

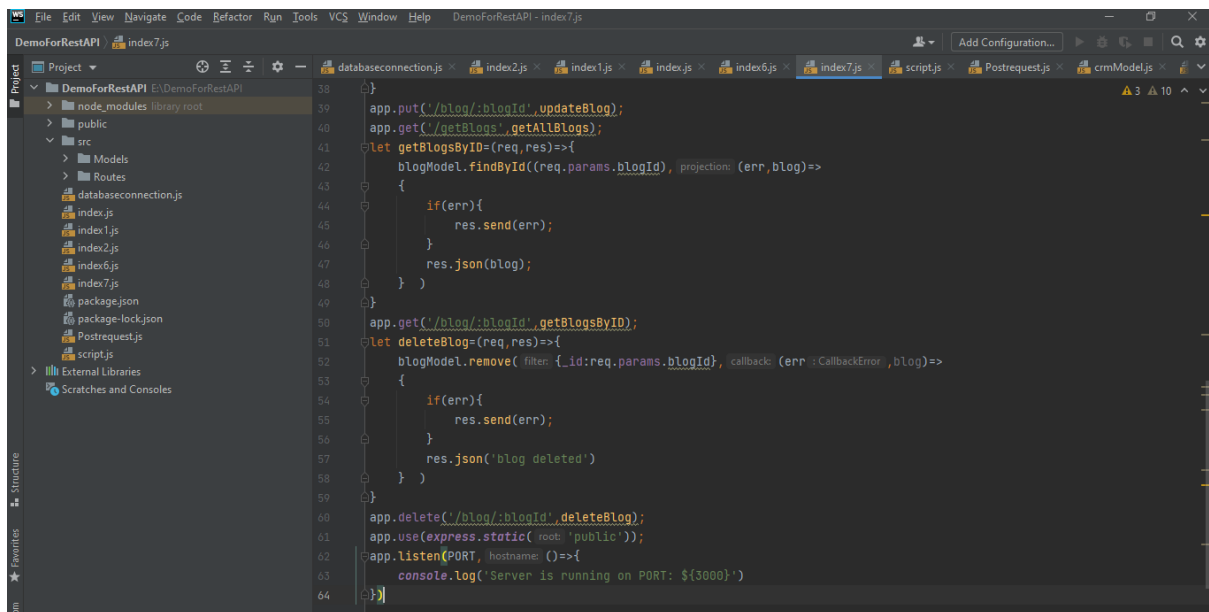
STEP 1: Firstly I had install three software .

1)IntelliJ Webstorm.

2)Mongo Db

3)Postman

STEP2: Written coding IntelliJ Webstrom.



```
38 }
39 app.put('/blog/:blogId', updateBlog);
40 app.get('/getBlogs', getAllBlogs);
41 let getBlogsById = (req, res) => {
42   blogModel.findById(req.params.blogId, projection: (err, blog) =>
43     {
44       if(err){
45         res.send(err);
46       }
47       res.json(blog);
48     }
49   )
50 app.get('/blog/:blogId', getBlogsById);
51 let deleteBlog = (req, res) => {
52   blogModel.remove({ _id: req.params.blogId }, callback: (err: CallbackError, blog) =>
53     {
54       if(err){
55         res.send(err);
56       }
57       res.json('blog deleted')
58     }
59   )
60 app.delete('/blog/:blogId', deleteBlog);
61 app.use(express.static('root: public'));
62 app.listen(PORT, hostname: () => {
63   console.log('Server is running on PORT: ${3000}')
64 })
```

STEP 3: Run code on server 3000 in cmd.

