akan mencoba membuat request berdasarkan dokumentasi API dari <https://dummyjson.com/>

Pilih method GET dan inputkan URL <https://dummyjson.com/products>

Jika Request berhasil, maka data product akan muncul di dalam response body. Kita juga dapat melihat status code **200 OK** menandakan request berhasil.



The screenshot shows the Postman interface with a successful API call. The URL is `https://dummyjson.com/products`. The response body is a JSON object:

```

"products": [
  {
    "id": 1,
    "title": "iPhone 9",
    "description": "An apple mobile which is nothing like apple",
    "price": 549,
    "discountPercentage": 12.96,
    "rating": 4.69,
    "stock": 94,
    "brand": "Apple",
    "category": "smartphones",
    "thumbnail": "https://dummyjson.com/image/i/products/1/thumbnail.jpg",
    "images": [
      "https://dummvison.com/image/i/products/1/1.jpg"
    ]
  }
]

```

Kemudian lakukan request POST menggunakan data <https://dummyjson.com/docs/products#add>

Tambahkan Headers seperti pada gambar di bawah ini

Sanber / Add Data

POST https://dummyjson.com/products/add

Params Auth Headers (8) Body Pre-req. Tests Settings

	KEY	VALUE
<input checked="" type="checkbox"/>	Content-Type	application/json
	Key	Value

