MERN Stack

C R U D Operations

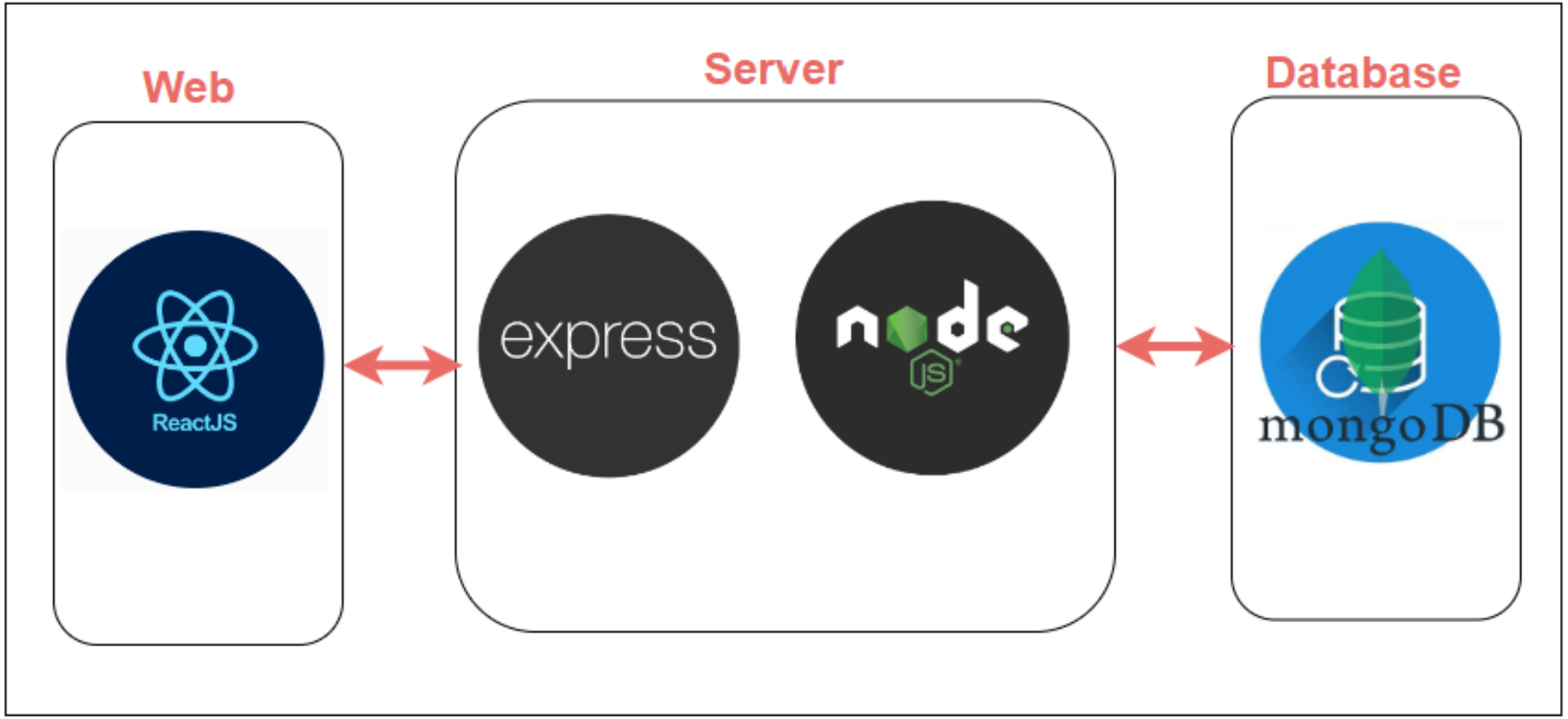
M MongoDB - Document Database

E ExpressJS – Server Side Web FW, Runs in NodeJS, has URL Routing & Handles HttpRequests and Responses

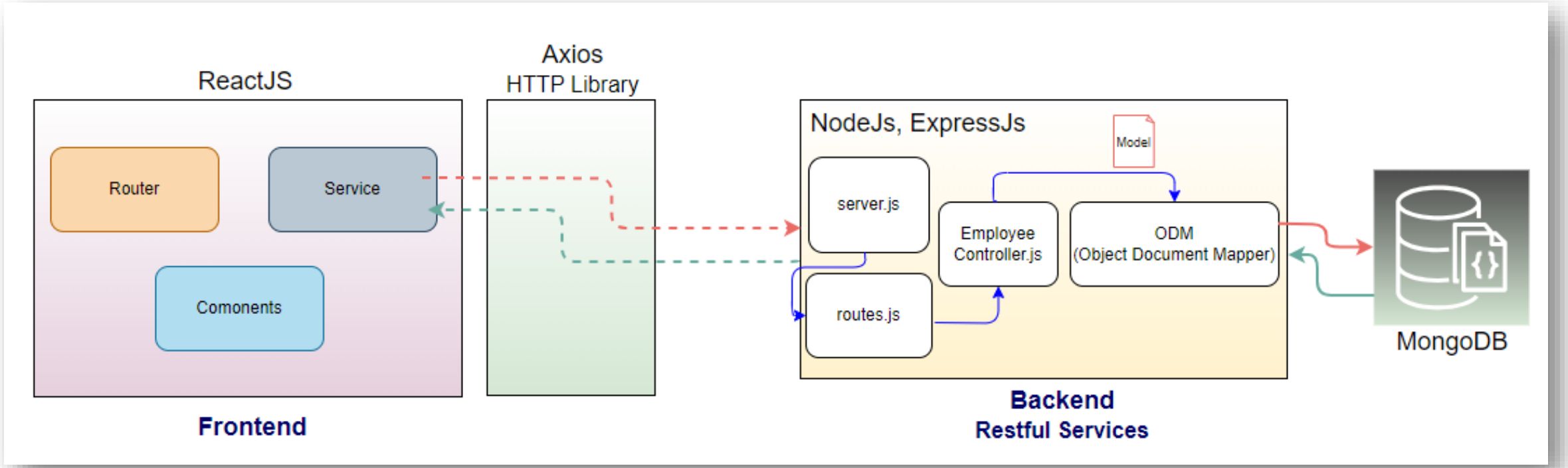
R ReactJS – Client side JavaScript Framework

N NodeJS – JavaScript Web Server

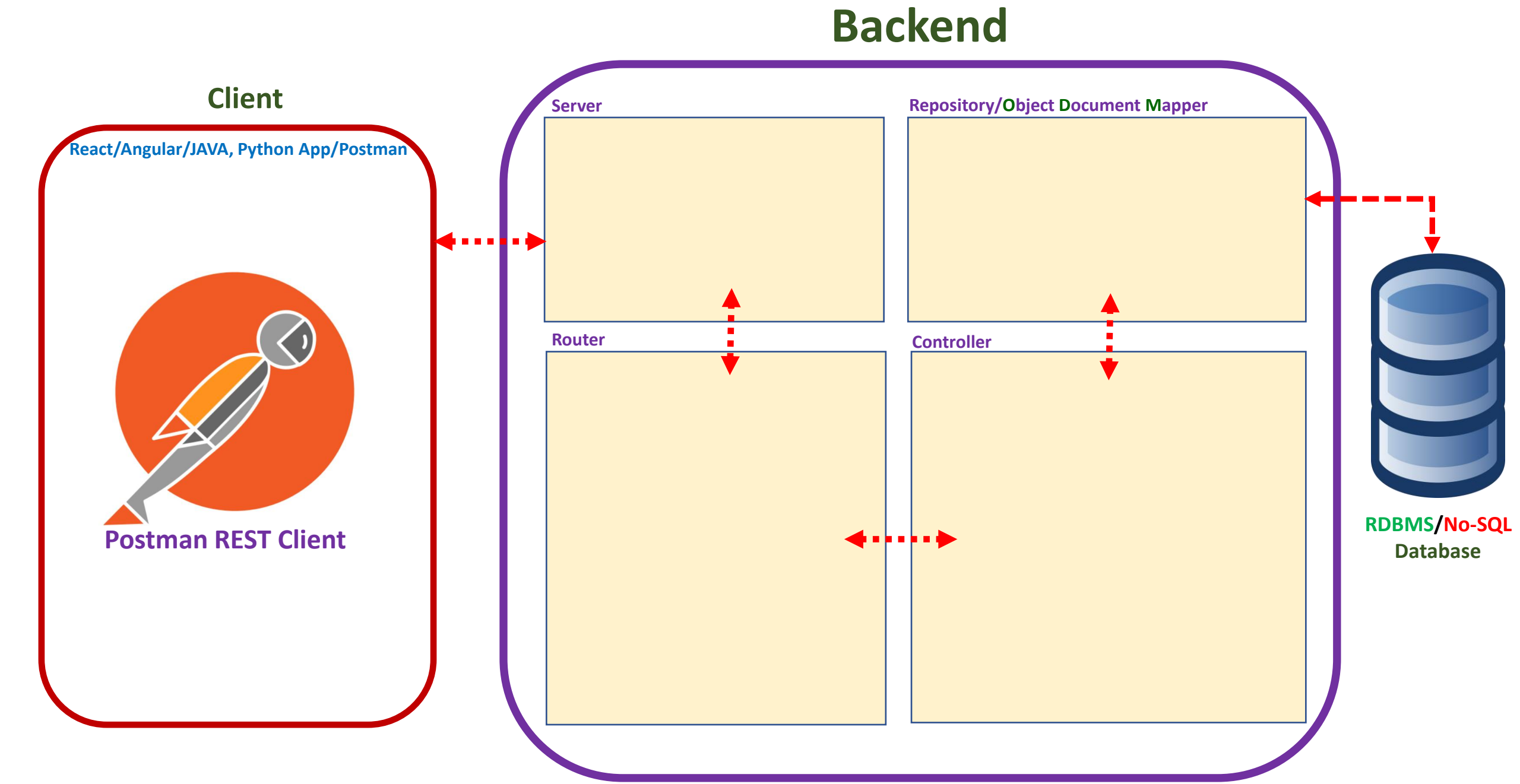
How MERN Stack Works



Architecture - Building Restful Services Using NodeJS, ExpressJS



Building REST API Using NodeJS & ExpressJS



STEPS – Implementing REST API in NodeJS, ExpressJS

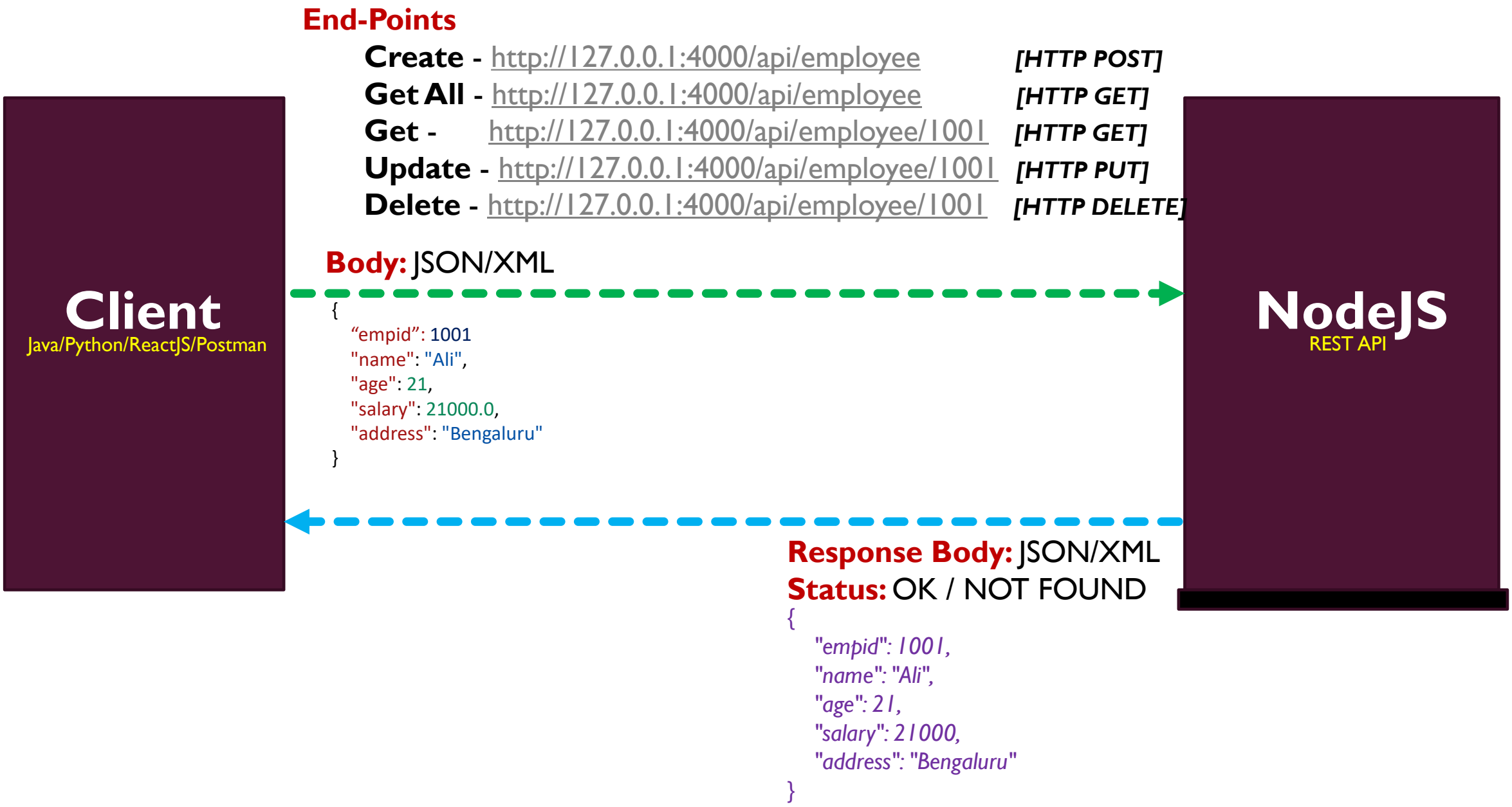
1. Project setup - NodeJS & ExpressJS

2. Design the Contract: API End-Points, Request, Response, Error Codes

3. API Implementation
   - Create server.js By Configuring all routes with project level path
   - Create routes.js with Routers (end-point & Handler Method Mapping)
   - Create Controller with Handler methods which Maps to End-Points
   - Create a Model Class with Schema
   - Create a db.js file with Database configuration & Return Database Connection

