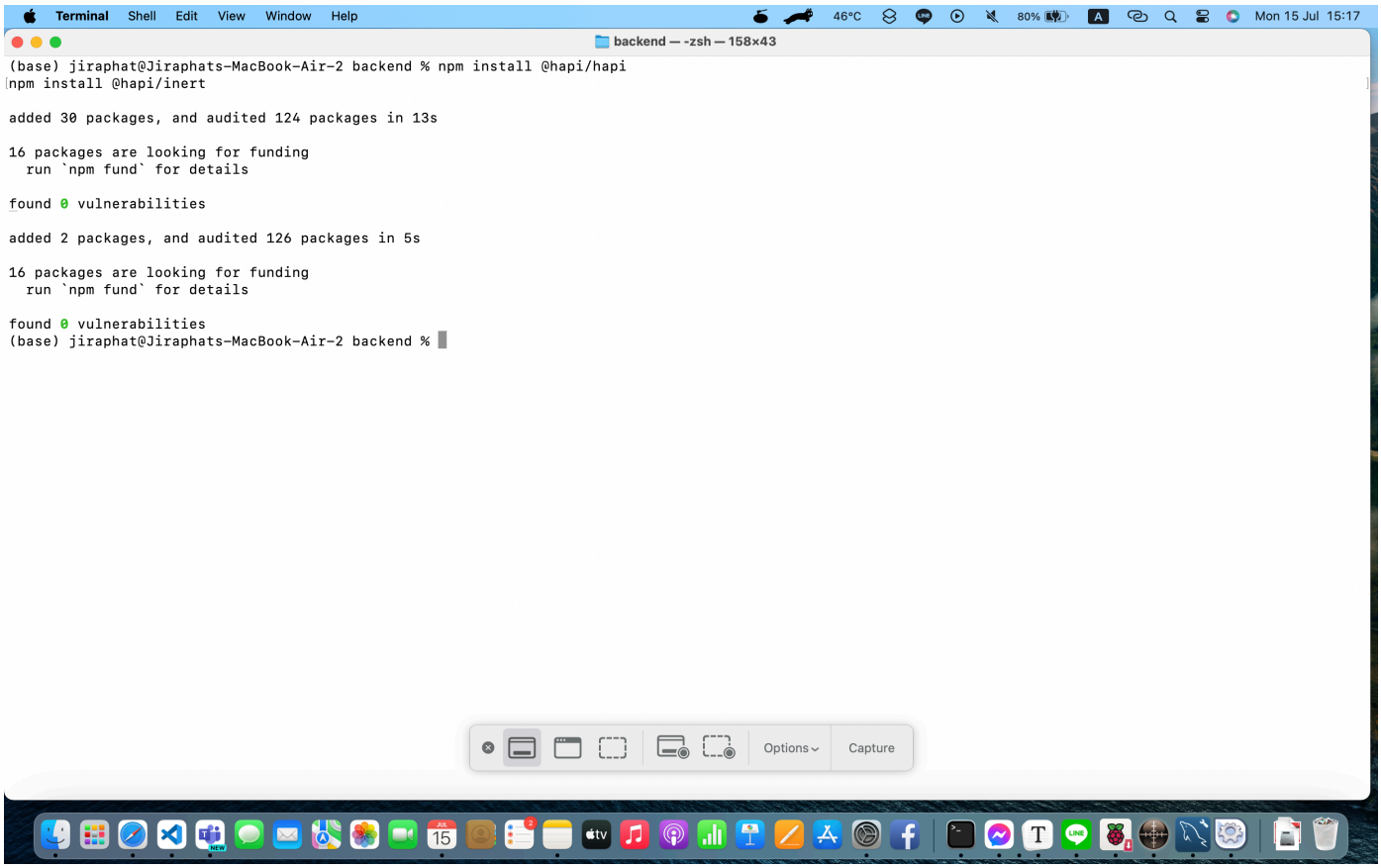
Lab3 ResfulAPI Server with Nodejs

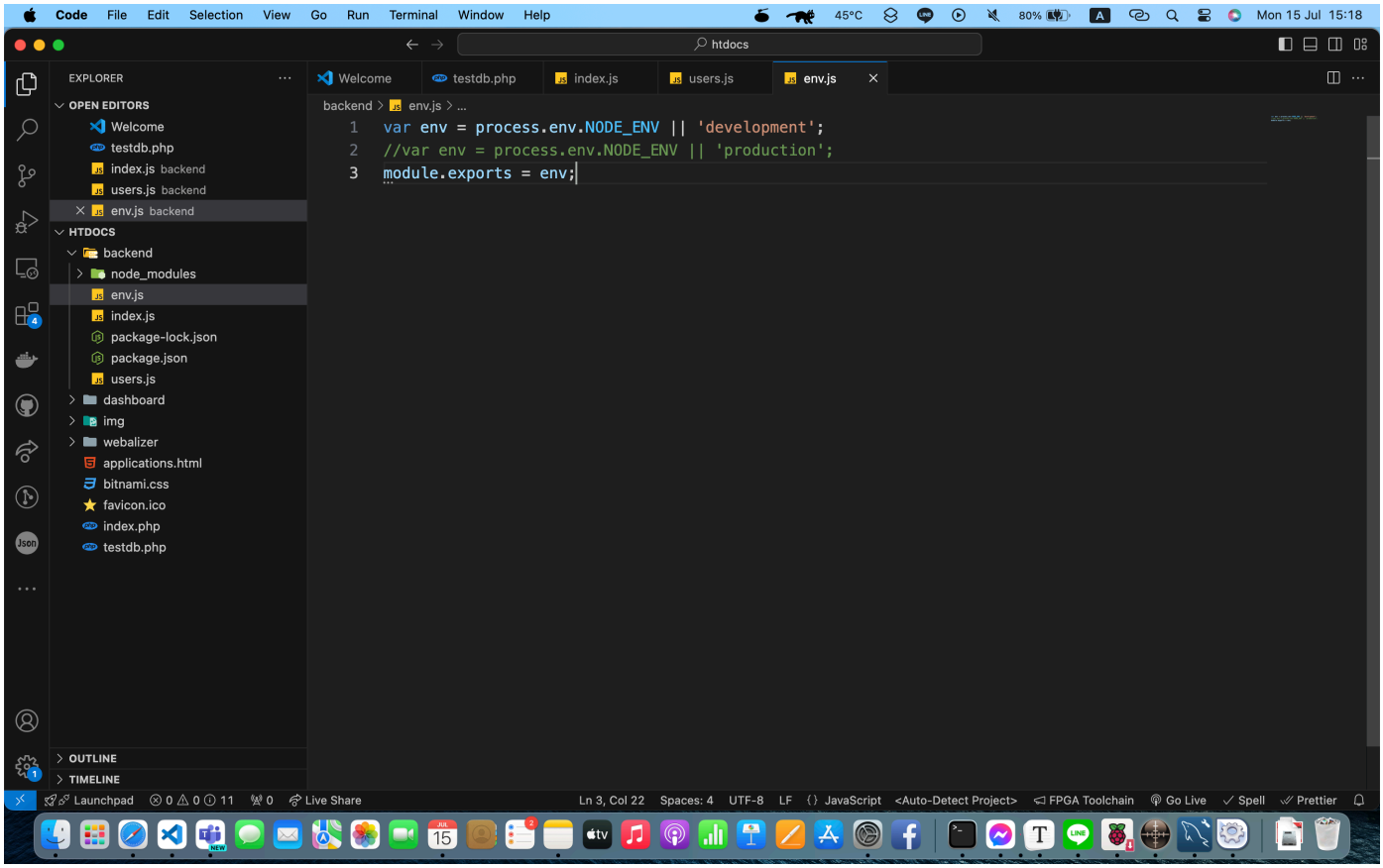
ติดตั้ง hapi API package

```
npm install @hapi/hapi
npm install @hapi/inert
```



สร้างไฟล์ env.js เพื่อเก็บ config ว่าทำงานอยู่ในสภาพแวดล้อมใด

```
var env = process.env.NODE_ENV || 'development';
//var env = process.env.NODE_ENV || 'production';
module.exports = env;
```



