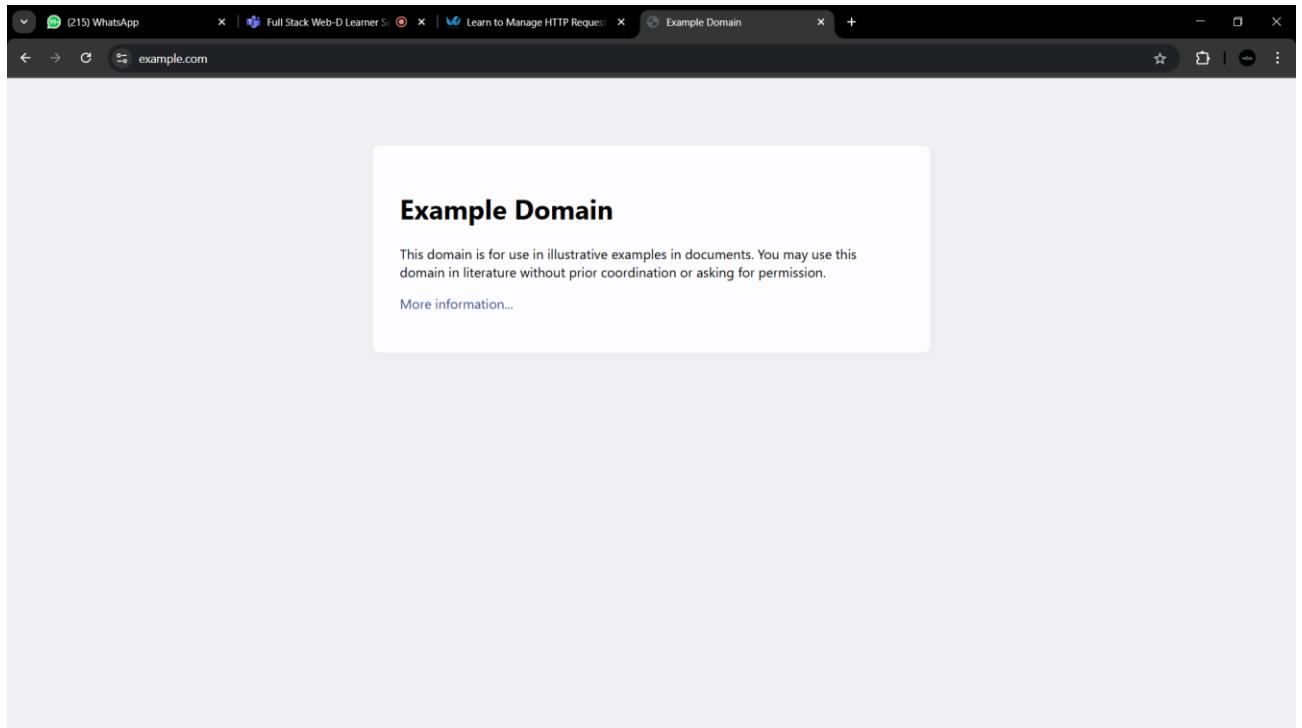


# June 10, 2025



(215) WhatsApp | Full Stack Web-D Learner | Learn to Manage HTTP Requests | Example Domains

ianao.org/help/example-domains

## Example Domains

As described in [RFC 2606](#) and [RFC 6761](#), a number of domains such as example.com and example.org are maintained for documentation purposes. These domains may be used as illustrative examples in documents without prior coordination with us. They are not available for registration or transfer.

We provide a web service on the example domain hosts to provide basic information on the purpose of the domain. These web services are provided as best effort, but are not designed to support production applications. While incidental traffic for incorrectly configured applications is expected, please do not design applications that require the example domains to have operating HTTP service.

### Further Reading

- [IANA-managed Reserved Domains](#)

Last revised 2017-05-13.

Domain Names Root Zone Registry .INT Registry .arpa Registry IDN Repository  
Number Resources Abuse Information  
Protocols Protocol Registries Time Zone Database  
About Us News Performance Excellence Archive Contact Us

The IANA functions coordinate the Internet's globally unique identifiers, and are provided by [Public Technical Identifiers](#), an affiliate of [ICANN](#).

File Edit Selection View Go Run Terminal Help ↺ → ⌘ my-express-app

```
js app.js > ...
js app.js > ...
1 // Require the Express module
2 const express = require('express');
3
4 // Create an Express application
5 const app = express();
6
7 // Define a route that redirects to a new URL
8 app.get('/redirect', (req, res) => {
9   // specify the URL to which the response should redirect
10   res.redirect('https://www.example.com');
11 });
12
13 // Start the server
14 app.listen(3000, () => {
15   console.log(`Server is running on http://localhost:3000`);
16 });
17 |
```

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

Server is running on http://localhost:3000

node + v 🔍 ... ^ ×

Launchpad Live Share RHDA analysis has failed

Ln 17, Col 1 Spaces: 4 UTF-8 CRLF { JavaScript Signed out Go Live Quokka

The screenshot shows a development environment with a code editor and a browser window.

**Code Editor:**

- File: app.js
- Content:

```
js app.js > ...
1 // Require Express and path modules
2 const express = require('express');
3 const path = require('path');
4
5 // Create an Express application
6 const app = express();
7
8 // Set the view engine to ejs and specify the views directory
9 app.set('view engine', 'ejs');
10 app.set('views', path.join(__dirname, 'views'));
11
12 // Define a route to render an ejs view
13 app.get('/render', (req, res) => {
14   // Render the 'index' view with a title variable
15   res.render('index', { title: 'Express Render Example' });
16 });
17
18 // Start the server
19 app.listen(3000, () => {
20   console.log('Server is running on http://localhost:3000');
21 });
22
```

**Browser:**

- Address bar: localhost:3000/render
- Page content: Hello!

Below the browser window, the terminal shows the command node and the current file path E:\my-express-app\app.js:552:12.

Hello!

