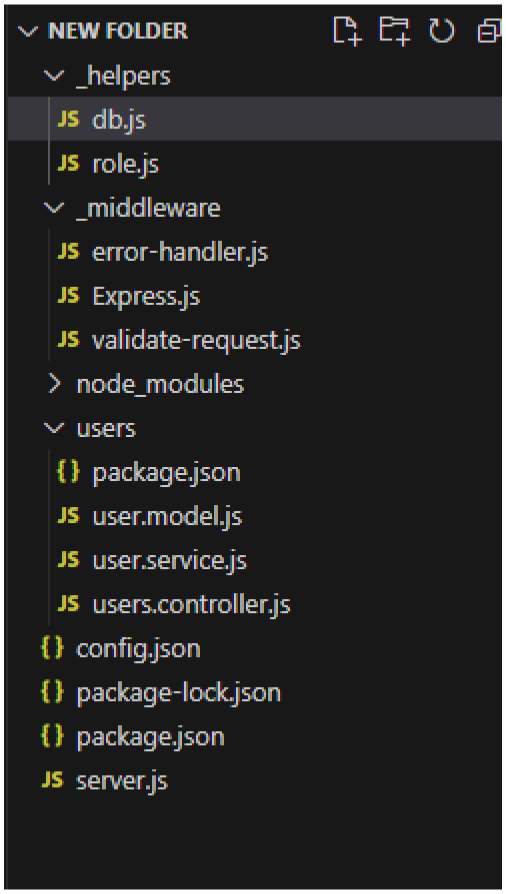
Node.js + MySQL - CRUD API Example

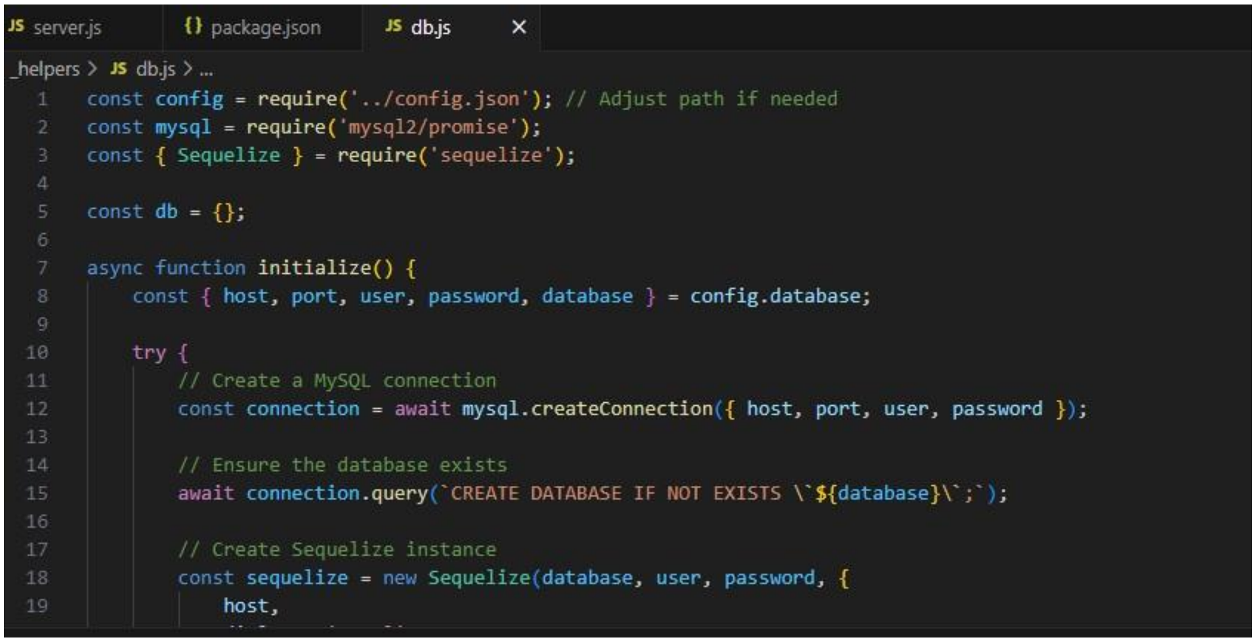
Louise Clark Pareja BSIT 3B

Make ang Directory and install the dependencies



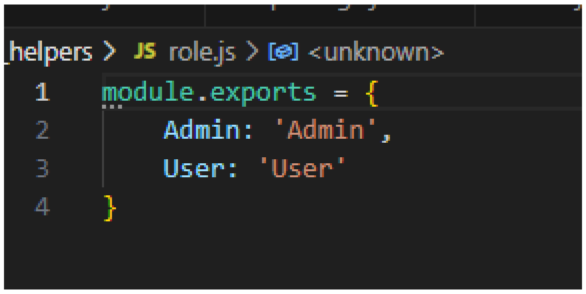
```
NEW FOLDER
  _helpers
    db.js
    role.js
  _middleware
    error-handler.js
    Express.js
    validate-request.js
  node_modules
  users
    package.json
    user.model.js
    user.service.js
    users.controller.js
  config.json
  package-lock.json
  package.json
  server.js
```