ปรับปรุง index.js

```
const hapi = require('@hapi/hapi');
var users = require('./users');
const env = require('./env.js');
const Movies = require('./respository/movie');

const express = require('express');
const app = express();

//------------------
const web_port = 3010;
const api_port = 3001;


//------------- express ----------
app.get('/', (req, res) => {
  res.send('<h1> Hello World main_server </h1>');

})

app.get('/users', function (req, res) {
  res.json(users.findAll());
});

app.get('/user/:id', function (req, res) {
  var id = req.params.id;
  res.json(users.findById(id));
});
```

```
app.listen(web_port, () => {
  console.log('Start web server at port ' + web_port);

})

//------------ hapi --------------

console.log('Running Environment: ' + env);


const init = async () => {

  const server = hapi.Server({
    port: api_port,
    host: '0.0.0.0',
    routes: {
      cors: true
    }
  });
  //---------

  await server.register(require('@hapi/inert'));

  server.route({
    method: "GET",
    path: "/",
    handler: () => {
      return '<h3> Welcome to API Back-end Ver. 1.0.0</h3>';
    }
  });


    //API: http://localhost:3001/api/movie/all
    server.route({
      method: 'GET',
      path: '/api/movie/all',
      config: {
          cors: {
              origin: ['*'],
              additionalHeaders: ['cache-control', 'x-requested-width']
          }
      },
      handler: async function (request, reply) {
          //var param = request.query;
          //const category_code = param.category_code;

          try {

              const responsedata = await Movies.MovieRepo.getMovieList();
              if (responsedata.error) {
                  return responsedata.errMessage;
              } else {
                  return responsedata;
              }
          } catch (err) {
```

```
            server.log(["error", "home"], err);
            return err;
        }

    }
});

  server.route({
    method: 'GET',
    path: '/api/movie/search',
    config: {
        cors: {
            origin: ['*'],
            additionalHeaders: ['cache-control', 'x-requested-width']
        }
    },
    handler: async function (request, reply) {
        var param = request.query;
        const search_text = param.search_text;
        //const title = param.title;

        try {

          const responsedata = await Movies.MovieRepo.getMovieSearch(search_text);
          if (responsedata.error) {
              return responsedata.errMessage;
          } else {
              return responsedata;
          }
      } catch (err) {
          server.log(["error", "home"], err);
          return err;
      }

    }
});


server.route({
  method: 'POST',
  path: '/api/movie/insert',
  config: {
      payload: {
          multipart: true,
      },
      cors: {
          origin: ['*'],
          additionalHeaders: ['cache-control', 'x-requested-width']
      }
  },
  handler: async function (request, reply) {

      const {
        title,
        genre,
```

```
          director,
          release_year
        } = request.payload;

        //const title = request.payload.title;
        //const genre = request.payload.genre;

        try {

          const responsedata = await Movies.MovieRepo.postMovie(title, genre,
director,release_year);
          if (responsedata.error) {
              return responsedata.errMessage;
          } else {
              return responsedata;
          }
      } catch (err) {
          server.log(["error", "home"], err);
          return err;
      }

    }
});




  await server.start();
  console.log('API Server running on %s', server.info.uri);

  //---------
};


process.on('unhandledRejection', (err) => {

  console.log(err);
  process.exit(1);
});

init();
```

