Lesson 10 Demo 01

Working with Response Methods

Objective: To work various response methods in Express.js for effective handling of data and requests

Tools Required: Ubuntu and Visual Studio

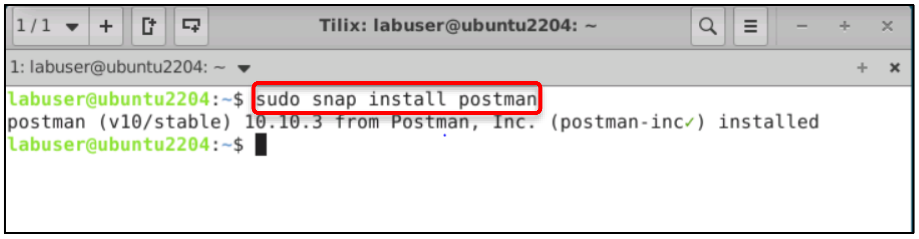
Prerequisites: Knowledge of JavaScript and NodeJs

Steps to be followed:

1. Install Postman on the system for verification purposes
2. Use express.json() response method in Express.js
3. Use express.raw() response method in Express.js
4. Use express.Router() response method in Express.js
5. Use express.static() response method in Express.js
6. Use express.text() response method in Express.js
7. Use express.urlencoded() response method in Express.js

Step 1: Install Postman on the system for verification purposes

1.1 Verify the response method by using the Postman application, and execute the following command in the system terminal to install Postman:
sudo snap install postman

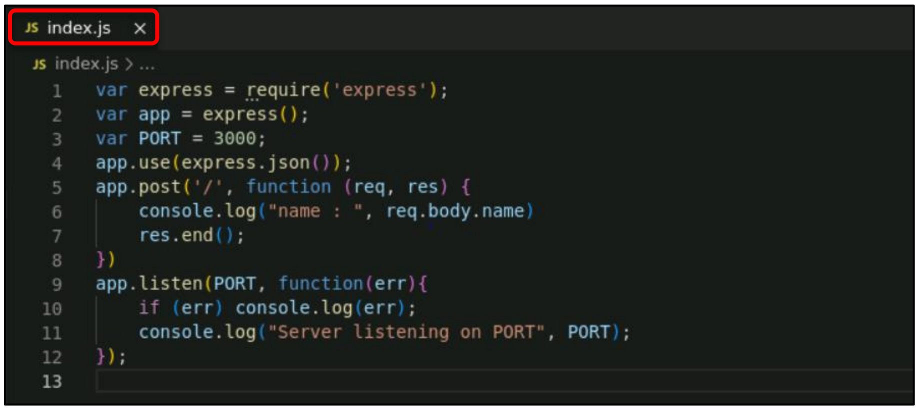


```
labuser@ubuntu2204:~$ sudo snap install postman
postman (v10/stable) 10.10.3 from Postman, Inc. (postman-inc✓) installed
labuser@ubuntu2204:~$
```

Step 2: Use express.json() response method in Express.js

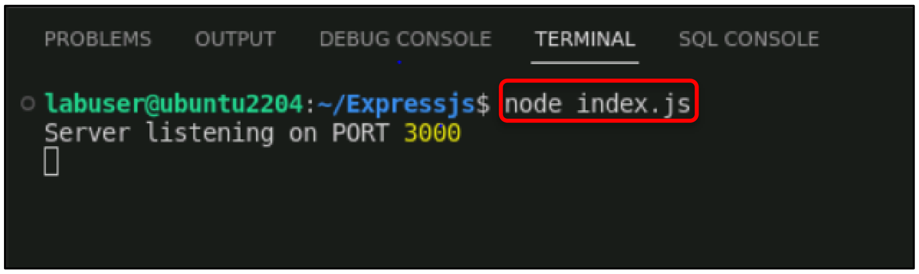
2.1 Open the created Expressjs folder in vs code and write the below code in the **index.js** file:

```
var express = require('express');
var app = express();
var PORT = 3000;
app.use(express.json());
app.post('/', function (req, res) {
  console.log("name : ", req.body.name)
  res.end();
})
app.listen(PORT, function(err){
   if (err) console.log(err);
   console.log("Server listening on PORT", PORT);
});
```



