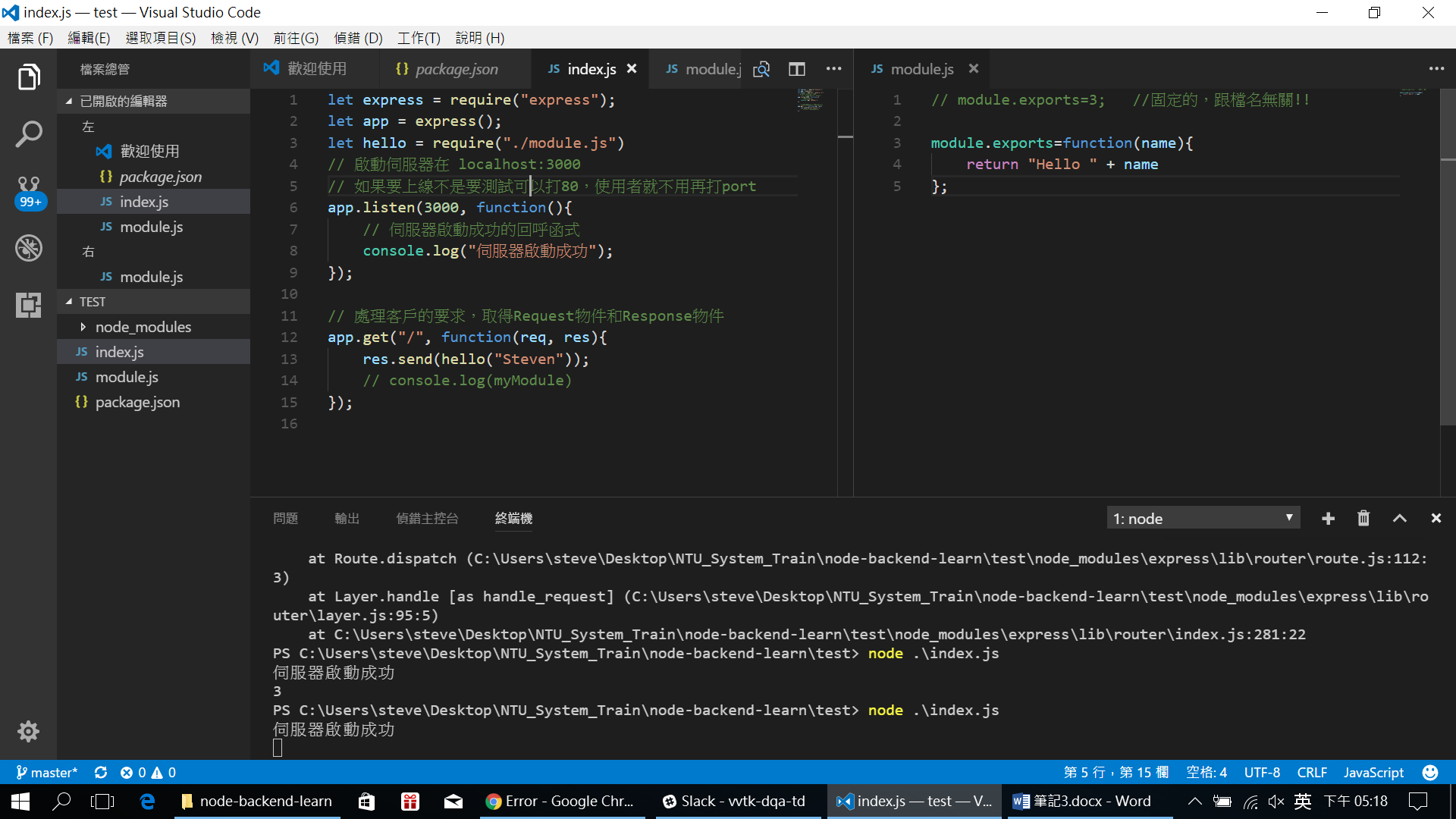
1. 寫一個己的模組!!!: (體會require)
   1. Ex: module.js
      1. 想讓吃什麼就放module.exports!!!
2. 例如:
3. let express = require("express");
4. let app = express();
5. let myModule = require("./module.js")
6. // 啟動伺服器在 localhost:3000
7. // 如果要上線不是要測試可以打80，使用者就不用再打port
8. app.listen(3000, function(){
9. // 伺服器啟動成功的回呼函式
10. console.log("伺服器啟動成功");
11. });
12. // 處理客戶的要求，取得Request物件和Response物件
13. app.get("/", function(req, res){
14. res.send("Hello");
15. console.log(myModule)
16. });
17. Module.js:
18. module.exports=3; //固定的，跟檔名無關!!
19. 則Terminal會印出3

