Node JS CRUD with MySQL Tutorial Example

By Hardik Savani March 14, 2022 Category : Node JS

Hello,

Today, node js crud example with mysql is our main topic. i explained simply step by step node.js express mysql crud example. you'll learn crud operation in node js using express mysql. this example will help you node js crud api with mysql. you will do the following things for node js crud api tutorial.

In this tutorial, we will create very simple way crud rest API using node js and mysql. we will use express, mysql, body-parser npm package for creating crud api with node.js and mysql. you will find following apis after follow full this example:

CRUD APIs

Method	EndPoint	Description
GET	api/items	Get All Items
GET	api/items/{id}	Get Single Item
POST	api/items	Create New Item
PUT	api/items/{id}	Update Item
Delete	api/items/{id}	Delete Item

Now you have to follow few step to create rest api with node js. we are giving you step by step from scratch so let's follow bellow step.

Step 1: Create Database and Table

In first step, you must have installed mysql in your system. You can use tools like SQLyog, PHPMyAdmin or similar tools.

Now you have to create new database using bellow step query, so we will use this database in our example:

```
CREATE DATABASE node_restapi;
```

After creating database we need to create items table with following query in "node_restapi" database:

Create Table:

```
CREATE TABLE `items` (
  `id` bigint UNSIGNED NOT NULL,
  `title` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `body` text COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

ALTER TABLE `items`
  ADD PRIMARY KEY (`id`);

ALTER TABLE `items`
  MODIFY `id` bigint UNSIGNED NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;
```

After created successfully items table, you can add some dummy records as like bellow:

Add Dummy Records to Table:

```
INSERT INTO `items` (`id`, `title`, `body`) VALUES
(1, 'Title 1', 'Body 1'),
(2, 'Title 2', 'Body 2'),
(3, 'Title 3', 'Body 3'),
(4, 'Title 4', 'Body 4');
```

Step 2: Create Node App

run bellow command and create node app.

```
mkdir my-app
cd my-app
npm init
```

Read Also: Node JS Rename All Files in Folder Example

Step 3: Install express mysql body-parser

Here we will install express, mysql, body-parser npm package.

```
npm install --save express mysql body-parser
```

Step 4: Create app.js file

app.js

```
const express = require('express');
const bodyParser = require('body-parser');
const app = express();
const mysql = require('mysql');
```

```
--------------------------------------------
--------------------------------------------*/
app.use(bodyParser.json());

/*------------------------------------------
--------------------------------------------
Database Connection
--------------------------------------------
--------------------------------------------*/
const conn = mysql.createConnection({
  host: 'localhost',
  user: 'root', /* MySQL User */
  password: 'root', /* MySQL Password */
  database: 'node_restapi' /* MySQL Database */
});

/*------------------------------------------
--------------------------------------------
Shows Mysql Connect
--------------------------------------------
--------------------------------------------*/
conn.connect((err) =>{
  if(err) throw err;
  console.log('Mysql Connected with App...');
});

/**
 * Get All Items
 *
 * @return response()
 */
app.get('/api/items',(req, res) => {
  let sqlQuery = "SELECT * FROM items";

  let query = conn.query(sqlQuery, (err, results) => {
    if(err) throw err;
    res.send(apiResponse(results));
  });
});

/**
 * Get Single Item
 *
 * @return response()
 */
app.get('/api/items/:id',(req, res) => {
  let sqlQuery = "SELECT * FROM items WHERE id=" + req.params.id;

  let query = conn.query(sqlQuery, (err, results) => {
    if(err) throw err;
    res.send(apiResponse(results));
  });
});

/**
 * Create New Item
 *
 * @return response()
 */
app.post('/api/items',(req, res) => {
  let data = {title: req.body.title, body: req.body.body};

  let sqlQuery = "INSERT INTO items SET ?";

  let query = conn.query(sqlQuery, data,(err, results) => {
    if(err) throw err;
    res.send(apiResponse(results));
 * Update Item
 *
 * @return response()
 */
app.put('/api/items/:id',(req, res) => {
  let sqlQuery = "UPDATE items SET title='"+req.body.title+"', body='"+req.body.body+"' WHERE id="+req.params.id;

  let query = conn.query(sqlQuery, (err, results) => {
    if(err) throw err;
    res.send(apiResponse(results));
  });
});

/**
 * Delete Item
 *
 * @return response()
 */
app.delete('/api/items/:id',(req, res) => {
  let sqlQuery = "DELETE FROM items WHERE id="+req.params.id+"";

  let query = conn.query(sqlQuery, (err, results) => {
    if(err) throw err;
      res.send(apiResponse(results));
  });
});

/**
 * API Response
 *
 * @return response()
 */
function apiResponse(results){
    return JSON.stringify({"status": 200, "error": null, "response": results});
}

/*--------------------------------------------
--------------------------------------------
Server listening
--------------------------------------------
--------------------------------------------*/
app.listen(3000,() =>{
  console.log('Server started on port 3000...');
});
```