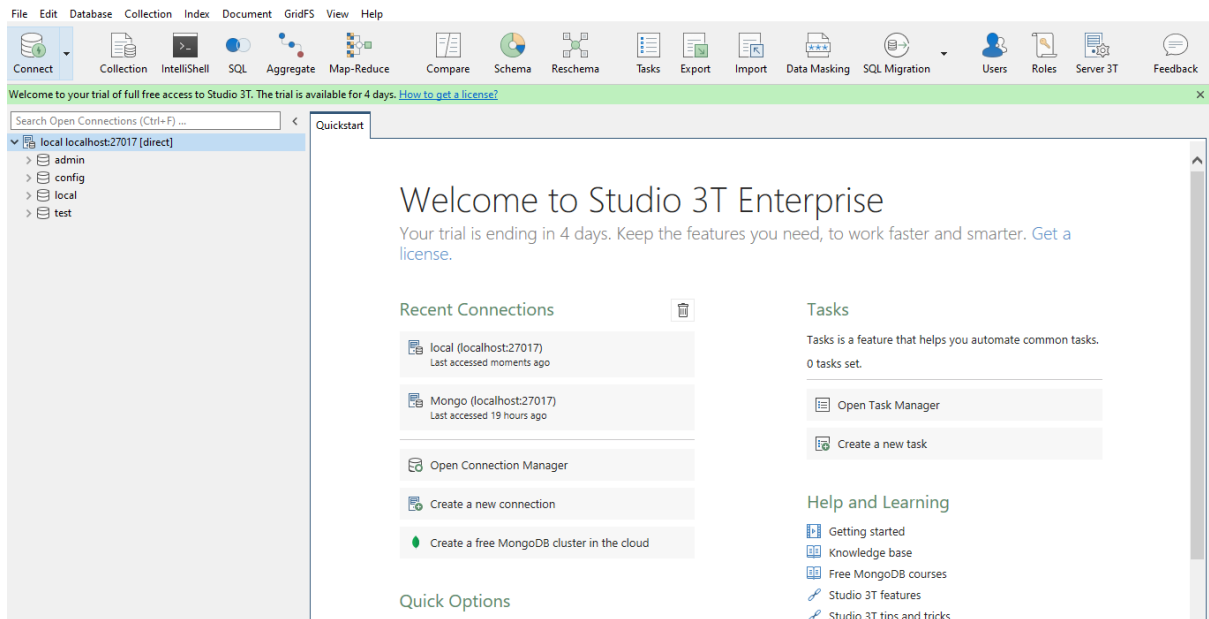
```
Microsoft Windows [Version 10.0.19041.985]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user>E:

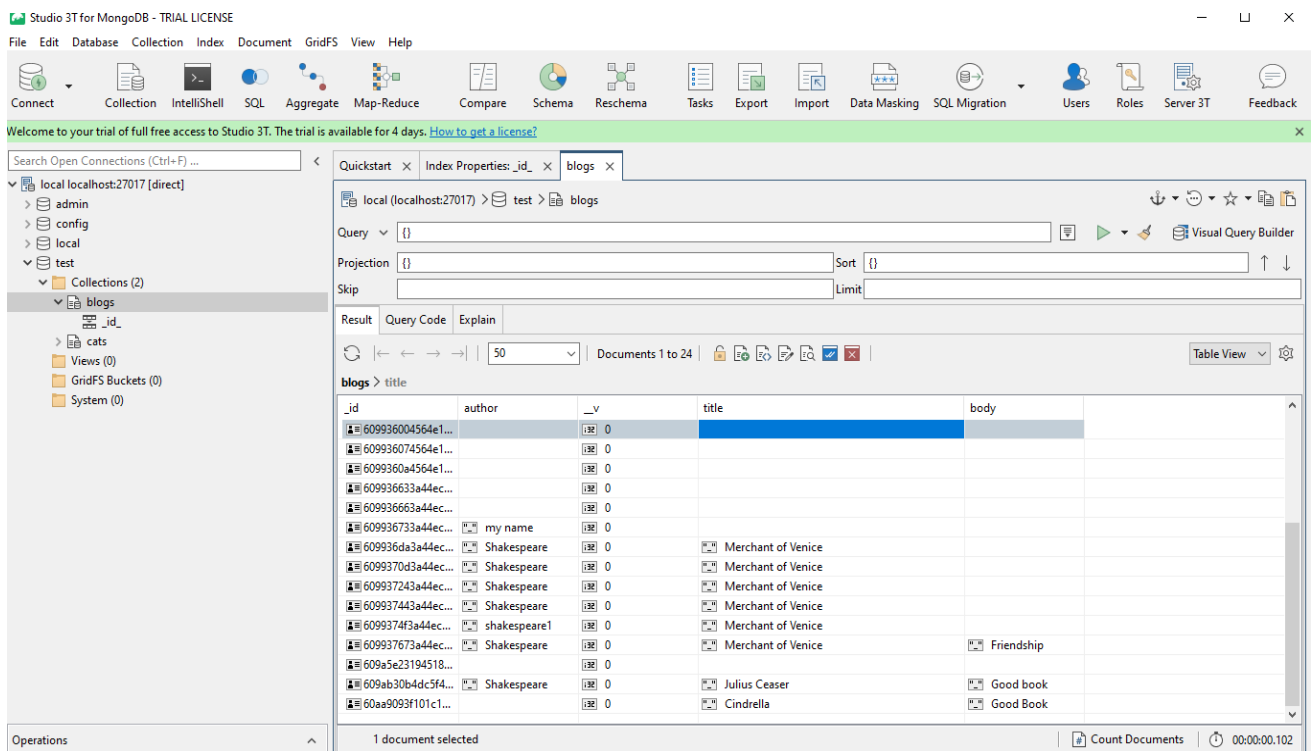
E:\>cd demoFORRESTAPI

E:\DemoForRestAPI>node index7.js
(node:6036) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discovery and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
(Use `node --trace-warnings ...` to show where the warning was created)
Server is running on PORT: ${3000}
```

