



Cookie in Nodejs

#Node JS Notes

What is a cookie?

- Cookies are small pieces of data sent from a website and stored in user's web browser while user is browsing that website.
- Every time the user loads that website back, the browser sends that stored data back to website or server, to distinguish user's previous activity.



Facebook Cookie Example

Cookies in use

Allowed Blocked

The following cookies were set when you viewed this page

Cookies

c_user

datr

dpr

fr

Name	c_user
Content	100013416614556
Domain	.facebook.com
Path	/

Block

Remove

Done

Cookies in use

Allowed Blocked

The following cookies were set when you viewed this page

doubleclick.net

facebook.com

Cookies

c_user

datr

dpr

Path	/
Send for	Secure connections only
Created	Thursday, January 21, 2021 at 10:18:15 AM
Expires	Friday, January 21, 2022 at 10:18:13 AM

Block

Remove

Done



How to Check Cookie in Google Chrome

- `chrome://settings/siteData?search=cookies`



What is use of Cookie ?

- To Stored users information
- Authentication
- Users preferences
- Local Storage
- Login Remember me
- Website Visit Counter



Cookie vs Session

- Cookie is same as Session
- Cookie will be visible in Browser
- Session will be hidden in browser.
- Cookie expire time we can set
- On Browser close Session value will be expire.



How Does Express Look Like?

```
var express = require('express');  
var app = express();  
app.get('/', function (req, res) {  
  res.send('Hello World!');  
});
```

```
var server = app.listen(3000, function () {  
  var host = server.address().address;  
  var port = server.address().port;  
  console.log('Example app listening at http://%s:%s', host, port);  
});
```



Installation

- By Default in Express Generator Cookie-parser will be there.
- For external installation Open your terminal, Browse to your app folder and install via below command
- `npm install -g cookie-parser`
- `npm install cookie-parser --save`



Using Cookie-Parser

- First import Cookie parser package in application.
- `var express = require('express');`
- **`var cookieParser = require('cookie-parser');`**
- `var app = express();`
- **`app.use(cookieParser());`**



Syntax

- Cookie-parser parses Cookie header and populate **req.cookies** with an object keyed by the cookie names.
- To set a new cookie lets define a new route in your express app like :
- `res.cookie('cookie_name' , 'cookie_value');`



How to Create Cookie ?

```
app.get('/create-cookie',function(req, res){  
    res.cookie('cookie_name' , 'cookie_value').send('Cookie is set');  
});
```



How to Set Cookie Expiration Time?

- Cookie expire time can be set easily by :
- `res.cookie('name' , 'value', {expire : new Date() + 9999});`



How to Print Cookie

```
app.get('/', function(req, res) {  
  console.log("Cookies : ", req.cookies);  
});
```