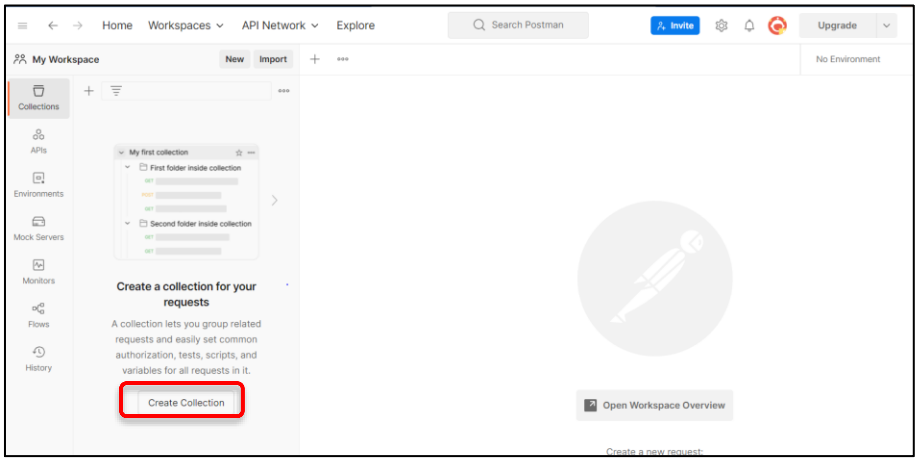
```
var express = require('express');
var app = express();
var PORT = 3000;
app.use(express.json());
app.post('/', function (req, res) {
    console.log("name : ", req.body.name)
    res.end();
})
app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```

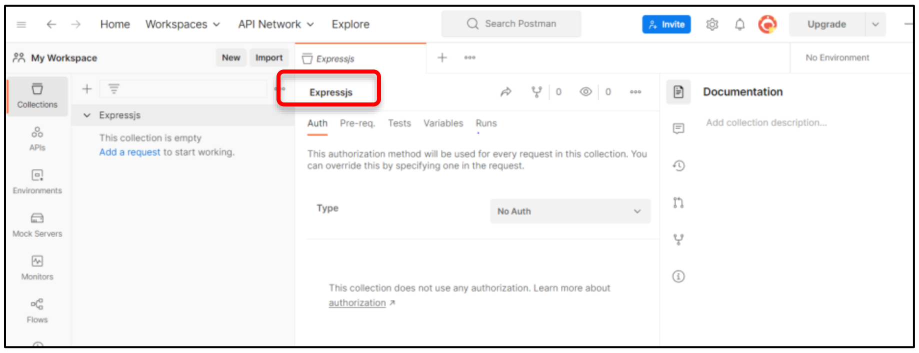
2.2 Run the **node index.js** command in the terminal



```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
```

2.3 Open Postman, either create an account or skip it for now, and once in the Postman workspace, create a collection for Express.js

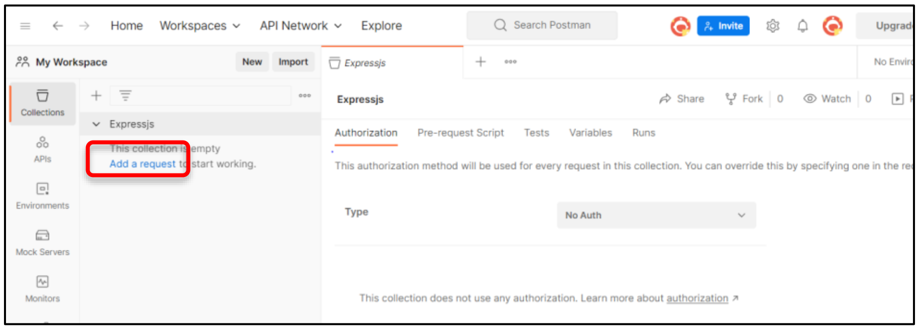


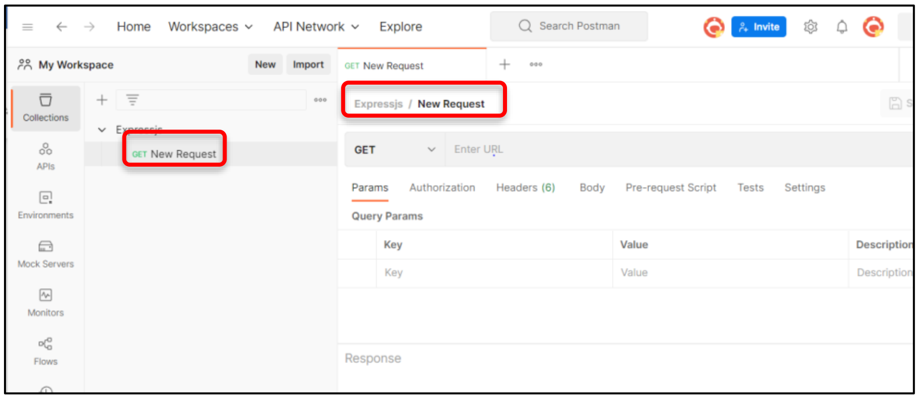


This authorization method will be used for every request in this collection. You can override this by specifying one in the request.

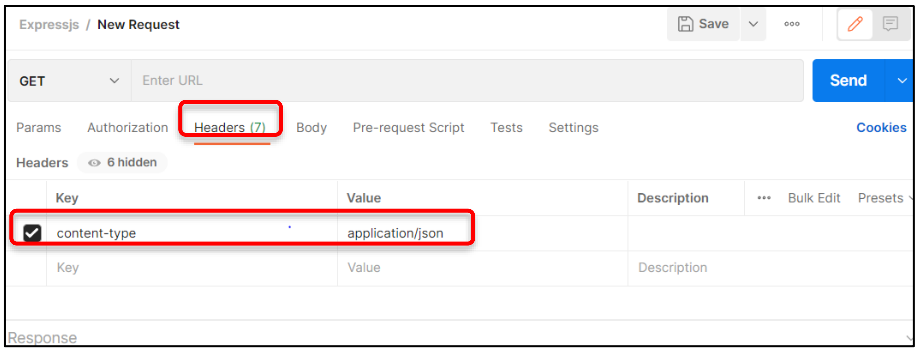
This collection does not use any authorization. Learn more about authorization

2.4 Add a new request to the collection





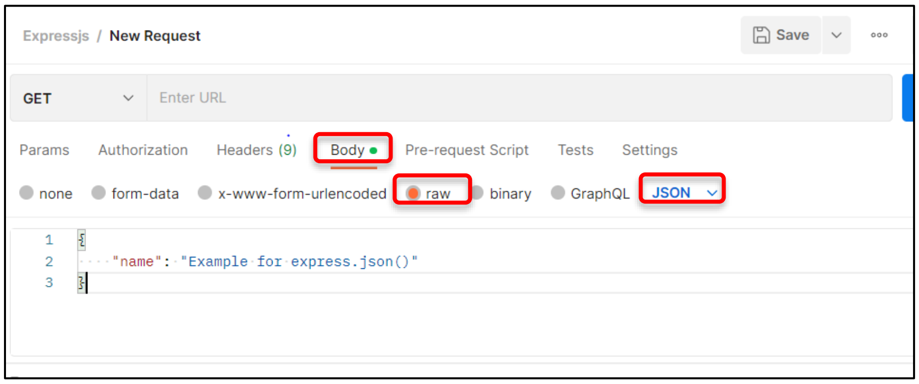
2.5 In the new request, navigate to **the Headers** option and write **key- content-type & value - application/json**



Key	Value	Description
content-type	application/json	

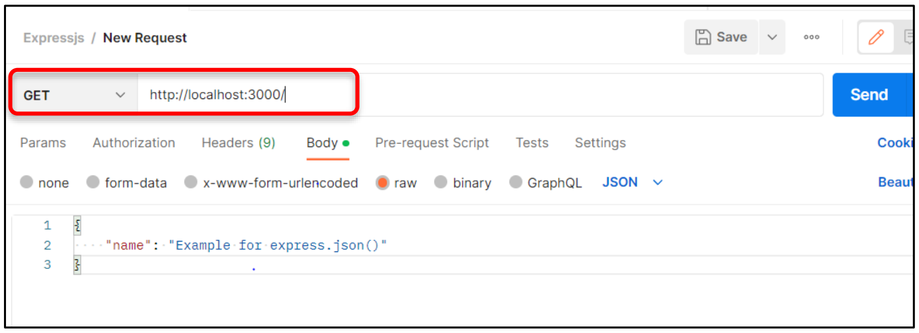
2.6 Navigate to the **Body** section, change the type to **raw**. Upon clicking **Body**, a drop-down menu will appear at the end of that option. From the drop-down menu, select **JSON** format and enter the following lines in the workspace:

```
{
  "name": "Example for express.json()"
}
```



```
{
    "name": "Example for express.json()"
}
```

2.7 Initiate a POST request to **http://localhost:3000/** and examine the output in the VSCode terminal where the program is executed



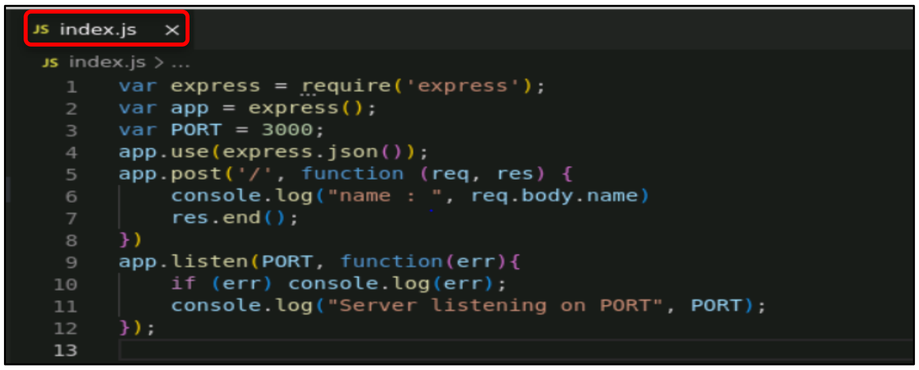
http://localhost:3000/

```
{
    "name": "Example for express.json()"
}
```

Step 3: Use express.raw() response method in Express.js

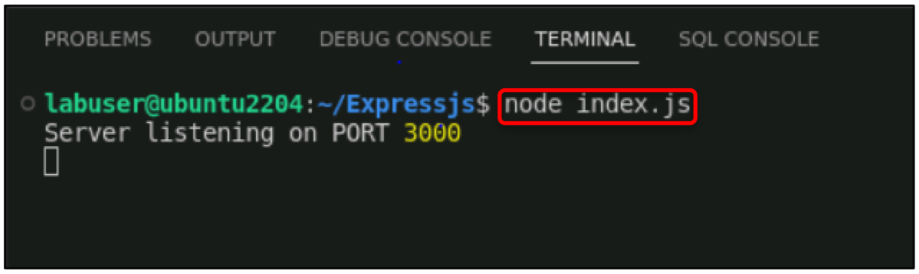
3.1 Add the following code in the **index.js** file:

```
var express = require('express');
var app = express();
var PORT = 3000;
app.use(express.json());
app.post('/', function (req, res) {
    console.log("name : ", req.body.name)
    res.end();
})
app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```



```
var express = require('express');
var app = express();
var PORT = 3000;
app.use(express.json());
app.post('/', function (req, res) {
    console.log("name : ", req.body.name)
    res.end();
})
app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```

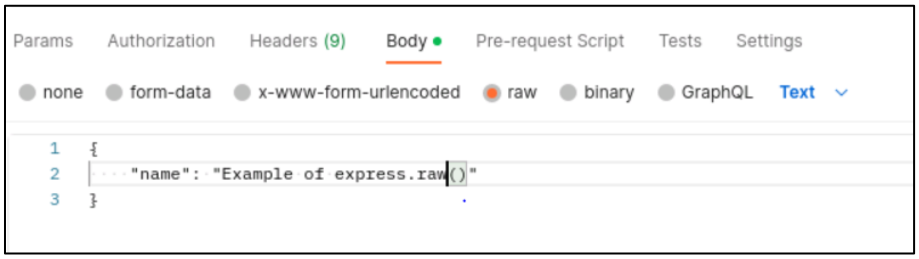
3.2 Run the **node index.js** command in the terminal



```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
```

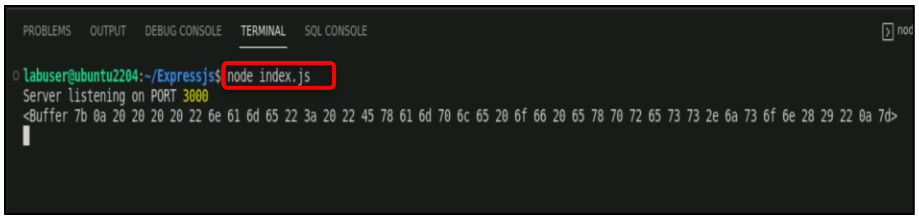
3.3 In the new request, navigate to the **Headers** option and write **key- content-type & value - application/ octet-stream.** Also, go to Body and change type to **Body**. After clicking on the Body, at the end of those options, a drop-down menu is present. Select **JSON** format from that and write the below lines in the workspace:

{
"name": "Example of express.raw()"
}



```
{
    "name": "Example of express.raw()"
}
```

3.4 Execute a POST request to **http://localhost:3000/** and verify the output in the VSCode terminal where the program is executed



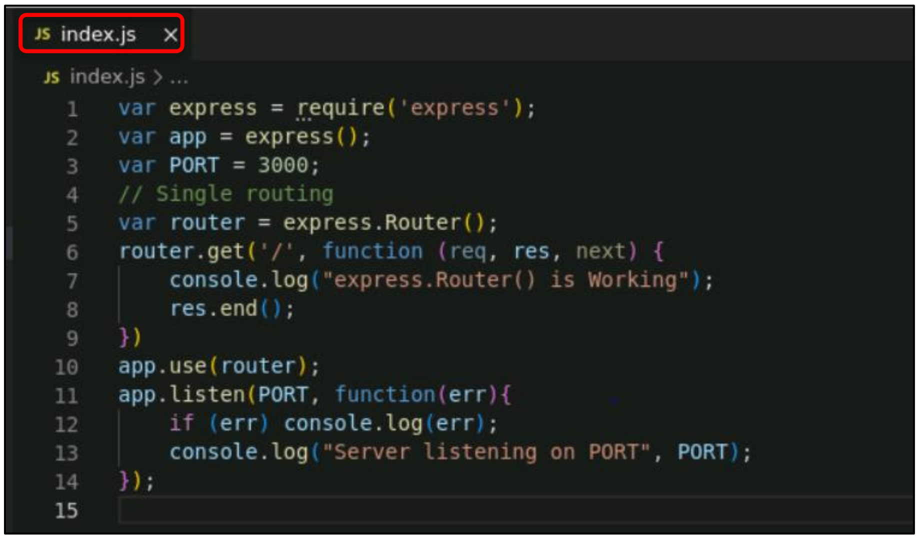
```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
<Buffer 7b 0a 20 20 20 20 22 6e 61 6d 65 22 3a 20 22 45 78 61 6d 70 6c 65 20 6f 66 20 65 78 70 72 65 73 73 2e 6a 73 6f 6e 28 29 22 0a 7d>
```

Step 4: Use express.Router() response method in Express.js

4.1 Add the following code in the **index.js** file

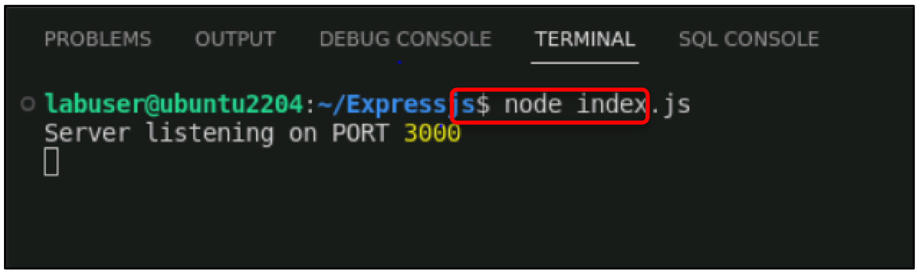
```
var express = require('express');
var app = express();
var PORT = 3000;
// Single routing
var router = express.Router();
```

