

Section 19 → Express.js with Node.js.

① Like JQuery : JS :: Express.js : Node.js.

→ Made specifically for web dev.

② Hyperterminal, & Atom.
(be in root folder)

→ First locate new dir & new folder (name.js)

→ then npm init, fill details

→ then new dir has two files → name.js & package.json

→ atom. → Opens file in atom.

} In cd/hyperterminal
Process to do
everytime.

③ Now we install Express

→ npm install express [be in the proj is] } Run this everytime.

→ Now in atom

→ // js hint esversion:6

```
const express = require("express");
```

```
const app = express();
```

```
app.listen(3000);
```

↳ port we are tuned to.

→ Now we run server in hyper, but it does nothing now, hit ctrl C.

(In atom you can add

```
app.listen(3000, function() {  
  console.log("server started");  
});
```

→ So when you run it on hyper (using node server.js) it shows something to confirm.

④ On browser & localhost:3000 ^{→ type this}
Says cannot GET /

→ So we add `app.get("/", function(request, response) {`
_{location (home page)} _{console.log(*Request);} _{short forms}
④a `});`

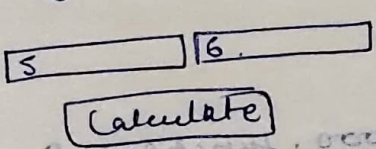
∴ `app.get("/", function(request, response) {`
_(req) _(res) _{short forms}
`response.send("Hello");` _{response} `});`

④b

→ 4a → for info in hyperterminal
→ 4b → for webpage showing.
④c → "/" → home page
→ "/contact"
→ "/bfo" etc.
→ diff pages.

⑤ Nodemon → ^{auto} restart your server
(optional) when changes made in code.
→ `npm install -g nodemon`

⑥ Previously we were sending req to server, which was sending html, css and js files. Now we send code, it is run on server side & only result is send back. Saving time, load faster.
(No code available → in View source) _{code on right click}

⑦ Making Calculator by user input

Result is 11.
we write + html for this and.
in atom inst of `res.send()` _(4b)
we type `res.sendFile("file.html")`

★ Instead of "file.html" you can use dirname + "file.html"
opens current dir when this file run not only locally, anywhere.

③ In html `<form action="" />`, ~~post~~ `method = "post"`

• HTTP return code meanings.

Use `app.post` to allow post request.

- **100** → hold on
- **200** → Here you go
- **300** → Go away (security)
- **400** → Your fault
- **500** → My fault.

→ below `const express = require("express");`
`const bodyParser = require("body-parser");`

④ ⇒ Using `bodyparser` we can use data submitted/typed/pasted as body elements in html & use them -
(see 10:30 mins).

→ `app.use(bodyParser bodyParser.urlencoded extension ({extended: true}));`