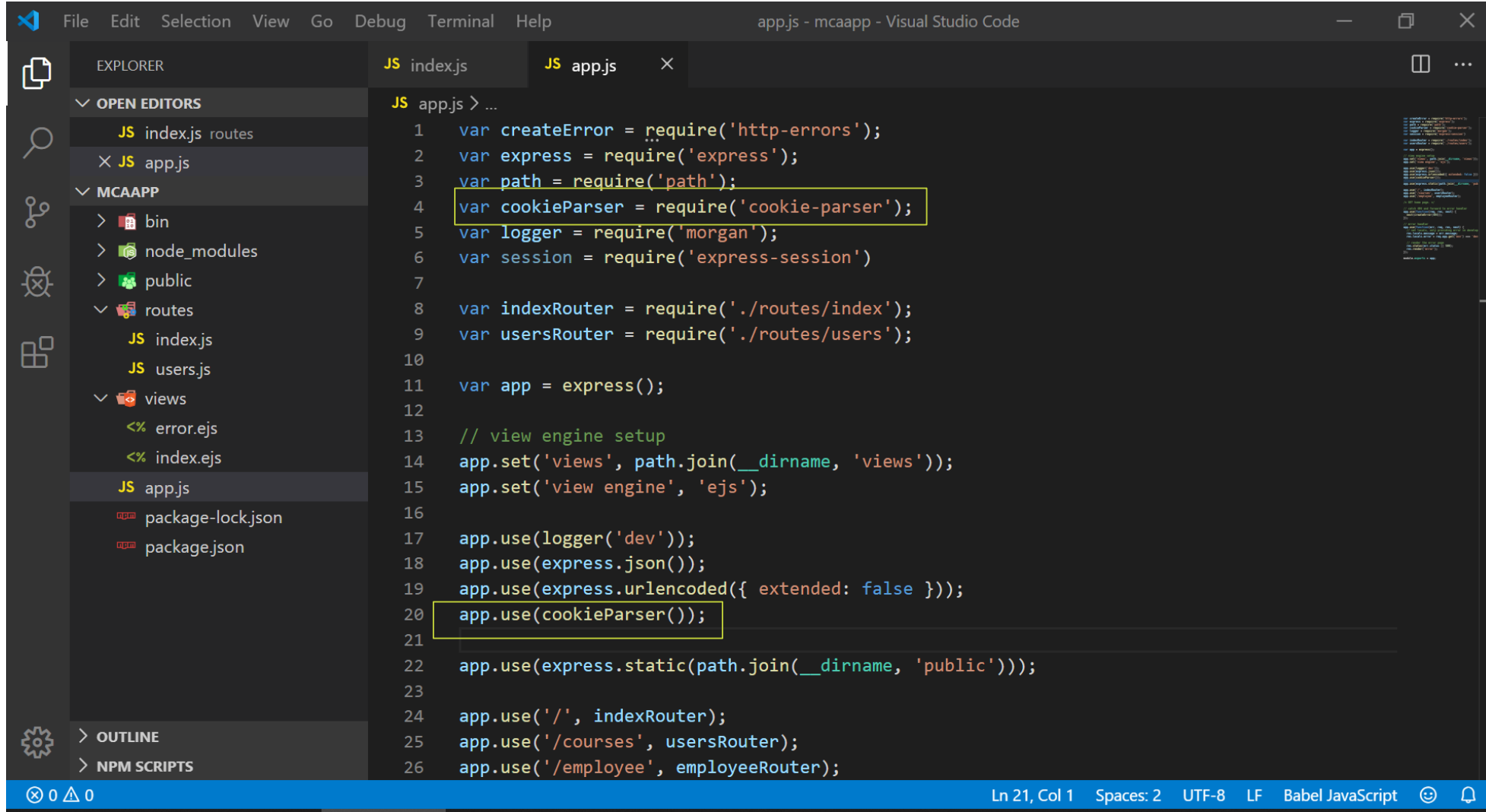


Delete Cookie

```
app.get('/clearcookie', function(req,res){  
    res.clearCookie('cookie_name');  
    res.send('Cookie deleted');  
});
```



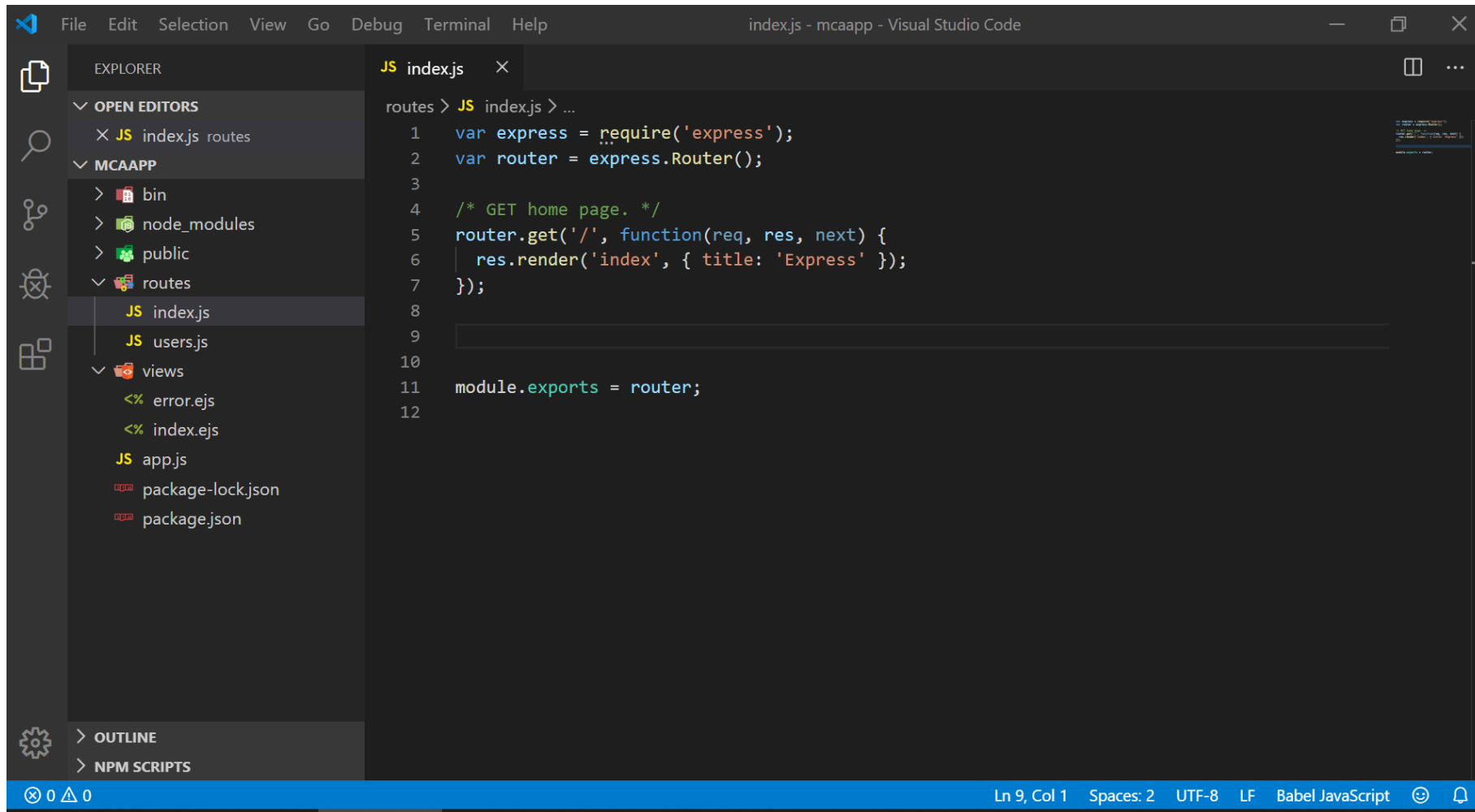
Default app.js



```
1  var createError = require('http-errors');
2  var express = require('express');
3  var path = require('path');
4  var cookieParser = require('cookie-parser');
5  var logger = require('morgan');
6  var session = require('express-session');
7
8  var indexRouter = require('./routes/index');
9  var usersRouter = require('./routes/users');
10
11 var app = express();
12
13 // view engine setup
14 app.set('views', path.join(__dirname, 'views'));
15 app.set('view engine', 'ejs');
16
17 app.use(logger('dev'));
18 app.use(express.json());
19 app.use(express.urlencoded({ extended: false }));
20 app.use(cookieParser());
21
22 app.use(express.static(path.join(__dirname, 'public')));
23
24 app.use('/', indexRouter);
25 app.use('/courses', usersRouter);
26 app.use('/employee', employeeRouter);
```