The screenshot shows a dark-themed code editor interface, likely Visual Studio Code, with the following details:

- File Explorer:** Shows the project structure under "OPEN EDITORS".
  - app.js**
  - index.ejs** (selected)
  - views**:
    - index.html**
    - index.ejs**
  - node\_modules**
  - public**:
    - index.html**
  - views**:
    - index.ejs**
  - app.js**
  - package-lock.json**
  - package.json**
- Terminal:** Shows the command "node" followed by a series of symbols.
- Browser Preview:** A browser window titled "localhost:3000/file" displays the content of "index.ejs". The page contains the following HTML:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
  </head>
  <body>
    <h1>Hello!</h1>
  </body>
</html>
```

```
File Edit Selection View Go Run Terminal Help < > my-express-app
app.js <-- index.ejs
app.js > ...
1 // Require Express and path modules
2 const express = require('express');
3 const path = require('path');
4
5 // Create an Express application
6 const app = express();
7
8 // Define a route to send a file
9 app.get('/file', (req, res) => {
10   // Set the path to the file
11   const filePath = path.join(__dirname, 'example.pdf');
12
13   // Send the file at the specified path
14   res.sendFile(filePath, function(err) {
15     if (err) {
16       console.log('Error sending file:', err);
17       res.status(500).send('Error sending file');
18     } else {
19       console.log('File sent successfully');
20     }
21   });
22 });
23
24 // Start the server
25 app.listen(3000, () => {
26   console.log('Server is running on http://localhost:3000');
27 });
28
...

```

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

Ln 28, Col 1 Spaces: 4 UTF-8 CRLF JavaScript Signed out Go Live Quokka

File Edit Selection View Go Run Terminal Help < > my-express-app
EXPLORER ... app.js index.html example.pdf
OPEN EDITORS app.js example.pdf
index.html public
example.pdf
MY-EXPRESS-APP
node\_modules
public
index.html
views
index.ejs
app.js
example.pdf
package-lock.json
package.json
JSON

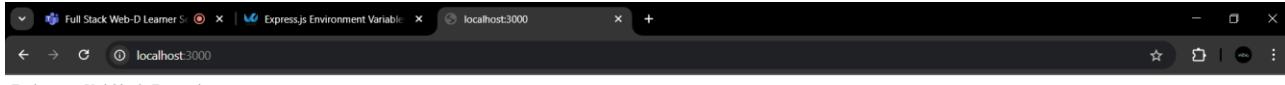
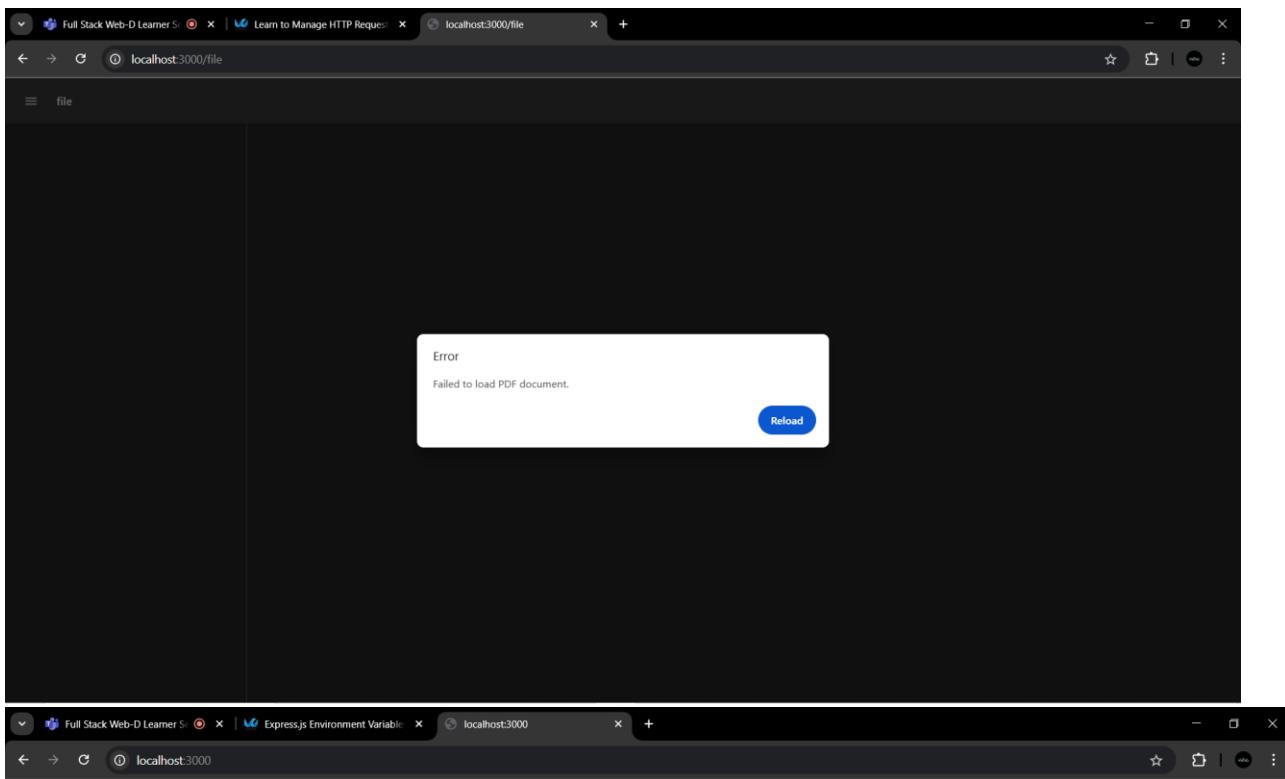
Open chat (Ctrl+I), or select a language (Ctrl+K M), or fill with template to get started.  
Start typing to dismiss or don't show this again.

Do you want to install the recommended 'vscode-pdf' extension from tomoki1207 for example.pdf?  
Install Show Recommendations

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Plain Text Signed out Go Live Quokka

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Plain Text Signed out Go Live Quokka



```
File Edit Selection View Go Run Terminal Help <- > my-express-app
OPEN EDITORS JS app.js .env
MY-EXPRESS-APP
node_modules
public
index.html
views
index.ejs
.env
app.js
example.pdf
package-lock.json
package.json

JS app.js > ...
1 require('dotenv').config(); // Load .env file 6.4k (gzipped: 2.8k)
2
3 const express = require('express');
4 const app = express();
5
6 const PORT = process.env.PORT || 3000; // Use environment variable or default to 3000
7
8 app.get('/', (req, res) => {
9   res.send('Environment Variables in Express.js');
10 });
11
12 app.listen(PORT, () => {
13   console.log(`Server is running on port ${PORT}`);
14 });