Run Node App:

All the required steps have been done, now you have to type the given below command and hit enter to run the Node app:

```
node app.js
```

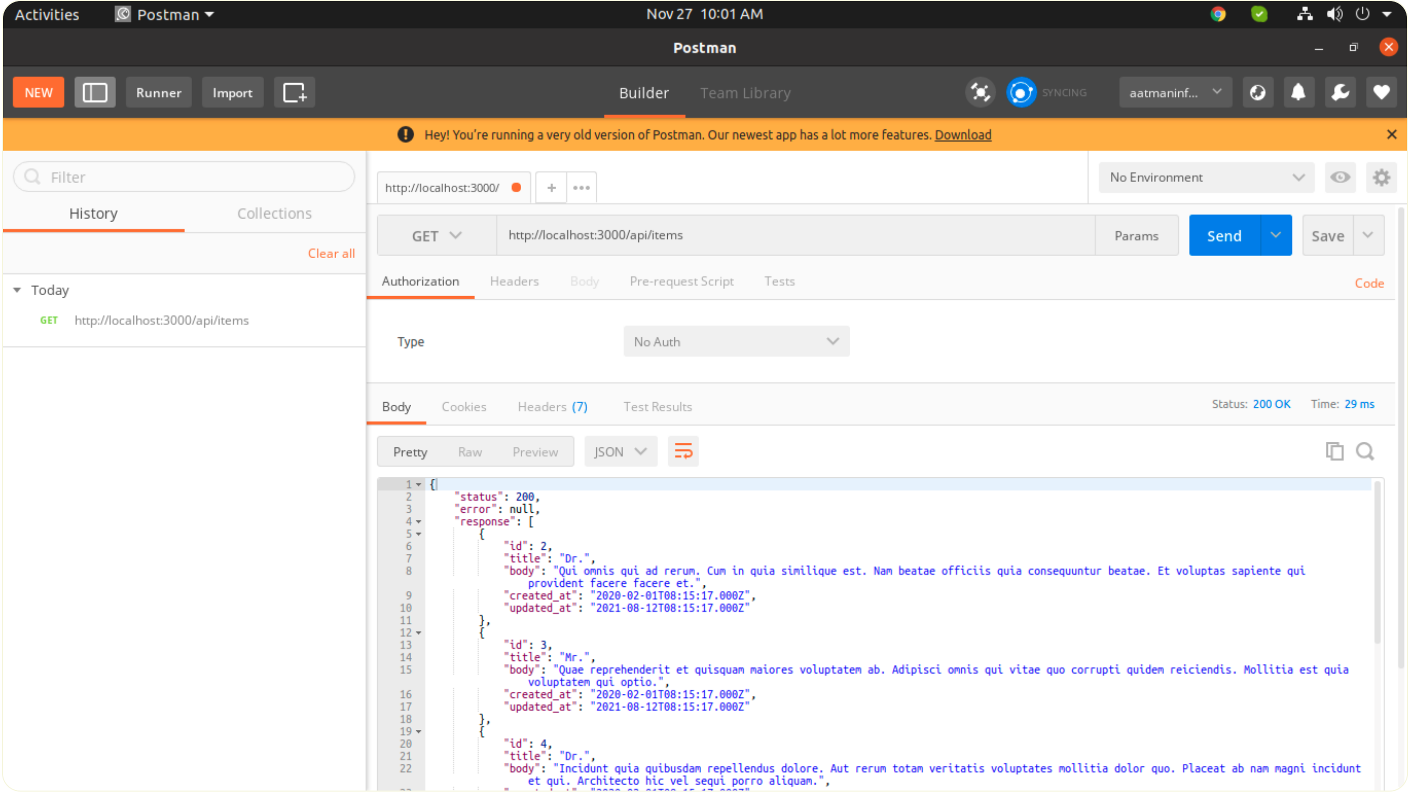
Now, Go to your web browser, type the given URL and check it's working:

```
http://localhost:3000
```

