20221025 니꼬스

```
import express from "express";

const PORT = 5000;

const app = express();

const gossipMiddleWare = (req, res, next) => {
  console.log("I'm in the middle ware!");
  return res.send("I have the super power NOW!");
  next();
  //next가 없다면 return res.send~ 이부분 호출은 되지 않을것이다.
  // next 함수를 호출한다면 이는 미들웨어임을 보여준다.
  // return이 임무를 중단시켜 next를 가지 못해experss의 handlehome 호출 실패!
  // middleware는 request에 응답하지 않는다. 다음 함수에 넘기기만 하는 함수일뿐이야.
};

const handleHome = (req, res) => {
  // next 쓰려고 리턴 무효 쓰기 위해 미들웨어=controller임) 만들어주기.
  // handlehome은 finalware
  return res.send("I love middleware");
};
app.get("/", gossipMiddleWare, handleHome);
// get 은 path가 필요햐다
// handle home 뒤에 ,찍고 다른 핸들러가 없기에
//next () 을해도 넘어가지가 않는다
const handleLogin = (req, res) => {
  return res.send("Have a nice lunch!!");
};

// 첫번쨔 칸은 선언, 두번째는 반응. 이 법칙을 기억하쇼!

const handleListening = () =>
  console.log("Server listening on http://localhost:${PORT}");

app.get("/login", handleLogin);

// login에 입장하면 handlelogin을 작동
//()이라는 함수를 넣어주어야 처리가 됨
// ()=> 의 뜻. 이 함수시 console.log를 해주라.

app.listen(PORT, handleListening);

// app.get 과 app.listen의 형식은 같다. port나 / , 그리고 그 뒤엔 함수
// 뒤에 /lalalalala 붙이면 갈수 있지만 기본 홈은 절대 갈수가 업승ㅁ -> 브라우저가 get/request 보내고 있기에.

// const handleClick = (event) => {

// }

// b.addEventListener("click", handleClick)

//next가 controller임. next는 다음 함수를 호출해준다
```

자세한 사항은 추후에 추가하여 줄 예정이다@!!! ㅇ ㅁㅇㄹ ㅇㄴ ㅁㅇㄴ ㄹㄴㅁㅇ ㅁㅇㄴㄹ ㅁㄴㅇ ㅱㅇㄴㄹ ㅁㄴㅇㄹ ㅇㄴ