

REST

Representational State Transfer

REST is an architectural style that defines a set of constraints to be used for creating web services.

Documentation

Search the docs



Getting started

Fundamentals ▾

Tools and libraries

Tutorials

API reference index

Twitter API ▾

Twitter Ads API ▾

Twitter for Webhooks

Search tweets

- [GET /2/tweets/search/all](#)
- [GET /2/tweets/search/recent](#)

Timelines

- [GET /2/users/:id/mentions](#)
- [GET /2/users/:id/timelines/reverse_chronological](#)
- [GET /2/users/:id/tweets](#)

Tweet counts

- [GET /2/tweets/counts/all](#)
- [GET /2/tweets/counts/recent](#)

Tweets lookup

- [GET /2/tweets](#)
- [GET /2/tweets/:id](#)

Volume streams

- [GET /2/tweets/sample/stream](#)
- [GET /2/tweets/sample10/stream](#)
- [GET /labs/1/tweets/stream/compliance](#)
- [GET /labs/1/tweets/stream/covid19](#)

method above applies to most other back end frameworks.



Make team documentation a breeze.

Get Stack Overflow where you work.

Get it

Use nouns instead of verbs in endpoint paths

We shouldn't use verbs in our endpoint paths. Instead, we should use the nouns which represent the endpoint that we're retrieving or manipulating as the pathname.

This is because our HTTP request method already has the verb. Having verbs in our API endpoints makes it unnecessarily long since it doesn't convey any new information. The chosen verb is often at the developer's whim. For instance, some like 'get' and some like 'retrieve', so it's just better to let the HTTP method tell us what an endpoint does.

The action should be indicated by the HTTP request method that we're making. The most common methods are GET, POST, PUT, and DELETE.

- GET retrieves resources.
- POST submits new data to the server.
- PUT updates existing data.

CRUD Operations

GET retrieves resources.

POST submits new data to the server

PUT updates existing data

PATCH update existing data partially

DELETE removes data

Quora posts

post → username ✓
content ✓

- ↓
- 1) VIEW
 - 2) individual
 - 3) Edit
 - 4) Delete
- }

resource
↓
CRUD

Creating RESTful APIs

GET	/posts	to get data for all posts	<u>INDEX</u> (main)
POST	/posts	to add a new post	CREATE
GET	/posts/:id	to get one post (using id)	VIEW
PATCH	/posts/:id	to update specific post	UPDATE
DELETE	/posts/:id	to delete specific post	DESTROY

EXPLORER

...

JS index.js

✕

REST_CLASS

> node_modules

> public

> views

JS index.js


{ } package-lock.json

{ } package.json

JS index.js > ...

```
1  const express = require("express");
2  const app = express();
3  const port = 8080;
4  const path = require("path");
5  ⚡
6  app.use(express.urlencoded({ extended: true }));
7
8  app.set("view engine", "ejs");
9  app.set("views", path.join(__dirname, "views"));
10
11 app.set(express.static(path.join(__dirname, "public")));
12
13 app.listen(port, () => {
14   console.log("listening to port : 8080");
15 });
```


JS index.js ×

JS index.js >  app.get("/") callback

```
5
6   app.use(express.urlencoded({ extended: true }));
7
8   app.set("view engine", "ejs");
9   app.set("views", path.join(__dirname, "views"));
10
11  app.set(express.static(path.join(__dirname, "public")));
12
13  app.get("/", (req, res) => {
14    res.send("serving working well!");
15  });
16
17  app.listen(port, () => {
18    console.log("listening to port : 8080");
19  });
20
```

Implement : GET /posts

Index Route

GET

/posts

to get data for all posts

```
let posts = [  
  {  
    username: "apnacollege",  
    content: "I love coding!",  
  },  
  {  
    username: "shradhakhapra",  
    content: "Hard work is important to achieve success",  
  },  
  {  
    username: "rahulkumar",  
    content: "I got selected for my 1st internship!",  
  },  
];
```

EXPLORER

...

JS index.js

<> index.ejs

REST_CLASS

node_modules

public

views

index.ejs

index.js

package-lock.json

package.json

views > <> index.ejs > html > body

1 <!DOCTYPE html>

2 <html lang="en">

3 <head>

4 <meta charset="UTF-8" />

5 <meta name="viewport" content="width=device-width, initial-scale=1.0" />

6 <title>All Posts</title>

7 </head>

8 <body>

9 <h1>Quora Posts</h1>

10

11 </body>

12 </html>

13

JS index.js > [e] posts

```
11 app.set('express.static', path.join(__dirname, 'public'));
12
13 let posts = [
14   {
15     username: "apnacollege",
16     content: "I love coding!",
17   },
18   {
19     username: "shradhakhapra",
20     content: "Hard work is important to achieve success",
21   },
22   {
23     username: "rahulkumar",
24     content: "I got selected for my 1st internship!",
25   },
26 ];
27
28 app.get("/posts", (req, res) => {
29   res.render("index.ejs", { posts });
30 });
```

EXPLORER

REST_CLASS

- node_modules
- public
- views
 - index.ejs

JS index.js

- package-lock.json
- package.json

JS index.js

index.ejs





views > index.ejs > html > body > ? > div > h4 > ?