Now, Go to your web Postman and check all APIs as like bellow:

1) Get Item API:

```
Method:GET
URL: http://localhost:3000/api/items
```



GET http://localhost:3000/api/items

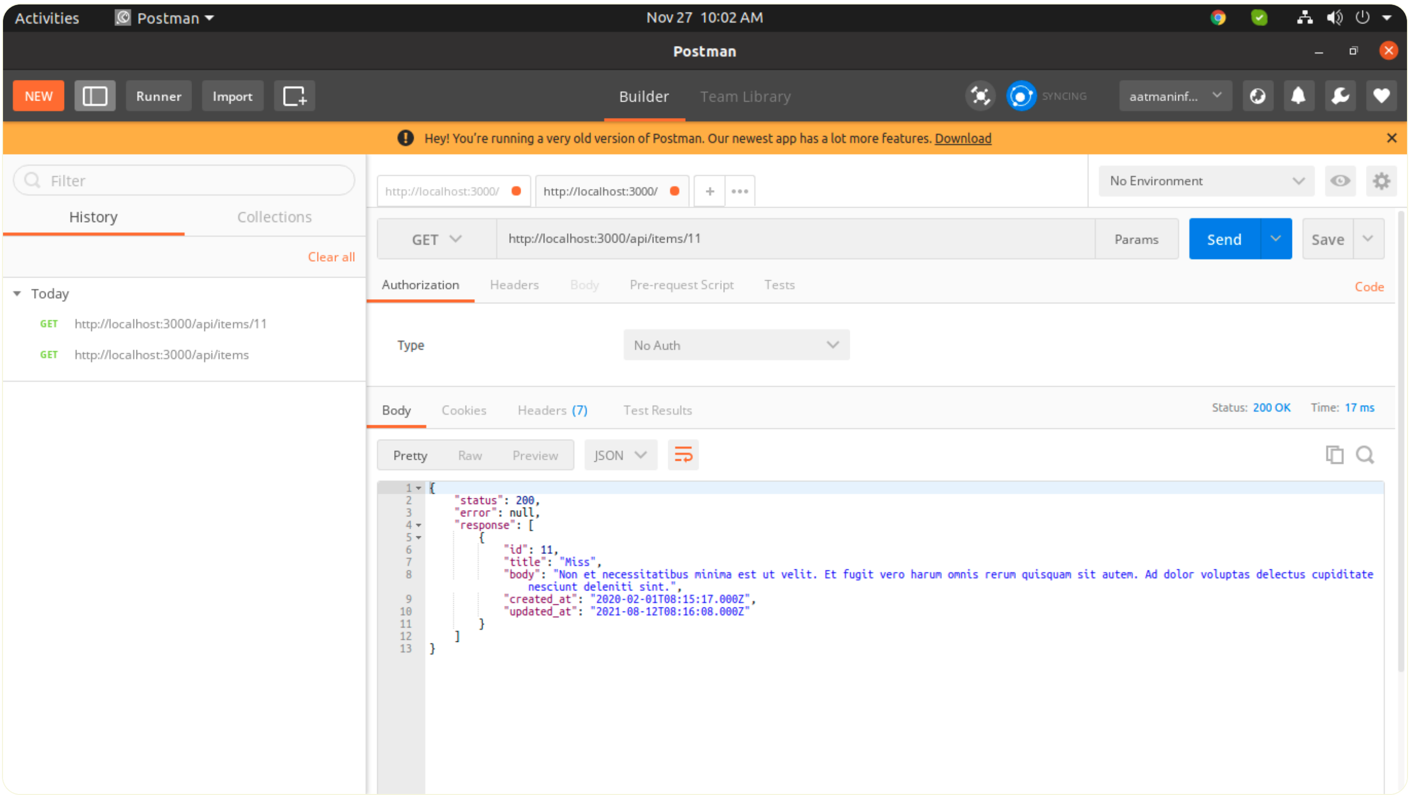
Type: No Auth

Status: 200 OK Time: 29 ms

```
{
    "status": 200,
    "error": null,
    "response": [
        {
            "id": 2,
            "title": "Dr.",
            "body": "Qui omnis qui ad rerum. Cum in quia similique est. Nam beatae officiis quia consequuntur beatae. Et voluptas sapiente qui
                provident facere facere et.",
            "created_at": "2020-02-01T08:15:17.000Z",
            "updated_at": "2021-08-12T08:15:17.000Z"
        },
        {
            "id": 3,
            "title": "Mr.",
            "body": "Quae reprehenderit et quisquam maiores voluptatem ab. Adipisci omnis qui vitae quo corrupti quidem reiciendis. Mollitia est quia
                voluptatem qui optio.",
            "created_at": "2020-02-01T08:15:17.000Z",
            "updated_at": "2021-08-12T08:15:17.000Z"
        },
        {
            "id": 4,
            "title": "Dr.",
            "body": "Incidunt quia quibusdam repellendus dolore. Aut rerum totam veritatis voluptates mollitia dolor quo. Placeat ab nam magni incidunt
                et qui. Architecto hic vel sequi porro aliquam.",
```

2) Get Single Item API:

```
Method:GET
URL: http://localhost:3000/api/items/{id}
```