4. Test API Using any REST Clients - Postman

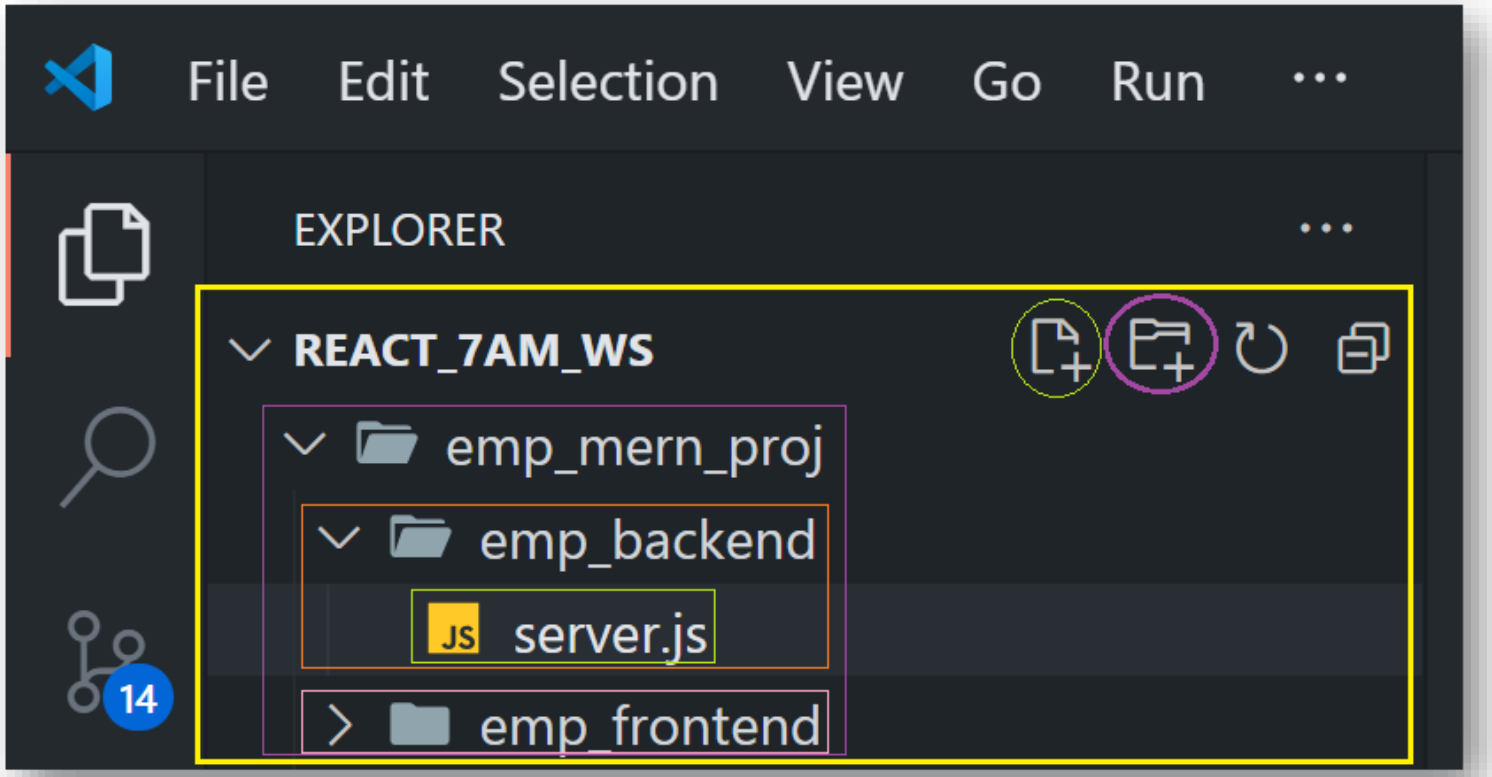
Designing the Contract



Backend

Project Setup

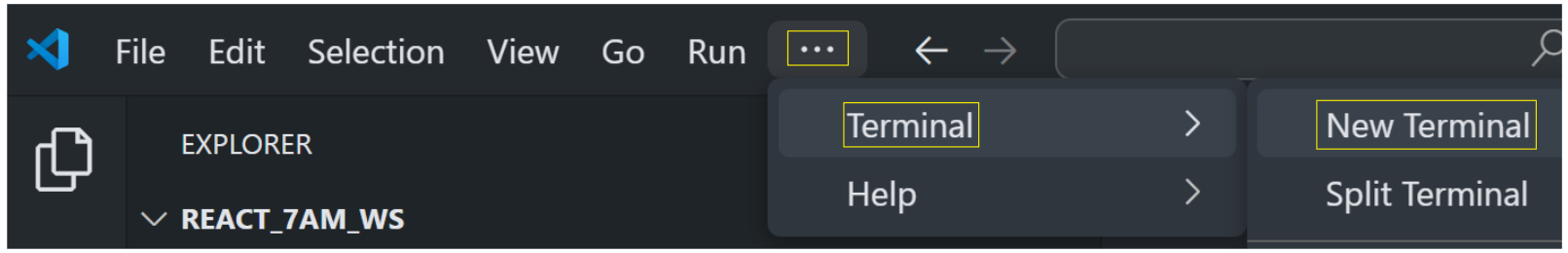
Project folders Creation



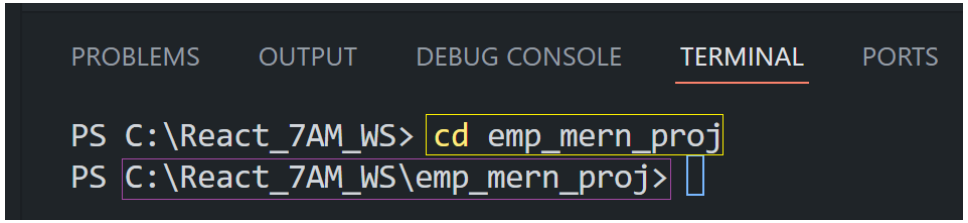
1. Create **emp_mern_proj** Folder inside Workspace folder
2. Create **emp_frontend** folder inside emp_mern_proj folder
3. Create **emp_backend** folder inside emp_mern_proj folder
4. Create **server.js** file inside emp_backend folder

npm init command - Initialize backed service

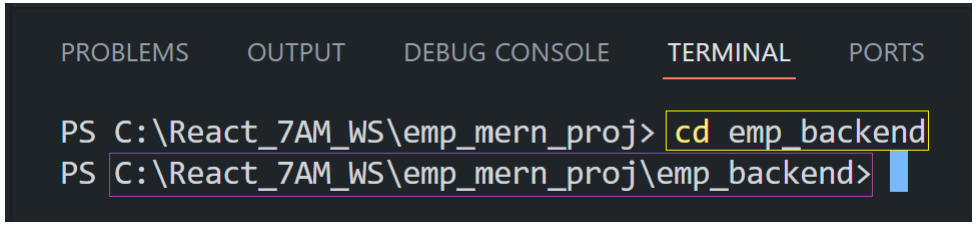
- Go to the terminal



- From workspace folder, Go to project folder using **cd emp_mern_proj**



- From emp_mern_proj folder, Go to backend folder using **cd emp_backend**



npm init command - Initialize backed service …

- From terminal, run npm init command from **emp_backend** folder

```
PS C:\React_7AM_WS\emp_mern_proj\emp_backend> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (emp_backend) emp_backend_service
version: (1.0.0)
description:
entry point: (server.js)
test command:
git repository:
keywords:
author:
license: (ISC)
```

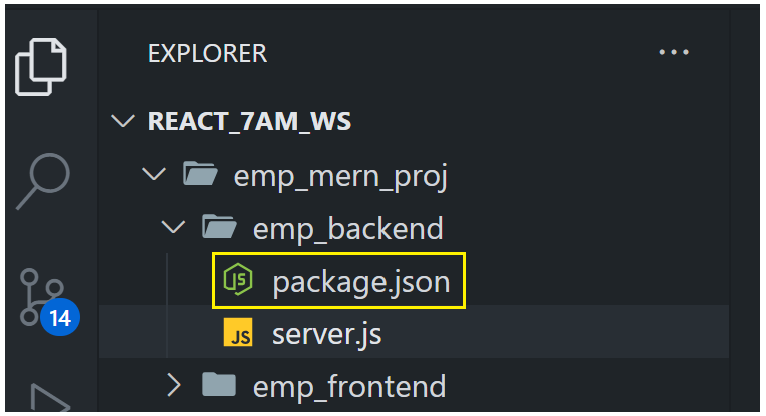
Press Enter

```
About to write to C:\React_7AM_WS\emp_mern_proj\emp_backend\package.json:

{
  "name": "emp_backend_service",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "",
  "license": "ISC"
}


Is this OK? (yes)
```

Press Enter

```
PS C:\React_7AM_WS\emp_mern_proj\emp_backend>
```

- **package.json** new file will be created in **emp_backend** folder



nodemon setup - Automatic Server Restart

- Install nodemon

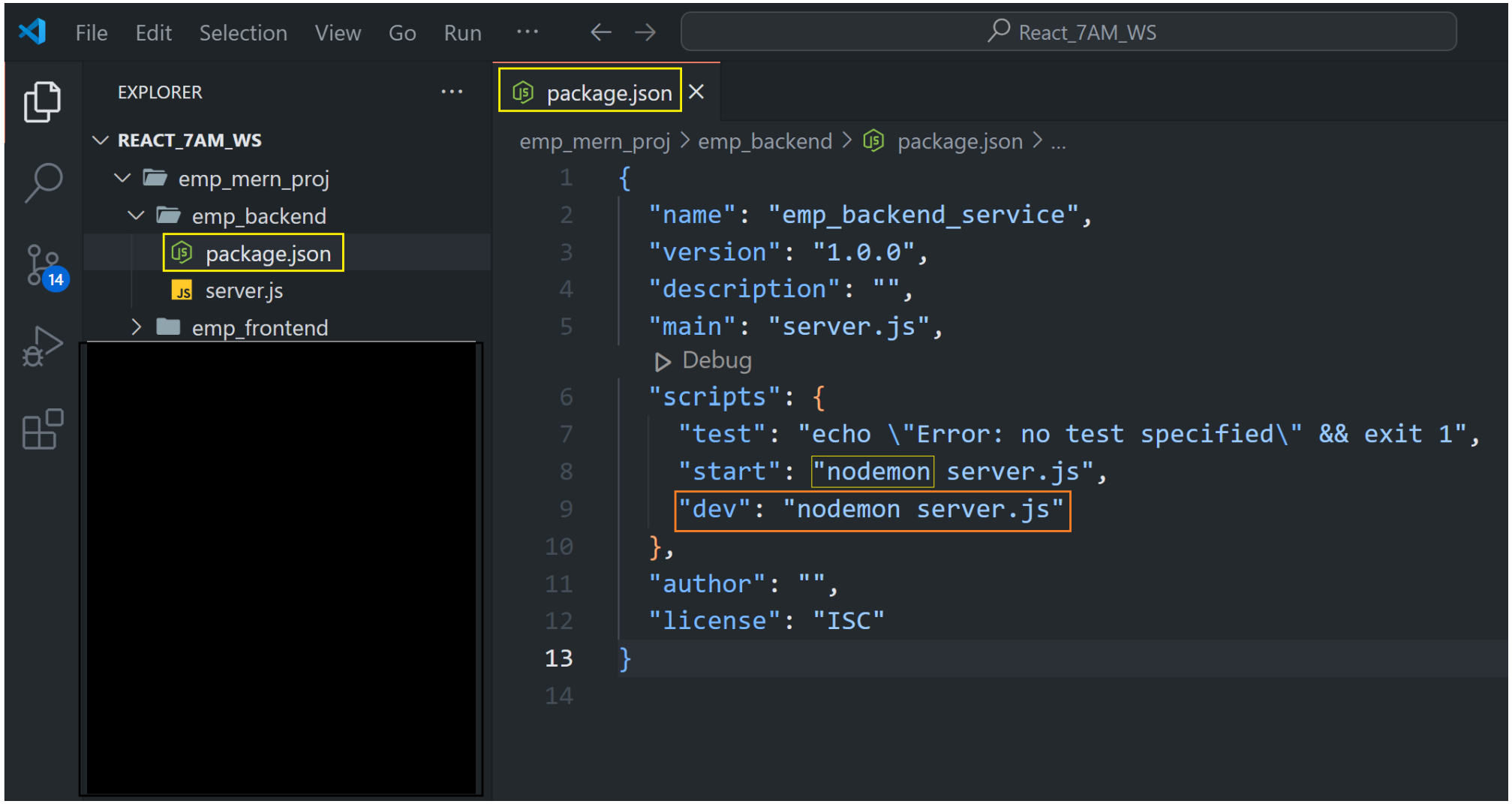
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_backend> npm install -g nodemon

added 29 packages in 2s

4 packages are looking for funding
  run `npm fund` for details
PS C:\React_7AM_WS\emp_mern_proj\emp_backend>
```

- Checking nodemon Version

```
C:\Users\shanv>nodemon -v
3.1.1

C:\Users\shanv>
```

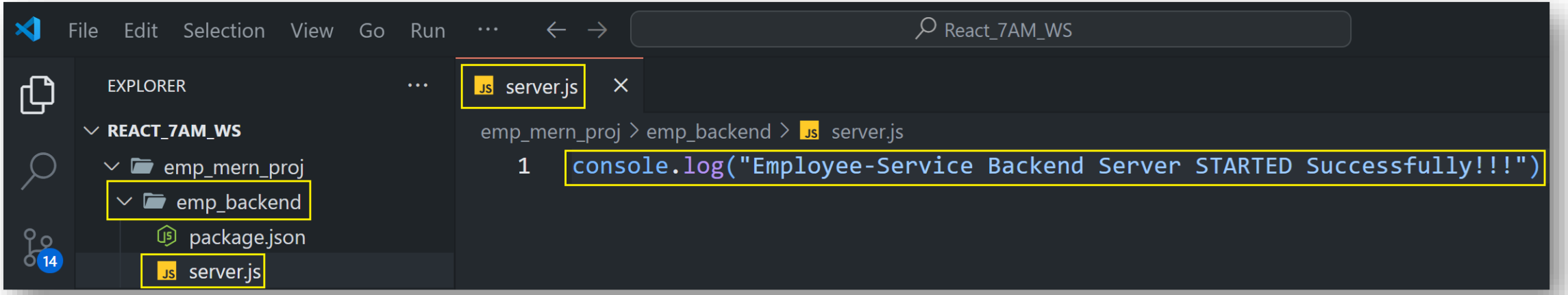
nodemon setup - Automatic Server Restart…

- Go to package.json
- Update scripts "**start**" & add "**dev**"

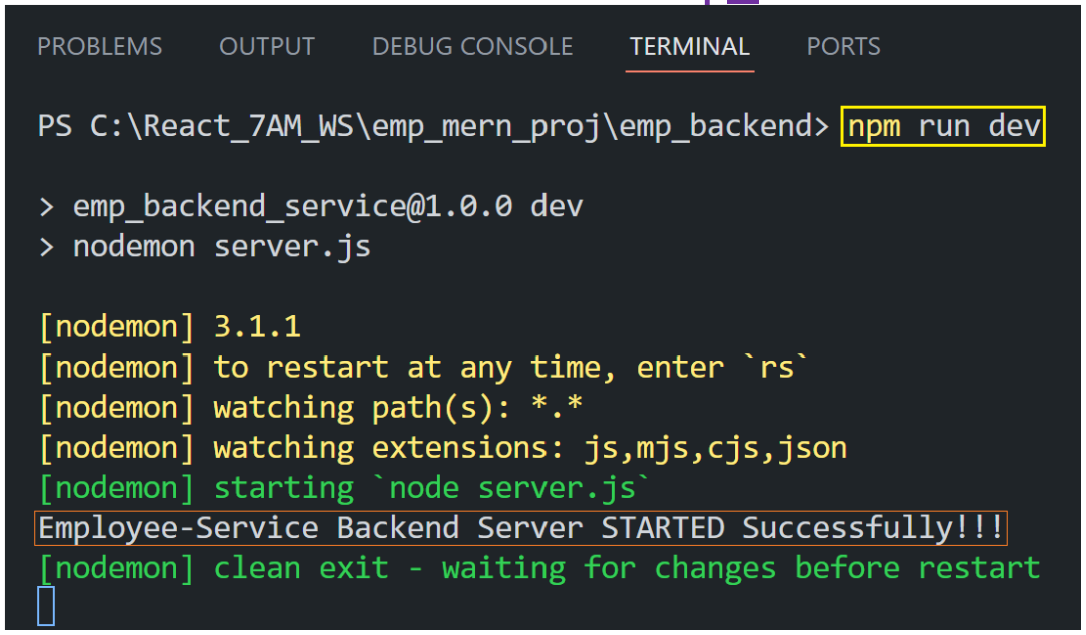


Testing Backend Server

- Add Log statement in emp_backend/server.js



- Start the backed server from emp_backend folder using **npm run dev** command



Install required software

- Install express

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_backend> npm i express

added 64 packages, and audited 65 packages in 3s

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_backend>
```

- Install cors – for connecting Frontend, Backend

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_backend> npm i cors

added 2 packages, and audited 87 packages in 925ms

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_backend>
```

- Install mongoose

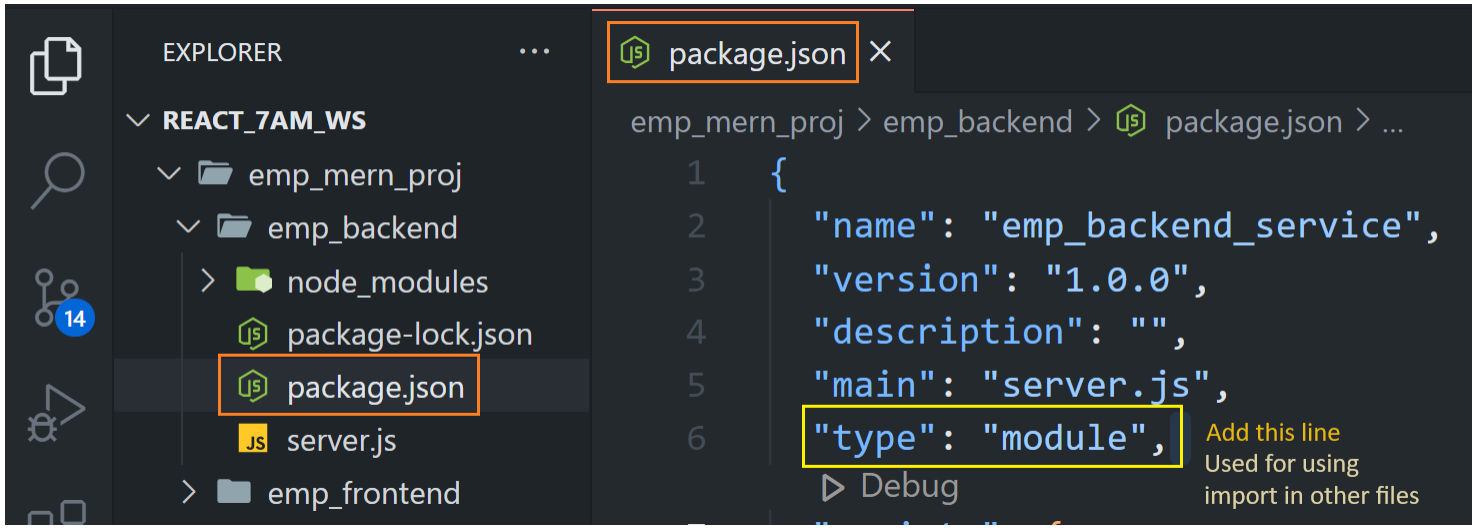
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_backend> npm i mongoose

added 20 packages, and audited 85 packages in 3s

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_backend>
```