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>All Posts</title>
7   </head>
8   <body>
9     <h1>Quora Posts</h1>
10    <% for(post of posts) { %>
11      <div>
12        <h3>@ <%= post.username %></h3>
13        <h4><%= post.content %></h4>
14      </div>
15    <% } %>
16  </body>
17 </html>
18
```

 All Posts

×

+

    localhost:8080/posts

Quora Posts

@ apnacollege

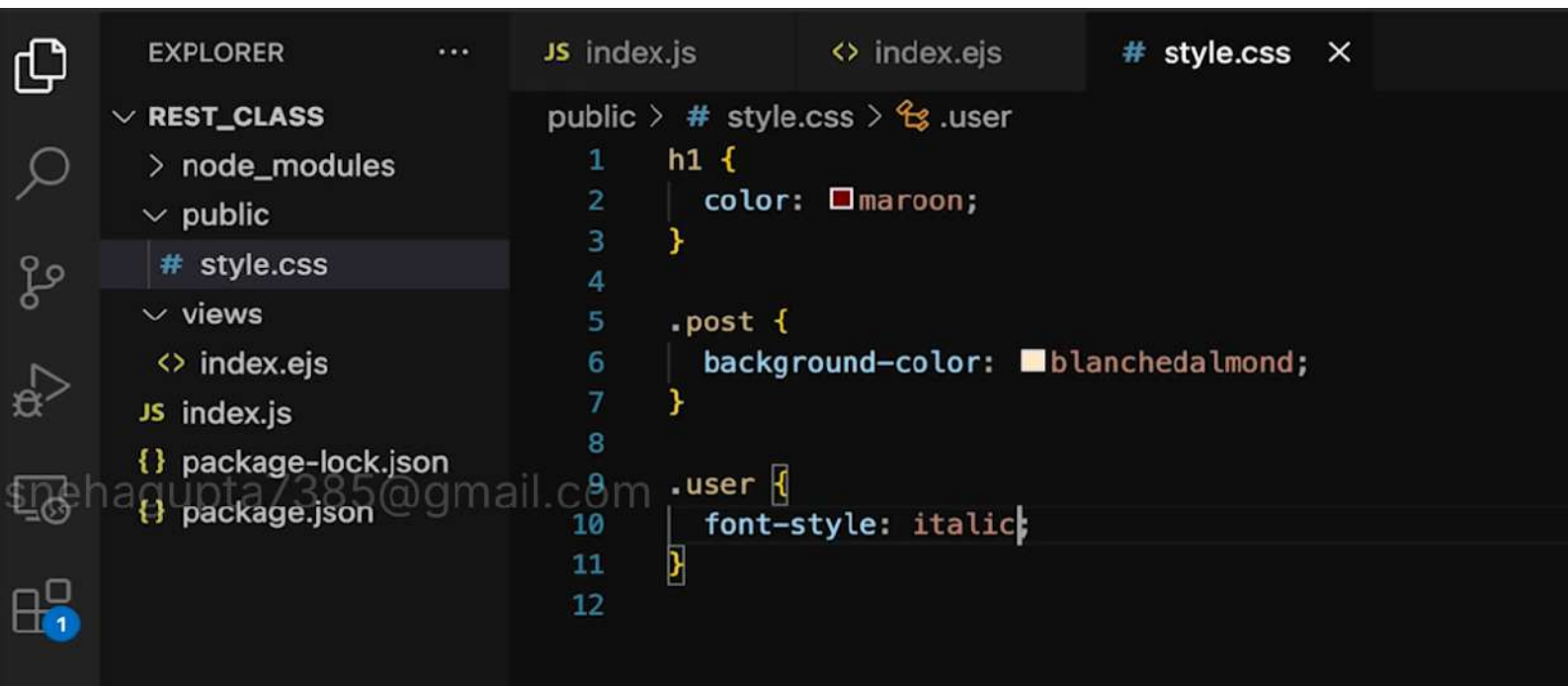
I love coding!

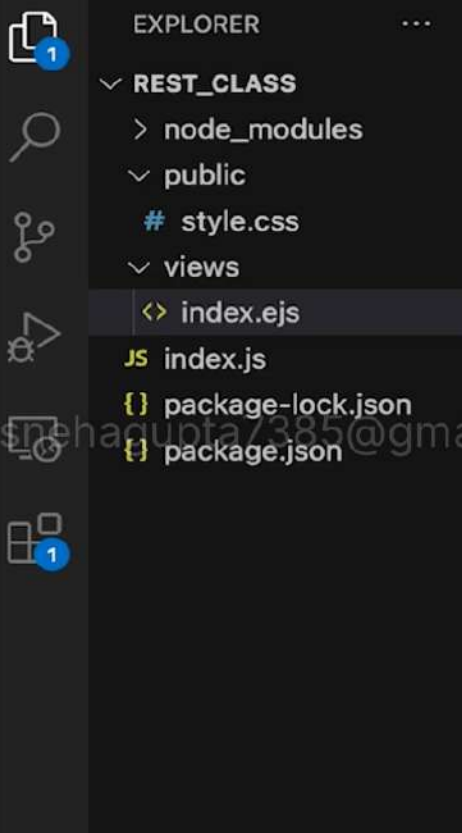
@ shradhakhapra

Hard work is important to achieve success

@ rahulkumar

I got selected for my 1st internship!





EXPLORER

- REST_CLASS
 - node_modules
 - public
 - style.css
 - views
 - index.ejs
- index.js
- package-lock.json
- package.json

views > index.ejs > html > head > link

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>All Posts</title>
7     <link rel="stylesheet" href="/style.css" />
8   </head>
9   <body>
10    <h1>Quora Posts</h1>
11    <% for(post of posts) { %>
12      <div class="post">
13        <h3 class="user">@<%= post.username %></h3>
14        <h4 class="content"><%= post.content %></h4>
15      </div>
16    <% } %>
17  </body>
18 </html>
19
```

Quora Posts

@apnacollege

I love coding!