Helpers > db.js



```
const config = require('../config.json'); // Adjust path if needed
const mysql = require('mysql2/promise');
const { Sequelize } = require('sequelize');

const db = {};

async function initialize() {
    const { host, port, user, password, database } = config.database;

    try {
        // Create a MySQL connection
        const connection = await mysql.createConnection({ host, port, user, password });

        // Ensure the database exists
        await connection.query(`CREATE DATABASE IF NOT EXISTS \`${database}\`;`);

        // Create Sequelize instance
        const sequelize = new Sequelize(database, user, password, {
            host,
```

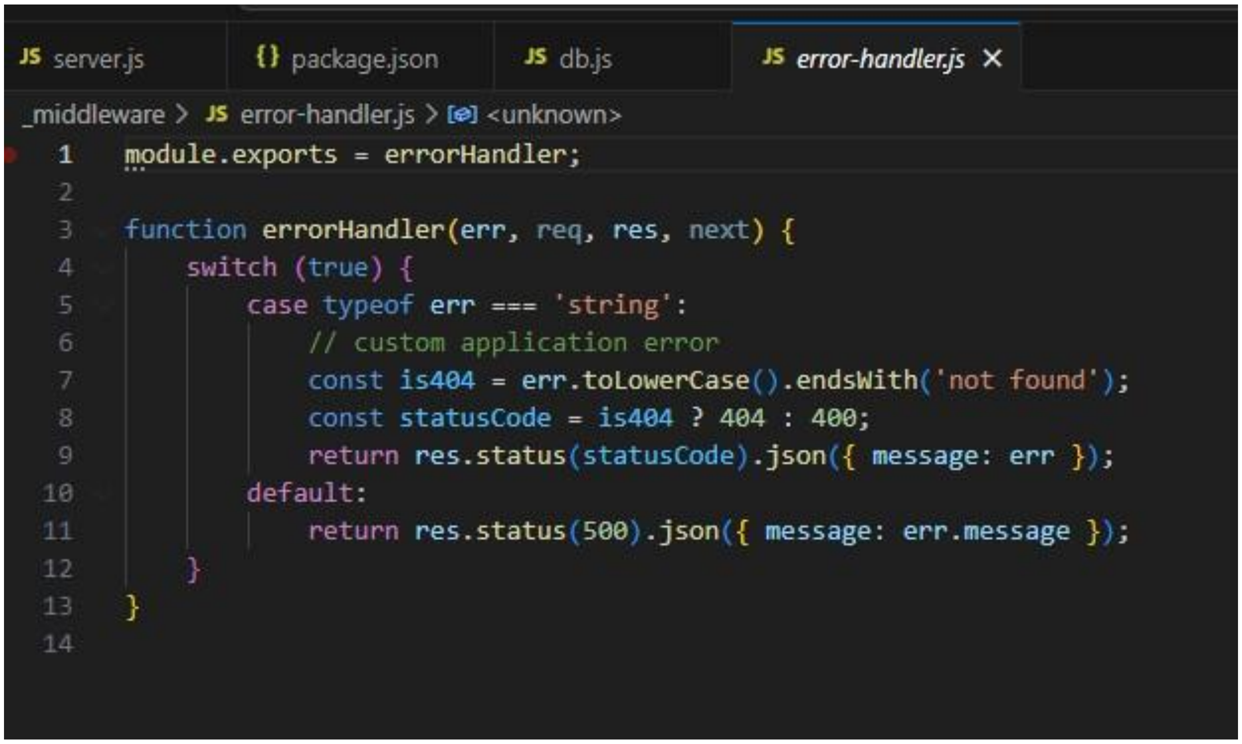
Role -Create an Admin and user role object defines all the roles in the example application,



_helpers > JS role.js > <unknown>

```js
module.exports = {
    Admin: 'Admin',
    User: 'User'
}
```

Error Handler -The error handler is a centralized function that catches and processes errors across the application, ensuring consistent error responses.