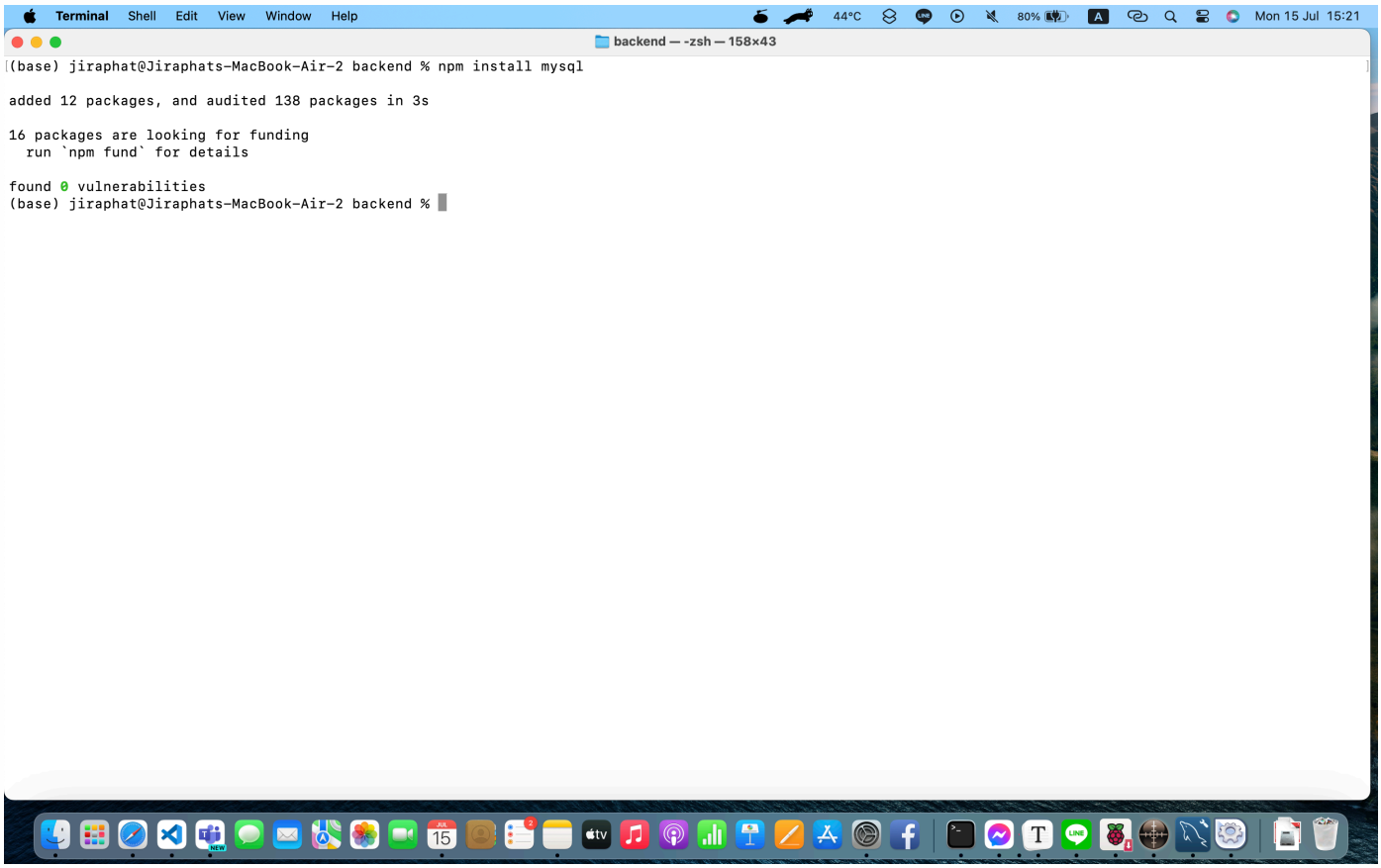
สร้างไฟล์ dbconfig.js เพื่อเก็บ config ใช้สำหรับติดต่อกับ db server (MySQL)

```
var dbconfig = {
    development: {
        //connectionLimit : 10,
        host     : 'localhost',
        port     : '3306',
        user     : 'root',
        password : '',
        database : 'moviedb'
    },
    production: {
```

```
        //connectionLimit : 10,
        host     : 'localhost',
        port     : '3306',
        user     : 'root',
        password : '',
        database : 'moviedb'
    }
    };
module.exports = dbconfig;
```

ติดตั้ง MySQL for Nodejs

```
npm install mysql
```



สร้าง folder **respository**

```
mkdir respository
```

สร้างไฟล์ชื่อ movie.js และใส่ code ตามนี้

```
var mysql = require('mysql');
const env = require('../env.js');
const config = require('../dbconfig.js')[env];

/*
async function getMovieList() {

    var Query;
```

```
    var pool  = mysql.createPool(config);

    return new Promise((resolve, reject) => {

       //Query = `SELECT * FROM movies WHERE warehouse_status = 1 ORDER BY CONVERT(
warehouse_name USING tis620 ) ASC `;
         Query = `SELECT * FROM movies`;

         pool.query(Query, function (error, results, fields) {
            if (error) throw error;

            if (results.length > 0) {
                pool.end();
                return resolve({
                    statusCode: 200,
                    returnCode: 1,
                    data: results,
                });
            } else {
                pool.end();
                return resolve({
                    statusCode: 404,
                    returnCode: 11,
                    message: 'No movie found',
                });
            }

        });

    });


}
*/

async function getMovieList() {

    var Query;
    var pool  = mysql.createPool(config);

    return new Promise((resolve, reject) => {

       //Query = `SELECT * FROM movies WHERE warehouse_status = 1 ORDER BY CONVERT(
warehouse_name USING tis620 ) ASC `;
         Query = `SELECT * FROM movies`;

         pool.query(Query, function (error, results, fields) {
            if (error) throw error;

            if (results.length > 0) {
                pool.end();
                return resolve(results);
            } else {
                pool.end();
                return resolve({
```

```
                    statusCode: 404,
                    returnCode: 11,
                    message: 'No movie found',
                });
            }

        });

    });


}


async function getMovieSearch(search_text) {

    var Query;
    var pool  = mysql.createPool(config);

    return new Promise((resolve, reject) => {

        Query = `SELECT * FROM movies WHERE title LIKE '%${search_text}%'`;

         pool.query(Query, function (error, results, fields) {
            if (error) throw error;

            if (results.length > 0) {
                pool.end();
                return resolve({
                    statusCode: 200,
                    returnCode: 1,
                    data: results,
                });
            } else {
                pool.end();
                return resolve({
                    statusCode: 404,
                    returnCode: 11,
                    message: 'No movie found',
                });
            }

        });

    });


}

async function postMovie(p_title,p_genre,p_director,p_release_year) {

    var Query;
    var pool  = mysql.createPool(config);

    return new Promise((resolve, reject) => {
```