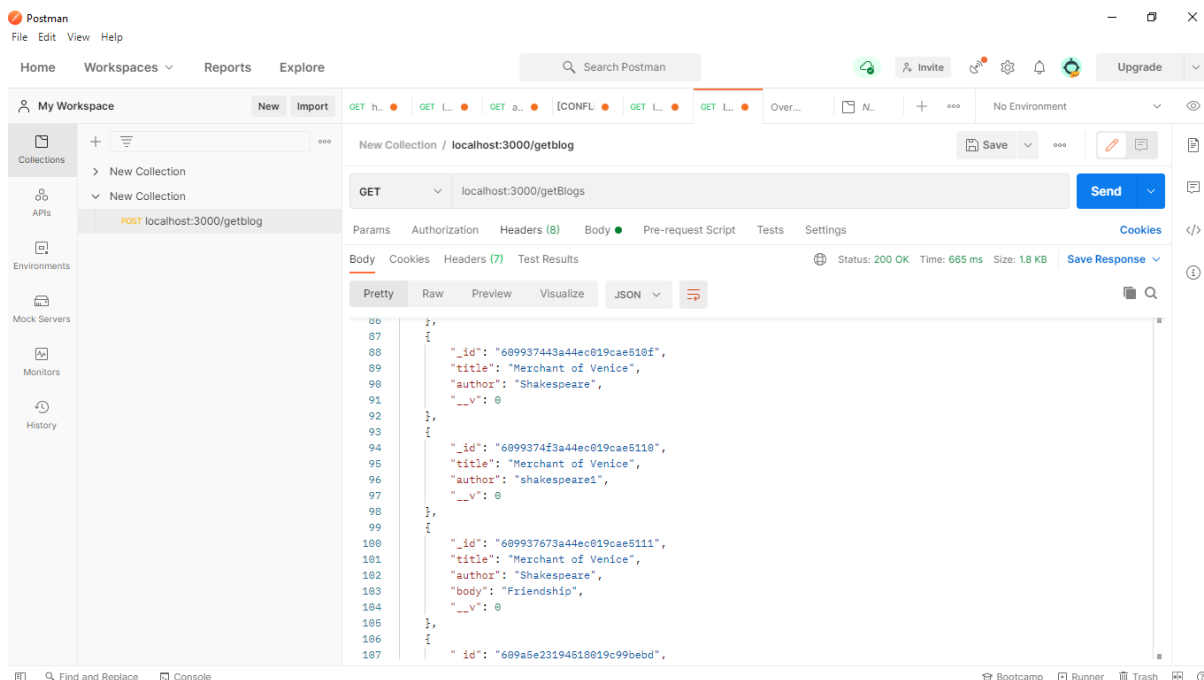
STEP 4: Start MONGO DB 3T Click Localhost:27017 to start localhost where data can be fetched.



