中介函式:

let express = require('express')

let app = express()

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(express.static("www")); //也是中介函式

app.use(function(req, res, next){ // 一定要這3個參數，next是指到下一個中介函數的參考!

console.log("中介函式啟動");

next(); //呼叫下一個中介函式，沒有就沒有

});

// 建立中介函式

app.use(function(req, res, next){

console.log("中介函式啟動2");

next(); //呼叫下一個中介函式，沒有就沒有

});

app.get('/hello', function (req, res) {

res.send('Hello World!')

})

app.listen(3000, function () {

console.log('Example app listening on port 3000!')

})

如此輸入: <http://localhost:3000/test.html> 就會看到button

而且console同時輸出”” 中介函式啟動” + “中介函式啟動2”

\*\* 中介函式可以加上路徑，即在某些路徑開頭才會到某些中介函式:

例如:

let express = require('express')

let app = express()

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(express.static("www")); //也是中介函式

app.use(function(req, res, next){ // 一定要這3個參數，next是指到下一個中介函數的參考!

console.log("中介函式啟動");

next(); //呼叫下一個中介函式，沒有就沒有

});

// 建立中介函式

app.use(function(req, res, next){

console.log("中介函式啟動2");

next(); //呼叫下一個中介函式，沒有就沒有

});

// 建立中介函式 連線到localhost:30000/users/XXXX

app.use("/users", function(req, res, next){ //這種仲介是在如果連線到users開頭才會用到該中介函式

console.log("users中介函式啟動");

next(); //呼叫下一個中介函式，沒有就沒有

});

app.get('/hello', function (req, res) {

res.send('Hello World!')

})

app.listen(3000, function () {

console.log('Example app listening on port 3000!')

})

若輸入localhost:3000/users:

中介函式啟動

中介函式啟動2

users中介函式啟動

可用在像是會員系統，前一定要先做認證!!!!

\*\* 從前端傳POST !! (用建立的html 傳post資料!)

之前的GET:

let express = require('express')

let app = express()

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(express.static("www")); //也是中介函式

//連線: localhost:3000/save?data=test

app.get('/save', function (req, res) {

console.log(req.query.data)

res.send('OK')

})

app.listen(3000, function () {

console.log('Example app listening on port 3000!')

})

輸入<http://localhost:3000/save?data=test>

可以再terminal看到test

但是通常傳資料上來會用POST:

GET: 把參數皆在網址後面用問號隔開

POST:把參數的資料放在另一個地方(request的body, http的要求，可以放很多資料像是檔案上傳等等，沒事不會用GET傳送重要資料)

但如果把剛剛的function直接改成POST會發現在瀏覽器會錯，因為用瀏覽器適用GET!!!!

www/test.html輸入:

let express = require('express')

let app = express()

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(express.static("www")); //也是中介函式

//連線: localhost:3000/save?data=test

app.post('/save', function (req, res) {

console.log(req.query.data)

res.send('OK')

})

app.listen(3000, function () {

console.log('Example app listening on port 3000!')

})

<script>

function saveData() {

// 用AJAX連線到後端

let req = new XMLHttpRequest();

req.open("post", "/save"); // 如果是get 就直接將參數寫在右邊

req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded"); // "application/x-www-form-urlencoded" 要傳送的格式

req.onload = function() {

alert(this.responseText);

};

req.send("data=ply"); // post將參數寫在這裡!

}

</script>

<button onclick="saveData();">Click Me!</button>

但有問題(出現undefined) 後端沒有抓到資料!!!!

解法:

先安裝套件

npm install body-parser --save

改為:

let express = require('express')

let app = express()

let parser = require("body-parser")

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(parser.urlencoded({extended:true}));

app.use(express.static("www")); //也是中介函式

// 改用POST

app.post('/save', function (req, res) {

console.log(req.body.data); // 把原本get的query改為body !!!

res.send('OK');

})

app.listen(3000, function () {

console.log('Example app listening on port 3000!')

