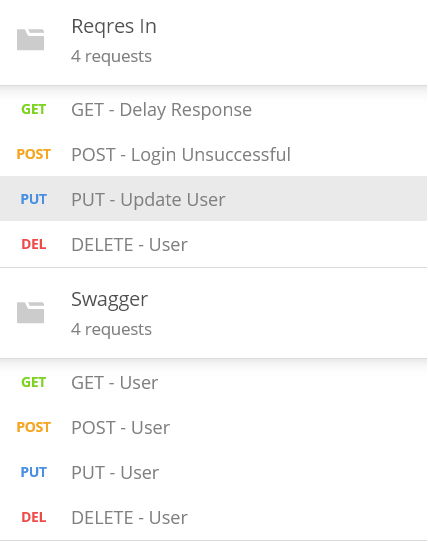
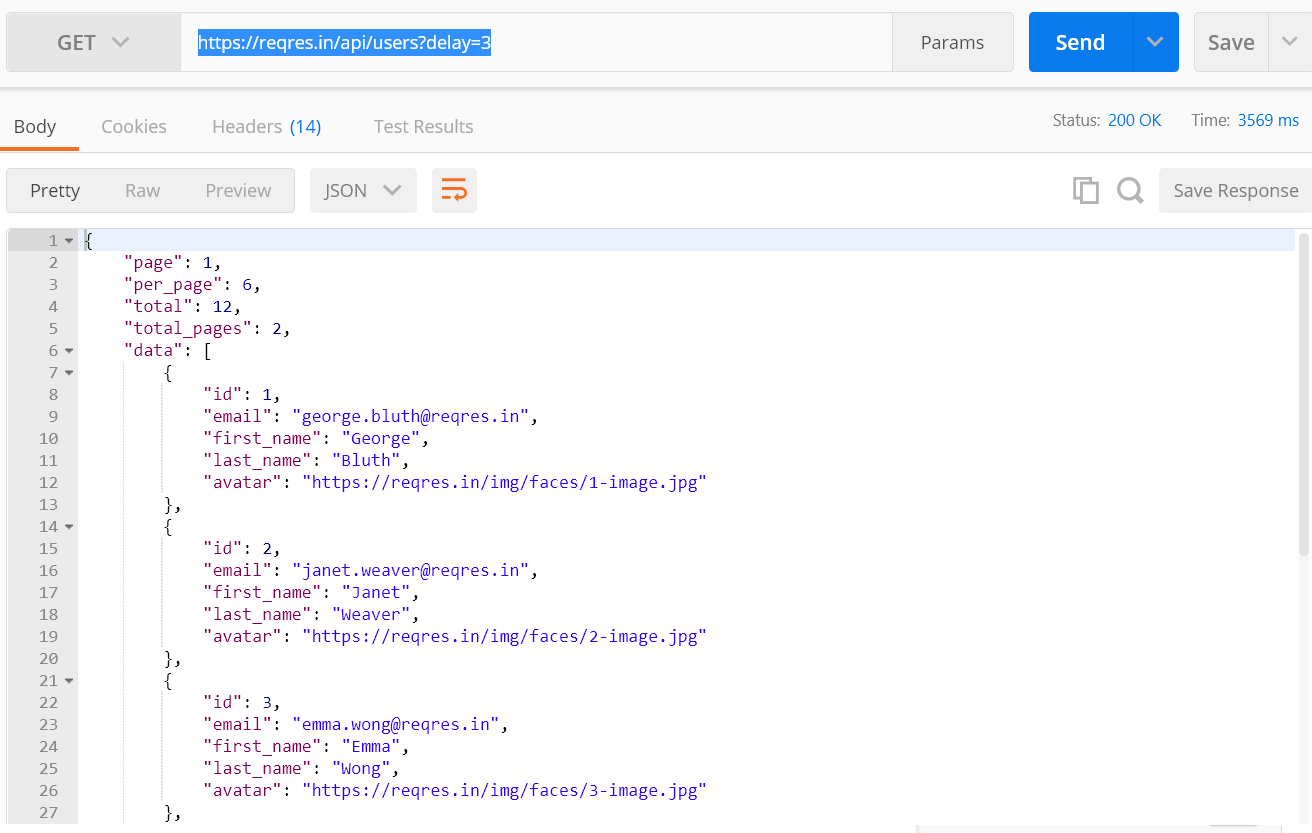
1. Find min 2 open API and create postman collection from it