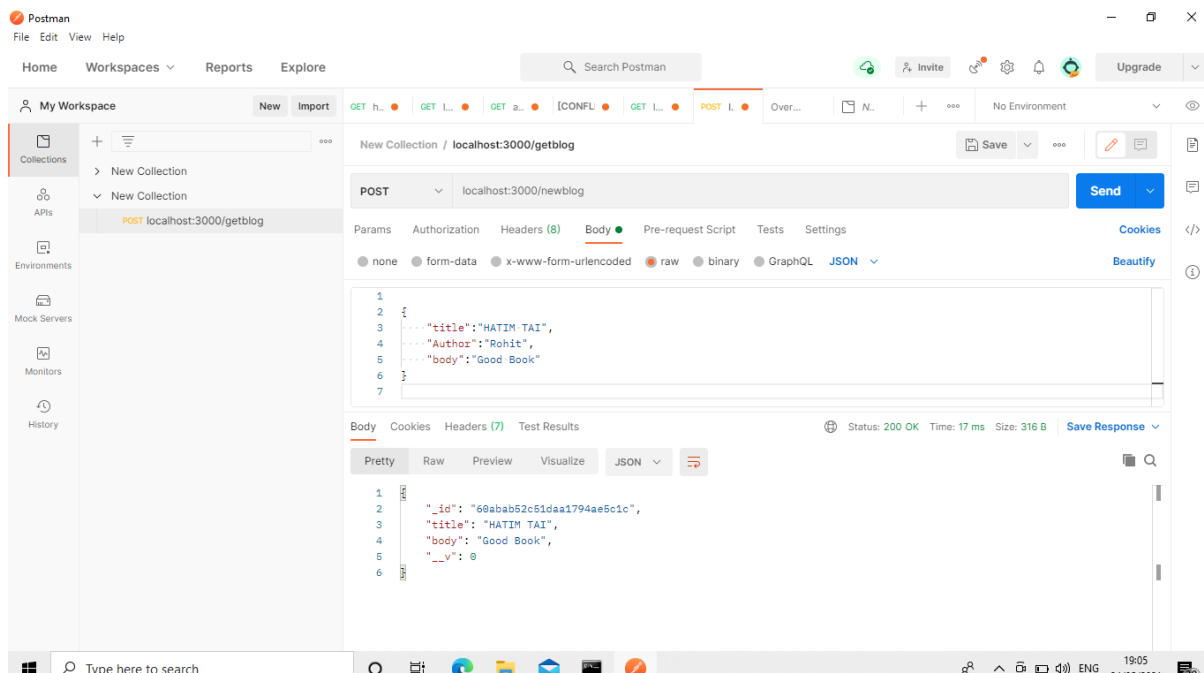
STEP 5: check whether all files available inside localhost package.



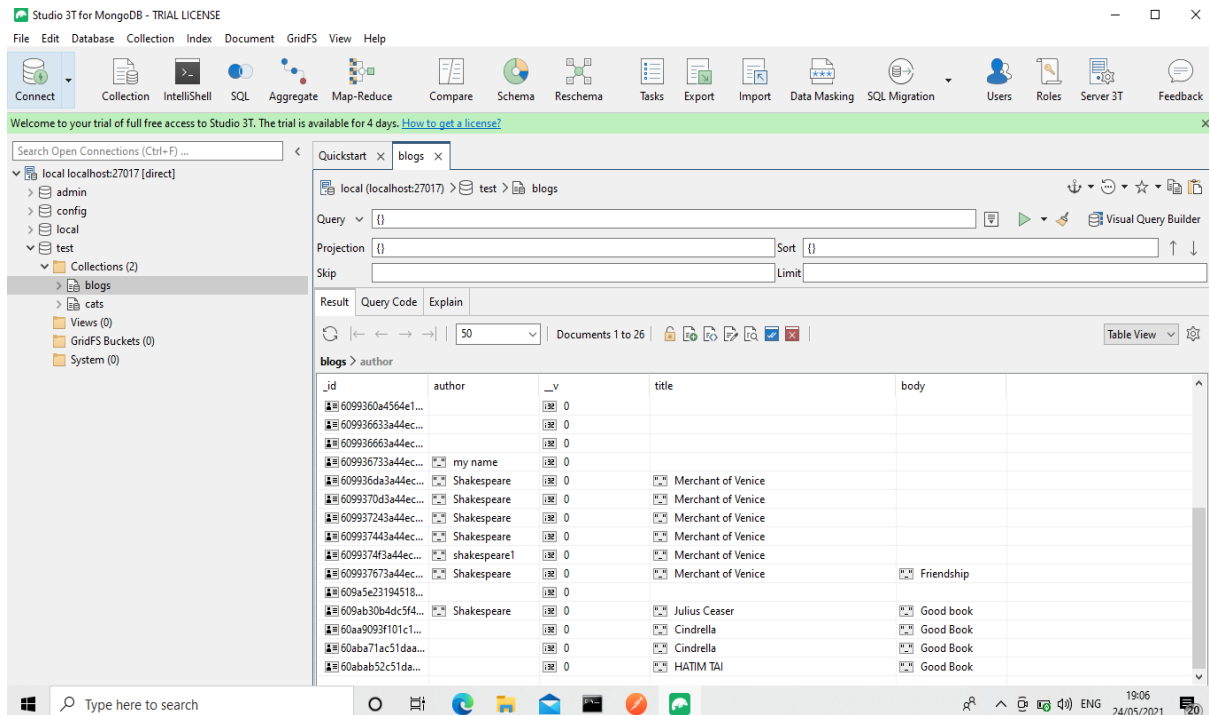
STEP 6:-then put internet on run postman and call get method all the data created in an database shown on console of postman of MongoDB.



