PROJECT

FOLLOW THIS STEPS

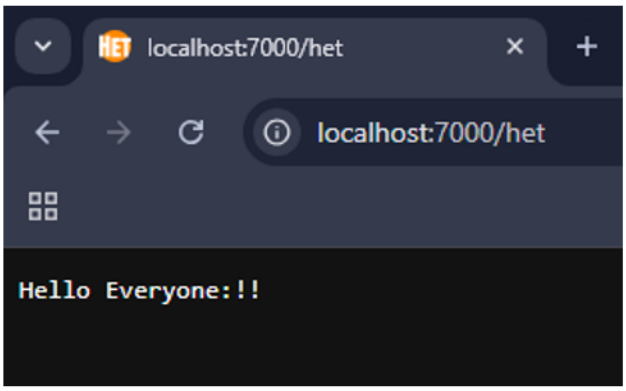
STEPS:

1. Create Folder – (**EXAMPLE**).
2. In **EXAMPLE** folder create a **CURD-OPERATION** Folder.
3. Create a Two files in **SERVER** Folder (**example.js, exampleexpress.js**).
4. Open Terminal tap on Launch Profile to **Git Bash**.
5. **Git-Bash:**

I. node - -version
II. npm init -y
III. mkdir foldername ex: (portfolio)
IV. npm install express . mongoose body -parser cors dotenv
V. npm install http
VI. **node example.js (PROGRAM:1)**

```
Admin@BLACK-DELL MINGW64 /e/example/curd-operation/server
$ node example.js
Server is running on http://localhost:7000
```

VII. Double click on http://localhost :7000 (**follow link**).
VIII. In URL (http://localhost:7000/het)
IX. In URL shows



X. **node exampleexpress.js (PROGRAM:1)**

```
Admin@BLACK-DELL MINGW64 /e/example/curd-operation/server
$ node exampleexpress.js
Server is running on http://localhost:7000
```

XI. Double click on http://localhost :7000 (**follow link**).
XII. In **URL** (http://localhost :7000/het)

XIII. **node exampleexpress.js (PROGRAM:2)**

```
Admin@BLACK-DELL MINGW64 /e/example/curd-operation/server
$ node exampleexpress.js
(node:8328) [MODULE_TYPELESS_PACKAGE_JSON] Warning: Module type of file:///E:/example/curd-operation/server/exampleexpress.js
n't parse as CommonJS.
Reparsing as ES module because module syntax was detected. This incurs a performance overhead.
To eliminate this warning, add "type": "module" to E:\example\curd-operation\server\package.json.
(Use `node --trace-warnings ...` to show where the warning was created)
Server is running on http://localhost:7000
```

XIV. Double click on http://localhost :7000 (**follow link**).

XV. In **URL** (http://localhost:7000**/?secret=1234**)

XVI. Create **.env file** (Terminal - **touch .env**)

XVII. **node exampleexpress.js (PROGRAM:3)**

XVIII. In package.json file add this (**"start": "nodemon index.js"**)

```
Debug
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "start": "nodemon index.js"
```

XIX. npm install --save-dev nodemon

XX. npm run start

XXI. node exampleexpress.js

```
Admin@BLACK-DELL MINGW64 /e/example/curd-operation/server
$ node exampleexpress.js
(node:1196) [MODULE_TYPELESS_PACKAGE_JSON] Warning: Module type of file:///E:/example/curd-operation/server/exampleexpress.js
n't parse as CommonJS.
Reparsing as ES module because module syntax was detected. This incurs a performance overhead.
To eliminate this warning, add "type": "module" to E:\example\curd-operation\server\package.json.
(Use `node --trace-warnings ...` to show where the warning was created)
mongodb://127.0.0.1:27017/hetdb
Server is running on http://localhost:7000
```

XXII. Double click on http://localhost :7000 (**follow link)**

XIII. In **URL (**http://localhost:7000/?hetdb**)**

XIV. In URL shows **portfolio.html file**

6. Create a **Folder**(portfolio) in Server ADD all files (**html,photos**).
7. Open Extensions Install **Thunder Clients**.
   - **Create a NEW REQUEST.**
   - **GET**

     http://localhost:7000

PROJECT

example.js file: (PROGRAM 1)

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

const server =http.createServer((req, res) => {
   if(req.method=== 'GET' && req.url==='/het'){

res.writeHead(200,{'Content-Type':'text/plain'})
res.end("Hello Everyone:!!");
}else{
   res.writeHead(404,{'Content-Type': "text/plan"})
   res.end("Not Found");
}
});

server.listen(7000,()=>{
   console.log("Server is running on http://localhost:7000")
})
```

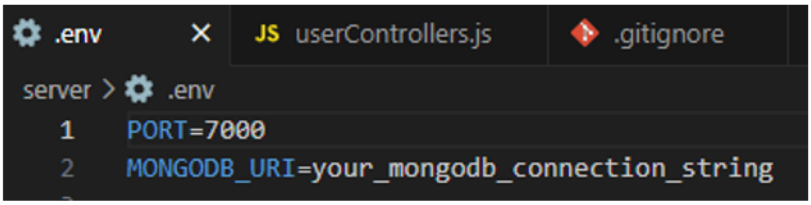
example.js file: (PROGRAM 2)

```
// PROGRAM : 2
import dotenv from "dotenv";
//import http from "http";
import express from 'express';
import path from 'path';
import { fileURLToPath } from 'url';
import mongoose from "mongoose";
dotenv.config();

const port=process.env.PORT=7000;
//const url=process.env.MONGOOSE_URL;
//const url = "mongodb://127.0.0.1:27017/db";
const url = "mongodb://127.0.0.1:27017/";
console.log(url);
const app =express();
const __filename=fileURLToPath(import.meta.url);
const __dirname=path.dirname(__filename);

// ,{useNewUrlParser:true,useUnifiedTopology:true}
mongoose.connect(url)
  .then(()=>{app.listen(port,()=>{
    console.log(`Server is running on http://localhost:${port}`);
  })}
```

```
  )
  .catch((err)=>console.error("Connection Failed",err))

app.use(express.static(path.join(__dirname,"portfolio")))

app.get("/",(req,res)=>{
  res.sendFile(path.join(__dirname,"portfolio","ex1.html"))
})
```

.env file:

```
PORT=7000
MONGODB_URI=your_mongodb_connection_string
```



PROJECT

userModels.js file:

```
import mongoose from "mongoose";
const userschemas = new mongoose.Schema(
  {
    name:{
      type:String,
      required:true
    },
    email:{
      type:String,
      required:true
    },
    password:{
      type:String,
      required:true
    },
    age:{
      type:Number,
      required:true
    }
  },
  {
    timestamps:true
  }
```

)

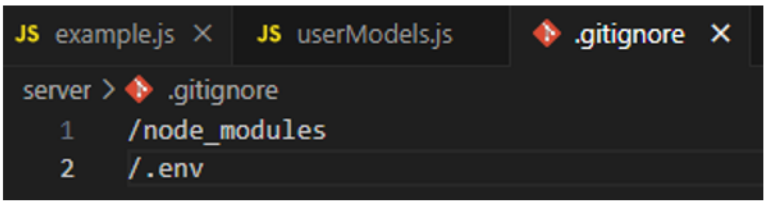
const User= mongoose.model('User',userschemas);

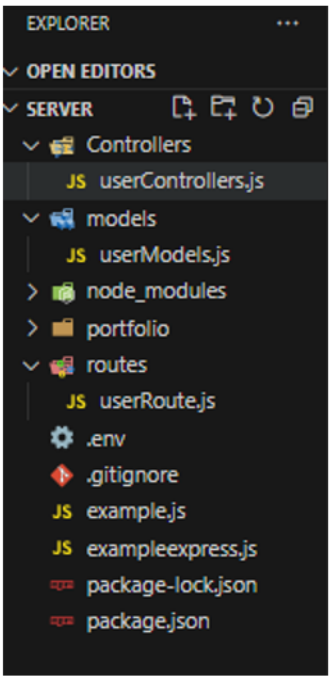
export default User;

.gitignore file:

/node_modules

/.env





Output:

```
Admin@BLACK-DELL MINGW64 /e/file/example/crud-operation/server
$ node example.js
mongodb://127.0.0.1:27017/
Server is running on http://localhost:7000
```

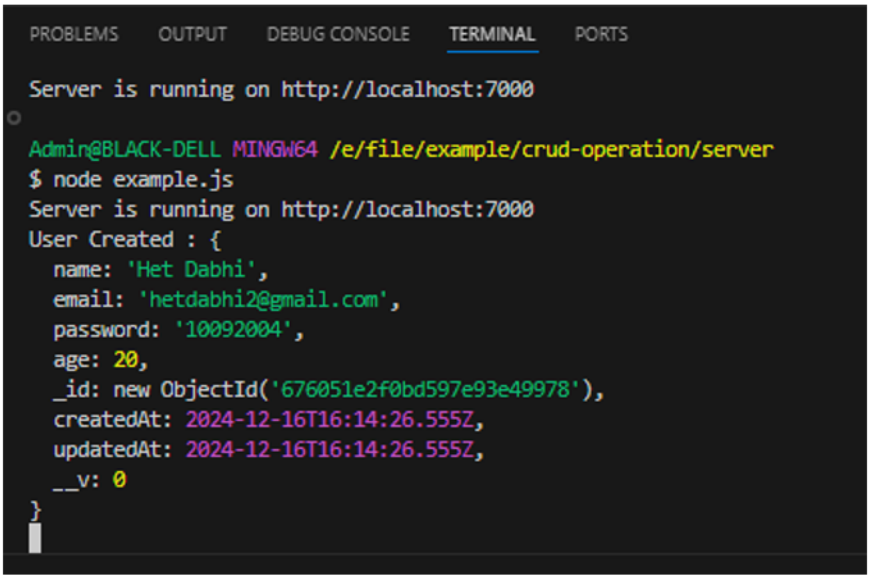
example.js file: (PROGRAM 3)

```
import dotenv from "dotenv";
import express from 'express';
import bodyparser from 'body-parser' ;
import mongoose from "mongoose";
dotenv.config();
import User from './models/userModels.js';

const app=express();
const port=process.env.PORT;
const url=process.env.MONGOOSE_URL;

app.use(bodyparser.json());
app.use(bodyparser.urlencoded({extended:false}));
// ,{useNewUrlParser:true,useUnifiedTopology:true})
mongoose.connect(url)
  .then(()=>{

    app.listen(port,()=>{
    console.log(` Server is running on http://localhost:${port}` );
  })}

  )
  .catch((err)=>console.error("Connection Failed",err))
```

PROJECT

```
async function create_User()
{
   const user = new User({
      name:"Het Dabhi",
      email:"hetdabhi2@gmail.com",
      password:"10092004",
      age:20
       });
       const result = await user.save();
       console.log("User Created :",result);
}
create_User();
```

Output:



PROJECT

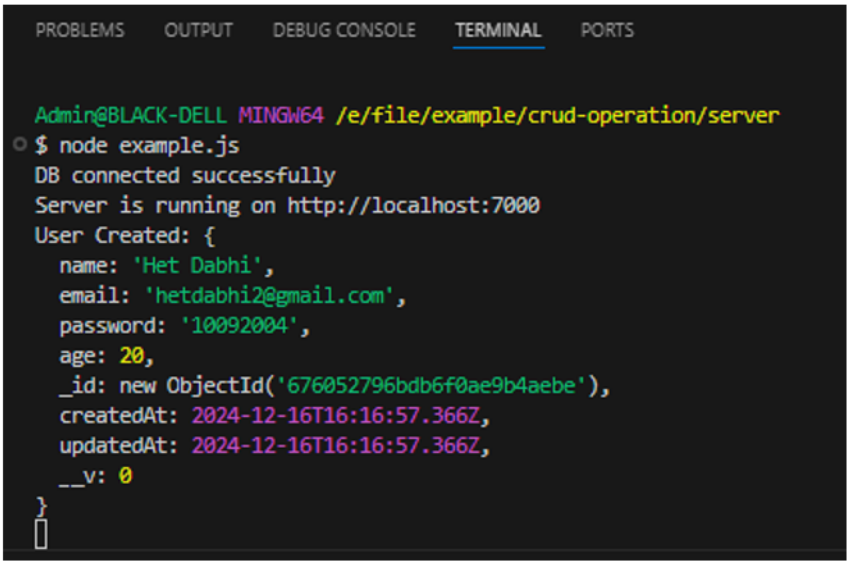
example.js file: (PROGRAM 2 & 3 Merged Code)

