JSON – Server API

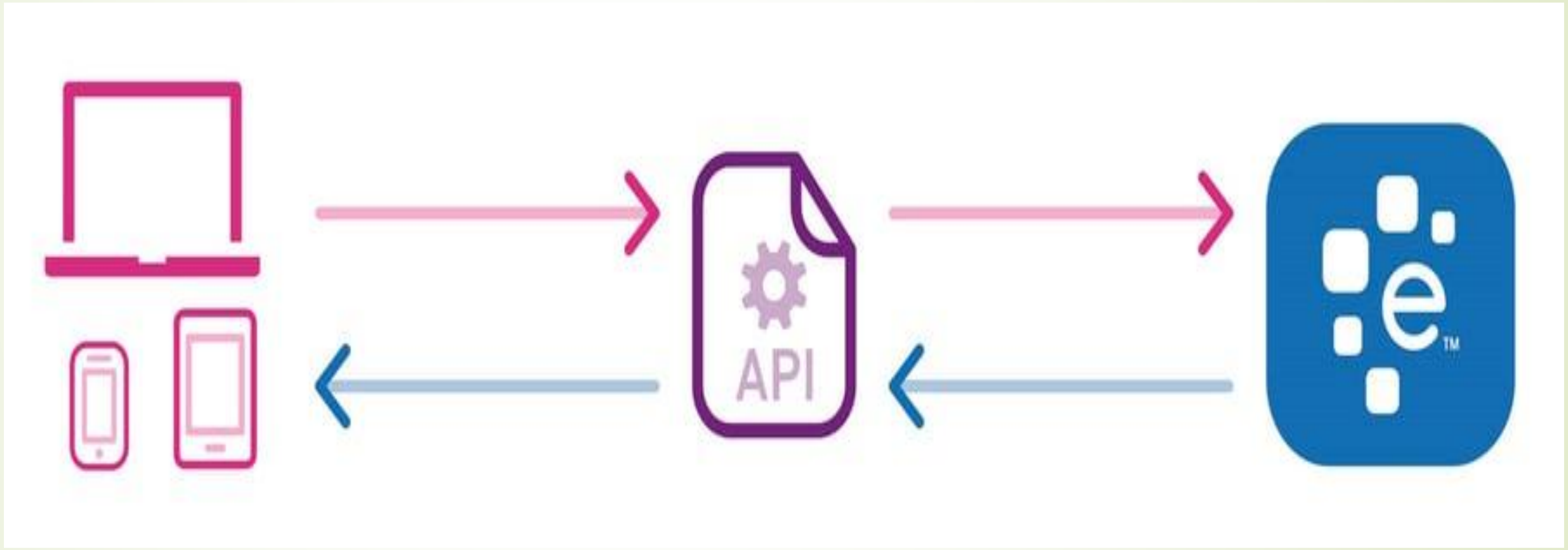
JSON

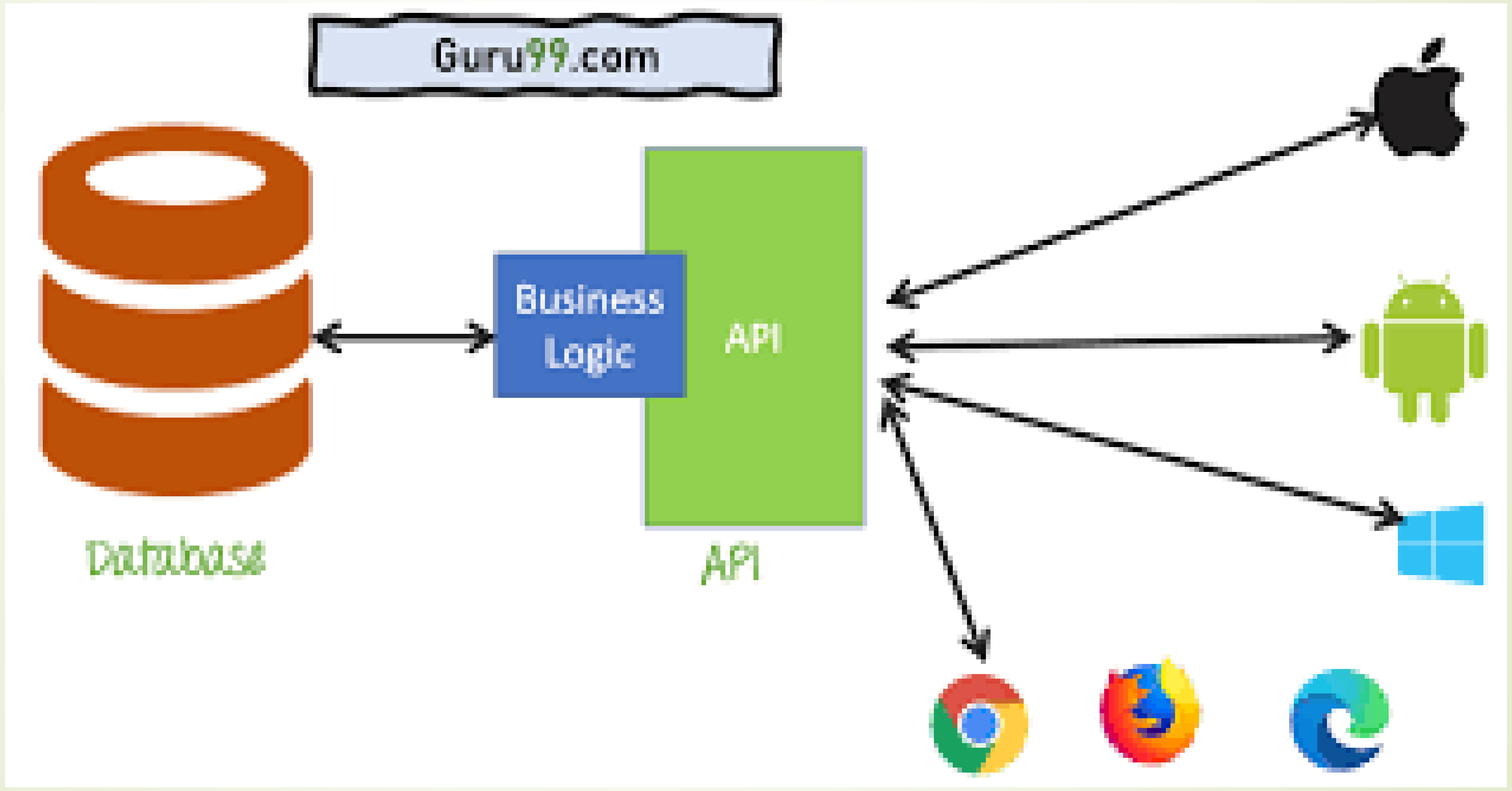
- JSON **stands for JavaScript Object Notation**. JSON is a lightweight format for storing and transporting data.
- JSON is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays. It is a common data format with diverse uses in electronic data interchange, including that of web applications with servers.

```
{
    ‘students’: [
        {“roll”:121, “fname”: “john”, “lname”: ”Patel”, “city”: “Rajkot”},
        {“roll”:122, “fname”: “jolly”, “lname”: ”Rathod”, “city”: “Rajkot”},
        {“roll”:123, “fname”: “jenish”, “lname”: ”Patel”, “city”: “Rajkot”},
        {“roll”:124, “fname”: “Sahid”, “lname”: ”Juneja”, “city”: “Rajkot”},
        {“roll”:125, “fname”: “Man”, “lname”: ”Raval”, “city”: “Rajkot”},
        {“roll”:126, “fname”: “Yash”, “lname”: ”Patel”, “city”: “Rajkot”},
        {“roll”:127, “fname”: “jasmin”, “lname”: ”Jadeja”, “city”: “Rajkot”},
        {“roll”:128, “fname”: “jack”, “lname”: ”Joshi”, “city”: “Rajkot”}
    ]
}
```

API

- API stands for **Application Programming Interface**. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses.


API


Guru99.com
Business Logic
API
Database
API

JSON - Server

- JSON Server is **a Node Module that you can use to create demo rest json webservice in less than a minute**. All you need is a JSON file for sample data

- ***Representational State Transfer** (REST) is a software architecture that imposes conditions on how an API should work. REST was initially created as a guideline to manage communication on a complex network like the internet.

How to install JSON-Server

- Open command prompt with administrative permission.
- Run following command

  ```
  npm install json-server -g
  ```

  *you must need active internet connection while install json-server
- Go to destination folder to store json file.

  ```
  mkdir jsondata
  cd jsondata
  json-server –watch jsonfileName.json
  ```

- Leave command prompt as it is running
- Open a browser and enter URL localhost:3000
- You found some pre defined API with fake data
- Open jsondata.json file in text editor
- Remove existing data ana create new API as per your testing requirements.
- Now test your newly created API.