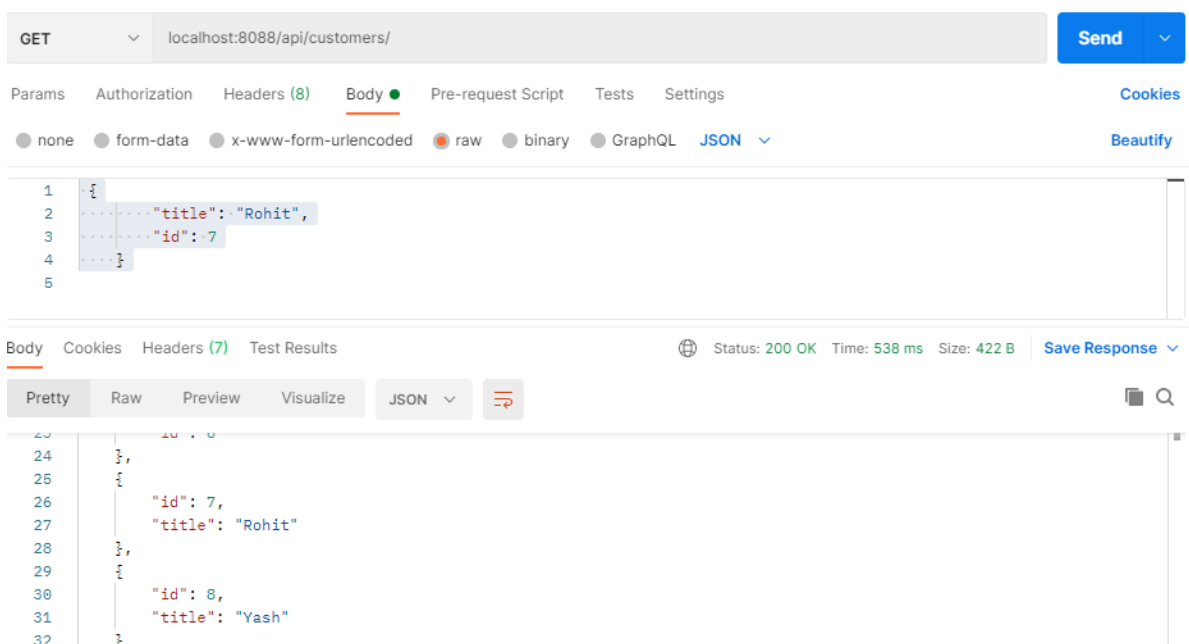
STEP 7:-then used post method in postman to create data in an database(Mongodb).



