

# REPORT

## Overview

- In this project we can create a trader, read a trader, update a trader and we can delete a trader. To perform all these operations I have created REST API's. for sending and receiving API responses we can use JSON format.

### 1. PostMapping:

POST method is used to handle post type request methods.  
In this project to add traders details we use post method to save traders details in database.

### 2. GetMapping:

GET method is used to handle get type request methods.  
to get all traders and a specific trader by id we can use get method to access from database.

### 3. DeleteMapping:

we would delete the particular order from the database to delete a particular trader we can use delete mapping method.

### 4. PutMapping:

PUT method is used to modify/update a resources.  
to update a trader details we use put method to change the details of a trader from database.

## Software requirements

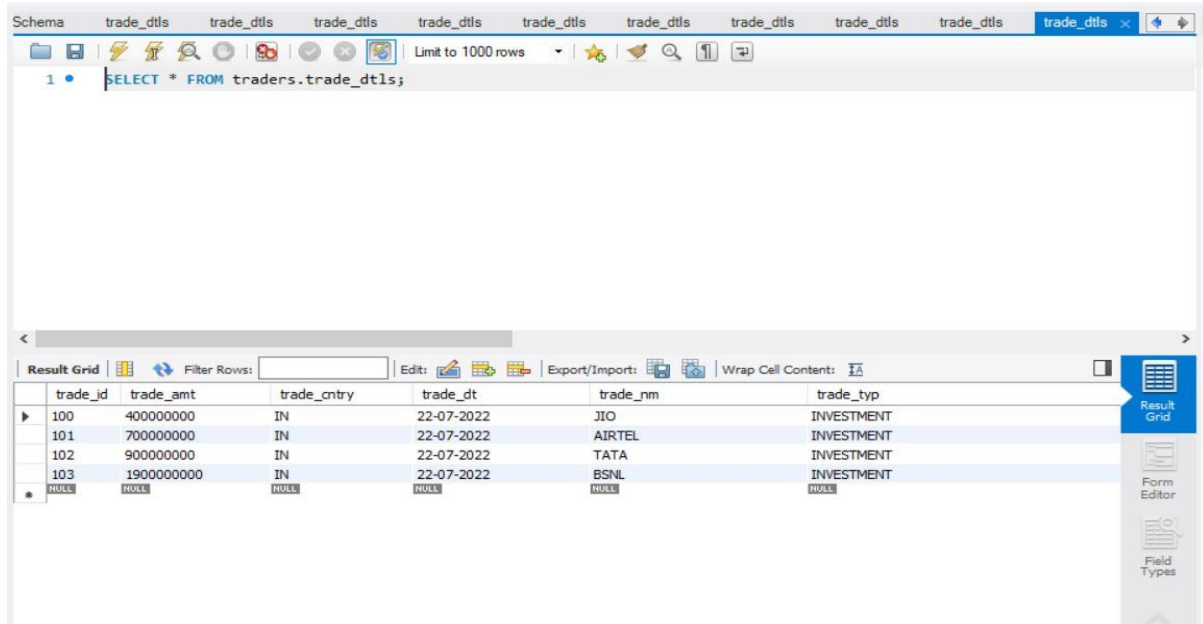
**Programming language:** Java.