- Install dotenv – for reading files

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_backend> npm i dotenv

added 1 package, and audited 88 packages in 799ms

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_backend>
```

Changes in package.json & server.js

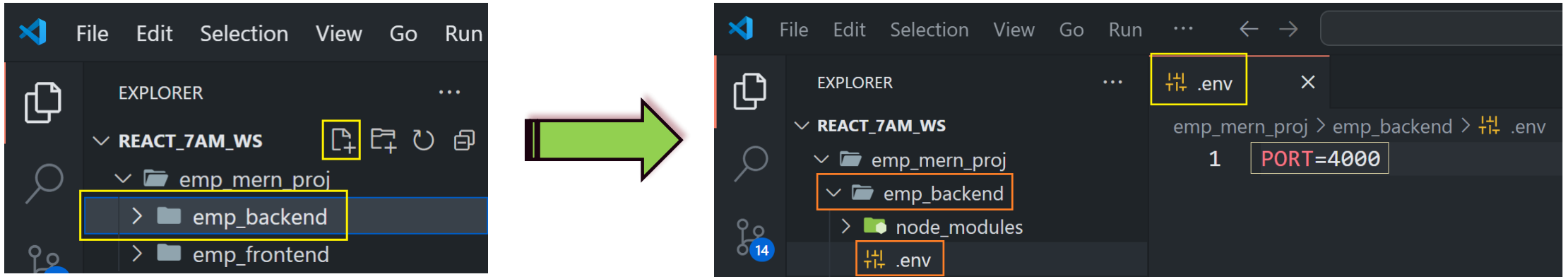
- Add "**type**": "**module**" field in **package.json** for using import statements in other files



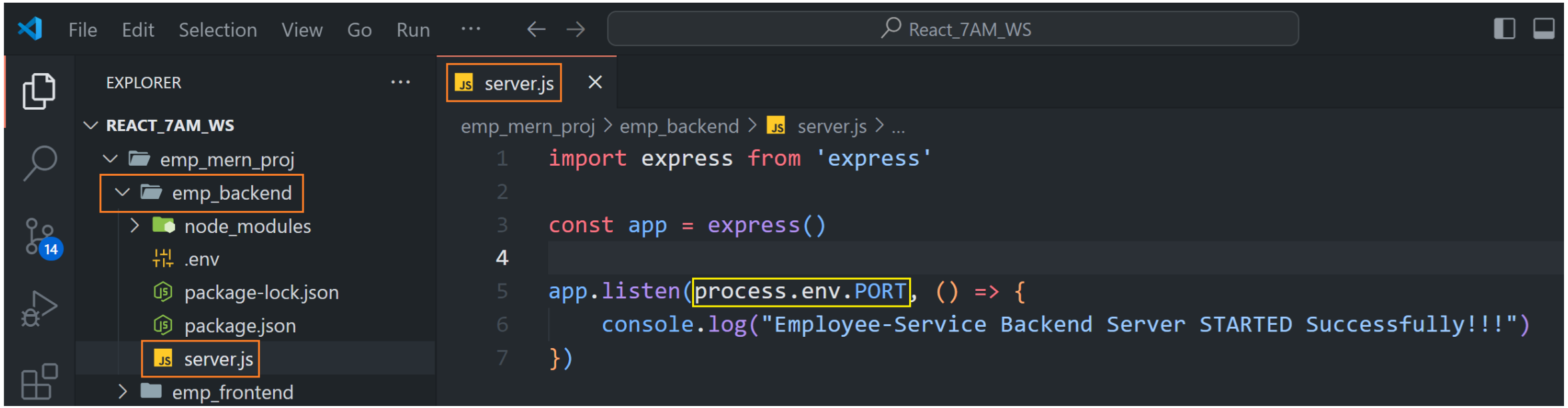
- Do the following changes in **server.js**

Create .env file in emp_backend folder AND define PORT Number

- Defining PORT Value in .env file

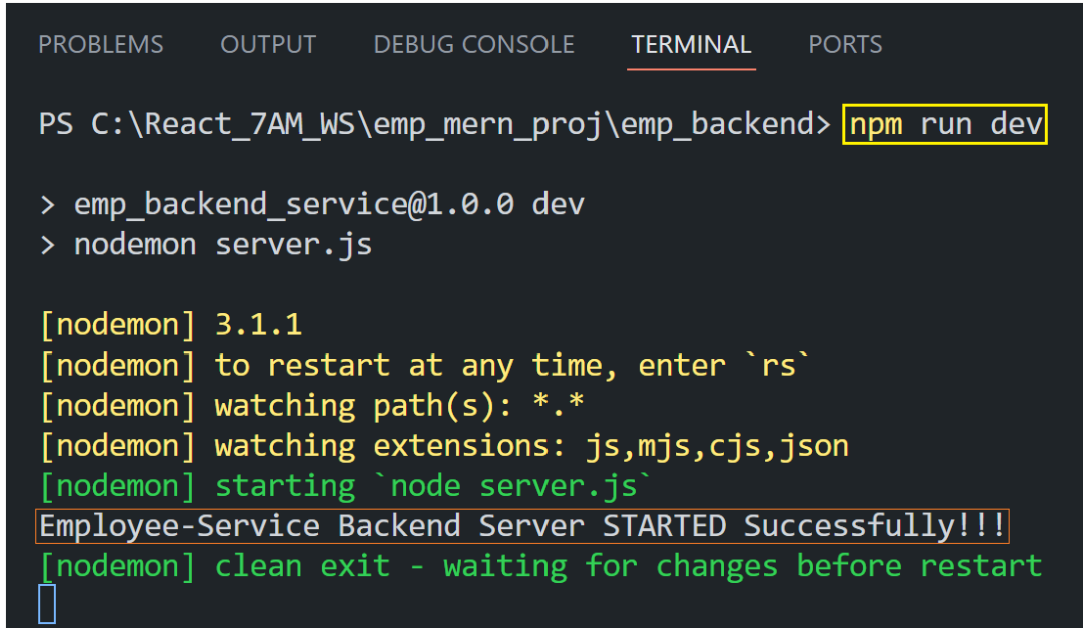


- Reading PORT Value from .env file in server.js



Start Backend Server

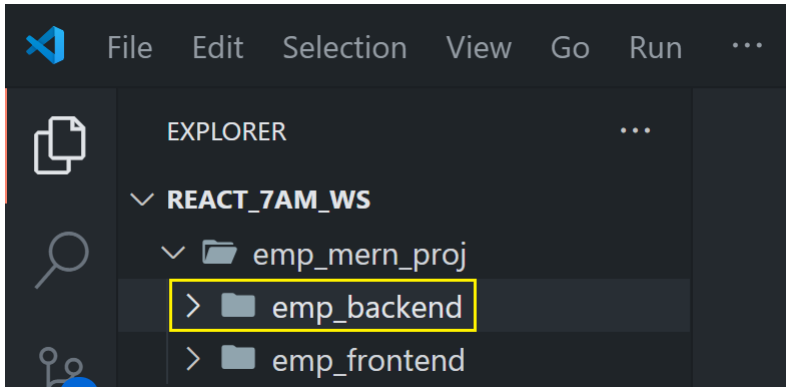
- Start the backed server from emp_backend folder using **npm run dev** command

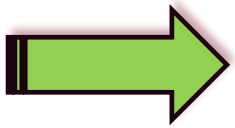


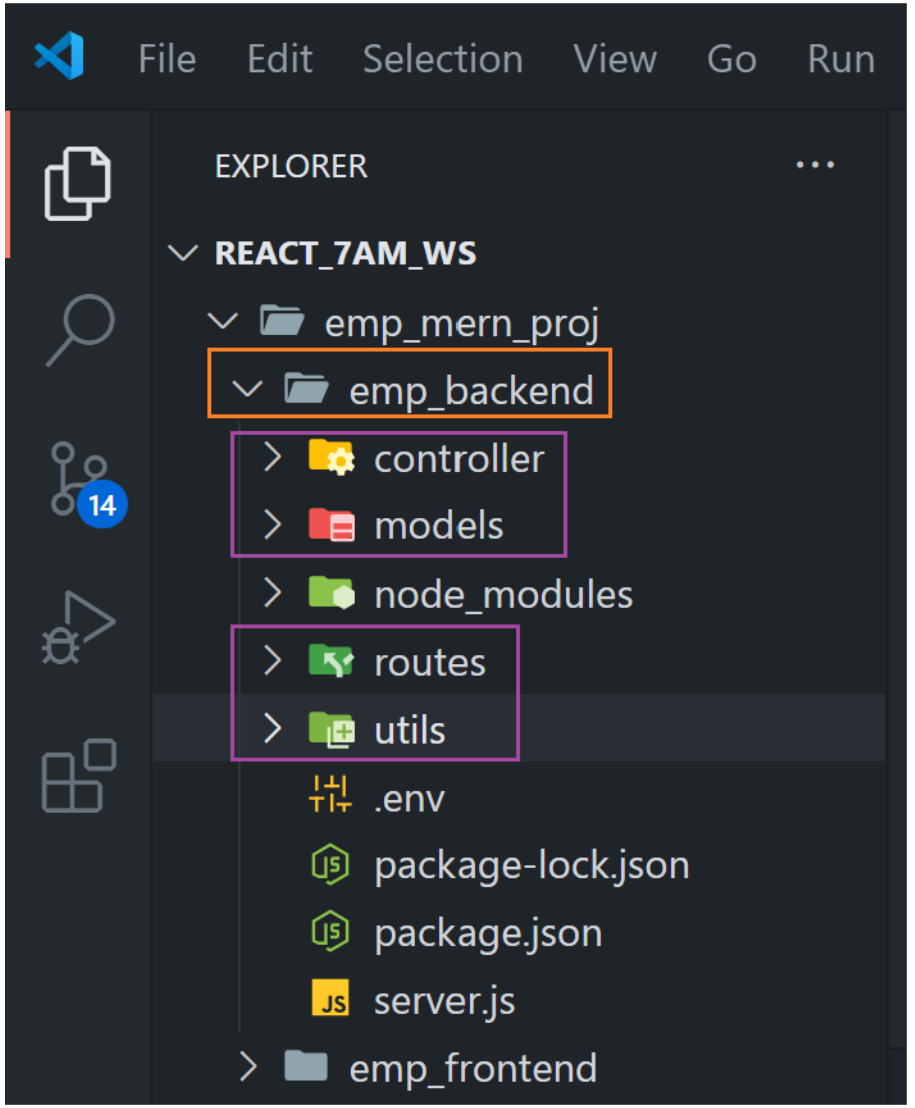
Backend

Package Creation

Create routes, models, controller, utils folders inside emp_backend folder







Configure/Setup

MongoDB Database

Start MongoDB Server

```
C:\Users\shanv>mongod
{"t":{"$date":"2024-05-25T22:14:31.502+05:30"},"s":"I",  "c":"NETWORK",  "id":4915701, "ctx":"thread1","msg":"Initialized wire specification","attr"
:{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":21},"outgo
ing":{"minWireVersion":6,"maxWireVersion":21},"isInternalClient":true}}}
{"t":{"$date":"2024-05-25T22:14:32.876+05:30"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx":"thread1","msg":"Automatically disabling TLS 1.0, to fo
rce-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2024-05-25T22:14:32.880+05:30"},"s":"I",  "c":"NETWORK",  "id":4648602, "ctx":"thread1","msg":"Implicit TCP FastOpen in use."}
{"t":{"$date":"2024-05-25T22:14:32.915+05:30"},"s":"I",  "c":"REPL",     "id":5123008, "ctx":"thread1","msg":"Successfully registered PrimaryOnlySer
vice","attr":{"service":"TenantMigrationDonorService","namespace":"config.tenantMigrationDonors"}}
{"t":{"$date":"2024-05-25T22:14:32.915+05:30"},"s":"I",  "c":"REPL",     "id":5123008, "ctx":"thread1","msg":"Successfully registered PrimaryOnlySer
vice","attr":{"service":"TenantMigrationRecipientService","namespace":"config.tenantMigrationRecipients"}}
{"t":{"$date":"2024-05-25T22:14:32.916+05:30"},"s":"I",  "c":"CONTROL",  "id":5945603, "ctx":"thread1","msg":"Multi threading initialized"}
{"t":{"$date":"2024-05-25T22:14:32.916+05:30"},"s":"I",  "c":"TENANT_M", "id":7091600, "ctx":"thread1","msg":"Starting TenantMigrationAccessBlockerR
egistry"}
{"t":{"$date":"2024-05-25T22:14:32.917+05:30"},"s":"I",  "c":"CONTROL",  "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting","attr":{"pid":
31548,"port":27017,"dbPath":"C:/data/db/","architecture":"64-bit","host":"LAPTOP-17C2SB1B"}}
{"t":{"$date":"2024-05-25T22:14:32.917+05:30"},"s":"I",  "c":"CONTROL",  "id":23398,   "ctx":"initandlisten","msg":"Target operating system minimum
version","attr":{"targetMinOS":"Windows 7/Windows Server 2008 R2"}}
{"t":{"$date":"2024-05-25T22:14:32.918+05:30"},"s":"I",  "c":"CONTROL",  "id":23403,   "ctx":"initandlisten","msg":"Build Info","attr":{"buildInfo":
{"version":"7.0.9","gitVersion":"3ff3a3925c36ed277cf5eafca5495f2e3728dd67","modules":[],"allocator":"tcmalloc","environment":{"distmod":"windows","d
istarch":"x86_64","target_arch":"x86_64"}}}}
```