Terminal
node + - x
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
Server is running on port 3000
Ln 14, Col 4 Spaces: 4 UTF-8 JavaScript Signed out Go Live Quokka
```

```
File Edit Selection View Go Run Terminal Help <- > my-express-app
OPEN EDITORS JS app.js .env
MY-EXPRESS-APP
node_modules
public
index.html
views
index.ejs
.env
app.js
example.pdf
package-lock.json
package.json

JS app.js > ...
1 # Application Settings
2 PORT=3000
3 HOST=127.0.0.1
4 PAGESIZE_DEFAULT=15
5
6 # Environment Mode
7 ENVIRONMENT=development
8
9 # MySQL Database Configuration
10 DB_HOST=localhost
11 DB_USER=root
12 DB_PASSWORD=password
13 DB_NAME=example_db
14 MYSQL_PORT=3306

Terminal
node + - x
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
Server is running on port 3000
Ln 14, Col 4 Spaces: 4 UTF-8 JavaScript Signed out Go Live Quokka
```

```
File Edit Selection View Go Run Terminal Help <- > my-express-app
OPEN EDITORS JS app.js .env
MY-EXPRESS-APP
node_modules
public
index.html
views
index.ejs
.env
app.js
example.pdf
package-lock.json
package.json

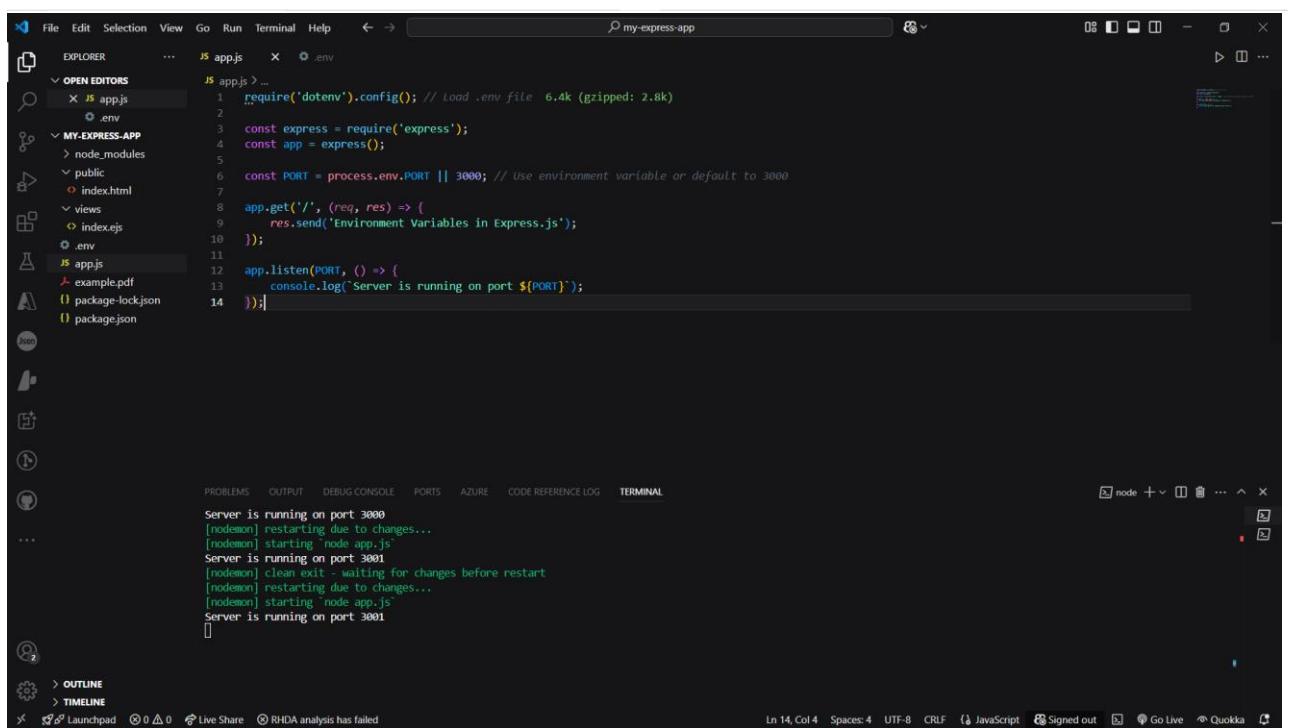
JS app.js > ...
1 # Application Settings
2 PORT=3000
3 HOST=127.0.0.1
4 PAGESIZE_DEFAULT=15
5
6 # Environment Mode
7 ENVIRONMENT=development
8
9 # MySQL Database Configuration
10 DB_HOST=localhost
11 DB_USER=root
12 DB_PASSWORD=password
13 DB_NAME=example_db
14 MYSQL_PORT=3306

Terminal
node + - x
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
Server is running on port 3000
Ln 14, Col 4 Spaces: 4 UTF-8 JavaScript Signed out Go Live Quokka
```

```
File Edit Selection View Go Run Terminal Help <- > my-express-app
OPEN EDITORS JS app.js .env
MY-EXPRESS-APP
node_modules
public
index.html
views
index.ejs
.env
app.js
example.pdf
package-lock.json
package.json

JS app.js > ...
1 # Application Settings
2 PORT=3000
3 HOST=127.0.0.1
4 PAGESIZE_DEFAULT=15
5
6 # Environment Mode
7 ENVIRONMENT=development
8
9 # MySQL Database Configuration
10 DB_HOST=localhost
11 DB_USER=root
12 DB_PASSWORD=password
13 DB_NAME=example_db
14 MYSQL_PORT=3306

Terminal
node + - x
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
Server is running on port 3000
Ln 14, Col 4 Spaces: 4 UTF-8 JavaScript Signed out Go Live Quokka
```



File Edit Selection View Go Run Terminal Help ↻ → ⌘ my-express-app

OPEN EDITORS JS app.js .env process.env

MY-EXPRESS-APP node\_modules public views index.html index.ejs .env app.js example.pdf package-lock.json package.json process.env

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```
[nodemon] starting "node app.js"
Database Host: undefined
Environment Mode: undefined
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Database Host: undefined
Environment Mode: undefined
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Debug mode enabled
[nodemon] clean exit - waiting for changes before restart
```