**Database:** MySql.

**Tools:** postman, Spring tool suite.

# Screenshots

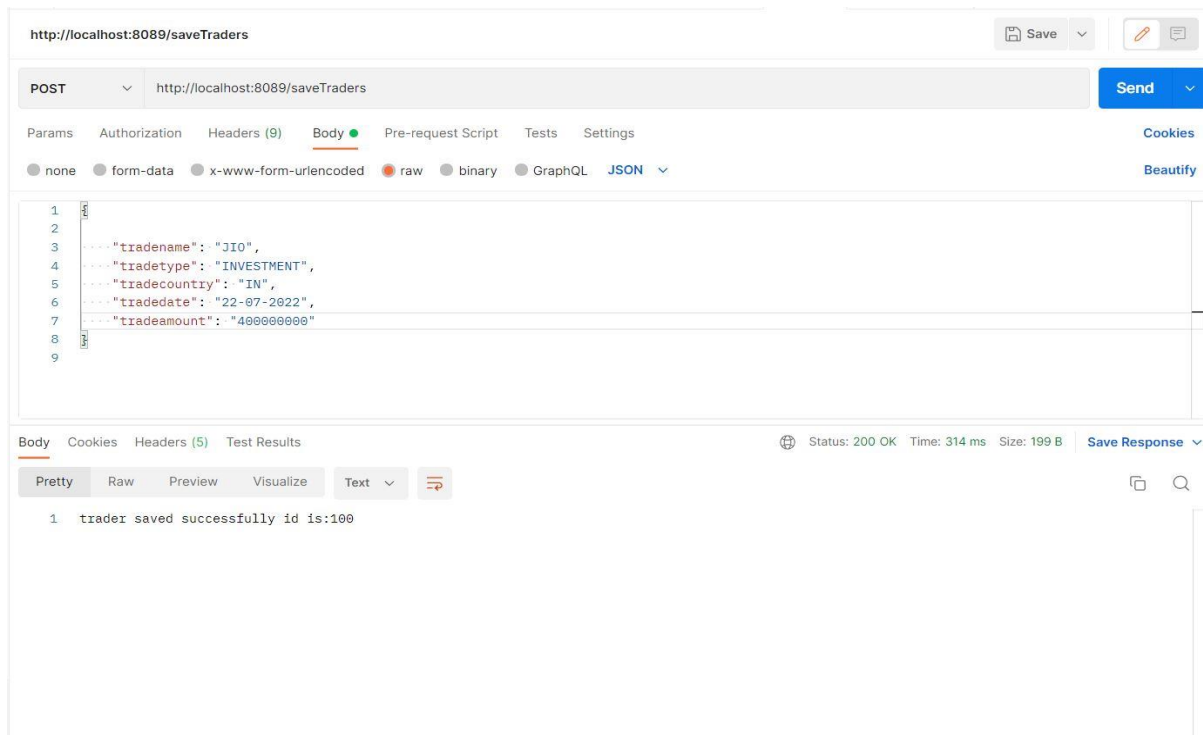
## Database:



The screenshot shows a database management interface with a query editor at the top and a result grid below. The query is `SELECT * FROM traders.trade_dtls;`. The result grid displays the following data:

trade_id	trade_amt	trade_cntry	trade_dt	trade_nm	trade_tpy
100	400000000	IN	22-07-2022	JIO	INVESTMENT
101	700000000	IN	22-07-2022	AIRTEL	INVESTMENT
102	900000000	IN	22-07-2022	TATA	INVESTMENT
103	1900000000	IN	22-07-2022	BSNL	INVESTMENT
NULL	NULL	NULL	NULL	NULL	NULL

## Add traders:



The screenshot shows a REST client interface with a POST request to `http://localhost:8089/saveTraders`. The request body is a JSON object:

```
{  "tradenam": "JIO",  "tradetyp": "INVESTMENT",  "tradecoun": "IN",  "tradedate": "22-07-2022",  "tradeamount": "400000000"}
```

The response status is 200 OK, and the response body is:

```
1 trader saved successfully id is:100
```

## Delete traders:

http://localhost:8089/deletetrade/103

DELETE http://localhost:8089/deletetrade/103

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "tradeid": 103,
3   "tradenname": "VODAFONE",
4   "tradetype": "INVESTMENT",
5   "tradecountry": "IN",
6   "tradedate": "22-07-2022",
7   "tradeamount": "100000000"
8 }
```

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 37 ms Size: 194 B Save Response

Pretty Raw Preview Visualize Text

```
1 trader deleted successfully!!!
```

## Get traders by id:

http://localhost:8089/getTradersbyid/101

GET http://localhost:8089/getTradersbyid/101

Params Authorization Headers (7) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "tradeid": 101,
3   "tradenname": "AIRTEL",
4   "tradetype": "INVESTMENT",
5   "tradecountry": "IN",
6   "tradedate": "22-07-2022",
7   "tradeamount": "700000000"
8 }
```

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 17 ms Size: 296 B Save Response

Pretty Raw Preview Visualize JSON

## Update traders:

http://localhost:8089/updateTrader

PUT http://localhost:8089/updateTrader

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1 2 3 4 5 6 7 8

```
1 {
2   "tradeid": 103,
3   "tradenname": "VODAFONE",
4   "tradetype": "INVESTMENT",
5   "tradecountry": "IN",
6   "tradedate": "22-07-2022",
7   "tradeamount": "1000000000"
8 }
```

