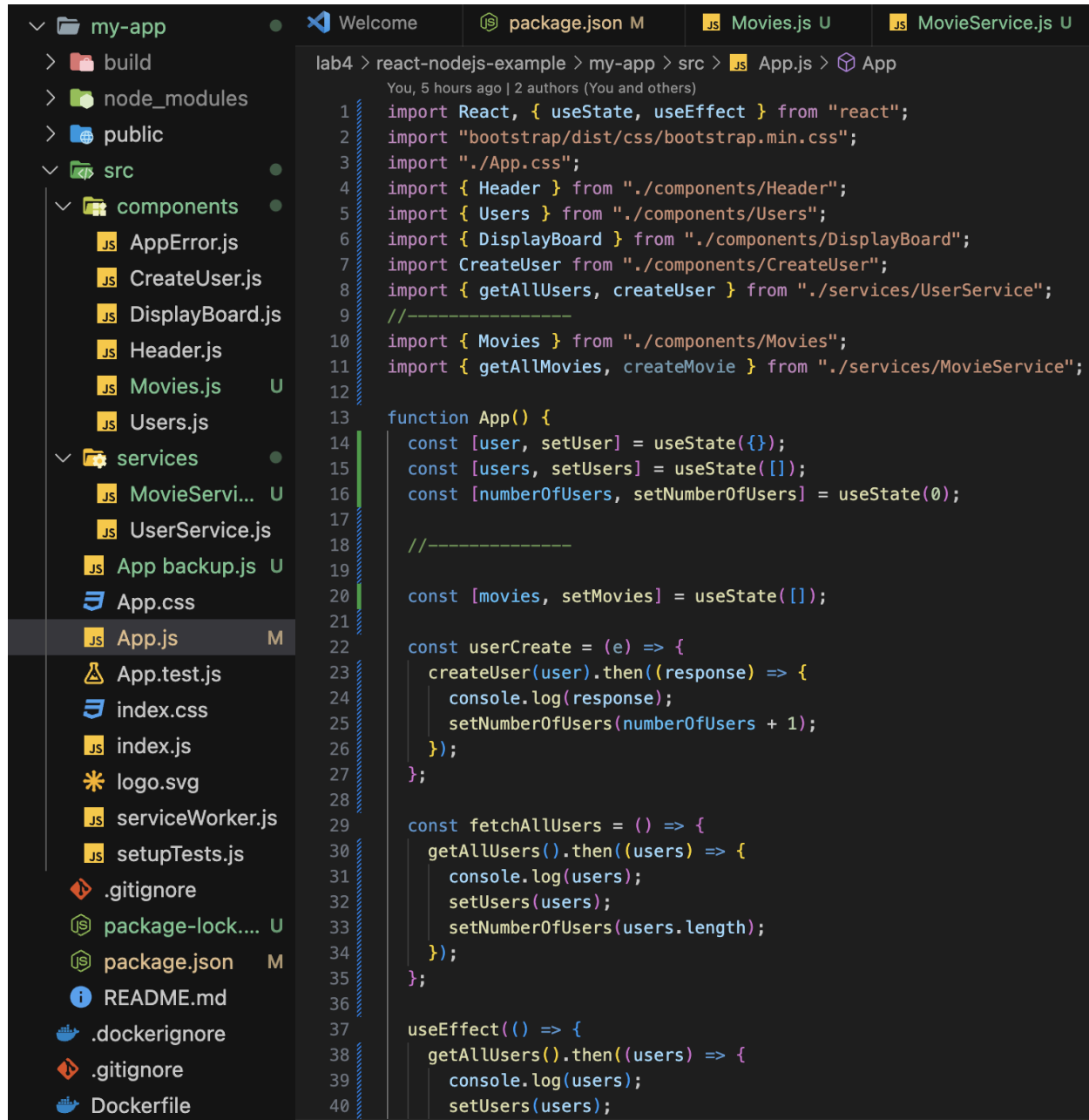


ฝั่ง frontend



The image shows a VS Code editor interface. On the left, a file explorer displays the project structure for 'my-app'. The 'src' directory contains 'components' and 'services' folders. 'components' includes 'AppError.js', 'CreateUser.js', 'DisplayBoard.js', 'Header.js', 'Movies.js' (marked with a 'U'), and 'Users.js'. 'services' includes 'MovieService.js' (marked with a 'U') and 'UserService.js'. Other files in 'src' are 'App backup.js' (marked with a 'U'), 'App.css', 'App.js' (marked with an 'M'), 'App.test.js', 'index.css', 'index.js', 'logo.svg', 'serviceWorker.js', and 'setupTests.js'. The main editor area shows the 'App.js' file with the following code:

```
lab4 > react-nodejs-example > my-app > src > JS App.js > App
You, 5 hours ago | 2 authors (You and others)
1  import React, { useState, useEffect } from "react";
2  import "bootstrap/dist/css/bootstrap.min.css";
3  import "./App.css";
4  import { Header } from "../components/Header";
5  import { Users } from "../components/Users";
6  import { DisplayBoard } from "../components/DisplayBoard";
7  import CreateUser from "../components/CreateUser";
8  import { getAllUsers, createUser } from "../services/UserService";
9  //-----
10 import { Movies } from "../components/Movies";
11 import { getAllMovies, createMovie } from "../services/MovieService";
12
13 function App() {
14   const [user, setUser] = useState({});
15   const [users, setUsers] = useState([]);
16   const [numberOfUsers, setNumberOfUsers] = useState(0);
17
18   //-----
19
20   const [movies, setMovies] = useState([]);
21
22   const userCreate = (e) => {
23     createUser(user).then((response) => {
24       console.log(response);
25       setNumberOfUsers(numberOfUsers + 1);
26     });
27   };
28
29   const fetchAllUsers = () => {
30     getAllUsers().then((users) => {
31       console.log(users);
32       setUsers(users);
33       setNumberOfUsers(users.length);
34     });
35   };
36
37   useEffect(() => {
38     getAllUsers().then((users) => {
39       console.log(users);
40       setUsers(users);
```

