

This is a placeholder for a title page

Table of Contents

- This is a placeholder for a title page.....1
- 1. Introduction.....3
 - 1.x VSCode.....3
 - 1.x.1 Prettier set up.....3
- 2. Express & MongoDB Setup.....4
 - 2.1. MongoDB Atlas Setup.....4
 - 2.1.1 Create an account at MongoDB.com.....4
 - 2.1.2 Log in at MongoDB.com.....4
 - 2.1.3 Create a new project at mongoDB.com.....4
 - 2.1.4 Create a new cluster.....4
 - 2.1.5 Create a new user.....4
 - 2.1.6 Change whitelist of IP addresses.....4
 - 2.1.7 Create a connection to database.....4
 - 2.2 Install Dependencies & Basic Express Setup.....4
 - 2.2.1 Create a file .gitignore. Add to that file folder node_modules/.....4
 - 2.2.2 Create GIT repository with [CLI=> git init].....4
 - 2.2.3 Create NPM with [CLI=> npm init].....4
 - 2.2.4 Install all the regular dependencies.....5
 - 2.2.5 Install all the developer dependencies.....5
 - 2.2.6 Create a main entry file – server.js.....5
 - 2.2.7 Change start scripts at package.json.....5
 - 2.2.8 Create a SERVER with simple test route.....5
- 3. User API Routes & JWT Authentication.....7

1. Introduction.

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

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

```
1 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`

```
1 {
2   "editor.defaultFormatter": "esbenp.prettier-vscode",
3   "editor.formatOnSave": true
4 }
```

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

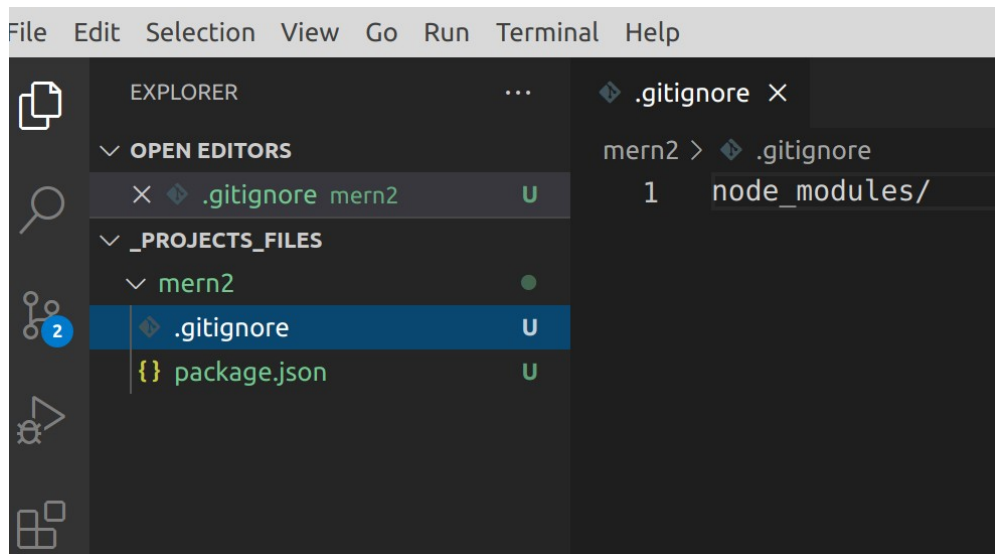
2.1.5 Create a new user

2.1.6 Change whitelist of IP addresses

2.1.7 Create a connection to database

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]

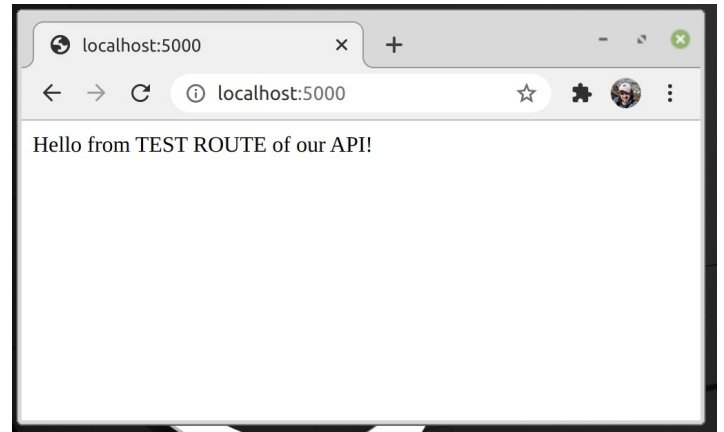
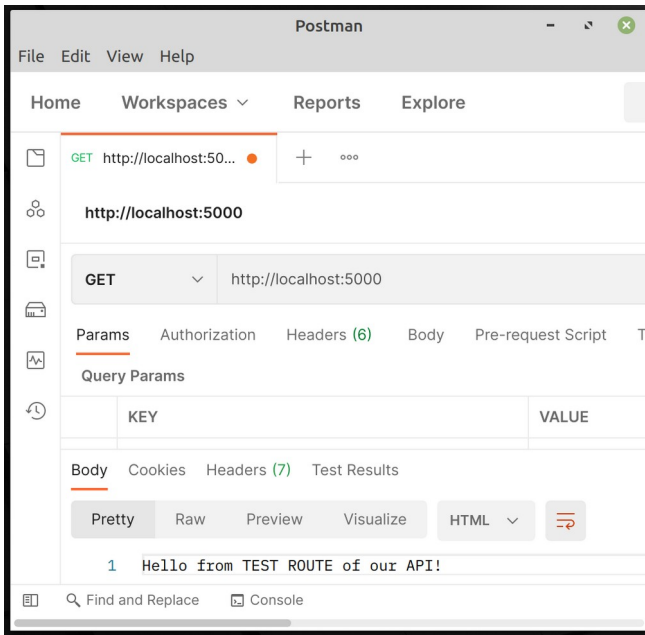
2.2.7 Change start scripts at package.json

```
{ } package.json X
{ } package.json > ...
1  {
2    "name": "mern-devconnector",
3    "version": "1.0.0",
4    "description": "social network MERN stack",
5    "main": "server.js",
   ▶ Debug
6    "scripts": {
7      "start": "node server",
8      "server": "nodemon server.js"
9    },
```

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

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



3. User API Routes & JWT Authentication