GET http://localhost:3000/api/items/11

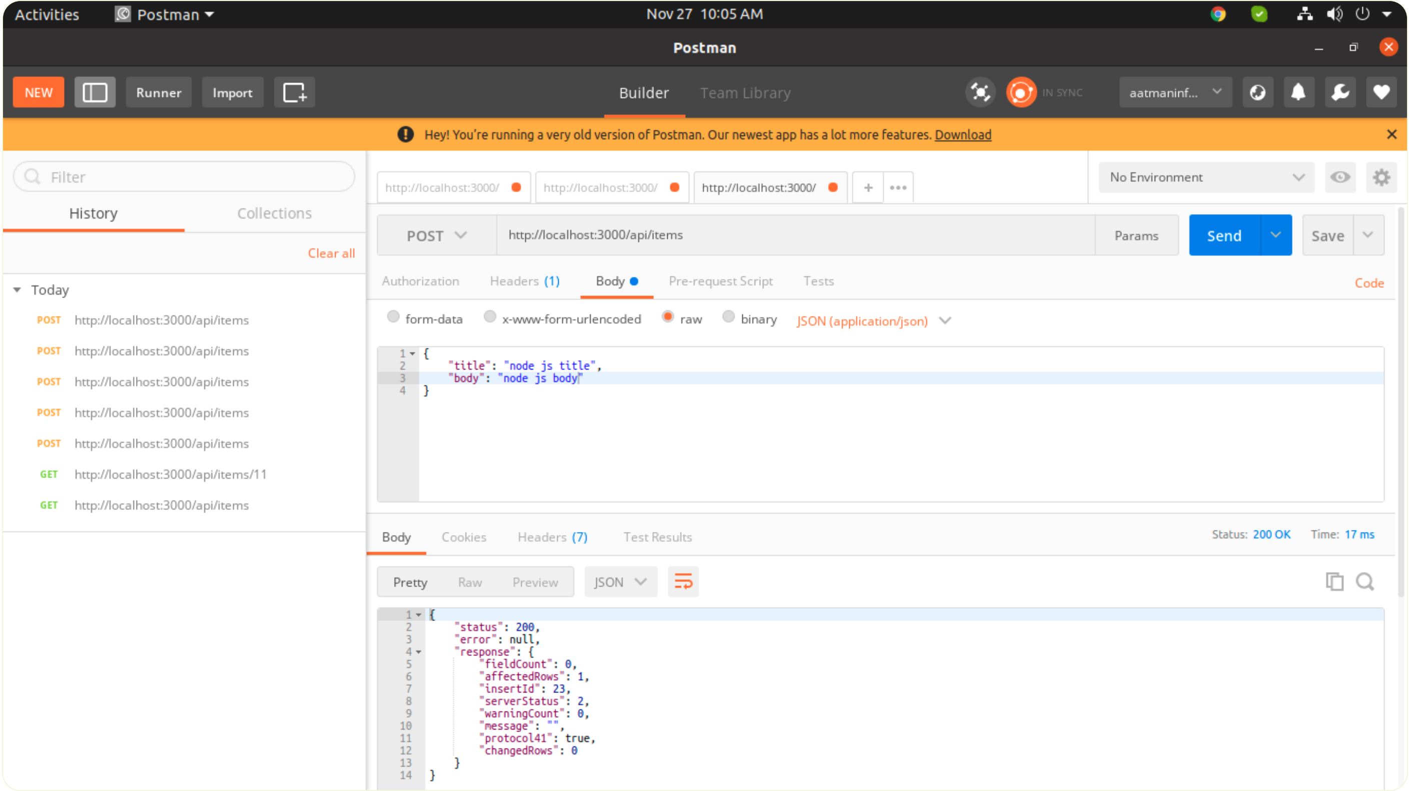
Type: No Auth

Status: 200 OK Time: 17 ms

```
{
    "status": 200,
    "error": null,
    "response": [
        {
            "id": 11,
            "title": "Miss",
            "body": "Non et necessitatibus minima est ut velit. Et fugit vero harum omnis rerum quisquam sit autem. Ad dolor voluptas delectus cupiditate
                nesciunt deleniti sint.",
            "created_at": "2020-02-01T08:15:17.000Z",