```
{"t":{"$date":"2024-05-25T22:14:34.450+05:30"},"s":"I",  "c":"STORAGE",  "id":22262,   "ctx":"initandlisten","msg":"Timestamp monitor starting"}
{"t":{"$date":"2024-05-25T22:14:34.454+05:30"},"s":"I",  "c":"NETWORK",  "id":23015,   "ctx":"listener","msg":"Listening on","attr":{"address":"127.
0.0.1"}}
{"t":{"$date":"2024-05-25T22:14:34.454+05:30"},"s":"I",  "c":"NETWORK",  "id":23016,   "ctx":"listener","msg":"Waiting for connections","attr":{"por
t":27017,"ssl":"off"}}
{"t":{"$date":"2024-05-25T22:14:34.454+05:30"},"s":"I",  "c":"CONTROL",  "id":8423403, "ctx":"initandlisten","msg":"mongod startup complete","attr":
{"Summary of time elapsed":{"Startup from clean shutdown?":true,"Statistics":{"Transport layer setup":"0 ms","Run initial syncer crash recovery":"0
ms","Create storage engine lock file in the data directory":"0 ms","Get metadata describing storage engine":"0 ms","Validate options in metadata aga
```

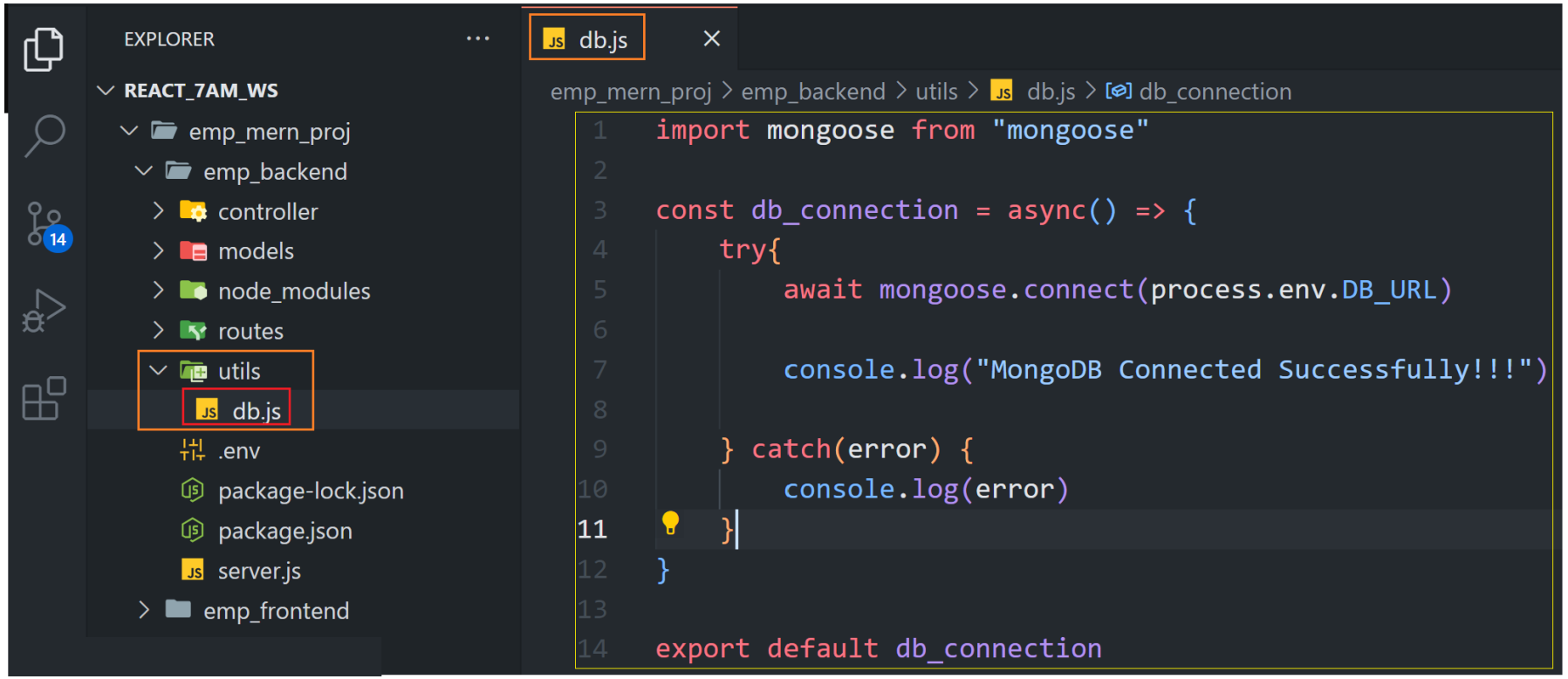
MongoDB Connection String

DB_RUL = mongodb://127.0.0.1:27017/**astidb**

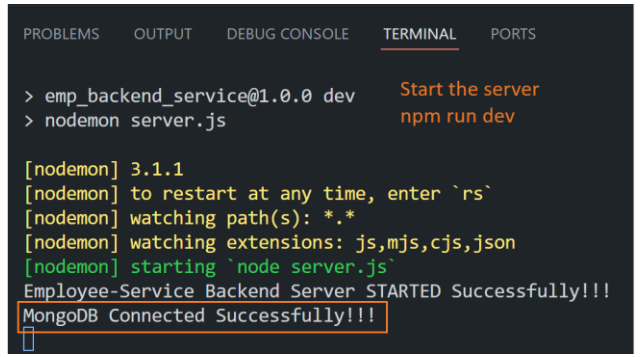
```
PS C:\Users\shanv> mongosh
Current Mongosh Log ID: 665219c848e302bdbf46b798
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.5
Using MongoDB:          7.0.9
Using Mongosh:          2.2.5
mongosh 2.2.6 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2024-05-18T12:05:26.386+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
  You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.
test>
```

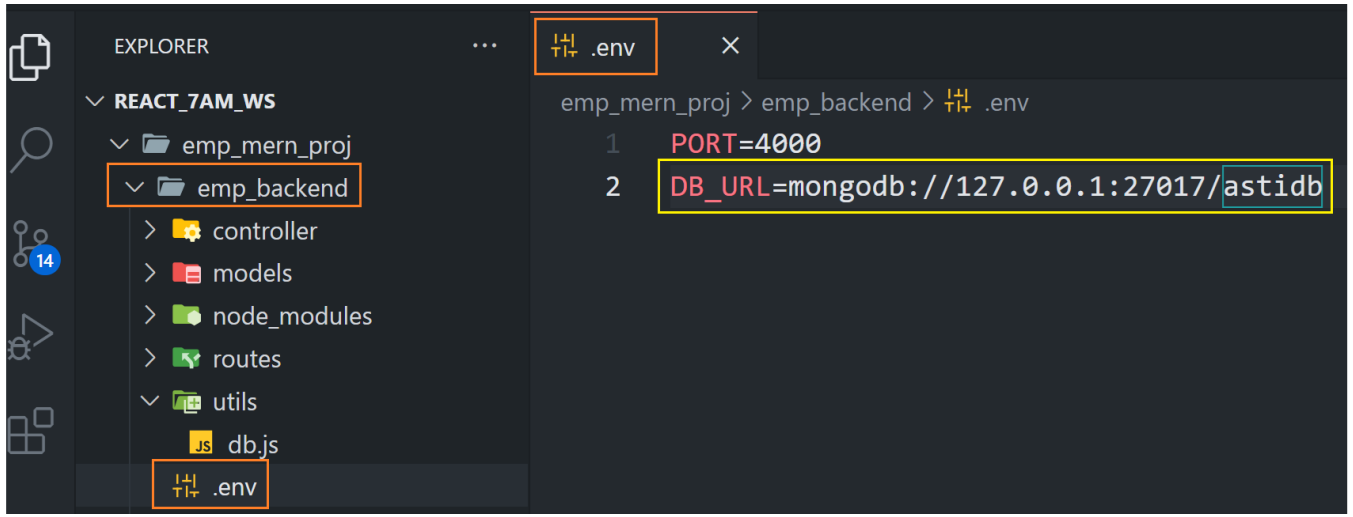
This is MongoDB Connection String

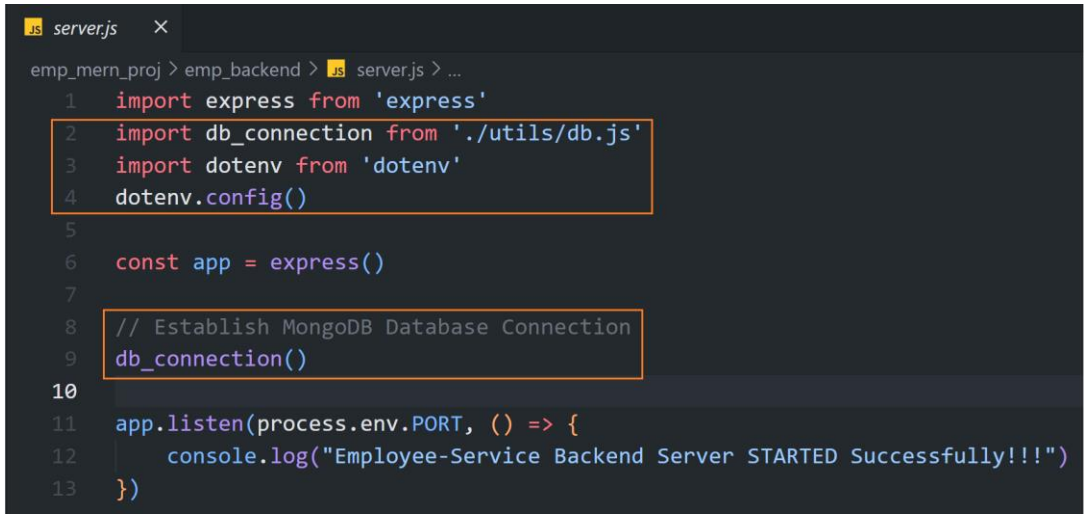
Configure Database connection



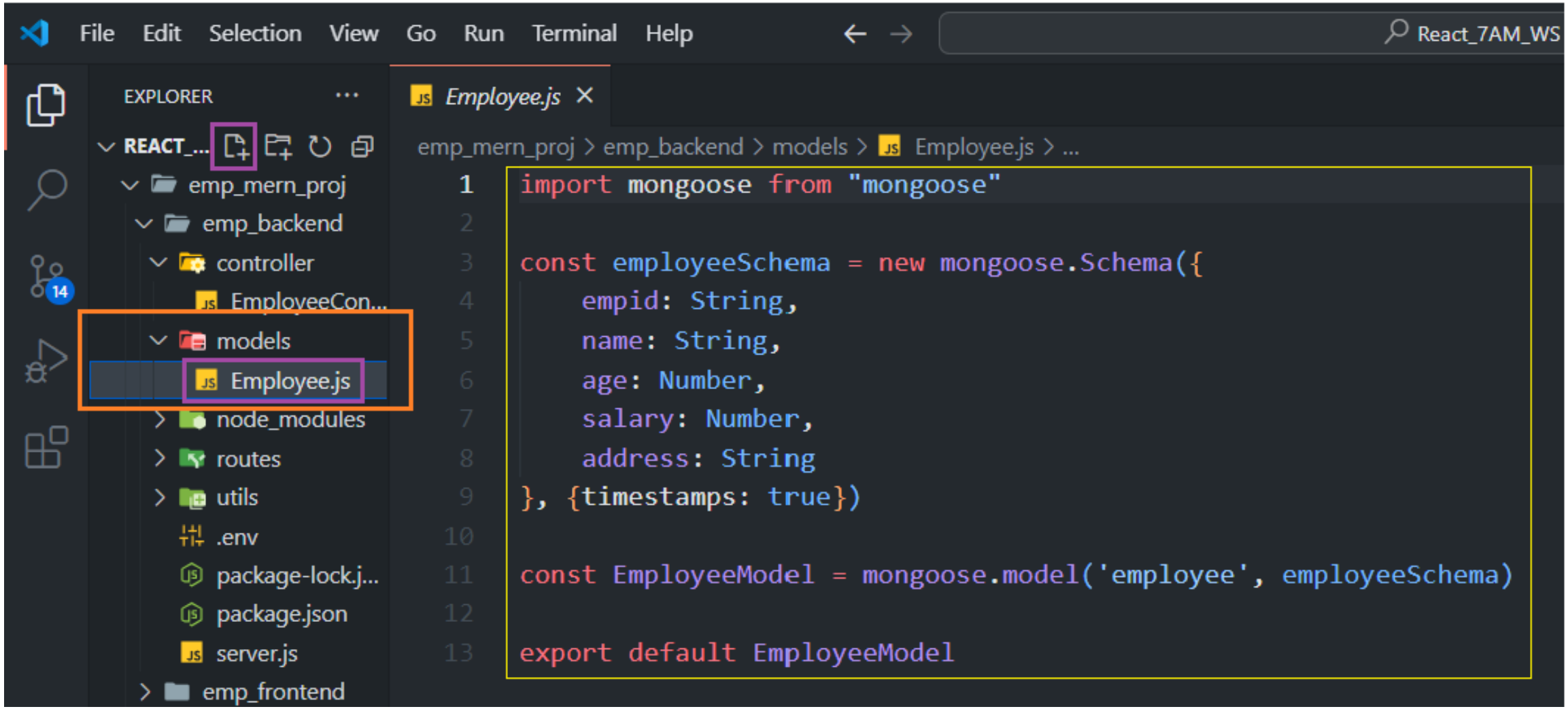
- ✓ Changes in .env
- ✓ Create new db.js
- ✓ Changes in server.js
- ✓ Start the server





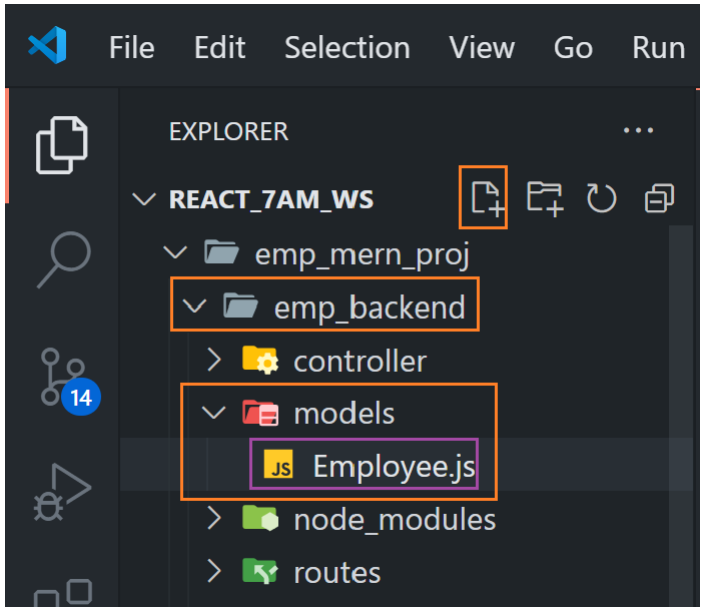


Create Model, Schema



Let us Create A
Employee Model

Create Employee Schema



Employee.js

emp_mern_proj > emp_backend > models > Employee.js > ...

```
1   import mongoose from "mongoose"
2
3   const employeeSchema = new mongoose.Schema({
4       empid: String,
5       name: String,
6       age: Number,
7       salary: Number,
8       address: String
9   }, {timestamps: true})
10
11  const EmployeeModel = mongoose.model('employee', employeeSchema)
12
13  export default EmployeeModel
```

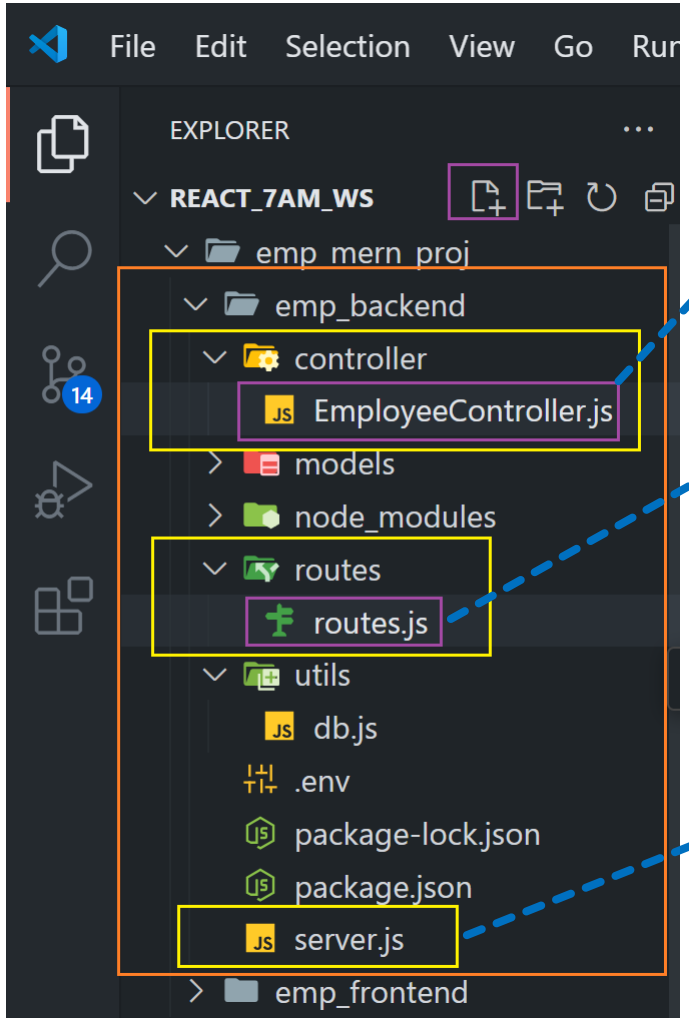
1

2

3

1. Create Employee Schema
2. Create Employee Model
3. Export Employee Model

Create POST Handler Method & Route for CREATE Employee



EmployeeController.js

emp_mern_proj > emp_backend > controller > EmployeeController.js > ...

```
1
2   const CreateEmployee = (request, response) => {
3       response.send("Create Employee Handler Method Got Called!!!")
4   }
5
6   export {CreateEmployee}
```

routes.js

emp_mern_proj > emp_backend > routes > routes.js > default

```
1   import express from 'express'
2   import { CreateEmployee } from '../controller/EmployeeController.js'
3
4   const routers = express.Router()
5
6   //Router for Create Employee
7   routers.post('/employee', CreateEmployee)
8
9   export default routers
```

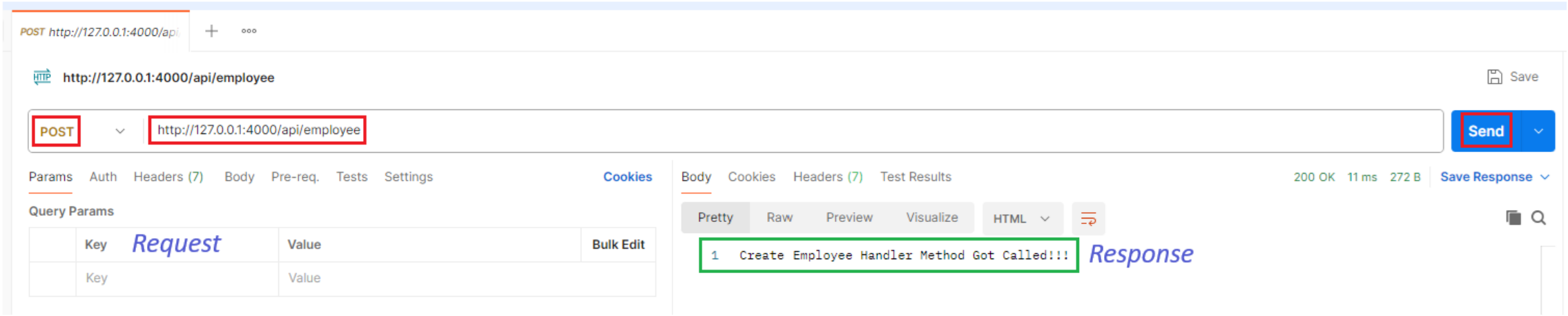
server.js

emp_mern_proj > emp_backend > server.js > ...

```
1   import express from 'express'
2   import db_connection from './utils/db.js'
3   import dotenv from 'dotenv'
4   import routers from './routes/routes.js'
5
6   dotenv.config()
7
8   const app = express()
9
10  // Establish MongoDB Database Connection
11  db_connection()
12
13  // Informing server about all routers
14  app.use('/api', routers)
15
16  app.listen(process.env.PORT, () => {
17      console.log("Employee-Service Backend Server STARTED Successfully!!!")
18  })
```

Start the **SERVER**

Testing POST Service End-Point using POSTMAN



CREATE
Employee API

Create Employee – Full Implementation – EmployeeController.js

EmployeeController.js

emp_mern_proj > emp_backend > controller > EmployeeController.js > ...

