

[01-15] Node.js(Express)와 MySQL 연동

학습목표 : Express Rest API를 백엔드로 사용하는 Node.js와 MySQL DB를 사용하여 Vue.js에서 간단한 crud 애플리케이션을 빌드하고자 하는 과정에 관하여

[\[Node.js\] express 프레임워크와 MySQL을 연동하여 CRUD 구현하기](#)

[Node.js\(Express\)와 MySQL 연동](#)

[Node.js express MySQL 연동 환경 구성](#)

[\[Node.js\]Express란](#)

[Vue.js, Node.js, Express 및 MySQL로 CRUD 앱 구축](#)

1) MySQL - 테스트용 데이터베이스 생성 (local에 mariadb가 설치되어 있는 상태)

```
CREATE DATABASE demo_db;
```

```
CREATE TABLE product(  
  product_id INT(11) PRIMARY KEY AUTO_INCREMENT,  
  product_name VARCHAR(200),  
  product_price DOUBLE  
)ENGINE=INNODB;
```

- HeidiSQL 새로생성 > root 1234
 - 좌측 Unnamed > 마우스 우클릭 > 새로 생성 > DB 생성 : demo_db
 - demo_db > 마우스우클릭 > 테이블생성 > product

1-1) MVC 모델

- models : db자료를 끌고와 사용될 곳, 컨트롤러의 제어를 view에 반영
- view : 유저에게 보여지는 화면
- controls : 유저의 행동이 컴퓨터로 전달, 컨트롤러는 model을 통해서 데이터를 가져옴

1-2) Node.js : 크로스플랫폼 오픈소스 자바스크립트 런타임 환경으로 윈도우, 리눅스, macOS 등을 지원한다. Node.js는 V8 자바스크립트 엔진으로 구동되며, 웹 브라우저 바깥에서 자바스크립트 코드를 실행할 수 있다.

1-3) Express : **Node.js**를 사용하여 쉽게 서버를 구성할 수 있게 만든 클래스와 라이브러리의 집합체

- 수많은 개발자들에게 개발 규칙을 강제하여 코드 및 구조의 통일성을 향상
- 가장 보편적으로 사용되며, 프레임워크로써 웹 애플리케이션을 만들기 위한 각종 라이브러리와 미들웨어가 내장되어 개발되기 편리하다.

1-4) Node.js & Express : 자바스크립트 코드를 실행해 서버를 구성하는 것

- 서버와 데이터베이스(MySql)연결하는 것이 2)에서 이어서 설명

2) 서버(Node.js, Express)와 데이터베이스(Mysql)연결

- VS Code 로 접속 //1)에서 DB를 만들어둔 상태에서 진행

2-1) npm install 이 원활하게 실행되도록 관리자 권한 설정

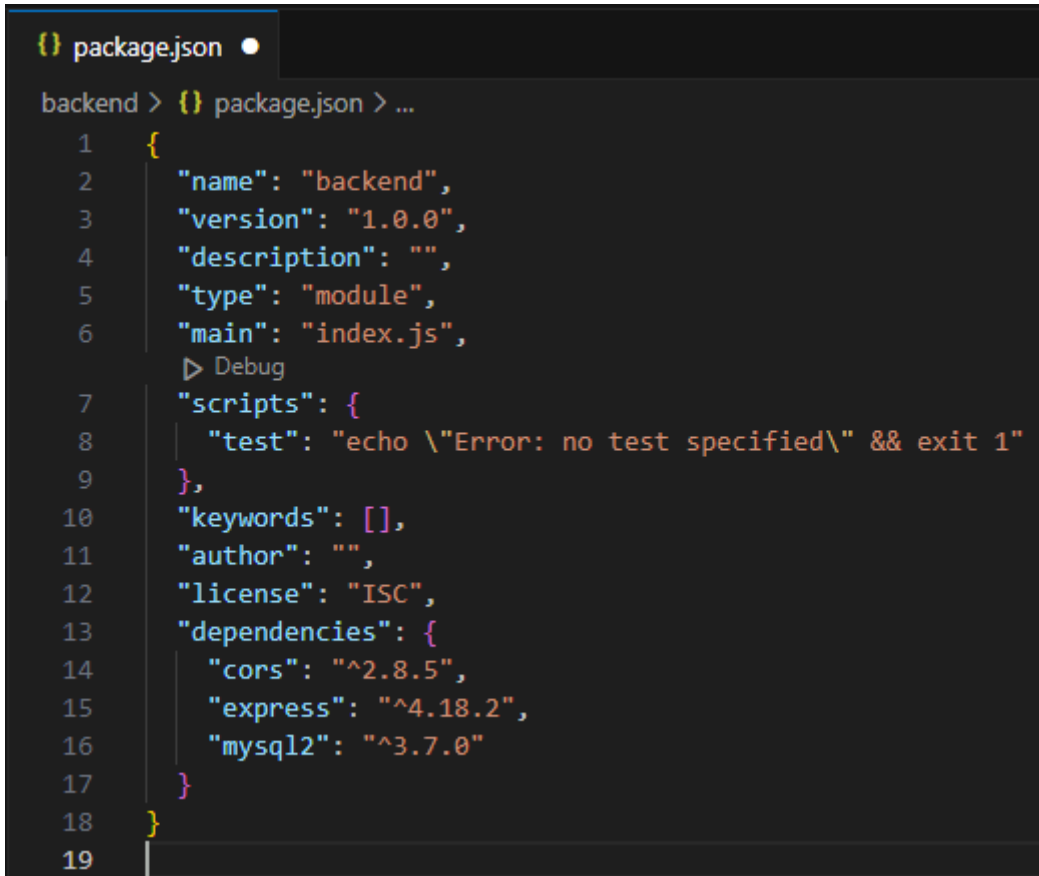
- powershell > 관리자 권한 실행 > [Set-ExecutionPolicy RemoteSigned](#) > Y