@shradhakhapra

Hard work is important to achieve success

@rahulkumar

I got selected for my 1st internship!

Implement : **POST /posts**

Create Route

POST **/posts** to add a new post

2 routes

- **Serve the form** **GET** **/posts/new**
- **Add the new post** **POST** **/posts**

1) (user, content) - form

2) POST new post add to list

EXPLORER

...

JS index.js

<> new.ejs

REST_CLASS

> node_modules

> public

views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

views > <> new.ejs > html > body > form > textarea

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Create New Post</title>
  </head>
  <body>
    <form>
      <input placeholder="enter username" name="username" /> <br /><br />
      <textarea placeholder="write your post" name="content"> </textarea>
      <button>Submit Post</button>
    </form>
  </body>
</html>
```

EXPLORER

...

JS index.js

<> new.ejs

✓ REST_CLASS

> node_modules

> public

✓ views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js > ...

```
25 | },  
26 |  
27 |  
28 | app.get("/posts", (req, res) => {  
29 |   res.render("index.ejs", { posts });  
30 | });  
31 |  
32 | app.get("/posts/new", (req, res) => {  
33 |   res.render("new.ejs");  
34 | });  
35 |
```

EXPLORER

...

JS index.js

<> new.ejs

X

REST_CLASS

> node_modules

> public

views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

views > <> new.ejs > html > body > form

1 <!DOCTYPE html>

2 <html lang="en">

3 <head>

4 <meta charset="UTF-8" />

5 <meta name="viewport" content="width=device-width, initial-scale=1.0" />

6 <title>Create New Post</title>

7 </head>

8 <body>

9 <form method="post" action="/posts">

10 <input placeholder="enter username" name="username" />

11 <textarea placeholder="write your post" name="content"> </textarea>

12 <button>Submit Post</button>

13 </form>

All Posts



Create New Post



sne



localhost:8080/posts/new?username=apnacollege&content=+ab

apnacollege

hello world!

Submit Post

All Posts



localhost:8080/posts



localhost:8080/posts

post request working

EXPLORER

REST_CLASS

- node_modules
- public
- views
 - index.ejs
 - new.ejs

JS index.js

- package-lock.json
- package.json

JS index.js

JS index.js > ...

```
27
28 app.get("/posts", (req, res) => {
29   res.render("index.ejs", { posts });
30 });
31
32 app.get("/posts/new", (req, res) => {
33   res.render("new.ejs");
34 });
35
36 app.post("/posts", (req, res) => {
37   console.log(req.body);
38   res.send("post request working");
39 });
```

EXPLORER

...

JS index.js

X

<> new.ejs

✓ REST_CLASS

> node_modules

> public

✓ views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js > ...

```
27
28 app.get("/posts", (req, res) => {
29   res.render("index.ejs", { posts });
30 });
31
32 app.get("/posts/new", (req, res) => {
33   res.render("new.ejs");
34 });
35
36 app.post("/posts", (req, res) => {
37   let { username, content } = req.body;
38   posts.push({ username, content });
39   res.send("post request working");
40 });
41
42 app.listen(port, () => {
43   console.log("listening to port : 8080");
44 });
```

All Posts



localhost:8080/posts



localhost:8080/posts

Quora Posts

@apnacollege

I love coding!

@shradhakhapra

Hard work is important to achieve success

@rahulkumar

I got selected for my 1st internship!

@apnacollege

Hello World!

Redirect

`res.redirect(URL)`

`res.send ()` → text
html
object
`res.render` → ejs

EXPLORER

...

JS index.js

X

<> new.ejs

REST_CLASS

> node_modules

> public

views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js > app.post("/posts") callback

```
26  };
27
28  app.get("/posts", (req, res) => {
29    |   res.render("index.ejs", { posts });
30  | });
31
32  app.get("/posts/new", (req, res) => {
33    |   res.render("new.ejs");
34  | });
35
36  app.post("/posts", (req, res) => {
37    |   let { username, content } = req.body;
38    |   posts.push({ username, content });
39    |   res.redirect("/posts");
40  | });
41
42  app.listen(port, () => {
43    |   console.log("listening to port : 8080");
44  | });
45
```

All Posts



Create New Post



localhost:8080/posts/new

apnacollege

Hello World!

Submit Post

All Posts

×

All Posts

×

+

localhost:8080/posts

Quora Posts

@apnacollege

I love coding!

@shradhakhapra

Hard work is important to achieve success

@rahulkumar

I got selected for my 1st internship!

@apnacollege

Hello World!

EXPLORER

...

JS index.js

<> index.ejs

<> new.ejs

REST_CLASS

> node_modules

> public

> views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

views > <> index.ejs > html > body > ? > ? > a

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

```
<meta name="viewport" content="width=device-width, initial-sca
<title>All Posts</title>
<link rel="stylesheet" href="/style.css" />
</head>
<body>
  <h1>Quora Posts</h1>
  <% for(post of posts) { %>
    <div class="post">
      <h3 class="user">@<%= post.username %></h3>
      <h4 class="content"><%= post.content %></h4>
    </div>
    <%}%>
  <br />
  <br />
  <a href="http://localhost:8080/posts/new">Create New Post</a>
</body>
</html>
```

abc Post

abc Posts

Implement : GET /posts/:id

Show Route

GET /posts/:id to get one post (using id)

id → 1, 2, 3, 4
1a, 2b, 3c
"abcdef", "ghij"

EXPLORER

...

JS index.js

X

<> index.ejs

<> new.ejs

REST_CLASS

> node_modules

> public

views

<> index.ejs

<> new.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js > app.get("/posts/:id") callback