```
1  import EmployeeModel from "../models/Employee.js"
2
3  const CreateEmployee = async(request, response) => {
4      //response.send("Create Employee Handler Method Got Called!!!")
5
6      try {
7          const {empid, name, age, salary, address} = request.body
8
9          const empModel = new EmployeeModel({empid, name, age, salary, address})
10         await empModel.save()
11         response.status(201).json({
12             success: true,
13             message: "Employee created successfully", empModel
14         })
15     } catch(error) {
16         console.log(error)
17         response.status(500).json({
18             success: false,
19             message: "Employee NOT Created due to Internal ERROR", empModel
20         })
21     }
22 }
23
24 export {CreateEmployee}
```

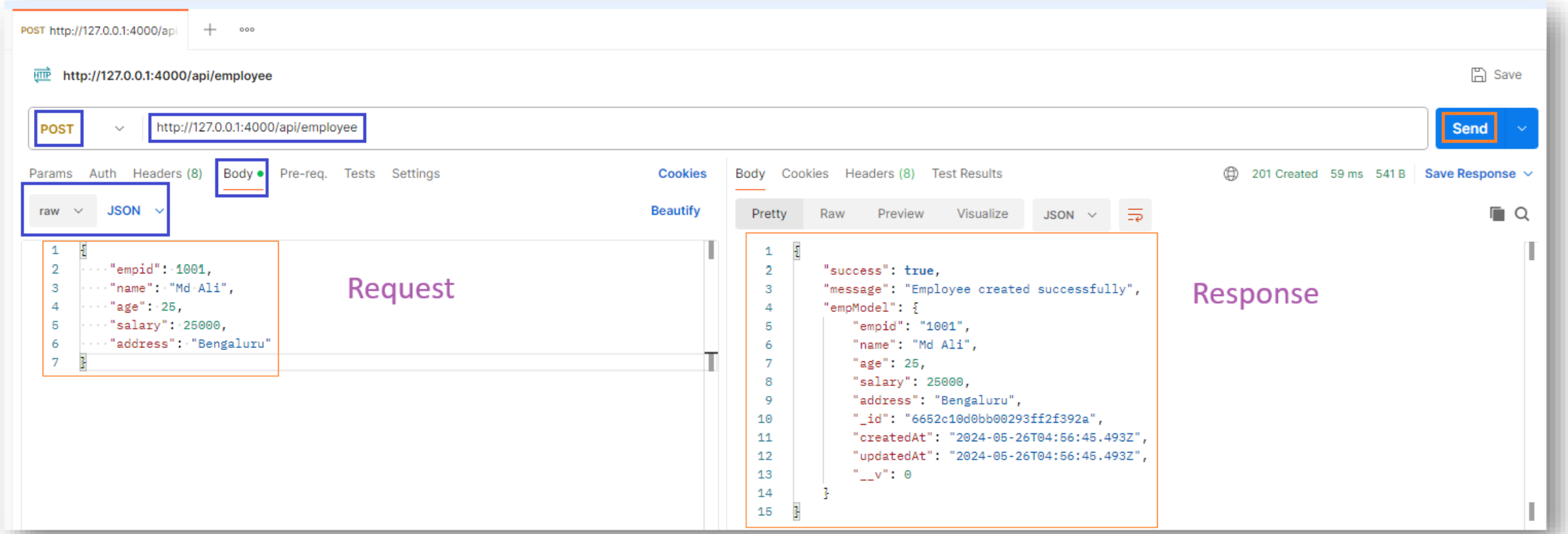
Create Employee – Full Implementation – server.js

server.js

emp_mern_proj > emp_backend > server.js > ...

```
1   import express from 'express'
2   import db_connection from './utils/db.js'
3   import dotenv from 'dotenv'
4   import routers from './routes/routes.js'
5   import cors from 'cors'
6
7   dotenv.config()
8
9   const app = express()
10
11  // Establish MongoDB Database Connection
12  db_connection()
13
14  // Use Middleware
15  app.use(express.json())
16  app.use(cors())
17
18  // Informing server about all routers
19  app.use('/api', routers)
20
21  app.listen(process.env.PORT, () => {
22      console.log("Employee-Service Backend Server STARTED Successfully!!!")
23  })
```

Testing POST/Create Service End-Point using POSTMAN




```
astidb> show collections
employee
employees
astidb> db.employees.find()
[
  {
    _id: ObjectId('6652c10d0bb00293ff2f392a'),
    empid: '1001',
    name: 'Md Ali',
    age: 25,
    salary: 25000,
    address: 'Bengaluru',
    createdAt: ISODate('2024-05-26T04:56:45.493Z'),
    updatedAt: ISODate('2024-05-26T04:56:45.493Z'),
    __v: 0
  }
]
astidb>
```

GET All

Employee API

Get All Employees – Implementation – EmployeeController.js

```
26  //Handler method for GET ALL Employees
27  const GetAllEmployees = async(request, response) => {
28      try {
29          const employees = await EmployeeModel.find()
30
31          if(!employees) {
32              return response.status(404).json({
33                  success: false,
34                  message: "Employee Data NOT Found in Database"
35              })
36          }
37
38          // ELSE If Employees are found then return those employees
39          return response.status(200).json({
40              success: true,
41              message: "Got List of Employees successfully",
42              employees
43          })
44
45      } catch(error) {
46          console.log(error)
47          return response.status(500).json({
48              success: false,
49              message: "Unable to Get Employees due to Internal ERROR"
50          })
51      }
52  }
53
54  export {CreateEmployee, GetAllEmployees}
```

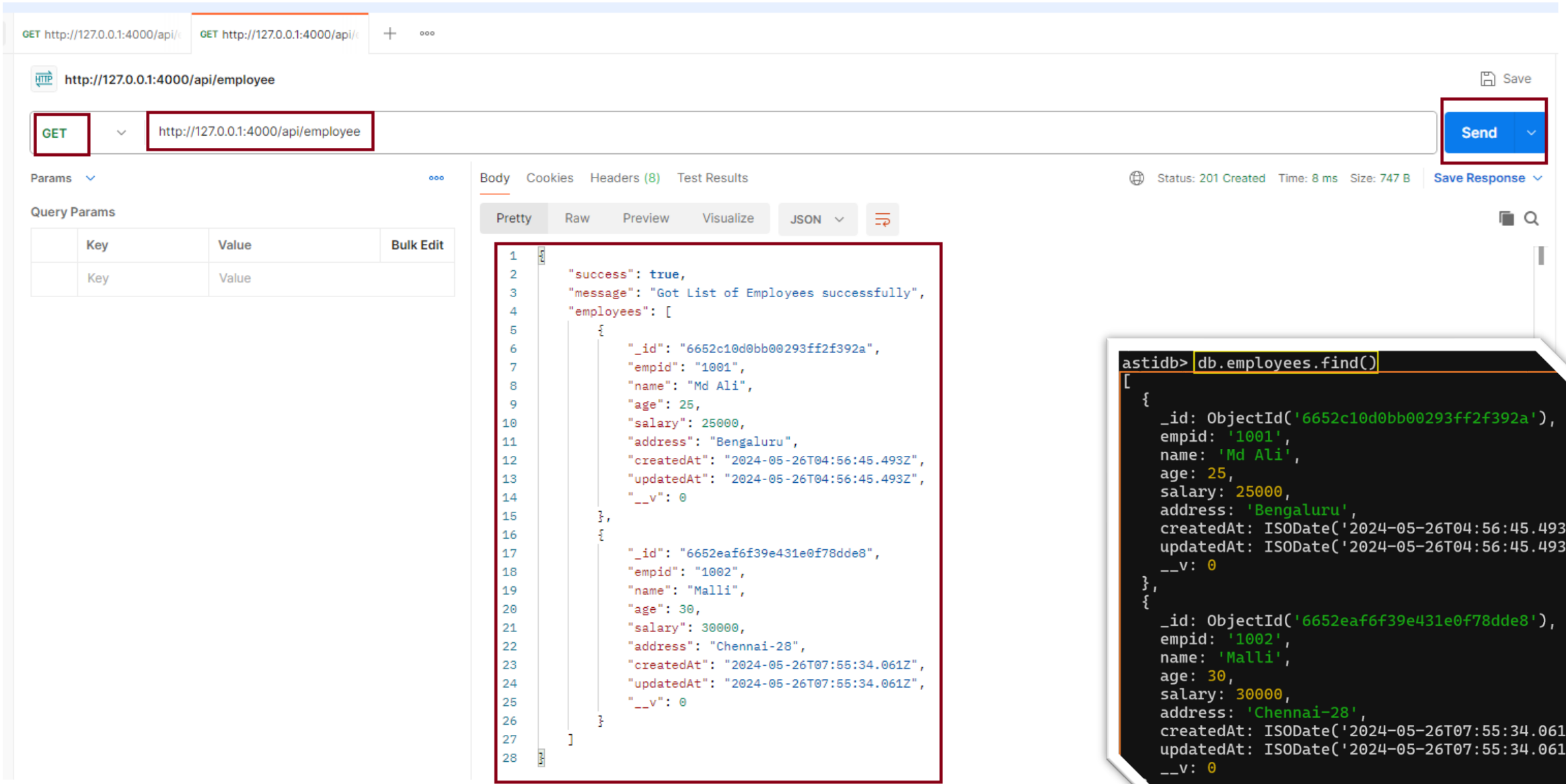
Get All Employees – Implementation – routes.js

routes.js

emp_mern_proj > emp_backend > routes > routes.js > default

```
1   import express from 'express'
2   import { CreateEmployee, GetAllEmployees } from '../controller/EmployeeController.js'
3
4   const routers = express.Router()
5
6   //Router for Create Employee
7   routers.post('/employee', CreateEmployee)
8
9   //Router for Get All Employees
10  routers.get('/employee', GetAllEmployees)
11
12  export default routers
```

Testing GET All End-Point using POSTMAN




```
astidb> db.employees.find()
[
  {
    _id: ObjectId('6652c10d0bb00293ff2f392a'),
    empid: '1001',
    name: 'Md Ali',
    age: 25,
    salary: 25000,
    address: 'Bengaluru',
    createdAt: ISODate('2024-05-26T04:56:45.493Z'),
    updatedAt: ISODate('2024-05-26T04:56:45.493Z'),
    __v: 0
  },
  {
    _id: ObjectId('6652eaf6f39e431e0f78dde8'),
    empid: '1002',
    name: 'Malli',
    age: 30,
    salary: 30000,
    address: 'Chennai-28',
    createdAt: ISODate('2024-05-26T07:55:34.061Z'),
    updatedAt: ISODate('2024-05-26T07:55:34.061Z'),
    __v: 0
```

GET By ID/PK

Employee API

GET Employee By Id/Pk – Implementation – EmployeeController.js

```
116  //Handler method for GET Employee By Id/Pk
117  const GetEmployee = async(request, response) => {
118      try {
119          const employeePk = request.params.id //Getting path variable
120          const employee = await EmployeeModel.findById(employeePk)
121
122          if(!employee) {
123              return response.status(404).json({
124                  success: false,
125                  message: "Employee Data NOT Found in Database"
126              })
127          }
128
129          // If Employee found then return employee
130          return response.status(200).json({
131              success: true,
132              message: "Got Employee successfully",
133              employee
134          })
135
136      } catch(error) {
137          console.log(error)
138          return response.status(500).json({
139              success: false,
140              message: "Unable to Get Employee due to Internal ERROR"
141          })
142      }
143  }
144
145  export {CreateEmployee, GetAllEmployees, GetEmployee, UpdateEmployee, DeleteEmployee}
```

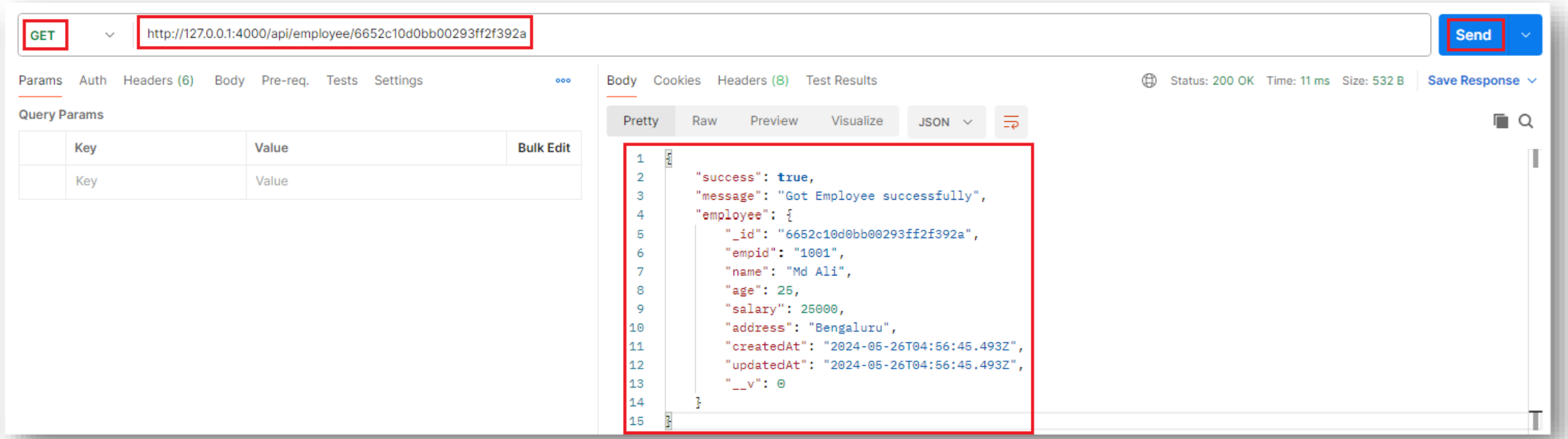
GET Employee By Id – Implementation – routes.js

routes.js

emp_mern_proj > emp_backend > routes > routes.js > ...

```
1  import express from 'express'
2  import { CreateEmployee, GetAllEmployees, GetEmployee, UpdateEmployee, DeleteEmployee } from '../controller/EmployeeController.js'
3
4  const routers = express.Router()
5
6  //Router for Create Employee
7  routers.post('/employee', CreateEmployee)
8
9  //Router for Get All Employees
10 routers.get('/employee', GetAllEmployees)
11
12 //Router for Get Employee By Id
13 routers.get('/employee/:id', GetEmployee)
14
15 //Router for Update Employee By Id
16 routers.put('/employee/:id', UpdateEmployee)
17
18 //Router for Update Employee By Id
19 routers.delete('/employee/:id', DeleteEmployee)
20
21 export default routers
```

Testing - GET Employee By ID/PK End-Point using POSTMAN




```
astidb> db.employees.find({"empid": "1001"})
[
  {
    _id: ObjectId('6652c10d0bb00293ff2f392a'),
    empid: '1001',
    name: 'Md Ali',
    age: 25,
    salary: 25000,
    address: 'Bengaluru',
    createdAt: ISODate('2024-05-26T04:56:45.493Z'),
    updatedAt: ISODate('2024-05-26T04:56:45.493Z'),
    __v: 0
  }
]
astidb>
```

UPDATE By ID/PK

Employee API

Update Employee By Id – Implementation – EmployeeController.js