```
router.get('/', function (req, res, next) {
   console.log("express.Router() is Working");
   res.end();
})
app.use(router);
app.listen(PORT, function(err){
   if (err) console.log(err);
   console.log("Server listening on PORT", PORT);
});
```



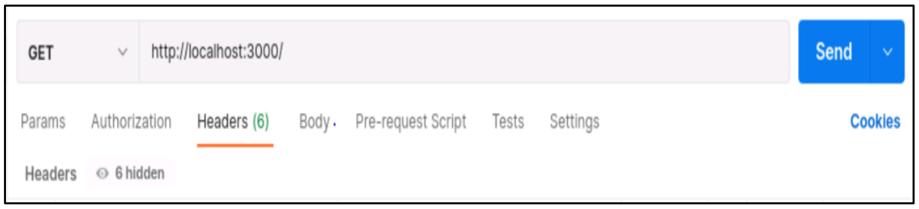
```
var express = require('express');
var app = express();
var PORT = 3000;
// Single routing
var router = express.Router();
router.get('/', function (req, res, next) {
    console.log("express.Router() is Working");
    res.end();
})
app.use(router);
app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```

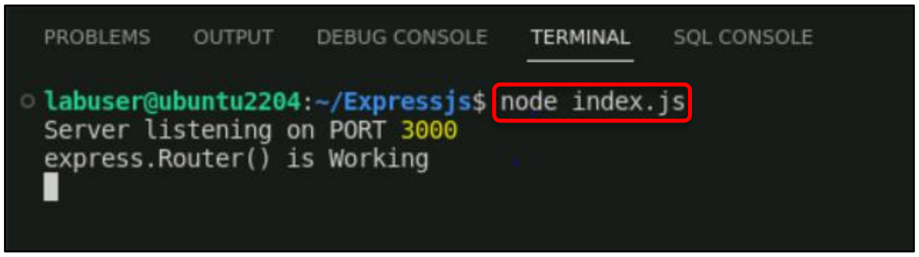
4.2 Run the **node index.js** command in the terminal



```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
```

4.3 Initiate a GET request to **http://localhost:3000/** and examine the output in the VSCode terminal where the program is executed



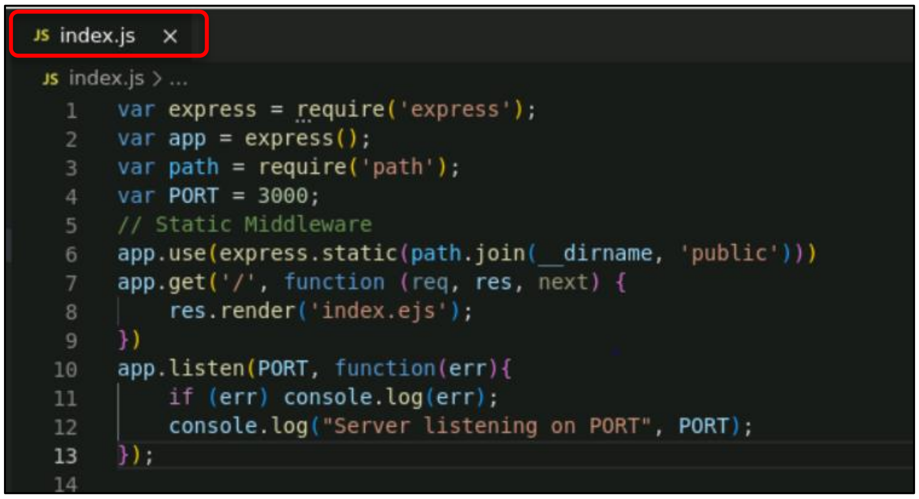


```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
express.Router() is Working
```