```
// Program  2 & 3 (Merged Code):

import dotenv from "dotenv";

import express from 'express';

import bodyParser from 'body-parser';  // Corrected the import name

import mongoose from "mongoose";

import path from 'path';

import { fileURLToPath } from 'url';

import User from './models/userModels.js';  // Assuming this is the user schema file


dotenv.config();


const app = express();

const port = process.env.PORT || 7000;  // Default to 7000 if not provided in .env

const url = process.env.MONGOOSE_URL;  // Ensure your .env has the correct MONGOOSE_URL


// Middleware to parse request bodies

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: false }));


// Static file serving (e.g., serving HTML files)
```

```
const __filename = fileURLToPath(import.meta.url);

const __dirname = path.dirname(__filename);

app.use(express.static(path.join(__dirname, "portfolio")));  // Assuming you have a "portfolio" folder


// MongoDB connection

mongoose.connect(url)

 .then(() => {

  console.log("DB connected successfully");

  app.listen(port, () => {

   console.log(`Server is running on http://localhost:${port}`);

  });

 })

 .catch((err) => console.error("Connection Failed", err));


// Route to serve an HTML file

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

 res.sendFile(path.join(__dirname, "portfolio", "ex1.html"));  // Serve ex1.html from the portfolio directory

});


// Function to create a user

async function createUser() {

 try {

  const user = new User({
```

```
    name: "Het Dabhi",
    email: "hetdabhi2@gmail.com",
    password: "10092004",  // Ideally, hash the password before storing it
    age: 20
  });
  const result = await user.save();
  console.log("User Created:", result);
 } catch (error) {
  console.error("Error creating user:", error);
 }
}
// Call the function to create a user
createUser();
```

Output:



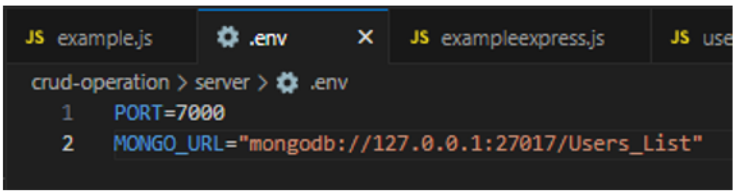
example.js file: (PROGRAM 4): (POST METHOD) -create

```
import express from 'express';
import dotenv from 'dotenv';
import mongoose from 'mongoose';
import route from './routes/userRoute.js';

// Load environment variables
dotenv.config();

const app = express();
const PORT = process.env.PORT || 7000;
const URL = process.env.MONGO_URL;

// Built-in body parsing middleware (replaces body-parser)
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

// Log the PORT to verify
console.log(` Port: ${PORT}` );

// Basic GET Route
app.get("/", (req, res) => {
  res.send("Hello");
});
```

```
// Connect to MongoDB
mongoose.connect(URL, {
  useNewUrlParser: true,
  useUnifiedTopology: true
})
  .then(() => {
    console.log("DB Connected Successfully");
    app.listen(PORT, () => {
      console.log(`Server is running on http://localhost:${PORT}`);
    });
  })
  .catch((err) => {
    console.error("Connection Failed:", err);
  });
// Use routes middleware
app.use('/api', route);
```

.env file:

```
PORT=7000
MONGO_URL="mongodb://127.0.0.1:27017/Users_List"
```



userControllers.js file:

```
import User from "../models/userModels.js";
export const createUser = async (req, res) => {
 try {
  const userData = new User(req.body);
  if (!userData) {
   res.status(404).json({ msg: "User not Found" });
  }
  await userData.save();
  res.status(200).json({ msg: "User got created successfully." });
 } catch (error) {
  res.status(500).json({ err: error });
 }
};
```

userRoute.js file:

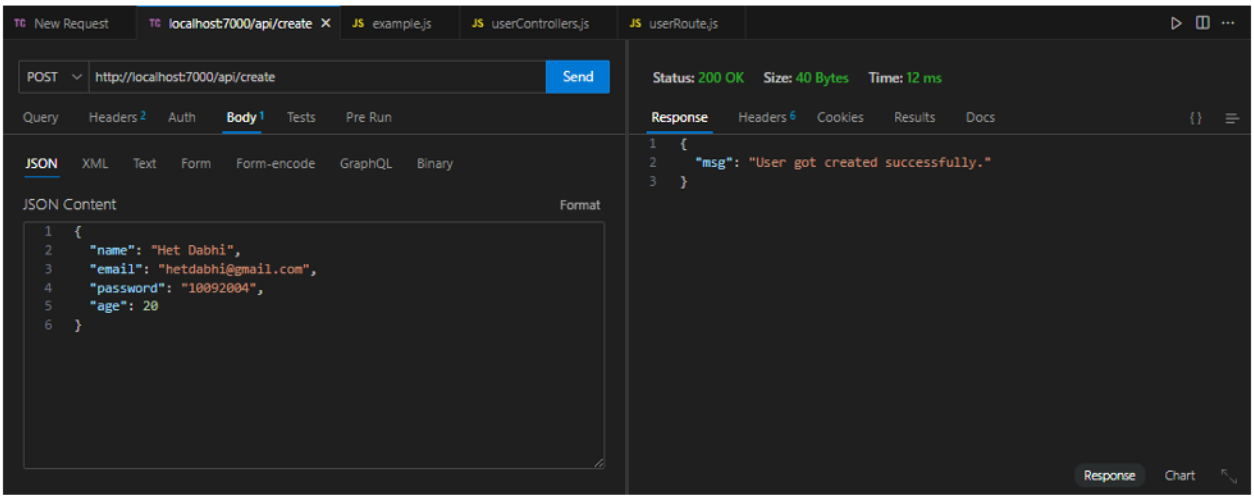
```
// Import necessary modules
import { createUser } from "../Controllers/userControllers.js"; // Controller to handle user creation logic
import express from "express"; // Express framework for routing
// Initialize a router object
const route = express.Router();
// Define a POST route for creating a user
route.post("/create", createUser);
```

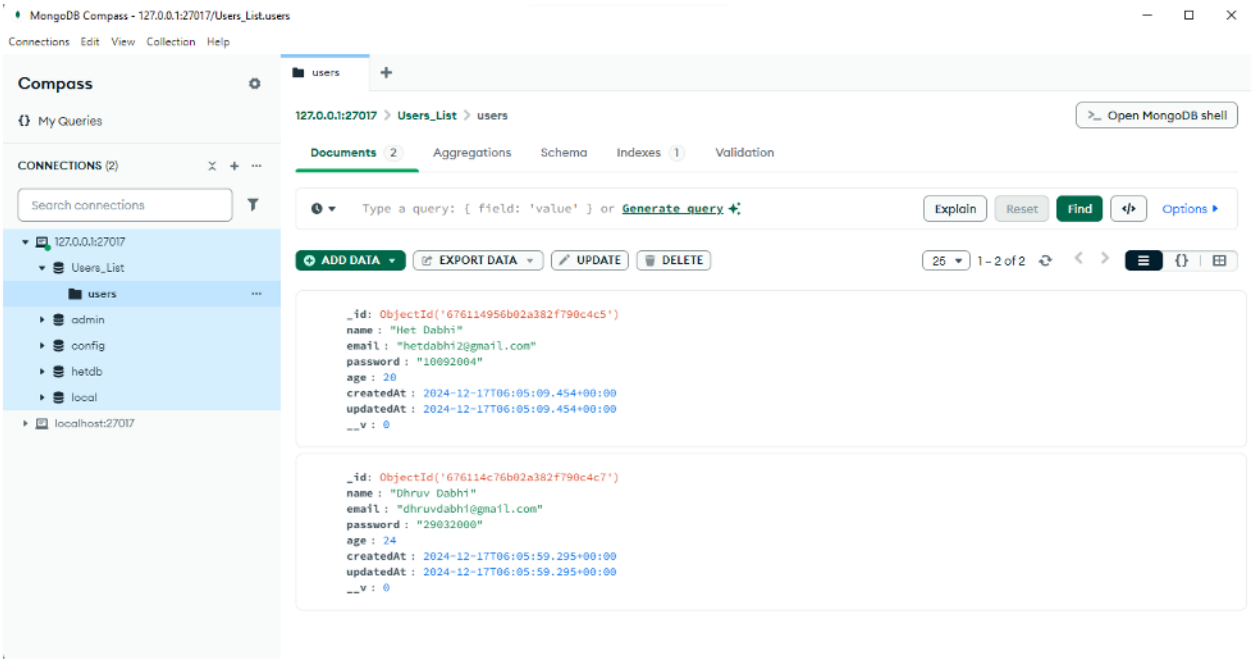
PROJECT

// Export the router for use in other parts of the app

export default route;

Thunder Client:





PROJECT

example.js file: (PROGRAM 5): (GET METHOD)

- Same as example.js file

userControllers.js file:

