

[Github URL](#)

W04-P1: Create a express web server to show your info

The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with files like 'client_blogs_14', 'server_blogs_14', '.env', and 'package-lock.json'. The main area has two code editors side-by-side. The left editor contains 'server.js' with code to set up an Express app and handle a root route. The right editor contains 'package.json' with details about the project. Below the editors is the Terminal tab, which shows the command 'node -r server.js' being run, followed by output from 'nodemon' indicating it's watching files and starting a server on port 5000. To the right of the code editors is a browser window showing the local host at port 5000, displaying the name 'liangtinglin'.

```

1 import express from "express";
2 const app = express();
3 const port = process.env.PORT || 5000;
4 app.use(express.json());
5 app.get("/", (req, res) => {
6   res.send("liangtinglin, 913410014");
7 }
8 app.listen(port, () => {
9   console.log(`Server is running on http://localhost:${port}`);
10})
11
12
13
14
15
16
17
18
19
20
  
```

```

{
  "name": "server_blogs_14",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "type": "module",
  "scripts": {
    "start": "node server.js",
    "dev": "nodemon server.js",
    "test": "echo \\"$Error: no test specified\\""
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^5.1.0",
    "nodemon": "^3.1.10"
  }
}
  
```

cfffb896 zero2005x Wed Oct 8 18:51:20 2025 +0800 W04-P1: Create a express web server to show your info

W04-P2: Create blog_xx table with 3 data, implement route /api/blog_xx to return a json array with 3 data

=> **SQL to create blog_xx table and 3 data**

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure, including the `wp1_demo_14` database and its `blog_14` table.
- Query Editor:** Contains the SQL query used to insert data into the table. The query inserts two rows of data, each with a title, description, category, and image URL.
- Data Output:** Shows the results of the query, indicating 9 rows inserted successfully.
- Messages:** Displays a message stating "Query returned successfully in 43 msec."
- Status Bar:** Shows "Total rows: 9" and "Query complete 00:00:00.043".
- Bottom Right:** Includes "CRLF" and "Ln 90, Col 7" indicators.

```

79      'travel',
80      '/images/photo-8.jpg',
81      'https://erogcveccbzsyhbputf.supabase.co/storage/v1/object/public/demo-xx/card-xx/photo-8.jpg'
82    ),
83    (
84      9,
85      'Rocks Of Queen Head In Yehliu Taiwan',
86      'Lorem ipsum dolor sit amet consectetur adipisicing elit.',
87      'travel',
88      '/images/photo-9.jpg',
89      'https://erogcveccbzsyhbputf.supabase.co/storage/v1/object/public/demo-xx/card-xx/photo-9.jpg'
90    );
  
```

=> show 3 data

The screenshot shows the pgAdmin 4 interface with the following details:

- Object Explorer:** Shows the database structure, including the `public.blog_14` table.
- Query Editor:** Contains the SQL query used to select all data from the `blog_14` table, ordered by `id` in ascending order.
- Data Output:** Shows the results of the query, displaying 9 rows of data in a table format.
- Table Headers:** The columns are `id`, `title`, `descrip`, `category`, `img`, and `remote_img`.
- Table Data:**