```
39 app.post("/posts", (req, res) => {
40   let { username, content } = req.body;
41   posts.push({ username, content });
42   res.redirect("/posts");
43 });
44
45 app.get("/posts/:id", (req, res) => {
46   let { id } = req.params;
47   let post = posts.find((p) => id === p.id);
48   console.log(post);
49   res.send("request working");
50 });
51
52 app.listen(port, () => {
53   console.log("listening to port : 8080");
54 });
55
```

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

```
{
  id: '2b',
  username: 'shradhakhapra',
```

← → ↻ ⓘ localhost:8080/posts/1a

request working

```
18: 20 ,  
username: 'shradhakhapra',  
content: 'Hard work is important to achieve success'  
+  
undefined I
```

EXPLORER

...

JS index.js

X

<> index.ejs

<> show.ejs

REST_CLASS

> node_modules

> public

views

<> index.ejs

<> new.ejs

<> show.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js > app.get("/posts/:id") callback

```
39 app.post("/posts", (req, res) => {
40   let { username, content } = req.body;
41   posts.push({ username, content });
42   res.redirect("/posts");
43 });
44
45 app.get("/posts/:id", (req, res) => {
46   let { id } = req.params;
47   console.log(id);
48   let post = posts.find((p) => id === p.id);
49   res.render("show.ejs", { post });
50 });
51
52 app.listen(port, () => {
53   console.log("listening to port : 8080");
54 });
55
```

🌐 All Posts



🌐 Post in Detail



🌐 localhost:8080/posts/2b


Here is your post in detail

Post id : 2b



@shradhakhapra

Hard work is important to achieve success

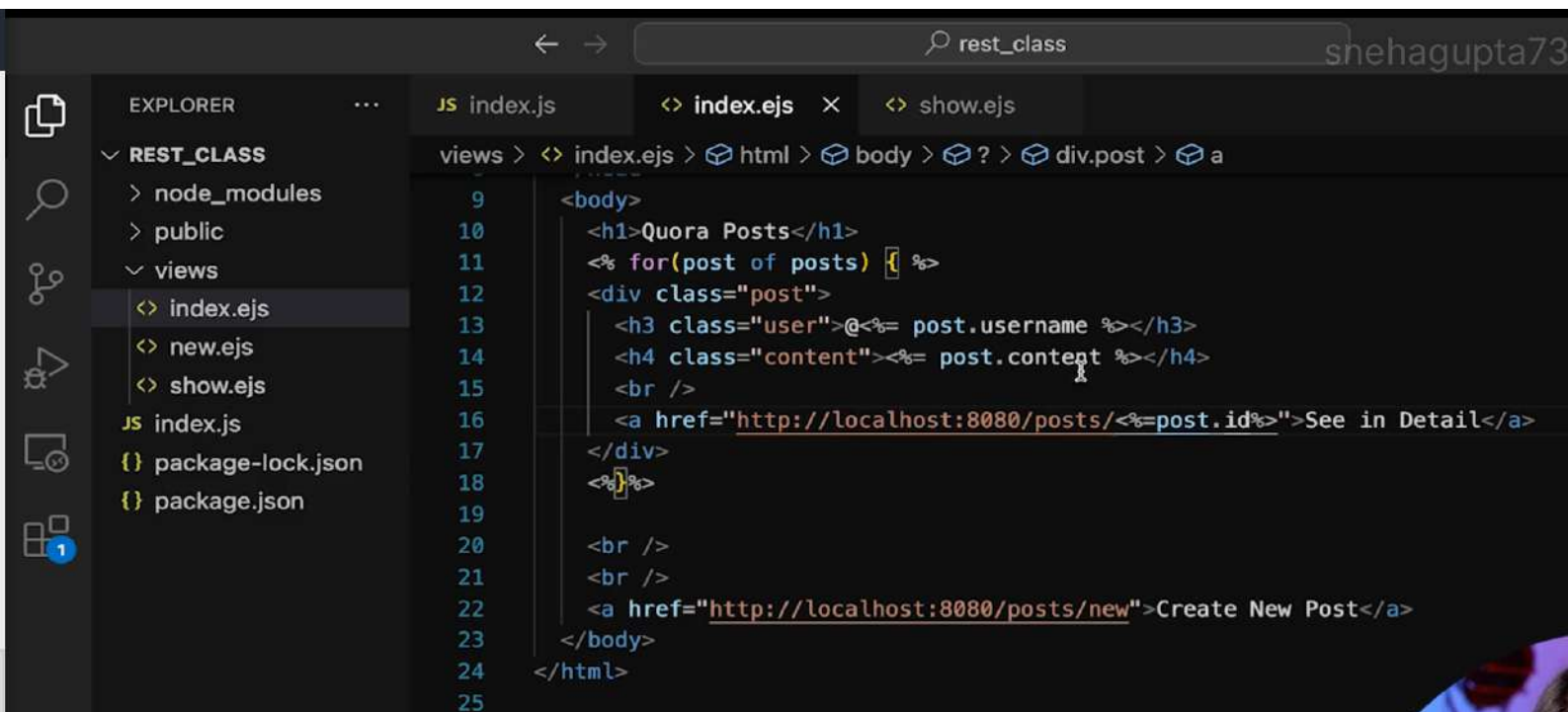


VS Code Explorer sidebar showing project structure:

- REST_CLASS
 - node_modules
 - public
 - views
 - index.ejs
 - new.ejs
 - show.ejs
- index.js
- package-lock.json
- package.json

1