Tambahkan Body seperti pada gambar di bawah ini

Sanber / Add Data

POST https://dummyjson.com/products/add

Params Auth Headers (8) Body Pre-req. Tests Settings Cookies Beautify

raw JSON

```

1 {
2   "title": "indomie"
3 }

```

Body 200 OK 803 ms 603 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": 101,
3   "title": "indomie"
4 }

```

Jika Request berhasil, maka data product akan muncul di dalam response body. Kita juga dapat melihat status code 200 OK menandakan request berhasil. Pada kasus ini, respon yang seharusnya diberikan adalah 201.

Membuat Variable

Pada aplikasi Postman, sebuah variabel dapat digunakan untuk menyimpan suatu nilai yang nantinya dapat digunakan pada request lainnya. Sebuah variabel dapat dibuat dan nantinya dapat digunakan pada request yang ada di dalam collection tersebut. Ada 5 Variable Scope yakni

1. Global variables

Dapat diakses dari manapun di dalam workspace dan memiliki cakupan terluas di dalam postman.

2. Collection variables

Jangkauan variable ini hanya di dalam collection saja. Tidak berubah berdasarkan environment yang dipilih.

3. Environment variables

Variable ini memungkinkan kita bekerja berdasarkan environment yang berbeda-beda. Kita dapat membuat dan memilih seperti local environment, staging atau production environment.

4. Data variables

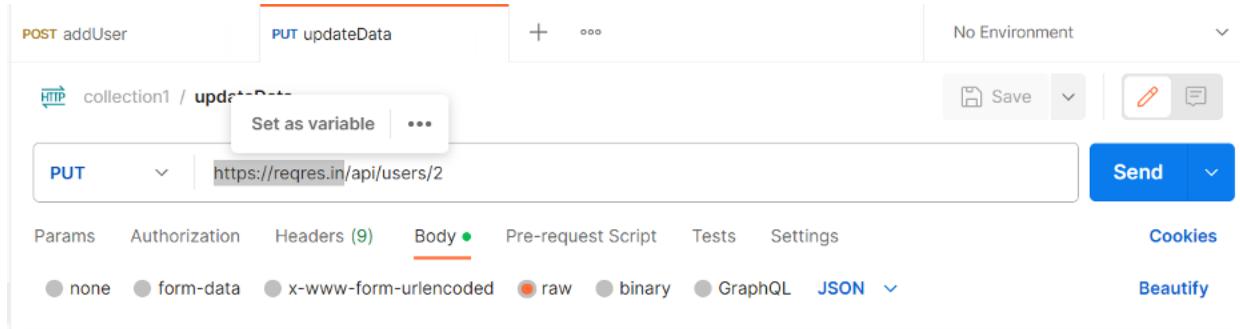
Merupakan variabel external dari dataset ketika menjalankan collection runner. Kita bisa mendapatkannya dari file CSV atau JSON.

5. Local variables

Disebut juga dengan temporary variables karena hanya dapat diakses di dalam request script. Setelah request selesai, variable sudah tidak available.

Membuat Collection Variable

Block data yang ingin dijadikan variabel



POST addUser PUT updateData + No Environment

HTTP collection1 / updateData

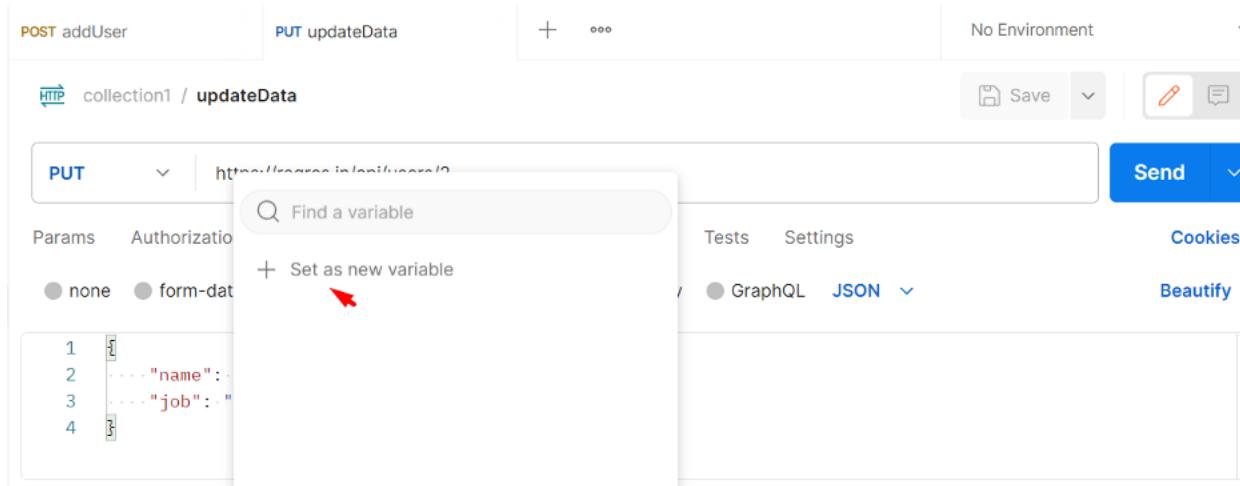
PUT https://reqres.in/api/users/2

Save Send

Params Authorization Headers (9) Body Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

kemudian klik set as new variable



POST addUser PUT updateData + No Environment

HTTP collection1 / updateData

PUT https://reqres.in/api/users/2

Save Send