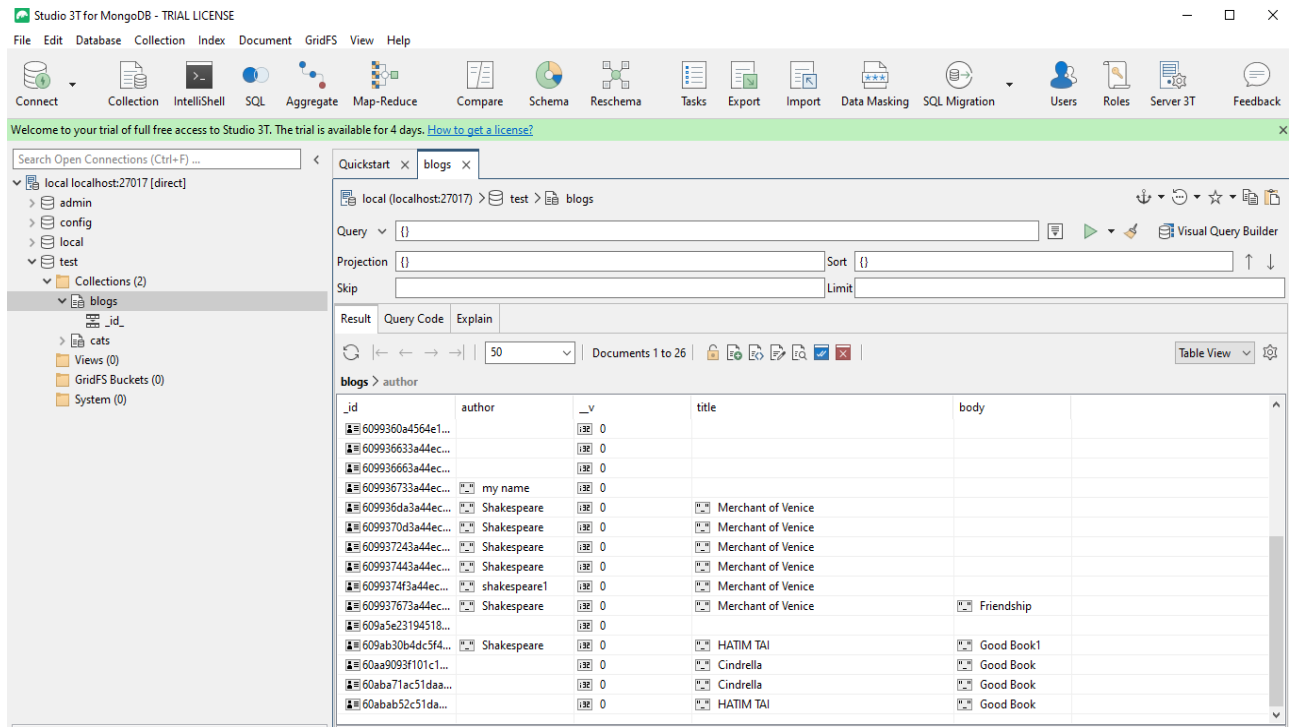
Step 8:-You can see that Mongoddb database data that has been created in an Mongoddb database.



STEP 9:-Now we will used put method to update database we will used put method in a postman to update the created method.



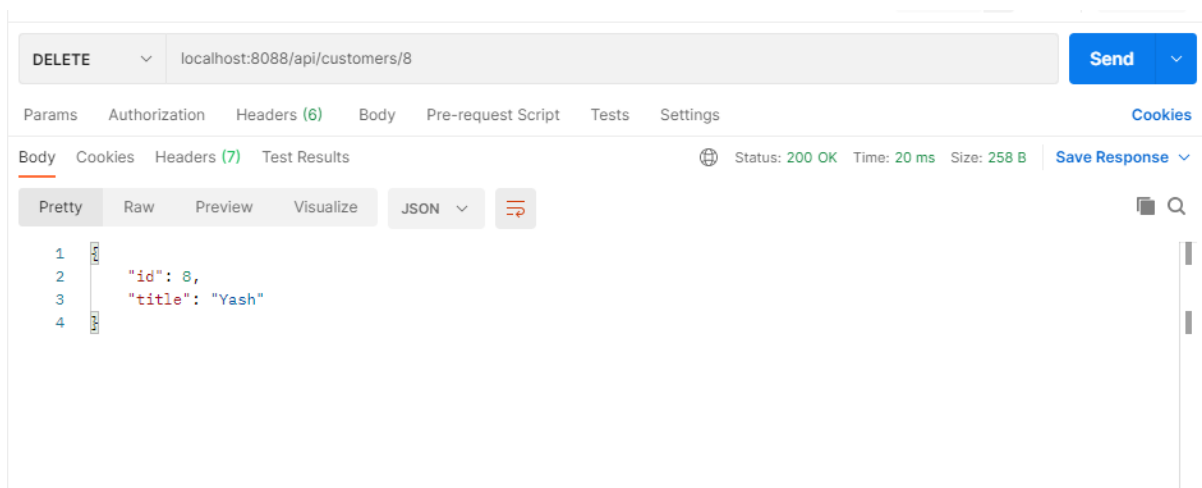
STEP 10:As you can see PuT METHOD it is reflected in a MongoDB.



