

Practical: Node.js application on EC2

Table of Contents

- [Step 1: Install Node.js](#)
- [Step 2: Create a small Node.js server](#)
- [Step 3: Run the server](#)
- [Step 4: Terminate the instance](#)

This activity involves installing NodeJS on an EC2 instance. The goal is to deploy an application that responds to HTTP requests.


We recommend connecting to your EC2 instance over ssh. It is possible to complete it by connecting with the Session Manager, but the experience has generally been smoother for students using ssh.

Step 1: Install Node.js

It is critical for you to learn to navigate and interpret the AWS documentation. The AWS cloud ecosystem is enormous. The best way to learn about services and how they interact with each other is by reading the documentation and getting hands-on.

For step 1, you will follow a tutorial provided by AWS in their Documentation.

Follow from the section labelled "Procedure". Stop when you reach "Creating an Amazon Machine Image".

The tutorial can be found here: [Tutorial: Setting Up Node.js on an Amazon EC2 Instance](https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/setting-up-node-on-ec2-instance.html) 
(<https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/setting-up-node-on-ec2-instance.html>)

Note: If you are connected using Session Manager then you need to make sure that you are the `ubuntu` user; your prompt should start with `ubuntu@`. If not, then you need to run `sudo su -l ubuntu` to switch to the ubuntu user. If not then you will likely see something like this:

```
ssm-user@ip-172-31-95-120:/usr$ source ~/.bashrc
ssm-user@ip-172-31-95-120:/usr$ nvm
Command 'nvm' not found, but there are 14 similar ones.
ssm-user@ip-172-31-95-120:/usr$
```

If you encounter this error, run `sudo su -l ubuntu` and then start again.

Step 2: Create a small Node.js server

For handling HTTP requests on port 3000 using Node.js, you can use the built-in `http` module. Here's a simple example of a Node.js server that listens on port 3000 and responds to requests:

1. Create a new directory and initialise a Node.js project:

```
mkdir my-node-server
cd my-node-server
npm init -y
```

2. Create a file named `server.js` and add the code:

```
nano server.js
```

copy/paste in the following code

```
const http = require('http');

const hostname = '0.0.0.0'; // to accept connections from outside localhost
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello, world!\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

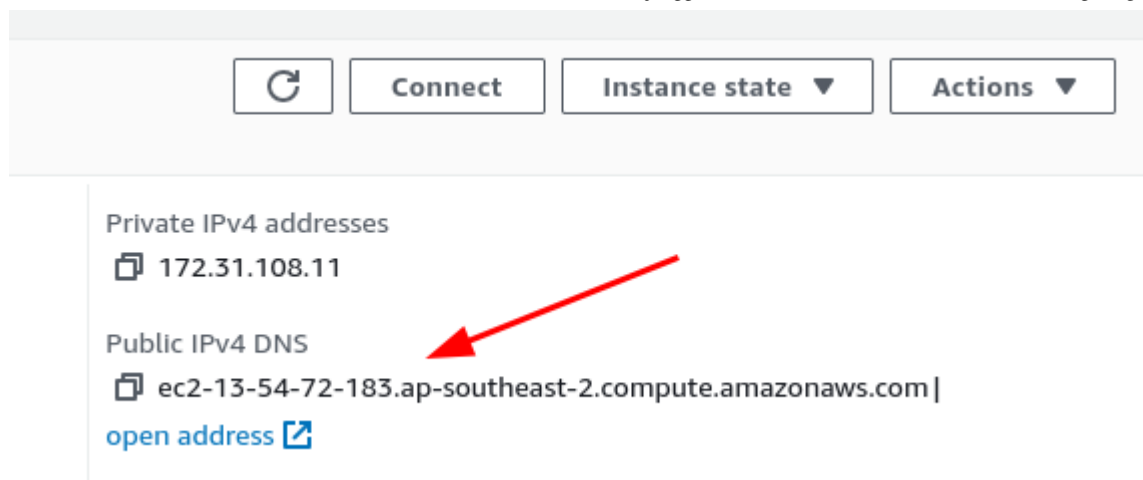
You can use `vi` or other text editors if you prefer.

Step 3: Run the server

```
node server.js
```

This will start an HTTP server that listens on port 3000. When you navigate to your server's address in a web browser, you should see the message "Hello, world!".


Use the EC2 instance details page on the AWS console to find your public DNS address. If your instance does not have a public address then you may have launched it in a private subnet.



Add `http://` to the start and the port number `:3000` to the end of the instance's public DNS name to create the complete URL, like so:

```
http://ec2-13-54-72-183.ap-southeast-2.compute.amazonaws.com:3000
```

Open your URL in your web browser to check that everything is working correctly.

You can read more about public IPv4 addresses in EC2 in the AWS documentation here: [Amazon EC2 instance IP addressing: Public IPv4 addresses](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-instance-addressing.html#concepts-public-addresses) 
(<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-instance-addressing.html#concepts-public-addresses>)

Common Problems

If you can't connect to the web server, check your instance's settings:

- using a public subnet
- security groups are correct

and check your URL:

- DNS name is for the correct instance
- port number matches what the server is listening on (3000 in this practical)

Step 4: Terminate the instance

If you are all done, terminate the instance. If you are continuing to the next exercise then you can terminate it later.