})

GET 是直接從req.query拿 POST從req.body拿

當然是從參考的文件看到的: <http://expressjs.com/zh-tw/4x/api.html>

跟request最有關聯:

<http://expressjs.com/zh-tw/4x/api.html#req>

<http://expressjs.com/zh-tw/4x/api.html#req.body>

研究body-parser: <https://www.npmjs.com/package/body-parser>

上傳檔案會用到不同body: multiparty (也可以試著搜尋 node upload file)

\*\* 接下來要準備存進資料庫!!! (Firebase不是SQL資料庫，所以不用下SQL指令，式JSON資料庫)

1. 先到FireBase: <https://firebase.google.com/>

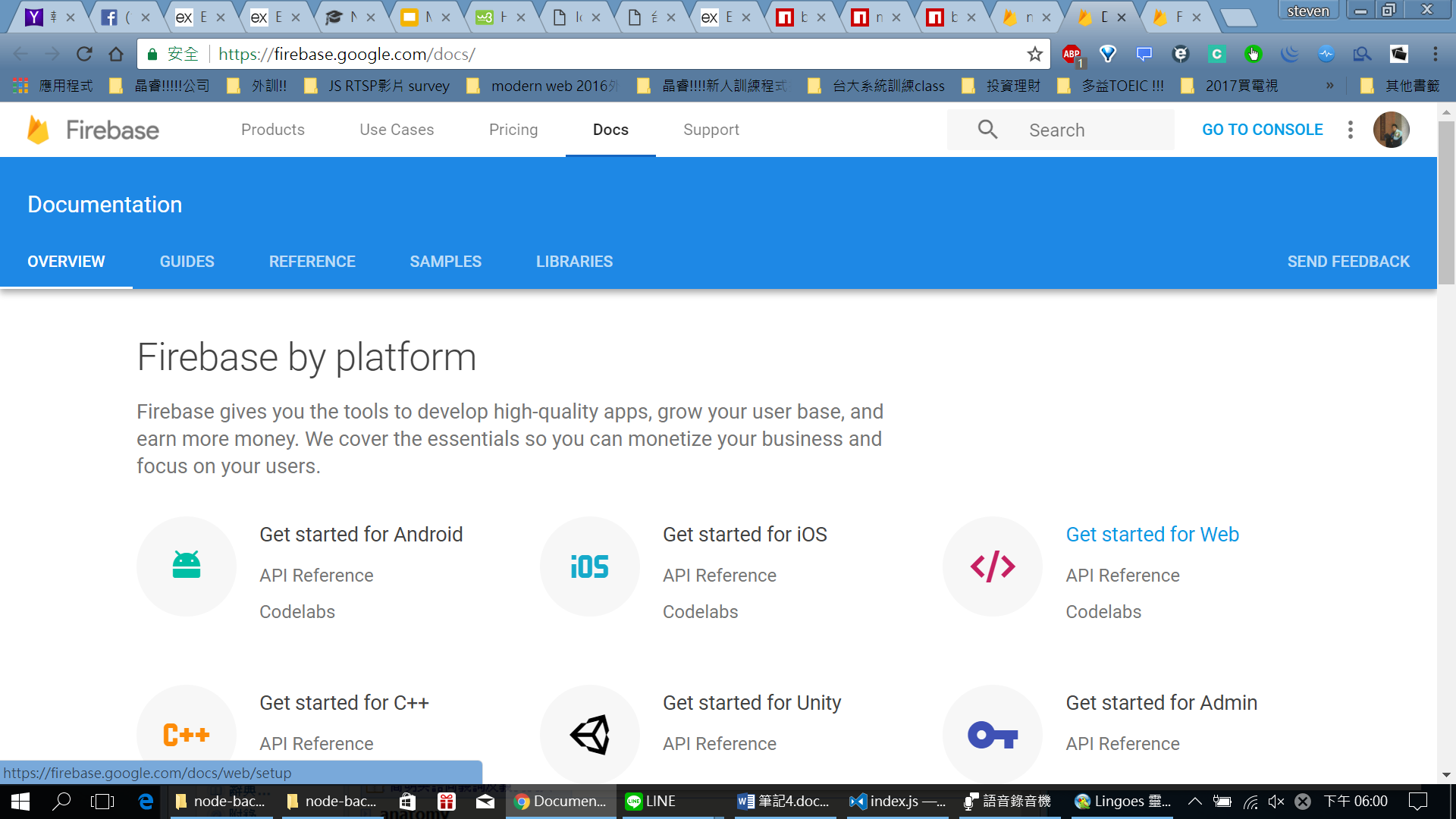
推薦原因: 不用自建資料庫<https://firebase.google.com/pricing/>

