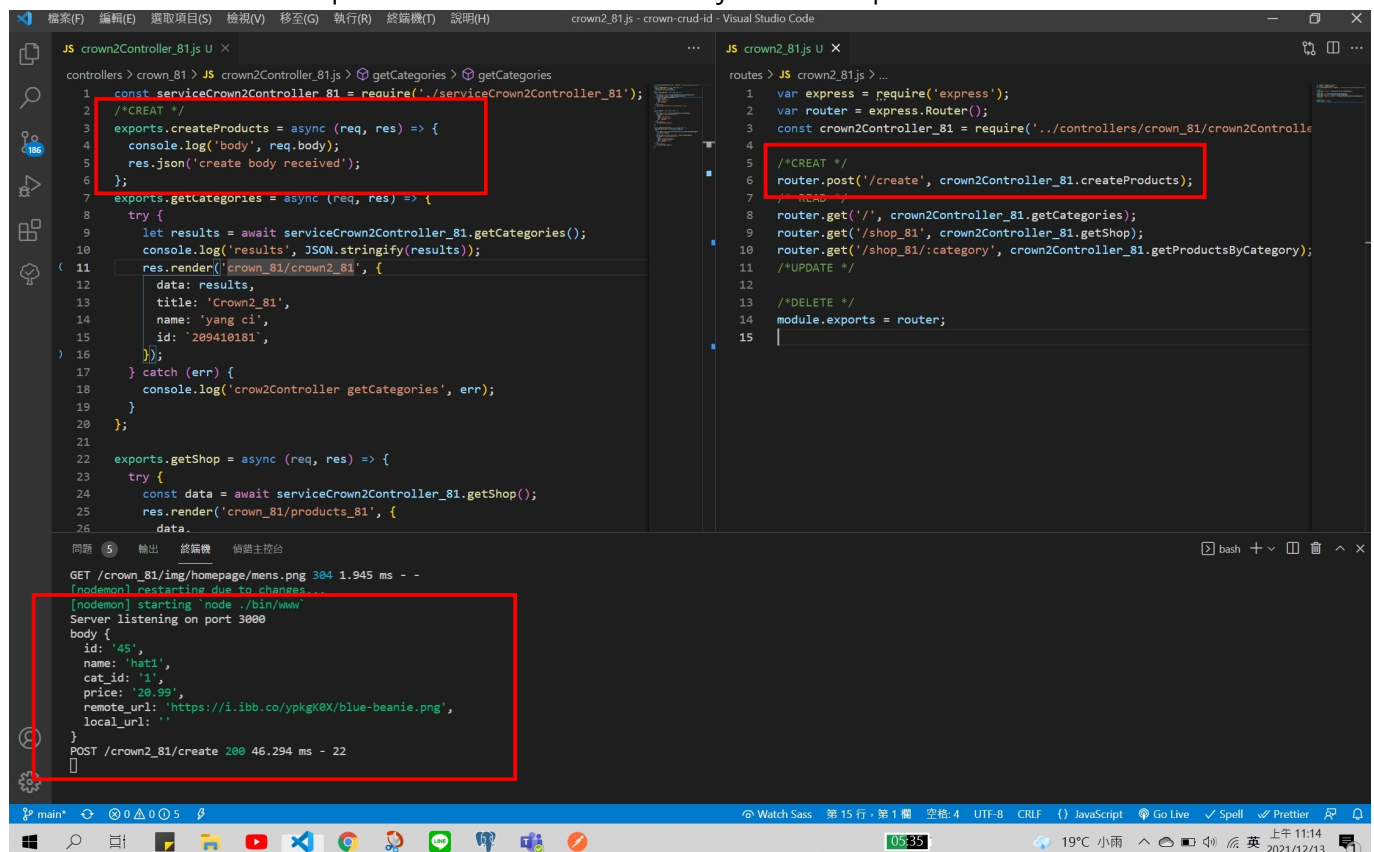


P1: Create json data of a product from Postman, and send it by POST request via route /crown2\_xx/create