	<code>id</code> [PK] integer	<code>title</code> character varying (255)	<code>descrip</code> text	<code>category</code> character varying (255)	<code>img</code> text	<code>remote_img</code> text
1	1	Seven Reasons Why Coffee Is Aweso...	Lorem ipsum dolor sit amet consectetur adipisicing elit.	lifestyle	/images/photo-1.jpg	https://erogcve...
2	2	Travel To Paris	Lorem ipsum dolor sit amet consectetur adipisicing elit.	travel	/images/photo-2.jpg	https://erogcve...
3	3	Coffee Brings Friendship	Lorem ipsum dolor sit amet consectetur adipisicing elit.	lifestyle	/images/photo-3.jpg	https://erogcve...
4	4	Coffee Make You Feel Good	Lorem ipsum dolor sit amet consectetur adipisicing elit.	lifestyle	/images/photo-4.jpg	https://erogcve...
5	5	Coffee Make You Calm	Lorem ipsum dolor sit amet consectetur adipisicing elit.	lifestyle	/images/photo-5.jpg	https://erogcve...
6	6	101 Tower In Taipei	Lorem ipsum dolor sit amet consectetur adipisicing elit.	travel	/images/photo-6.jpg	https://erogcve...
7	7	Sun Rise From The Mountain	Lorem ipsum dolor sit amet consectetur adipisicing elit.	travel	/images/photo-7.jpg	https://erogcve...
8	8	Serene Lake With Trees	Lorem ipsum dolor sit amet consectetur adipisicing elit.	travel	/images/photo-8.jpg	https://erogcve...
- Status Bar:** Shows "Total rows: 9" and "Query complete 00:00:00.105".
- Bottom Right:** Includes "CRLF" and "Ln 1, Col 1" indicators.

=> implement route /api/blog_xx

The screenshot shows the VS Code interface with two tabs open: 'server.js' and 'JSON Editor'. The 'server.js' tab contains the Node.js code for the server, which includes importing express and database, setting up a port, and defining a route to return a JSON array of blog posts. The 'JSON Editor' tab shows a JSON array with three blog entries, each containing fields like id, title, descrip, category, img, and remote_img.

```

1 import express from "express";
2 import db from "./utils/database.js";
3 const app = express();
4 const port = process.env.PORT || 5000;
5
6 app.use("/api/blogs_14", async (req, res, next) => {
7   const results = await db.query("SELECT * FROM blog_14");
8   console.log("results", JSON.stringify(results));
9   res.json(results.rows);
10 }
11 )
12
13 app.get("/", (req, res, next) => {
14   res.send("liangtinglin, 913410014");
15 }
16 )
17
18 app.listen(port, () => {
19   console.log(`Server is running on http://localhost:${port}`);
20 }
)

```

```

{
  "id": 1,
  "title": "Seven Reasons Why Coffee Is Awesome",
  "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit.",
  "category": "lifestyle",
  "img": "/images/photo-1.jpg",
  "remote_img": "https://erogcveccbzsyhbgputf.supabase.co/storage/v1/objectxx/card-xx/photo-1.jpg"
},
{
  "id": 2,
  "title": "Travel To Paris",
  "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit.",
  "category": "travel",
  "img": "/images/photo-2.jpg",
  "remote_img": "https://erogcveccbzsyhbgputf.supabase.co/storage/v1/objectxx/card-xx/photo-2.jpg"
},
{
  "id": 3,
  "title": "Coffee Brings Friendship",
  "descrip": "Lorem ipsum dolor sit amet consectetur adipisicing elit.",
  "category": "lifestyle",
  "img": "/images/photo-3.jpg"
}

```

893ac84 zero2005x Wed Oct 8 19:52:38 2025 +0800 W04-P2: Create blog_xx table with 3 data, implement route /api/blog_xx to return a json array with 3 data

W04-P3: from client side to get json data from Node

=> modified client and server code

The screenshot shows the VS Code interface with two tabs open: 'BlogNodePage_14.jsx' and 'server.js'. The 'BlogNodePage_14.jsx' tab contains the React component code for displaying blog posts. It uses useState and useEffect hooks to fetch data from the '/api/blogs_14' endpoint and update the state. The 'server.js' tab shows the Node.js server code, which is identical to the previous screenshot but includes a catch block for errors in the fetch operation.

```

1 import { FaGlobe } from "react-icons/fa";
2 import { FaMugSaucer } from "react-icons/fa6";
3
4 import Blog_14 from "../components/Blog_14";
5 import { useState, useEffect } from "react";
6
7 const api_url = "http://localhost:3000/api/blogs_14";
8
9 const BlogNodePage_14 = () => {
10   const [name, setName] = useState("liangtinglin");
11   const [id, setId] = useState(913410014);
12   const [blogs_14, setBlogs_14] = useState([]);
13
14   const fetchBlogFromNodeServer = async () => {
15     try {
16       const response = await fetch(api_url);
17       const data = await response.json();
18       console.log("data from node server", data);
19       setBlogs_14(data);
20     } catch (error) {
21       console.log("fetchBlogFromNodeServer error", error);
22     }
23   };
24
25   useEffect(() => {
26     fetchBlogFromNodeServer();
27   }, []);
28
29   return (
30     <>
31       <section>...
32         <div>...
33           <h2>
34             | Blog from Node Page -- {name}, {id}