(不只式資料庫，算是完整的後端解決方案喔!!!)

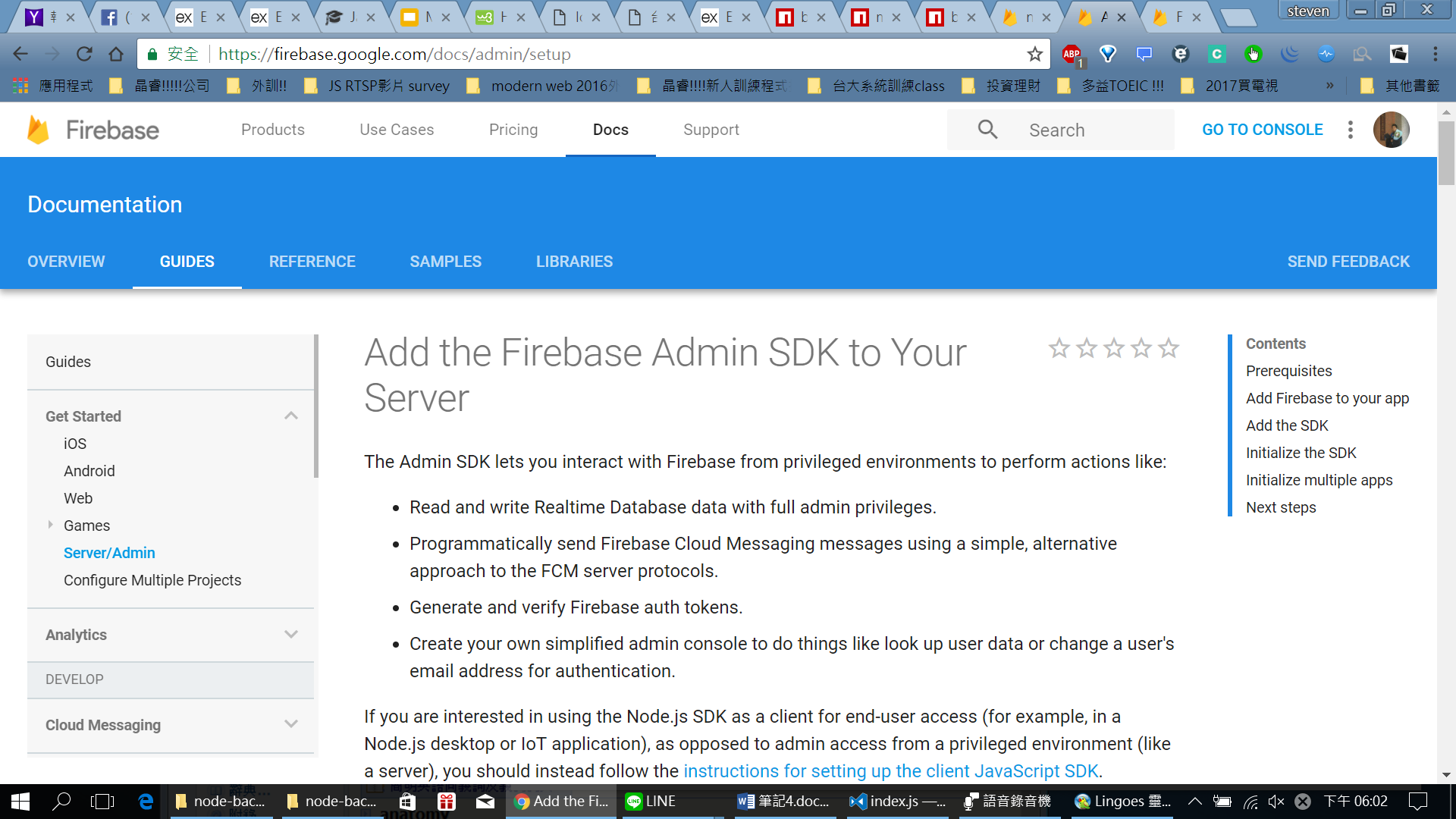
1. 