```
views > <> show.ejs > html > head > link
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Post in Detail</title>
7     <link rel="stylesheet" href="/style.css" />
8   </head>
9   <body>
10    <h2>Here is your post in detail</h2>
11    <p>Post id : <%= post.id %></p>
12    <div class="post">
13      <h3 class="user">@<%= post.username %></h3>
14      <h4><%= post.content %></h4>
15    </div>
16  </body>
17 </html>
18
```

All Posts



Post in Detail



localhost:8080/posts

Quora Posts

@apnacollege

I love coding!

[See in Detail](#)

@shradhakhapra

Hard work is important to achieve success

[See in Detail](#)

@rahulkumar

I got selected for my 1st internship!

[See in Detail](#)

[Create New Post](#)

Post in Detail



Post in Detail



localhost:8080/posts/1a

Here is your post in detail

Post id : 1a

@apnacollege

I love coding!

Create id for Posts

UUID Package

Universally unique identifier

npm install uuid

Quickstart

To create a random UUID...

1. Install

```
npm install uuid
```

2. Create a UUID (ES6 module syntax)

```
import { v4 as uuidv4 } from 'uuid';  
uuidv4(); // ⇒ '9b1deb4d-3b7d-4bad-9bdd-2b0d7b3dcb6d'
```

... or using CommonJS syntax:

```
const { v4: uuidv4 } = require('uuid');  
uuidv4(); // ⇒ '1b9d6bcd-bbfd-4b2d-9b5d-ab8dfbbd4bed'
```

```
const express = require("express");
const app = express();
const port = 8080;
const path = require("path");
const {v4:uuidv4} = require('uuid');

app.use(express.urlencoded({extended:true}));
app.use(express.json());

app.set("view engine","ejs");
app.set("views",path.join(__dirname,"views"));
app.use(express.static(path.join(__dirname,"public,
app.use(express.static(path.join(__dirname,"public,
```

```
3 app.use(express.static(path.join(__dirname, "public")))
```

```
4  
5 let posts = [
```

```
6   {
```

```
7     id: uuidv4(),
```

```
8     username: "Sneha Gupta",
```

```
9     content: "I Love coding"
```

```
10  },
```

```
11  {
```

```
12    id: uuidv4(),
```

```
13    username: "Adarsh Goyal",
```

```
14    content: "Make money and have world knee d
```

```
15  },
```

```
16  {
```

```
17    id: uuidv4(),
```

```
18    username: "Anchal Goyal",
```

```
19    content: "I wanna be a doctor so that i ca
```

```
20  }
```



```
40
41 app.get("/posts/new", (req, res) => {
42   res.render("new.ejs");
43 })
44 app.post("/posts", (req, res) => {
45   let {username, content} = req.body;
46   let id = uuidv4();
47   posts.push({id, username, content});
48   res.redirect("/posts");
49 })
50 app.get("/posts/:id", (req, res) => {
51   let {id} = req.params;
52   let post = posts.find((p) => id === p.id);
53   res.render("show.ejs", {post});
54 })
```



```
47 app.get("/posts/:id", (req, res) => {  
48   let { id } = req.params;  
49   console.log(id);  
50   let post = posts.find((p) => id === p.id);  
51   res.render("show.ejs", { post });  
52 });
```



```
54 app.patch("/posts/:id", (req, res) => {  
55   let { id } = req.params;  
56   console.log(id);  
57   res.send("patch request working");  
58 });
```



PATCH



http://localhost:8080/posts/01786cd6-ee59-4908-bee8-19605bedc2f7

Parameters

Body

Headers

Authorization

Pre-request Script

Tests

Content Type

application/x-www-form-urlencoded



Override

Request Body

content

I love apna-college

Parameter 2

Value 2

Status: 200 • OK Time: 84 ms Size: 21 B

HTML

Raw

Headers

7

Test Results

Response Body

1







patch request working

```
52     });  
53  
54     app.patch("/posts/:id", (req, res) => {  
55         let { id } = req.params;  
56         💡 let newContent = req.body.content;  
57         console.log(newContent);  
58         res.send("patch request working");  
59     });
```

