

20/04/2023

File structure → to keep things modular

- \* index.js / server.js ↗ mapping
- \* Routers folder

todos.js

↳ /createTodo

↓ Route is hit and each route is mapped to a controller which contain some business logic.

Controller will create entry in the database.

- \* models

↳ todo object  
(schema)

Schema is basically structure of the data.

Note → Controller will have logic.  
Model will have structure.

Also controller will be needing the model.

- \* config

↳ database.js

↳ Have some function which has work of connection via mongoose.

↗ port number

Note → app.listen(3000);

This is a bad practice. We will be creating .env file which have port number & URL already defined.

nodemon

This will automatically restart the server & we don't have write node index.js again & again.

In package.json

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

Note → npm run dev → To start server via nodemon

npm i nodemon → Command to install nodemon

.env file

PORT = 3000

DATABASE\_URL = mongo ://127.0.0.1:  
27017/bhavyaDB

loopback  
address & is  
localhost

config folder → database.js

First install dot env via npm i dotenv

```
const mongoose = require('mongoose');
require("dotenv").config();
const dbConnect = () => {
  // Steps of mongoose connection
}
```

}

```
module.exports = dbConnect;
```

models → Todo.js

Creation of schema

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

```
const todoSchema = new mongoose.Schema
```

```
(
```

```
{
```

```
title : {
```

```
type : String,
```

```
required : true,
```

```
maxLength : 50,
```

```
}
```

```
3
```

```
);
```

```
module.exports = mongoose.model ("Todo",  
todoSchema);
```

Note → default : Date.now()

↳ If value not entered

controller → create Todo.js

1) Import the model

```
const Todo = require ("../models/Todo");
```

2) Now if we have come to the controller,  
we have most probably hit the route. So  
now we need to create the route handler.

```
exports.createTodo = async (req, res) =>
```

```
{  
  try {  
    //Extract title and description from req body  
    const {title,description} = req.body;  
    //Create a new Todo object & insert in db  
    const response = await Todo.create({title,  
      description});  
  
    res.status(200).json(  
      {  
        success: true,  
        data: response,  
        message: "Entry done"  
      }  
    );  
  }  
  catch (err) {  
    console.error(err);  
    console.log(err);  
    res.status(500).json({  
      success: false,  
      data: "internal server error",  
      message: err.message,  
    });  
  }  
}
```

Routes → todo.js

```
const express = require ("express");  
const router = express.Router();  
//Creating route  
const {createTodo} = require("../controller/create  
Todo");
```

```
// Define API route → post request say
router.post("/createTodo", createTodo);

module.exports = router;

index.js
const express = require("express");
const app = express();
// Load config file from .env
require("dotenv").config();
// Port does not load, then 4000 is used
const PORT = process.env.PORT || 4000;
// middleware → for parsing json req body
app.use(express.json());
// import routes for todo API
import todoRoutes = require("./routes/todo");
// mounting routes
app.use("/app/v1", todoRoutes);
// start server
app.listen(PORT, () => {
    console.log("Started");
});

// connect to database
const dbConnect = require("./config/database");
dbConnect();

// Default route
app.get("/", (req, res) => {
    res.send('<h1> Homepage </h1>');
})
```

The mounting is done so as to increase the

classmate

Date \_\_\_\_\_

Page \_\_\_\_\_

readability and also do version

VI → version ±