=> Create a new POST request from Postman with a new json data of product



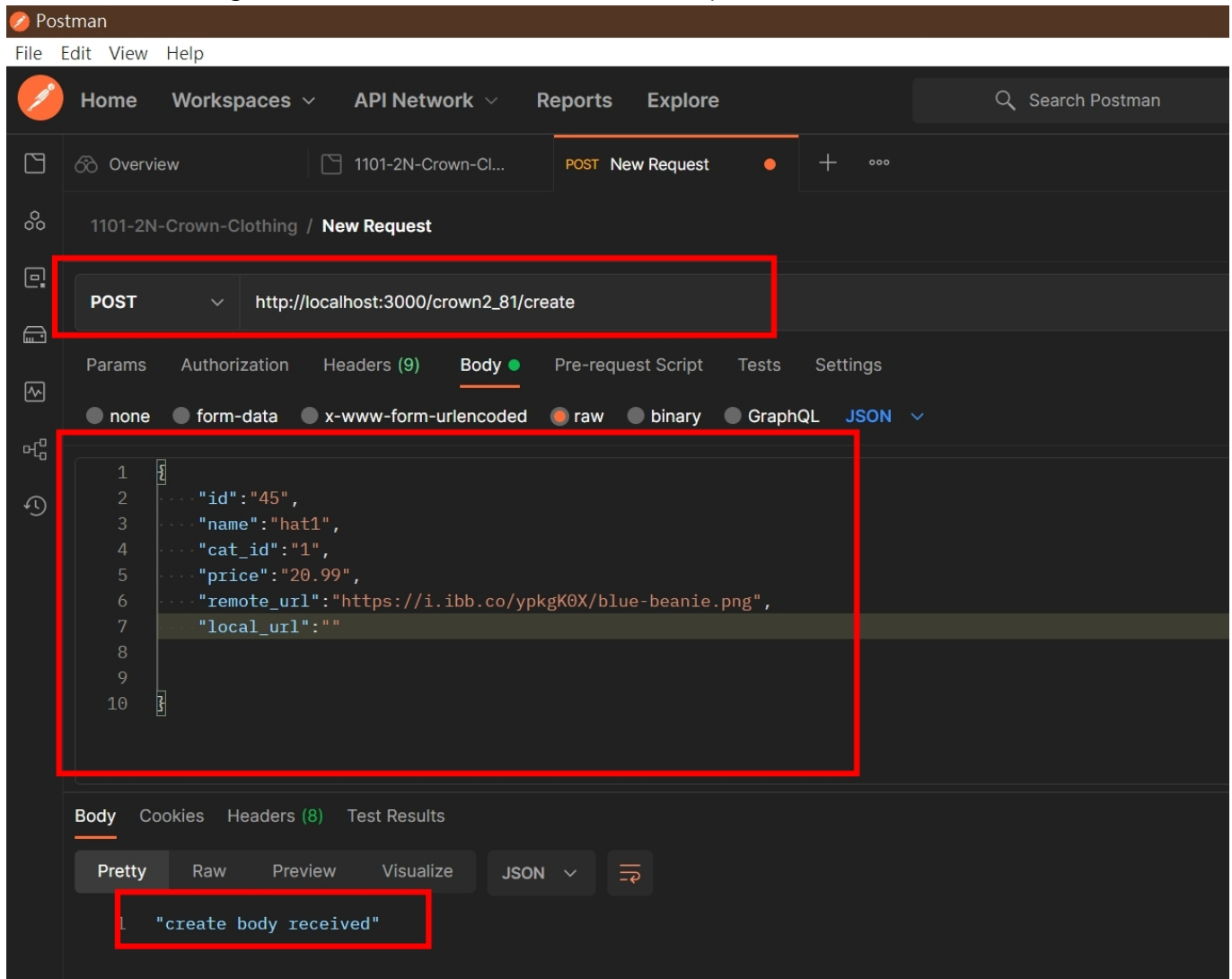
The screenshot shows the Visual Studio Code editor with two JavaScript files and a terminal window. The left file, `controllers/crown_81.js`, contains a `createProducts` function that logs the request body and sends a JSON response. The right file, `routes/crown2_81.js`, defines a `POST /create` route that calls `createProducts`. The terminal at the bottom shows the output of a GET request and a POST request, with the POST response body highlighted by a red box.

```
controllers/crown_81.js
1 const serviceCrown2Controller_81 = require('../serviceCrown2Controller_81');
2 /*CREAT */
3 exports.createProducts = async (req, res) => {
4   console.log('body', req.body);
5   res.json('create body received');
6 };
7 exports.getCategories = async (req, res) => {
8   try {
9     let results = await serviceCrown2Controller_81.getCategories();
10    console.log('results', JSON.stringify(results));
11    res.render('crown_81/crown2_81', {
12      data: results,
13      title: 'Crown2_81',
14      name: 'yang ci',
15      id: '209410181',
16    });
17   } catch (err) {
18     console.log('crow2Controller getCategories', err);
19   }
20 };
21
22 exports.getShop = async (req, res) => {
23   try {
24     const data = await serviceCrown2Controller_81.getShop();
25     res.render('crown_81/products_81', {
26       data:
27     });
28   } catch (err) {
29     console.log('crow2Controller getShop', err);
30   }
31 };
32
33 module.exports = serviceCrown2Controller_81;
```

```
routes/crown2_81.js
1 var express = require('express');
2 var router = express.Router();
3 const crown2Controller_81 = require('../controllers/crown_81/crown2Controller_81');
4
5 /*CREAT */
6 router.post('/create', crown2Controller_81.createProducts);
7 /*READ */
8 router.get('/', crown2Controller_81.getCategories);
9 router.get('/shop_81', crown2Controller_81.getShop);
10 router.get('/shop_81/:category', crown2Controller_81.getProductsByCategory);
11 /*UPDATE */
12
13 /*DELETE */
14 module.exports = router;
15
```

```
GET /crown_81/img/homepage/mens.png 304 1.945 ms - -
[nodemon] restarting due to changes...
Server listening on port 3000
body {
  id: '45',
  name: 'hat1',
  cat_id: '1',
  price: '20.99',
  remote_url: 'https://i.ibb.co/ypkgK0X/blue-beanie.png',
  local_url: ''
}
POST /crown2_81/create 200 46.294 ms - 22
```