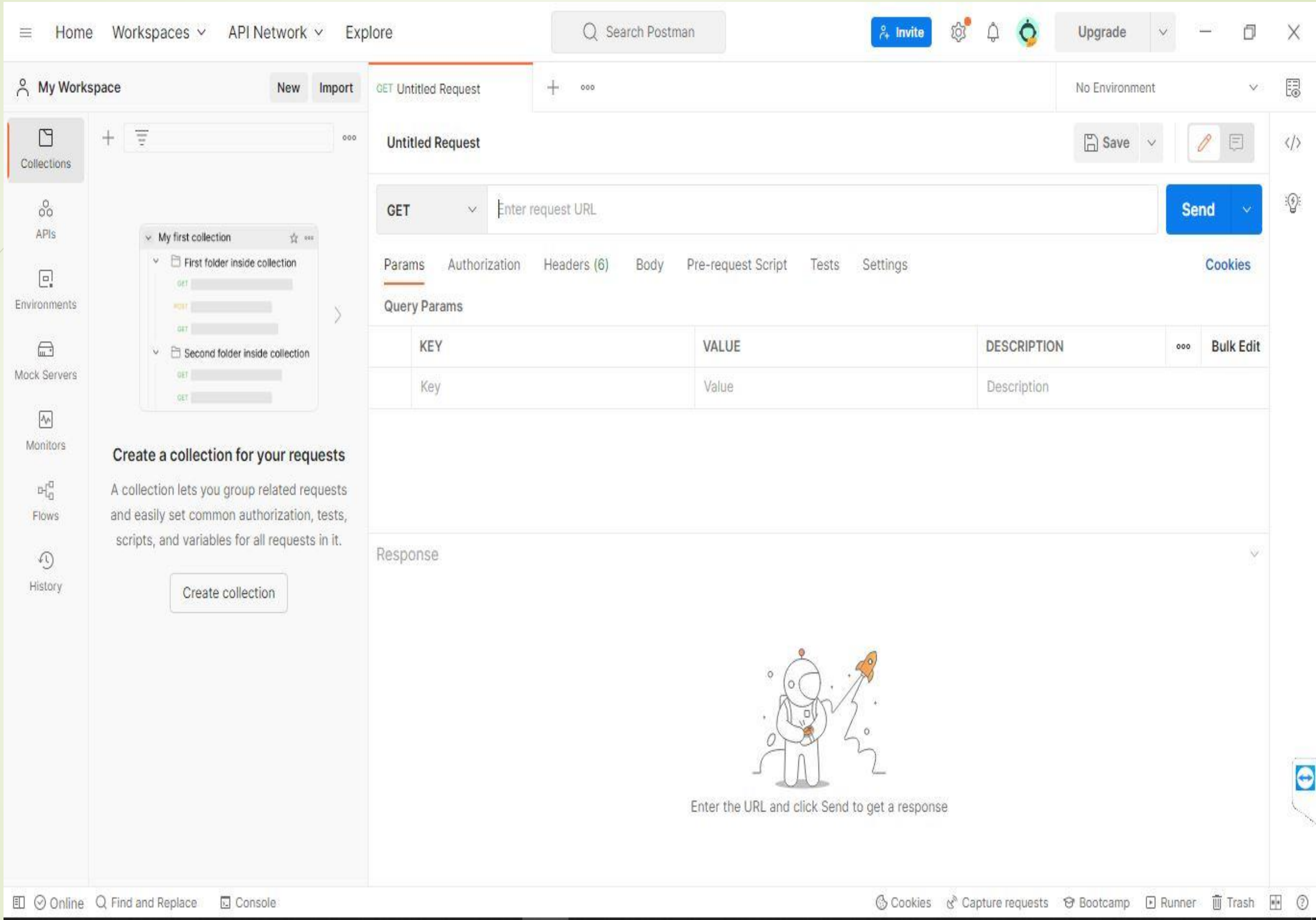
  localhost:3000/newAPIname

Basic API Operations

GET	:	get data from API
POST	:	to Add Data in API
PUT	:	update API data
DELETE	:	to remove data from API.

POSTMAN

- Postman is an API platform for developers to design, build, test and iterate their APIs.



Create a collection for your requests

A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.

Create collection

KEY	VALUE	DESCRIPTION
Key	Value	Description

Enter the URL and click Send to get a response

To get data from API via postman

- Create a new request in postman and must set request type to GET.

- localhost:3000/yourAPIname

  Or

- localhost:3000/yourAPIname/id

To post data with Postman

- Create new request in postman must set method to POST and set URL of Your API.
- select data to raw data and set data format to JSON in body of API Request.
- Add JSON data and click on send you will get posted data with new id.