```
import User from "../models/userModels.js";


export const createUser = async (req, res) => {
 try {
  const userData = new User(req.body);
  if (!userData) {
   res.status(404).json({ msg: "User not Found" });
  }
  await userData.save();
  res.status(200).json({ msg: "User got created successfully." });
 } catch (error) {
  res.status(500).json({ err: error });
 }
}


export const getAll =  async (req,res)=>{
  try {
     const userData= await User.find();
  if(!userData){res.status(404).json({msg:"User not found"});
  }
```

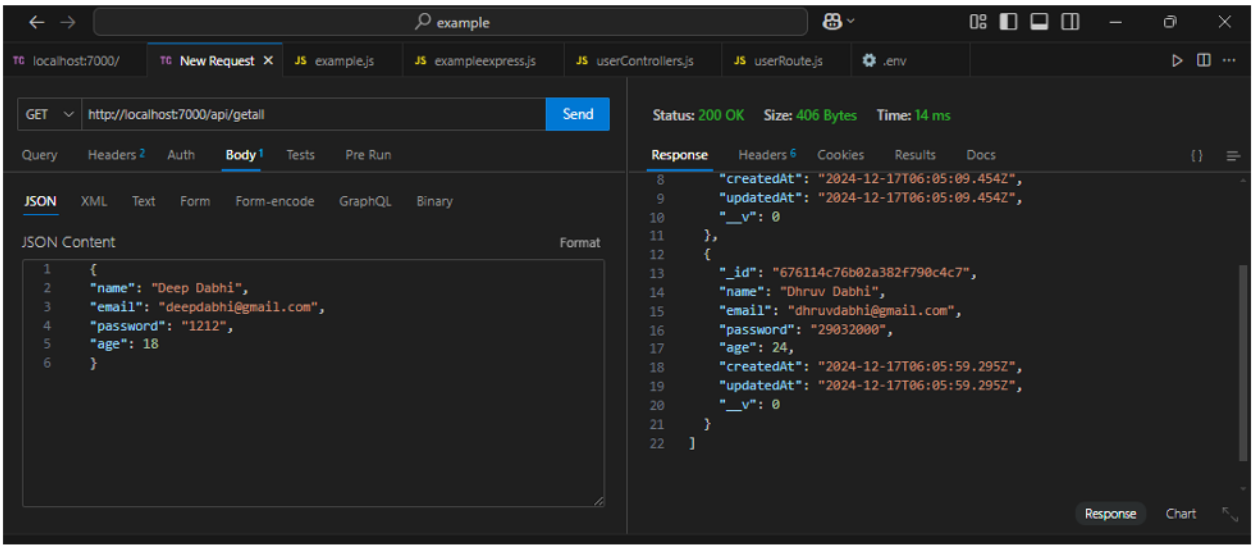
```
    res.status(200).json(userData);
    }

catch(error){
    res.status(500).json({err:error})
    }
}
```

userRoute.js file:

```
// Import necessary modules
import { createUser, getAll } from "../Controllers/userControllers.js"; // Controller for user-related logic
import express from "express"; // Express framework for routing

// Initialize a router object
const route = express.Router();
// Define a POST route for creating a user
route.post("/create", createUser);
// Define a GET route for retrieving all users
route.get("/getall", getAll);
// Export the router for use in other parts of the app
export default route;
```

PROJECT

Thunder Client:



example.js file: (PROGRAM 6): (GET METHOD) - finduser

- Same as example.js file

userControllers.js file:

```
import User from "../models/userModels.js";
export const createUser = async (req, res) => {
 try {
  const userData = new User(req.body);
  if (!userData) {
   res.status(404).json({ msg: "User not Found" });
  }
  await userData.save();
  res.status(200).json({ msg: "User got created successfully." });
 } catch (error) {
  res.status(500).json({ err: error });
 }
}


export const getAll =  async (req,res)=>{
  try {
     const userData= await User.find();
  if(!userData){res.status(404).json({msg:"User not found"});
  }
```

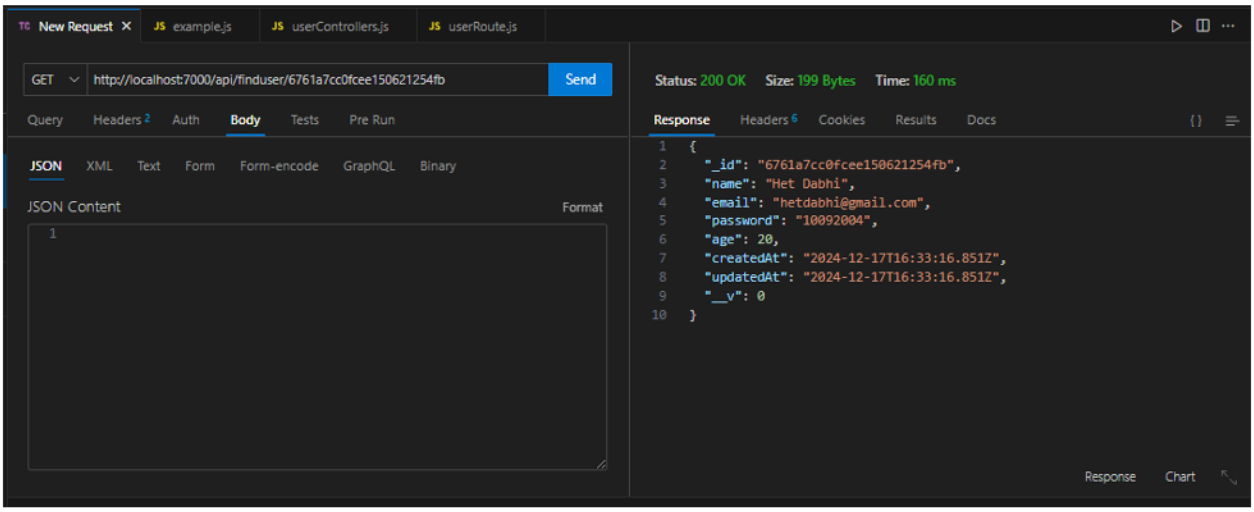
```
    res.status(200).json(userData);
    }

catch(error){
    res.status(500).json({err:error})
    }
}

export const findUserById = async(req,res)=> {
    try{
        const userid= req.params.id;

     const userData= await User.findById(userid);

     if(!userData){
        res.status(404).json({message:"User not found."});
       }
       res.status(200).json(userData);
    }
     catch(error){
       res.status(500).json({err:error});
    }

}
```

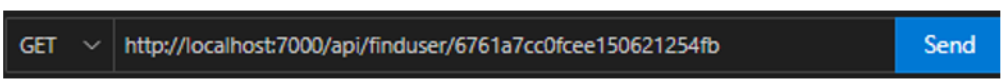
PROJECT

userRoute.js file:

```
import { createUser,getAll,findUserById } from
"../Controllers/userControllers.js";

import express from "express"; // Express framework for routing


// Initialize a router object

const route = express.Router();

// Define a POST route for creating a user

route.post("/create", createUser);

route.get("/getall", getAll);

route.get("/finduser/:id",findUserById);

export default route;
```

Thunder Clients:





example.js file: (PROGRAM 7): (UPDATE METHOD) - update

- Same as example.js file

userControllers.js file:

```
import User from "../models/userModels.js";
export const createUser = async (req, res) => {
 try {
  const userData = new User(req.body);
  if (!userData) {
   res.status(404).json({ msg: "User not Found" });
  }
  await userData.save();
  res.status(200).json({ msg: "User got created successfully." });
 } catch (error) {
  res.status(500).json({ err: error });
 }
}

export const getAll =  async (req,res)=>{
  try {
     const userData= await User.find();
  if(!userData){res.status(404).json({msg:"User not found"});
  }
  res.status(200).json(userData);
```

```
  }

catch(error){
  res.status(500).json({err:error})
  }
}

export const findUserById = async(req,res)=> {
  try{
     const userid= req.params.id;

   const userData= await User.findById(userid);

   if(!userData){
     res.status(404).json({message:"User not found."});
    }
    res.status(200).json(userData);
  }
   catch(error){
    res.status(500).json({err:error});
  }

}
```

PROJECT

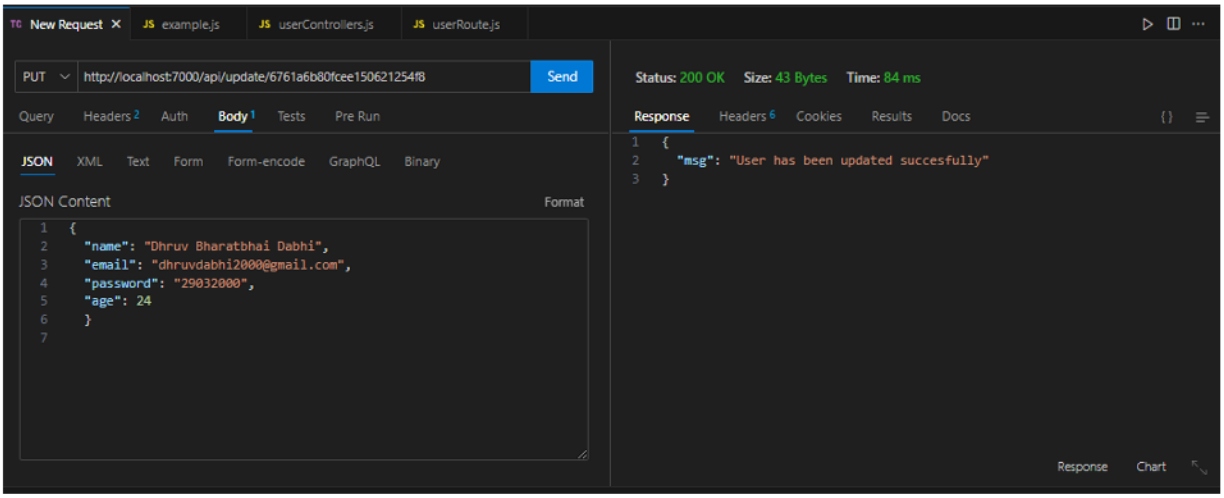
```
export const updateUser = async(req,res)=>{
  try{
     const userid= req.params.id;

    const userExist= await User.findById(userid);

    if(!userExist){
      res.status(404).json({message:"User not found."});
     }
     const updatedata= await
User.findByIdAndUpdate(userid,req.body,{new:true});

     res.status(200).json({msg:"User has been updated succesfully"});
   }
    catch(error){
     res.status(500).json({err:error});
   }
}
```

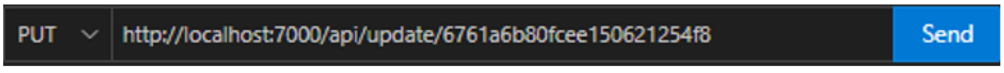
PROJECT

userRoute.js file:

```
import { createUser,getAll,findUserById,updateUser } from "../Controllers/userControllers.js";

import express from "express"; // Express framework for routing


// Initialize a router object

const route = express.Router();

// Define a POST route for creating a user

route.post("/create", createUser);

route.get("/getall", getAll);

route.get("/finduser/:id",findUserById);

route.put("/update/:id",updateUser);

export default route;
```

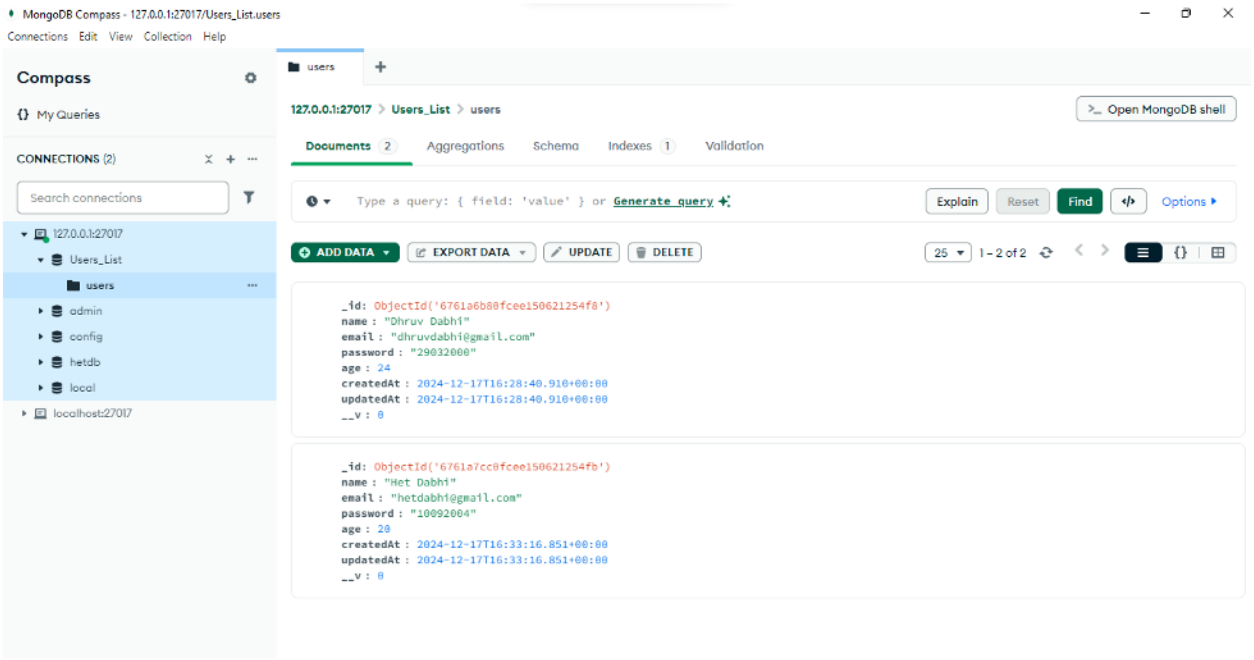
Thunder Clients:



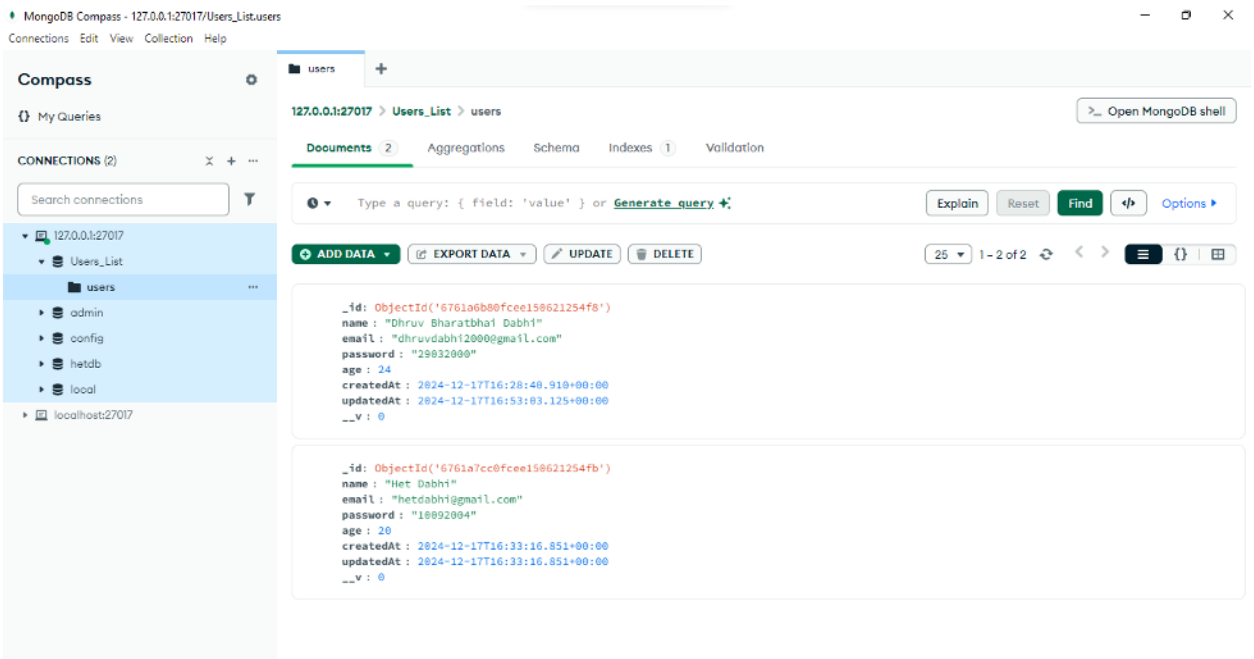


PROJECT

Before:



After:



example.js file: (PROGRAM 8): (DELETE METHOD) - delete

- Same as example.js file

userControllers.js file:

```
import User from "../models/userModels.js";
export const createUser = async (req, res) => {
 try {
  const userData = new User(req.body);
  if (!userData) {
   res.status(404).json({ msg: "User not Found" });
  }
  await userData.save();
  res.status(200).json({ msg: "User got created successfully." });
 } catch (error) {
  res.status(500).json({ err: error });
 }
}

export const getAll =  async (req,res)=>{
  try {
     const userData= await User.find();
  if(!userData){res.status(404).json({msg:"User not found"});
  }
  res.status(200).json(userData);
```

```
    }

catch(error){
    res.status(500).json({err:error})
    }
}

export const findUserById = async(req,res)=> {
    try{
        const userid= req.params.id;

     const userData= await User.findById(userid);

     if(!userData){
        res.status(404).json({message:"User not found."});
       }
       res.status(200).json(userData);
    }
     catch(error){
       res.status(500).json({err:error});
    }

 }
```

```
export const updateUser = async(req,res)=>{
  try{
      const userid= req.params.id;

    const userExist= await User.findById(userid);

    if(!userExist){
      res.status(404).json({message:"User not found."});
     }
     const updatedata= await
User.findByIdAndUpdate(userid,req.body,{new:true});

      res.status(200).json({msg:"User has been updated succesfully"});
   }
    catch(error){
      res.status(500).json({err:error});
   }
}

export const DeleteUser = async(req,res)=>{
 try{
     const userid= req.params.id;

  const  userData= await User.findById(userid);
```

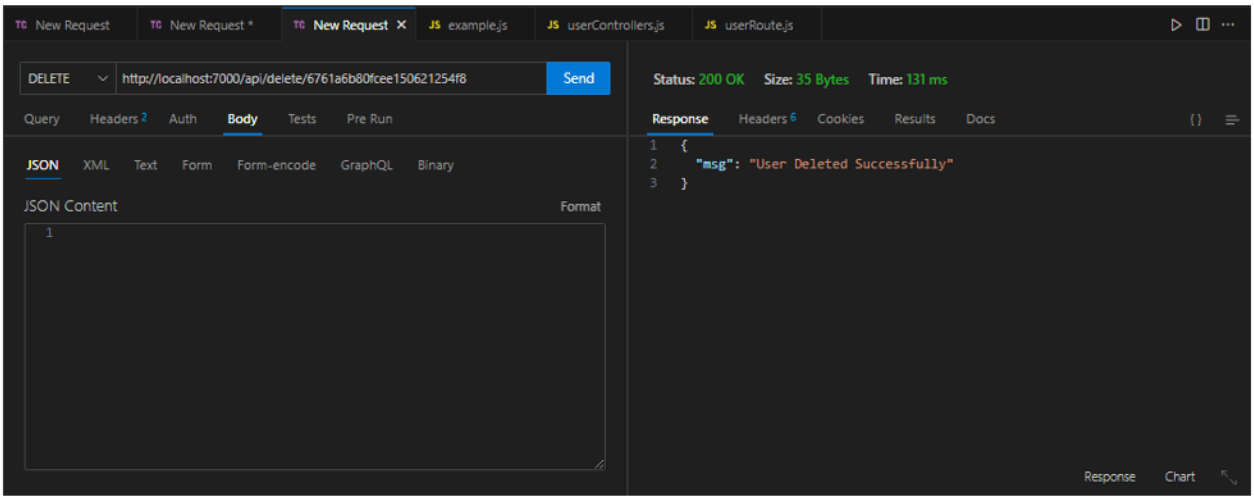
```
  if(!userData){
    res.status(404).json({message:"User not found."});
   }
    const deleteData = await User.findByIdAndDelete(userid);
    res.status(200).json({msg:"User Deleted Successfully"});
 }
  catch(error){
    res.status(500).json({err:error});
 }
}
```

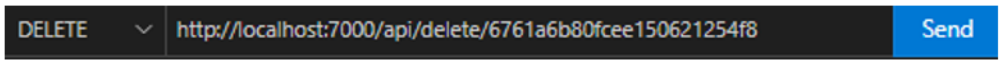
userRoute.js file:

```
import { createUser,getAll,findUserById,updateUser,DeleteUser } from "../Controllers/userControllers.js";
import express from "express"; // Express framework for routing