另外module.exports是函式的話:



開啟瀏覽器會發現Hello Steven

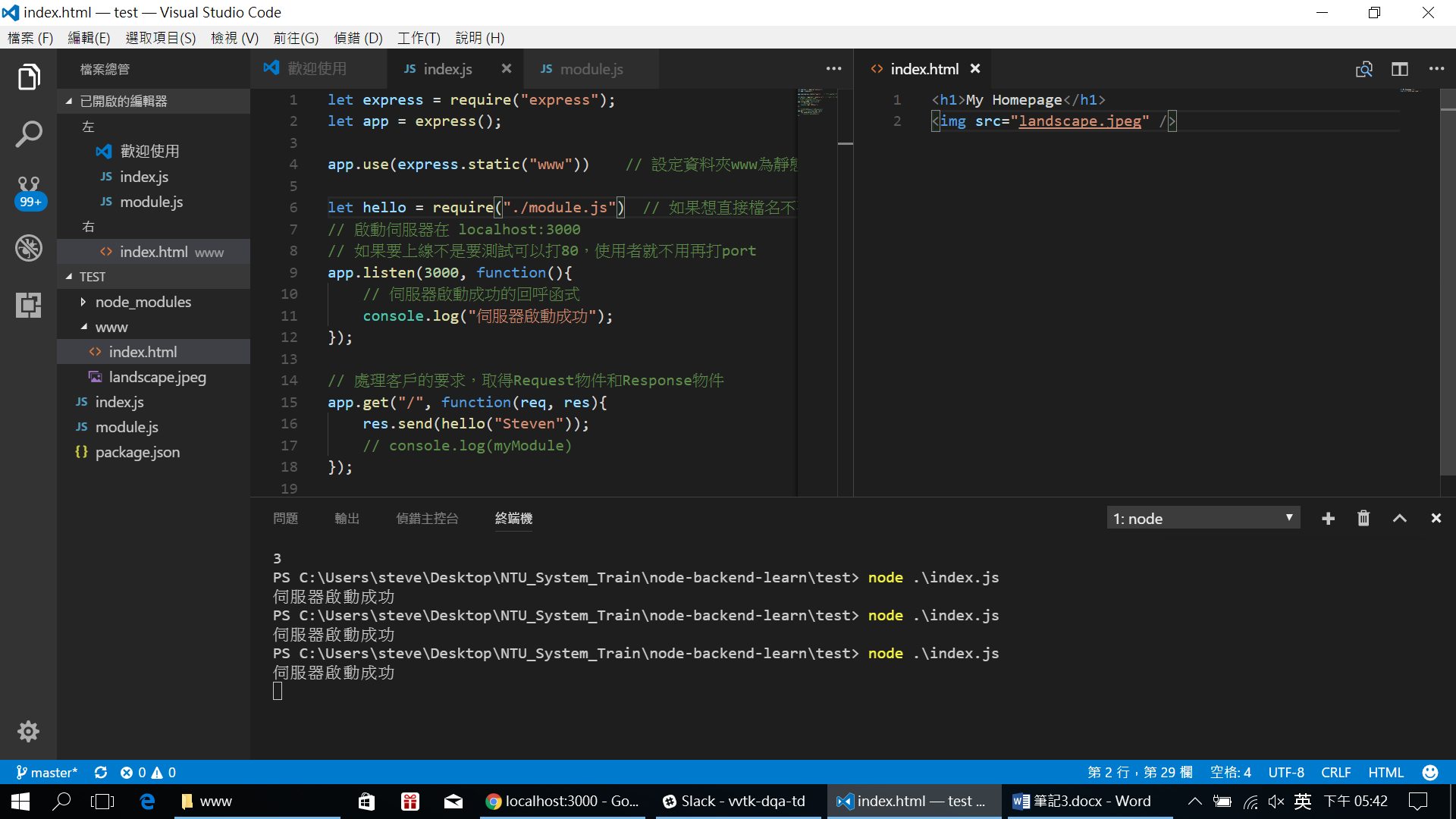
3. 網站的基本處理: https://docs.google.com/presentation/d/15oia2FxhgADiBXb-q87OdLLtX4T1jRZflT9phKwEAsk/edit#slide=id.p