Index.js Route



```
index.js - mcaapp - Visual Studio Code

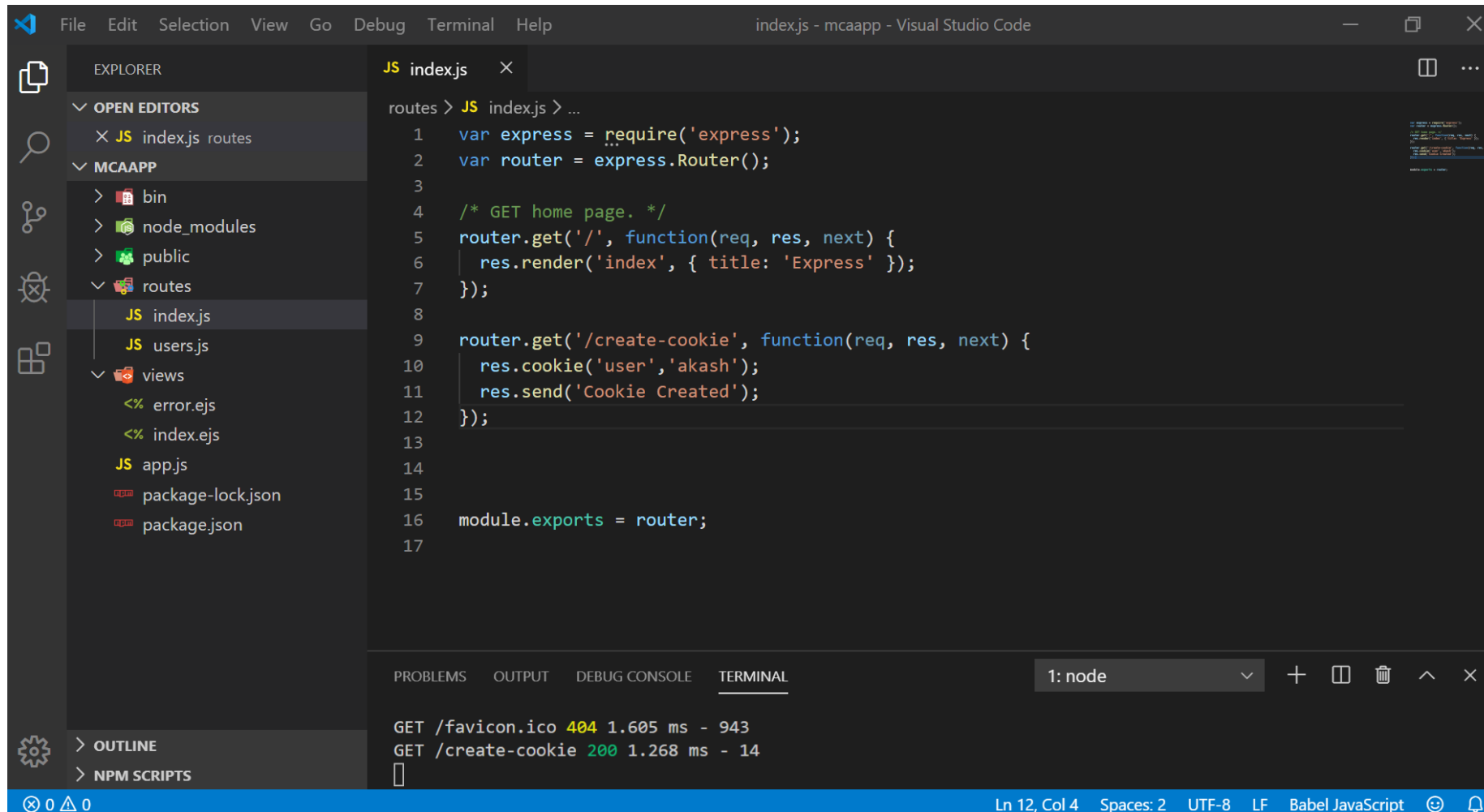
EXPLORER
├── OPEN EDITORS
│   └── JS index.js routes
├── MCAAPP
│   ├── bin
│   ├── node_modules
│   ├── public
│   ├── routes
│   │   ├── JS index.js
│   │   ├── JS users.js
│   │   └── views
│   │       ├── <% error.ejs
│   │       ├── <% index.ejs
│   │       ├── JS app.js
│   │       ├── package-lock.json
│   │       └── package.json
│   └── OUTLINE
│       └── NPM SCRIPTS
└── 0 0 0

JS index.js
1  var express = require('express');
2  var router = express.Router();
3
4  /* GET home page. */
5  router.get('/', function(req, res, next) {
6    res.render('index', { title: 'Express' });
7  });
8
9
10
11 module.exports = router;
12

Ln 9, Col 1  Spaces: 2  UTF-8  LF  Babel JavaScript
```