Step 5: Use express.static() response method in Express.js

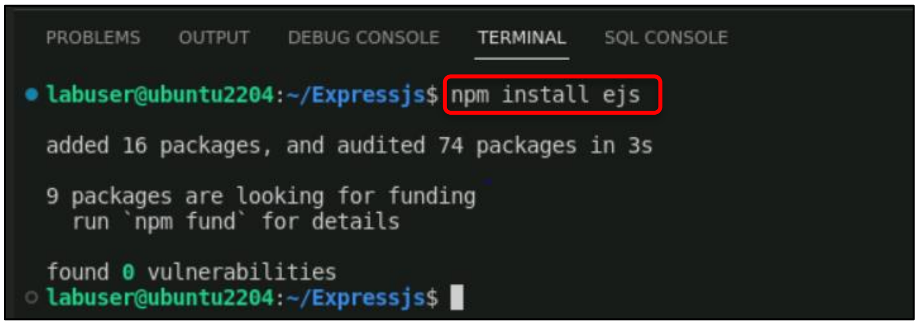
5.1 Write the following code in the **index.js** file

```
var express = require('express');
var app = express();
var path = require('path');
var PORT = 3000;
// Static Middleware
app.use(express.static(path.join(__dirname, 'public')))
app.get('/', function (req, res, next) {
   res.render('index.ejs');
})
app.listen(PORT, function(err){
   if (err) console.log(err);
   console.log("Server listening on PORT", PORT);
});
```



```
var express = require('express');
var app = express();
var path = require('path');
var PORT = 3000;
// Static Middleware
app.use(express.static(path.join(__dirname, 'public')))
app.get('/', function (req, res, next) {
    res.render('index.ejs');
})
app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```

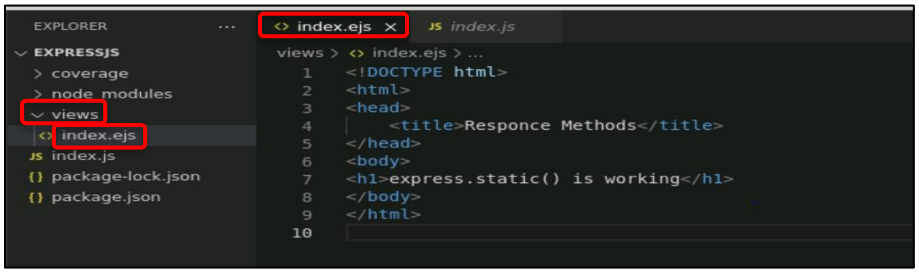
5.2 Run the command below in the terminal to add **ejs** to the project:
npm install ejs



```
labuser@ubuntu2204:~/Expressjs$ npm install ejs

added 16 packages, and audited 74 packages in 3s

9 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
labuser@ubuntu2204:~/Expressjs$
```

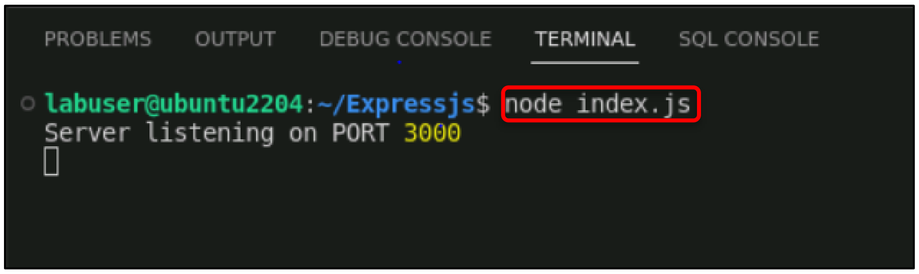
5.3 Create a folder named **views,** then create an **index.ejs** file and add the following code in that file:

```
<!DOCTYPE html>
<html>
<head>
        <title>Responce Methods</title>
</head>
<body>
<h1>express.static() is working</h1>
</body>
</html>
```



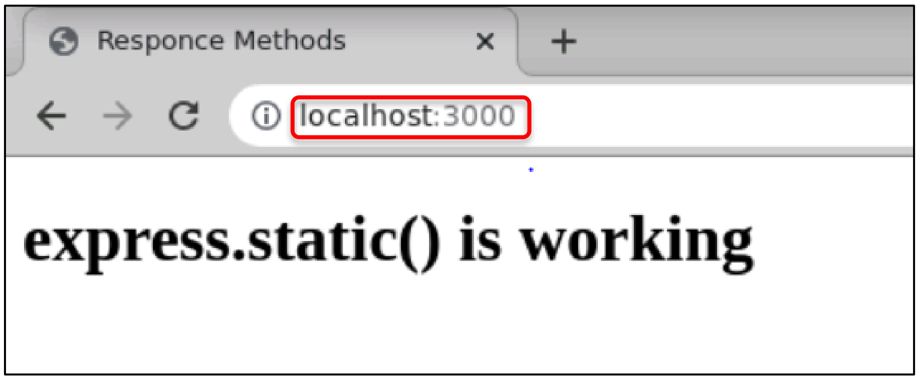
```
<!DOCTYPE html>
<html>
<head>
    <title>Responce Methods</title>
</head>
<body>
<h1>express.static() is working</h1>
</body>
</html>
```

5.4 Run the **node index.js** command in the terminal