```
54  //Handler method for UPDATE Employee
55  const UpdateEmployee = async(request, response) => {
56      try {
57          const employeePk = request.params.id //Getting path variable
58
59          const updatedEmployee = await EmployeeModel.findByIdAndUpdate(
60              employeePk,
61              request.body,
62              {new: true}
63          )
64
65          if(!updatedEmployee) {
66              return response.status(404).json({
67                  success: false,
68                  message: "Employee Data NOT Found in Database for UPDATE"
69              })
70          }
71
72          response.status(200).json({
73              success: true,
74              message: "Employee updated successfully",
75              updatedEmployee
76          })
77
78      }catch(error) {
79          console.log(error)
80          response.status(500).json({
81              success: false,
82              message: "Employee NOT Updated due to Internal ERROR"
83          })
84      }
85  }
86
87  export {CreateEmployee, GetAllEmployees, UpdateEmployee}
```

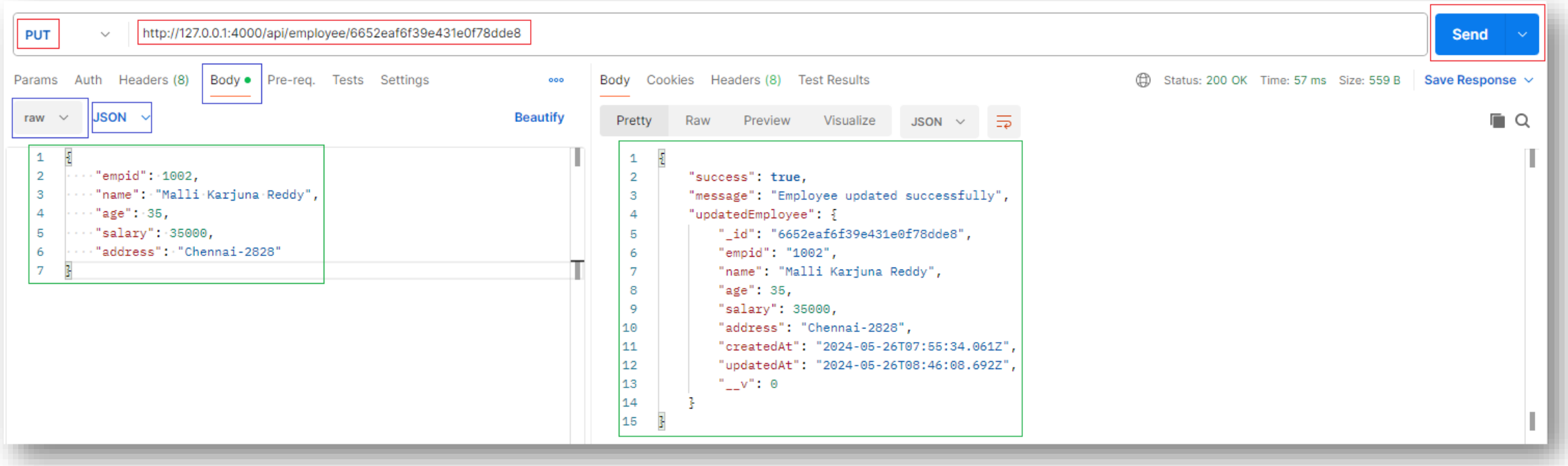
Update Employee By Id – Implementation – routes.js

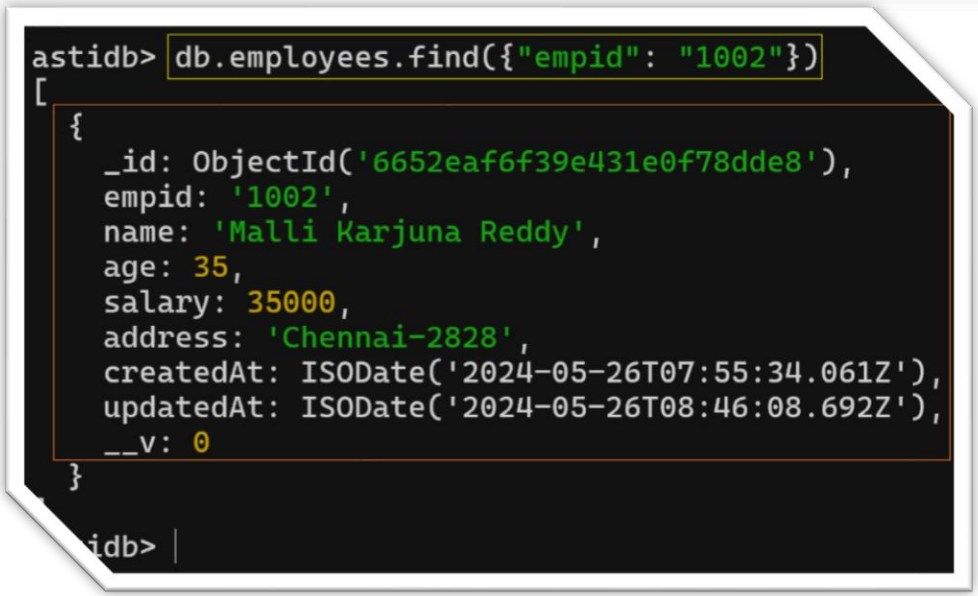
routes.js

emp_mern_proj > emp_backend > routes > routes.js > default

```
1   import express from 'express'
2   import { CreateEmployee, GetAllEmployees, UpdateEmployee } from '../controller/EmployeeController.js'
3
4   const routers = express.Router()
5
6   //Router for Create Employee
7   routers.post('/employee', CreateEmployee)
8
9   //Router for Get All Employees
10  routers.get('/employee', GetAllEmployees)
11
12  //Router for Update Employee By Id
13  routers.put('/employee/:id', UpdateEmployee)
14
15  export default routers
```

Testing - Update Employee By PK End-Point using POSTMAN





DELETE By ID/PK

Employee API

Delete Employee By Id – Implementation – EmployeeController.js

EmployeeController.js

emp_mern_proj > emp_backend > controller > EmployeeController.js > ...

```
87  //Handler method for Delete Employee By Id/Pk
88  const DeleteEmployee = async(request, response) => {
89      try {
90          const employeePk = request.params.id //Getting path variable
91
92          const deletedEmployee = await EmployeeModel.findByIdAndDelete(employeePk)
93
94          if(!deletedEmployee) {
95              return response.status(404).json({
96                  success: false,
97                  message: "Employee Data NOT Found in Database for DELETE"
98              })
99          }
100
101         response.status(200).json({
102             success: true,
103             message: "Employee deleted successfully",
104             deletedEmployee
105         })
106
107     } catch(error) {
108         console.log(error)
109         response.status(500).json({
110             success: false,
111             message: "Employee NOT Deleted due to Internal ERROR"
112         })
113     }
114 }
115
116 export {CreateEmployee, GetAllEmployees, UpdateEmployee, DeleteEmployee}
```

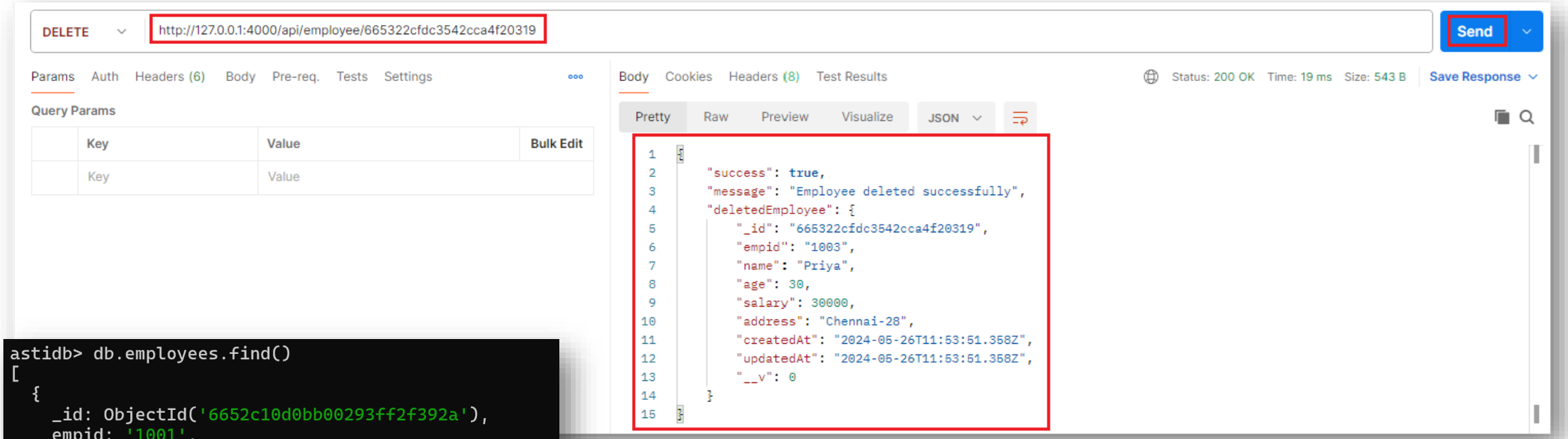
Delete Employee By Id – Implementation – routes.js

routes.js

emp_mern_proj > emp_backend > routes > routes.js > ...

```
1  import express from 'express'
2  import { CreateEmployee, GetAllEmployees, UpdateEmployee, DeleteEmployee } from '../controller/EmployeeController.js'
3
4  const routers = express.Router()
5
6  //Router for Create Employee
7  routers.post('/employee', CreateEmployee)
8
9  //Router for Get All Employees
10 routers.get('/employee', GetAllEmployees)
11
12 //Router for Update Employee By Id
13 routers.put('/employee/:id', UpdateEmployee)
14
15 //Router for Update Employee By Id
16 routers.delete('/employee/:id', DeleteEmployee)
17
18 export default routers
```

Testing - Delete Employee By PK End-Point using POSTMAN




```
astidb> db.employees.find()
[
  {
    _id: ObjectId('6652c10d0bb00293ff2f392a'),
    empid: '1001',
    name: 'Md Ali',
    age: 25,
    salary: 25000,
    address: 'Bengaluru',
    createdAt: ISODate('2024-05-26T04:56:45.493Z'),
    updatedAt: ISODate('2024-05-26T04:56:45.493Z'),
    __v: 0
  },
  {
    _id: ObjectId('6652eaf6f39e431e0f78dde8'),
    empid: '1002',
    name: 'Malli Karjuna Reddy',
    age: 35,
    salary: 35000,
    address: 'Chennai-2828',
    createdAt: ISODate('2024-05-26T07:55:34.061Z'),
    updatedAt: ISODate('2024-05-26T08:46:08.692Z'),
    __v: 0
  }
]
```

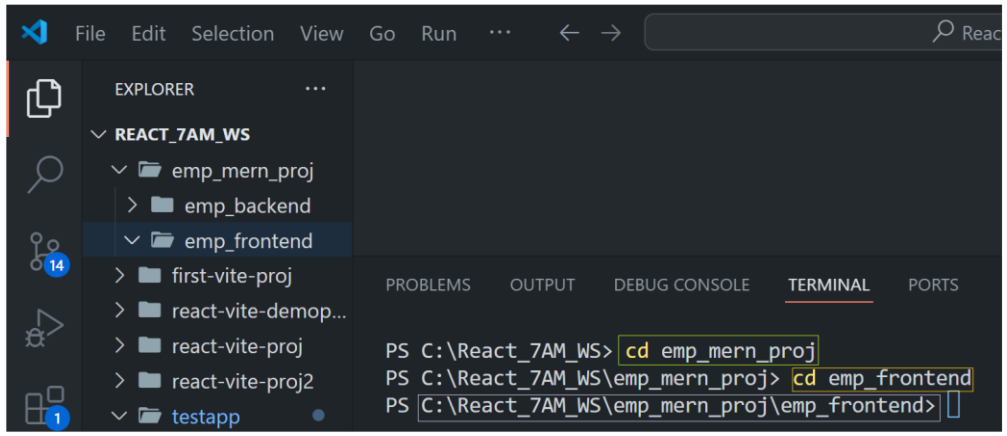
```
astidb> db.employees.find({"empid": "1003"})

astidb>
```

Frontend

Implementation

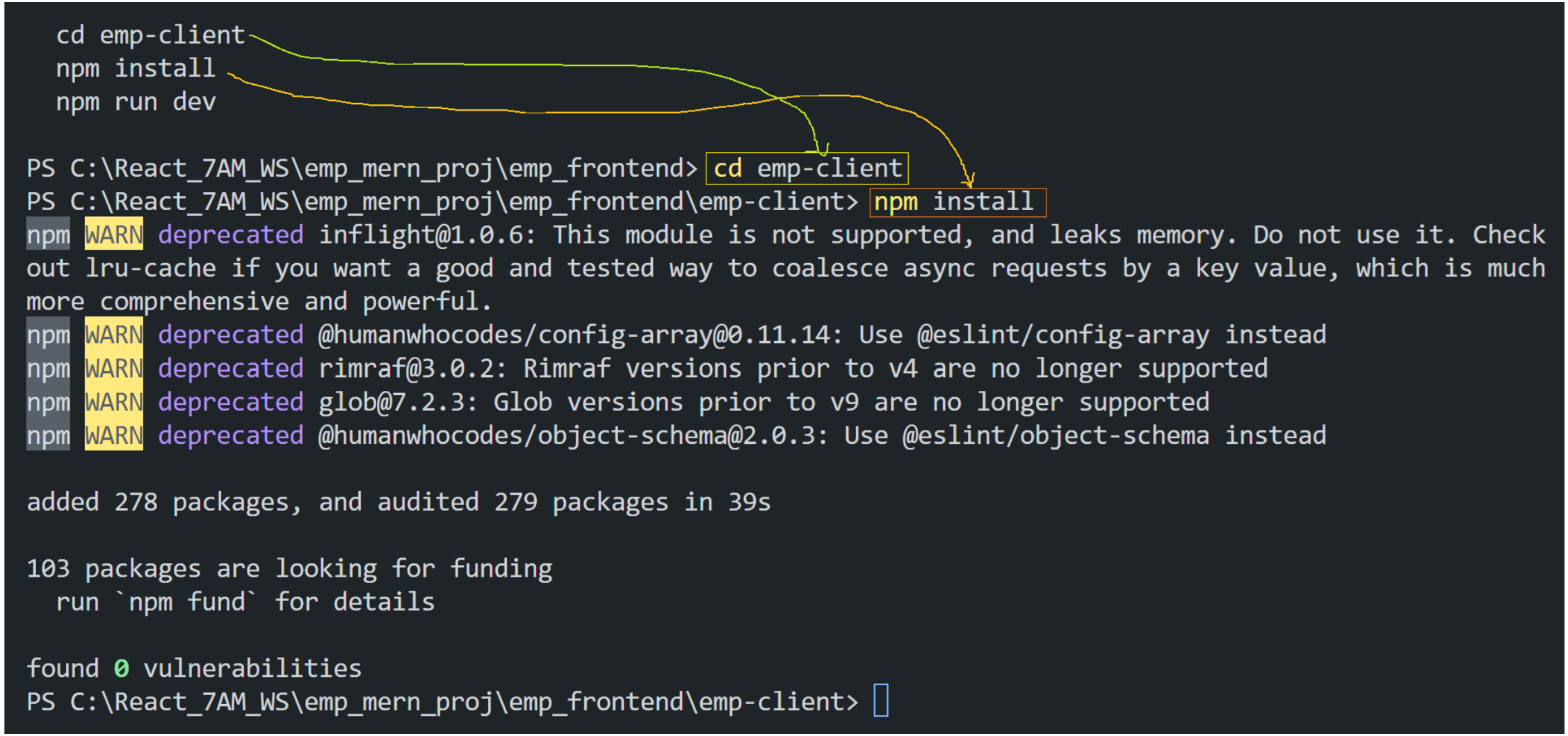
Create React Application

- ✓ From Terminal go to emp_frontend folder



- ✓ Create React Application using **vite** command

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_frontend> npm create vite@latest
√ Project name: ... emp-client
√ Select a framework: » React
√ Select a variant: » JavaScript

Scaffolding project in C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client...

Done. Now run:

  cd emp-client
  npm install
  npm run dev

PS C:\React_7AM_WS\emp_mern_proj\emp_frontend>
```


cd emp-client
npm install
npm run dev
PS C:\React_7AM_WS\emp_mern_proj\emp_frontend> cd emp-client
PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client> npm install
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm WARN deprecated @humanwhocodes/config-array@0.11.14: Use @eslint/config-array instead
npm WARN deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated @humanwhocodes/object-schema@2.0.3: Use @eslint/object-schema instead
added 278 packages, and audited 279 packages in 39s
103 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client>

Installing Required Software/packages

- ✓ Install Axios

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client> npm i axios

added 9 packages, and audited 288 packages in 1s

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client>
```

- ✓ Install Bootstrap

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client> npm i bootstrap

added 2 packages, and audited 290 packages in 4s

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client>
```

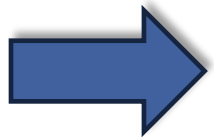
Installing Required Software/packages …

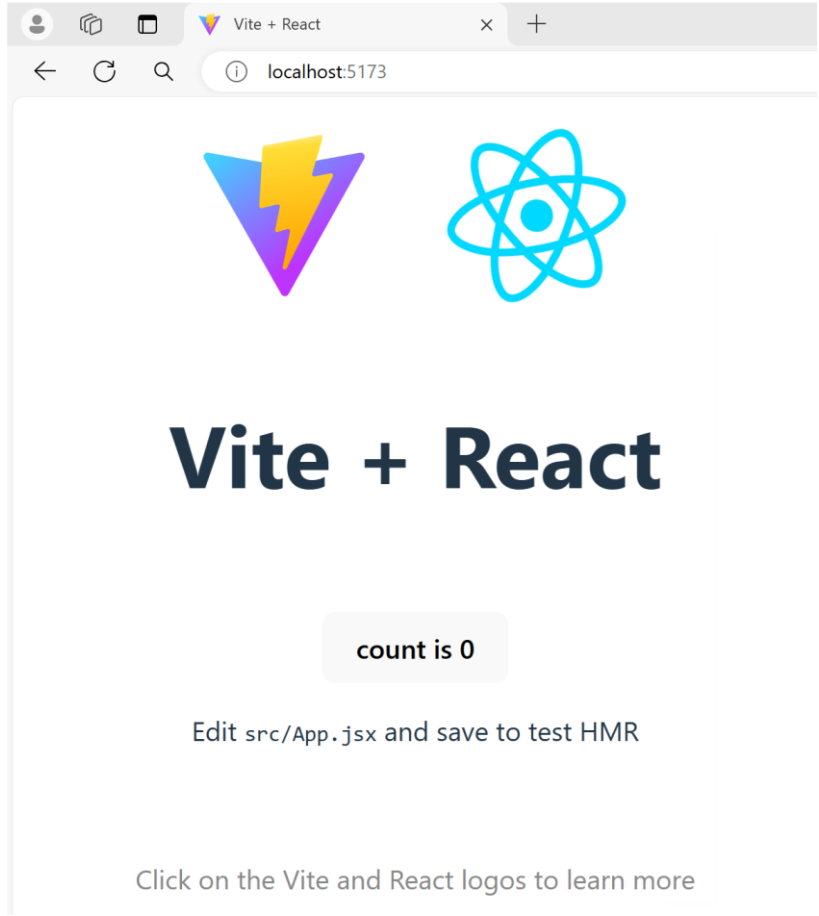
- ✓ Install **Hot toast** For adding Beautiful Notifications to React Application