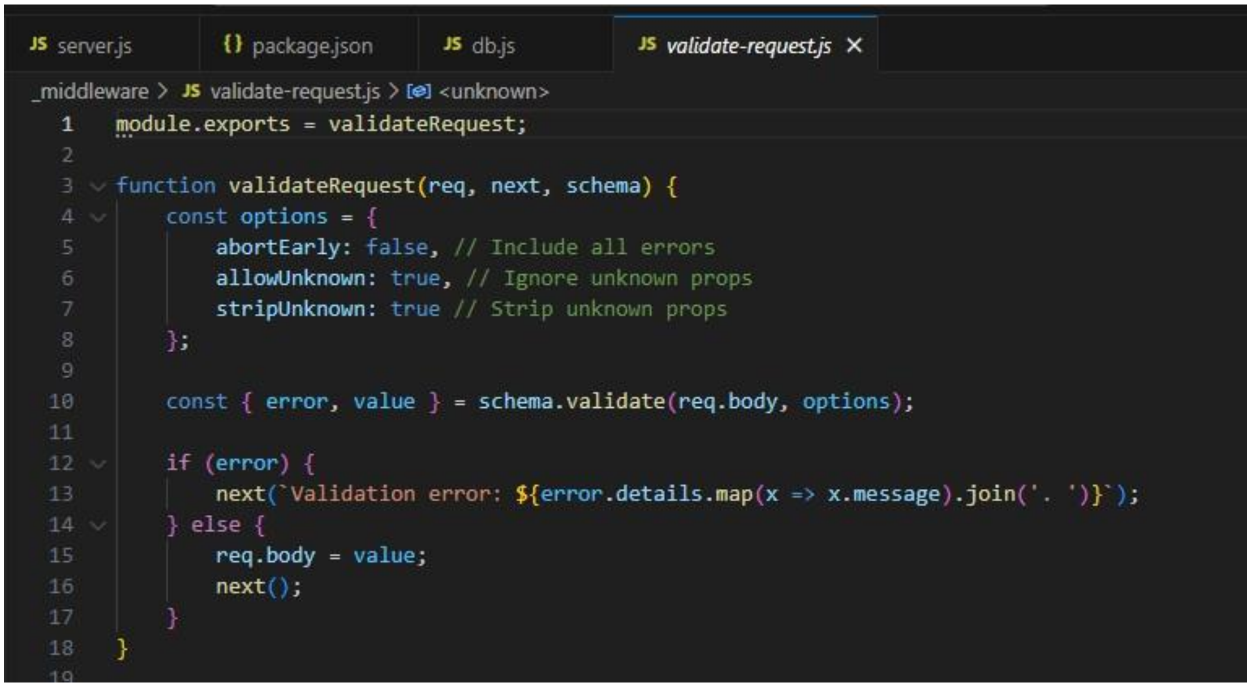


server.js | package.json | db.js | error-handler.js

_middleware > JS error-handler.js > <unknown>

```js
module.exports = errorHandler;

function errorHandler(err, req, res, next) {
    switch (true) {
        case typeof err === 'string':
            // custom application error
            const is404 = err.toLowerCase().endsWith('not found');
            const statusCode = is404 ? 404 : 400;
            return res.status(statusCode).json({ message: err });
        default:
            return res.status(500).json({ message: err.message });
    }
}
```

Validate req. - create an validate request middleware function validates the body of a request against a Joi .

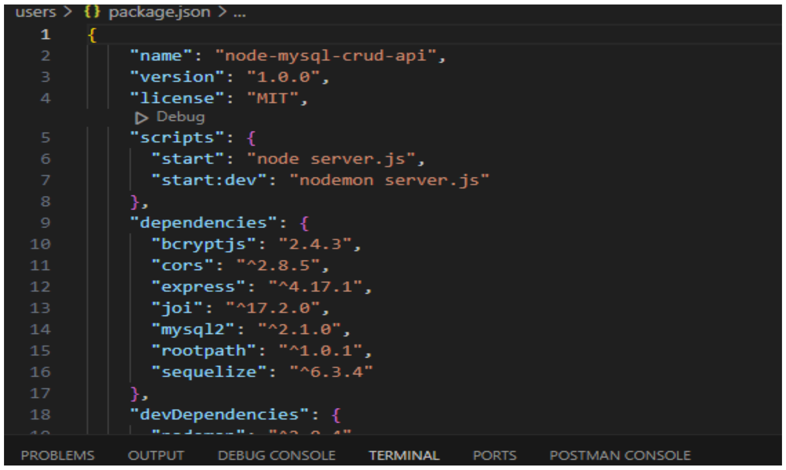


server.js | package.json | db.js | validate-request.js

_middleware > validate-request.js > <unknown>

```js
module.exports = validateRequest;

function validateRequest(req, next, schema) {
    const options = {
        abortEarly: false, // Include all errors
        allowUnknown: true, // Ignore unknown props
        stripUnknown: true // Strip unknown props
    };

    const { error, value } = schema.validate(req.body, options);

    if (error) {
        next(`Validation error: ${error.details.map(x => x.message).join('. ')}`);
    } else {
        req.body = value;
        next();
    }
}
```