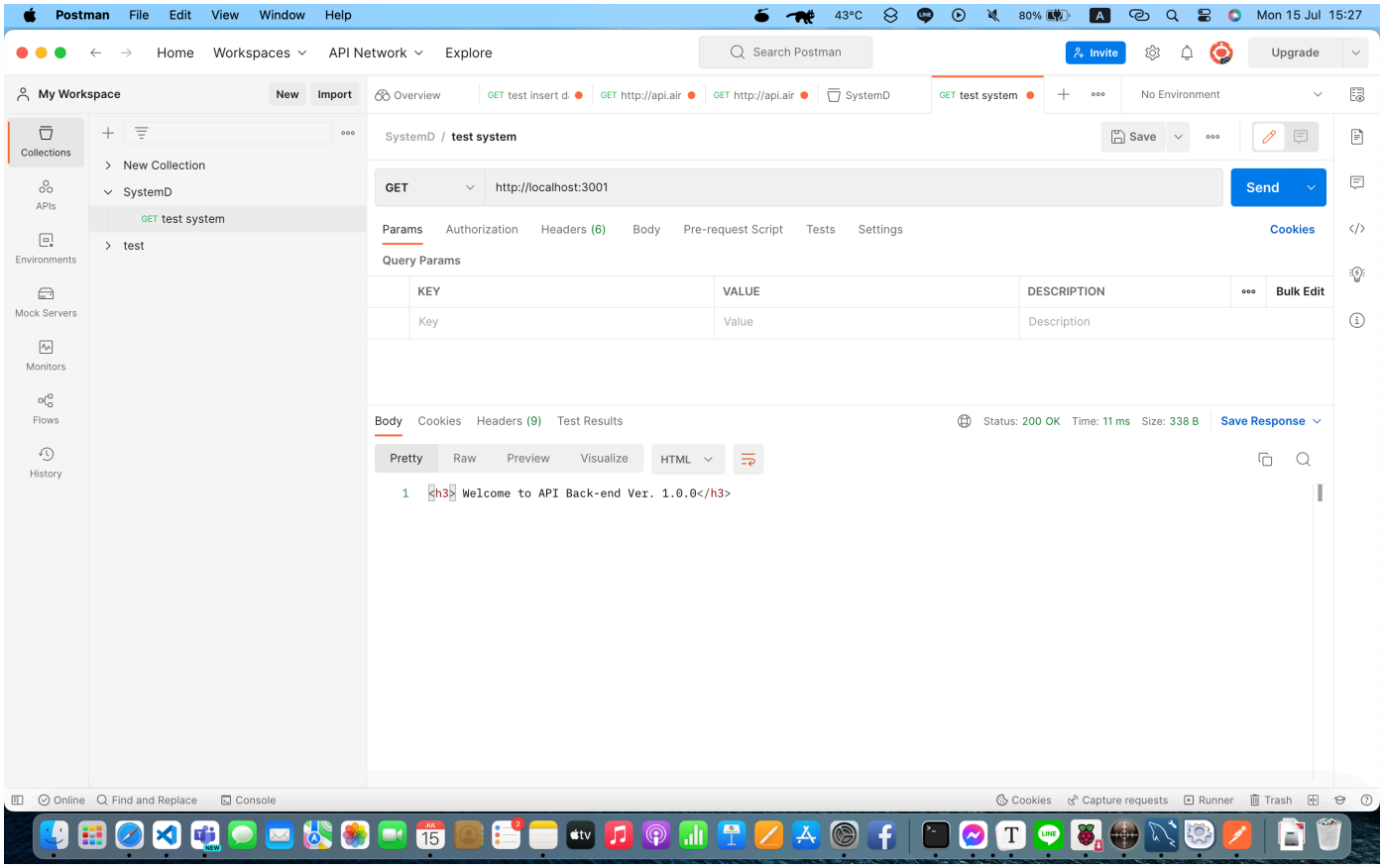
```
        //Query = `SELECT * FROM movies WHERE title LIKE '%${search_text}%'`;

        var post  = {
            title: p_title,
            genre: p_genre,
            director: p_director,
            release_year: p_release_year
        };

        console.log('post is: ', post);


        Query = 'INSERT INTO movies SET ?';
        pool.query(Query, post, function (error, results, fields) {
        //pool.query(Query, function (error, results, fields) {

            if (error) throw error;

            if (results.length > 0) {
                pool.end();
                return resolve({
                    statusCode: 200,
                    returnCode: 1,
                    messsage: 'Movie list was inserted',
                });
            }


        });


    });


}

module.exports.MovieRepo = {
    getMovieList: getMovieList,
    getMovieSearch: getMovieSearch,
    postMovie: postMovie,

};
```

