

## การสร้างโปรเจ็ค Node.js

- 1) สร้าง directory สำหรับโปรเจ็ค
- 2) ใช้คำสั่ง npm init เพื่อเริ่มต้นโปรเจ็ค โดยระบุข้อมูลต่าง ๆ เพื่อใช้ในการสร้างไฟล์ package.json

```
Command Prompt

E:\Teaching_Git\NodeJs\Material>mkdir books_api
E:\Teaching_Git\NodeJs\Material>cd books_api
E:\Teaching_Git\NodeJs\Material\books_api>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (books_api) books-api
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author: Kesinee Boonchuay
license: (ISC)
About to write to E:\Teaching_Git\NodeJs\Material\books_api\package.json:
{
  "name": "books-api",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Kesinee Boonchuay",
  "license": "ISC"
}

Is this OK? (yes) yes
E:\Teaching_Git\NodeJs\Material\books_api>
```

- 3) เมื่อสำเร็จพบไฟล์ package.json ใน directory นั้น

```
E:\Teaching_Git\NodeJs\Material\books_api>dir/w
Volume in drive E is DATA
Volume Serial Number is 5CB3-5557

Directory of E:\Teaching_Git\NodeJs\Material\books_api

[.]                [..]                package.json
1 File(s)          222 bytes
2 Dir(s)  817,682,976,768 bytes free

E:\Teaching_Git\NodeJs\Material\books_api>code .
```

- 4) ติดตั้ง package ที่ใช้งานในโปรเจกต์ โดยใช้คำสั่ง `npm install <ชื่อ package>` โดยใช้โปรเจกต์นี้จะใช้ packages คือ `express cors` และ `firebase-admin` เมื่อติดตั้งแล้ว จะพบชื่อ packages ดังกล่าวใน dependencies ของไฟล์ `package.json` (สามารถติดตั้งโดยการเพิ่มในไฟล์ `package.json` โดยตรงและสั่ง `npm install` หลังจากเพิ่มได้)

```
() package.json > {} dependencies
1  {
2    "name": "books-api",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "author": "Kesinee Boonchuay",
10   "license": "ISC",
11   "dependencies": {
12     "cors": "^2.8.5",
13     "express": "^4.18.1",
14     "firebase-admin": "^11.0.1"
15   }
16 }
17
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

(c) Microsoft Corporation. All rights reserved.

E:\Teaching\_Git\NodeJs\Material\books\_api>npm install express

[#####] / reify: get-intrinsic: http fetch GET 200 https://re...  
[#####] | reify: fresh: http fetch GET 200 https://registry.n...

added 57 packages, and audited 58 packages in 4s

7 packages are looking for funding  
run 'npm fund' for details

found 0 vulnerabilities

E:\Teaching\_Git\NodeJs\Material\books\_api>npm install cors

added 2 packages, and audited 60 packages in 1s

7 packages are looking for funding  
run 'npm fund' for details

found 0 vulnerabilities

E:\Teaching\_Git\NodeJs\Material\books\_api>npm install firebase-admin

added 163 packages, and audited 223 packages in 19s

14 packages are looking for funding  
run 'npm fund' for details

found 0 vulnerabilities

E:\Teaching\_Git\NodeJs\Material\books\_api>

## การสร้าง API (Book)

- 1) เริ่มต้นด้วยการเรียกใช้ package ต่างๆ สำหรับการพัฒนา API เช่น express และ cors

```
1 var express = require("express");
2 var cors = require("cors");
3
4 const app = express();
5 const port = process.env.PORT || 3000;
6
7 app.use(express.json());
8 app.use(
9   express.urlencoded({
10     extended: true
11   })
12 );
13
14 app.use(cors());
```

- 2) เชื่อมต่อกับฐานข้อมูล Firebase โดยจะต้องระบุ databaseURL และ path ไปยังไฟล์ซึ่งเป็น key ของ firebase