4. 靜態內容的處理(網頁有一堆東西，如果都要一個一個寫也太多了吧!!)

所以:

假設我建一個資料夾較www，把index.html 或是圖片放上去

再設定: app.use(express.static("www")) // 設定資料夾www為靜態檔案的資料夾



示範對應http status code / 修改Response Header

let express = require("express");

let app = express();

app.use(express.static("www")) // 設定資料夾www為靜態檔案的資料夾

let hello = require("./module.js") // 如果想直接檔名不要./，則要把檔案放到node\_modules資料夾

// 啟動伺服器在 localhost:3000

// 如果要上線不是要測試可以打80，使用者就不用再打port

app.listen(3000, function(){

// 伺服器啟動成功的回呼函式

console.log("伺服器啟動成功");

});

// 處理客戶的要求，取得Request物件和Response物件

app.get("/test", function(req, res){

// res.send(hello("Steven"));

// 設定Response Header (可以觀察設前設後開發者工具變化)

res.set("Content-Type", "text/plain") // 這樣的話就不會把tag當作html的格式來處理，會變純文字

// 設定HTTP Status Code

res.status(404).send("404 Error!")

});

Res.download >>

res.download("./package.json")

就可以讓使用者download該檔

Res.redirector >> 導向到其他地方

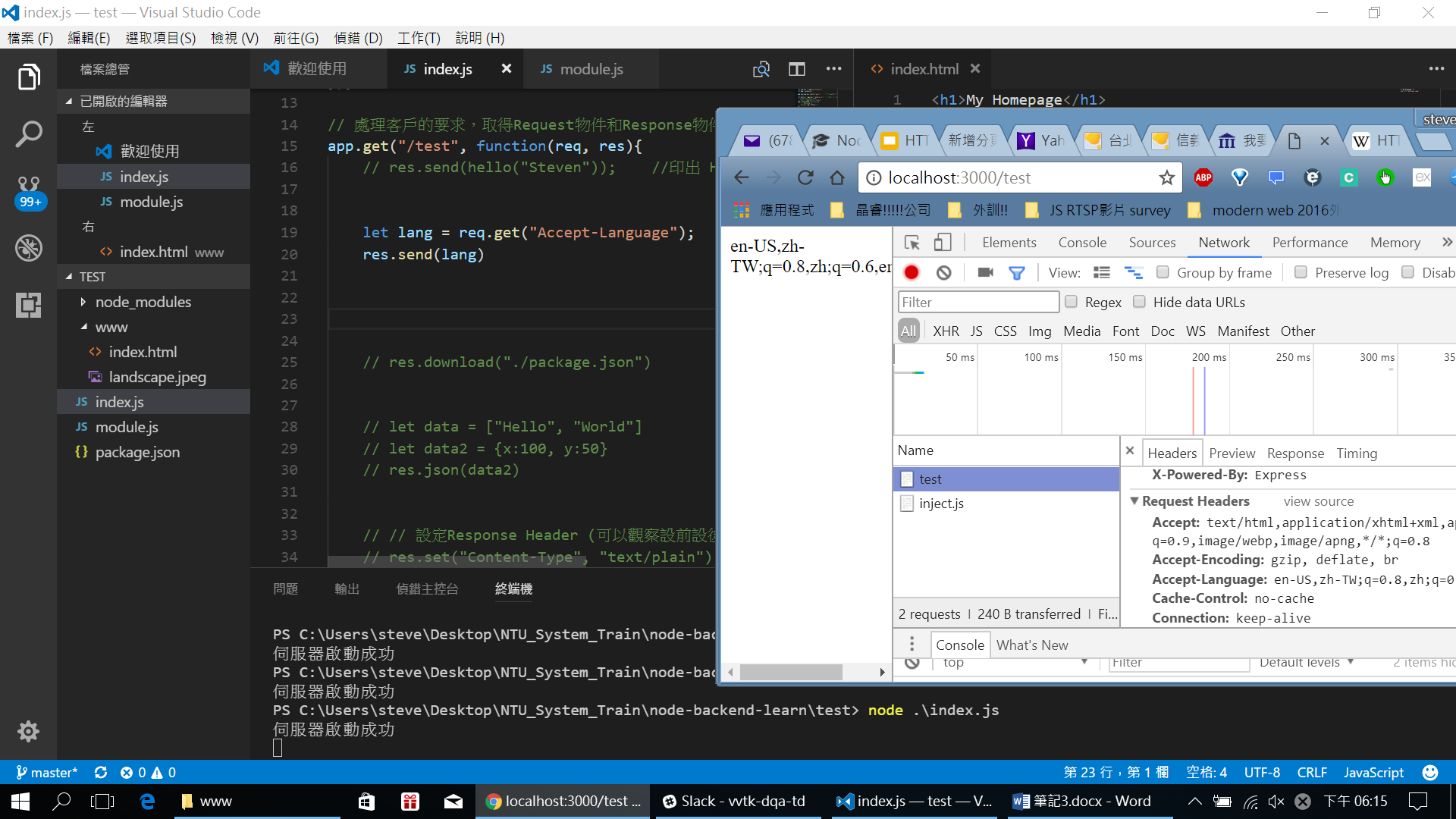
Request:

Ex: 拿request Header

// 取得Request Header!!

let lang = req.get("Accept-Language");

res.send(lang)



Request Path:

res.send(req.path)