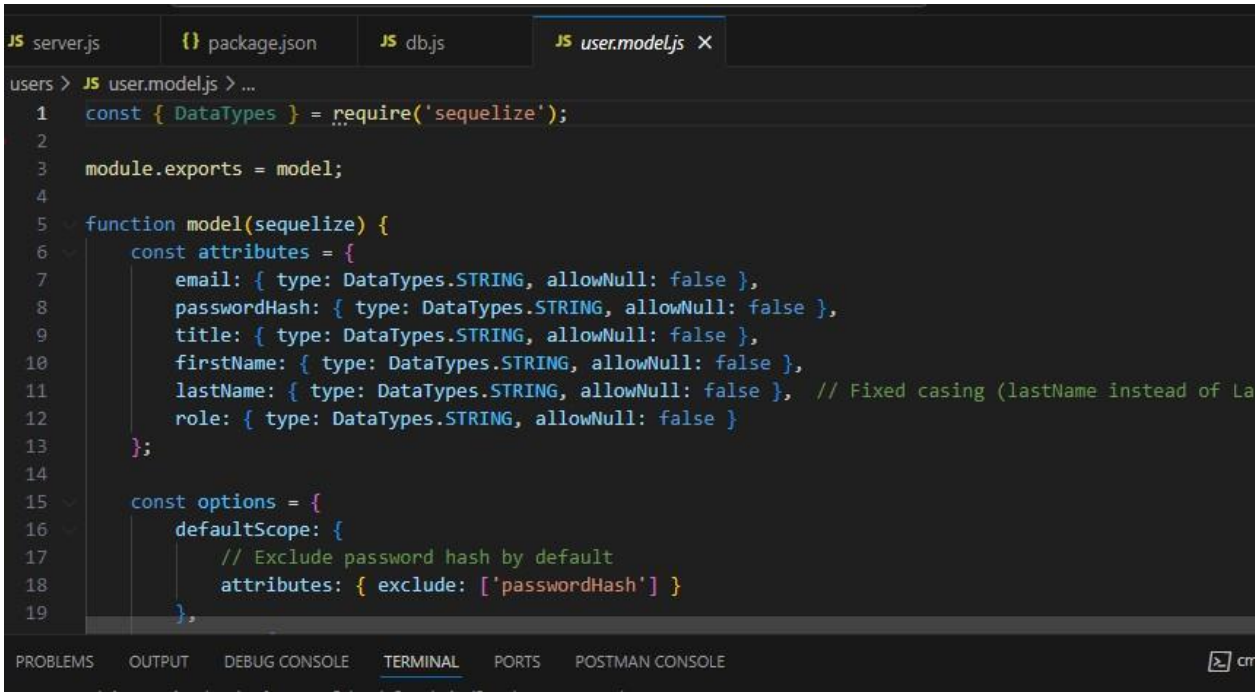
Package json- package.json file contains project configuration information including Node.js package dependencies that get installed when you run npm install.



users > package.json > ...

```json
{
    "name": "node-mysql-crud-api",
    "version": "1.0.0",
    "license": "MIT",
    "scripts": {
      "start": "node server.js",
      "start:dev": "nodemon server.js"
    },
    "dependencies": {
      "bcryptjs": "2.4.3",
      "cors": "^2.8.5",
      "express": "^4.17.1",
      "joi": "^17.2.0",
      "mysql2": "^2.1.0",
      "rootpath": "^1.0.1",
      "sequelize": "^6.3.4"
    },
    "devDependencies": {
```

User model -The `User` model, defined with Sequelize, represents the users table in MySQL. It provides full CRUD (Create, Read, Update, Delete) functionality.

