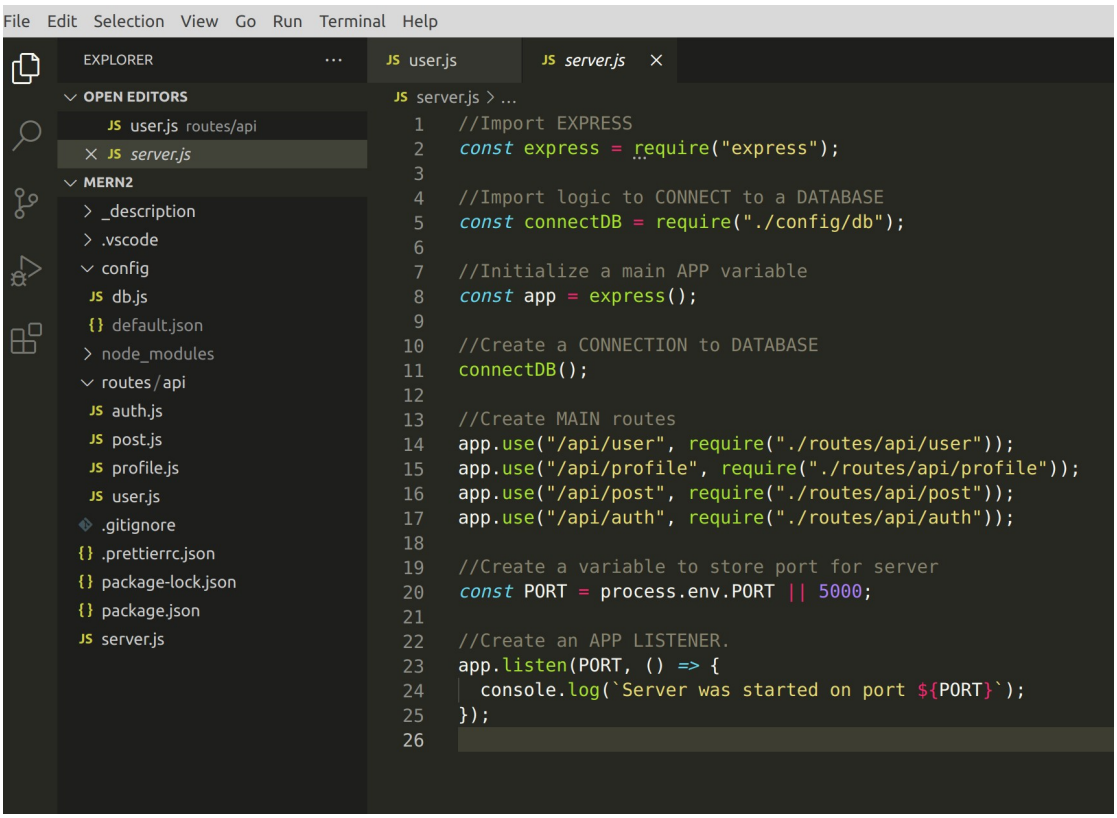

File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
user.js routes/api
server.js
MERN2
_description
.vscode
config
db.js
default.json
node_modules
routes / api
auth.js
post.js
profile.js
user.js
.gitignore
.prettierrc.json
package-lock.json
package.json
server.js
user.js
server.js
server.js > ...
1 //Import EXPRESS
2 const express = require("express");
3
4 //Import logic to CONNECT to a DATABASE
5 const connectDB = require("./config/db");
6
7 //Initialize a main APP variable
8 const app = express();
9
10 //Create a CONNECTION to DATABASE
11 connectDB();
12
13 //Create MAIN routes
14 app.use("/api/user", require("./routes/api/user"));
15 app.use("/api/profile", require("./routes/api/profile"));
16 app.use("/api/post", require("./routes/api/post"));
17 app.use("/api/auth", require("./routes/api/auth"));
18
19 //Create a variable to store port for server
20 const PORT = process.env.PORT || 5000;
21
22 //Create an APP LISTENER.
23 app.listen(PORT, () => {
24 console.log(`Server was started on port ${PORT}`);
25 });
26