ทดสอบการทำงาน

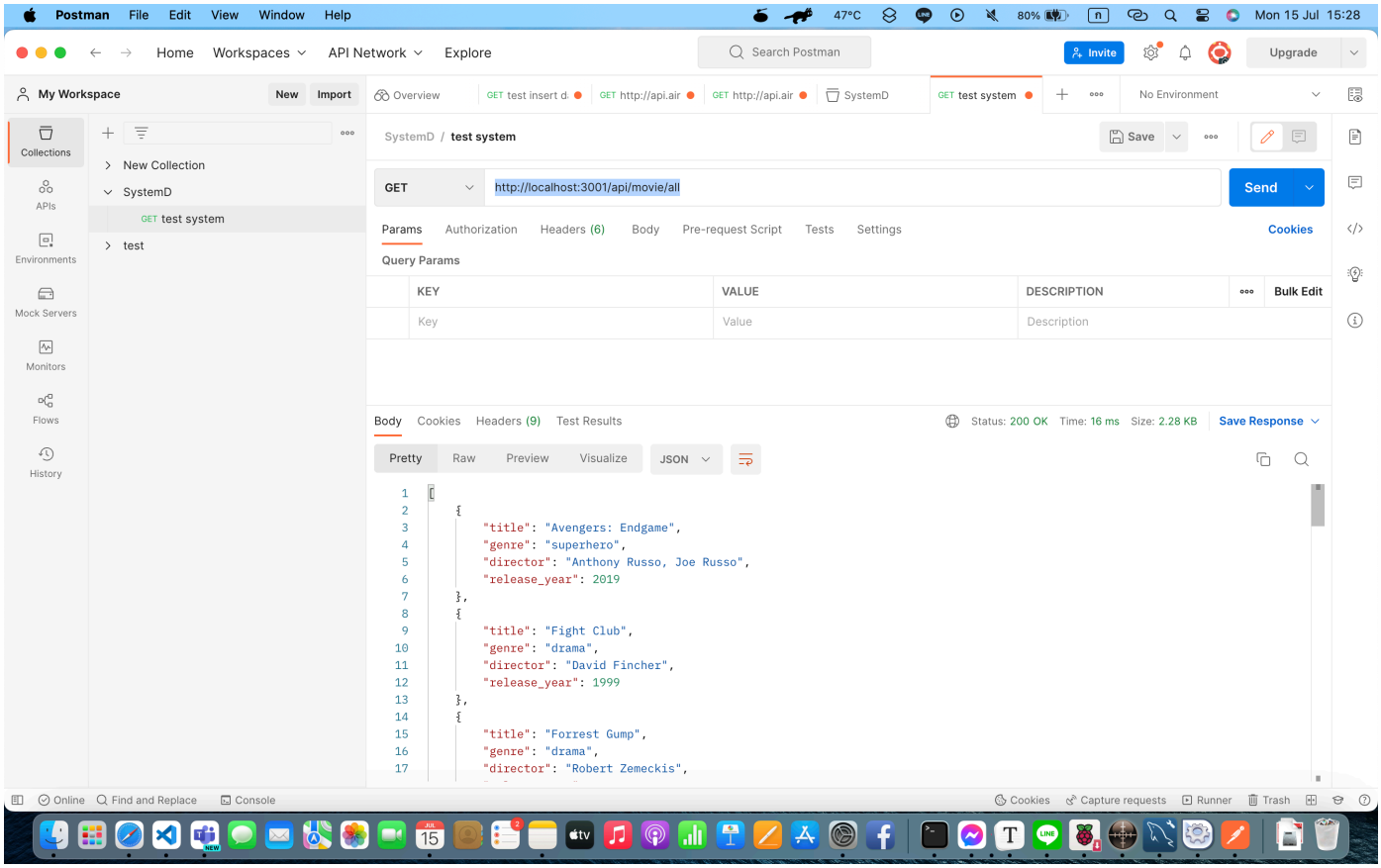
```
cd ..
nodemon index.js
```

โดยใช้ postman

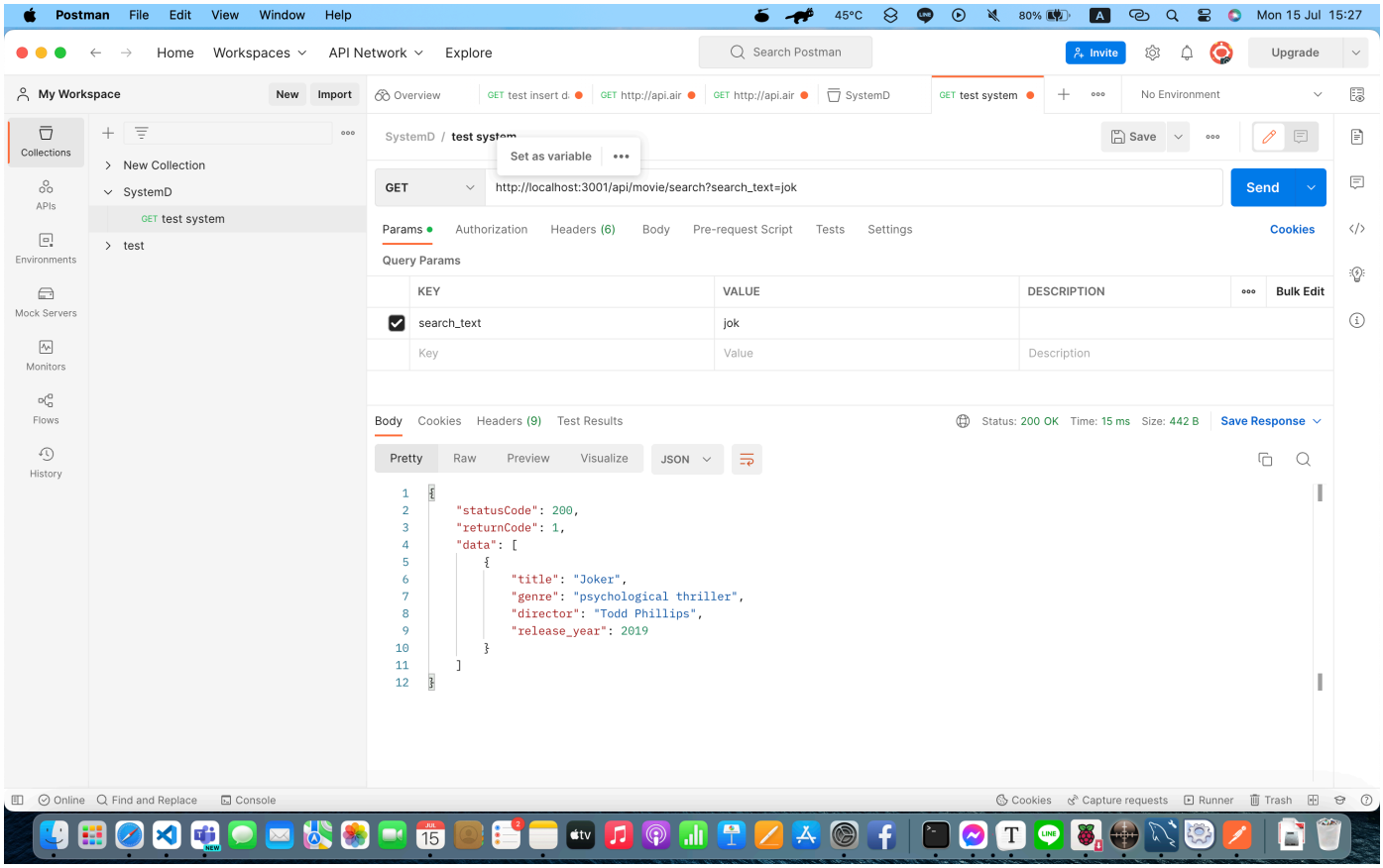
http://localhost:3001