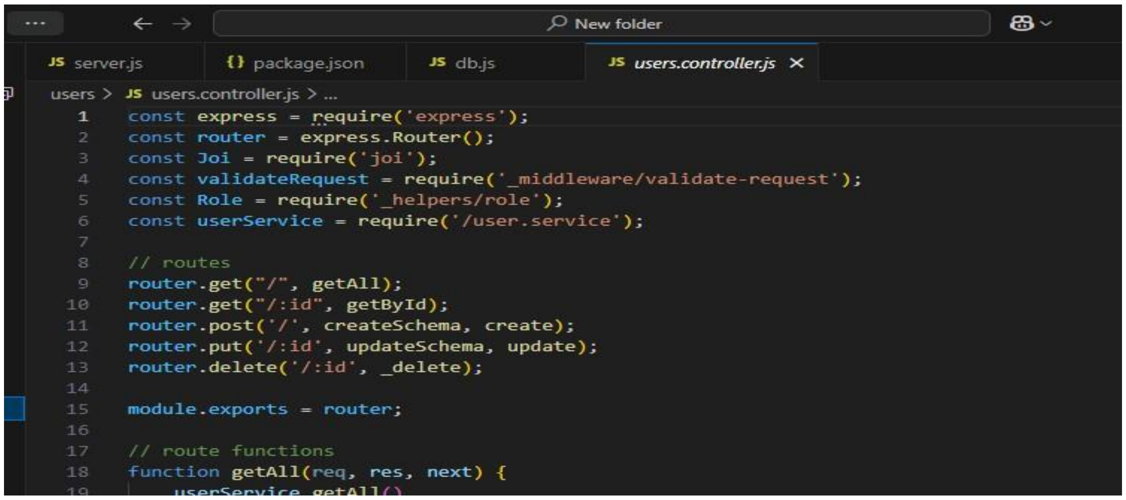


server.js
package.json
db.js
user.model.js
users > user.model.js > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE

```
const { DataTypes } = require('sequelize');

module.exports = model;

function model(sequelize) {
    const attributes = {
        email: { type: DataTypes.STRING, allowNull: false },
        passwordHash: { type: DataTypes.STRING, allowNull: false },
        title: { type: DataTypes.STRING, allowNull: false },
        firstName: { type: DataTypes.STRING, allowNull: false },
        lastName: { type: DataTypes.STRING, allowNull: false },  // Fixed casing (lastName instead of La
        role: { type: DataTypes.STRING, allowNull: false }
    };

    const options = {
        defaultScope: {
            // Exclude password hash by default
            attributes: { exclude: ['passwordHash'] }
        },
```

user.control.js -The users controller handles all /users routes for the Node.js + MySQL CRUD API.

Route Definitions ,Implementation Functions, Schema Validation



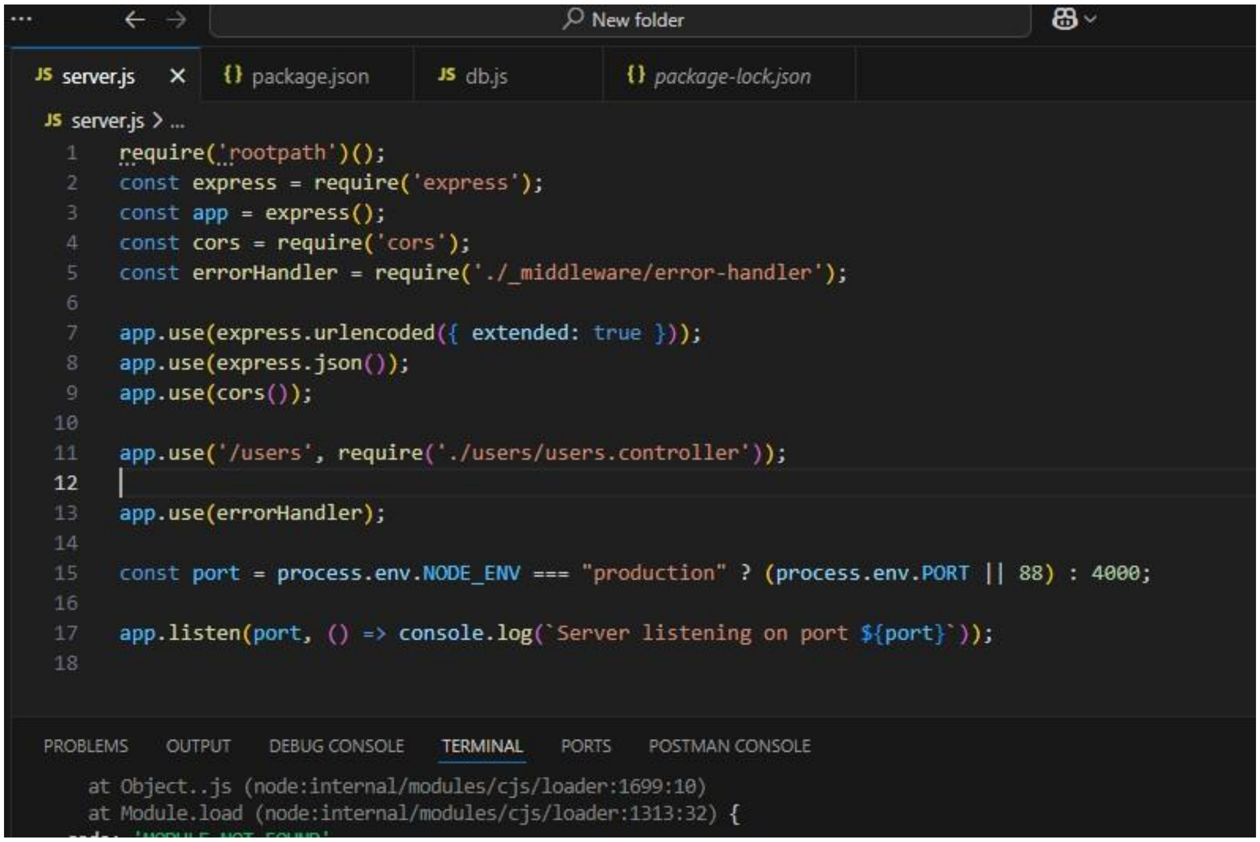
```
const express = require('express');
const router = express.Router();
const Joi = require('joi');
const validateRequest = require('_middleware/validate-request');
const Role = require('_helpers/role');
const userService = require('/user.service');

// routes
router.get("/", getAll);
router.get("/:id", getById);
router.post('/', createSchema, create);
router.put('/:id', updateSchema, update);
router.delete('/:id', _delete);

module.exports = router;

// route functions
function getAll(req, res, next) {
    userService.getAll()
```