2-2) 디렉토리 및 라이브러리 설치

- 디렉토리 생성 : vue-axios > backend : 새 디렉토리 생성해서 진행
- express 및 mysql, cors 라이브러리 설치
 - npm init : 설치전 초기화
 - npm install express mysql2 cors : 라이브러리 셋을 한번에 설치
 - npm install -g express-generator : 이 코드는 추가 코드 작성이 필요해서 비권장
 - CORS(Cross-Origin Resource Sharing)란 자신이 속하지 않은 다른 도메인, 다른 프로토콜, 혹은 다른 포트에 있는 리소스를 요청하는 방식
 - mysql2 : 마리아디비랑 mysql은 둘다 2로 통일
 - -g express-generator 는 매번 설치해야함

2-3) package.json 수정 : json 데이터 호출을 위한 준비

- "type": "module", : 자동생성이 안되므로 추가할 것



```
{} package.json ●
backend > {} package.json > ...
1  {
2    "name": "backend",
3    "version": "1.0.0",
4    "description": "",
5    "type": "module",
6    "main": "index.js",
7    "scripts": {
8      "test": "echo \"Error: no test specified\" && exit 1"
9    },
10   "keywords": [],
11   "author": "",
12   "license": "ISC",
13   "dependencies": {
14     "cors": "^2.8.5",
15     "express": "^4.18.2",
16     "mysql2": "^3.7.0"
17   }
18 }
19
```

3) index.js (root 디렉토리에서 생성) //node.js와 mysql 연동과정

```
import Express from "express";
import cors from "cors";
import Router from "../router/router.js";

const app = Express();
app.use(Express.json());
app.use(cors()); // 모두 허용, 모든 도메인에서 제한 없이 해당 서버에 요청을 보내고 응답 받음
app.use(Router());
app.listen(5000, ()=> console.log('Server running at https://localhost:5000'));
```

- app.listen(5000) : 백엔드서버가 db와 소통 및 화면에 표시될 포트, express사용해 서버열기
- Route 작성 : 클라이언트의 요청에 대응하는 route를 설정

3-1) database.js (backend > config > database.js)

- `const pool = mysql.createPool` 이 더 보편적이라고 함

```
import mysql from "mysql2";

const db = mysql.createConnection({
  host: 'localhost', //만약 외부연결시 '168.126.66.1',
  user: 'root',
  password: '1234',
  database: 'demo_db'
});

export default db;
```

- import : mysql2 라고 npm install express 로 가져올 데이터베이스(mariadb,mysql)를 의미
- export : 생성한 db 를 오픈
 - 여기까지가 외부의 db를 연결해, 변수 db로 선언

3-2) productModel.js (backend > router > Models > productModel.js)

```
import db from "../../config/database.js";
export const getProducts = (result) => {
  db.query("SELECT * FROM product",(err, results) =>{
    if(err){
      console.log(err);
      result(err, null); //err 보내주고(같은err은미충돌), null 로 비워두고
    }else{
      result(null, results); //기존거 날리고(충돌방지), result 넣어라
    }
  });
}
```

- import : database.js 를 호출해 가져옴
- export : getProducts 는 이제 db 에서 어떤 조건,구문을 불러올 것인가에 대한 변수 선언
- CRUD 관련 구문이 들어옴
- 여기서 만든걸 controllers > product.js 에서 사용

3-3) Product.js (backend > router > controllers > Product.js)

//model을 컨트롤러(C)의 product.js 에서 보여줘 화면에 뿌리나?

```
import {getProducts} from "../Models/productModel.js";
//get
export const showProducts = (req, res) => {
  getProducts((err, results) => {
    if(err){
      res.send(err);
    }else{
      res.json(results);
    }
  });
}
```

- import : productModel.js 를 호출해 가져옴
- export : showProducts 는 getProducts 의 결과를 요청받고(req), 반응해줌(res)

3-4) router.js (backend > router > router.js)

```
import express from "express";
import {showProducts,} from "../controllers/product.js"

const router = express.Router();

//product.js 의 showProducts 를 가져오겠음
router.get('/product', showProducts)

export default Router;
```

4)

- C:\work\vue-axios> vue create frontend
- vue 관련 3파일 받아서 생성, 복불함
- C:\work\vue-axios> cd .\frontend\
- C:\work\vue-axios\frontend> npm install vue-router axios bulma
 - vue-router : 관련 라이브러리
 - axios : 백엔드서버 통해서 만들어진 내용을 클라이언트에 뿌려줌
 - bulma : css 간편 처리 라이브러리