可回傳主機後面的path

如此可以再針對path做事

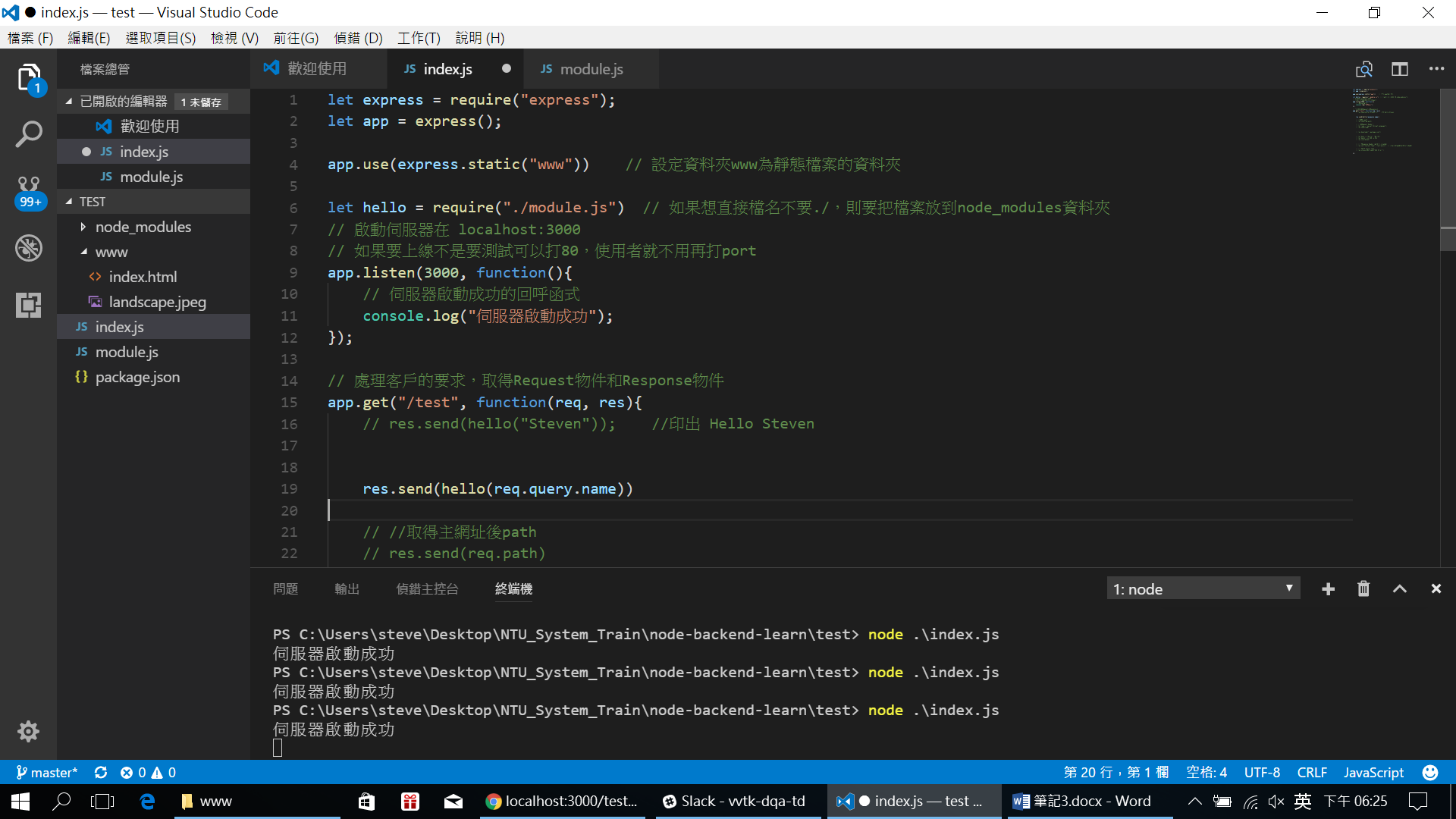
\*\* 重要:

Query :

主網址? 參數名=參數值&參數名=參數值

例如輸入:

res.send(hello(req.query.name))