server.js - The server.js file serves as the entry point for the Node.js CRUD API.

Configures middleware.

Binds controllers to routes.

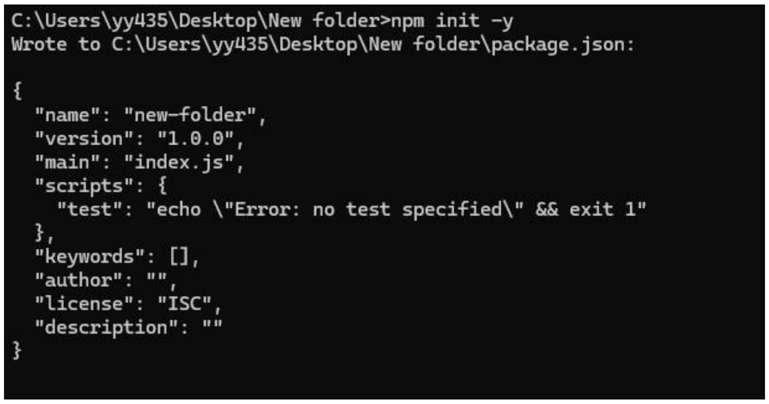
Starts the Express server.



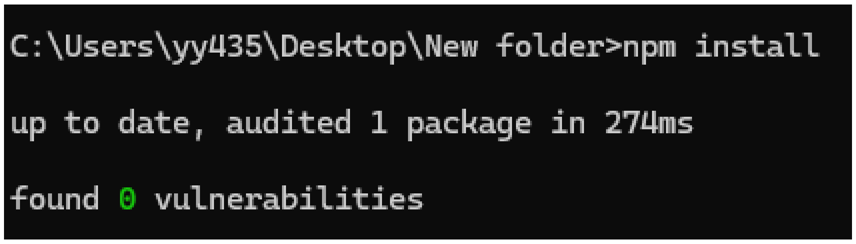
```javascript
require('rootpath')();
const express = require('express');
const app = express();
const cors = require('cors');
const errorHandler = require('./_middleware/error-handler');

app.use(express.urlencoded({ extended: true }));
app.use(express.json());
app.use(cors());

app.use('/users', require('./users/users.controller'));

app.use(errorHandler);

const port = process.env.NODE_ENV === "production" ? (process.env.PORT || 80) : 4000;

app.listen(port, () => console.log(`Server listening on port ${port}`));
```

```
    at Object..js (node:internal/modules/cjs/loader:1699:10)
    at Module.load (node:internal/modules/cjs/loader:1313:32) {
```