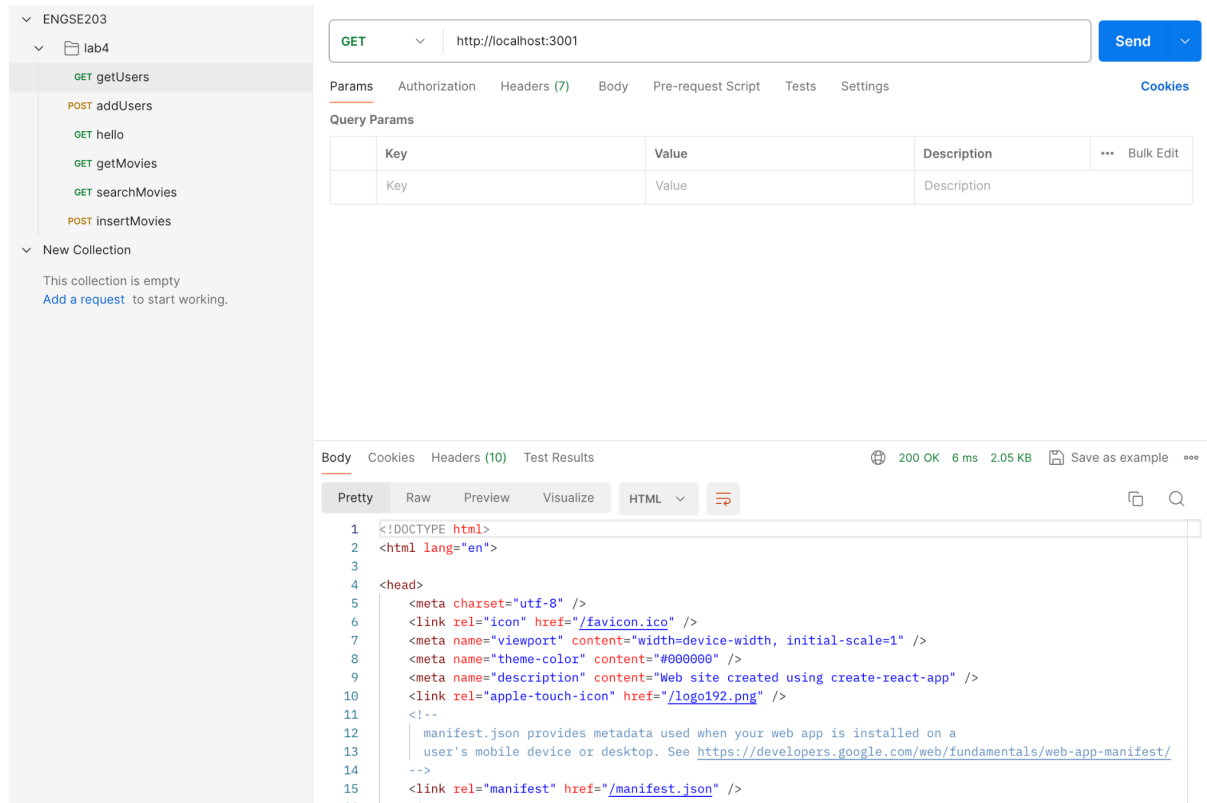
รูปที่ 1 app.js ไม่ต้องแก้ไข

```

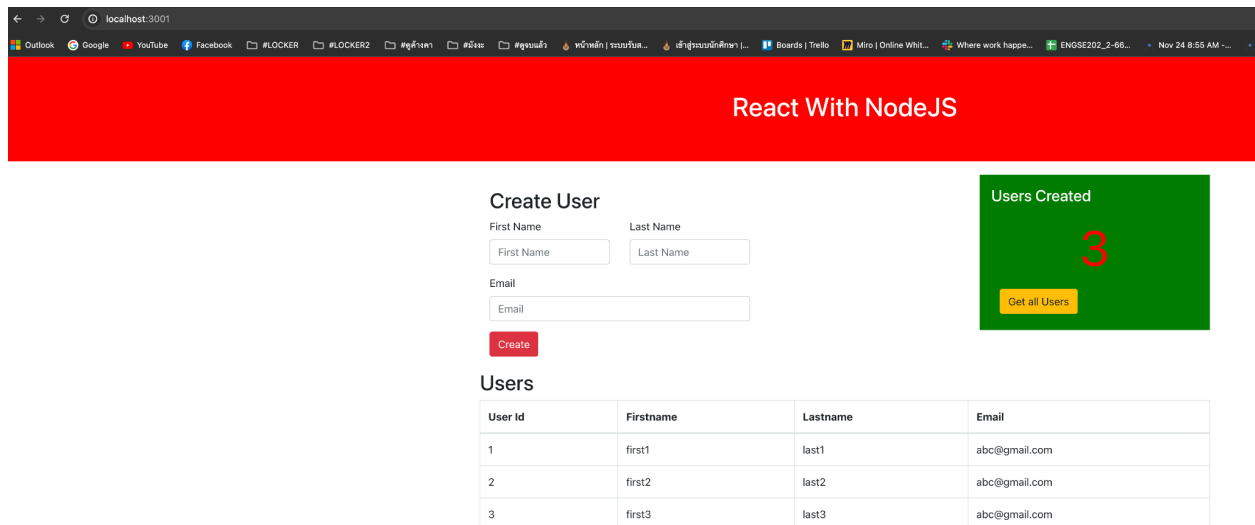
lab4 > react-nodejs-example > my-app > package.json > {} dependencies
You, 4 hours ago | 2 authors (bbachi and others)
1  {
2    "name": "my-app",
3    "version": "0.1.0",
4    "private": true,
5    "dependencies": {
6      "@testing-library/jest-dom": "^4.2.4",
7      "@testing-library/react": "^9.5.0",
8      "@testing-library/user-event": "^7.2.1",
9      "bootstrap": "^4.5.0",
10     "react": "^16.13.1",
11     "react-bootstrap": "^1.0.1",
12     "react-dom": "^16.13.1",
13     "react-scripts": "3.4.1"
14   },
15   "scripts": {
16     "start": "PORT=3001 react-scripts start",
17     "build": "react-scripts build",
18     "test": "react-scripts test",
19     "eject": "react-scripts eject"
20   },
21   "proxy": "http://localhost:4000",
22   "eslintConfig": {
23     "extends": "react-app"
24   },
25   "browserslist": {
26     "production": [
27       ">0.2%",
28       "not dead",
29       "not op_mini all"
30     ],
31     "development": [
32       "last 1 chrome version",
33       "last 1 firefox version",
34       "last 1 safari version"
35     ]
36   }
37 }