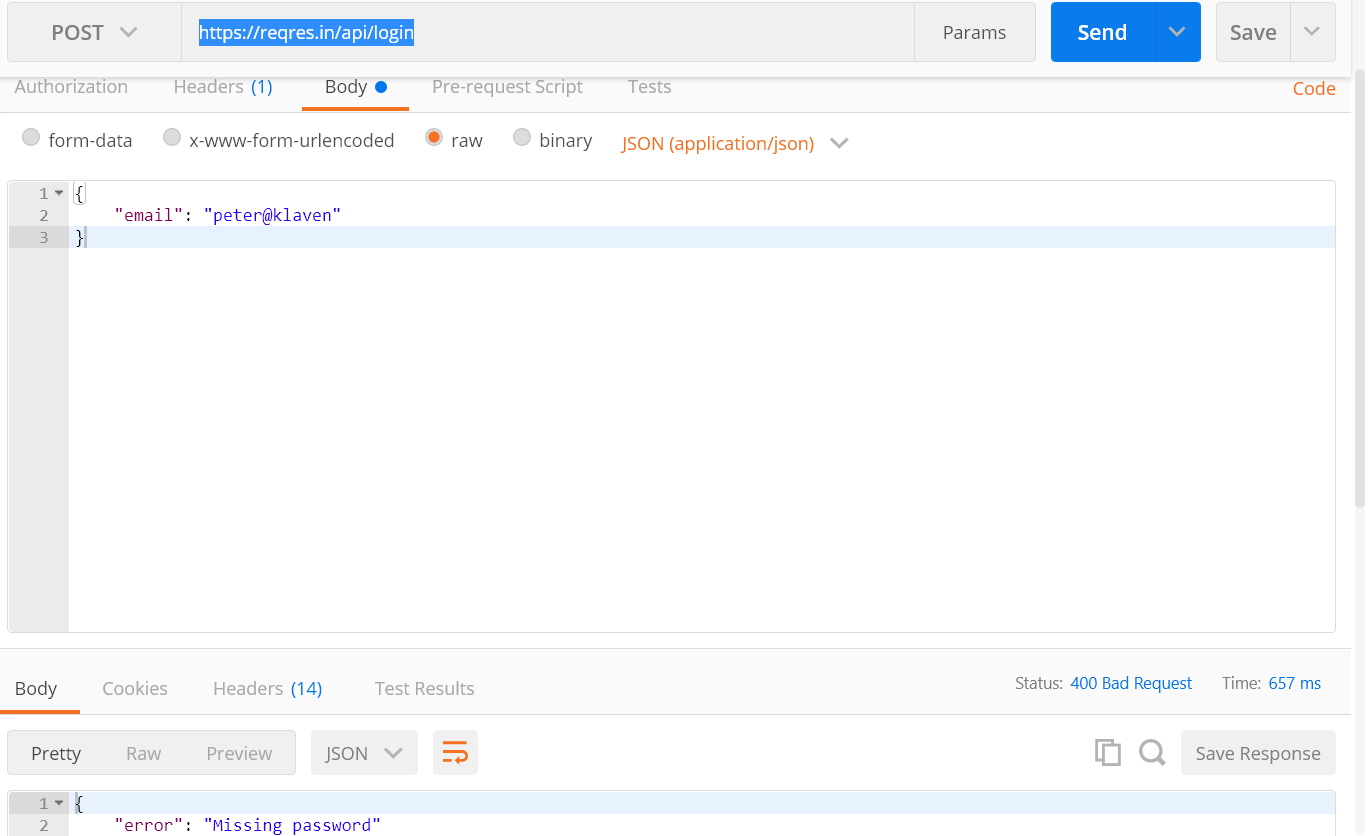
* **REGRES.IN**
* **GET**

- URL : <https://reqres.in/api/users?delay=3>

Method ini digunakan untuk memperoleh data user dengan status delay response. Pada postman berstatus 200 OK yang artinya berhasil diperoleh.