```

```

1 import express from "express";
2 import db from "./utils/database.js";
3 import cors from "cors";
4
5 const app = express();
6 const port = process.env.PORT || 5000;
7
8 app.use(cors());
9
10 app.use("/api/blogs_14", async (req, res, next) => {
11   const results = await db.query("SELECT * FROM blog_14");
12   console.log("results", JSON.stringify(results));
13   res.json(results.rows);
14 }
15 );
16
17 app.get("/", (req, res, next) => {
18   res.send("liangtinglin, 913410014");
19 }
20 );
21
22 app.listen(port, () => {
23   console.log(`Server is running on http://localhost:${port}`);
24 }
)

```

=> Chrome, show 3 blogs

Blog From Node Page -- Liangtinglin, 913410014

```

data BlogHomePage_14.jsx:18
from node server
  ▼ (9) [{}]
    ▼ (9) [{}]
      ▼ (9) [{}]
        ▷ 0: {id: 1, title: 'Seven Re'}
        ▷ 1: {id: 2, title: 'Travel T'}
        ▷ 2: {id: 3, title: 'Coffee B'}
        ▷ 3: {id: 4, title: 'Coffee M'}
        ▷ 4: {id: 5, title: 'Coffee M'}
        ▷ 5: {id: 6, title: '101 Tow'}
        ▷ 6: {id: 7, title: 'Sun Rise'}
        ▷ 7: {id: 8, title: 'Serene L'}
        ▷ 8: {id: 9, title: 'Rocks Of'}
        ▷ length: 9
      ▶ [[Prototype]]: Array(0)

data BlogHomePage_14.jsx:18
from node server
  ▼ (9) [{}]
    ▼ (9) [{}]
      ▼ (9) [{}]
        ▷ 0: {id: 1, title: 'Seven Re'}
        ▷ 1: {id: 2, title: 'Travel T'}
        ▷ 2: {id: 3, title: 'Coffee B'}
        ▷ 3: {id: 4, title: 'Coffee M'}
        ▷ 4: {id: 5, title: 'Coffee M'}
        ▷ 5: {id: 6, title: '101 Tow'}
        ▷ 6: {id: 7, title: 'Sun Rise'}
        ▷ 7: {id: 8, title: 'Serene L'}
        ▷ 8: {id: 9, title: 'Rocks Of'}
        ▷ length: 9
      ▶ [[Prototype]]: Array(0)
  
```

ed89ef0 zero2005x Wed Oct 8 20:47:12 2025 +0800 W04-P3: from client side
to get json data from Node

W04-logs: git logs of w04

Commits

- main
- All users
- All time

- Commits on Oct 8, 2025
 - W04-P3: from client side to get json data from Node**
zero2005x committed now
 - W04-P2: Create blog_xx table with 3 data, implement route /api/blog_xx to return a json array with 3 data**
zero2005x committed 1 hour ago
 - W04-P1: Create a express web server to show your info**
zero2005x committed 2 hours ago
- Commits on Oct 6, 2025
 - refactor: Improve code formatting and consistency in App_14.jsx**
zero2005x committed 2 days ago
- Commits on Oct 1, 2025
 - fix: Update .gitignore to include additional directories and files**
<https://github.com/zero2005x/1141-2N-DEMO-LIANGTINGLIN-14>