```

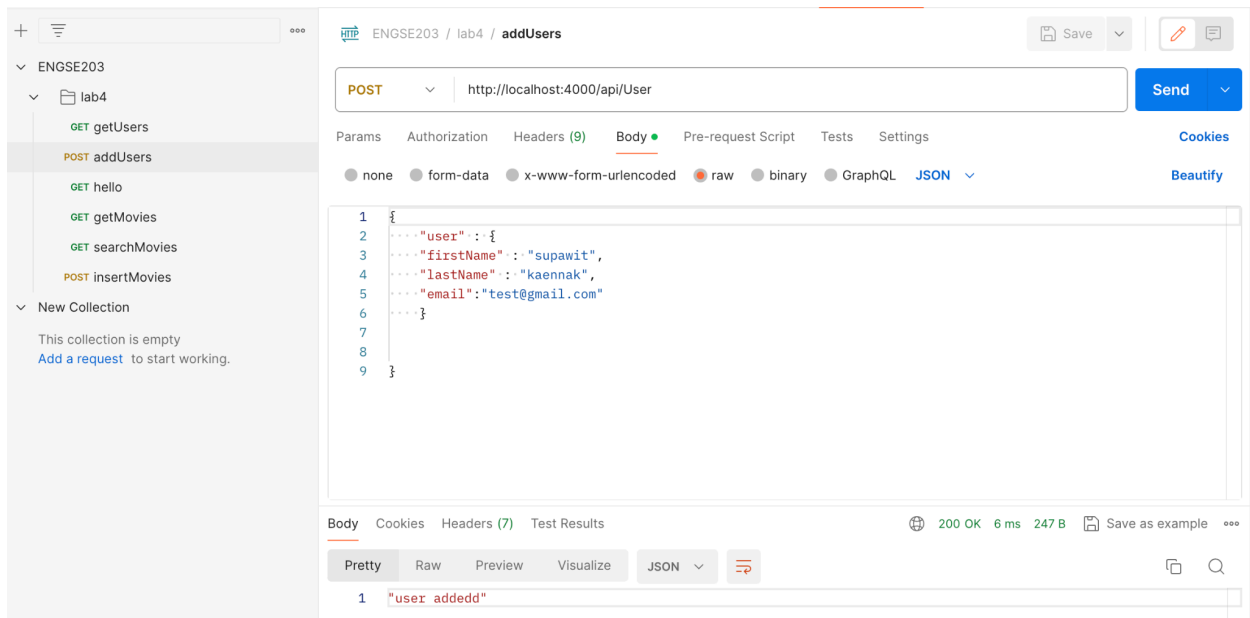
รูปที่ 2 package.json เปลี่ยน port เป็น 4000



รูปที่ 3 Postman ใช้ทดสอบ get ข้อมูลจาก ตาราง users



รูปที่ 4 ผลลัพธ์หน้า react



รูปที่ 5 ใช้ postman ทดสอบ post ข้อมูล โดยเขียนเป็น ไฟล์ json

## React With NodeJS

### Create User

First Name

Last Name

Email

Create

Users Created

9

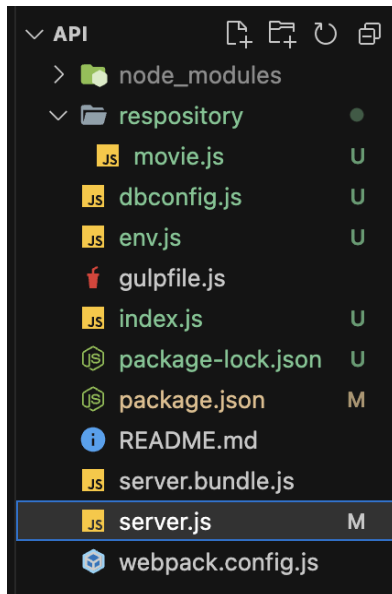
Get all Users

### Users

User Id	Firstname	Lastname	Email
1	first1	last1	abc@gmail.com
2	first2	last2	abc@gmail.com
3	first3	last3	abc@gmail.com
4	test	test	test
5	test	test	test
6	test	test	test
7	test	test	test
8	test	test	test
9	supawit	kaennak	test@gmail.com