* **POST**

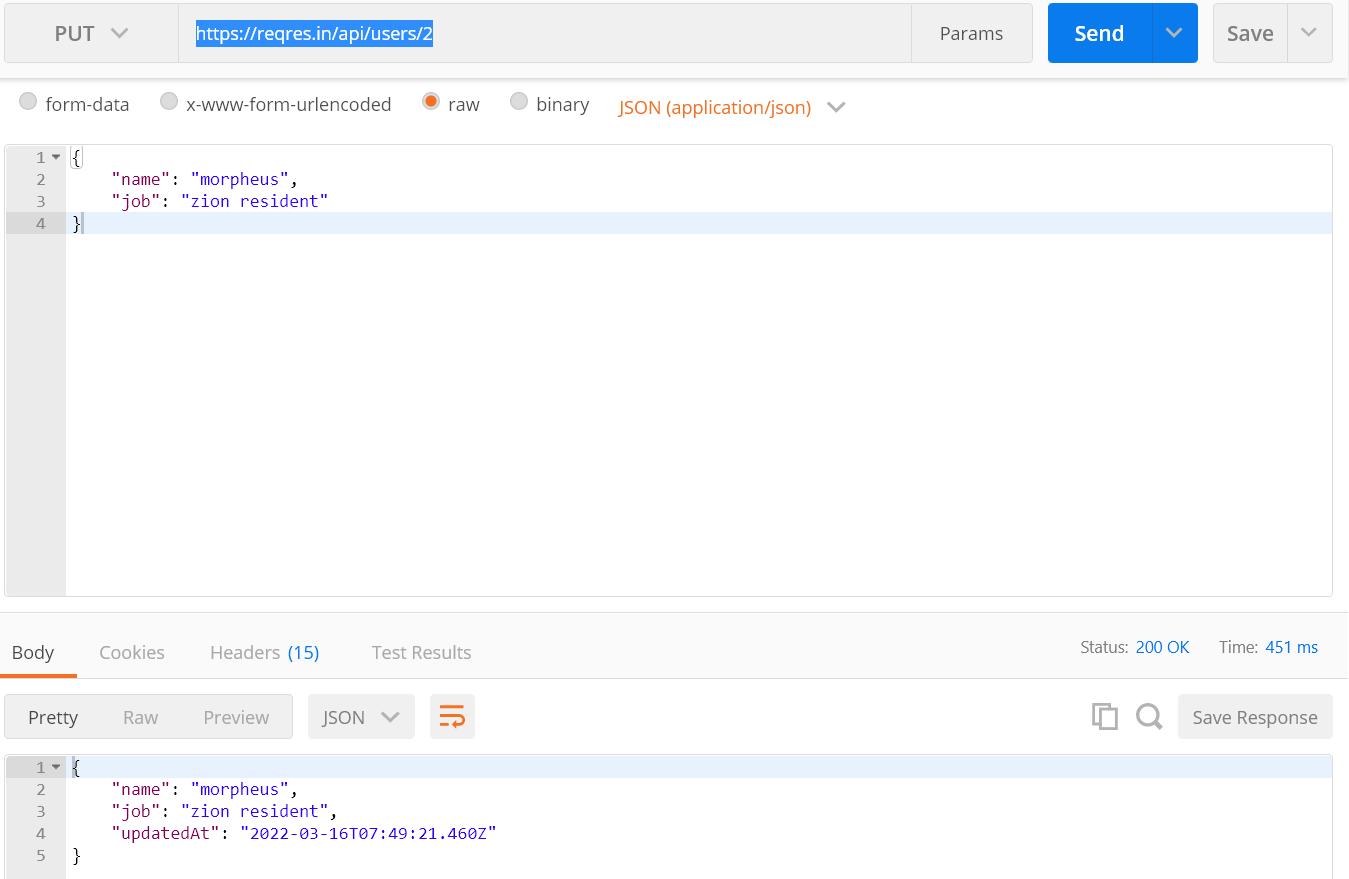
**-** URL : <https://reqres.in/api/login>



Method ini digunakan menambah data untuk login ke website, dimana pada body kita memberikan value email : peter@klaven, tetapi hasil yang kita peroleh berstatus 400 yang artinya error yang terjadi saat server tidak memahami permintaan dari pengunjung website. Ketidakpahaman ini disebabkan karena permintaan tersebut tidak tepat atau corrupt di tengah proses pengiriman data. Error-nya terjadi dikarenakan kita tidak memberikan value untuk password.

