EXPERIMENT 8

AIM: Install and deploy first application in node js.

OBJECTIVE OF THE EXPERIMENT:

- To provide knowledge on installing and deploying an application in node.js.
- To understand how the working of the process takes place.

OUTCOME OF THE EXPERIMENT:

- Install node.js and npm..
- Create a new application using node.js.
- Deploy the created application using node.js.

Step 1: Install Node.js

1. **Download Node.js**: Go to the Node.js website and download the LTS version for your operating system.
2. **Install Node.js**: Run the installer and follow the instructions. This will also install npm (Node Package Manager).

Verify Installation: Open your terminal (Command Prompt, PowerShell, or terminal on macOS/Linux) and run:

```
node -v
npm -v
```

You should see the version numbers for Node.js and npm.

Step 2: Create a Simple Node.js Application

Create a Project Directory: Navigate to your desired folder in the terminal and create a new directory for your app.

```
mkdir my-node-app
cd my-node-app
```

Initialize the Project: Create a package.json file. This file manages the app's dependencies and metadata.

```
npm init -y
```
Create the Application File: Create an index.js file.

Write Basic Code: Open index.js and add the following code:
```
const http = require('http');
const hostname = '127.0.0.1';
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}/`);
});
```

Step 3: Run the Application

Start the Server: In your terminal, run the following command:

```
node index.js
```

Access the Application: Open your web browser and go to http://127.0.0.1:3000/. You should see "Hello World".

Step 4: Deploying Locally

To run your application in the background, you can use tools like pm2 or nodemon. Here's how to use pm2:

Install pm2:

```
npm install -g pm2
```

Start your application with pm2:
```
pm2 start index.js
```

Add 4 more applications in pm2 :

Step 1: Create Sample Applications

Create a Project Directory (if you haven't already):

```
mkdir my-pm2-apps
cd my-pm2-apps
```

Create Application Files: Create four simple Node.js application files: app1.js app2.js app3.js and app4.js

Add Sample Code: Open each file in a text editor and add the following code:

app1.js:

```
const http = require('http');
const port = 3001;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello from App 1\n');
});
server.listen(port, () => {
  console.log(`App 1 running at http://localhost:${port}/`);
});
```

app2.js:

```
const http = require('http');
const port = 3002;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello from App 2\n');
});
server.listen(port, () => {
  console.log(`App 2 running at http://localhost:${port}/`);
});
```

app3.js:

```
const http = require('http');
const port = 3003;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello from App 3\n');
});
```

```
server.listen(port, () => {
  console.log(`App 3 running at http://localhost:${port}/`);
});
```

app4.js:

```
const http = require('http');
const port = 3004;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello from App 4\n');
});
server.listen(port, () => {
  console.log(`App 4 running at http://localhost:${port}/`);
});
```

Step 2: Start Applications with PM2

Now you can start these applications using pm2. In your terminal, run the following commands:

```
pm2 start app1.js --name app1
pm2 start app2.js --name app2
pm2 start app3.js --name app3
pm2 start app4.js --name app4
```

Step 3: Verify Applications in PM2

After starting the applications, you can check that they are running:

```
pm2 list
```

You should see a list of all four applications with their respective IDs, names, and statuses.

Step 4: Access the Applications

You can access each application in your web browser:

- App 1: http://localhost:3001
- App 2: http://localhost:3002
- App 3: http://localhost:3003
- App 4: http://localhost:3004

Checking PM2 Status

List All Processes: This command shows all the processes managed by pm2.
pm2 list

View Process Details: To get more details about a specific process (replace <app_id> with the actual ID or name):
pm2 show <app_id>

View Logs: This command will display logs for a specific application. You can use the app ID or name.
pm2 logs <app_id>

View All Logs: To see logs from all applications managed by pm2:
pm2 logs

Managing Processes

Stop a Process: To stop a specific application:
pm2 stop <app_id>

Restart a Process: To restart a specific application:
pm2 restart <app_id>

Delete a Process: To remove a specific application from pm2:
pm2 delete <app_id>

Restart All Processes: To restart all applications managed by pm2:
pm2 restart all

Other Useful Commands

Monitor Resource Usage: To monitor CPU and memory usage of your applications:
pm2 monit

Save Process List: To save the current process list for automatic restart on server reboot:
pm2 save

Startup Script Generation: To generate a startup script to run pm2 on server reboot:
pm2 startup

Delete All Processes: To stop and delete all applications managed by pm2:
pm2 delete all

1. Using pm2 list

Run this command in your terminal:

pm2 list
This will display a table with information about all your running applications. The first column will show the **ID** of each application, along with the name, status, and other details.