Create Route for Cookie



The screenshot shows the Visual Studio Code interface with a project named 'mcaapp'. The Explorer sidebar on the left shows the file structure, including a 'routes' directory. The 'index.js' file in the 'routes' directory is open in the editor. The code defines an Express.js application with two routes: a home page and a route to create a cookie. The terminal at the bottom shows the results of running the application, including a successful GET request to the '/create-cookie' endpoint.

```
index.js - mcaapp - Visual Studio Code

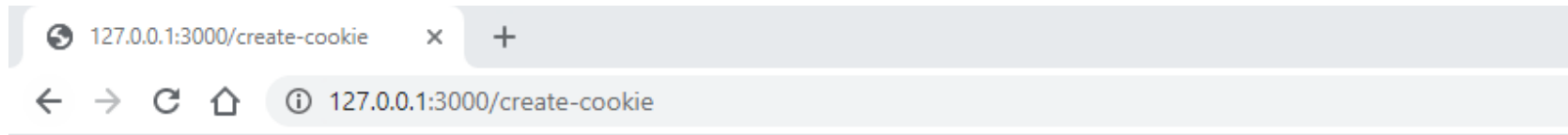
EXPLORER
  OPEN EDITORS
    JS index.js routes
  MCAAPP
    bin
    node_modules
    public
    routes
      JS index.js
      JS users.js
    views
      error.ejs
      index.ejs
    app.js
    package-lock.json
    package.json

JS index.js
1 var express = require('express');
2 var router = express.Router();
3
4 /* GET home page. */
5 router.get('/', function(req, res, next) {
6   res.render('index', { title: 'Express' });
7 });
8
9 router.get('/create-cookie', function(req, res, next) {
10   res.cookie('user', 'akash');
11   res.send('Cookie Created');
12 });
13
14
15
16 module.exports = router;
17

TERMINAL
1: node
GET /favicon.ico 404 1.605 ms - 943
GET /create-cookie 200 1.268 ms - 14
```