Params Authorization Tests Settings Cookies Beautify

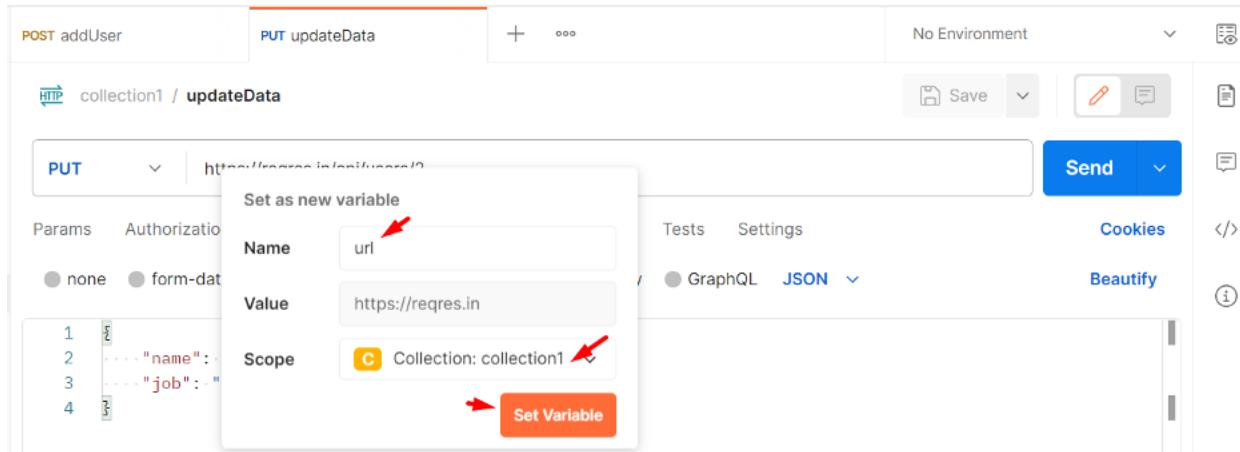
none form-data GraphQL JSON

Find a variable

+ Set as new variable

1 {
2 "name": "
3 "job": "
4 }

kemudian input nama variable dan pilih scope collection, lalu klik tombol Set Variable maka variable akan otomatis terbuat



POST addUser PUT updateData + No Environment

HTTP collection1 / updateData

PUT https://reqres.in/api/users/2

Save Send

Params Authorization Tests Settings Cookies Beautify

none form-data GraphQL JSON

Find a variable

+ Set as new variable

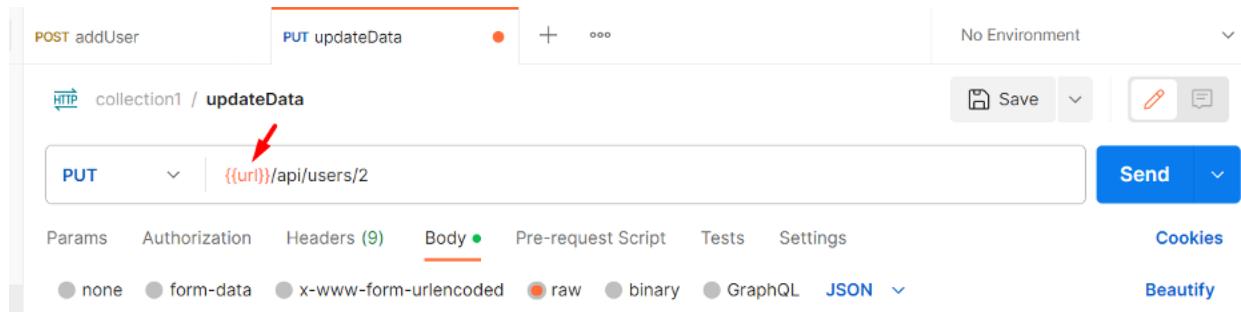
Name url

Value https://reqres.in

Scope Collection: collection1

Set Variable

Penulisan format variable menggunakan tanda double kurung kurawal seperti ini
 {{variable}}



POST addUser PUT updateData + No Environment

HTTP collection1 / updateData

PUT {{url}}/api/users/2

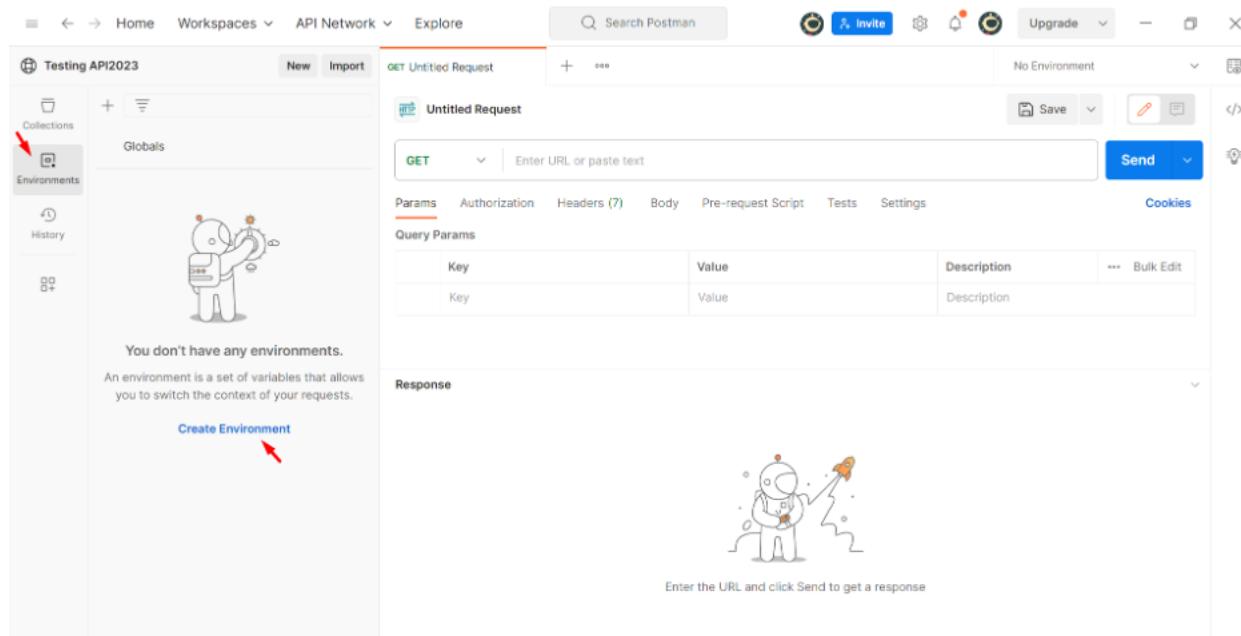
Save Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

Membuat Environment Variable

Klik Menu Environments lalu klik Create Environment



Home Workspaces API Network Explore

Testing API2023 New Import

GET Untitled Request

Untitled Request

GET Enter URL or paste text

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

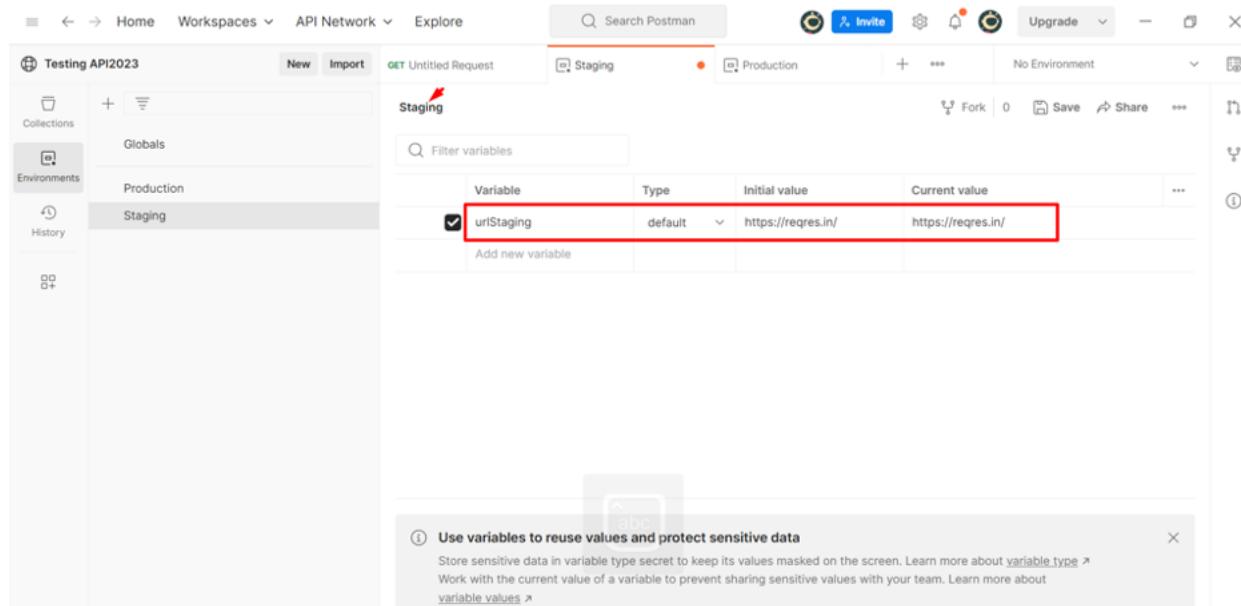
Response

You don't have any environments.

An environment is a set of variables that allows you to switch the context of your requests.

Create Environment

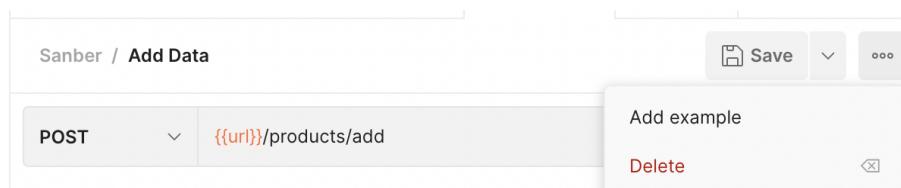
Buat nama environment dan input data variable seperti variable name, type, dan value. Simpan perubahan dengan mengetikkan Ctrl + S.
 Untuk mengaktifkan environment variable, pilih environment pada dropdown yang ada di bagian atas workspace. Default data berupa No Environment.



The screenshot shows the Postman interface with the 'Testing API2023' collection selected. In the 'Environments' tab, the 'Staging' environment is chosen. A table lists environment variables, with one named 'urlStaging' highlighted by a red box. The 'Initial value' and 'Current value' for this variable are both set to 'https://reqres.in/'. A tooltip at the bottom left of the interface reads: 'Use variables to reuse values and protect sensitive data'.