รูปที่ 6 ผลลัพธ์เมื่อทดสอบ post ค่า และทำการกด get all users

## ส่วน API



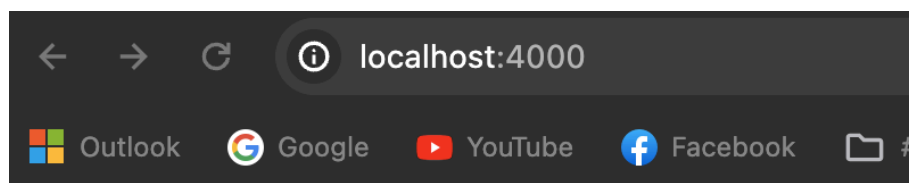
```
server.js > [?] users
You, 4 hours ago | 2 authors (bbachi and others)
1  const express = require('express');
2  const path = require('path');
3  const app = express(),
4      bodyParser = require("body-parser");
5      port = 4000;
6
7  // place holder for the data
8  const users = [
9      {
10         firstName: "first1",
11         lastName: "last1",
12         email: "abc@gmail.com"
13     },
14     {
15         firstName: "first2",
16         lastName: "last2",
17         email: "abc@gmail.com"
18     },
19     {
20         firstName: "first3",
21         lastName: "last3",
22         email: "abc@gmail.com"
23     }
24 ];
25
26 app.use(bodyParser.json());
27 app.use(express.static(path.join(__dirname, '../my-app/build')));
28
29 app.get('/api/users', (req, res) => {
30     console.log('api/users called!')
31     res.json(users);
32 });
33
34 app.post('/api/user', (req, res) => {
35     const user = req.body.user;
36     console.log('Adding user::::', user);
37     users.push(user);
38     res.json("user addedd");
39 });
```

รูปที่ 7 ไฟล์ server.js แก้ไข port เป็น 4000

```
package.json > {} scripts > dev
You, 1 second ago | 2 authors (bbachi and others)

1  {
2    "name": "react-nodejs-example",
3    "version": "1.0.0",
4    "description": "example project react with nodejs",
5    "main": "server.js",
6    "scripts": {
7      "start": "node server.bundle.js",
8      "build": "webpack",
9      "dev": "nodemon ./index.js localhost 4000"
10   },
11   "repository": {
12     "type": "git",
13     "url": "git+https://github.com/bbachi/react-nodejs-example.git"
14   },
15   "author": "Bhargav Bachina",
16   "license": "ISC",
17   "bugs": {
18     "url": "https://github.com/bbachi/react-nodejs-example/issues"
19   },
20   "homepage": "https://github.com/bbachi/react-nodejs-example#readme",
21   "dependencies": {
22     "@hapi/hapi": "^21.3.2",
23     "@hapi/inert": "^7.1.0",
24     "express": "^4.17.1",
25     "http-proxy-middleware": "^1.0.4",
26     "mysql": "^2.18.1"
27   },
28   "devDependencies": {
29     "del": "^5.1.0",
30     "fancy-log": "^1.3.3",
31     "gulp": "^4.0.2",
32     "gulp-zip": "^5.0.1",
33     "nodemon": "^2.0.4",
34     "webpack": "^4.43.0",
35     "webpack-cli": "^3.3.11",
36     "webpack-stream": "^5.2.1"
37   }
38 }
```

รูปที่ 8 ไฟล์ package.json แก้ ตรง dev เปลี่ยนจาก server.js เป็น index.js

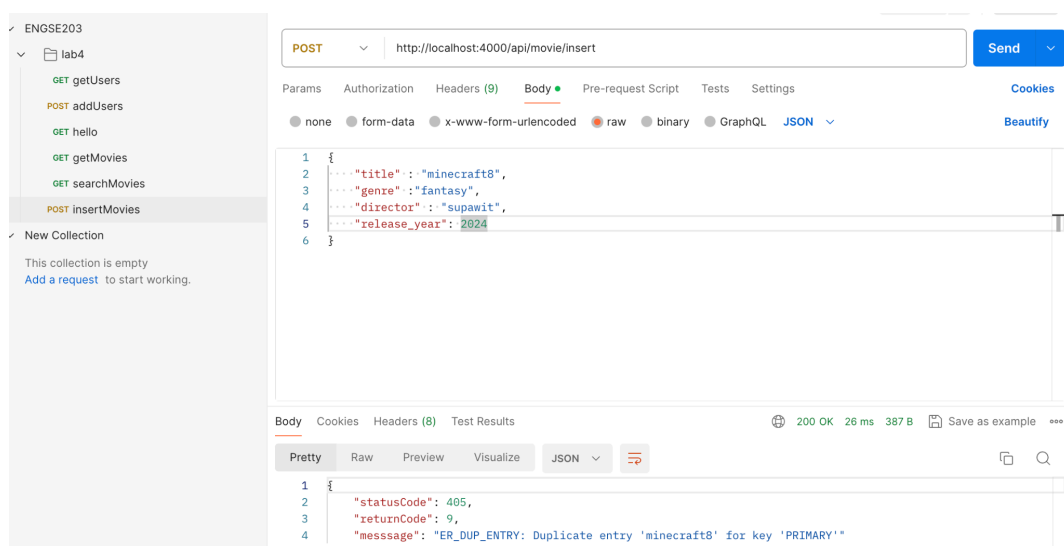


**Welcome to API Back-end Ver. 1.0.0**

รูปที่ 9 ทดสอบเข้าเว็บ

```
server.js M  env.js U  movie.js U X  dbconfig.js U  index.js U  pac
respository >  movie.js > postMovie > <function>
125   };
126
127   console.log('post is: ', post);
128
129
130   Query = 'INSERT INTO movies SET ?';
131   pool.query(Query, post, function (error, results, fields) {
132     //pool.query(Query, function (error, results, fields) {
133
134
135
136     // if (error) throw error;
137
138     if (error) {
139       console.log('error_code_msg', error.code+': '+error.sqlMessage);
140       pool.end();
141       return resolve({
142         statusCode: 405,
143         returnCode: 9,
144         message: error.code+': '+error.sqlMessage,
145       });
146     }
147     else{
148       console.log('results: ', results);
149       if(results.affectRows > 0){
150         pool.end();
151         return resolve({
152           statusCode: 200,
153           returnCode: 1,
154           message: 'Movie list was inserted'
155         });
156       }
157     }
158   });
159 }
160 }
```