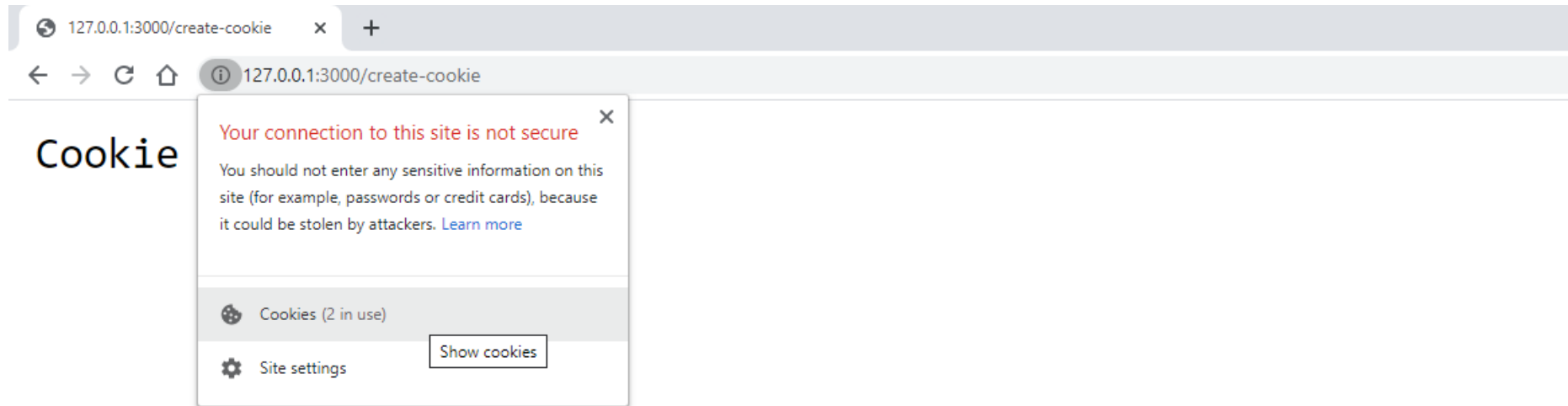
Run Project



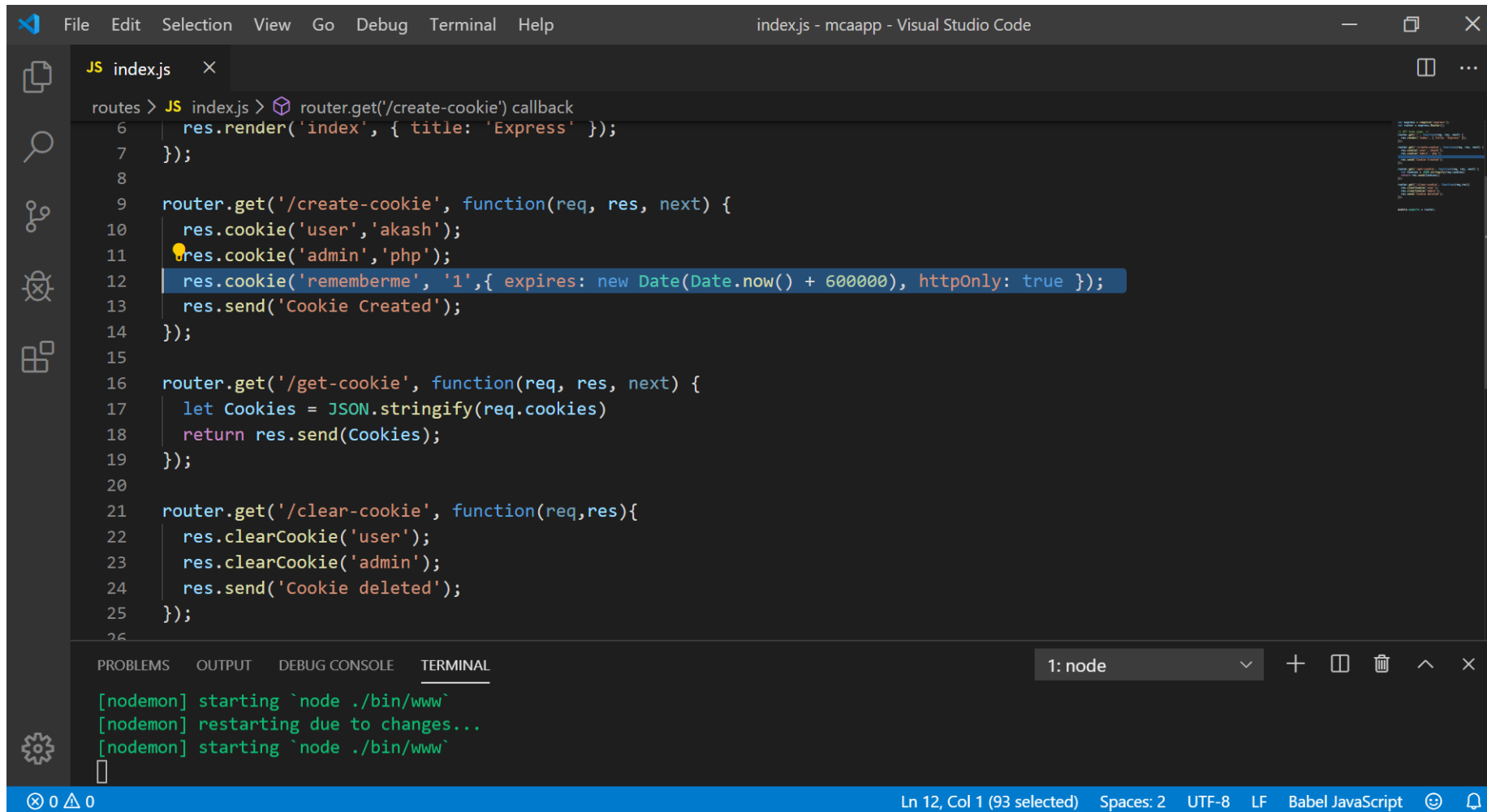
Cookie Created



Check Cookie



Create Cookie With Expire Time



```
index.js - mcaapp - Visual Studio Code

JS index.js
routes > JS index.js > router.get('/create-cookie') callback
6   res.render('index', { title: 'Express' });
7   });
8
9   router.get('/create-cookie', function(req, res, next) {
10    res.cookie('user', 'akash');
11    res.cookie('admin', 'php');
12    res.cookie('rememberme', '1', { expires: new Date(Date.now() + 600000), httpOnly: true });
13    res.send('Cookie Created');