Simpan Testing Data

Ketika mengganti request body atau parameter kemudian mengirimkan ulang request tersebut, maka response body yang sudah didapatkan sebelumnya akan terhapus. Maka sebelum itu disarankan untuk dapat menyimpan response body yang sudah didapatkan dengan melakukan **Add Example**.



The screenshot shows the 'Sanber / Add Data' collection in Postman. A 'POST' request is selected with the URL {{url}}/products/add. A modal dialog box titled 'Add example' is open, containing 'Add example' and 'Delete' buttons.

Chaining Request

Dalam melakukan request ada 5 method yang sering digunakan berikut penjelasannya

- GET : Digunakan untuk meminta resource ke sebuah server.
- POST : Digunakan untuk menambahkan data ke dalam server.

- PUT : Digunakan untuk melakukan update resource ke sebuah server, dengan PUT method bisa mengirimkan secara keseluruhan data atau replace
- PATCH : Digunakan untuk memodifikasi data secara parsial atau sebagian saja jadi bukan secara keseluruhan seperti put.
- DELETE : Digunakan untuk melakukan menghapus resource ke sebuah server.

membuat request : Isi nama request, pilih method yang akan digunakan (GET, POST, PUT, DELETE dll), isi link URL, lalu klik Send untuk mengirimkan request ke server. Request menyertakan URL dari API endpoint serta metode permintaan HTTP.

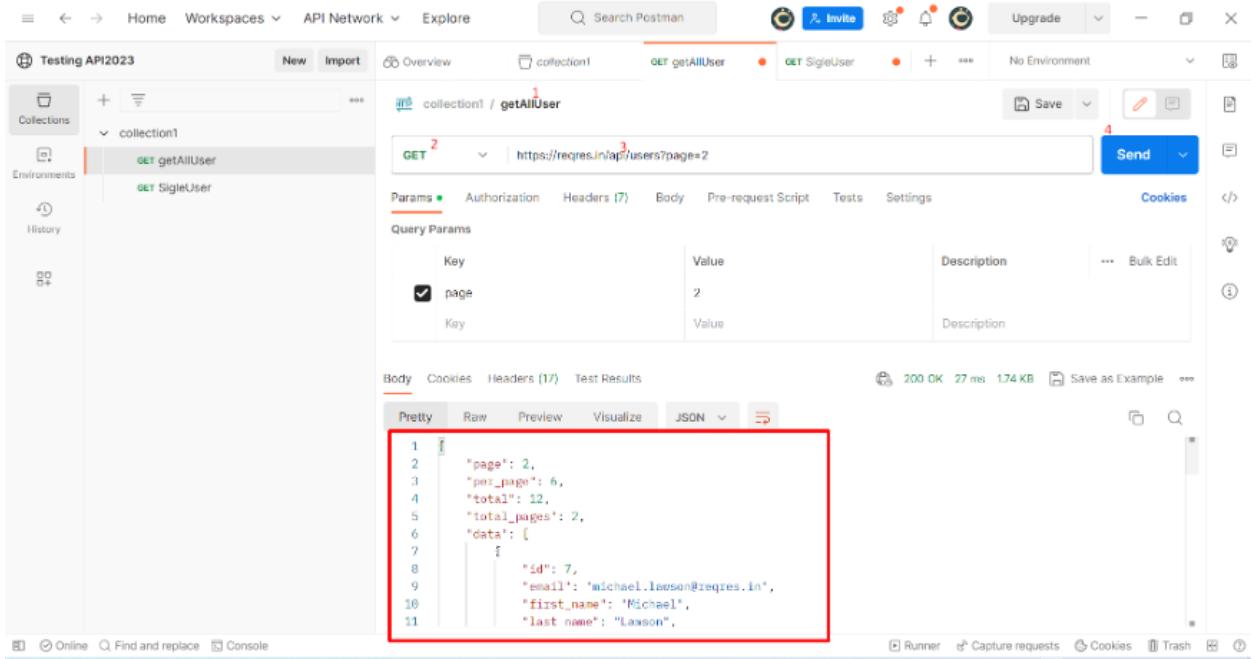
jika sudah mengirimkan request maka server akan memberi respon dengan status code yang berbeda-beda, respon dari server biasanya berbentuk JSON. JavaScript Object Notation adalah format file berbasis teks yang umumnya digunakan dalam proses pertukaran data antara server dan klien.

Melakukan Pengujian API

Contoh pengujian API pada <https://reqres.in/>

1. Get All User

Kotak merah menunjukkan respon body untuk data yang diminta.



The screenshot shows the Postman interface with the 'Testing API2023' collection selected. Under 'collection1', the 'GET getAllUser' request is highlighted. The 'Params' tab shows a 'page' parameter set to 2. The 'Body' tab displays the JSON response:

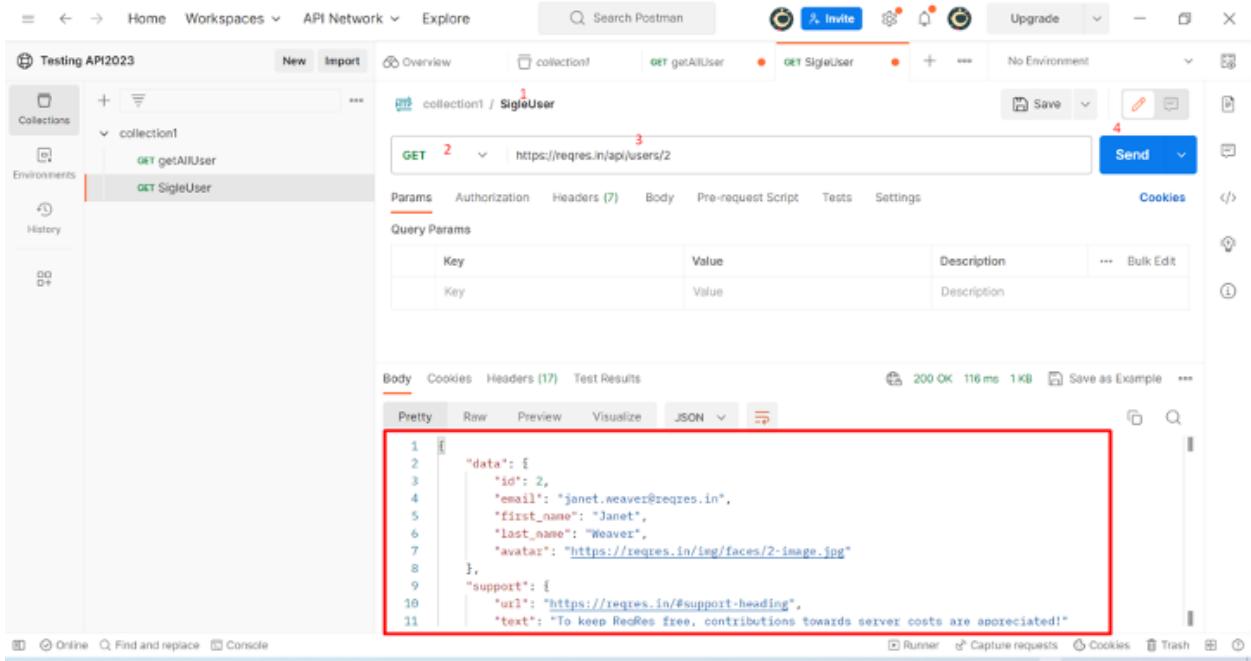
```