รูปที่ 10 โค้ดในไฟล์ movie.js แก่ตามรูปภาพ



รูปที่ 11 postman ใช้เช็คการ post ข้อมูล

## กลับมาที่ส่วน frontend

```
lab4 > react-nodejs-example > my-app > src > components > JS Movies.js > [0] Movies > [0] Movies.js
1  import React from 'react'
2
3  export const Movies = ({movies}) => {
4
5      console.log('movies length:::', movies.length)
6      if (movies.length === 0) return null
7
8      const MovieRow = (movie, index) => {
9
10         return(
11             <tr key = {index} className={index%2 === 0?'odd':'even'}>
12                 <td>{index + 1}</td>
13                 <td>{movie.title}</td>
14                 <td>{movie.genre}</td>
15                 <td>{movie.director}</td>
16                 <td>{movie.release_year}</td>
17             </tr>
18         )
19     }
20
21     const movieTable = movies.map((movie, index) => MovieRow(movie, index))
22
23     return(
24         <div className="container">
25             <h2>Movies</h2>
26             <table className="table table-bordered">
27                 <thead>
28                     <tr>
29                         <th>Movie Id</th>
30                         <th>Title</th>
31                         <th>Genre</th>
32                         <th>Director</th>
33                         <th>Release</th>
34                     </tr>
35                 </thead>
36                 <tbody>
37                     {movieTable}
38                 </tbody>
39             </table>
40         </div>
41     )
42 }
```

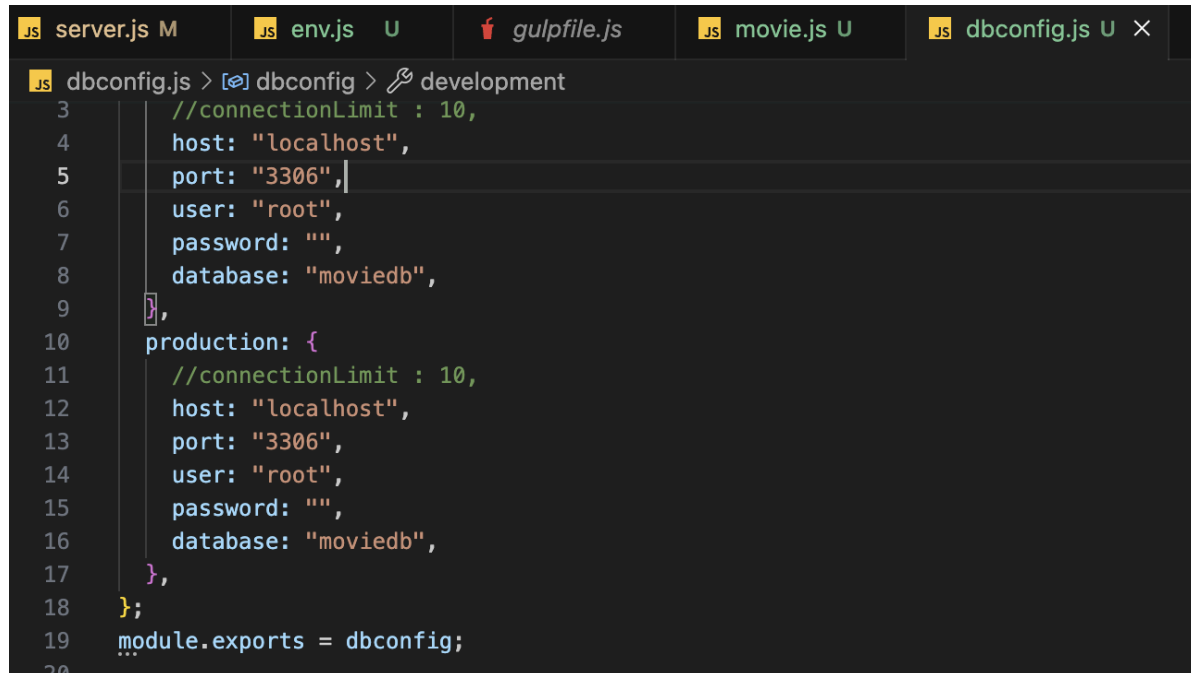
## รูปที่ 12 โค้ดในไฟล์ .js แก่ตามรูปภาพ

```
lab4 > react-nodejs-example > my-app > src > services > JS MovieService.js > ...
1  export async function getAllMovies() {
2
3      try{
4          //const response = await fetch('/api/users');
5          const response = await fetch('http://10.2.6.100:3002/api/movie/all');
6          //const response = await fetch('/api/movie/all');
7          return await response.json();
8      }catch(error) {
9          return [];
10     }
11 }
12
13
14 export async function createMovie(data) {
15     const response = await fetch(`/api/user`, {
16         method: 'POST',
17         headers: {'Content-Type': 'application/json'},
18         body: JSON.stringify({user: data})
19     })
20     return await response.json();
21 }
22
```