* **PUT**

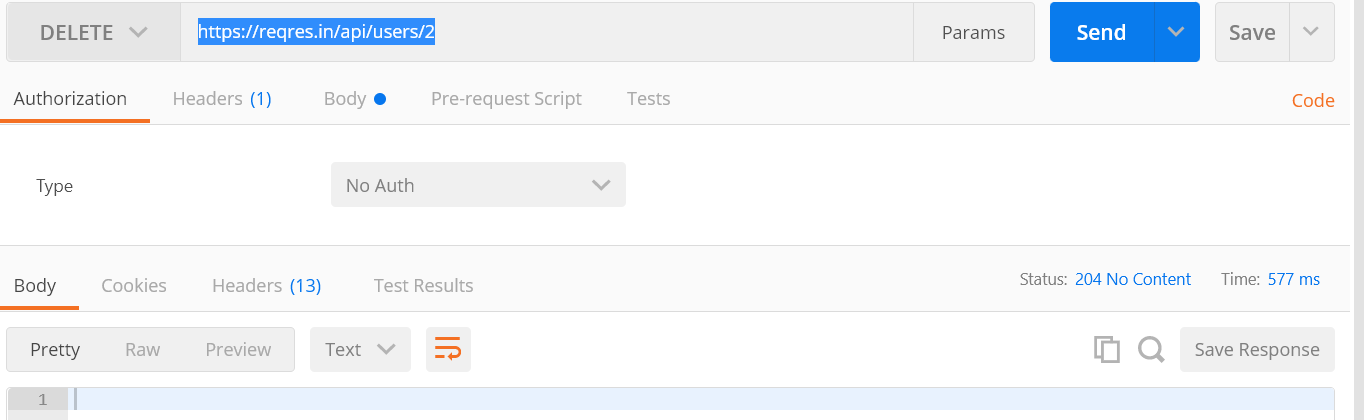
**-** URL : <https://reqres.in/api/users/2>

****

Method ini digunakan untuk mengubah data atau update data, data yang akan diubah pada data user ini adalah dengan id 2. Value yang diberikan adalah "name": "morpheus", "job": "zion resident". Pada postman berstatus 200 OK yang artinya data berhasil di-update.

* **DELETE**

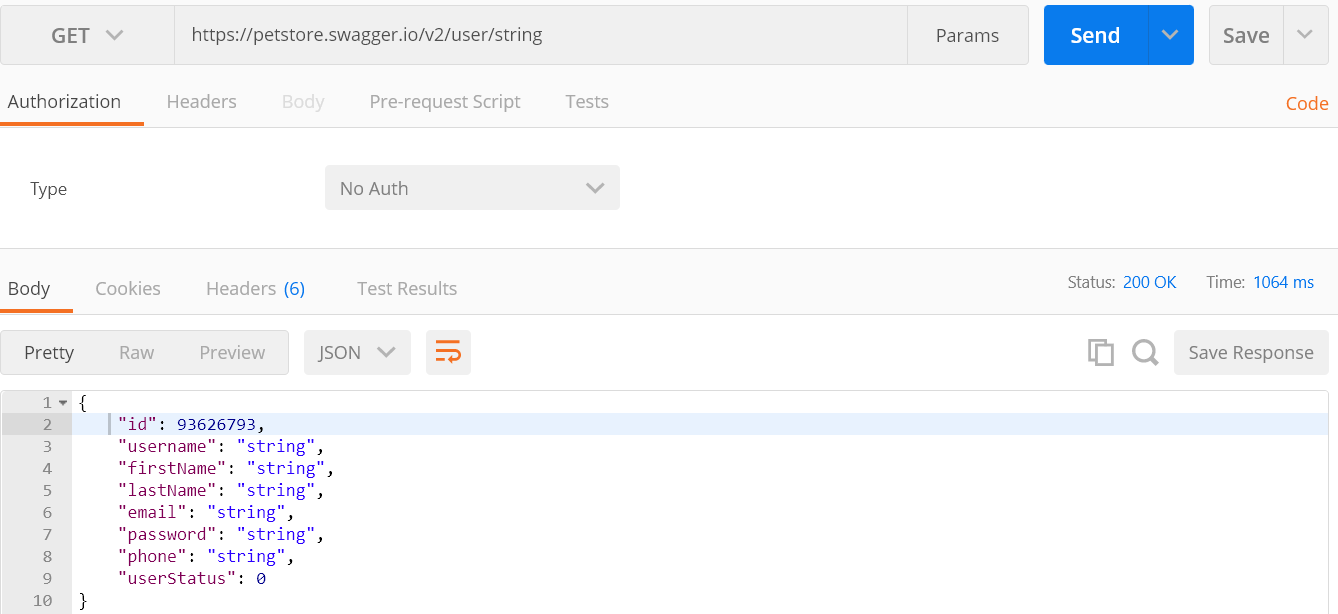
**-** URL : <https://reqres.in/api/users/2>