=> code for handling route /crown2\_xx/create with console output



P2: Create code for creating a product, and verify

=> crown2Controller code

```

JS crown2Controller_81.js U X JS serviceCrown2Controller_81.js U JS Shop_81.js U
controllers > crown_81 > JS crown2Controller_81.js > createProducts > createProducts
1  const serviceCrown2Controller_81 = require('./serviceCrown2Controller_81');
2  /*CREAT */
{ 3  exports.createProducts = async (req, res) => {
4      console.log('body', req.body);
5      //res.json('create body received');
6      try {
7          let results = await serviceCrown2Controller_81.create(req.body);
8          console.log('results', JSON.stringify(results));
9          res.json('Data: {msg:creating success}');
10     } catch (err) {}
11 };
12 exports.getCategories = async (req, res) => {
13     try {
14         let results = await serviceCrown2Controller_81.getCategories();
15         console.log('results', JSON.stringify(results));
16         res.render('crown_81/crown2_81', {
17             data: results,
18             title: 'Crown2_81',
19             name: 'yang ci',
20             id: `209410181`,
21         });
22     } catch (err) {
23         console.log('crow2Controller getCategories', err);
24     }
25 };

```

=> serviceController code

```

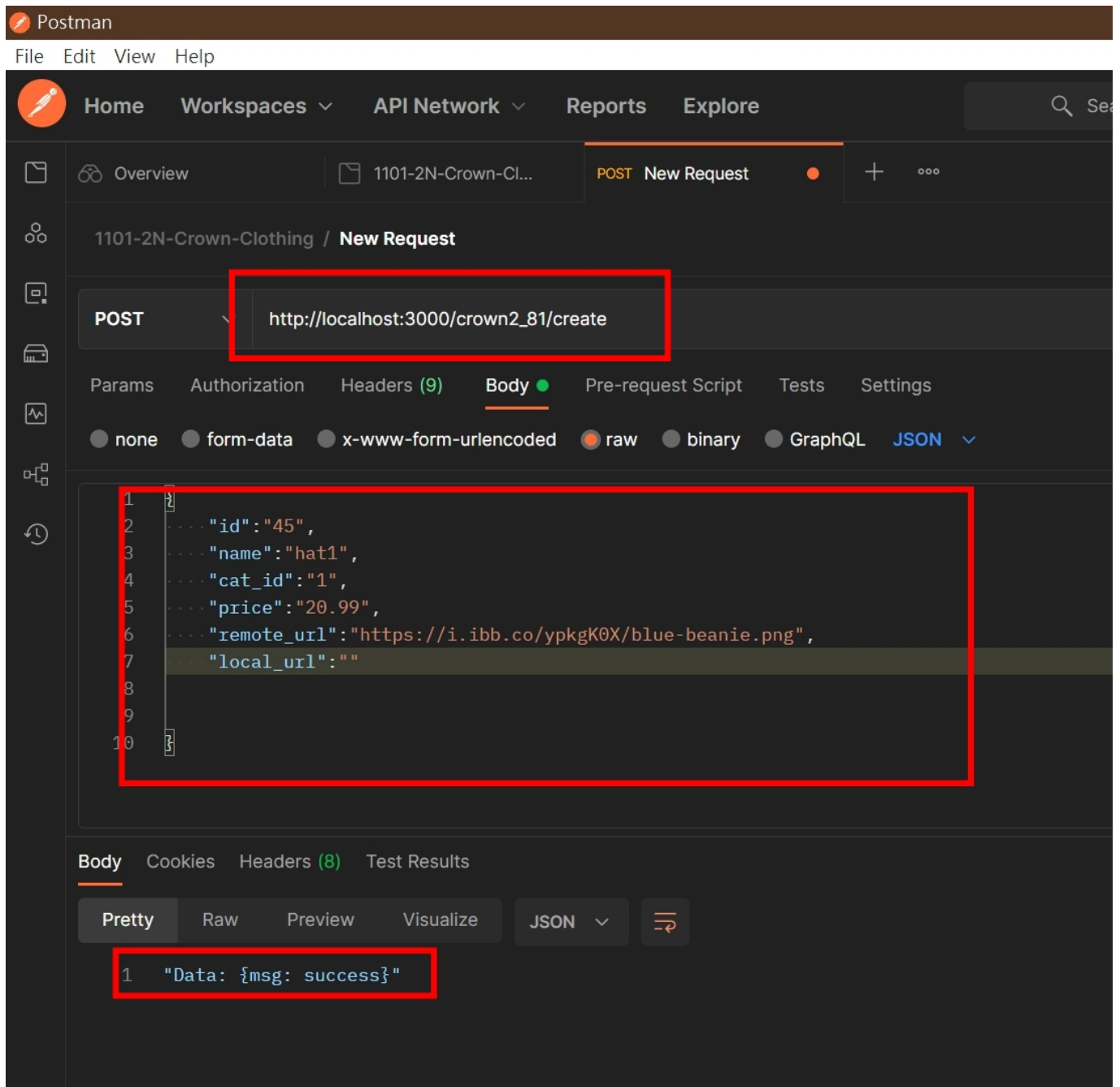
JS crown2Controller_81.js U JS serviceCrown2Controller_81.js U X JS Shop_81.js U
controllers > crown_81 > JS serviceCrown2Controller_81.js > getCategories > getCategories
1  const Category_81 = require('../../models/crown_81/Category_81');
2  const Shop_81 = require('../../models/crown_81/Shop_81');
3  /*CREATE */
4  exports.create = async (body) => {
5      try {
6          return await Shop_81.create(body);
7      } catch (err) {
8          console.log('create', err);
9      }
10 };
11 /*READ */