## รูปที่ 13 โค้ดในไฟล์ movieservice.js แก่ตามรูปภาพ

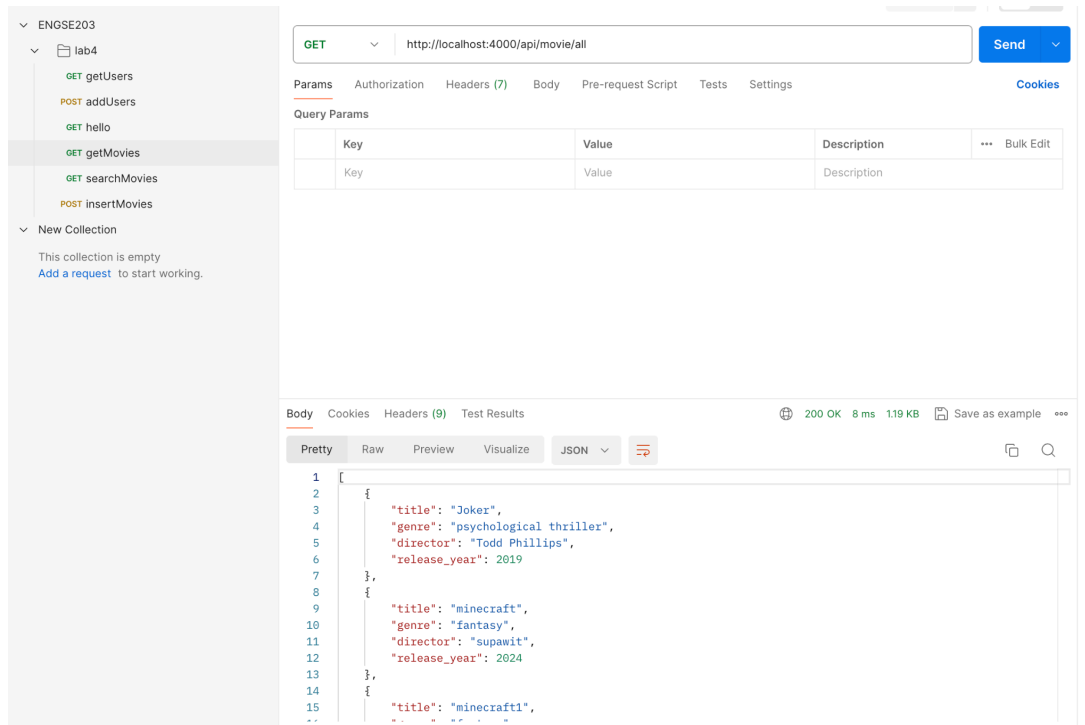


กลับมาที่ส่วน API

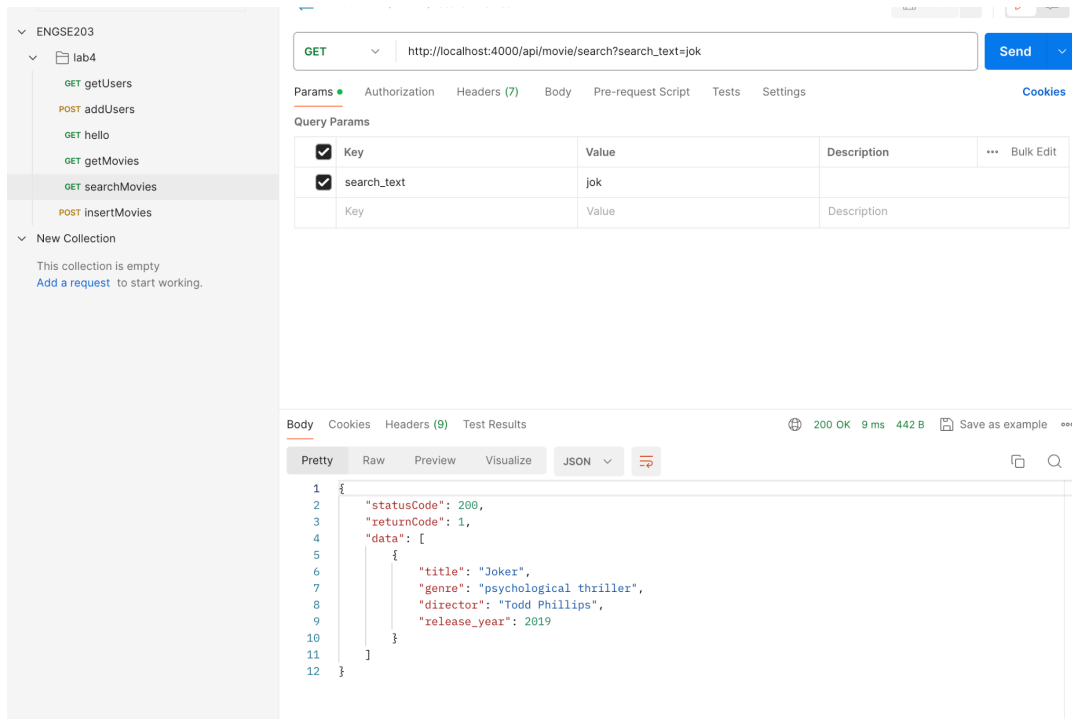


```
server.js M  env.js U  gulpfile.js  movie.js U  dbconfig.js U X
dbconfig.js > [?] dbconfig > development
3 //connectionLimit : 10,
4 host: "localhost",
5 port: "3306",
6 user: "root",
7 password: "",
8 database: "moviedb",
9 },
10 production: {
11 //connectionLimit : 10,
12 host: "localhost",
13 port: "3306",
14 user: "root",
15 password: "",
16 database: "moviedb",
17 },
18 };
19 module.exports = dbconfig;
20
```