```
50     let post = posts.find((p) => id === p.id);
51     res.render("show.ejs", { post });
52 });
53
54 app.patch("/posts/:id", (req, res) => {
55     let { id } = req.params;
56     let newContent = req.body.content;
57     let post = posts.find((p) => id === p.id);
58     post.content = newContent;
59     console.log(post);
60     res.send("patch request working");
61 });
62
63 app.listen(port, () => {
64     console.log("listening to port : 8080");
65 });
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
at next (/Users/shradhakhapra/WebDevelopment/Backend/rest_class/
js:280:10)
b619f7e5-c07a-4077-8d1e-96b33f273730
{
  id: 'b619f7e5-c07a-4077-8d1e-96b33f273730',
  username: 'apnacollege',
  content: 'I love apna-college'
}
```



EXPLORER

REST_CLASS

> node_modules

> public

> views

<> index.ejs

<> new.ejs

<> show.ejs

JS index.js


{ } package-lock.json

{ } package.json

JS index.js

index.ejs

show.ejs

JS index.js >  app.patch("/posts/:id") callback

45 };

46

47 app.get("/posts/:id", (req, res) => {

48 let { id } = req.params;

49 console.log(id);

50 let post = posts.find((p) => id === p.id);

51 res.render("show.ejs", { post });

52 });

53

54 app.patch("/posts/:id", (req, res) => {

55 let { id } = req.params;

56 let newContent = req.body.content;

57 let post = posts.find((p) => id === p.id);

58 post.content = newContent;

59 console.log(post);

60 res.send("patch request working");

61 });


62

- REST
- GraphQL
- Realtime
- Settings

My Wor... > Collections

Search

+ New ?



Collections are empty

Add new

PATCH Untitled +

PATCH http://localhost:8080/posts/01786cd6-ee59-4908-bee8-19605be

Parameters Body Headers Authorization Pre-request Script Te

Content Type application/x-www-form-urlencoded Override

Request Body

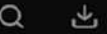
Status: 500 Internal Server Error Time: 89 ms Size: 1.53 KB

HTML Raw Headers 8 Test Results

Response Body

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="utf-8">
5 <title>Error</title>
6 </head>
7 <body>
8 <pre>TypeError: Cannot set properties of undefined (setting &#3
  /Users/shradhakhapra/WebDevelopment/Backend/rest_class/index.js
    [as handle_request]
    (/Users/shradhakhapra/WebDevelopment/Backend/rest_class/node mo
```

HOPPSCOTCH



REST

GraphQL

Realtime

Settings

My Wor... > Collections

Search

+ New ? 🗑️

Collections are empty

Add new

PATCH Untitled • +

PATCH ▾ http://localhost:8080/posts/b619f7e5-c07a-4077-8d1e-96b33f273730

Parameters Body Headers Authorization Pre-request Script Tests

Content Type application/x-www-form-urlencoded ▾ ⌂ Override

Request Body

Status: 200 • OK Time: 71 ms Size: 21 B

HTML Raw Headers 7 Test Results

Response Body

1 patch request working


```
62
63   app.listen(port, () => {
64     |   console.log("listening to port : 8080");
65   });
66
```

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

at next (/Users/shradhakhapra/WebDevelopment/Backend/re
js:280:10)

b619f7e5-c07a-4077-8d1e-96b33f273730

{

id: 'b619f7e5-c07a-4077-8d1e-96b33f273730',

username: 'apnacollege',

content: 'I love apna-college'