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client> npm i react-hot-toast

added 2 packages, and audited 292 packages in 3s

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

found 0 vulnerabilities
PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client>
```

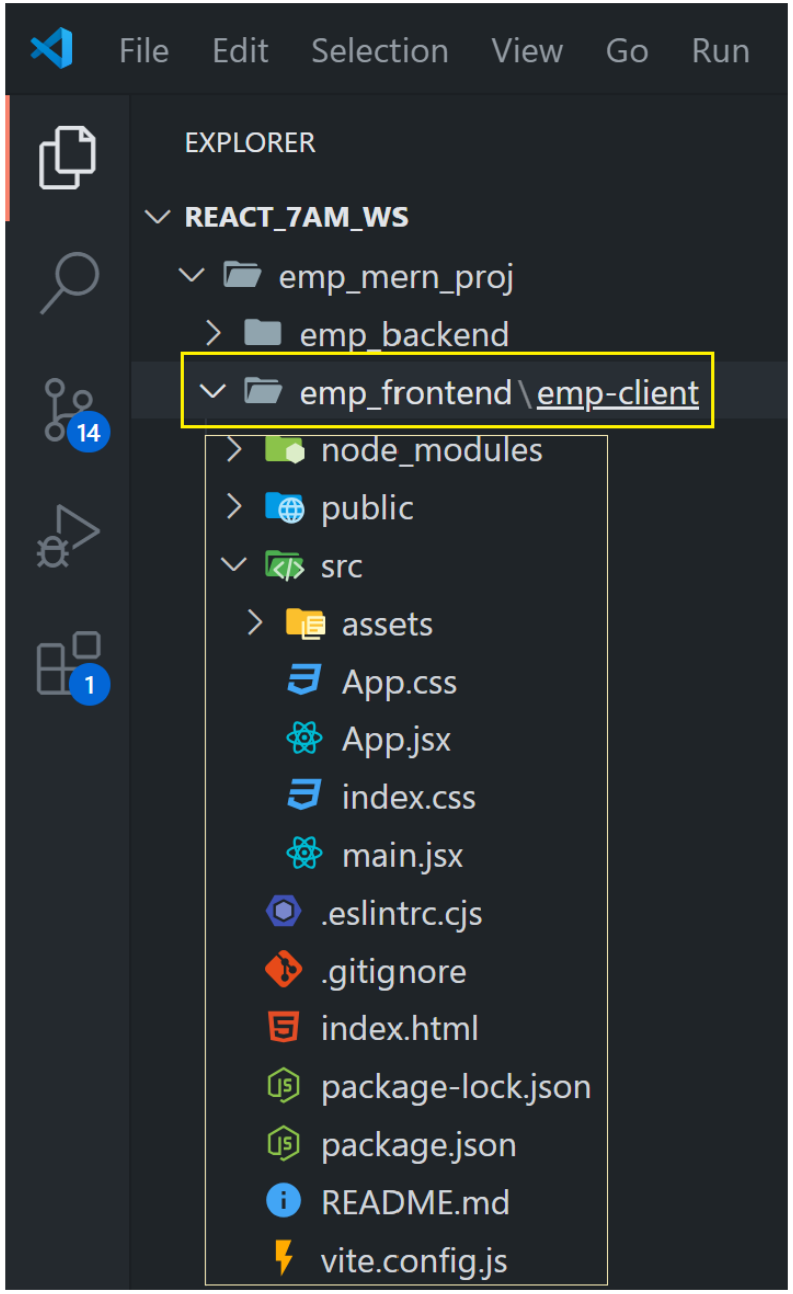
- ✓ Run the client application

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\React_7AM_WS\emp_mern_proj\emp_frontend\emp-client> npm run dev

> emp-client@0.0.0 dev
> vite


  VITE v5.3.1  ready in 238 ms

  ➜  Local:   http://localhost:5173/
  ➜  Network: use --host to expose
  ➜  press h + enter to show help
```



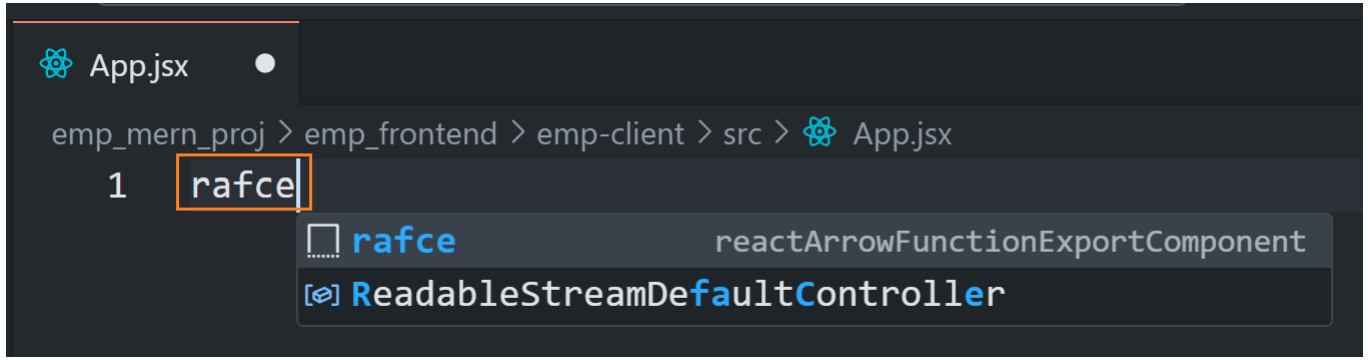


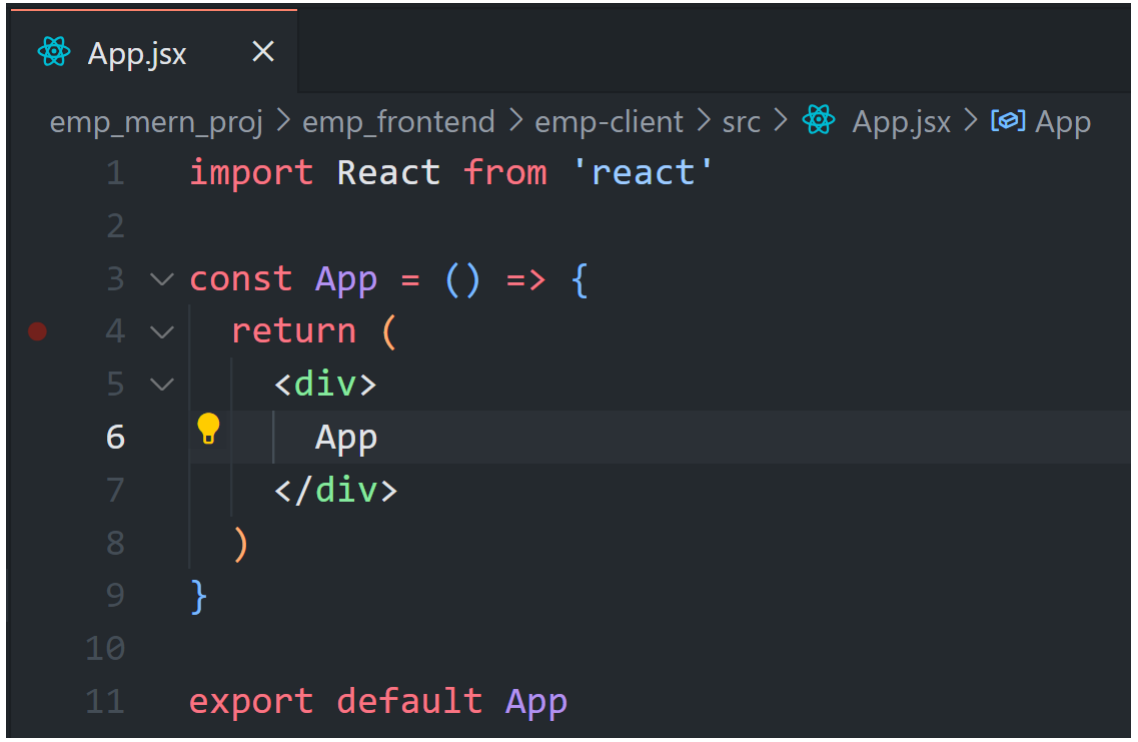
React Application Folder Structure



Existing React Application Cleanup

- ✓ Delete existing code from **App.jsx &** Create a function





✓ Delete import index.css from **main.jsx**

main.jsx

emp_mern_proj > emp_frontend > emp-client > src > main.jsx

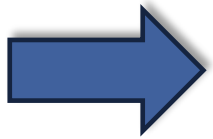
```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

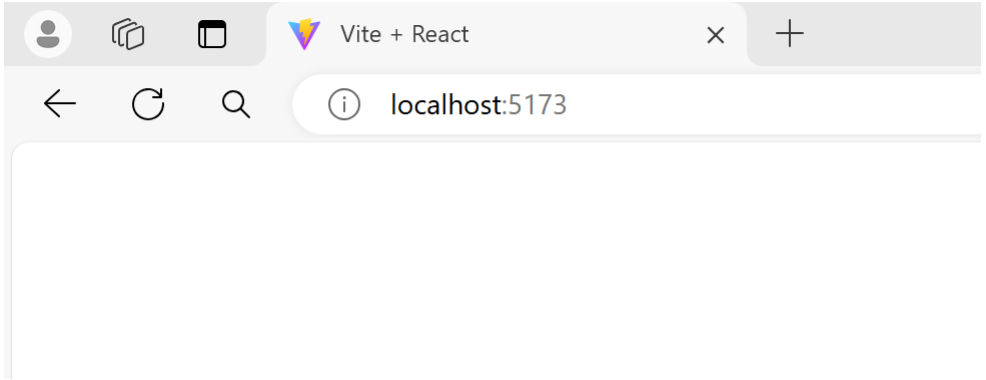
✓ Delete all code from **App.css**

App.css

emp_mern_proj > emp_frontend > emp-client > src > App.css

```
1
```





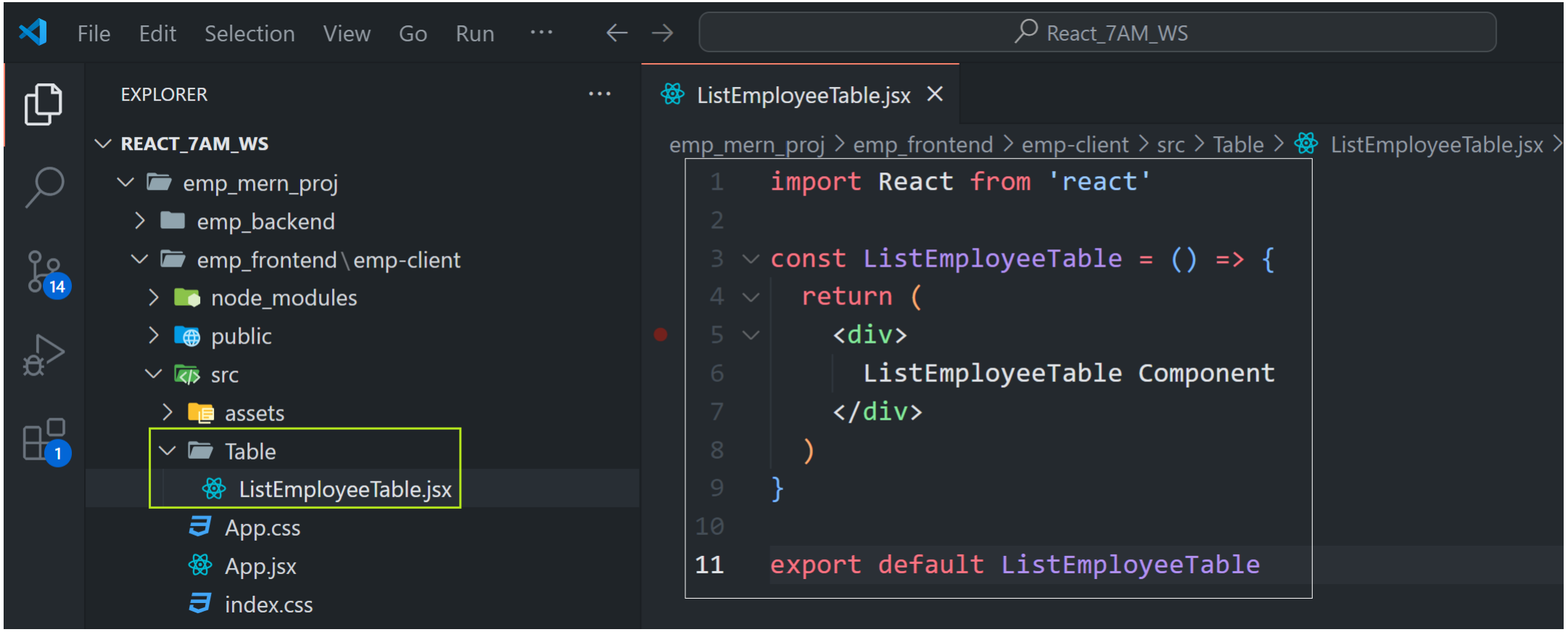
✓ Integrating Bootstrap into React application – **App.jsx**

App.jsx

emp_mern_proj > emp_frontend > emp-client > src > App.jsx > App

```
1  import React from 'react'
2  import 'bootstrap/dist/css/bootstrap.min.css';
3  import 'bootstrap/dist/js/bootstrap.bundle.min.js';
4
5  const App = () => {
6    return (
7      <div>
8        App
9      </div>
10   )
11 }
12
13 export default App
```

- ✓ Create a new folder called "**Table**" in "**src**" folder
- ✓ Create react component called "**ListEmployeeTable.jsx**" inside "**Table**" folder
- ✓ Create **rafce**



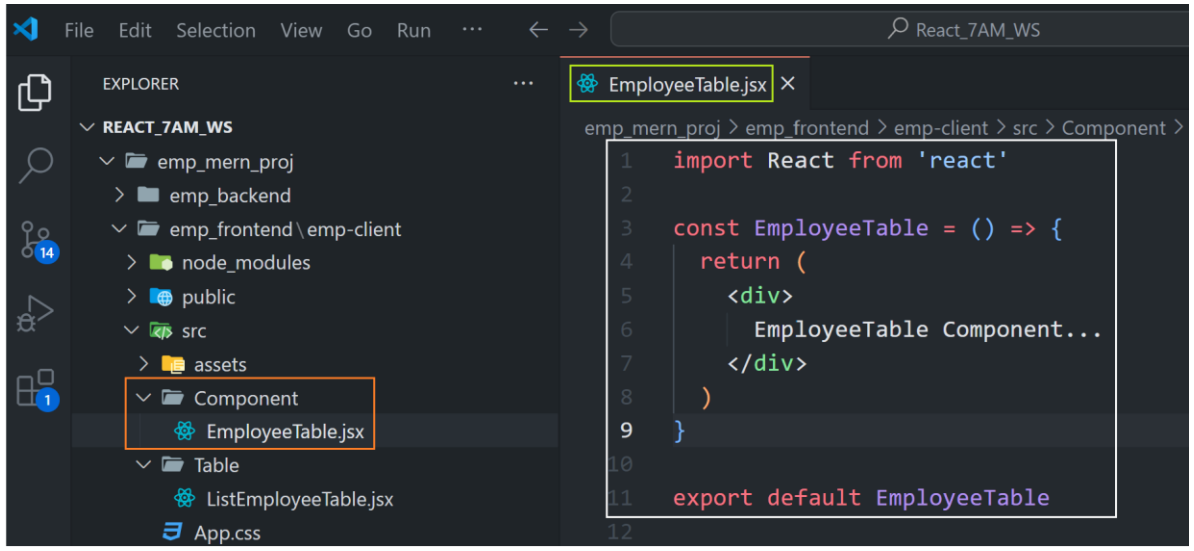
✓ Call ListEmployeeTable.jsx component from App.jsx

App.jsx

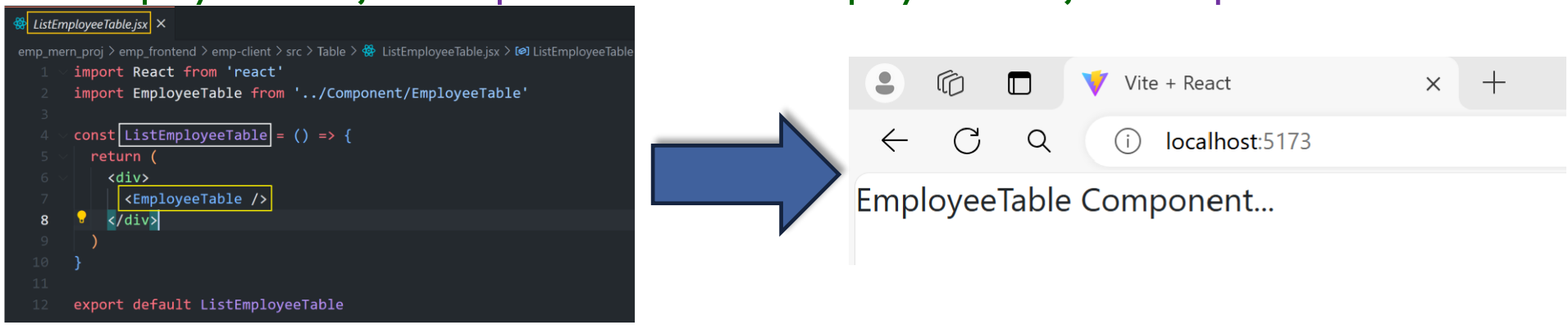
emp_mern_proj > emp_frontend > emp-client > src > App.jsx > App