            "updated_at": "2021-08-12T08:16:08.000Z"
        }
    ]
}
```

3) Create Item API:

```
Method:POST
URL: http://localhost:3000/api/items
Data:
{
        "title": "node js title",
        "body": "node js body"
}
```



Today

POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
GET	http://localhost:3000/api/items/11
GET	http://localhost:3000/api/items

POST http://localhost:3000/api/items

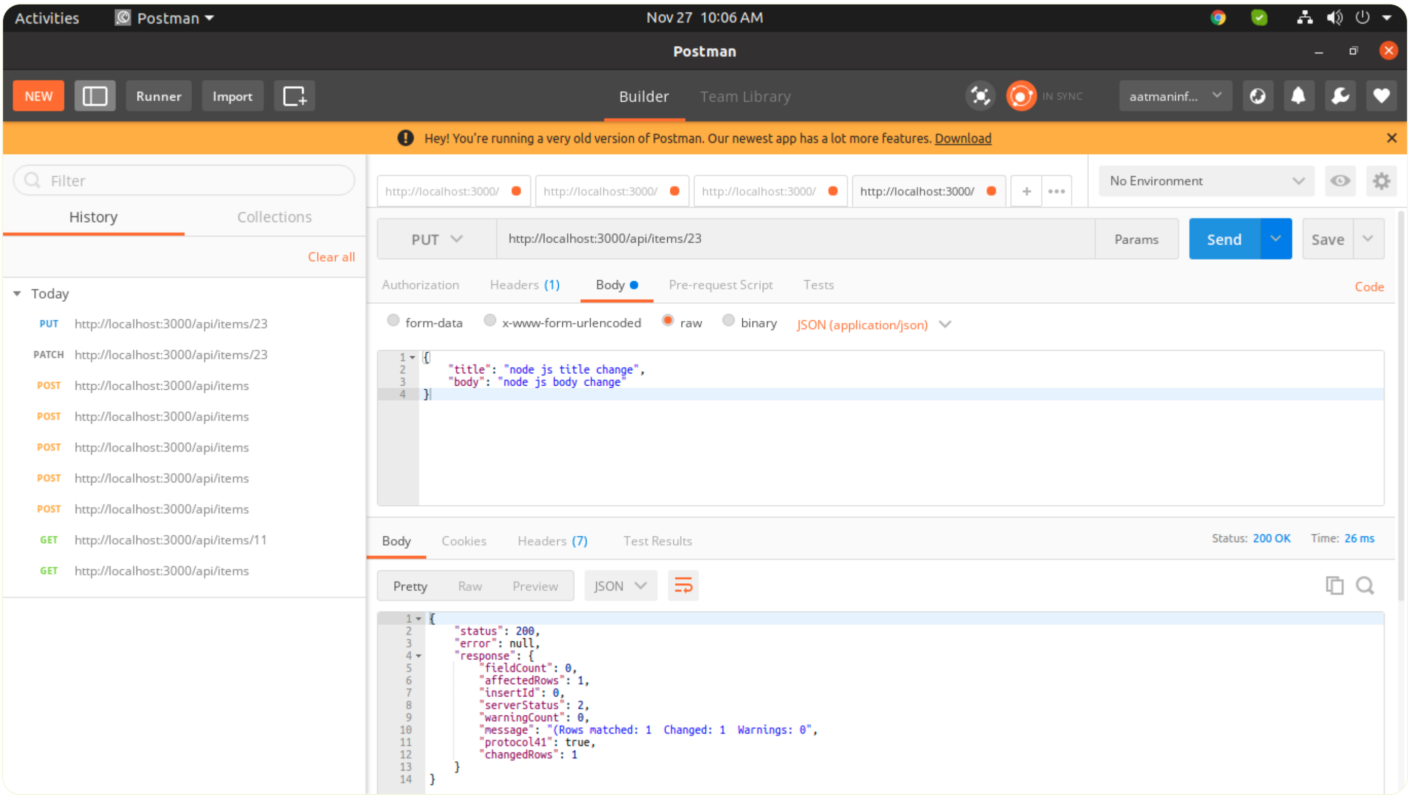
```
{
    "title": "node js title",
    "body": "node js body"
}
```