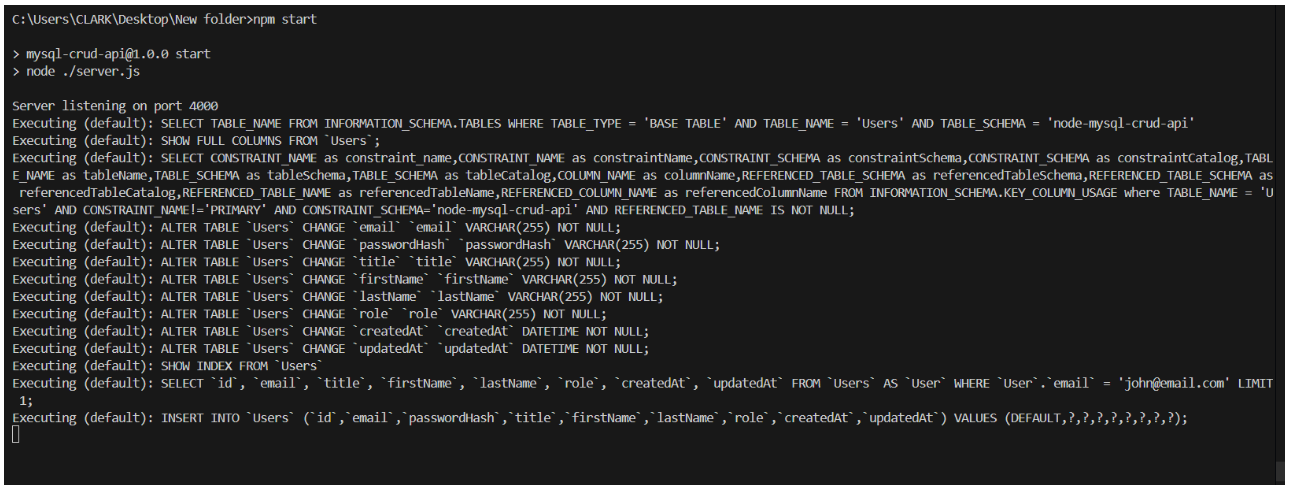
Package.json



```
C:\Users\yy435\Desktop\New folder>npm init -y
Wrote to C:\Users\yy435\Desktop\New folder\package.json:

{
  "name": "new-folder",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

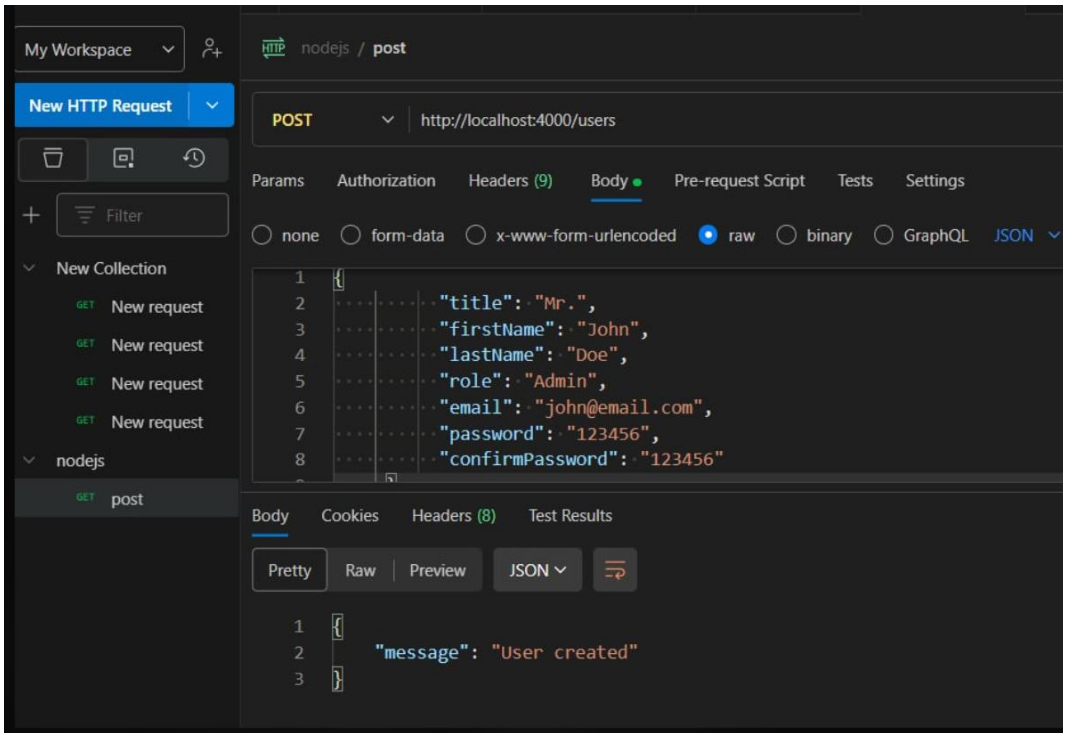