# Table of Contents

# 1. Introduction.

There will be a general information about project, MERN stack and all side tecnologies

## 1.x VSCode

### 1.x.1 Prettier set up

https://glebbahmutov.com/blog/configure-prettier-in-vscode/#settings

**VSCode setup** #

To use the Prettier we have just installed from VSCode we need to install the Prettier VSCode extension:

1. Launch VS Code Quick Open (Ctrl+P)
2. Run the following command

```
ext install esbenp.prettier-vscode
```

Because you might have global settings related to code formatting, I prefer having in each repository a file with local workspace VSCode settings. I commit this file `.vscode/settings.json` to source control to make sure everyone uses the same extension to format the code.

.vscode/settings.json

```
{
  "editor.defaultFormatter": "esbenp.prettier-vscode",
  "editor.formatOnSave": true
}
```

Now every time we save a JavaScript file, it will be formatted using Prettier automatically. Here is me formatting `projectA/index.js`

# 2. Express & MongoDB Setup

## 2.1. MongoDB Atlas Setup

### 2.1.1 Create an account at MongoDB.com

### 2.1.2 Log in at MongoDB.com

### 2.1.3 Create a new project at mongoDB.com

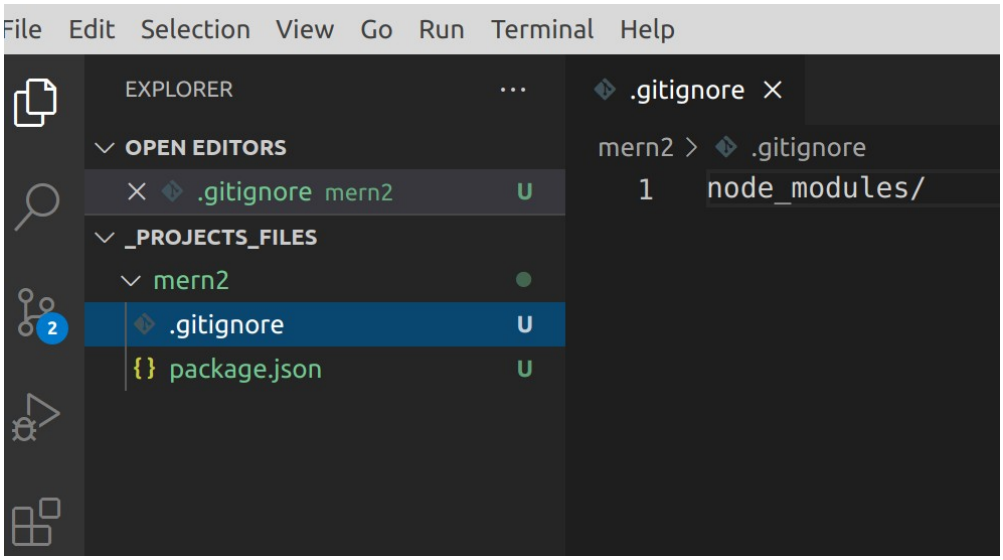
### 2.1.4 Create a new cluster

### 2.1.5 Create a new user

### 2.1.6 Change whitelist of IP addresses

## 2.2 Install Dependencies & Basic Express Setup

### 2.2.1 Create a file .gitignore. Add to that file folder node_modules/




### 2.2.2 Create GIT repository with [CLI=> git init]

### 2.2.3 Create NPM with [CLI=> npm init]

### 2.2.4 Install all the regular dependencies

[CLI=> npm i express express-validator bcryptjs config gravatar jsonwebtoken mongoose request]

### 2.2.5 Install all the developer dependencies

[CLI=> npm i -D nodemon concurrently]

