

Practical 11

Link to my GitHub repository:

<https://github.com/RaviThakur322/Node-JS-Practical-GF2023485999/tree/main>

Create the website and track the patterns. If there is redundancy resolve it using the different path module and if no redundancy which path module you will be executing and why? Answer with the code along with the your opinion.

1. app.js

```
Practical 11 > JS app.js > ...
 1  import express from 'express';
 2  import normalizePath from './middlewares/normalizePath.js';
 3
 4  const app = express();
 5  const port = 3000;
 6
 7  app.use(normalizePath);
 8
 9  app.get('/', (req, res) => {
10    | res.send('Welcome to the homepage');
11  });
12
13  app.get('/user/profile', (req, res) => {
14    | res.send('User Profile Page');
15  });
16
17  app.get('/*splat', (req, res) => {
18    | res.send('Fallback page');
19  });
20
21  app.listen(port, () => {
22    | console.log(`Server listening at http://localhost:${port}`);
23  });|
```

2. routes/index.js

Practical 11 > routes > JS index.js > ...

```
1  const express = require('express');
2  const router = express.Router();
3
4  router.get('/user/profile', (req, res) => {
5    |   res.send('User Profile Page');
6  });
7
8  router.get('*', (req, res) => {
9    |   res.send('Fallback page');
10 });
11
12 module.exports = router;
```

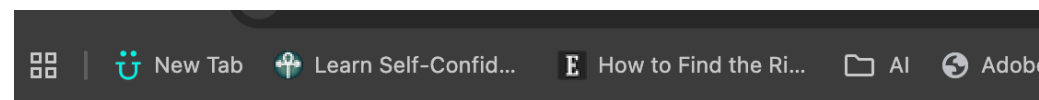
3. middlewares/normalizePath.js

Practical 11 > middlewares > JS normalizePath.js > [⌕] default

```
1  import path from 'path';
2
3  function normalizePath(req, res, next) {
4    |   const originalPath = req.path;
5    |   const normalizedPath = path.posix.normalize(originalPath);
6
7    |   if (normalizedPath !== originalPath) {
8    |     |   console.log(`Redirecting from ${originalPath} to ${normalizedPath}`);
9    |     |   return res.redirect(301, normalizedPath);
10   |   }
11
12   |   next();
13 }
14
15 export default normalizePath;
```

Output:

```
(Use `node --trace-warnings ...` to show where the warning was emitted)
Server listening at http://localhost:3000
>
```



Welcome to the homepage