****

Method ini digunakan untuk menghapus data, pada gambar di atas data yang dihapus dengan id 2. Pada postman berstatus 204, yang artinya id 2 telah berhasil dihapus maka menghasilkan message no content.

* **SWAGGER**
* **GET**

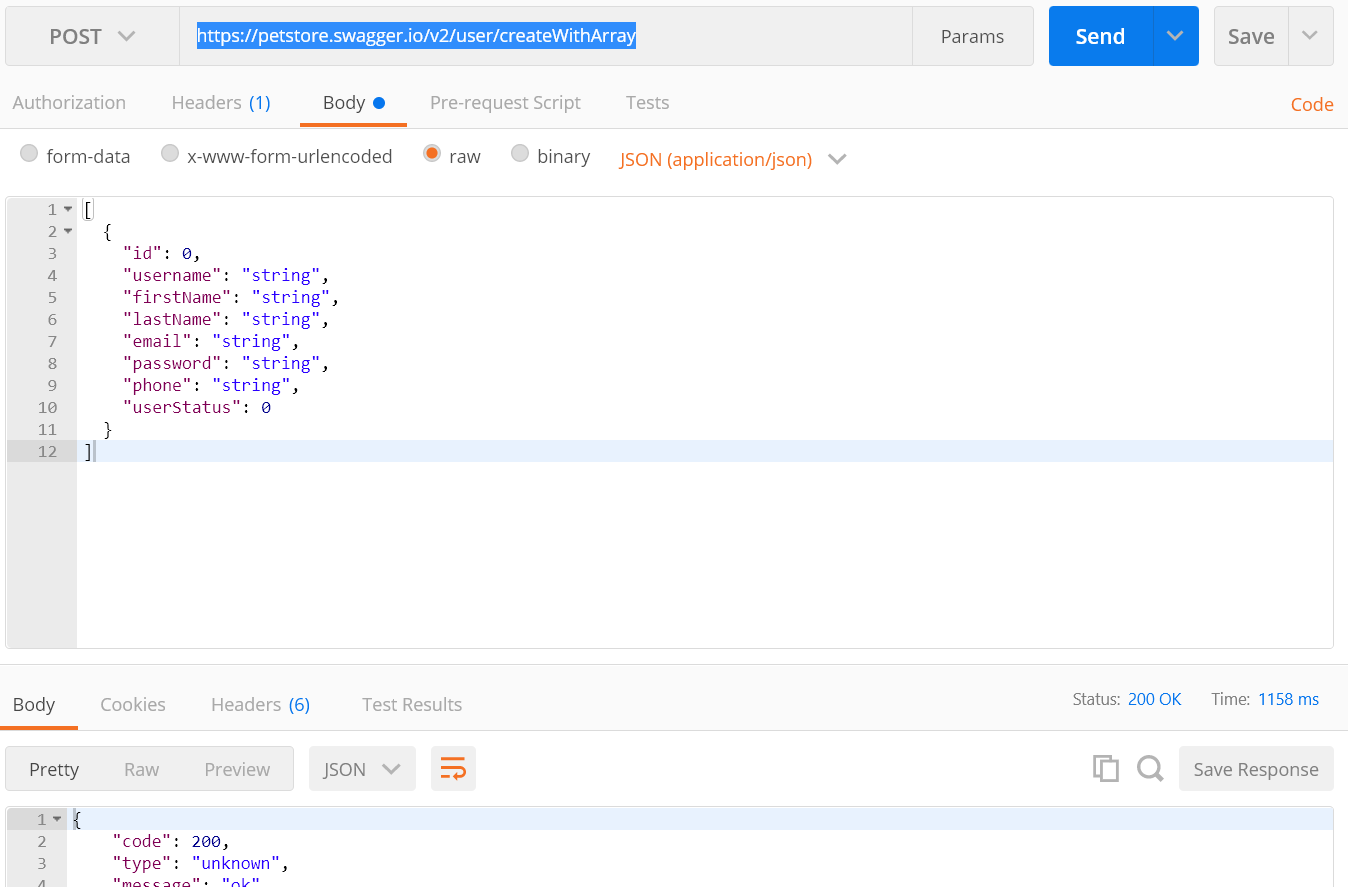
**-** URL : <https://petstore.swagger.io/v2/user/string>