// Initialize a router object
const route = express.Router();
// Define a POST route for creating a user
route.post("/create", createUser);
route.get("/getall", getAll);
route.get("/finduser/:id",findUserById);
route.put("/update/:id",updateUser);
route.delete("/delete/:id",DeleteUser);
export default route;
```

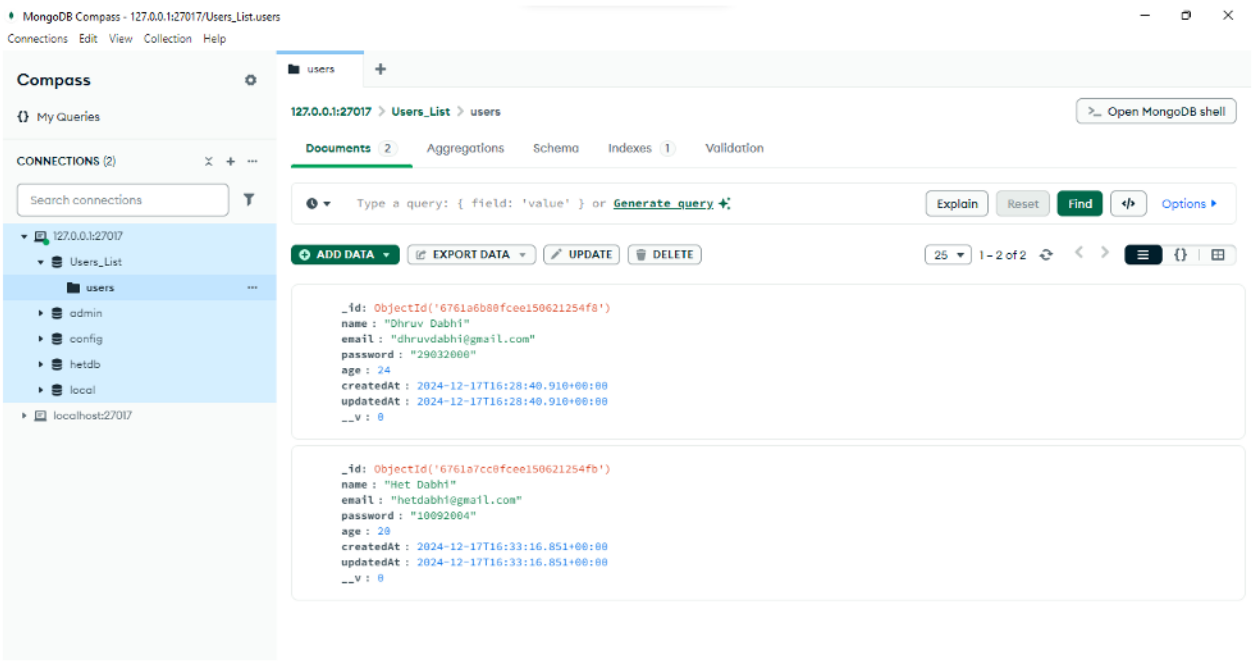
PROJECT

Thunder Clients:



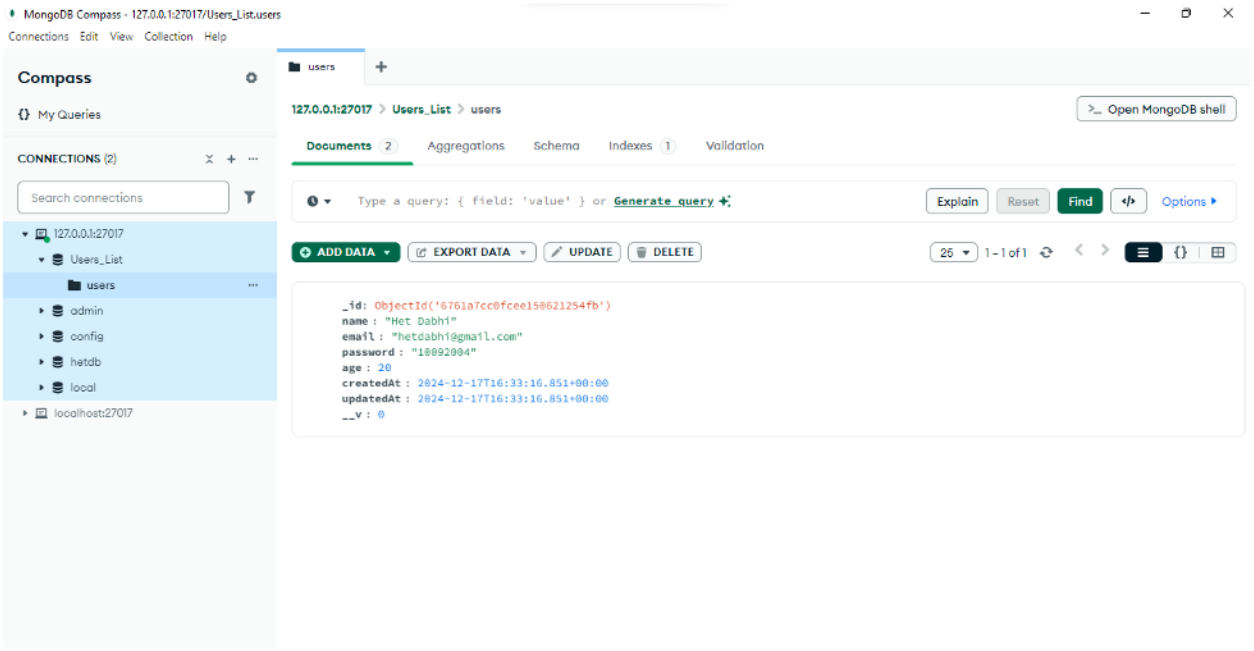


Before:



PROJECT

After:



exampleexpress.js: (PROGRAM 1)

```
const express = require('express');
const path = require('path');
const app = express();
const PORT = 7000;

// Serve static files (HTML, images, etc.) from the "portfolio" directory
app.use(express.static(path.join(__dirname, 'portfolio')));

app.get(['/', '/portfolio'], (req, res) => {
  res.sendFile(path.join(__dirname, 'portfolio', 'ex1.html'));
});

app.use((req, res) => {
  res.status(404).send('Not Found');
});

app.listen(PORT, () => {
  console.log(` Server is running on http://localhost:${PORT}` );
});
```

exampleexpress.js: (PROGRAM 2)

```
import express from 'express';
import path from 'path';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';

dotenv.config();

const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);

const app = express();
app.use(express.json());

const port = process.env.PORT || 7000;
const secret = process.env.SECRET || '1234';

// Middleware for authorization
app.use((req, res, next) => {
  req.isAuthorized = req.query.secret === secret;
  next();
});

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

```
  if (req.isAuthorized) {
    res.sendFile(path.join(__dirname, "portfolio", "ex1.html"));
  } else {
    res.status(401).send("Unauthorized");
  }
});

app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

.env: (PROGRAM:2)

PORT=7000

exampleexpress.js: (PROGRAM 3)

```
import dotenv from "dotenv";
import express from "express";
import path from "path";
import { fileURLToPath } from "url";
import mongoose from "mongoose";

// Load environment variables from .env file
dotenv.config();

const port = process.env.PORT; // Port for the server
const url = process.env.MONGOS_URL; // MongoDB connection URL
console.log(url);

const app = express();

// Resolve __dirname for ES Modules
const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);

// Connect to MongoDB
mongoose.connect(url)
  .then(() => {
    // Start the server after successful connection
```

```
    app.listen(port, () => {
        console.log(`Server is running on http://localhost:${port}`);
    });
  })
  .catch((err) => console.error("Connection Failed", err));

// Middleware to serve static files
app.use(express.static(path.join(__dirname, "portfolio")));

// Route to serve the ex1.html file
app.get("/", (req, res) => {
  res.sendFile(path.join(__dirname, "portfolio", "ex1.html"));
});
```

.env: (PROGRAM:3)

```
PORT=7000
MONGOS_URL=mongodb://127.0.0.1:27017/hetdb
```