```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
```

5.5 Run http://localhost:3000 in the browser and check the output

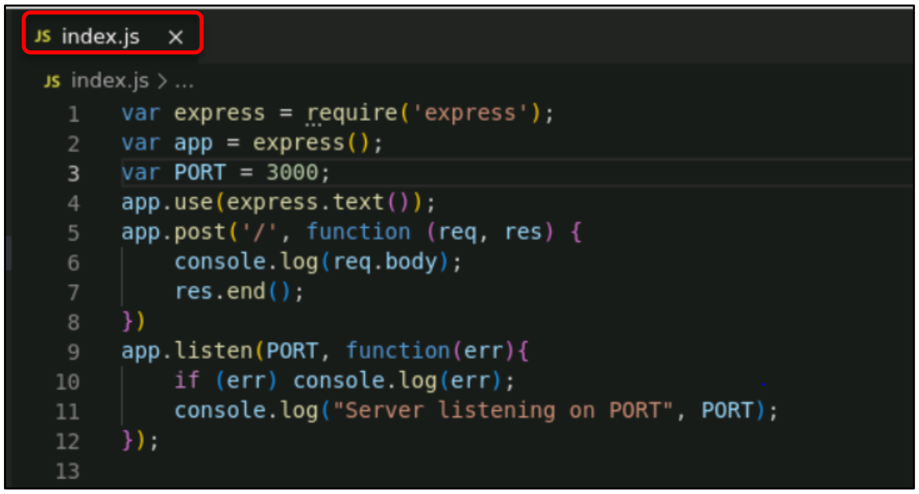


express.static() is working

Step 6: Use express.text() response method in Express.js

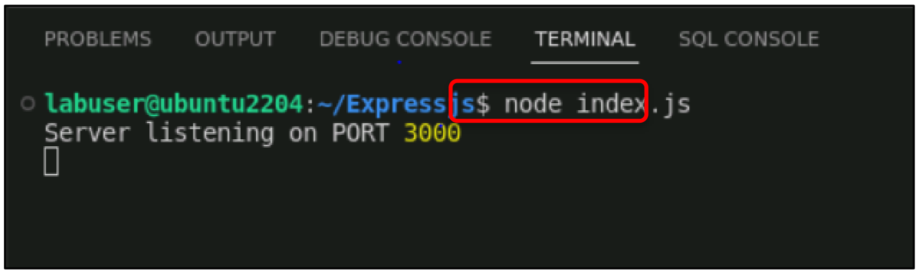
6.1 Open the Expressjs folder in Vscode and write the below code in the **index.js** file:

```
var express = require('express');
var app = express();
var PORT = 3000;

app.use(express.text());

app.post('/', function (req, res) {
   console.log(req.body);
   res.end();
})

app.listen(PORT, function(err){
   if (err) console.log(err);
   console.log("Server listening on PORT", PORT);
});
```



```
var express = require('express');
var app = express();
var PORT = 3000;
app.use(express.text());
app.post('/', function (req, res) {
    console.log(req.body);
    res.end();
})
app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```

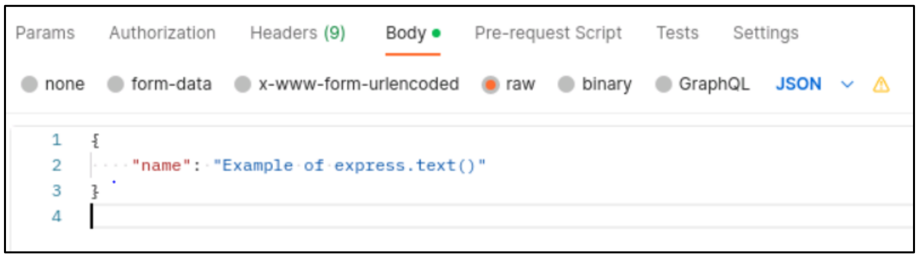
6.2 Run the **node index.js** command in the terminal.



```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
```

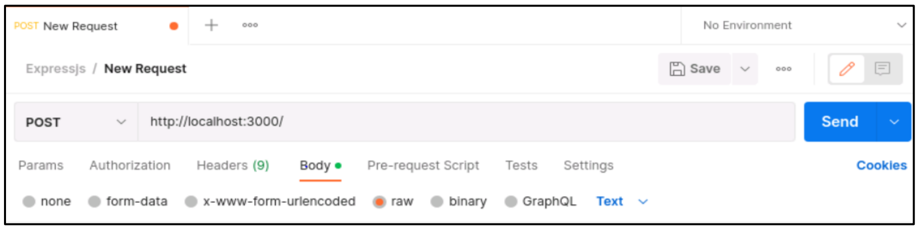
6.3 In the new request, navigate to the **Headers** option and write **key- content-type & value - application/ octet-stream.** Also, go to Body and change type to **Body**. After clicking on the Body, at the end of those options, a drop-down menu is present. Select **JSON** format from that and write the below lines in the workspace:

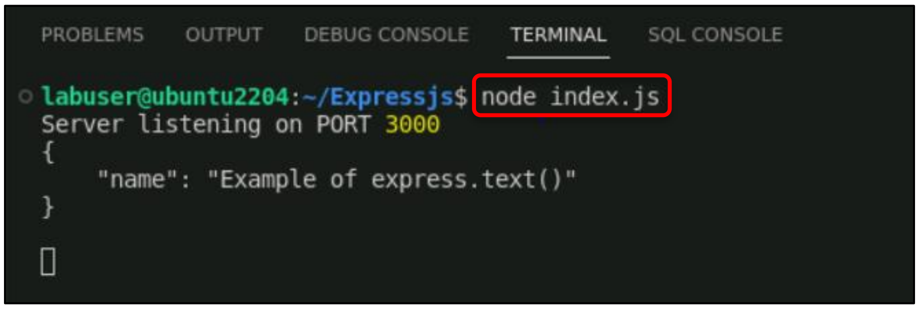
{
"name": "Example of express.text()"
}



```
{
    "name": "Example of express.text()"
}
```