```
1 //Firebase Real Time
2 var firebase = require("firebase-admin");
3 var serviceAccount = require("./firebase_key.json");
4
5 firebase.initializeApp({
6   credential: firebase.credential.cert(serviceAccount),
7   databaseURL:
8     "https://flutter-book-api-default-rtdb.asia-southeast1.firebaseio.com",
9 });
10
11 var db = firebase.database();
```

- 3) การกำหนด endpoint สำหรับเรียกดูข้อมูลหนังสือทั้งหมด (get)

```
1 //Get all books
2 app.get("/books", function (req, res) {
3   res.setHeader("Content-Type", "application/json");
4
5   var booksReference = db.ref("books");
6
7   booksReference.on(
8     "value",
9     function (snapshot) {
10      res.json(snapshot.val());
11      booksReference.off("value");
12    },
13    function (errorObject) {
14      res.send("The read failed: " + errorObject.code);
15    }
16  );
17 });
```

- 4) การกำหนด endpoint สำหรับเรียกดูข้อมูลหนังสือตาม id ที่ระบุ (get)

```
1 //Get a book by id
2 app.get("/books/:bookid", function (req, res) {
3   res.setHeader("Content-Type", "application/json");
4   var bookid = Number(req.params.bookid);
5   var booksReference = db.ref("books");
6
7   booksReference
8     .orderByChild("bookid")
9     .equalTo(bookid)
10    .on(
11      "child_added",
12      function (snapshot) {
13        res.json(snapshot.val());
14        booksReference.off("value");
15      },
16      function (errorObject) {
17        res.send("The read failed: " + errorObject.code);
18      }
19    );
20 });
```

5) การกำหนด endpoint สำหรับเพิ่มข้อมูลหนังสือใหม่ (post)

```
1 //Add new book
2 app.post("/books", function (req, res) {
3   var bookidValue = req.body.bookid;
4   var titleValue = req.body.title;
5   var shortDescriptionValue = req.body.shortDescription;
6   var authorValue = req.body.author;
7   var categoryValue = req.body.category;
8   var isbnValue = req.body.isbn;
9   var pageCountValue = req.body.pageCount;
10  var priceValue = req.body.price;
11  var publishedDateValue = req.body.publishedDate;
12  var thumbnailUrlValue = req.body.thumbnailUrl;
13
14  var referencePath = "/books/" + bookidValue + "/";
15
16  //Add to Firebase
17  var bookReference = db.ref(referencePath);
18  if (bookReference !== null) {
19    bookReference.update(
20      {
21        bookid: bookidValue,
22        title: titleValue,
23        shortDescription: shortDescriptionValue,
24        author: authorValue,
25        category: categoryValue,
26        isbn: isbnValue,
27        pageCount: pageCountValue,
28        price: priceValue,
29        publishedDate: publishedDateValue,
30        thumbnailUrl: thumbnailUrlValue,
31      },
32      function (error) {
33        if (error) {
34          res.send("Data could not be saved." + error);
35        } else {
36          res.send("");
37        }
38      }
39    );
40  }
41 });
```

6) การกำหนด endpoint สำหรับแก้ไขข้อมูลหนังสือ (put)

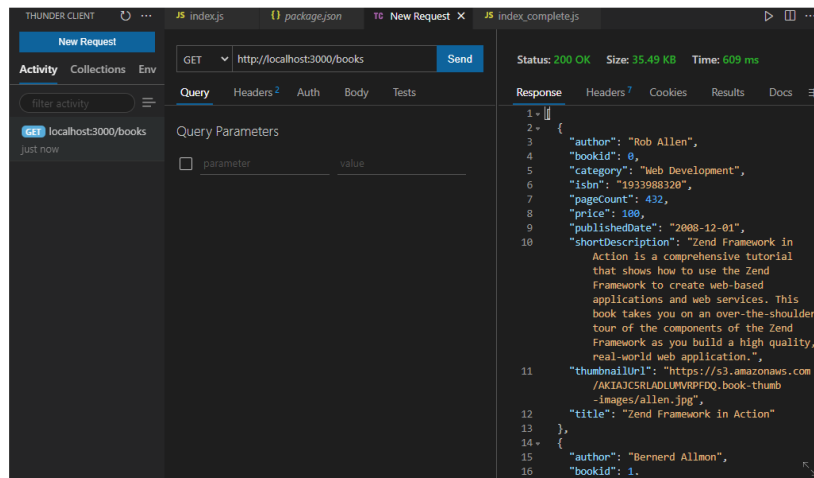
```
1 //Edit a book by id
2 app.put("/books/:bookid", function (req, res) {
3
4
5     var bookidValue = Number(req.params.bookid);
6     var titleValue = req.body.title;
7     var shortDescriptionValue = req.body.shortDescription;
8     var authorValue = req.body.author;
9     var categoryValue = req.body.category;
10    var isbnValue = req.body.isbn;
11    var pageCountValue = req.body.pageCount;
12    var priceValue = req.body.price;
13    var publishedDateValue = req.body.publishedDate;
14    var thumbnailUrlValue = req.body.thumbnailUrl;
15
16    var referencePath = "/books/" + bookidValue + "/";
17
18    var bookReference = db.ref(referencePath);
19    if (bookReference !== null) {
20        bookReference.update(
21            {
22                bookid: bookidValue,
23                title: titleValue,
24                shortDescription: shortDescriptionValue,
25                author: authorValue,
26                category: categoryValue,
27                isbn: isbnValue,
28                pageCount: pageCountValue,
29                price: priceValue,
30                publishedDate: publishedDateValue,
31                thumbnailUrl: thumbnailUrlValue,
32            },
33            function (error) {
34                if (error) {
35                    res.send("Data could not be saved." + error);
36                } else {
37                    res.send("");
38                }
39            }
40        );
41    }
42 });
```

7) การกำหนด endpoint สำหรับลบข้อมูลหนังสือ (delete)