14  });
15
16  router.get('/get-cookie', function(req, res, next) {
17    let Cookies = JSON.stringify(req.cookies)
18    return res.send(Cookies);
19  });
20
21  router.get('/clear-cookie', function(req, res){
22    res.clearCookie('user');
23    res.clearCookie('admin');
24    res.send('Cookie deleted');
25  });
26

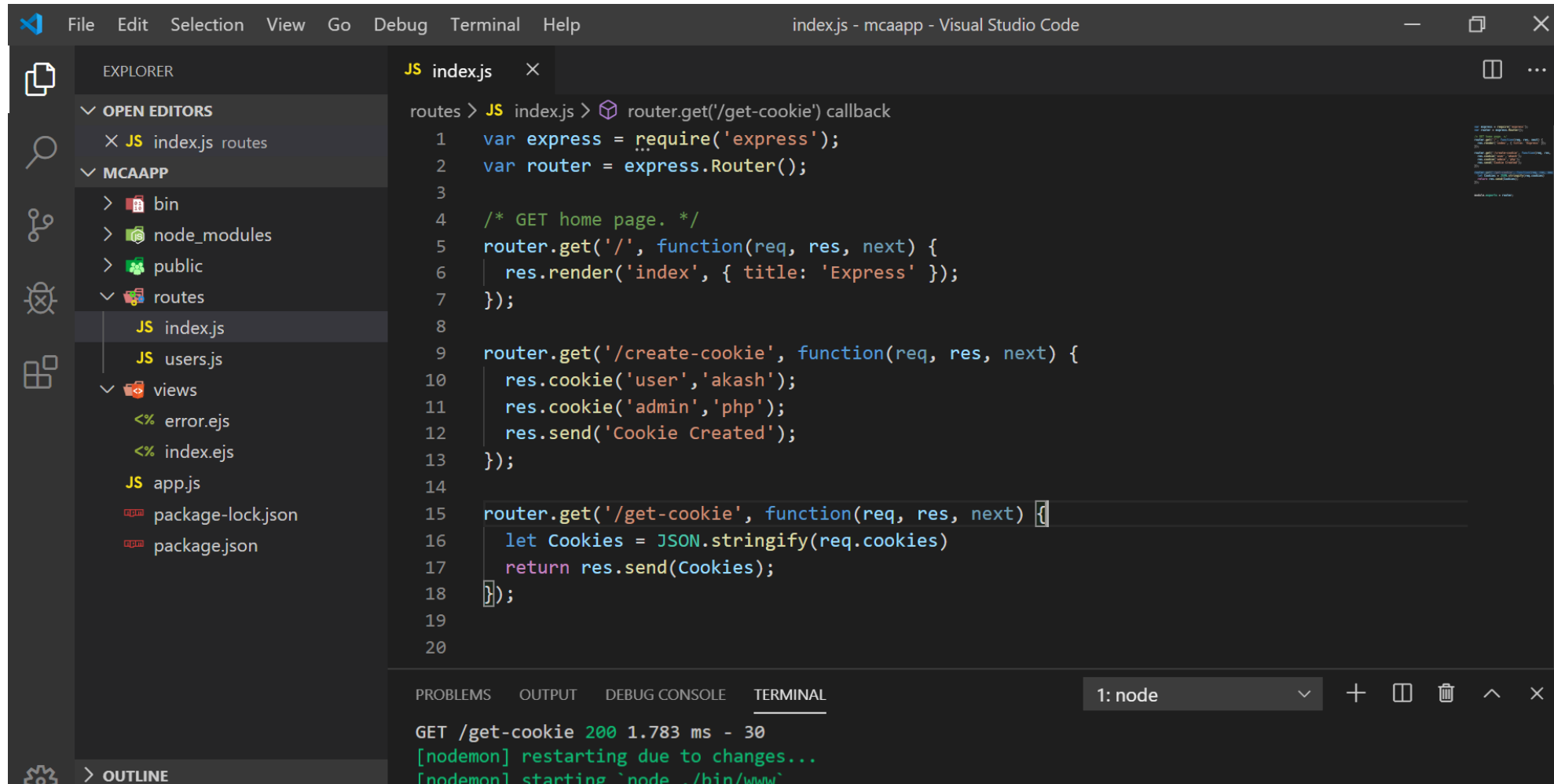
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: node
[nodemon] starting `node ./bin/www`
[nodemon] restarting due to changes...
[nodemon] starting `node ./bin/www`

Ln 12, Col 1 (93 selected) Spaces: 2 UTF-8 LF Babel JavaScript
```



