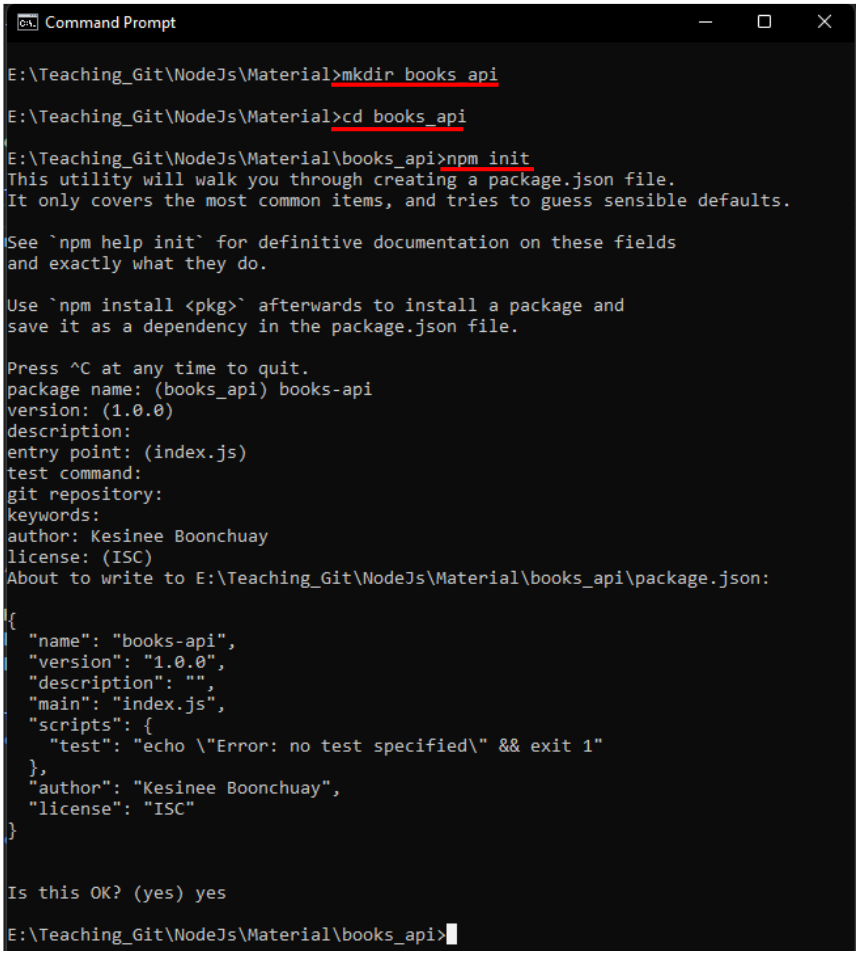
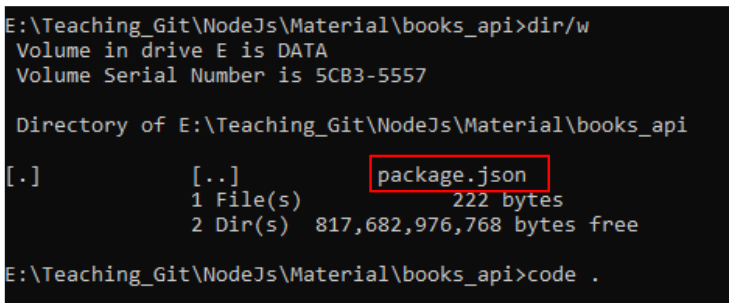
การสร้างโปรเจ็ค Node.js