不要選最右邊的喔!! 那是前端再用(前端可以直接透過Firebase存取資料庫)的…，我們是寫Server後端喔!!!

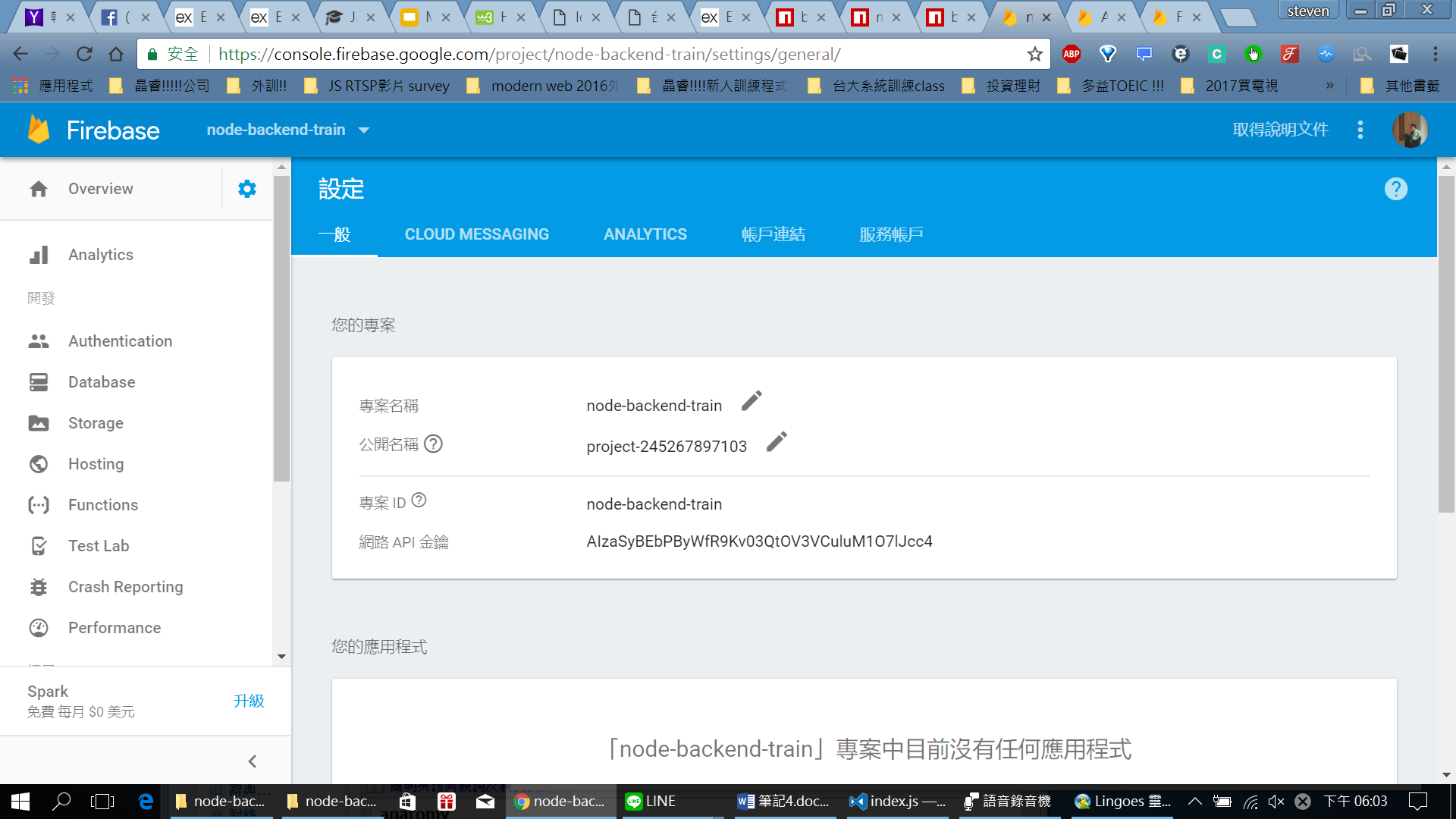
改點選右上的取得文件:



改選擇Get started for Admin:

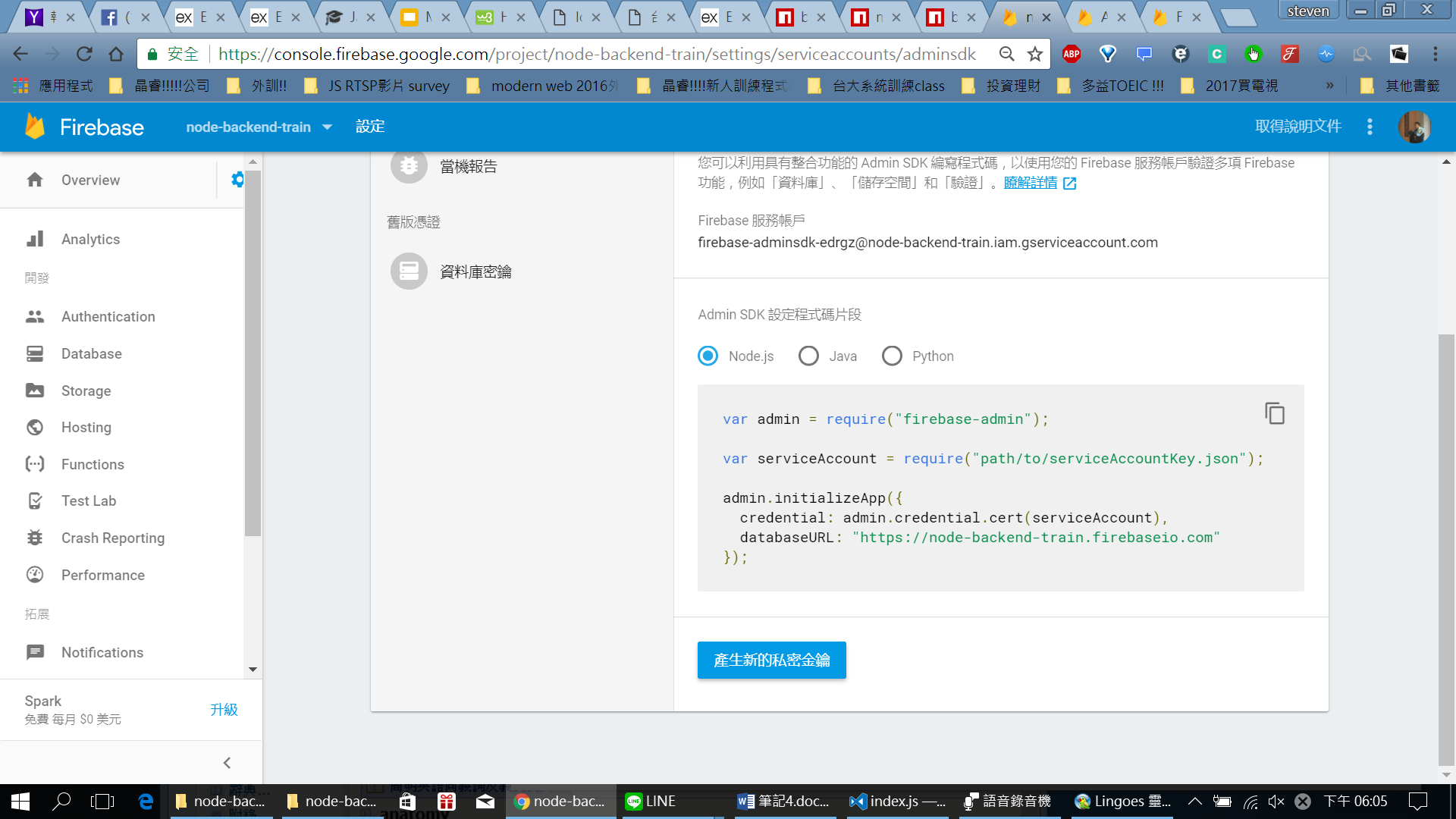