```jsx
import React from 'react'
import 'bootstrap/dist/css/bootstrap.min.css';
import 'bootstrap/dist/js/bootstrap.bundle.min.js';
import ListEmployeeTable from './Table/ListEmployeeTable';
import {Toaster} from 'react-hot-toast'

const App = () => {
  return (
    <div>
      <Toaster></Toaster>
      <ListEmployeeTable />
    </div>
  )
}
```

- ✓ Create a new folder called "**Component**" in "**src**" folder
- ✓ Create react component called "**EmployeeTable.jsx**" inside "**Component**" folder
- ✓ Create **rafce**

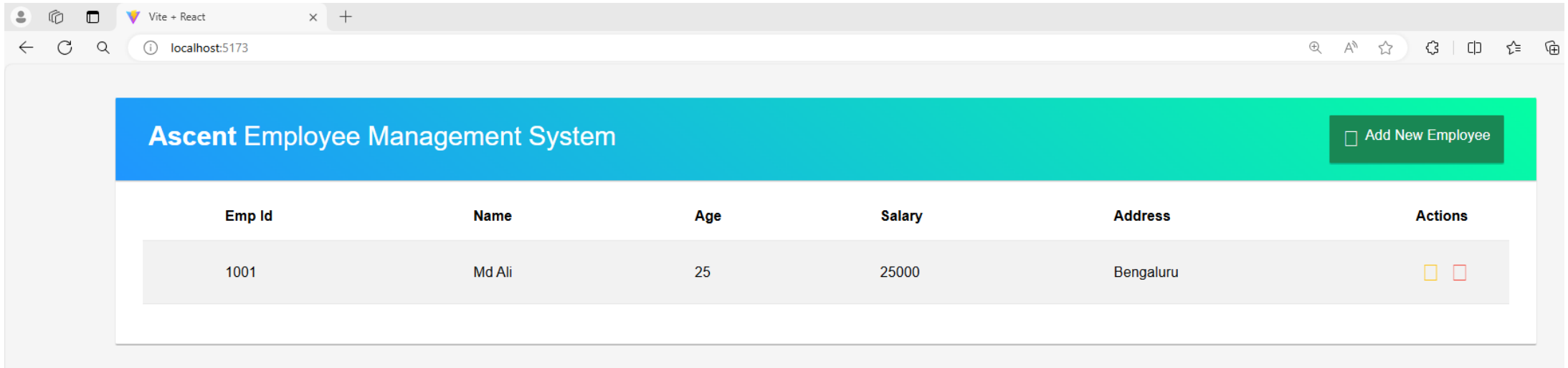


- ✓ Call EmployeeTable.jsx component from ListEmployeeTable.jsx Component

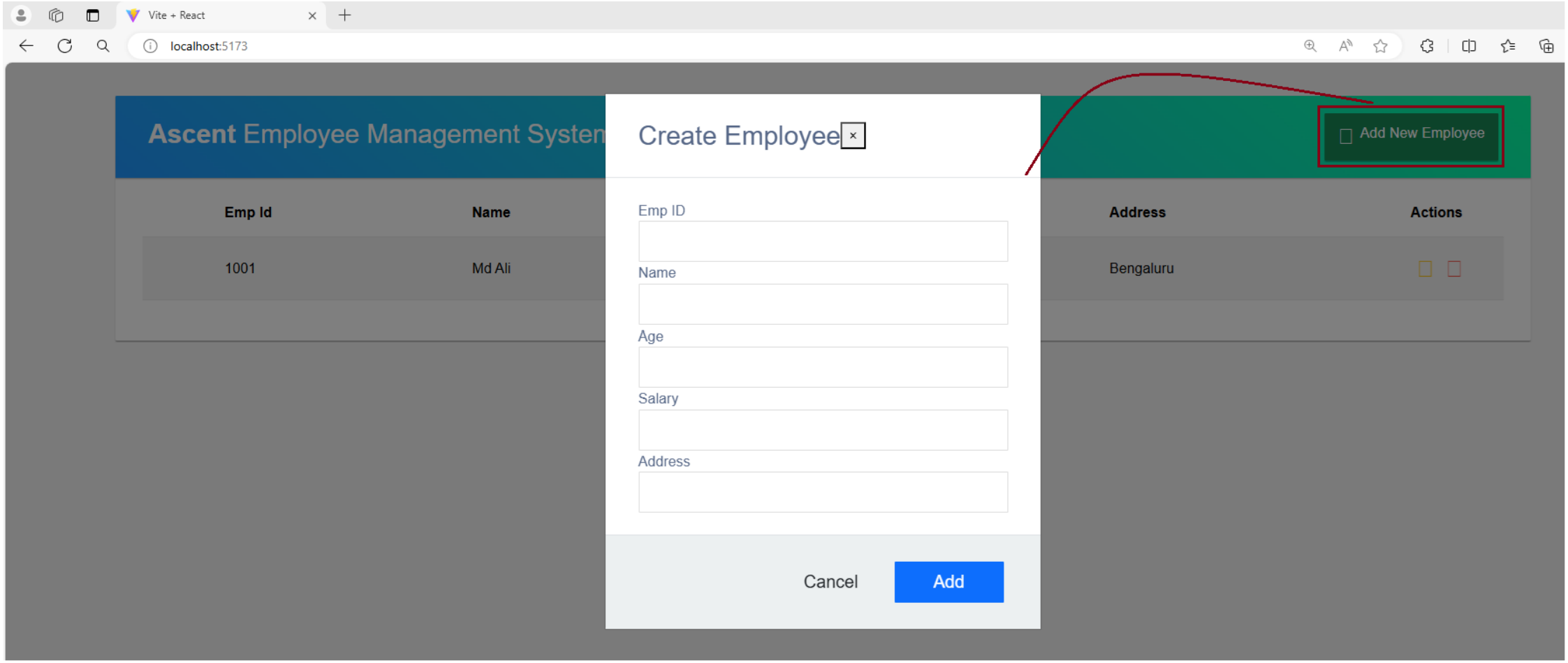


Adding JSX code in List Employee

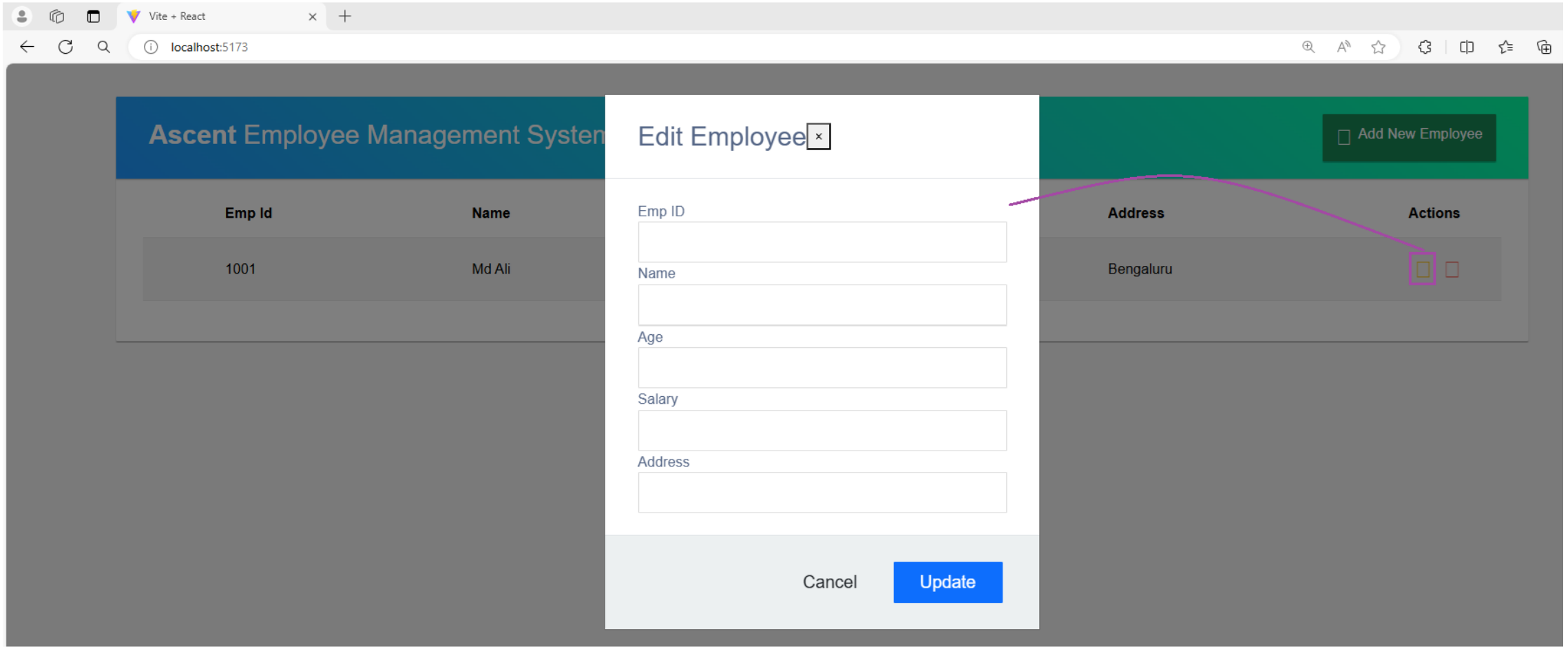
- ✓ Add JSX code in "**EmployeeTable.jsx**" for listing existing Employees in DB
- ✓ Add CSS Code in App.css


Vite + React
localhost:5173
Ascent Employee Management System
Add New Employee
Emp Id
Name
Age
Salary
Address
Actions
1001
Md Ali
25
25000
Bengaluru

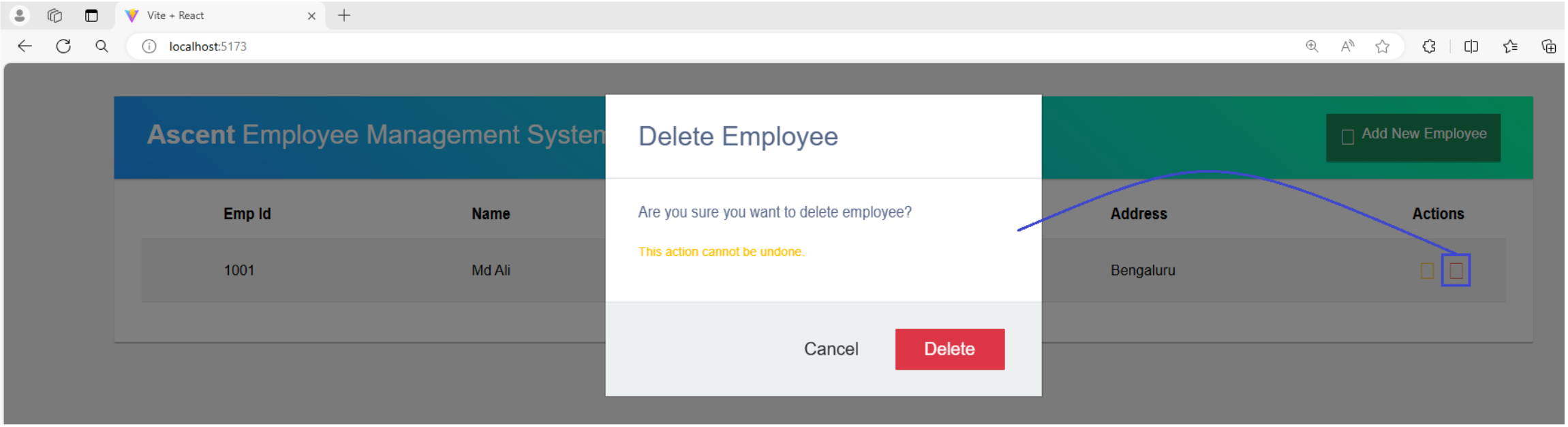
Design Create Employee Form



Design Update Employee Form



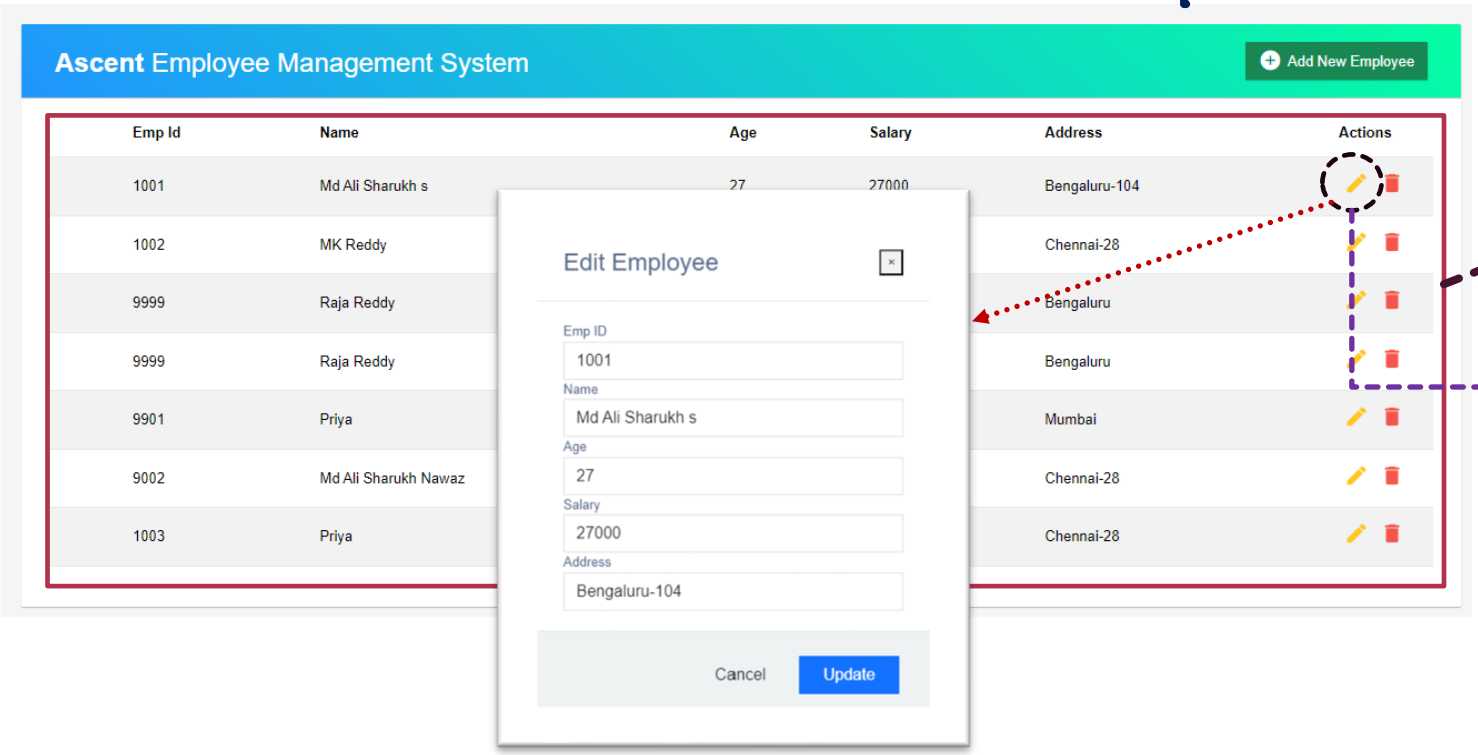
Design Delete Employee Form



Update Employee

Update Functionality

ListEmployeeTable.js



EmployeeTable.js

- Two Activities has to be done
  - Need to pass id which is going to update
  - Need to pass employee object
- From EmployeeTable.js, When we include EmployeeTable Component we need to pass a function called UpdateEmployee through which we can send id, employee object
- From EmployeeTable, when click on update icon will call UpdateEmployee Function with **id** so that this function will call get employee end point to get employee from DB
- From ListEmployeeTable we include EditEmployee component and we pass handleOnSubmit(), employee obj, handleChange()
- handleChange() will get latest employee values
- When user clicks on update button handleOnSubmit() will call update end-point to update the employee details in DB.