http://localhost:3001/api/movie/all

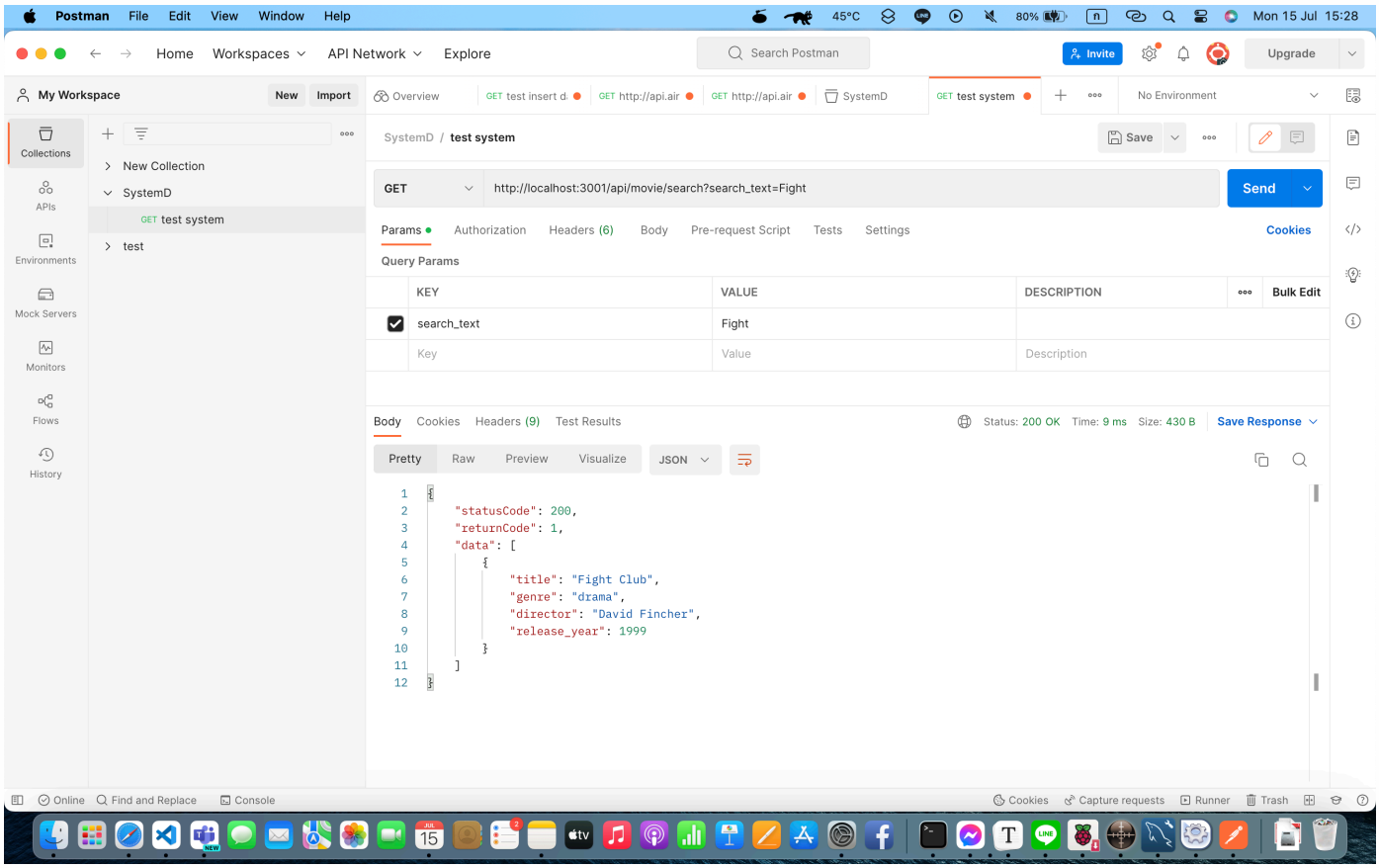


http://localhost:3001/api/movie/search?search_text=jok



ทดสอบเพิ่มเติมเปลี่ยนชื่อหนัง

http://localhost:3001/api/movie/search?search_text=Fight



เพิ่มเติม

เพิ่มการ ลบข้อมูล

โดยการปรับปรุง code ของการ search เปลี่ยน sql จาก insert เป็น delete