

Installing nodemon dependencies:

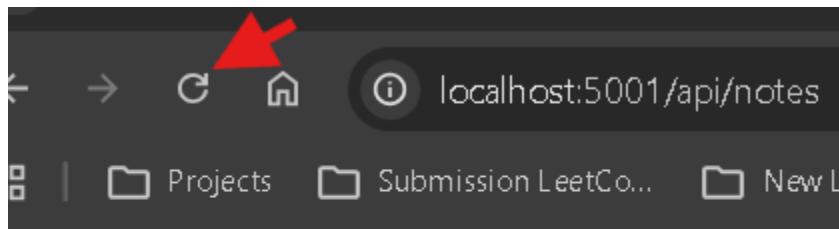
While server is running...

```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard> cd .\backend\  
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run dev  
  
> backend@1.0.0 dev  
> node server.js  
  
Server started on PORT: 5001...  
█
```

If we change the code to:

```
server.js  JS server.js  X  
> JS server.js > app.get("/api/notes") callback  
1 import express from "express"  
2 const app = express()  
3  
4 app.get("/api/notes", (req, res) => {  
5   res.send("you got 10 notes")  
6 })  
7  
8 app.listen(5001, () => {  
9   console.log("Server started on PORT: 5001...")  
10 })
```

And refresh it:



you got 5 notes

Does not changes, to get the change:

To close the server:

Click: Ctrl + c

click : y key

Click: enter key

Need to use the Command:

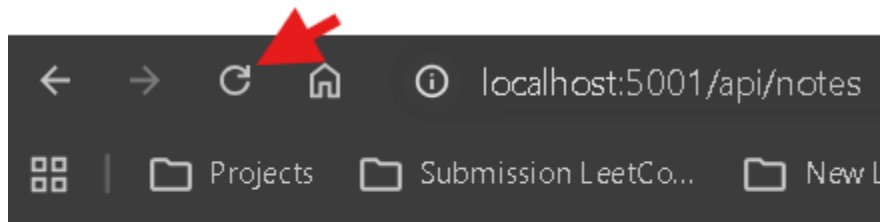
npm run dev

```
Server started on PORT: 5001...
Terminate batch job (Y/N)? y
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run dev

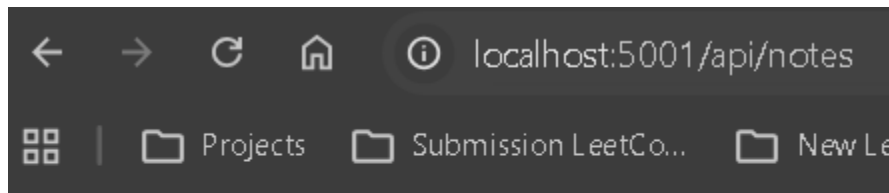
> backend@1.0.0 dev
> node server.js

Server started on PORT: 5001...
```

Then refresh it:



you got 5 notes



you got 10 notes

To make things easier like re-executing **npm run dev**, after change in code.

Install the nodemon as the Dev dependencies:

Command: **npm install nodemon -D**

```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm install nodemon -D

added 29 packages, and audited 98 packages in 2s

16 packages are looking for funding
  run `npm fund` for details

7 vulnerabilities (3 low, 4 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
```

Added in package.json:

```
{ } package.json X JS server.js
backend > { } package.json > { } scripts > abc dev
13 "dependencies": {
14   "express": "..."
15 },
16 "devDependencies": {
17   "nodemon": "^3.1.11"
18 }
```

And change:

```
"scripts": {
  "dev": "node server.js"
},
```

To:

```
"scripts": {  
  "dev": "nodemon server.js"  
},
```

Now run the server: `npm run dev`

```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run dev  
  
> backend@1.0.0 dev  
> nodemon server.js  
  
[nodemon] 3.1.11  
[nodemon] to restart at any time, enter `rs`  
[nodemon] watching path(s): *.*  
[nodemon] watching extensions: js,mjs,cjs,json  
[nodemon] starting node server.js  
Server started on PORT: 5001...
```

Watching extensions

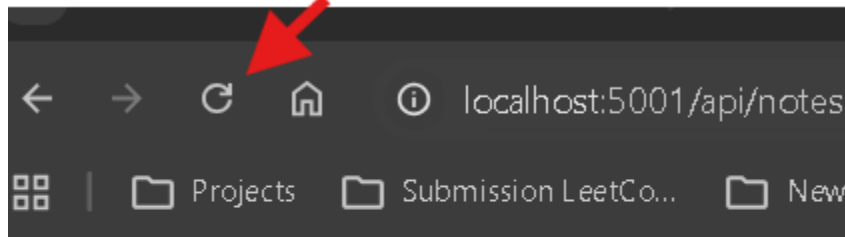
Change the code:

```
✓ app.get("/api/notes", (req, res) => {  
  res.send("you got 5 notes")  
})
```

In terminal:

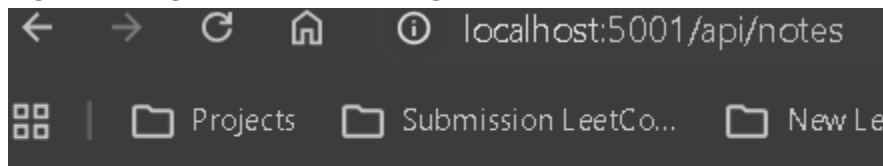
```
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server started on PORT: 5001...
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server started on PORT: 5001...
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server started on PORT: 5001...
█
```

No need to stop running
Refresh the page:



you got 10 notes

It gets change without stopping the server and rerun it:



you got 5 notes

During the development: We use dev

```
"scripts": {
  "dev": "nodemon server.js",
  "start": "node server.js"
},
```

After deployment: we use start(no need to change the code).

Entire below code is route:

```
app.get("/api/notes", (req, res) => {  
  res.send("you got 5 notes")  
})
```

Routes code in server.js:

```
import express from "express"  
const app = express()
```

```
// http get  
app.get("/api/notes", (req, res) => {  
  res.status(200).send("You got 5 notes")  
})  
// http post  
app.post("/api/notes", (req, res) => {  
  // Gets data in form of json  
  res.status(201).json({message: "Note created successfully!"})  
})  
  
// :id <- is dynamic id  
// Eg: http://localhost:5001/api/notes/2312 <- where id = 2312  
// http put  
app.put("/api/notes/:id", (req, res) => {  
  res.status(200).json({message: "Note updated successfully!"})  
})  
// http delete  
app.delete("/api/notes/:id", (req, res) => {  
  res.status(200).json({message: "Note deleted successfully!"})  
})  
  
app.listen(5001, () => {  
  console.log("Server started on PORT: 5001...")  
})
```