1
2   "page": 2,
3   "per_page": 6,
4   "total": 12,
5   "total_pages": 2,
6   "data": [
7     {
8       "id": 7,
9       "email": "michael.lawson@reqres.in",
10      "first_name": "Michael",
11      "last_name": "Lanson",

```

2. Get Single User

Hanya menampilkan 1 data user sesuai ID yang diinputkan



The screenshot shows the Postman interface with the 'Testing API2023' collection selected. Under 'collection1', the 'GET SingleUser' request is highlighted. The 'Params' tab shows an empty 'Key' column. The 'Body' tab displays the JSON response:

```

1
2   "data": {
3     "id": 2,
4     "email": "janet.weaver@reqres.in",
5     "first_name": "Janet",
6     "last_name": "Weaver",
7     "avatar": "https://reqres.in/img/faces/2-image.jpg"
8   },
9   "support": {
10     "url": "https://reqres.in/#support-heading",
11     "text": "To keep ReqRes free, contributions towards server costs are appreciated!"

```

3. Create User

Pada request dengan metode post biasanya harus menyertakan request body, yakni data yang akan dikirimkan.

The screenshot shows the Postman application interface. The left sidebar displays collections, environments, and history. The main workspace shows a collection named "collection1" containing three requests: "getAllUser", "SingleUser", and "addUser". The "addUser" request is selected and configured as a POST method to the URL <https://reqres.in/api/users>. The "Body" tab is active, showing a JSON payload with fields "name" and "job". The "Params", "Authorization", and "Headers" tabs are also visible. The top navigation bar includes links for Home, Workspaces, API Network, Explore, and a search bar. The top right features upgrade options and user account settings.

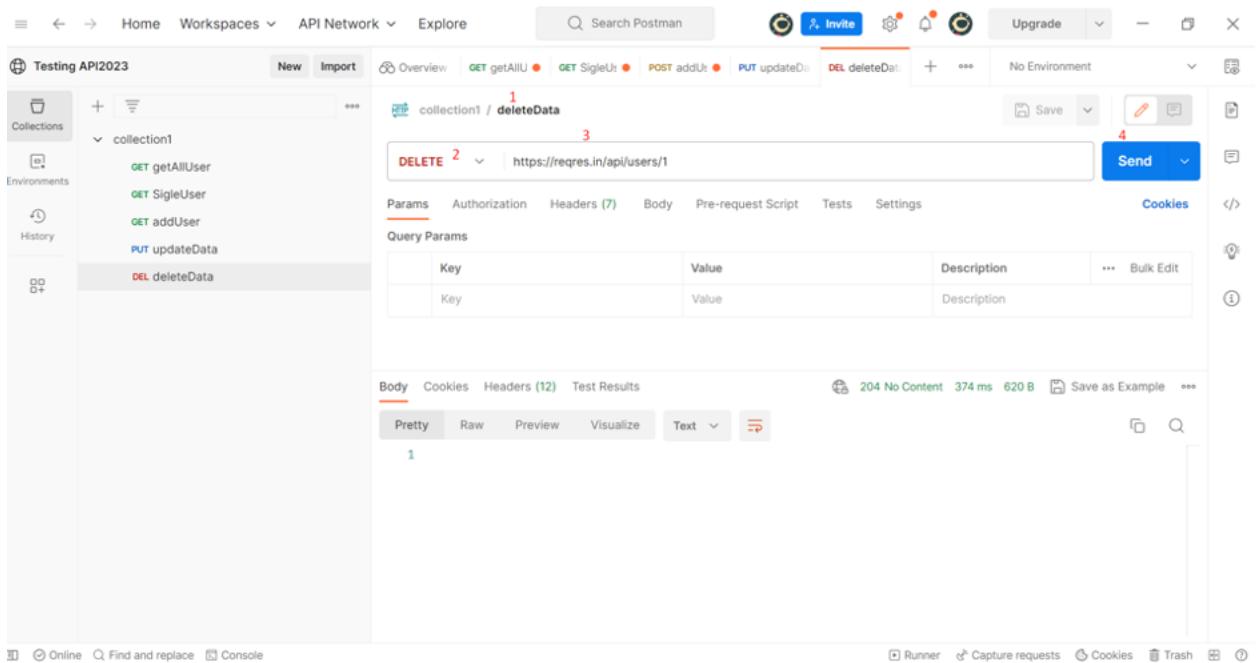
4. Update User

Metode Put dan Patch dapat mengedit existing data (data yang sudah ada).

The screenshot shows the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', 'API Network', 'Explore', a search bar, and various account and settings icons. Below the header, the left sidebar displays 'Testing API2023' with sections for 'Collections' (including 'collection1' with 'getAllUser', 'SigleUser', 'addUser', and 'updateData'), 'Environments', and 'History'. The main workspace shows a collection named 'collection1 / updateData'. A 'PUT' request is selected with the URL 'https://reqres.in/api/users/2'. The 'Body' tab is active, showing a JSON payload with fields 'name' and 'job'. The response section shows a 200 OK status with a response time of 386 ms and a size of 787 B. The bottom navigation bar includes links for 'Online', 'Find and replace', 'Console', 'Runner', 'Capture requests', 'Cookies', 'Trash', and other settings.

5. Delete User

Metode delete digunakan untuk menghapus data berdasarkan ID yang diinputkan.



Writing Test

Writing Test bertujuan untuk memastikan atau memvalidasi bahwa repon API sudah sesuai dengan apa yang diharapkan, bahwa integrasi antar services sudah berfungsi dengan baik. Kita dapat menulis Test Scripts untuk API request di Postman menggunakan JavaScript. Misalnya, Kita dapat menulis Test untuk memvalidasi error handling API dengan mengirimkan request dengan data yang tidak lengkap atau parameter yang salah.

Untuk membuat Test, Postman sudah menyediakan Snippets yang bisa digunakan untuk mempermudah penulisan test, kemudian kita juga dapat modifikasi agar sesuai dengan logika pengujian yang akan kita lakukan, ada banyak Snippets yang dapat kita gunakan Snippets “Status Code : Code is 200”, “Response body : Contains string” dll.

Cara membuat Test

Klik tab Tests, pilih Snippets yang akan digunakan, pengujian ini akan berjalan setelah permintaan berjalan

The screenshot shows the Postman interface with a red arrow pointing to the 'Tests' tab in the top navigation bar. The 'Tests' tab is highlighted in orange. Below it, the 'Body' tab is also highlighted in orange. The main area displays a JavaScript test script:

```
1 pm.test("Status code is 201", function () {  
2     pm.response.to.have.status(201);  
3 });  
4 pm.test("Cek respon body", function () {  
5     pm.expect(pm.response.text()).to.include("Ari");  
6 });
```

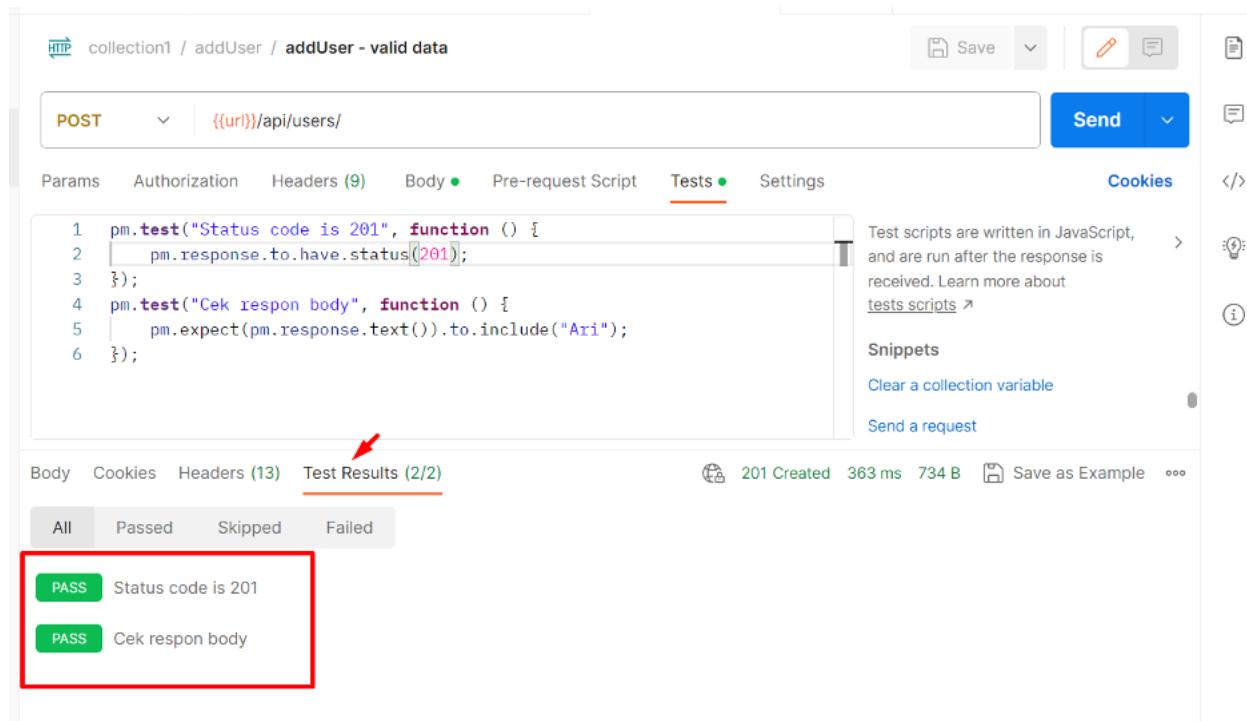
To the right of the script, there is a sidebar with the following content:

Test scripts are written in JavaScript, and are run after the response is received. Learn more about [tests scripts](#).

Snippets

- [Clear a collection variable](#)
- [Send a request](#)
- [Status code: Code is 200](#)
- [Response body: Contains string](#)
- [Response body: JSON value check](#)
- [Response body: Is equal to a string](#)
- [Response headers: Content-Type header check](#)
- [Response time is less than 200ms](#)
- [Status code: Successful POST request](#)

untuk melihat hasil pengecekan ada di tab Test Results



The screenshot shows the Postman interface with a collection named "collection1 / addUser / addUser - valid data". The "Tests" tab is selected, displaying the following JavaScript code:

```

1 pm.test("Status code is 201", function () {
2     pm.response.to.have.status(201);
3 });
4 pm.test("Cek respon body", function () {
5     pm.expect(pm.response.text()).to.include("Ari");
6 });

```

Below the code, the "Test Results (2/2)" tab is highlighted with a red arrow pointing to it. The results show two successful tests:

- PASS Status code is 201
- PASS Cek respon body

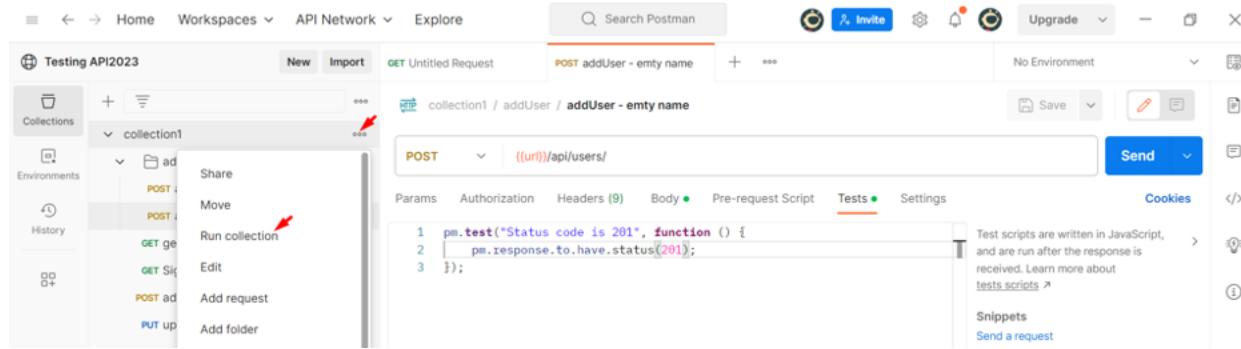
At the bottom right, the status is shown as 201 Created, 363 ms, 734 B.

Collection Runner & Monitor

Collection Runner adalah sekumpulan request yang dapat bekerja secara bersamaan dan sangat berguna dalam melakukan automasi testing, collection runner akan mencatat hasil pengujian dari request yang sudah kita kirim. Kita juga dapat mengkonfigurasi pengujian sesuai dengan yang dibutuhkan, dan juga bisa menjadwalkan pengujian dijalankan.

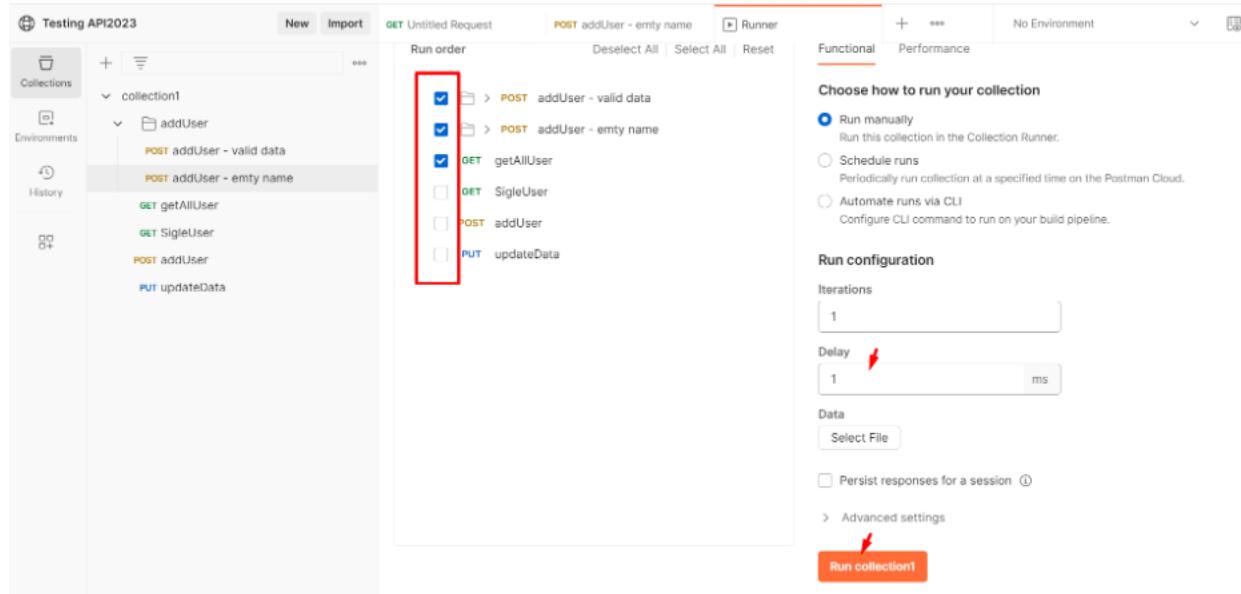
Cara menjalankan Collection Runner di Postman :

Klik icon titik tiga, klik Run Collection



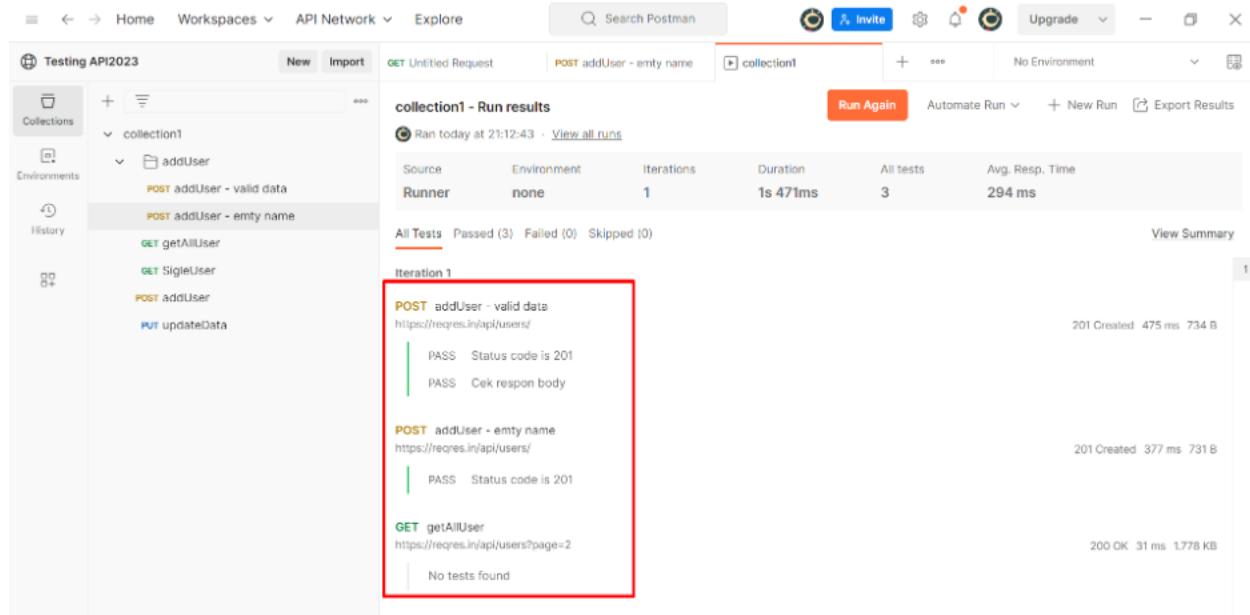
The screenshot shows the Postman interface with a collection named "collection1" selected. In the context menu for a specific request, the "Run collection" option is highlighted with a red arrow.

Pilih request yang ingin dijalankan, input waktu yang diinginkan, klik Run collection



The screenshot shows the Postman Runner interface. In the "Run order" section, several requests from the "collection1" are listed and checked. A red box highlights the checked requests. At the bottom right, a red arrow points to the "Run collection1" button.