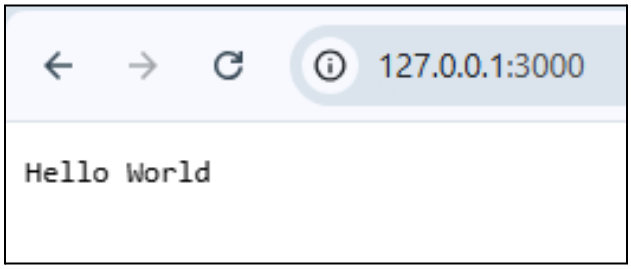
2. Using pm2 show

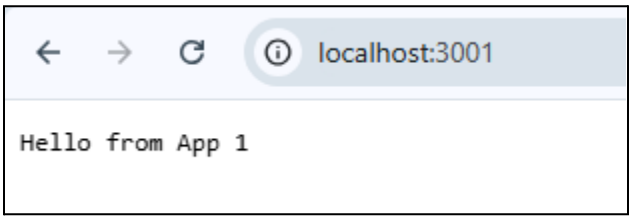
If you want to get details about a specific application by name or ID, you can use:

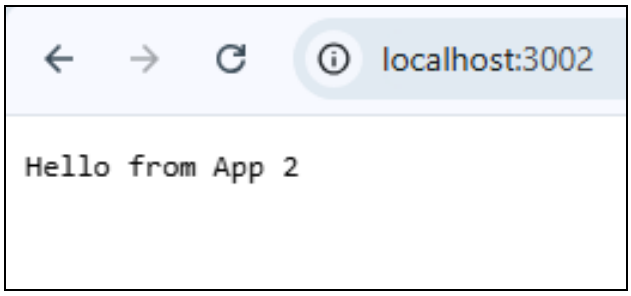
pm2 show <app_name>
or

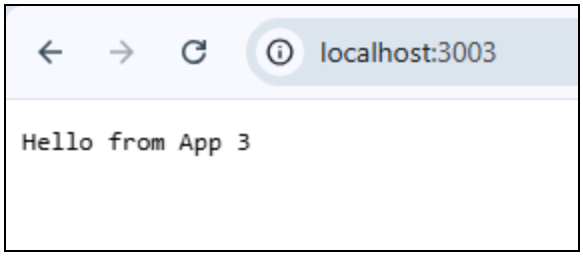
pm2 show <app_id>

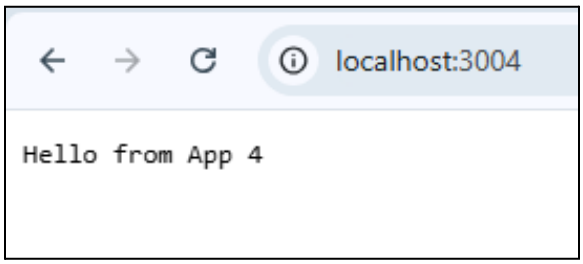
OUTPUTS:












```
PS C:\Users\CBIT\Desktop\Ali AIML\Experiment 8\my-node-app\my-pm2-apps> pm2 list
```

id	name	namespace	version	mode	pid	uptime	↺	status	cpu	mem	user	watching
1	app1	default	1.0.0	fork	27720	8m	0	online	0%	41.7mb	CBIT	disabled
3	app1	default	1.0.0	fork	0	0	15	errored	0%	0b	CBIT	disabled
2	app2	default	1.0.0	fork	0	0	15	errored	0%	0b	CBIT	disabled
4	app2	default	1.0.0	fork	10340	6m	0	online	0%	41.2mb	CBIT	disabled
5	app3	default	1.0.0	fork	10000	6m	0	online	0%	41.4mb	CBIT	disabled
6	app4	default	1.0.0	fork	5336	5m	0	online	0%	41.2mb	CBIT	disabled
0	index	default	1.0.0	fork	0	0	15	errored	0%	0b	CBIT	disabled