

EdX and its Members use cookies and other tracking technologies for performance, analytics, and marketing purposes. By using this website, you accept this use. Learn more about these technologies in the [Privacy Policy](#).



[Course](#) > [Expres...](#) > [Imple...](#) > [Imple...](#)

Implement REST API routing

Video: Implementing REST API Routing

The video player displays a code editor with the following JavaScript code for Express.js routing:

```
12
13 app.get('/profile', (req, res) => {
14   res.send(profile)
15 })
16
17 app.post('/profile', (req, res) => {
18   profile = req.body
19   console.log('created', profile)
20   res.sendStatus(201)
21 })
22
23 app.put('/profile', (req, res) => {
24   Object.assign(profile, req.body)
25   console.log('updated', profile)
26   res.sendStatus(204)
27 })
28
29 app.delete('/profile', (req, res) => {
30   profile = {}
31   console.log('deleted', profile)
32   res.sendStatus(204)
33 })
34
35 app.listen(3000)
36
```

The file explorer on the right shows a project structure with folders for lessons, modules, and project files, and files for package.json, server.js, and test.sh.

Video player controls: 0:00 / 0:00, 1.50x, and other standard controls.

Video

[Download video file](#)

Transcripts

[Download SubRip \(.srt\) file](#)

[Download Text \(.txt\) file](#)

Implementing REST API Routing

Servers must have routes, otherwise they are not useful at all. As a very basic example, consider this Express route which serves Hello World string to requests made to / (root address):

```
app.get('/', (req, res) => {  
  res.send('Hello World')  
})
```

You can notice that `app.get()` is referring to the GET HTTP method. That's what browsers will use to navigate to a URL in a browser.

The first argument is a URL string. It could be a regular expression as well. The second argument is a request handler with request and response objects.

If you have two routes in `app.js`:

```
const {homePage, getUsers} = require('./routes')  
  
app.get('/', homePage)  
app.get('/users', getUsers)
```

The first one basically takes care of all the GET requests to the home page (/), such as `http://localhost:3000/` and triggers the `homePage` method. The second takes care of requests to `/users`, such as `http://localhost:3000/users` and triggers the `getUsers` method.

Both of the routes process URLs in a case-insensitive manner and in the same way with trailing slashes.