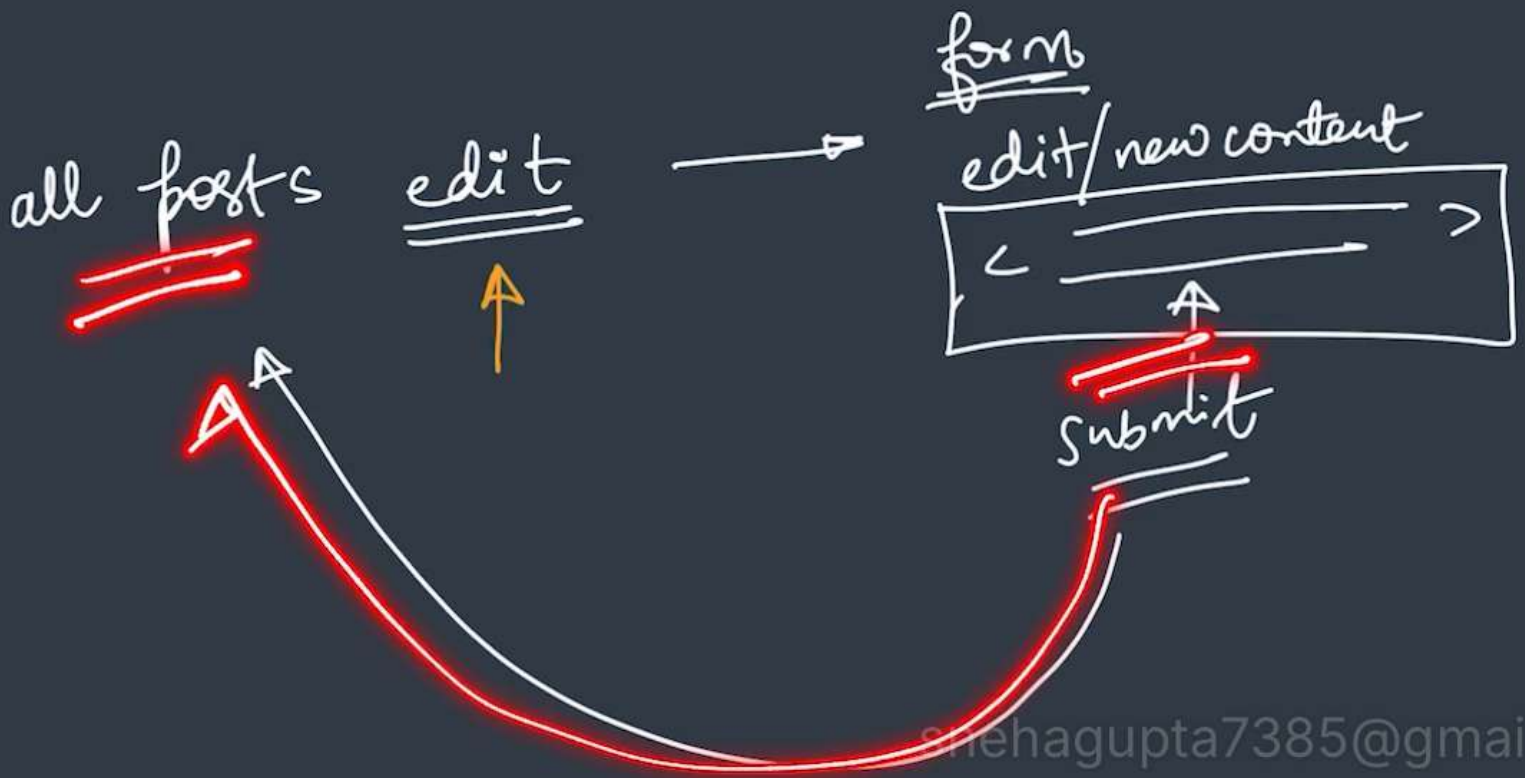
}

shenagupta7385@gmail.com

Implement : PATCH /posts/:id

Update Route

PATCH /posts/:id to update specific post



snehagupta7385@gmail

Create Form for **Update**

Edit Route

Serve the edit form

GET

`/posts/:id/edit`



Create Form for Update

Edit Route

Serve the edit form

GET

`/posts/:id/edit`

method - override

*html — get
— post*

EXPLORER

REST_CLASS

> node_modules

> public

views

<> edit.ejs

<> index.ejs

<> new.ejs

<> show.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js

<> edit.ejs

JS index.js > app.patch("/posts/:id") callback

```
50 let post = posts.find((p) => id === p.id);
51 res.render("show.ejs", { post });
52 });
```

53

54 app.patch("/posts/:id", (req, res) => {

55 let { id } = req.params;

56 let newContent = req.body.content;

57 let post = posts.find((p) => id === p.id);

58 post.content = newContent;

59 console.log(post);

60 res.send("patch request working");

61 });

62

63 app.get("/posts/:id/edit", (req, res) => {

64 let { id } = req.params;

65 let post = posts.find((p) => id === p.id);

66 res.render("edit.ejs", { post });

67 });

68

method-override DT

3.0.0 • Public • Published 5 years ago

 Readme

 Code

Beta

 4 Dependencies

 1,300+

method-override

npm v3.0.0 downloads 2.9M/month build no longer available coverage 100%

Lets you use HTTP verbs such as PUT or DELETE in places where the `client` doesn't support it.

Install

This is a **Node.js** module available through the **npm registry**. Installation is done using the **npm install** command:

```
$ npm install method-override
```

API

NOTE It is very important that this module is used **before** any module that needs to know the method of the request (for example, it *must* be used prior to the `csurf` module).

methodOverride(getter, options)

Create a new middleware function to override the `req.method` property with a new value. This

snehagupta7385@gmail.com

← →

rest_class

EXPLORER

...

JS index.js

<> edit.ejs

×

REST_CLASS

> node_modules

> public

views

<> edit.ejs

<> index.ejs

<> new.ejs

<> show.ejs

JS index.js

{ } package-lock.json

{ } package.json

views > <> edit.ejs > html > body > form > textarea > ?

1 <!DOCTYPE html>

2 <html lang="en">

3 <head>

4 <meta charset="UTF-8" />

5 <meta name="viewport" content="width=device-width, in

6 <title>Edit Post</title>

7 </head>

8 <body>

9 <h2>Edit your post</h2>

10 <p>Post id : <%= post.id %></p>

11 <p>Post username :<%= post.username %></p>

12 <form>

13 <textarea><%= post.content %></textarea>

14 <button>Submit</button>

15 </form>

16 </body>

abc content

JS index.js

<> edit.ejs

views > <> edit.ejs > html > body > form

```
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.
6     <title>Edit Post</title>
7 </head>
8 <body>
9     <h2>Edit your post</h2>
10    <p>Post id : <%= post.id %></p>
11    <p>Post username : @<%= post.username %></p>
12    <form method="post" action="/posts/<%=post.id%>?_method=PATCH">
13        <textarea rows="10" cols="35"><%= post.content %></textarea>
14        <button>Submit</button>
15    </form>
16 </body>
17 </html>
18
```


JS index.js



<> edit.ejs

JS index.js > ...

```
1  const express = require("express");
2  const app = express();
3  const port = 8080;
4  const path = require("path");
5  const { v4: uuidv4 } = require("uuid");
6  const methodOverride = require("method-override");
7
8  app.use(express.urlencoded({ extended: true }));
9  app.use(methodOverride("_method"));
10
11 app.set("view engine", "ejs");
12 app.set("views", path.join(__dirname, "views"));
13
14 app.use(express.static(path.join(__dirname, "public")));
15
16 let posts = [
17   {
18     id: uuidv4(),
19     username: "apnacollege",
```



```
55
56 app.patch("/posts/:id", (req, res) => {
57   let { id } = req.params;
58   let newContent = req.body.content;
59   let post = posts.find((p) => id === p.id);
60   post.content = newContent;
61   console.log(post);
62   res.redirect("/posts");
63 });
64
65 app.get("/posts/:id/edit", (req, res) => {
66   let { id } = req.params;
67   let post = posts.find((p) => id === p.id);
68   res.render("edit.ejs", { post });
69 });
70
71 app.listen(port, () => {
72   console.log("listening to port : 8080");
73 });
```

```
app.patch("/posts/:id", (req, res) => {  
  let {id} = req.params;  
  let newContent = req.body.content;  
  console.log(newContent);  
  let post = posts.find((p) => id === p.id);  
  post.content = newContent;  
  console.log(post);  
  res.redirect("/posts");  
})  
  
app.get("/posts/:id/edit", (req, res) => {  
  let {id} = req.params;  
  let post = posts.find((p) => id === p.id);  
  res.render("edit.ejs", {post});  
})
```

```
</head>
<body>
  <h2>Edit Your Post</h2>
  <h4>Post_ID: <%= post.id %></h4>
  <h4>Post Username: @<%= post.username %></h4>
  <form action="/posts/<%=post.id%>?_method=PATCH" method="post">
    <textarea name="content" id="content" cols="30" rows="10"><%= post.content %></textarea>
    <br>
    <button>Submit</button>
  </form>
</body>
</html>
```