Ln 6, Col 2 Spaces: 4 UTF-8 CRLF ⓘ JavaScript ⓘ Signed out ⓘ Go Live ⓘ Quokka ⓘ

File Edit Selection View Go Run Terminal Help ↻ → ⌘ my-express-app

OPEN EDITORS JS config.js .env process.env

MY-EXPRESS-APP node\_modules public views index.html index.ejs .env app.js config.js example.pdf package-lock.json package.json process.env

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Debug mode enabled
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Debug mode enabled
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting node app.js
Debug mode enabled
[nodemon] clean exit - waiting for changes before restart
```

Ln 12, Col 3 Spaces: 4 UTF-8 CRLF ⓘ JavaScript ⓘ Signed out ⓘ Go Live ⓘ Quokka ⓘ

The screenshot displays two identical instances of the Visual Studio Code (VS Code) interface, each showing a Node.js application named "my-express-app".

**File Structure:**

- Root:** my-express-app
- MY-EXPRESS-APP:**
  - node\_modules
  - public
    - index.html
  - views
    - index.ejs
  - .env
- App.js:** (Selected in both instances)
- config.js**
- example.pdf**
- package-lock.json**
- package.json**
- process.env**

**Terminal Output (Both Instances):**

```
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
1
2
3
4
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Debug mode enabled
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node app.js`
Port: 3001
DB Host: localhost
[nodemon] clean exit - waiting for changes before restart
```

**Status Bar:**

Ln 4, Col 41 | Spaces: 4 | UTF-8 | CRLF | JavaScript | Signed out | Go Live | Quokka

```
File Edit Selection View Go Run Terminal Help ← → ⌘ my-express-app
```

OPEN EDITORS

MY-EXPRESS-APP

node\_modules

public

index.html

views

index.ejs

.env

app.js

config.js

example.pdf

package-lock.json

package.json

process.env

```
app.js >
  5 // Define a route for the root URL ('/') with a GET request method
  6 app.get('/', (req, res) => {
  7   // Send a response to the client
  8   res.send('Welcome to the Home Page!');
  9 });
10
11 // Define a route for the '/about' URL with a GET request method
12 app.get('/about', (req, res) => {
13   // Send a response to the client
14   res.send('About Us');
15 });
16
17 // Define a route for the '/contact' URL with a GET request method
18 app.get('/contact', (req, res) => {
19   // Send a response to the client
20   res.send('Contact Us');
21 });
22
23 // Start server
24 app.listen(port, () => console.log(`Server is running at http://localhost:${port}`));
```

Code Reference Log

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

[nodemon] starting "node app.js"

Port: 3001

DB Host: localhost

[nodemon] clean exit - waiting for changes before restart

[nodemon] restarting due to changes...

[nodemon] starting "node app.js"

Port: 3000

DB Host: localhost

[nodemon] clean exit - waiting for changes before restart

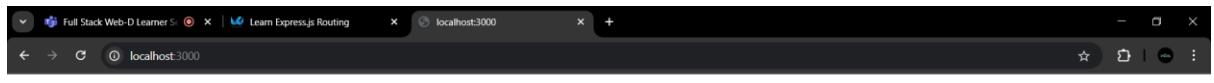
[nodemon] restarting due to changes...

[nodemon] starting "node app.js"

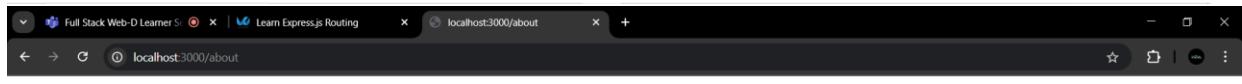
Server is running at http://localhost:3000

[nodemon] clean exit - waiting for changes before restart

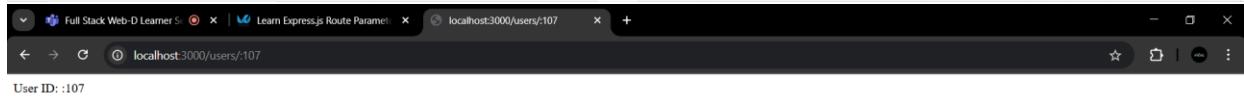
Ln 24, Col 86 Spaces: 4 UTF-8 CRLF ↵ JavaScript ⌂ Signed out ⌂ Go Live ⌂ Quokka ⌂



Welcome to the Home Page!



About Us



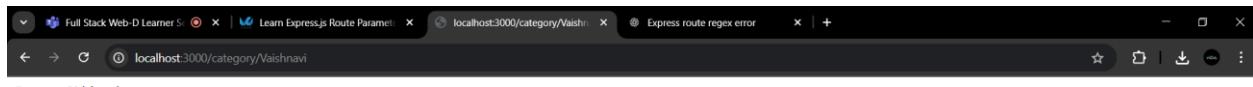
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure for "MY-EXPRESS-APP". Files include app.js, config.js, .env, process.env, index.html, index.ejs, .env, app.js, config.js, example.pdf, package-lock.json, package.json, and process.env.
- Code Editor:** The app.js file is open, displaying code for an Express.js application. It defines a route for '/users/:userId' and starts the server on port 3000.
- Terminal:** Shows logs from nodemon indicating clean exit, restarting due to changes, and starting the node app.js. It also shows the server is running at http://localhost:3000.
- Browsers:** Two browser tabs are open: "Full Stack Web-D Learner" and "Learn Express.js Route Parameters". Both tabs show the URL localhost:3000/users/107/posts/107.

User ID: 107, Post ID: 107

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "MY-EXPRESS-APP". Files include app.js, config.js, .env, process.env, index.html, index.ejs, package-lock.json, package.json, and example.pdf.
- Code Editor:** The main editor window displays the content of app.js. The code defines a route for '/users/:userId/posts/:postId' and starts the server on port 3000.
- Terminal:** The terminal shows the output of the Node.js application, indicating it crashed due to file changes and is restarting.
- Status Bar:** Shows the current file is app.js, line 5, column 38, with 38 spaces, using UTF-8 encoding, and CRLF line endings. It also indicates JavaScript is the active language and the user is signed out.



**my-express-app**