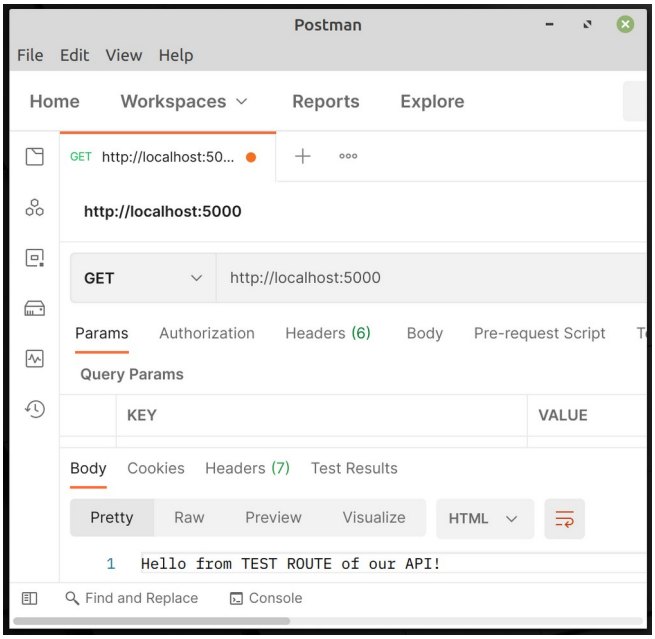
### 2.2.6 Create a main entry file – server.js

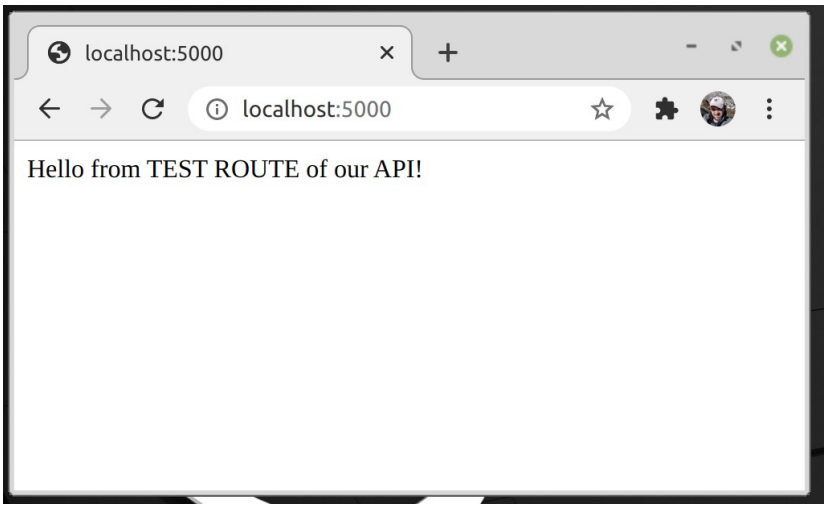
[CLI=> touch server.js]

### 2.2.7 Change start scripts at package.json


Now to start server: [CLI=> npm run server]. It will start SERVER.JS file with NODEMON

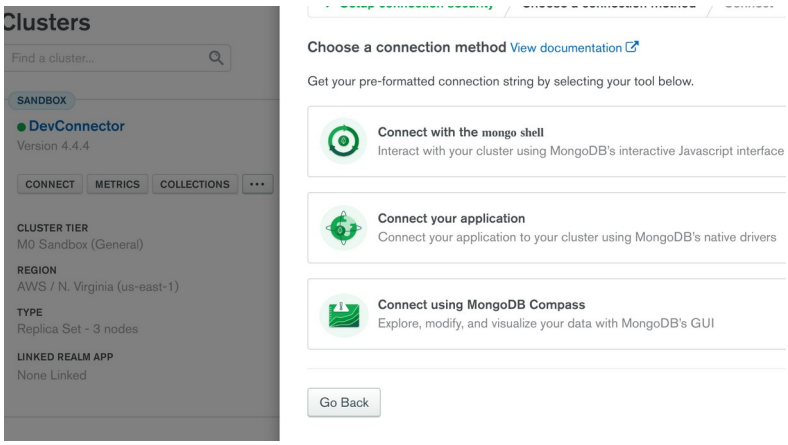
### 2.2.8 Create a SERVER with simple test route