Status: 200 OK Time: 17 ms

```
{
    "status": 200,
    "error": null,
    "response": {
        "fieldCount": 0,
        "affectedRows": 1,
        "insertId": 23,
        "serverStatus": 2,
        "warningCount": 0,
        "message": "",
        "protocol41": true,
        "changedRows": 0
    }
}
```

4) Update Item API:

```
Method:PUT
URL: http://localhost:3000/api/items/{id}
Data:
{
        "title": "node js title",
        "body": "node js body"
}
```



Today

PUT	http://localhost:3000/api/items/23
PATCH	http://localhost:3000/api/items/23
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
POST	http://localhost:3000/api/items
GET	http://localhost:3000/api/items/11
GET	http://localhost:3000/api/items

PUT http://localhost:3000/api/items/23

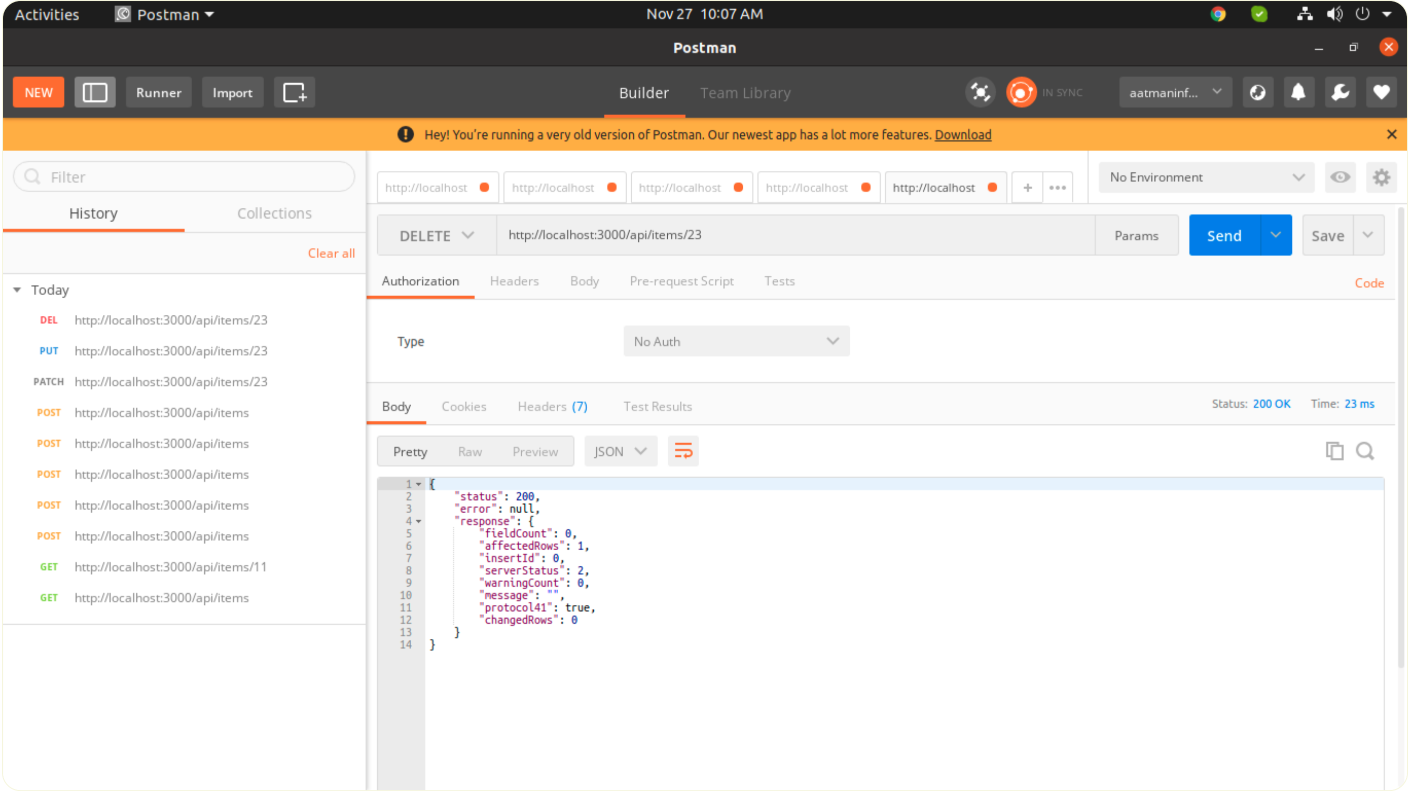
```
{
    "title": "node js title change",
    "body": "node js body change"
}
```

Status: 200 OK Time: 26 ms

```
{
    "status": 200,
    "error": null,
    "response": {
        "fieldCount": 0,
        "affectedRows": 1,
        "insertId": 0,
        "serverStatus": 2,
        "warningCount": 0,
        "message": "(Rows matched: 1  Changed: 1  Warnings: 0",
        "protocol41": true,
        "changedRows": 1
    }
}
```

5) Delete Item API:

> Read Also: Node JS Express Image Upload Rest API Example



DEL http://localhost:3000/api/items/23
PUT http://localhost:3000/api/items/23
PATCH http://localhost:3000/api/items/23
