[01-12] Node.js(Express)와 MySQL 연동

1) index.js

```
import express from "express"; //express install 한거 써야겠지?
import cors from "cors"; //cors install 한거 써야겠지?
import Router from "./router/router.js";    // router랑 헷갈리지 말라고 Router로...

import path from 'path';
const __dirname = path.resolve(); // 안해주면 아래 __dirname 부분 에러난다.  (CommonJS에서
사용하던 __dirname 변수가 ES 모듈에서는 없기 때문에 => type : moduile)

const app = express(); //그냥 express 쓰겠다는거야 이름만 app이지..

app.use(express.json()); //express 사용할 때 json 포멧 사용할거야
app.use(cors()); //난 모든 도메인에서 요청 받을거야.
app.use(Router); //내가 만들어놓은 router.js 사용 할거야
app.listen(3000, () => console.log('Server running at http://localhost:3000')); // 5000번으로
express서버 열거야. 열리면 아래 콘솔 찍어줘


/*frontend에서 build한 dist디렉토리 바라볼수있는 구문*/
app.use('/', express.static( path.join(__dirname, '/dist') )); //이 부분이 없으면 아래코드에서
dist/index.html을 로드하지 못해
app.get('/', (req, res)=>{ //기본 경로 '/'을 통해 빌드된 dist/index.html 파일을 로드시킬거야.
  res.sendFile(path.join(__dirname, '/dist/index.html'));
})

/*새로고침시 오류나는거 해결 코드*/ /*이해하려 하지 말구 그냥 첨부만 해놓자.*/
app.get('/*', function(req, res) {
  res.sendFile(path.join(__dirname, '/dist/index.html'), function(err) {
    if (err) {
      res.status(500).send(err)
    }
  })
})
```

- import : 앞서 설치한바 대로 express, cors , router.js 도 호출함
- app 은 express 로 선언해서 사용, 이후 app.use는 express에 대한 연결
- app.use : express에 관한 json 데이터를 사용, cors 와 router또한 사용
- app.listen : express를 포트, 주소
    - 이외 forntend에서 build한 것 로드등

2) database.js (backend > config > database.js)

```
import mysql from  "mysql2";

const db = mysql.createPool({
  host: 'localhost',
  user: 'root',
  password: '1234',
  database: 'hotel' //     port번호가 3306이 아니면 port: '3308' 이런식으로 포트번호 써줘야함
});
export default db;
```

- DB 와의 접속 설정
- createPool을 통해, host, pw, database를 어떤것을 할지 선택

3) productModel.js (backend > router > models > productModel.js)

```
import db from "../../config/database.js"   // database에 const로 선언해놓은 db가져올거임+_+!

// Get
export const getTest = (result) => {
  db.query("SELECT * FROM ORDERS", (err, results) => { // database.js에서 연동한 DB
데이터베이스에 쿼리 던질거야.
      if(err){ //쿼리 던졌는데 에러 떳다!
          console.log("Error : " + err); //에러내용 뭔데
          result(err, null); // 에러일땐 result 자리 비워둔다 :
      }else{
          result(null, results)
      }
  });
}

export const getOrders = (result) => {
  db.query("SELECT  * FROM orders left JOIN room ON orders.ROOM_RoomNum =
room.RoomNum WHERE USER_UserNum=1;", (err, results) => { // database.js에서 연동한 DB
데이터베이스에 쿼리 던질거야.
      if(err){ //쿼리 던졌는데 에러 떳다!
          console.log("Error : " + err); //에러내용 뭔데
          result(err, null); // 에러일땐 result 자리 비워둔다 :
      }else{
          result(null, results)
      }
  });
}
```

- database.js 의 자료를 받아와서 만질곳, CRUD가 들어오는 곳
- getTest 는 db로 부터 SELECT * FROM ORDERS 를 호출
- getOrders 는 db로 부터 SELECT * FROM orders left JOIN room ON orders.ROOM_RoomNum = room.RoomNum WHERE USER_UserNum=1 를 호출

4) product.js (backend > router > controllers > Product.js)

```
import {getTest,getOrders} from "../models/productModel.js"; //{} 해주는 이유는 여러개 넣기 위함이다. model에 추가하면 그만큼 넣어주자

// Get
export const showTest = (req, res) => {  //req, res 형태로 나오는걸 promise "약속" 형태라 한다 async await 요놈들도 promise 형태로 나온다는데
  getTest((err, results) => {
      if(err){
          res.send(err);   //화면(VUE) 에 send 할거야 err를
      } else {
          res.json(results);  //JSON 형태로 보낼거야, index.js에 express.JSON 사용한다 했지? 여기서 쓸라고..
      }});}

export const showOrders = (req, res) => {  //req, res 형태로 나오는걸 promise "약속" 형태라 한다 async await 요놈들도 promise 형태로 나온다는데
  getOrders((err, results) => {
      if(err){
          res.send(err);   //화면(VUE) 에 send 할거야 err를
      } else {
          res.json(results);  //JSON 형태로 보낼거야, index.js에 express.JSON 사용한다 했지? 여기서 쓸라고..
      } });}
```

- Model 과 View의 중계 페이지
- import : productModel.js 에서 선언한 것들을 가져오기 위해
- export : showTest, Order는 productModel.js 의 것을 불러와 화면에 뿌리기(send) 위함

- 즉 router의 models 에서 db에서 사용할 내용을 선언하고, controllers 에서 사용해 화면에 전달
    - router > models > productModel.js
    - router > controllers > Product.js.

5) router.js (backend > router > router.js)

```
import Express from "express"; //express 사용할거야.
import {showOrders, showTest} from "./controllers/product.js";

const router = Express.Router(); //express의 router 사용하기위함.

router.get('/test', showTest)
router.get('/orders', showOrders)

export default router;  // 아니 그래서 Router랑 router랑 뭔차인데... 왜 어떤건 대문자고 어떤건 소문자임
```

- VUE에서 axios 호출 받으면 model(DB connection : product.js)에 요청하는 페이지
- import Router from "./router/router.js"; : 대문자 Router 은 router.js를 의미
- const router = Express.Router() : 소문자 router 은 Express를 의미?!