****

Method ini digunakan untuk memperoleh data username string. Pada postman berstatus 200 OK yang artinya berhasil diperoleh.

* **POST**

**-** URL : <https://petstore.swagger.io/v2/user/createWithArray>

****

Method ini digunakan untuk menambahkan list dari user, kita menambahkan value :

[

{

"id": 0,

"username": "string",

"firstName": "string",

"lastName": "string",

"email": "string",

"password": "string",

"phone": "string",

"userStatus": 0

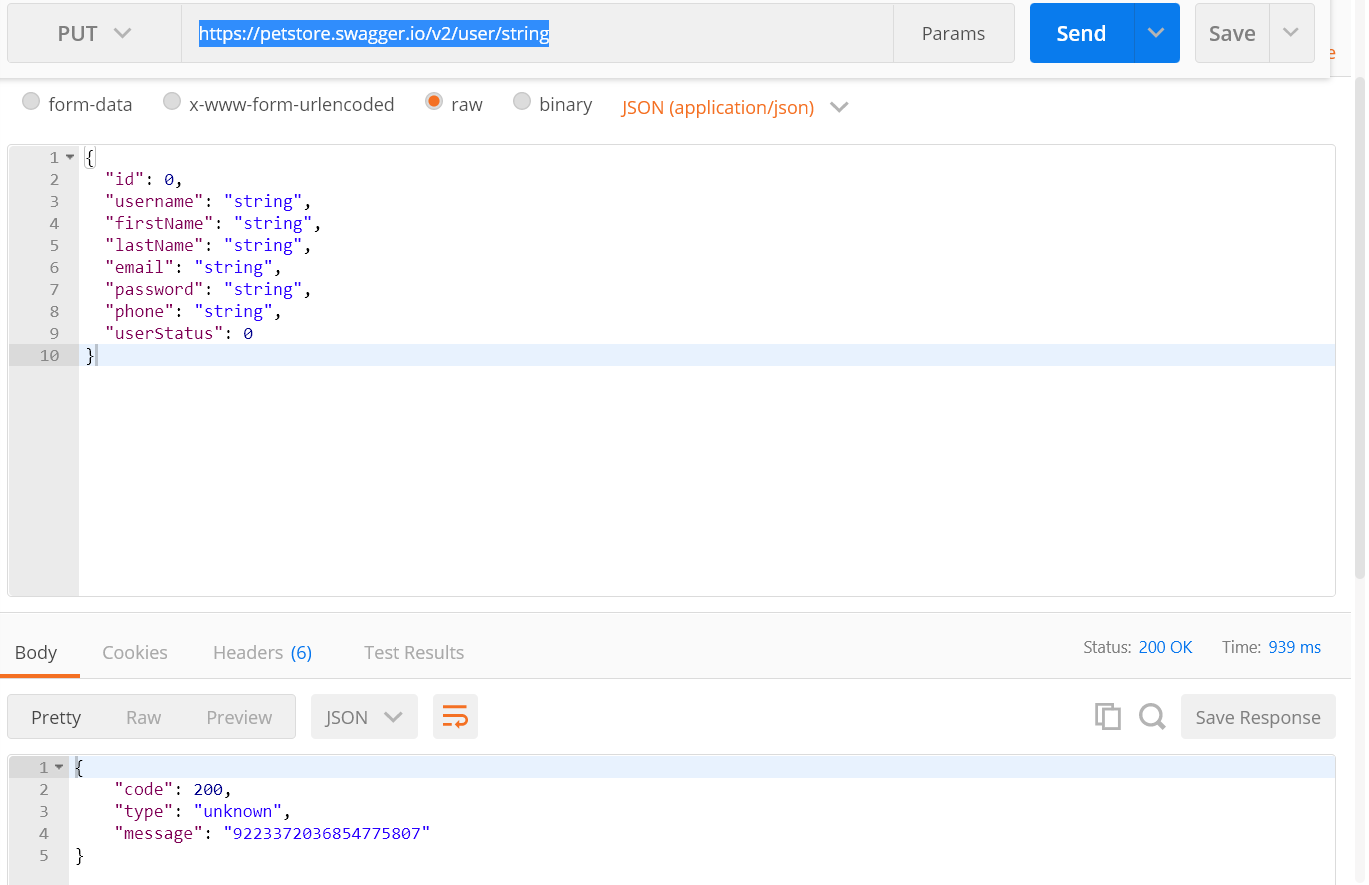
}

]

Pada postman berstatus 200 OK yang artinya data berhasil ditambahkan.