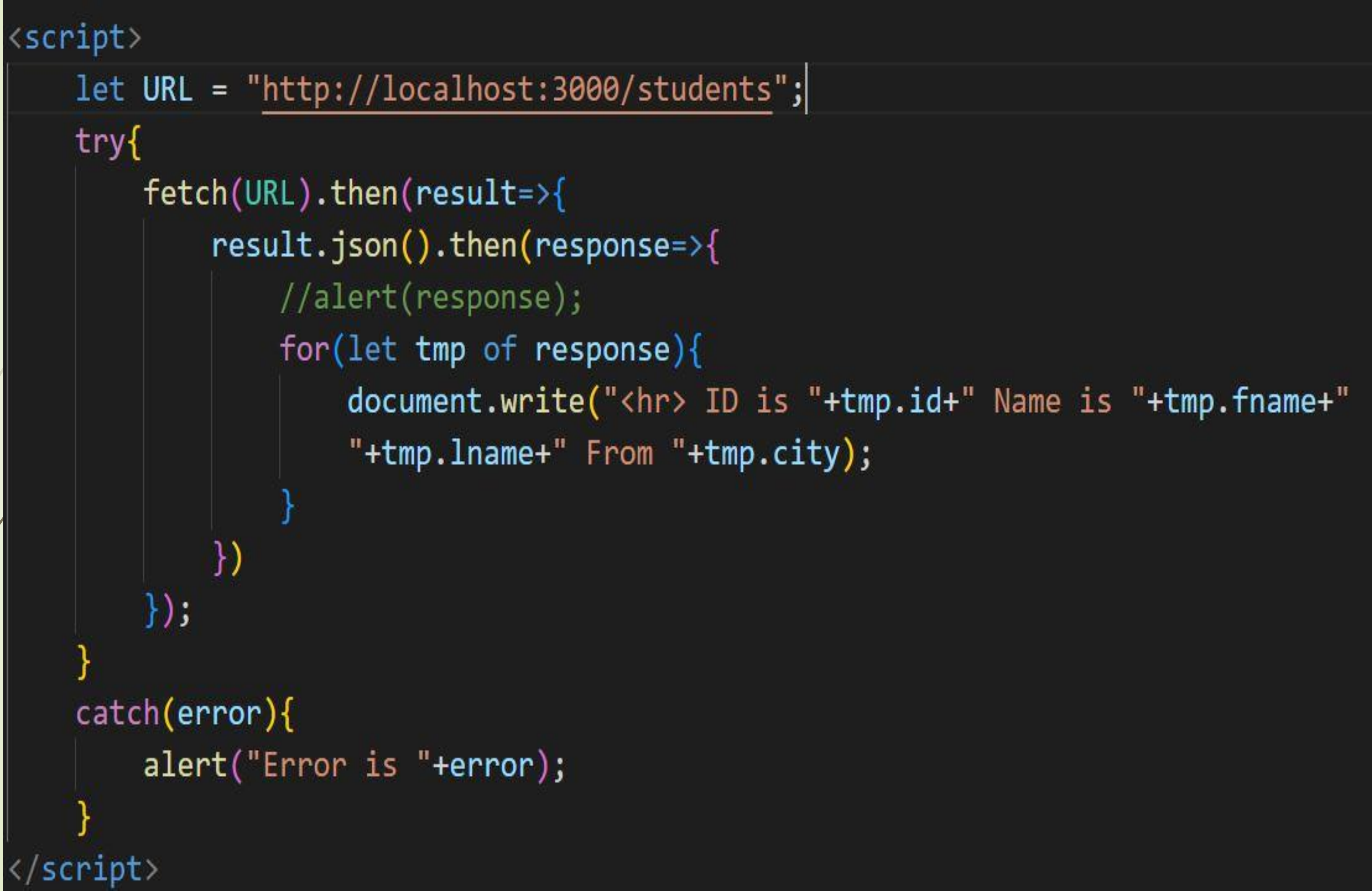
Update data with API – PUT method

- create new request for API and set URL and set request method to PUT.
- Must set id for update data and set data in body of request with raw format and JSON type.

To Delete data with API

- create new request for API and set URL and set request method to DELETE.

JSON - Server data fetch in Java Script



```
<script>
    let URL = "http://localhost:3000/students";
    try{
        fetch(URL).then(result=>{
            result.json().then(response=>{
                //alert(response);
                for(let tmp of response){
                    document.write("<hr> ID is "+tmp.id+" Name is "+tmp.fname+"
                    "+tmp.lname+" From "+tmp.city);
                }
            })
        });
    }
    catch(error){
        alert("Error is "+error);
    }
</script>
```

Thank you.