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 41 ms Size: 298 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "tradeid": 103,
3   "tradenname": "VODAFONE",
4   "tradetype": "INVESTMENT",
5   "tradecountry": "IN",
6   "tradedate": "22-07-2022",
7   "tradeamount": "1000000000"
8 }
```

## Get all traders:

http://localhost:8089/getallTraders

GET http://localhost:8089/getallTraders

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 156 ms Size: 691 B Save Response

Pretty Raw Preview Visualize JSON

```
2 {
3   "tradeid": 100,
4   "tradenname": "JIO",
5   "tradetype": "INVESTMENT",
6   "tradecountry": "IN",
7   "tradedate": "22-07-2022",
8   "tradeamount": "4000000000"
9 },
10 {
11   "tradeid": 101,
12   "tradenname": "AIRTEL",
13   "tradetype": "INVESTMENT",
14   "tradecountry": "IN",
15   "tradedate": "22-07-2022",
16   "tradeamount": "7000000000"
17 },
18 {
19   "tradeid": 102,
20   "tradenname": "TATA",
21   "tradetype": "INVESTMENT",
22   "tradecountry": "IN",
23   "tradedate": "22-07-2022",
24   "tradeamount": "9000000000"
25 },
26 {
27   "tradeid": 103,
28   "tradenname": "BSNL",
29   "tradetype": "INVESTMENT",
30   "tradecountry": "IN",
```

## Code:

```
Trade_DtIsCo... Trade_DtIs.java SpringbootT... Trade_DtIsRe... UserReposito... SpringbootTp...
1 package com.Traders.Controller;
2
3
4 import java.util.List;
5
6
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.web.bind.annotation.DeleteMapping;
9 import org.springframework.web.bind.annotation.GetMapping;
10 import org.springframework.web.bind.annotation.PathVariable;
11 import org.springframework.web.bind.annotation.PostMapping;
12 import org.springframework.web.bind.annotation.PutMapping;
13 import org.springframework.web.bind.annotation.RequestBody;
14 import org.springframework.web.bind.annotation.RequestParam;
15 import org.springframework.web.bind.annotation.RestController;
16
17 import com.Traders.Repository.TradeService;
18
19 import com.Traders.model.Trade_DtIs;
20
21
22
23 @RestController
24 public class Trade_DtIsController
25 {
26     @Autowired
27     TradeService service;
28
29
30     //get traders by id
31     @GetMapping("/getTradersbyid/{id}")
32     private Trade_DtIs gettraders(@PathVariable ("id") int tradeid)
33     {
34         return service.gettradeById(tradeid);
35     }
36 }
```

```
Trade_DtIsCo... Trade_DtIs.java SpringbootT... Trade_DtIsRe... UserReposito... SpringbootTp... application...
37 //save traders
38 @PostMapping("/saveTraders")
39 private String savetraders(@RequestBody Trade_DtIs traders)
40 {
41     service.saveOrUpdate(traders);
42     return "trader saved successfully id is:"+traders.getTradeid();
43 }
44
45
46 //delete trader by id
47 @DeleteMapping("/deletetrade/{id}")
48 private String deletetrade(@PathVariable ("id") int tradeid)
49 {
50
51     service.delete(tradeid);
52     return "trader deleted successfully!!!";
53 }
54
55
56 //get all traders
57 @GetMapping("/getAllTraders")
58 private List<Trade_DtIs> getAllTraders()
59 {
60     return service.getAlltraders();
61 }
62
63
64 //update trader
65 @PutMapping("/updateTrader")
66 private Trade_DtIs update(@RequestBody Trade_DtIs Traders)
67 {
68     service.saveOrUpdate(Traders);
69     return Traders;
70 }
71
72 }
```

```
10 @Service
11 public class TradeService {
12
13     @Autowired
14     Trade_DtIsRepo traderepo;
15
16
17
18     public List<Trade_DtIs> getAlltraders()
19     {
20         List<Trade_DtIs> traders = new ArrayList<Trade_DtIs>();
21         traderepo.findAll().forEach(trade1 -> traders.add(trade1));
22         return traders;
23     }
24
25
26
27     public Trade_DtIs gettradeById(int id)
28     {
29         return traderepo.findById(id).get();
30     }
31
32
33     public void delete(int id)
34     {
35         traderepo.deleteById(id);
36     }
37
38
39     public void saveOrUpdate(Trade_DtIs traders)
40     {
41         traderepo.save(traders);
42     }
43
44
45     public void update(Trade_DtIs traders, int tradeid)
46     {
47         traderepo.save(traders);
48     }
49
50 }
```