```

File Edit Selection View Go Run Terminal Help ⌘ ⌘ my-express-app
OPEN EDITORS JS app.js JS config.js .env process.env
  ✓ app.js > ...
  1 const express = require('express');
  2 const app = express();
  3
  4 app.get('/category/:categoryName', (req, res) => {
  5   const { categoryName } = req.params;
  6
  7   // Regex validation here instead
  8   if (/^a-zA-Z+$/i.test(categoryName)) {
  9     res.send(`Category: ${categoryName}`);
 10   } else {
 11     res.status(400).send('Invalid category name. Only letters are allowed.');
 12   }
 13 });
 14
 15 app.listen(3000, () => {
 16   console.log('The server is running on port: 3000');
 17 });
 18
 19

```

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```

at E:\my-express-app\node_modules\path-to-regexp\dist\index.js:94:74
at Array.map (<anonymous>)
at pathToRegexp (E:\my-express-app\node_modules\path-to-regexp\dist\index.js:294:25)
at Object.match (E:\my-express-app\node_modules\path-to-regexp\dist\index.js:264:30)
at matcher (E:\my-express-app\node_modules\router\lib\layer.js:86:23)
at new Layer (E:\my-express-app\node_modules\router\lib\layer.js:93:62)
at Function.route (E:\my-express-app\node_modules\router\index.js:428:17)

Node.js v22.16.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
The server is running on port: 3000

```

OUTLINE TIMELINE

Launchpad 0 △ 0 Live Share RHDA analysis has failed

File Edit Selection View Go Run Terminal Help ⌘ ⌘ my-express-app
OPEN EDITORS JS app.js JS config.js .env process.env
 ✓ app.js > ...
 1 // app.js
 2 const express = require('express');
 3 const app = express();
 4
 5 app.get('/', (req, res) => {
 6 res.send('Welcome to the homepage!');
 7 });
 8
 9 app.get('/users', (req, res) => {
 10 // Simulate fetching user data
 11 const users = [{ id: 1, name: 'Alex' }, { id: 2, name: 'Jane' }];
 12 res.json(users);
 13 });
 14
 15 app.listen(3000, () => {
 16 console.log('Server is running on port 3000');
 17 });
 18

PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```

at matcher (E:\my-express-app\node_modules\router\lib\layer.js:86:23)
at new Layer (E:\my-express-app\node_modules\router\lib\layer.js:93:62)
at Function.route (E:\my-express-app\node_modules\router\index.js:428:17)

Node.js v22.16.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
The server is running on port: 3000
[nodemon] restarting due to changes...
[nodemon] starting 'node app.js'
Server is running on port 3000
[nodemon] clean exit - waiting for changes before restart

```

OUTLINE TIMELINE

Launchpad 0 △ 0 Live Share RHDA analysis has failed



```
[{"id": 1, "name": "Alex"}, {"id": 2, "name": "Jane"}]
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like app.js, config.js, .env, process.env, index.html, index.ejs, and package.json.
- Code Editor:** Displays the content of app.js, which is an Express.js application. It includes routes for home, user creation, and user retrieval, along with middleware for JSON parsing and memory storage.
- Terminal:** Shows the output of nodemon starting the server on port 3000 and a clean exit message.
- Status Bar:** Shows the current file is app.js, with 33 lines of code, and other details like node version and Quokka status.

The screenshot shows the Hoppscotch API client interface with the following details:

- Request:** A POST request to `http://localhost:3000/users` is being prepared.
- Body:** The raw JSON body is set to `{"name": "Vaishnavi"}`.
- Response:** The response shows a successful creation with status 201, time 13 ms, and size 265 B. The response body is `New user Vaishnavi added successfully.`.
- Workspace:** The right side shows a workspace with collections, imports, and a help section.

A screenshot of a web browser window. The address bar shows 'localhost:3000/user'. The page content is a JSON object: { "name": "Vaishnavi" }.

```
{ "name": "Vaishnavi" }
```

A screenshot of the Visual Studio Code interface. The title bar says 'my-express-app'. The Explorer sidebar shows a project structure with files like app.js, config.js, .env, and index.html. The main editor tab shows the content of app.js:

```
9 // Home route
10 app.get('/', (req, res) => {
11   res.send('Welcome to the homepage!');
12 });
13
14 // PUT /users/:id - Update and store user data
15 app.put('/users/:id', (req, res) => {
16   const userId = req.params.id;
17   updatedData = req.body; // Store the updated data
18
19   res.send(`User record with ID: ${userId} updated successfully.`);
20 });
21
22 // GET /user - Display stored user data
23 app.get('/user', (req, res) => {
24   if (updatedData) {
25     res.status(200).json(updatedData); // Send back the stored JSON
26   } else {
27     res.status(404).send('No user data available.');
28   }
29 }
30 );
31
32 // Start the server
33 app.listen(3000, () => {
34   console.log('Server is running on port 3000');
35 });
36 |
```

The bottom status bar shows node, node 36, Col 1, Spaces: 4, UTF-8, CRLF, JavaScript, Signed out, Go Live, Quokka, and Quokka analysis has failed.

The screenshot shows the Hopscotch API testing tool interface. A DELETE request is made to `http://localhost:3000/users/107`. The request body is set to `application/json` and contains the JSON payload `{"name": "Vaishnavi"}`. The response status is `200 OK`, time taken is `65 ms`, and the size is `274 B`. The response body shows the message `User record with ID: 107 deleted successfully.`

The screenshot shows a browser window with the URL `localhost:3000/user`. The page displays the message `No user data available.`

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure for "MY-EXPRESS-APP".
- Editor:** Displays the file `app.js` containing the following code:

```
// Middleware to parse JSON bodies
app.use(express.json());

// store the latest user data (in memory)
let updatedData = null;

// Home route
app.get('/', (req, res) => {
  res.send('Welcome to the homepage!');
});

// PUT /users/:id - Update and store user data
app.delete('/users/:id', (req, res) => {
  const userId = req.params.id;
  // simulate deleting user data
  res.send(`User record with ID: ${userId} deleted successfully.`);
});

// GET /user - Display stored user data
app.get('/user', (req, res) => {
  if (updatedData) {
    res.status(200).json(updatedData); // Send back the stored JSON
  } else {
    res.status(404).send('No user data available.');
  }
});

// Start the server
app.listen(3000, () => {
  console.log('Server is running on port 3000');
});

[nodemon] restarting due to changes...
[nodemon] starting "node app.js"
Server is running on port 3000
```

- Terminal:** Shows the command `node` and the output of nodemon starting the server.
- Status Bar:** Shows the line number (Ln 23, Col 1), spaces (Spaces: 4), encoding (UTF-8 CRLF), file type (JavaScript), and signed out status.