Runner akan menjalankan request yang sudah dipilih dan akan menampilkan result seperti dibawah ini.

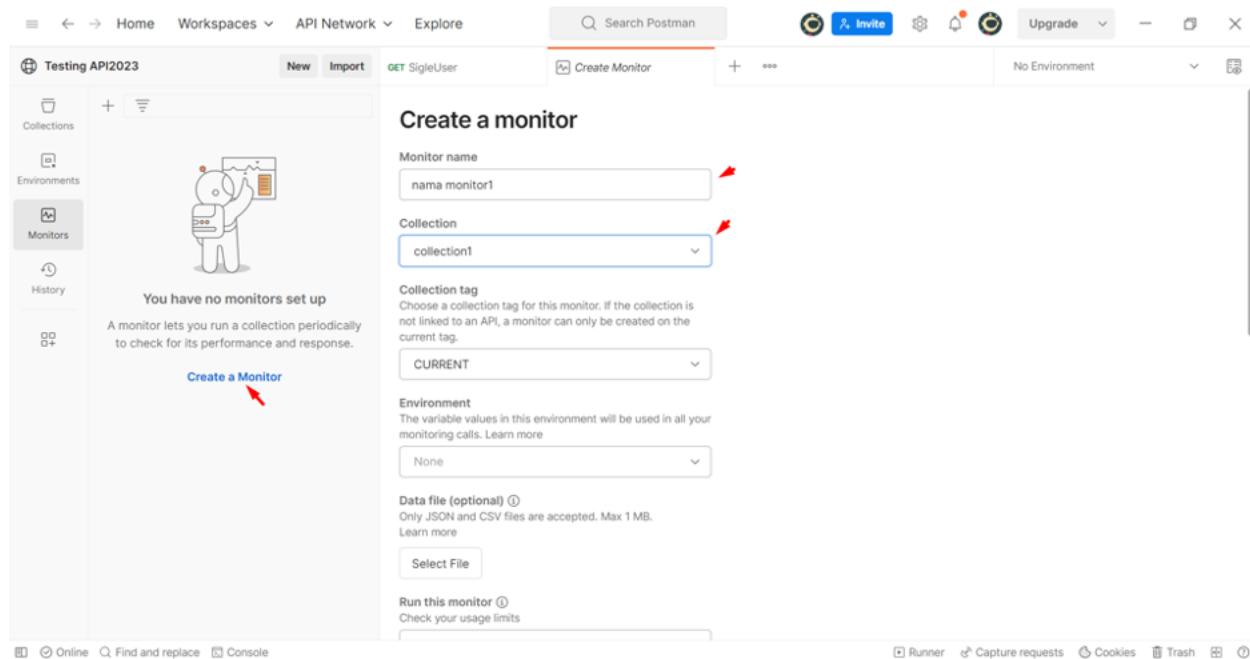


The screenshot shows the Postman interface with a collection named "Testing API2023". The "collection1" section is expanded, showing several API endpoints: "POST addUser - valid data", "POST addUser - empty name", "GET getAllUser", "GET SingleUser", "POST addUser", and "PUT updateData". The "POST addUser - valid data" step is highlighted with a red box. The response details show a green "PASS" status with the message "Status code is 201". Below it, another test step for "POST addUser - empty name" also shows a green "PASS" status with the message "Status code is 201". The overall summary indicates 3 tests passed, 0 failed, and 0 skipped.

Postman Monitor

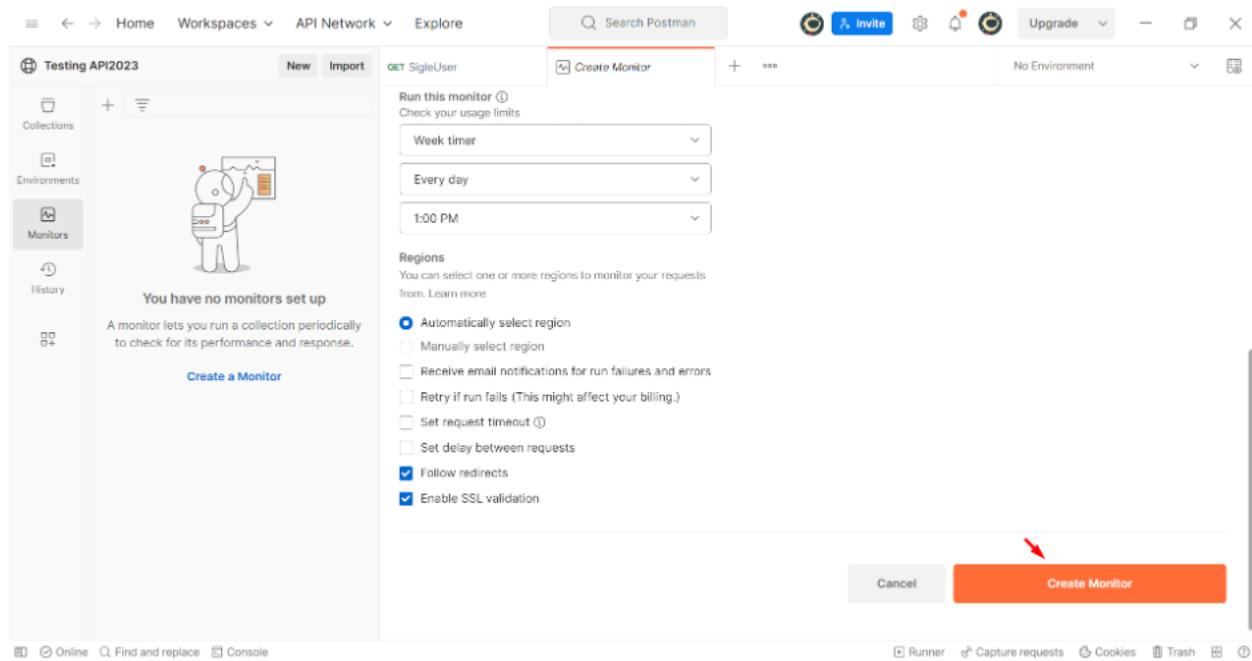
Postman monitoring memungkinkan untuk melakukan monitoring berdasarkan collection dan environment. Postman monitoring dapat dilakukan secara tim maupun personal, tergantung kebutuhan. Berikut langkah-langkah dari monitoring API menggunakan postman:

Klik Create Minitor, isi nama Monitor, Pilih Collection



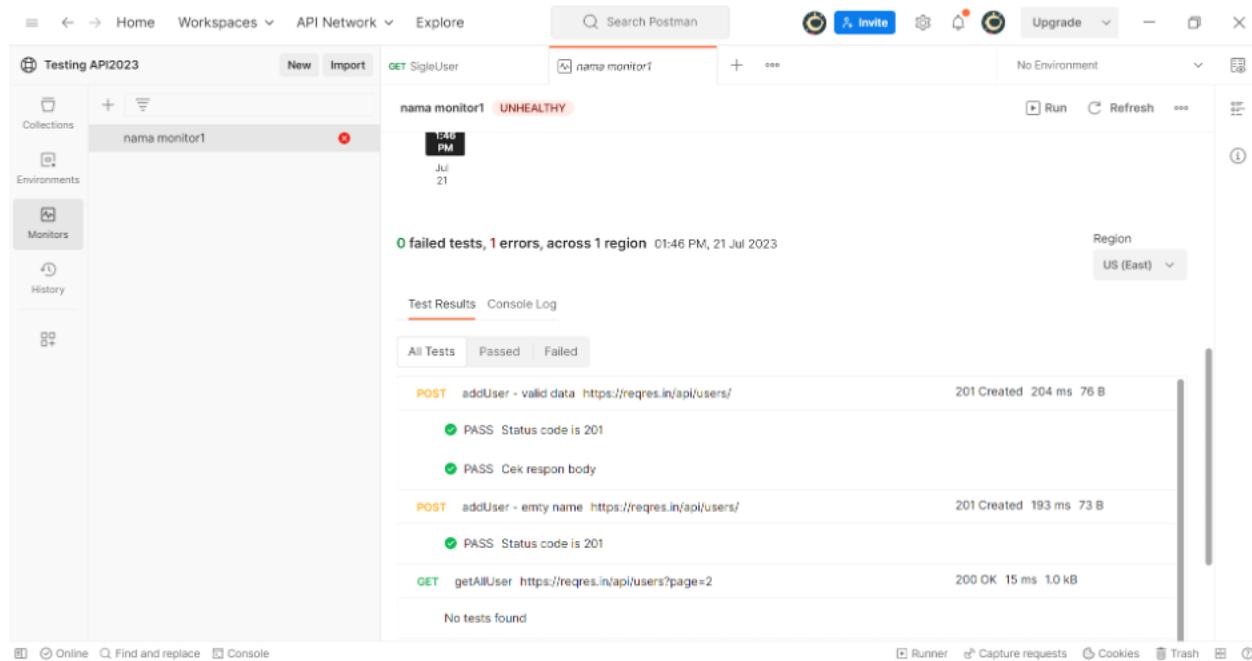
The screenshot shows the Postman interface with a collection named "Testing API2023". On the left sidebar, the "Monitors" icon is selected. A central dialog box titled "Create a monitor" is open. It contains fields for "Monitor name" (set to "nama monitor1") and "Collection" (set to "collection1"). There are also sections for "Collection tag" (set to "CURRENT"), "Environment" (set to "None"), "Data file (optional)" (with a note about accepted file types), and "Run this monitor" (with a note about usage limits). At the bottom left of the dialog, there is a blue "Create a Monitor" button, which is also highlighted with a red arrow.