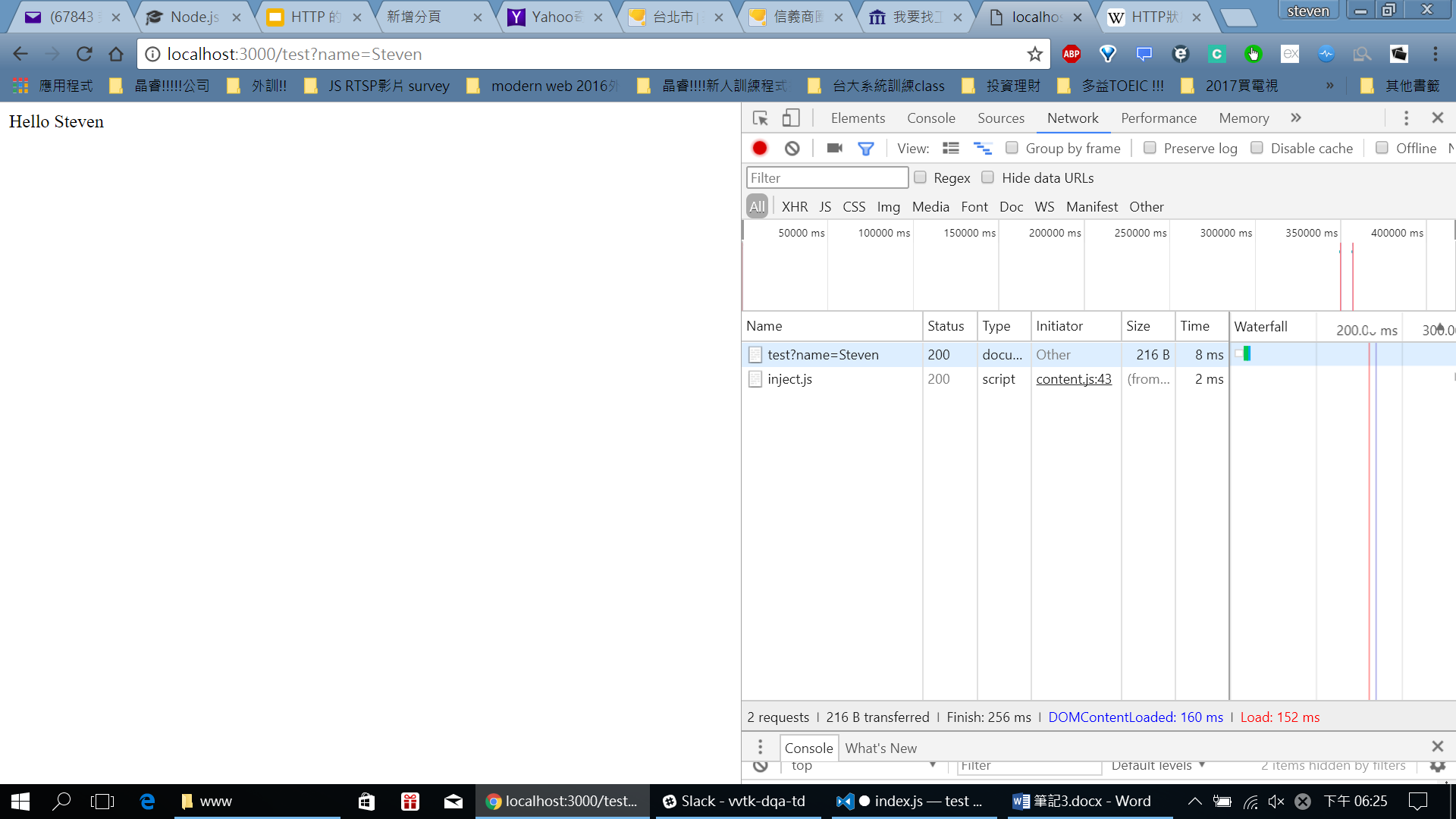
因為剛已經定義hello function(在自定義module囉)了

在網址輸入<http://localhost:3000/test?name=Steven>

則因為在req.ruery.name就可以撈到Steven了

再將之拋到自定義module內的hello function內

就可以看到印出來hello Steven囉!!!



\*\* !!!! 透過路徑來指定參數(不用’?’)

直接透過網址的變化來對應不同的功能!!!!

現在有一堆人喜歡用這樣搞!!