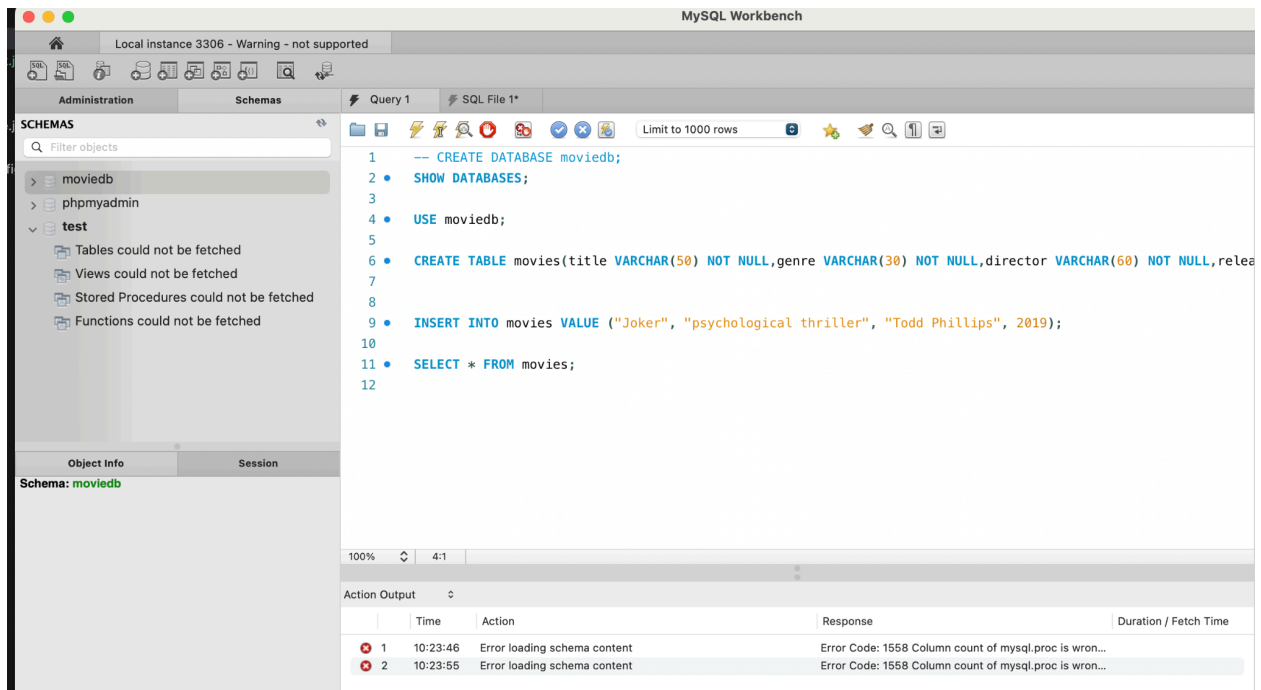
รูปที่ 14 โค้ดในไฟล์ dbconfig.js แก่ตามรูปภาพ



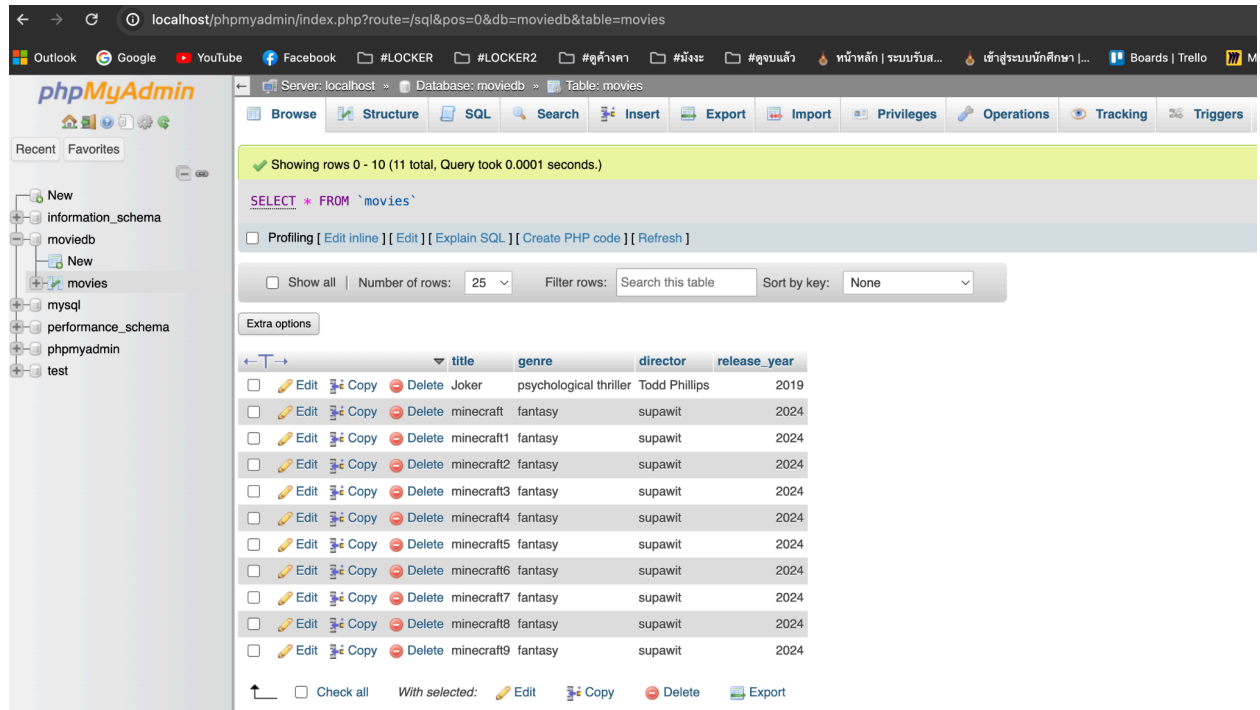
รูปที่ 15 postman ใช้เช็คการ get ข้อมูล ของ api/movie/all