POST http://localhost:3000/api/items
POST http://localhost:3000/api/items
POST http://localhost:3000/api/items
POST http://localhost:3000/api/items
POST http://localhost:3000/api/items
GET http://localhost:3000/api/items/11
GET http://localhost:3000/api/items

DELETE http://localhost:3000/api/items/23

Type: No Auth

Status: 200 OK Time: 23 ms

```
{
    "status": 200,
    "error": null,
    "response": {
        "fieldCount": 0,
        "affectedRows": 1,
        "insertId": 0,
        "serverStatus": 2,
        "warningCount": 0,
        "message": "",
        "protocol41": true,
        "changedRows": 0
    }
}
```

Now you can check your own.

I hope it can help you...

Tags : CRUD Node JS

Hardik Savani

I'm a full-stack developer, entrepreneur and owner of Aatman Infotech. I live in India and I love to write tutorials and tips that can help to other artisan. I am a big fan of PHP, Laravel, Angular, Vue, Node, Javascript, JQuery, Codeigniter and Bootstrap from the early stage. I believe in Hardworking and Consistency.

Follow Me:

We are Recommending you

- How to Create a REST API using Node.js and MySQL?
- How to Create PDF File in Node JS?
- How to Create Basic API in Node JS Express?
- How to find Files by Extension in Node JS?
- How to Move File in Folder using Node JS?
- How to Rename a Folder in Node JS?
- How to Get File Size in Node JS?
- How to Copy File to Another Directory in Node JS?
- Node JS Rename All Files in Folder Example