The screenshot shows the Hoppscotch API testing tool interface with the following details:

- Request:** A POST request to `http://localhost:3000/user`.
- Body:** Contains the following JSON payload:

```
{
  "name": "Vaishnavi"
}
```

- Response:** Status: 200 • OK, Time: 7 ms, Size: 244 B.
- Response Body:** New user created.
- Right Panel:** Personal Workspace, Collections, and Import options.

The screenshot shows the Hoppscotch interface. A PUT request is made to `http://localhost:3000/user`. The request body is a JSON object with a single key-value pair: `"name": "Vaishnavi"`. The response status is 200 OK, time taken is 6 ms, and size is 244 B. The response body contains the message `User data updated`.

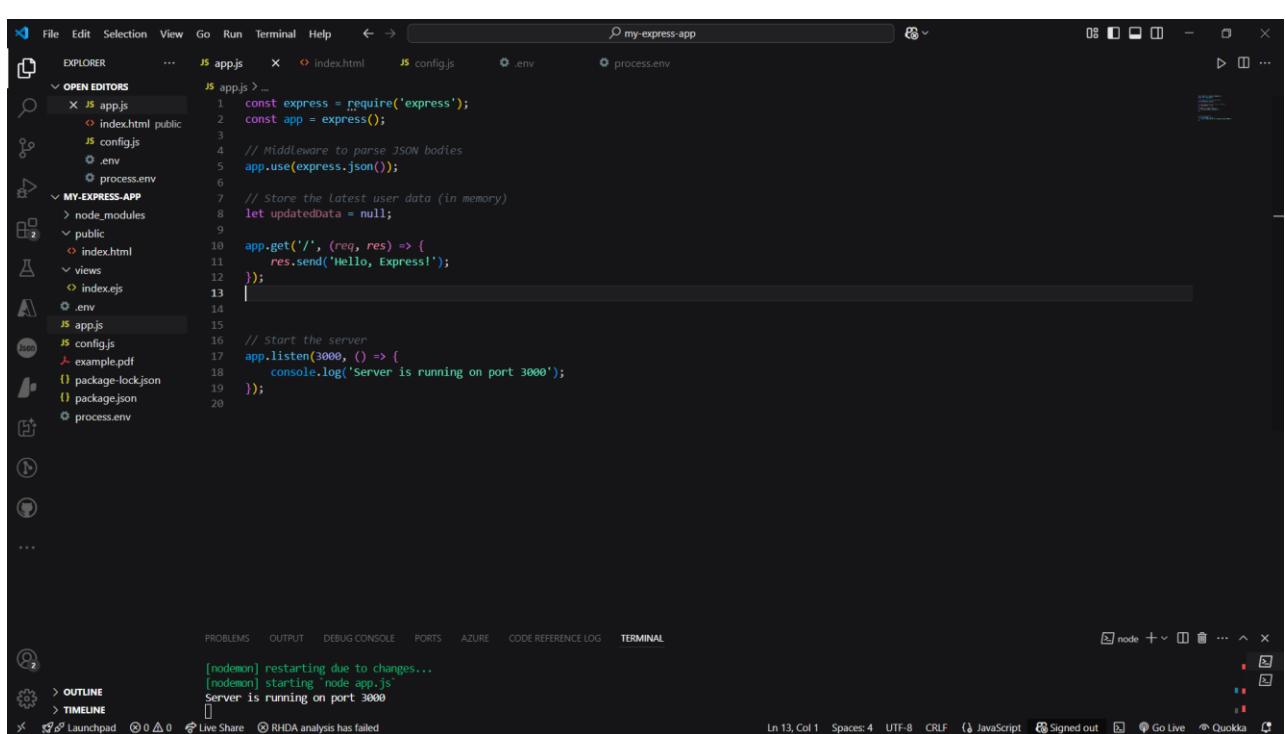
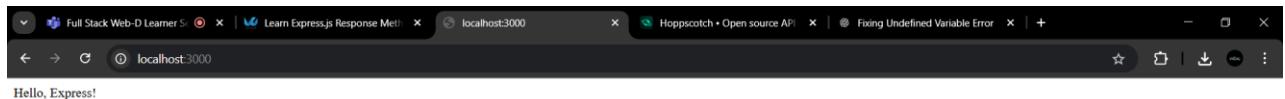
The screenshot shows the VS Code interface with the project `my-express-app` open. The `app.js` file is the active editor, displaying the following code:

```
app.use(express.json());
let updatedData = null;

app.route('/user')
    .get((req, res) => {
        // Fetch user data
        res.send('fetching user data');
    })
    .post((req, res) => {
        // Add a new user
        res.send('New user created');
    })
    .put((req, res) => {
        // Update user data
        res.send('User data updated');
    });

// Start the server
app.listen(3000, () => {
    console.log('Server is running on port 3000');
});
```

The terminal at the bottom shows the output of running the application with `node app.js`, indicating it's listening on port 3000.



