1. Mappa készítés (controller, router, middleware)

2. npm init -y

3. serves.js létrehozása

4. routes mappa létr.

5. controllers mappa létr.

6. middleware mapp létr.

7. .env létr.

8. .env PORT=8000

9. server.js require http

10. server.js http server create

11. server.js req,res

12. res.writeHead(200, + content.type

13. res.end() + JSON.stringify

14. npm i dotenv --save

15. require("dotenv").conf

16. const port = process.env.PORT || 8000

17. server.listen + log

18. index.js létr.

19. npm i express

20. require express

21. const app = express();

22. app.get(“/”)

23. req,res + res.send(“html element”);

24. 16 lépés újra (const port = process.env.PORT || 8000)

25. most app.listen(port, (`üzi`)

26. dotenv sor hozzáadás

27. res.status indej.js (200).send(“html element”)

28. router require pl. const userrouter = require("./routes/users")

29. még egy router const products = require("./routes/products")

30.  app.use("/users", userrouter)

31.  app.use("/products", productsrouter)

32.  users.js new route

33.  const express = require("express")const router = express.Router()

A maradékot a fájloknál lehet látni:

index.js:  
const express = require("express");

const app = express();

app.get("/",(req,res) => {

    res.status(200).send("<h1>Hewo mewo</h1>")

});

const userrouter = require("./routes/users")

const productsrouter = require("./routes/products")

app.use("/users", userrouter)

app.use("/products", productsrouter)

require("dotenv").config();

const port = process.env.PORT || 8000;

app.listen(port, () =>{

    console.log(`fut a szero ${port}`)

})

server.js:  
const http = require("http");

const server = http.createServer((req,res) =>{

    res.writeHead(200,{

        'content-type':'application.json'

});

res.end(JSON.stringify({data:"hello node"}));

});

require("dotenv").config();

const port = process.env.PORT || 8000;

server.listen(port, () => console.log(`server run ${port}`));

./routes/users.js:  
const express = require("express")

const router = express.Router()

router.get("/", (req, res) => {

    res.send("Users route");

})

module.exports = router;

Új fájl:

.controller/usercontroller.js:

const getUsers = (req,res) =>{

    res.send("Users route")

}

const getUsersById = (req,res) => {

 const userId = req.params.userid;

 res.send(`Users with id: ${userId}`)

}

const createUser = (req,res) => {

 const newUser = req.body;

 res.status(201).send(`User created: ${newUser}`)

}

module.exports = {

    getUsers,

    getUsersById,

    createUser

}

Új route(ok) a users.js:

router.get("/", userController.getUsers)

router.get("/:id", userController.getUsersById)

router.post("/", userController.createUser)

a

Ez törlődik:

router.get("/", (req, res) => {

    res.send("Users route");

})

Prisma:  
  
npm i prisma@prisma/client  
npx prisma init  
DATABASE\_URL=”mysql://root:@localhost:3306/mydb  
npx prisma migrate dev –name init

Prisma/scema.prisma:

generator client {

  provider = "prisma-client-js"

}

datasource db {

  provider = "mysql"

  url      = env("DATABASE\_URL")

}

model User {

  id Int @id @default(autoincrement())

  email String @unique

  password String

  name String

  createdAt DateTime @default(now())

}

Változik a userController.js:  
  
const {PrismaClient} = require("@prisma/client");

const prisma = new PrismaClient();

const getUsers = async (req,res) =>{

    try {

        const users = await prisma.user.findMany()

        res.json(users)

    } catch (error) {

        console.error(error)

        res.status(500).json({error: "Failed to fetch users"})

    }

}

const getUsersById = async (req,res) => {

 const userId = parseInt(req.params.userid);

 try {

    const user = await prisma.user.findUnique({

        where: {id: userId},

    });

    if (user) {

        res.json(user);

    } else {

        res.status(404).json({error: "User not found"});

    }

}

catch (error){

    res.status(500).json({error: "Failed to fetch"})

}

}

const createUser = async (req,res) => {

 const {name, email} = req.body;

 try{

    const newUser = await prisma.user.create({

        data: {name, email},

    });

    res.status(201).json(newUser)

 } catch (error)

 {

    res.status(500).json({error: "Failed to create user"})

 }

 res.status(201).send(`User created: ${JSON.stringify(newUser)}`)

}

const deleteUser = async (req,res) => {

    const {id} = req.params;

    try {

        await prisma.user.delete({

            where: {id: parseInt(id)}

        })

        res.status(204).send()

    } catch (error) {

        res.status(500).json({error: "Failed to delete user"})

    }

};

module.exports = {

    getUsers,

    getUsersById,

    createUser,

    deleteUser

}

Postmanbe test!