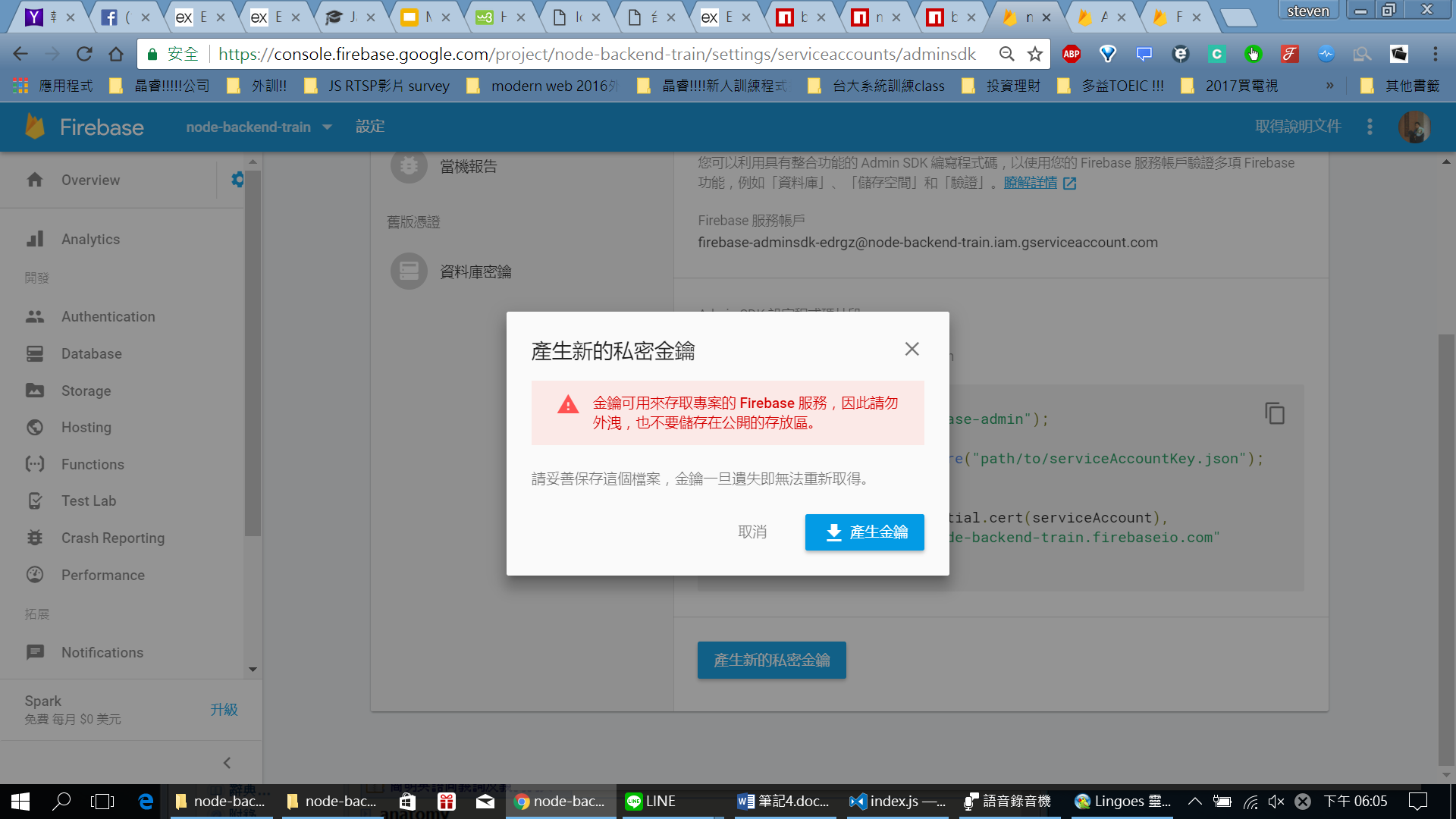
家下來需要一把private key: (點選左邊設定的專案設定，再點選服務帳戶)