```

=> model Shop\_xx code

```
models > crown_81 > JS Shop_81.js > [?] Shop_81 > create
1  const db = require('../../utils/database');
2
3  const Shop_81 = class Shop_81 {
4    constructor(id, name, cat_id, price, remote_url, local_url) {
5      this.id = id;
6      this.name = name;
7      this.cat_id = cat_id;
8      this.price = price;
9      this.remote_url = remote_url;
10     this.local_url = local_url;
11   }
12   //CREATE
13   static async create(body) {
14     const { id, name, cat_id, price, remote_url, local_url } = body;
15     const query = {
16       text: `INSERT INTO shop_81(id, name, cat_id, price, remote_url, local_url)`,
17       values: [id, name, cat_id, price, remote_url, local_url],
18     };
19     return db.query(query);
20   }
21 }
```

=> Postman request, json data and result



=> Verify by pgAdmin

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Servers' tree is expanded to show the 'public' schema under the 'crown\_81' database. The 'Tables (2)' folder is highlighted, showing 'category\_81' and 'shop\_81'. The 'shop\_81' table is selected, and its data is displayed in the 'Data Output' tab. The query editor shows the following SQL:

```
1 SELECT * FROM public.shop_81
2 ORDER BY id ASC
```

The query result is a table with the following columns: id, name, cat\_id, price, remote\_url, and local\_url. The data is as follows:

id	name	cat_id	price	remote_url	local_url
27	32 Striped Sweater	4	45	https://i.libb.co/KmSkMbH/striped-sweater.png	/img/womens/striped-sweater.png
28	33 Yellow Track Suit	4	135	https://i.libb.co/v1cwNf/yellow-track-suit.png	/img/womens/yellow-track-suit.png
29	34 White Blouse	4	20	https://i.libb.co/qBcrsJg/white-vest.png	/img/womens/white-vest.png
30	35 Camo Down Vest	5	325	https://i.libb.co/xJS0T3Y/camo-vest.png	/img/mens/camo-vest.png
31	36 Floral T-shirt	5	20	https://i.libb.co/qMQ75QZ/floral-shirt.png	/img/mens/floral-shirt.png
32	37 Black & White Longsleeve	5	25	https://i.libb.co/55z32tw/long-sleeve.png	/img/mens/long-sleeve.png
33	38 Pink T-shirt	5	25	https://i.libb.co/RvwnBL8/pink-shirt.png	/img/mens/pink-shirt.png
34	39 Jean Long Sleeve	5	40	https://i.libb.co/VpW4x5t/roll-up-jean-shirt.png	/img/mens/roll-up-jean-shirt.png
35	40 Burgundy T-shirt	5	25	https://i.libb.co/mhQVWt6/polka-dot-shirt.png	/img/mens/polka-dot-shirt.png
36	45 hat1	1	20.99	https://i.libb.co/ypkgK0X/blue-beanie.png	