Jared’s Amazing API

GET Requests

GET requests will return the user data based upon the routing and inputted URL parameters.

GET all authors - /authors

Sending a GET request to the /authors route will return a JSON array of every author in the database. It might look like this


```
[
    {
        "au_id": "172-32-1176",
        "au_lname": "Johnathan",
        "au_fname": "Joestar",
        "phone": "408 496-7223",
        "address": "10932 Bigge Rd.",
        "city": "Menlo Park",
        "state": "CA",
        "zip": "94025",
        "contract": 1
    },
    {
        "au_id": "172-32-1337",
        "au_lname": "John",
        "au_fname": "Johnson",
        "phone": "408 496-7223",
        "address": "10932 Bigge Rd.",
        "city": "Menlo Park",
        "state": "CA",
        "zip": "94025",
        "contract": 1
    },
    {
        "au_id": "213-46-8915",
        "au_lname": "Green",
        "au_fname": "Marjorie",
        "phone": "415 986-7020",
        "address": "309 63rd St. #411",
        "city": "Oakland",
        "state": "CA",
        "zip": "94618",
        "contract": 1
    },
    {
        "au_id": "238-95-7766",
```

GET a specific author - /authors/{Insert ID here}

Sending a GET request with an author's ID in the url will return a JSON object with the specified authors details. The request will look like this: **/authors/172-84-4843** and will return something like this

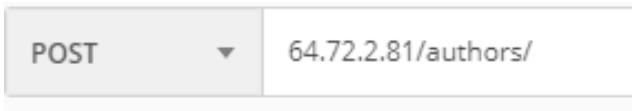

```
[
    {
        "au_id": "172-32-1176",
        "au_lname": "Johnathan",
        "au_fname": "Joestar",
        "phone": "408 496-7223",
        "address": "10932 Bigge Rd.",
        "city": "Menlo Park",
        "state": "CA",
        "zip": "94025",
        "contract": 1
    }
]
```


POST Requests

Sending a POST request with an attached JSON of an author object will add said author into the database. Though be warned the author object has some formatting requirements that will be explained further below.

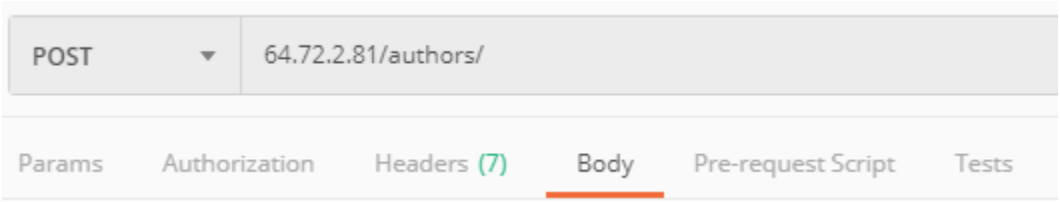
POST a new author - /authors

To utilize the POST request a JSON object containing an author will need to be attached to the POST request before being sent. The HTTP request app we'll be using in this example is Postman. First begin by creating a **POST** request.

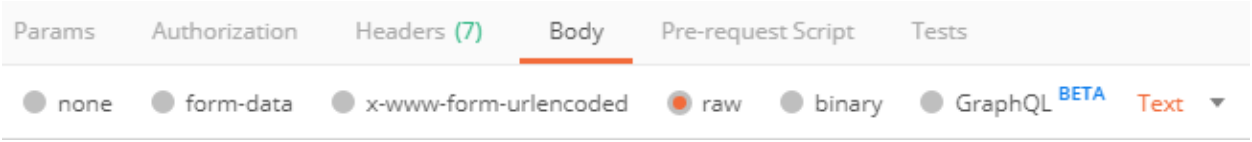


3

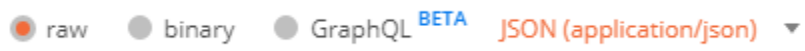
Next change your selection just below from **params** to **body**.