6.4 Execute a POST request to **http://localhost:3000/** and inspect the output in the VSCode terminal where the program is executed



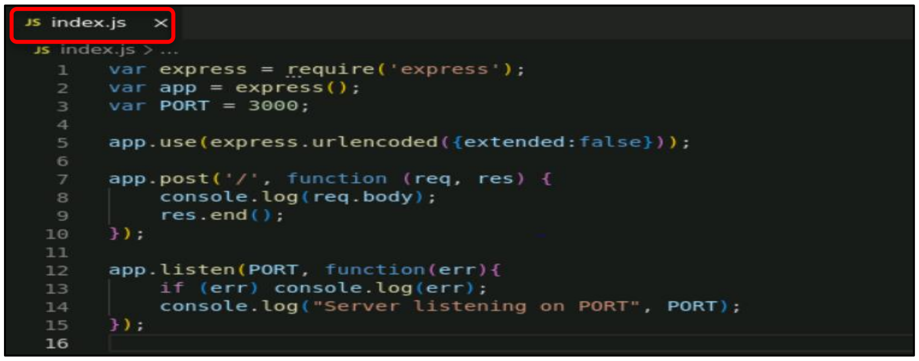


```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
{
    "name": "Example of express.text()"
}
```

Step 7: Use express.urlencoded () response method in Express.js

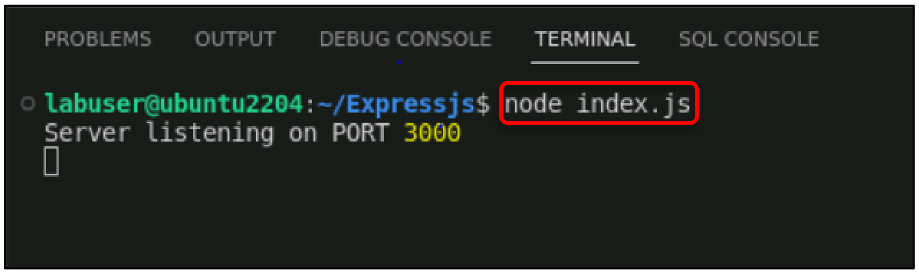
7.1 Open the Expressjs folder in the Vscode and write the below code in the **index.js** file:

```
var express = require('express');
var app = express();
var PORT = 3000;

app.use(express.urlencoded({extended:false}));

app.post('/', function (req, res) {
   console.log(req.body);
   res.end();
});

app.listen(PORT, function(err){
   if (err) console.log(err);
   console.log("Server listening on PORT", PORT);
});
```



```
var express = require('express');
var app = express();
var PORT = 3000;

app.use(express.urlencoded({extended:false}));

app.post('/', function (req, res) {
    console.log(req.body);
    res.end();
});

app.listen(PORT, function(err){
    if (err) console.log(err);
    console.log("Server listening on PORT", PORT);
});
```

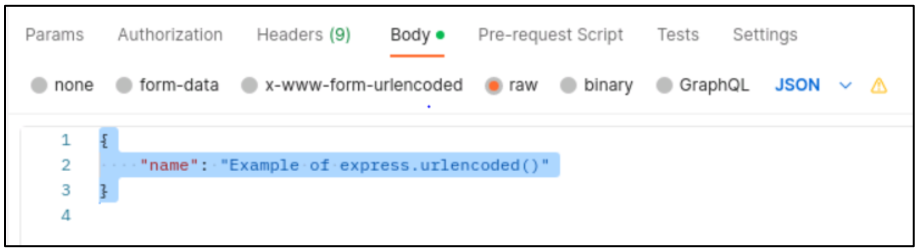
7.2 Run the **node index.js** command in the terminal



```
labuser@ubuntu2204:~/Expressjs$ node index.js
Server listening on PORT 3000
```

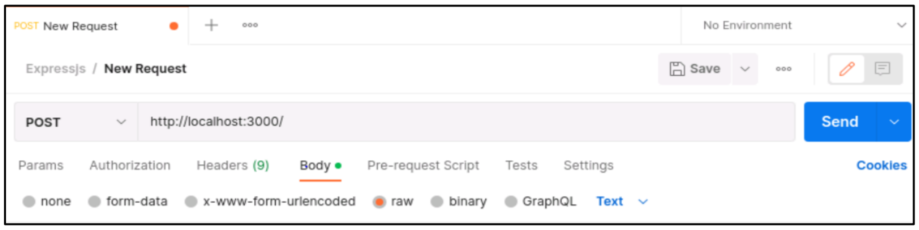
7.3 In the new request, navigate to the **Headers** option and write **key- content-type & value - application/ octet-stream.** Also, go to Body and change type to **Body**. After clicking on Body, at the end of those options, a drop-down menu is present. Select **JSON** format from that and write the below lines in the workspace:

{
"name": "Example of express.urlencoded()"
}



```
{
    "name": "Example of express.urlencoded()"
}
```

7.4 Initiate a GET request to **http://localhost:3000/** and examine the output in the VSCode terminal where the program is executed



POST http://localhost:3000/

Body: raw, Text

By following these steps, you have successfully implemented diverse response methods in Express.js for efficient handling of data and requests.