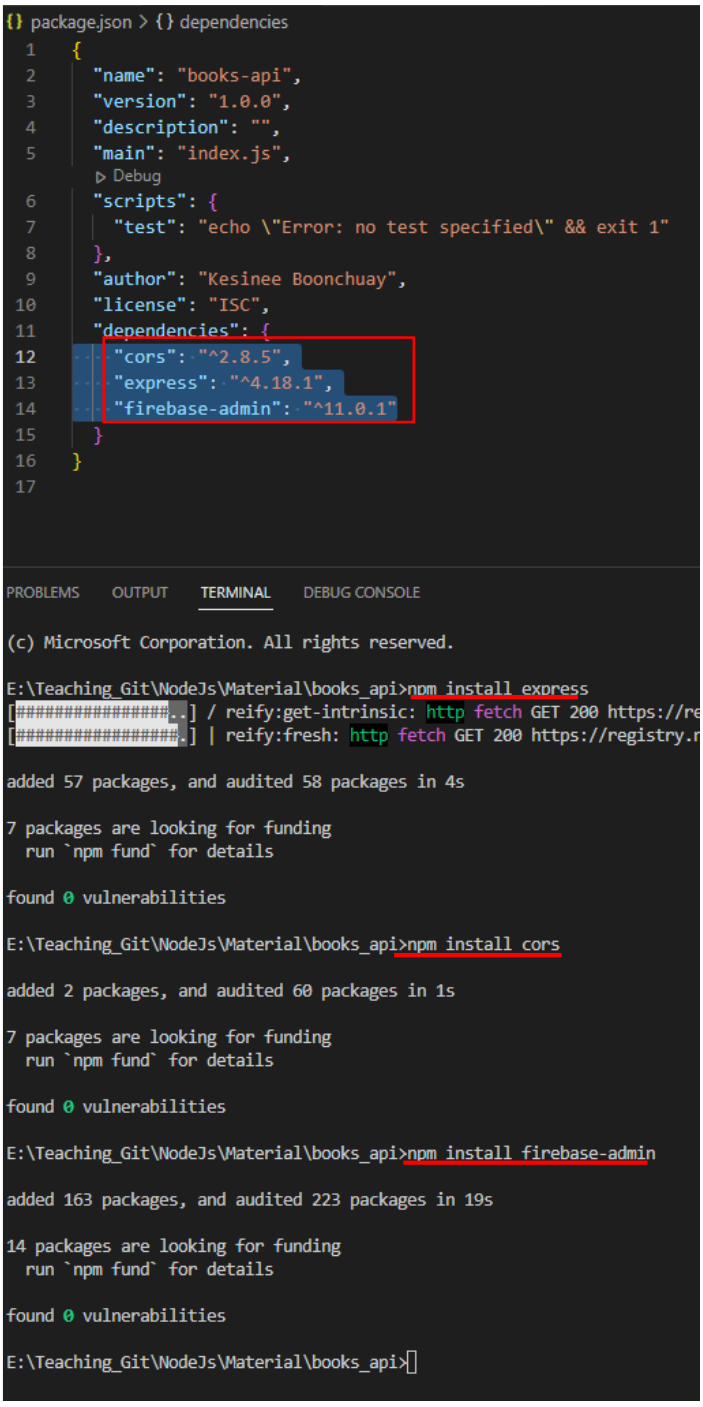
1) สร้าง directory สำหรับโปรเจ็ค
2) ใช้คำสั่ง npm init เพื่อเริ่มต้นโปรเจ็ค โดยระบุข้อมูลต่าง ๆ เพื่อใช้ในการสร้างไฟล์ package.json



3) เมื่อสำเร็จพบไฟล์ package.json ใน directory นั้น

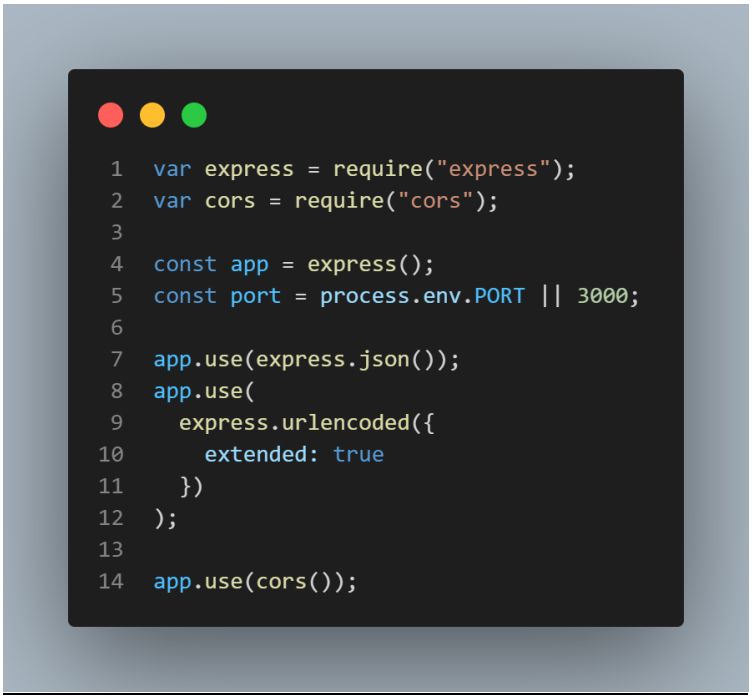


4) ติดตั้ง package ที่ใช้งานในโปรเจ็ค โดยใช้คำสั่ง npm install <ชื่อ package> โดยใช้โปรเจ็คนี้จะใช้ packages คือ express corrs และ firebase-admin เมื่อติดตั้งแล้ว จะพบชื่อ packages ดังกล่าวใน dependencies ของไฟล์ package.json (สามารถติดตั้งโดยการเพิ่มในไฟล์ package.json โดยตรงและสั่ง npm install หลังจากเพิ่มได้)



การสร้าง API (Book)