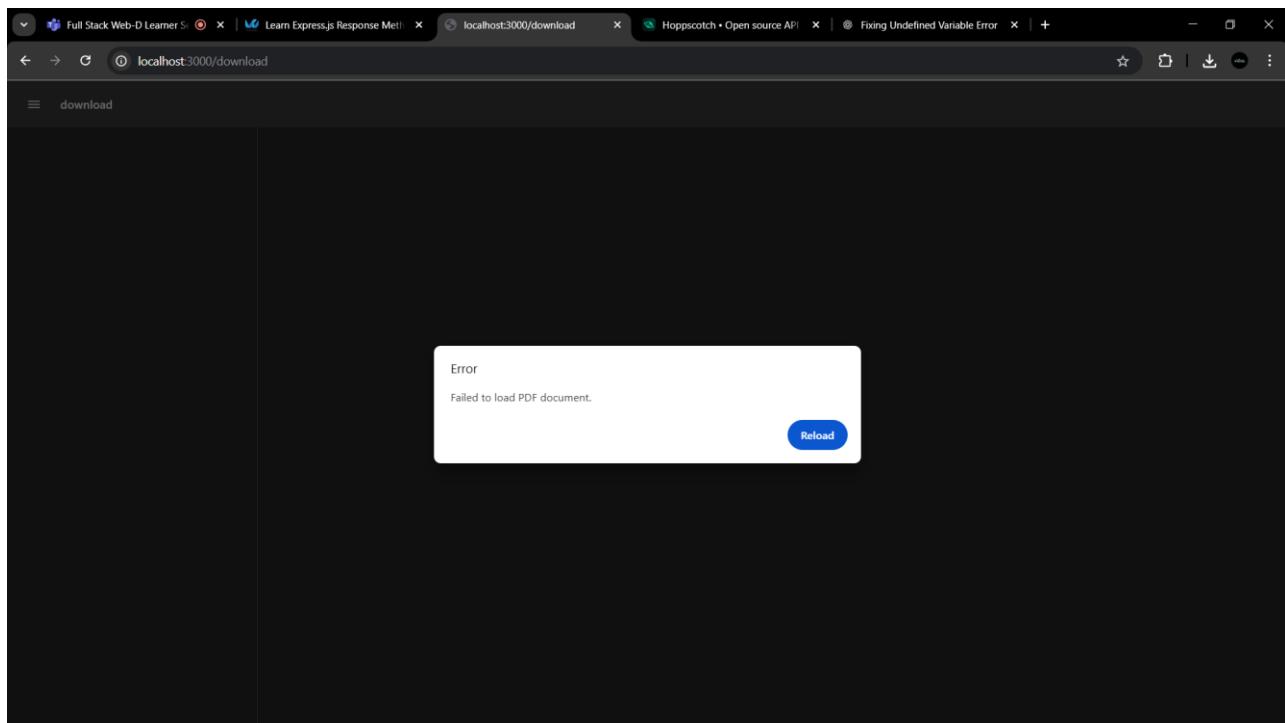
A screenshot of a web browser window showing the JSON response at `localhost:3000/user`. The response is:

```
{  
  "name": "Alex",  
  "age": 30  
}
```

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar shows the project structure for a Node.js application named "my-express-app". The "EXPLORER" view displays files like `app.js`, `index.html`, `config.js`, `.env`, and `process.env`. The "TERMINAL" view at the bottom shows the output of running the application with nodemon:

```
[nodemon] starting "node app.js"  
[nodemon] Server is running on port 3000  
[nodemon] clean exit - waiting for changes before restart
```

The status bar at the bottom indicates the code is in JavaScript mode, has 20 lines, 4 spaces, and is using CRLF line endings.

A screenshot of a code editor interface for a Node.js project named "my-express-app".  
**File Structure:**

- OPEN EDITORS:
  - app.js
  - index.html
  - config.js
  - .env
  - process.env
- MY-EXP...:
  - files
  - node\_modules
  - public
    - index.html
  - views
    - index.ejs
  - .env
  - app.js
  - config.js
  - package-lock.json
  - package.json
  - process.env

```
JS app.js > ...
1 const express = require('express');
2 const app = express();
3
4 // Middleware to parse JSON bodies
5 app.use(express.json());
6
7 const path = require('path');
8
9 app.get('/download', (req, res) => {
10   const filePath = path.join(__dirname, 'files', 'example.pdf');
11   res.sendFile(filePath);
12 });
13
14
15
16
17 // Start the server
18 app.listen(3000, () => {
19   console.log('Server is running on port 3000');
20 });
21
22
```

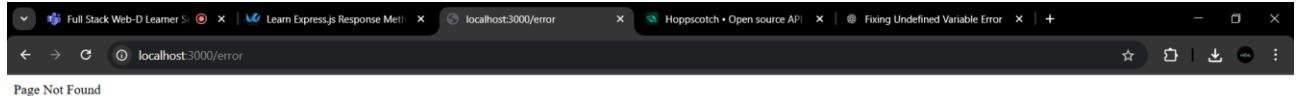
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL

```
[nodemon] starting `node app.js`
Server is running on port 3000
Error: ENOENT: no such file or directory, stat 'E:\my-express-app\files\example.pdf'
```

Ln 13, Col 1 Spaces:4 UTF-8 CRLF ↴ JavaScript ↴ Signed out ↴ Go Live ↴ Quokka ↴

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like `app.js`, `index.html`, `config.js`, `.env`, and `process.env`.
- Code Editor:** Displays the `app.js` file content:const express = require('express');
const app = express();
// Middleware to parse JSON bodies
app.use(express.json());
const path = require('path');
app.get('/download', (req, res) => {
 const filePath = path.join(\_\_dirname, 'files', 'example.pdf');
 res.sendFile(filePath);
});
// Start the server
app.listen(3000, () => {
 console.log('Server is running on port 3000');
});
- Terminal:** Shows the output of the node command:[nodemon] starting `node app.js`
Server is running on port 3000
Error: ENOENT: no such file or directory, stat 'E:\my-express-app\files\example.pdf'



The screenshot shows the VS Code interface with the title bar "my-express-app". The Explorer sidebar on the left lists files: app.js, index.html, config.js, .env, process.env, MY-EXPRESS-APP (containing files, example.pdf, node\_modules, public, views, index.ejs, .env, package-lock.json, package.json, process.env), and a folder icon. The terminal tab is active, showing the command line:

```
[nodemon] starting `node app.js`
Server is running on port 3000
[nodemon] clean exit - waiting for changes before restart
```

The status bar at the bottom indicates "Ln 12, Col 1" and "Spaces: 4 - UTF-8 CRLF".