```
C:\Users\yy435\Desktop\New folder>npm install

up to date, audited 1 package in 274ms

found 0 vulnerabilities
```



```
C:\Users\CLARK\Desktop\New folder>npm start

> mysql-crud-api@1.0.0 start
> node ./server.js

Server listening on port 4000
Executing (default): SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_NAME = 'Users' AND TABLE_SCHEMA = 'node-mysql-crud-api'
Executing (default): SHOW FULL COLUMNS FROM `Users`;
Executing (default): SELECT CONSTRAINT_NAME as constraint_name,CONSTRAINT_NAME as constraintName,CONSTRAINT_SCHEMA as constraintSchema,CONSTRAINT_SCHEMA as constraintCatalog,TABL
E_NAME as tableName,TABLE_SCHEMA as tableSchema,TABLE_SCHEMA as tableCatalog,COLUMN_NAME as columnName,REFERENCED_TABLE_SCHEMA as referencedTableSchema,REFERENCED_TABLE_SCHEMA as
 referencedTableCatalog,REFERENCED_TABLE_NAME as referencedTableName,REFERENCED_COLUMN_NAME as referencedColumnName FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE where TABLE_NAME = 'U
sers' AND CONSTRAINT_NAME!='PRIMARY' AND CONSTRAINT_SCHEMA='node-mysql-crud-api' AND REFERENCED_TABLE_NAME IS NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `email` `email` VARCHAR(255) NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `passwordHash` `passwordHash` VARCHAR(255) NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `title` `title` VARCHAR(255) NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `firstName` `firstName` VARCHAR(255) NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `lastName` `lastName` VARCHAR(255) NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `role` `role` VARCHAR(255) NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `createdAt` `createdAt` DATETIME NOT NULL;
Executing (default): ALTER TABLE `Users` CHANGE `updatedAt` `updatedAt` DATETIME NOT NULL;
Executing (default): SHOW INDEX FROM `Users`
Executing (default): SELECT `id`, `email`, `title`, `firstName`, `lastName`, `role`, `createdAt`, `updatedAt` FROM `Users` AS `User` WHERE `User`.`email` = 'john@email.com' LIMIT
 1;
Executing (default): INSERT INTO `Users` (`id`,`email`,`passwordHash`,`title`,`firstName`,`lastName`,`role`,`createdAt`,`updatedAt`) VALUES (DEFAULT,?,?,?,?,?,?,?,?);
```

Using Postman
I can POST //localhost:4000/users and Send and result is



POST http://localhost:4000/users

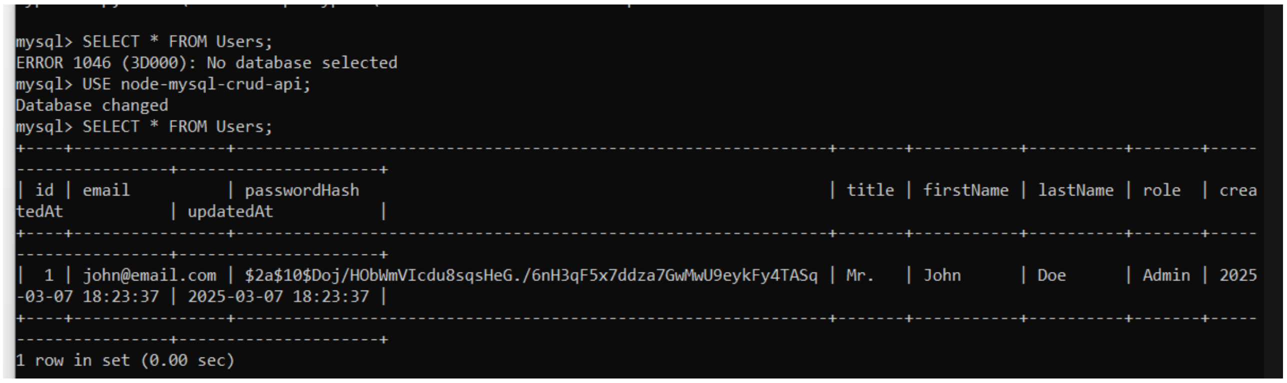
Body: [✓ raw] JSON

```
{
        "title": "Mr.",
        "firstName": "John",
        "lastName": "Doe",
        "role": "Admin",
        "email": "john@email.com",
        "password": "123456",
        "confirmPassword": "123456"
```

```
{
    "message": "User created"
}
```

Then I trace it to my Mysql

USE node-mysql-crud-api;
Database changed
mysql> SELECT * FROM Users;



```
mysql> SELECT * FROM Users;
ERROR 1046 (3D000): No database selected
mysql> USE node-mysql-crud-api;
Database changed
mysql> SELECT * FROM Users;
+----+----------------+--------------------------------------------------------------+-------+-----------+----------+-------+-----
----------------+---------------------+
| id | email          | passwordHash                                                 | title | firstName | lastName | role  | crea
tedAt            | updatedAt           |
+----+----------------+--------------------------------------------------------------+-------+-----------+----------+-------+-----
----------------+---------------------+
|  1 | john@email.com | $2a$10$Doj/HObWmVIcdu8sqsHeG./6nH3qF5x7ddza7GwMwU9eykFy4TASq | Mr.   | John      | Doe      | Admin | 2025
-03-07 18:23:37 | 2025-03-07 18:23:37 |
+----+----------------+--------------------------------------------------------------+-------+-----------+----------+-------+-----
----------------+---------------------+
1 row in set (0.00 sec)
```