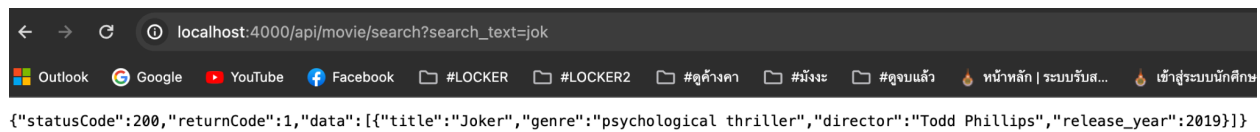
รูปที่ 16 postman ใช้ get ข้อมูลของ api/movie/search



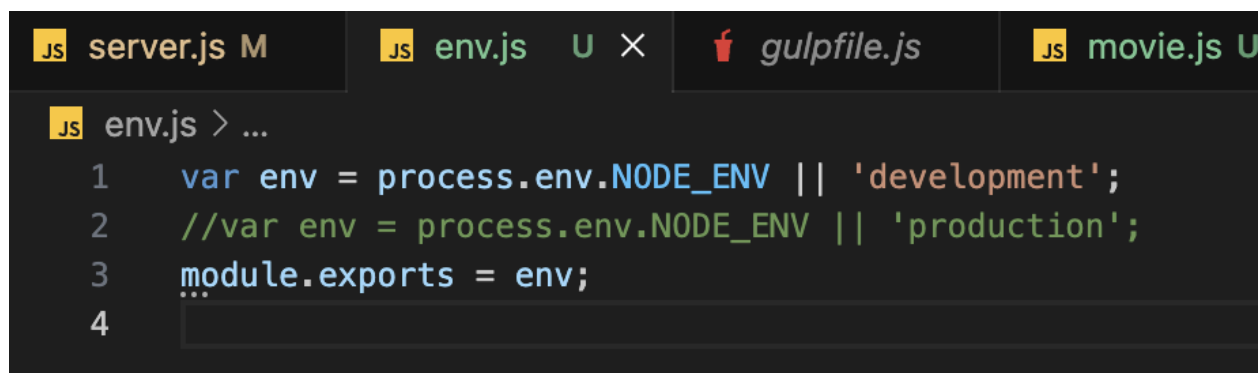
รูปที่ 17 mysql workbench ใช้ในการสร้างตาราง และ แทรกข้อมูล



รูปที่ 18 phpmyadmin เปิดไว้เปรียบเทียบกับ mysql workbench



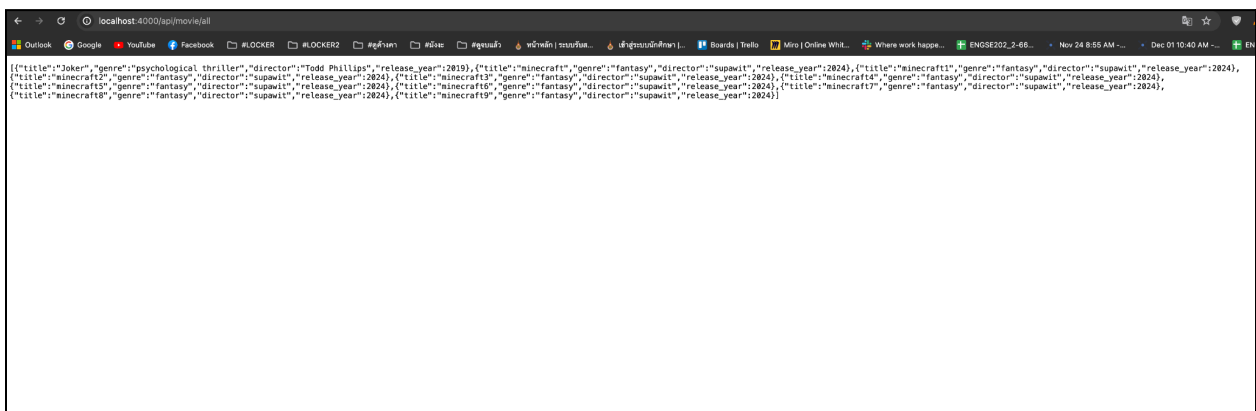
รูปที่ 19 ลองเช็ค api ผ่าน browser ดู



รูปที่ 20 โค้ดในไฟล์ env.js แก่ตามรูปภาพ

```
server.js M  env.js U  movie.js U  dbconfig.js U  server.bundle.js  index.js U X
index.js > ...
1  const hapi = require('@hapi/hapi');
2  const env = require('./env.js');
3  const Movies = require('./respository/movie');
4
5  const express = require('express');
6  const app = express();
7
8  const path = require('path');
9  |   bodyParser = require("body-parser");
10
11 //-----
12 const api_port = 4000;
13 const web_port = 4001;
14 |
15 //----- express -----
16
17 // place holder for the data
18 const users = [
19   {
20     firstName: "first1",
21     lastName: "last1",
22     email: "abc@gmail.com"
23   },
24   {
25     firstName: "first2",
26     lastName: "last2",
27     email: "abc@gmail.com"
28   },
29   {
30     firstName: "first3",
31     lastName: "last3",
32     email: "abc@gmail.com"
33   },
34   {
35     firstName: "first4",
36     lastName: "last4",
37     email: "abc4@gmail.com"
38   }
39 ];
40
```

รูปที่ 21 โค้ดในไฟล์ index.js แก่ตามรูปภาพ



รูปที่ 22 เช็ค api/movie/all ผ่าน browser ดู