Quora Posts

@Sneha Gupta

I Love my job

[See more info...](#) [Edit form...](#)

@Adarsh Goyal

Make money and have world knee down to u

[See more info...](#) [Edit form...](#)

@Anchal Goyal

I wanna be a doctor so that i can help people

[See more info...](#) [Edit form...](#)

@Bhaiya

Good jobs! Keep it up

[See more info...](#) [Edit form...](#)







[Create New Post](#)

Implement : **/posts/:id**

Destroy Route

DELETE **/posts/:id** to delete specific post


```
1  <link rel="stylesheet" href="/style.css" />
2  </head>
3  <body>
4    <h1>Quora Posts</h1>
5    <% for(post of posts) { %>
6      <div class="post">
7        <h3 class="user">@<%= post.username %></h3>
8        <h4 class="content"><%= post.content %></h4>
9        <br />
10       <a href="http://localhost:8080/posts/<%=post.id%>">See in Detail</a>
11       <a href="http://localhost:8080/posts/<%=post.id%>/edit">Edit</a>
12       <form method="post" action="/posts/<%=post.id%>?_method=DELETE">
13         <button>Delete Post</button>
14       </form>
15     </div>
16   <%}%>
17   <br />
18   <br />
```



EXPLORER

REST_CLASS

- node_modules
- public
- views
 - edit.ejs
 - index.ejs
 - new.ejs
 - show.ejs
- JS index.js
- package-lock.json
- package.json

JS index.js

```
JS index.js > app.delete("/posts/:id") callback
61 console.log(post);
62 res.redirect("/posts");
63 });
64
65 app.get("/posts/:id/edit", (req, res) => {
66   let { id } = req.params;
67   let post = posts.find((p) => id === p.id);
68   res.render("edit.ejs", { post });
69 });
70
71 app.delete("/posts/:id", (req, res) => {
72   let { id } = req.params;
73   let post = posts.find((p) => id === p.id);
74   res.send("delete success");
75 });
76
77 app.listen(port, () => {
78   console.log("listening to port : 8080");
79 });
```

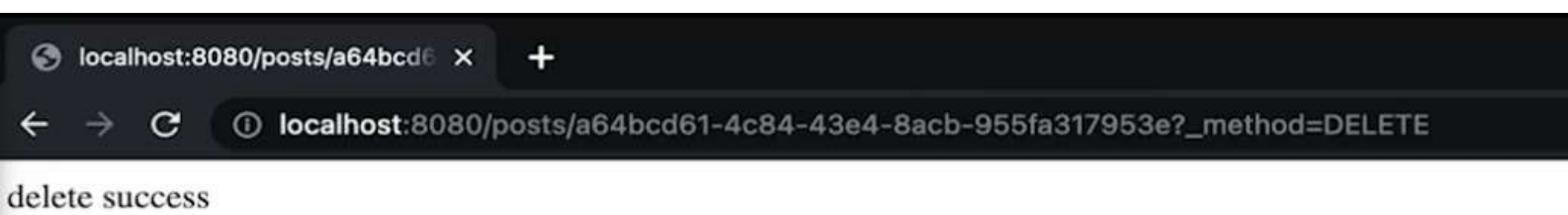
PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

```
{
  id: '2eaa42b5-c88a-469b-9bb8-151e6052487b',
  username: 'rahulkumar',
  content: 'I got selected for my 1st internship! '
}
```



EXPLORER

REST_CLASS

> node_modules

> public

> views

<> edit.ejs

<> index.ejs

<> new.ejs

<> show.ejs

JS index.js

{ } package-lock.json

{ } package.json

JS index.js

<> index.ejs

JS index.js > app.delete("/posts/:id") callback

65 app.get("/posts/:id/edit", (req, res) => {

66 | let { id } = req.params;

67 | let post = posts.find((p) => id === p.id);

68 | res.render("edit.ejs", { post });

69 | });

70

71 app.delete("/posts/:id", (req, res) => {

72 | let { id } = req.params;

73 | posts = posts.filter((p) => id !== p.id);

74 | res.redirect("/posts");

75 | });

76

77 app.listen(port, () => {

78 | console.log("listening to port : 8080");

79 | });

80

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

at next (/Users/shradhakhapra/WebDevelopment/Backend/rest_class,

js:280:10)

[nodemon] restarting due to changes...

[nodemon] starting `node index.js`

listening to port : 8080

Quora Posts

@apnacollege

I love coding!

[See in Detail](#) [Edit](#)

Delete Post

@shradhakhapra [shradhagupta7385@gmail.com](#)

Hard work is important to achieve success

[See in Detail](#) [Edit](#)

Delete Post

@rahulkumar

I got selected for my 1st internship!

[See in Detail](#) [Edit](#)

Delete Post

[Create New Post](#)

Quora Posts

@shradhakhapra

Hard work is important to achieve success

[See in Detail](#) [Edit](#)

Delete Post

@rahulkumar

I got selected for my 1st internship!

[See in Detail](#) [Edit](#)

Delete Post

[Create New Post](#)