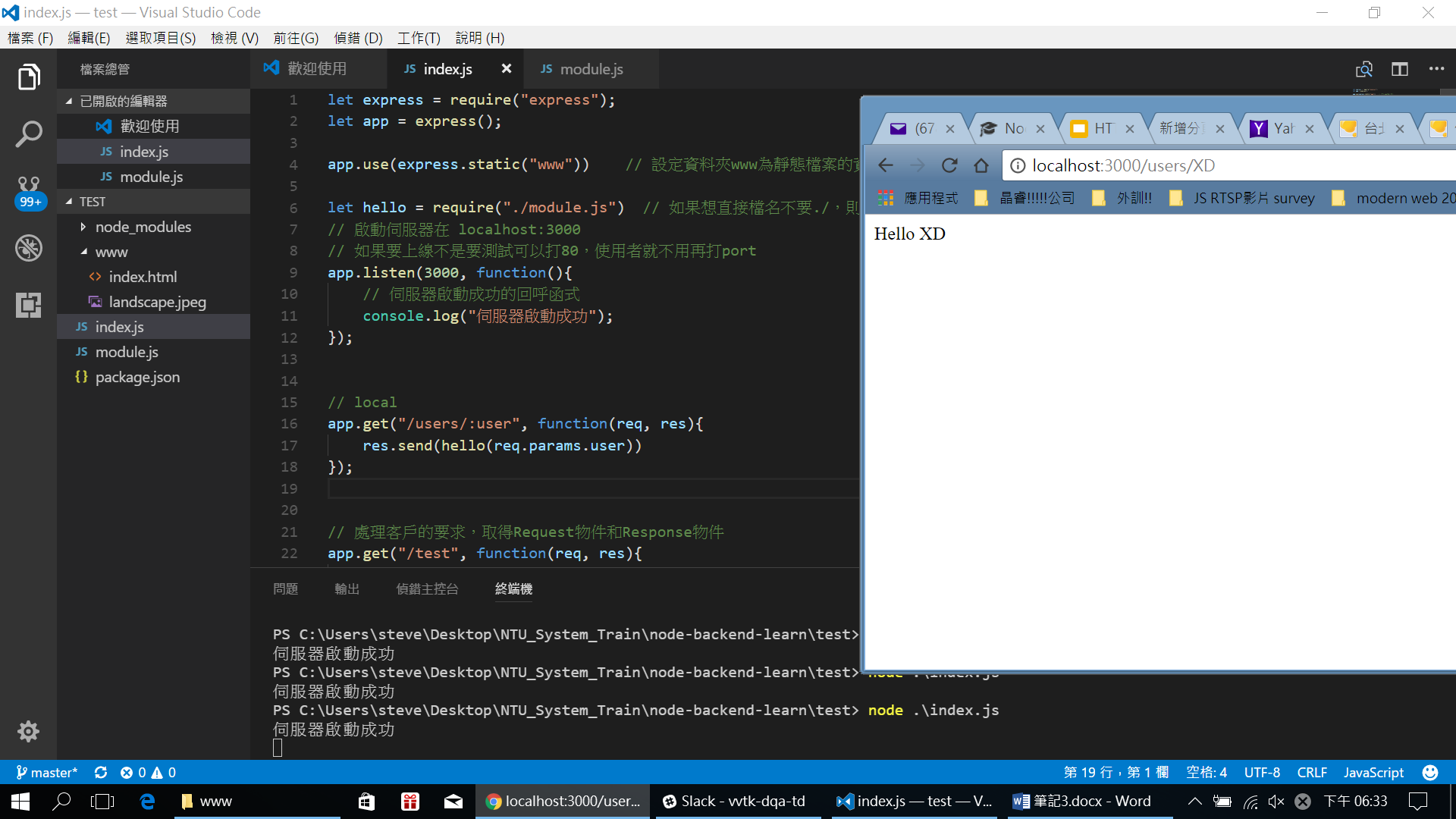
假設路徑/users

// local

app.get("/users/:user", function(req, res){

res.send(hello(req.params.user))

});



仲介軟體(函式):

// 中介函式，會先進入這裡才會繼續做其他的(例如可以用在log)

let middle = function(req, res, next){

console.log("Middleware Run");

next();

};

app.use(middle)

app.get("/test\_middle", function(req, res){

res.send("GOO!!!!!!")

});

如此會先走到middle function才會繼續做其他的事

所以實際上會先輸出Middleware run!!

後續才會印Go!!