Pilih konfigurasi mana aja yang diinginkan, klik Create Monitor



The screenshot shows the Postman interface for creating a monitor. In the top right, there's a search bar and several icons. Below it, the main area has tabs for 'Testing API2023' (selected), 'New', 'Import', and 'Create Monitor'. A sub-menu for 'Create Monitor' is open, showing options like 'Run this monitor' (with 'Check your usage limits'), 'Week timer' (set to 'Every day' at '1:00 PM'), and 'Regions' (with 'Automatically select region' checked). At the bottom right of this menu is a red arrow pointing to the 'Create Monitor' button.

Result monitor



The screenshot shows the Postman interface after a monitor has been created. The 'Testing API2023' collection is selected. A monitor named 'nama monitor1' is listed under the 'Monitors' tab, marked as 'UNHEALTHY'. The status bar shows '0 failed tests, 1 errors, across 1 region' at '01:46 PM, 21 Jul 2023'. The 'Region' dropdown is set to 'US (East)'. The 'Test Results' section shows a table of test logs:

Action	Request	Status	Time
POST	addUser - valid data https://reqres.in/api/users/	201 Created	204 ms 76 B
PASS	Status code is 201		
PASS	Cek respon body		
POST	addUser - empty name https://reqres.in/api/users/	201 Created	193 ms 73 B
PASS	Status code is 201		
GET	getAllUser https://reqres.in/api/users?page=2	200 OK	15 ms 1.0 kB
No tests found			