The screenshot shows the Studio 3T interface for MongoDB. The left sidebar displays the database structure, including the 'test' database and the 'blogs' collection. The main window shows the 'blogs' collection with a query result table. The table has columns: _id, author, _v, title, and body. The data is as follows:

_id	author	_v	title	body
6099360a4564e1...		0		
609936633a44ec...		0		
609936663a44ec...		0		
609936733a44ec...	my name	0		
609936da3a44ec...	Shakespeare	0	Merchant of Venice	
6099370d3a44ec...	Shakespeare	0	Merchant of Venice	
609937243a44ec...	Shakespeare	0	Merchant of Venice	
609937443a44ec...	Shakespeare	0	Merchant of Venice	
6099374f3a44ec...	shakespeare1	0	Merchant of Venice	
609937673a44ec...	Shakespeare	0	Merchant of Venice	Friendship
609a5e23194518...		0		
609ab30b4dc5f4...	Shakespeare	0	HATIM TAI	Good Book1
60aa903f101c1...		0	Cindrella	Good Book
60aba71ac51daa...		0	Cindrella	Good Book
60abab52c51da...		0	HATIM TAI	Good Book

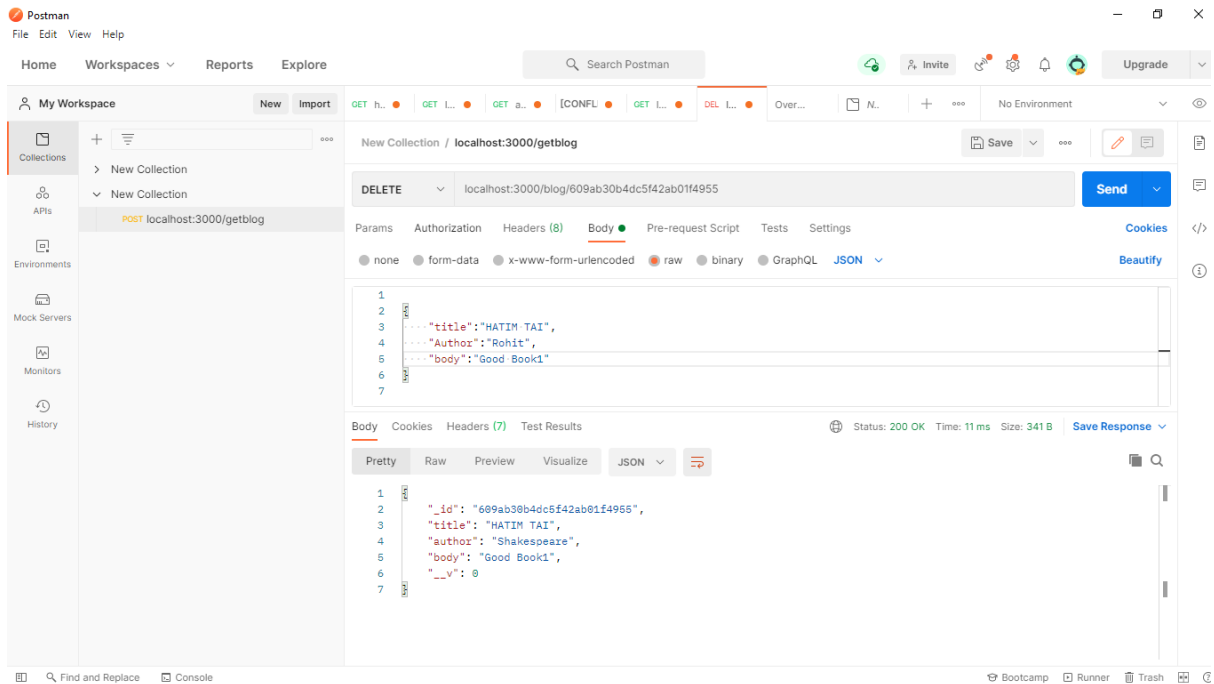
STEP 11:-DELETE METHOD USED FOR DELETING PARTICULAR ID.