```
1 //Delete a book by id
2 app.delete("/books/:bookid", function (req, res) {
3   res.setHeader("Content-Type", "application/json");
4   var bookid = Number(req.params.bookid);
5   var booksReference = db.ref("books/" + bookid);
6   if (booksReference !== null) {
7     booksReference.remove();
8     return res.send({
9       error: false,
10      message: "Delete book id =" + bookid.toString(),
11    });
12  }
13
14  if (error) throw error;
15 });
16
```

## การรันและเรียกใช้ API

- 1) โปรแกรม Node.js สามารถรันโดยใช้คำสั่ง `node <ไฟล์ที่ต้องการรัน>` เช่น `node index.js` เมื่อมีการแก้ไขไฟล์ จะต้องทำการรันใหม่ทุกครั้ง
- 2) หากไม่ต้องการรันใหม่เมื่อแก้ไขไฟล์ให้ติดตั้ง nodemon โดยใช้คำสั่ง `npm install -g nodemon` และรันโดยใช้คำสั่ง `nodemon <ไฟล์ที่ต้องการรัน>`
- 3) เมื่อโปรแกรมได้รันขึ้นมาแล้ว การเรียกดูผลสามารถทำได้โดยใช้เครื่องมือในการทดสอบ API เช่น Postman ในตัวอย่างนี้จะใช้ extension ของ Visual Studio Code ชื่อว่า Thunder Client โดยทำการสร้าง New Request และระบุ API ที่ต้องการเรียกใช้งานดังภาพ



- 4) นอกจาก Thunder Client สามารถใช้ extension ของ Visual Studio Code อีกตัวหนึ่งชื่อว่า Rest Client โดยสร้างไฟล์ `.http` เพื่อเก็บการเรียก API และทำการใช้งานโดยกด Send Request เพื่อแสดงผลดังภาพประกอบ