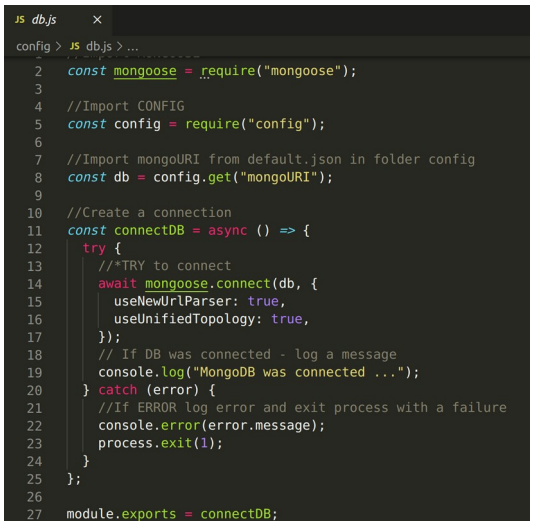




# 2.3 Connecting To MongoDB With Mongoose

## 2.3.1 Create a connection to database




## 2.3.2 Create a connection logic on server



```
server.js
server.js > ...
1   //Import EXPRESS
2   const express = require("express");
3
4   //Import logic to CONNECT to a DATABASE
5   const connectDB = require("./config/db");
6
7   //Initialize a main APP variable
8   const app = express();
9
10  //Create a CONNECTION to DATABASE
11  connectDB();
12
13  //Create a TEST ROUTE (for GET request to root folder '/')
14  app.get("/", (req, res) => res.send("Hello from TEST ROUTE of our API!"));
15
16  //Create a variable to store port for server
17  const PORT = process.env.PORT || 5000;
18
19  //Create an APP LISTENER.
20  app.listen(PORT, () => {
21    console.log(`Server was started on port ${PORT}`);
22  });
23
```

```
stslon@stslon-System-Product-Name:/media/stslon/8CB04F45B04F354C/
> mern-devconnector@1.0.0 server
> nodemon server

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js`
(node:24960) Warning: Accessing non-existent property 'MongoError
(Use `node --trace-warnings ...` to show where the warning was cr
Server was started on port 5000
(node:24960) DeprecationWarning: Listening to events on the Db cl
MongoDB was connected ...
```

## 2.4 Create route files with Express Router

```
server.js
server.js > ...
1   //Import EXPRESS
2   const express = require("express");
3
4   //Import logic to CONNECT to a DATABASE
5   const connectDB = require("./config/db");
6
7   //Initialize a main APP variable
8   const app = express();
9
10  //Create a CONNECTION to DATABASE
11  connectDB();
12
13  //Create MAIN routes
14  app.use("/api/user", require("./routes/api/user"));
15  app.use("/api/profile", require("./routes/api/profile"));
16  app.use("/api/post", require("./routes/api/post"));
17  app.use("/api/auth", require("./routes/api/auth"));
18
19  //Create a variable to store port for server
20  const PORT = process.env.PORT || 5000;
21
22  //Create an APP LISTENER.
23  app.listen(PORT, () => {
24    console.log(`Server was started on port ${PORT}`);
25  });
26
```

```
EXPLORER
OPEN EDITORS
  user.js routes/api U
MERN2
  _description
  .vscode
  config
    db.js
    default.json
  node_modules
  routes/api
    auth.js U
    post.js U
    profile.js U
    user.js U
  .gitignore
  .prettierrc.json
  package-lock.json
  package.json
  server.js M
```

```
user.js
routes > api > user.js > ...
1   //Import EXPRESS
2   const express = require("express");
3
4   //Import EXPRESS ROUTER
5   const router = express.Router();
6
7   router.get("/login", (req, res) => {
8     res.send("hello from api/user/login");
9   });
10
11  router.get("/", (req, res) => {
12    res.send("hello from api/user");
13  });
14
15  module.exports = router;
16
```

# 3. User API Routes & JWT Authentication

## 3.1 Create USER MODEL (SCHEMA)

User.js

models > User.js > ...

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

const UserSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true,
  },
  email: {
    type: String,
    unique: true,
    required: true,
  },
  password: {
    type: String,
    required: true,
  },
  avatar: {
    type: String,
  },
  date: {
    type: Date,
    default: Date.now,
  },
});

module.exports = User = mongoose.model("user", UserSchema);
```