1) เริ่มต้นด้วยการเรียกใช้ package ต่างๆ สำหรับการพัฒนา API เช่น express และ corrs



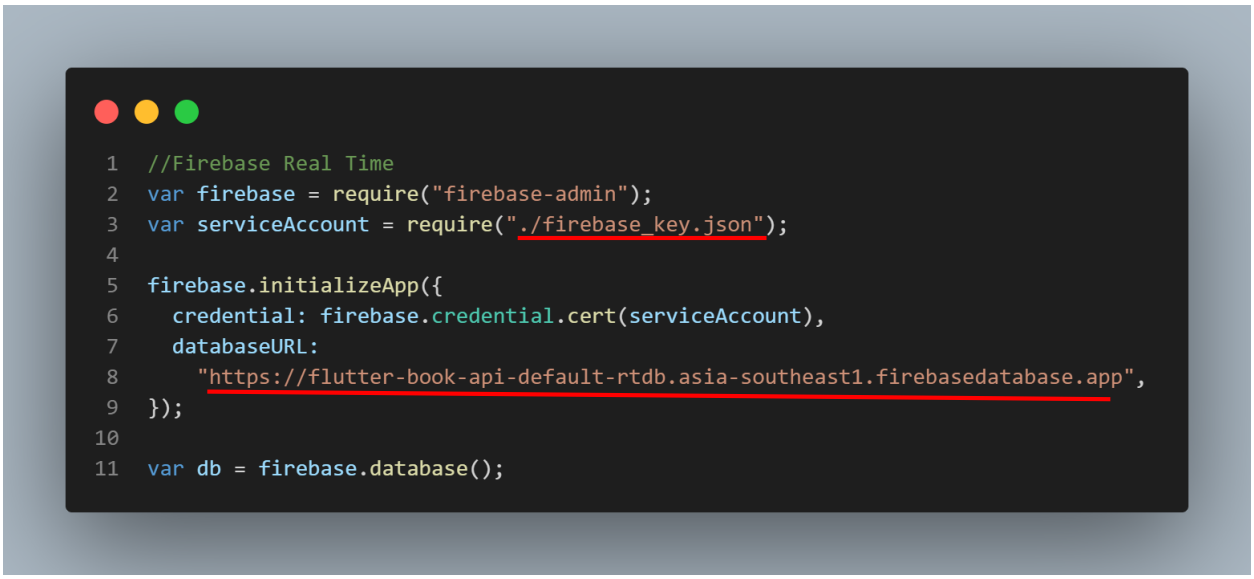
```
var express = require("express");
var cors = require("cors");

const app = express();
const port = process.env.PORT || 3000;

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

app.use(cors());
```

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



```
//Firebase Real Time
var firebase = require("firebase-admin");
var serviceAccount = require("./firebase_key.json");

firebase.initializeApp({
  credential: firebase.credential.cert(serviceAccount),
  databaseURL:
    "https://flutter-book-api-default-rtdb.asia-southeast1.firebasedatabase.app",
});

var db = firebase.database();
```

3) การกำหนด endpoint สำหรับเรียกดูข้อมูลหนังสือทั้งหมด (get)

```
//Get all books
app.get("/books", function (req, res) {
  res.setHeader("Content-Type", "application/json");

  var booksReference = db.ref("books");

  booksReference.on(
    "value",
    function (snapshot) {
      res.json(snapshot.val());
      booksReference.off("value");
    },
    function (errorObject) {
      res.send("The read failed: " + errorObject.code);
    }
  );
});
```

4) การกำหนด endpoint สำหรับเรียกดูข้อมูลหนังสือตาม id ที่ระบุ (get)

```
//Get a book by id
app.get("/books/:bookid", function (req, res) {
  res.setHeader("Content-Type", "application/json");
  var bookid = Number(req.params.bookid);
  var booksReference = db.ref("books");

  booksReference
    .orderByChild("bookid")
    .equalTo(bookid)
    .on(
      "child_added",
      function (snapshot) {
        res.json(snapshot.val());
        booksReference.off("value");
      },
      function (errorObject) {
        res.send("The read failed: " + errorObject.code);
      }
    );
});
```