* **PUT**

**-** URL : <https://petstore.swagger.io/v2/user/string>

****

Method ini digunakan untuk mengubah data atau update data, data yang akan diubah pada data user ini adalah dengan username string. Value yang diberikan

{

"id": 0,

"username": "string",

"firstName": "string",

"lastName": "string",

"email": "string",

"password": "string",

"phone": "string",

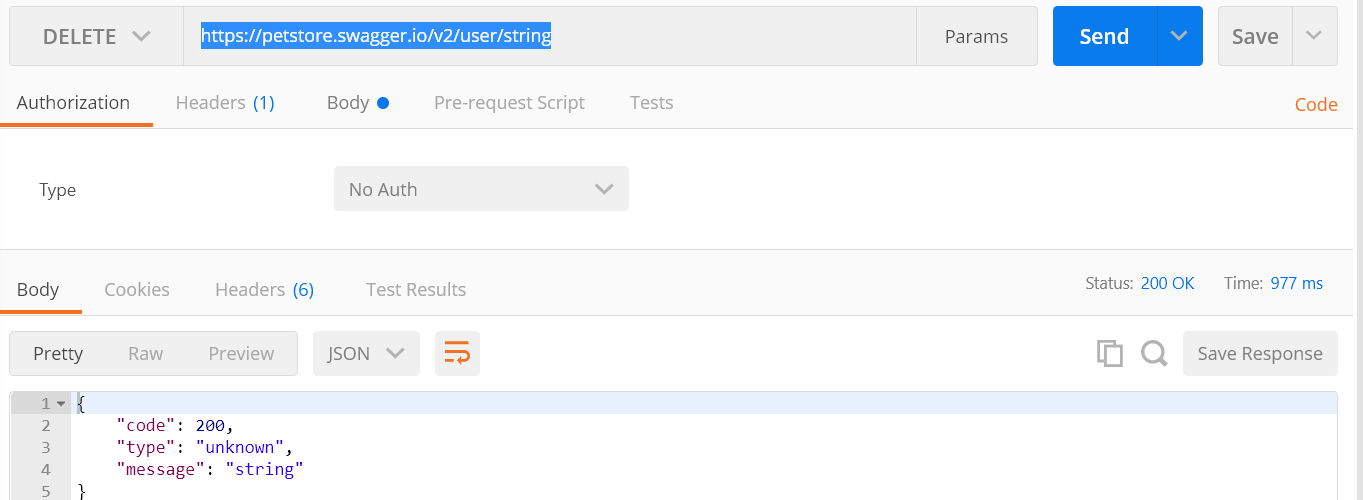
"userStatus": 0

}

Pada postman berstatus 200 OK yang artinya data berhasil diubah atau di-update.