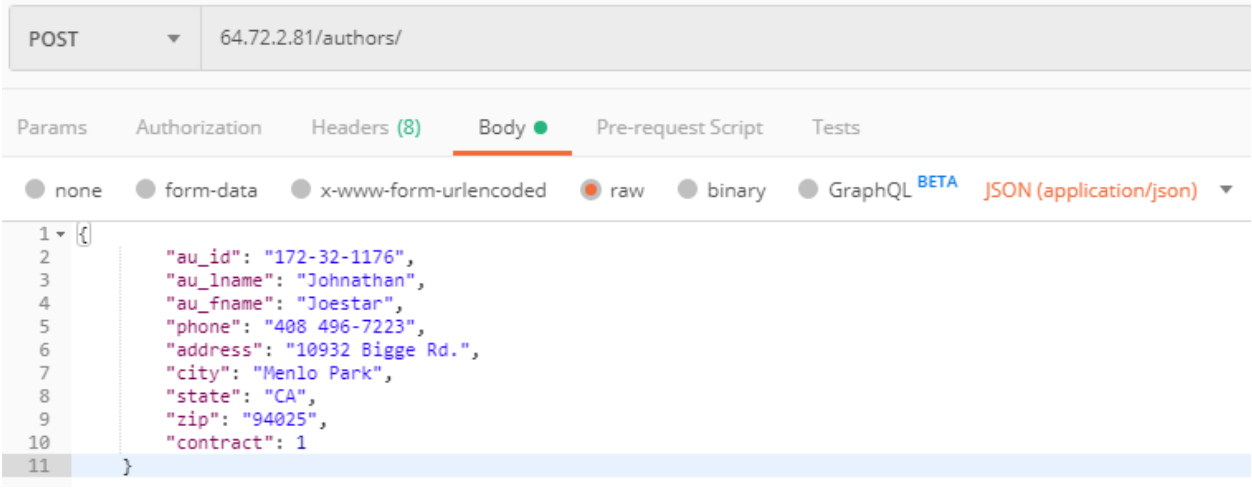
Next change the body type from **none** to **raw**.



Next change the text type from **Text** to **Json.**



Next create a JSON representation of our author object. I recommend sending a GET request for authors and then copying one from the returned array as a template to edit.



At this point you may edit this template to create a new author. The formatting will now be explained.

An author object has 4 requirements when being created, **1.** The ID must be unique and not already in the database. **2.** The ID must be formatted as such **XXX-XX-XXXX** where **X** is any number 0-9. **3.** The author's phone number must be formatted as such **XXX XXX-XXXX** where **X** is any number 0-9. **4.** The author's zip code must be formatted as such **XXXXX** where **X** is any number 0-9.

PUT Requests

PUT requests may be used to update an existing author in the database.

PUT an existing author - /authors

The PUT request to /authors may be used to update an existing author in the database. In its current implementation **ONLY THE AUTHORS LAST NAME, FIRST NAME, AND CONTRACT** may be updated. To update a currently existing author follow the steps shown above in the POST request to set up your Postman to up to the point of copying and pasting an author template

The JSON will resemble you creating a new author.

```
{
    "au_id": "172-32-1176",
    "au_lname": "Johnathan",
    "au_fname": "Joestar",
    "phone": "408 496-7223",
    "address": "10932 Bigge Rd.",
    "city": "Menlo Park",
    "state": "CA",
    "zip": "94025",
    "contract": 1
}
```

DO NOT EDIT THE ID, only change the first name, last name, and contract. Any other changes will not be updated to the database.

DELETE Requests

The DELETE request can be used to remove an author from the database. **ONLY AUTHORS THAT YOU HAVE ADDED TO THE DATABASE MAY BE REMOVED, ANY PRE-EXISTING AUTHORS CANNOT BE REMOVED DUE TO FOREIGN KEYS FROM OTHER TABLES RELYING ON THE PRE-EXISTING AUTHORS.**

DELETE an existing author - /authors/{Insert ID here}

To DELETE an existing author is simple and easy. Send a DELETE request to the /authors route with an included ID in the url. It should look something like this **/authors/193-43-5224.**

Simple as that and the author will be removed from the database. Once again, do not delete any authors that already exist within the database as there are other tables in the database that require them to exist. Therefore you will receive an error upon doing so and the express server will crash.