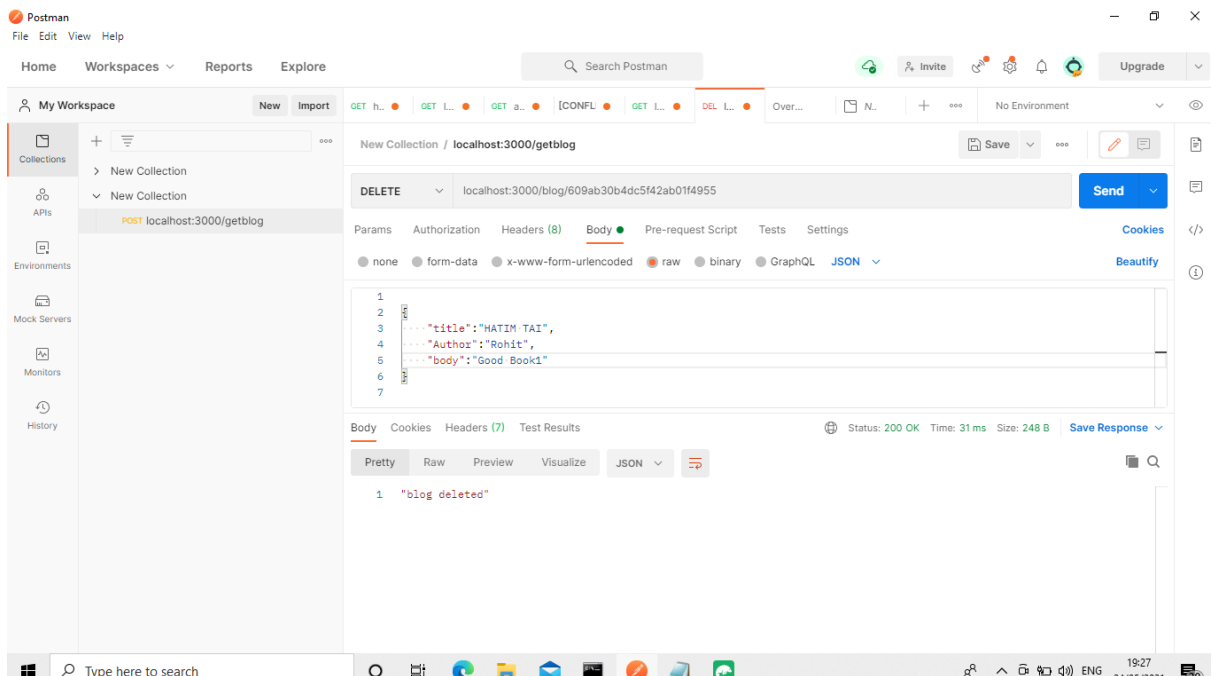
The screenshot shows a REST client interface with a DELETE request to 'localhost:8088/api/customers/8'. The response is a 200 OK status with a body containing the following JSON:

```
1 {
2   "id": 8,
3   "title": "Yash"
4 }
```

STEP 12:-It get deleted from an particular POST.



STEP 13:-it updated in MongoDB.



STEP 14:-it updated on delete Method.

The screenshot displays the Postman application interface. At the top, a collection named 'New Collection / localhost:3000/getblog' is selected. The request method is set to 'GET' and the URL is 'localhost:8088/api/customers'. The 'Send' button is visible. Below the request bar, the 'Body' tab is active, showing a JSON response. The status is '200 OK', the time is '8 ms', and the size is '398 B'. The JSON response is formatted in 'Pretty' mode and shows a list of customer objects.

```
10  {
11    "title": "Tyler",
12    "id": 3
13  },
14  {
15    "title": "Alice",
16    "id": 4
17  },
18  {
19    "title": "Candice",
20    "id": 5
21  },
22  {
23    "id": 6
24  },
25  {
26    "id": 7,
27    "title": "Rohit"
28  }
```