* **DELETE**

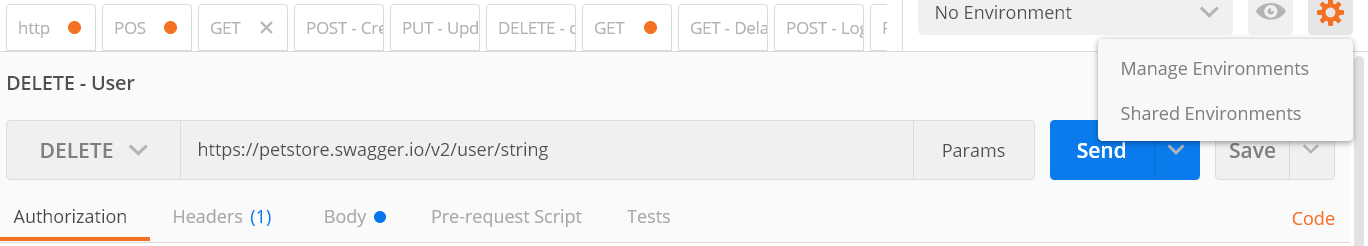
**-** URL : <https://petstore.swagger.io/v2/user/string>

****

Method ini digunakan untuk menghapus data, pada gambar di atas data yang dihapus dengan username string. Pada postman berstatus 200, yang artinya username string telah berhasil dihapus.

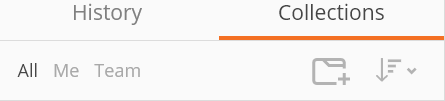
1. Explain the postman parts

* **Setting Environment And Collection**

****

Setting environment ini akan memudahkan kita dalam membedakan environment API dev dan API live.

* **Collection**



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.

* **Add a Request**

Pada bagian ini kita akan membuat request dengan method GET, POST, PUT, DELETE.

* GET → Berfungsi untuk memperoleh data
* POST → Berfungsi untuk menambahkan / create data
* PUT → Berfungsi untuk mengubah data / update data
* DELETE → Berfungsi untuk menghapus data
* **New**

Kita dapat membuat *request, collection,* atau *environment* baru pada bagian ini.

* **Import**

Import digunakan untuk mengimport kumpulan atau *collection* API dari file, folder, link, raw text, ataupun fitur terbaru dapat mengimport collection API dari code repository.

* **Open New**

Open new digunakan untuk open new tab, postman window atau runner window

* **History**

Request-request API yang sebelumnya sudah diuji akan tersimpan di bagian history ini.

* **Request Tab**

Request tab digunakan untuk menampilkan tab atau judul request API yang sedang dikerjakan.

* **Request URL**

Disebut juga dengan sebutan ‘endpoint’ , merupakan tujuan API yang akan diuji.

* **Send**

Send digunakan untuk menjalankan perintah menuju URL yang tertulis.

* **Save**

Save digunakan untuk menyimpan request baru pada collection ataupun menyimpan request baru yang telah mengalami perubahan.

* **Params**

Kita dapat menambah parameter tertentu apabila diperlukan disini (biasanya jika kita menggunakan request dalam bentuk GET).

* **Authorization**

Authorization digunakan untuk mengakses API terkadang dibutuhkan otorisasi yang tepat seperti bentuk bearer token, nama, dan password.

* **Headers**

Headers digunakan untuk pengaturan headers.

* **Body**

Dapat memberikan detail-detail tertentu yang akan melengkapi sebuah request ( biasanya apabila kita menggunakan request POST).

* **Response**

Response merupakan output yang akan didapat setelah menjalankan sebuah request. Response bisa jadi sukses (sesuai dengan apa yang diharapkan), bisa juga gagal.