最下面產生金鑰

記得選 NODE JS





\*\* 需要安裝firebase-admin套件:

npm install firebase-admin –save

把剛剛網頁的:

var admin = require("firebase-admin");  
  
var serviceAccount = require("path/to/serviceAccountKey.json");  
  
admin.initializeApp({  
  credential: admin.credential.cert(serviceAccount),  
  databaseURL: "https://node-backend-train.firebaseio.com"  
});

貼到index.js 做初始化firebase:

let express = require('express')

let app = express()

let parser = require("body-parser")

// 初始化firebase

var admin = require("firebase-admin");

var serviceAccount = require("./serviceAccountKey.json"); // 改成自己的路徑

admin.initializeApp({

credential: admin.credential.cert(serviceAccount), //告訴google你是誰

databaseURL: "https://node-backend-train.firebaseio.com"

});

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(parser.urlencoded({extended:true}));

app.use(express.static("www")); //也是中介函式

// 改用POST

app.post('/save', function (req, res) {

console.log(req.body.data); // 把原本get的query改為body !!!

res.send('OK');

})

app.listen(3000, function () {

console.log('Example app listening on port 3000!')

})

1. 將本來req.body.data存至資料庫:

let express = require('express')

let app = express()

let parser = require("body-parser")

// 初始化firebase

var admin = require("firebase-admin");

var serviceAccount = require("./serviceAccountKey.json"); // 改成自己的路徑

admin.initializeApp({

credential: admin.credential.cert(serviceAccount), //告訴google你是誰

databaseURL: "https://node-backend-train.firebaseio.com"

});

// 建立中介函式 (順序有意義!!! 上下順序有差!!!)

app.use(parser.urlencoded({extended:true}));

app.use(express.static("www")); //也是中介函式

// 改用POST

app.post('/save', function (req, res) {

console.log(req.body.data); // 把原本get的query改為body !!!

// 存進資料庫

let database = admin.database(); //開啟資料庫

let ref = database.ref("/message"); //取得參考路徑 放在message裡面

// 儲存資料!

ref.set(req.body.data, function(error){ //有一個call back確認是否成功!!!

if(error){

res.send("Failed!!")

}

else{

res.send("Saved!!")