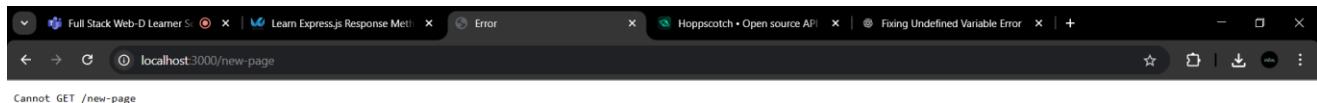
The screenshot shows the VS Code interface with the title bar "my-express-app". The Explorer sidebar is identical to the previous screenshot. The terminal tab is active, showing the command line:

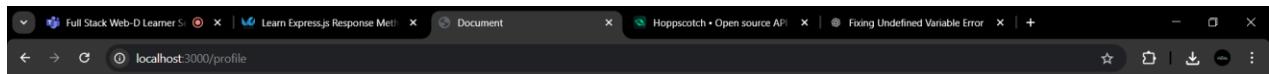
```
[nodemon] starting `node app.js`
Server is running on port 3000
[nodemon] clean exit - waiting for changes before restart
```

The code editor shows the app.js file with the following content:

```
const express = require('express');
const app = express();
// Middleware to parse JSON bodies
app.use(express.json());
const path = require('path');
app.get('/old-page', (req, res) => {
  res.redirect('/new-page');
});
// start the server
app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

The status bar at the bottom indicates "Ln 12, Col 1" and "Spaces: 4 - UTF-8 CRLF".





Hello

The screenshot shows a code editor interface with the title bar 'my-express-app'. The left sidebar (EXPLORER) shows a file tree for a 'MY-EXPRESS-APP' project. The 'profile.ejs' file is selected in the tree and is also open in one of the editors at the top. The code editor shows the following content for profile.ejs:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
  </head>
  <body>
    <h1>Hello</h1>
  </body>
</html>
```

The bottom status bar shows the terminal output: '[nodemon] starting 'node app.js'', 'Server is running on port 3000', and '[nodemon] clean exit - waiting for changes before restart'.

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows the project structure under "MY-EXPRESS-APP".
- Open Editors:** Displays the content of `app.js` and `index.ejs`.
- Terminal:** Shows the command-line output from running the application.
- Bottom Bar:** Includes tabs for "Full Stack Web-D Learner", "Learn Express.js Response Met...", "localhost:3000/setheader", "Hoppscotch • Open source API", and "Fixing Undefined Variable Error".

```
const express = require('express');
const app = express();
const path = require('path');
app.set('view engine', 'ejs');

app.get('/profile', (req, res) => {
  const user = { name: 'Anjali', age: 25 };
  res.render('profile', { user });
});

// Start the server
app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

[nodemon] starting `node app.js`  
[nodemon] Server is running on port 3000  
[nodemon] clean exit - waiting for changes before restart

## Header Set

The screenshot shows the VS Code interface with the title bar "my-express-app". The Explorer sidebar on the left lists files like app.js, index.ejs, profile.ejs, index.html, config.js, .env, and process.env. The app.js editor tab is active, displaying the following code:

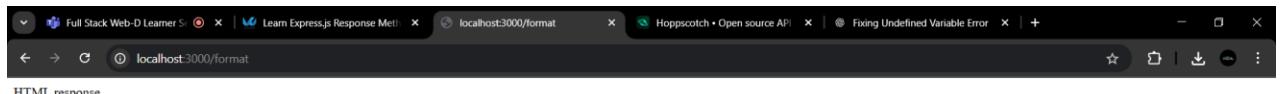
```
const express = require('express');
const app = express();
// Middleware to parse JSON bodies
app.use(express.json());
const path = require('path');
app.get('/setheader', (req, res) => {
  res.set('Content-Type', 'text/html');
  res.send(<h1>Header Set</h1>');
});
// Start the server
app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

The terminal at the bottom shows the output of nodemon starting the application.

The screenshot shows the VS Code interface with the title bar "my-express-app". The Explorer sidebar on the left lists files like app.js, index.ejs, profile.ejs, index.html, config.js, .env, and process.env. The app.js editor tab is active, displaying the following code:

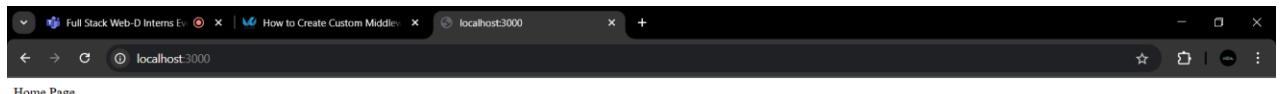
```
const express = require('express');
const app = express();
const path = require('path');
app.get('/end', (req, res) => {
  res.status(204).end();
  console.log("Its end");
});
// Start the server
app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

The terminal at the bottom shows the output of the application responding with "Its end" for the /end endpoint.



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure under "MY-EXPRESS-APP".
  - Files: example.pdf, node\_modules, public, index.html.
  - Views: index.ejs, profile.ejs.
  - Environment: .env.
- Code Editor (Center):** Displays the content of `app.js`. The code defines an Express app, sets up middleware for JSON bodies, and handles different response formats (text/plain, text/html, application/json) based on the requested format.
- Terminal (Bottom):** Shows the output of running the application. It includes the Node.js version, the command used to start the server (node app.js), and the port it's running on (3000). It also indicates that nodemon is restarting due to changes and that the server is running on port 3000.



The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure under "MY-EXPRESS-APP".
- Editor:** Displays the contents of `app.js` with syntax highlighting for JavaScript.
- Terminal:** Shows the output of the Node.js application running on port 3000, including logs from nodemon and server requests.
- Bottom Status Bar:** Includes icons for file operations, a search bar, and system status like temperature and battery level.

```
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node app.js`
Server running on port 3000
GET request to / at 4:52:16 pm
GET request to /Favicon.ico at 4:52:16 pm
GET request to /about at 4:52:25 pm
```

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows the project structure under "MY-EXPRESS-APP".
- Editor:** Displays the contents of `app.js` with syntax highlighting for JavaScript.
- Terminal:** Shows the output of the Node.js application running on port 3000, including logs from nodemon and server requests.
- Bottom Status Bar:** Includes icons for file operations, a search bar, and system status like temperature and battery level.

```
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node app.js`
Server running on port 3000
GET request to / at 4:52:16 pm
GET request to /Favicon.ico at 4:52:16 pm
GET request to /about at 4:52:25 pm
```