5) การกำหนด endpoint สำหรับเพิ่มข้อมูลหนังสือใหม่ (post)

```
//Add new book
app.post("/books", function (req, res) {
  var bookidValue = req.body.bookid;
  var titleValue = req.body.title;
  var shortDescriptionValue = req.body.shortDescription;
  var authorValue = req.body.author;
  var categoryValue = req.body.category;
  var isbnValue = req.body.isbn;
  var pageCountValue = req.body.pageCount;
  var priceValue = req.body.price;
  var publishedDateValue = req.body.publishedDate;
  var thumbnailUrlValue = req.body.thumbnailUrl;

  var referencePath = "/books/" + bookidValue + "/";

  //Add to Firebase
  var bookReference = db.ref(referencePath);
  if (bookReference !== null) {
    bookReference.update(
      {
        bookid: bookidValue,
        title: titleValue,
        shortDescription: shortDescriptionValue,
        author: authorValue,
        category: categoryValue,
        isbn: isbnValue,
        pageCount: pageCountValue,
        price: priceValue,
        publishedDate: publishedDateValue,
        thumbnailUrl: thumbnailUrlValue,
      },
      function (error) {
        if (error) {
          res.send("Data could not be saved." + error);
        } else {
          res.send("");
        }
      }
    );
  }
});
```

6) การกำหนด endpoint สำหรับแก้ไขข้อมูลหนังสือ (put)

```
//Edit a book by id
app.put("/books/:bookid", function (req, res) {


  var bookidValue = Number(req.params.bookid);
  var titleValue = req.body.title;
  var shortDescriptionValue = req.body.shortDescription;
  var authorValue = req.body.author;
  var categoryValue = req.body.category;
  var isbnValue = req.body.isbn;
  var pageCountValue = req.body.pageCount;
  var priceValue = req.body.price;
  var publishedDateValue = req.body.publishedDate;
  var thumbnailUrlValue = req.body.thumbnailUrl;

  var referencePath = "/books/" + bookidValue + "/";

  var bookReference = db.ref(referencePath);
  if (bookReference !== null) {
    bookReference.update(
      {
        bookid: bookidValue,
        title: titleValue,
        shortDescription: shortDescriptionValue,
        author: authorValue,
        category: categoryValue,
        isbn: isbnValue,
        pageCount: pageCountValue,
        price: priceValue,
        publishedDate: publishedDateValue,
        thumbnailUrl: thumbnailUrlValue,
      },
      function (error) {
        if (error) {
          res.send("Data could not be saved." + error);
        } else {
          res.send("");
        }
      }
    );
  }
});
```

7) การกำหนด endpoint สำหรับลบข้อมูลหนังสือ (delete)

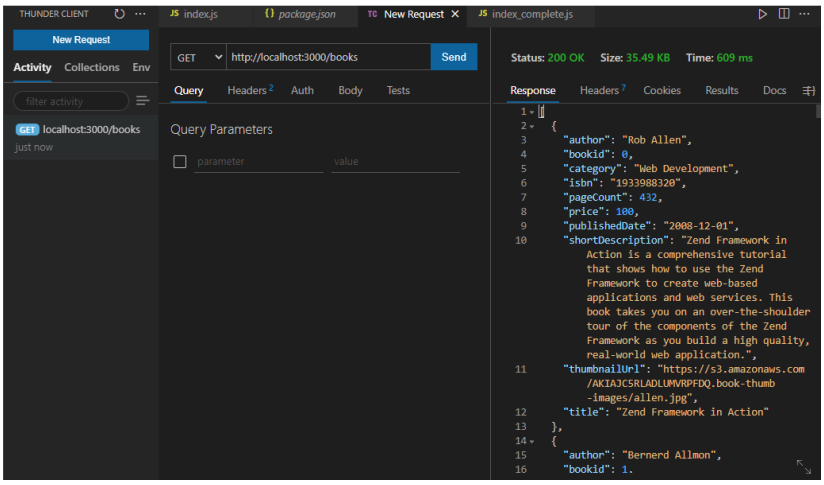
```
//Delete a book by id
app.delete("/books/:bookid", function (req, res) {
  res.setHeader("Content-Type", "application/json");
  var bookid = Number(req.params.bookid);
  var booksReference = db.ref("books/" + bookid);
  if (booksReference !== null) {
    booksReference.remove();
    return res.send({
      error: false,
      message: "Delete book id =" + bookid.toString(),
    });
  }

  if (error) throw error;
});

```

การรันและเรียกใช้ API

1) โปรเจ็ค Node.js สามารถรันโดยใช้คำสั่ง node <ไฟล์ที่ต้องการรัน> เช่น node index.js เมื่อมีการแก้ไขไฟล์ จะต้องทำการรันใหม่ทุกครั้ง
2) หากไม่ต้องการันใหม่เมื่อแก้ไขไฟล์ให้ติดตั้ง nodemon โดยใช้คำสั่ง npm install -g nodemon และรันโดยใช้คำสั่ง nodemon <ไฟล์ที่ต้องการรัน>
3) เมื่อโปรเจ็คได้รันขึ้นมาแล้ว การเรียกดูผลสามารถทำได้โดยใช้เครื่องเมื่อในการทดสอบ 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 เพื่อเก็บการเรียน API และทำการใช้งานโดยกด Send Request เพื่อแสดงผลดังภาพประกอบ