}

});

// res.send('OK');

})

app.listen(3000, function () {

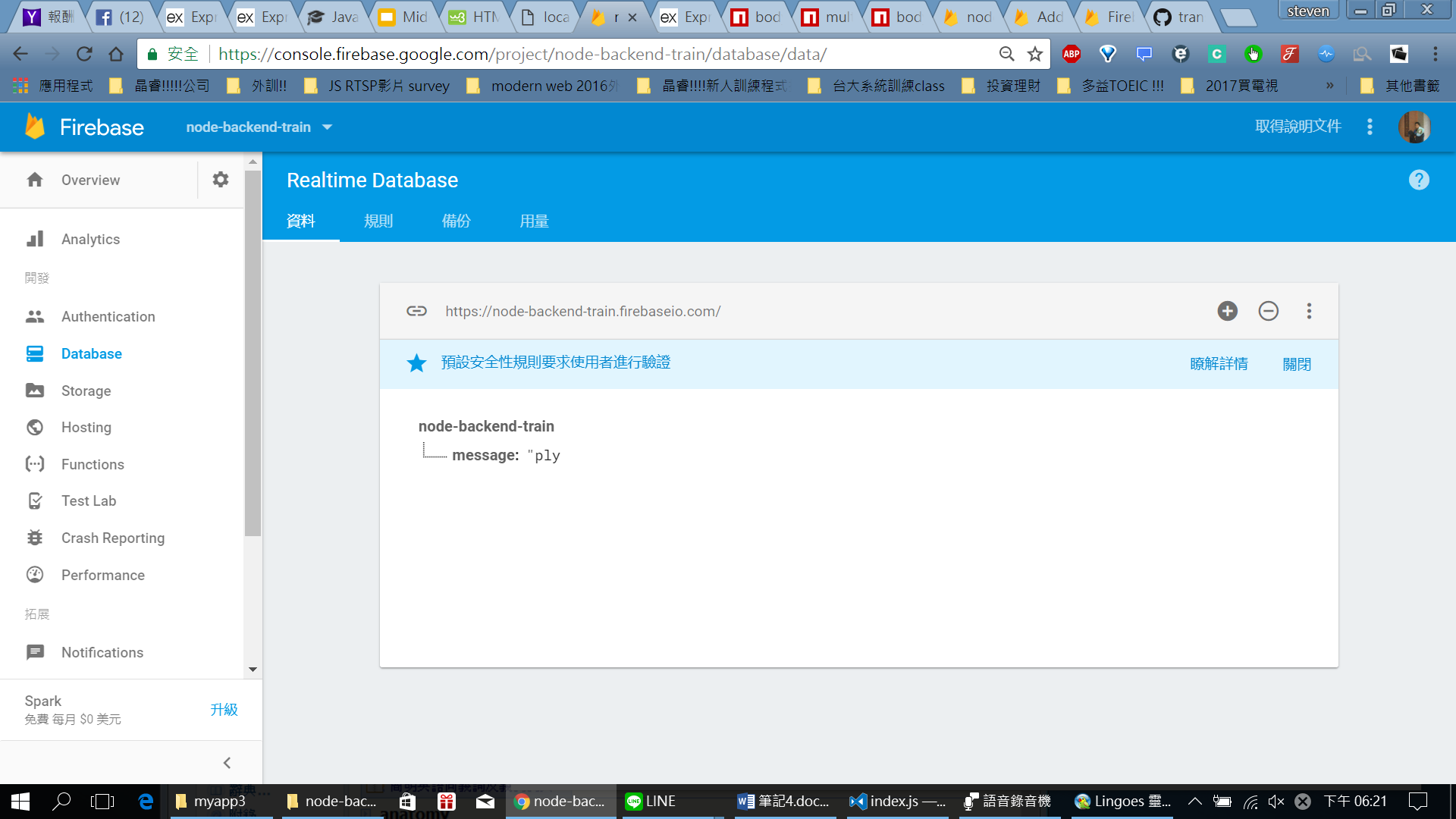
console.log('Example app listening on port 3000!')

})

再到剛剛的網頁localhost:3000/test.html點click button後就可以看到Saved 成功通知!

接下來再連至上面資料庫網址: https://node-backend-train.firebaseio.com

看到資料囉!!!:



(若路徑有前面重複 會蓋掉舊的!!!!)