Print Cookie as Json Response

Print cookie as Json format using json.stringify function



The screenshot shows the Visual Studio Code editor with a project named 'mcaapp'. The Explorer sidebar on the left shows the file structure: 'bin', 'node_modules', 'public', 'routes' (containing 'index.js' and 'users.js'), and 'views' (containing 'error.ejs' and 'index.ejs'). The 'index.js' file is open in the editor, showing the following code:

```
routes > JS index.js > router.get('/get-cookie') callback
1  var express = require('express');
2  var router = express.Router();
3
4  /* GET home page. */
5  router.get('/', function(req, res, next) {
6    res.render('index', { title: 'Express' });
7  });
8
9  router.get('/create-cookie', function(req, res, next) {
10    res.cookie('user', 'akash');
11    res.cookie('admin', 'php');
12    res.send('Cookie Created');
13  });
14
15  router.get('/get-cookie', function(req, res, next) {
16    let Cookies = JSON.stringify(req.cookies)
17    return res.send(Cookies);
18  });
19
20
```

The terminal at the bottom shows the command 'GET /get-cookie' being executed, with a response time of 200 ms and a status of 300. The terminal output shows '[nodemon] restarting due to changes...' and '[nodemon] starting `node ./bin/www`'.



Json Response of Cookie



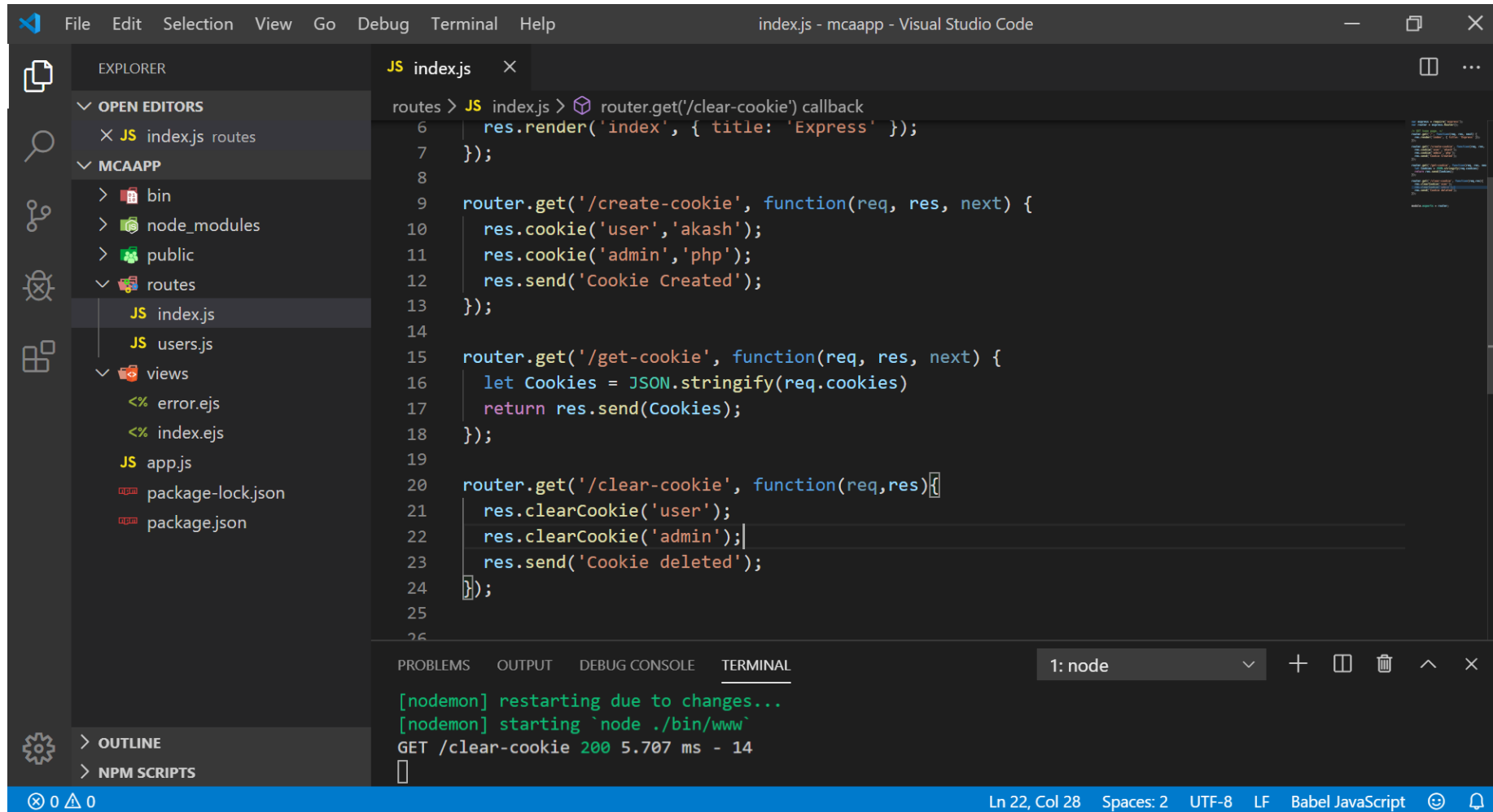
The screenshot shows a web browser with two tabs: '127.0.0.1:3000/create-cookie' and '127.0.0.1:3000/get-cookie'. The active tab is '127.0.0.1:3000/get-cookie'. The browser's address bar shows the URL '127.0.0.1:3000/get-cookie'. The main content area displays a JSON response with line numbers 1 through 7 on the left. The JSON is as follows:

```
1 // 20200121132639
2 // http://127.0.0.1:3000/get-cookie
3
4 {
5   "user": "akash",
6   "admin": "php"
7 }
```



Delete Cookie

```
res.clearCookie('mycookies', { expires: new Date(), path: '/' });
```



The screenshot shows the Visual Studio Code editor with a project named 'mcaapp'. The Explorer sidebar on the left shows the file structure: 'bin', 'node_modules', 'public', 'routes' (containing 'index.js', 'users.js'), and 'views' (containing 'error.ejs' and 'index.ejs'). The main editor window displays 'index.js' with the following code:

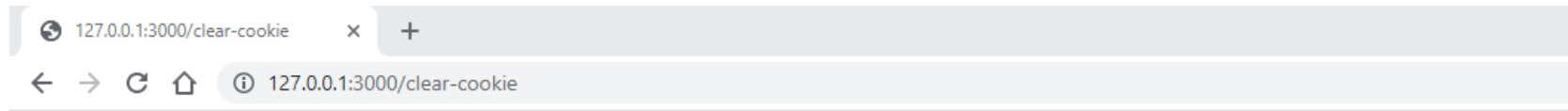
```
routes > JS index.js > router.get('/clear-cookie') callback
6   res.render('index', { title: 'Express' });
7   });
8
9   router.get('/create-cookie', function(req, res, next) {
10     res.cookie('user', 'akash');
11     res.cookie('admin', 'php');
12     res.send('Cookie Created');
13   });
14
15   router.get('/get-cookie', function(req, res, next) {
16     let Cookies = JSON.stringify(req.cookies)
17     return res.send(Cookies);
18   });
19
20   router.get('/clear-cookie', function(req, res){
21     res.clearCookie('user');
22     res.clearCookie('admin');
23     res.send('Cookie deleted');
24   });
25
26
```

The bottom of the editor shows the TERMINAL panel with the following output:

```
1: node
[nodemon] restarting due to changes...
[nodemon] starting `node ./bin/www`
GET /clear-cookie 200 5.707 ms - 14
```



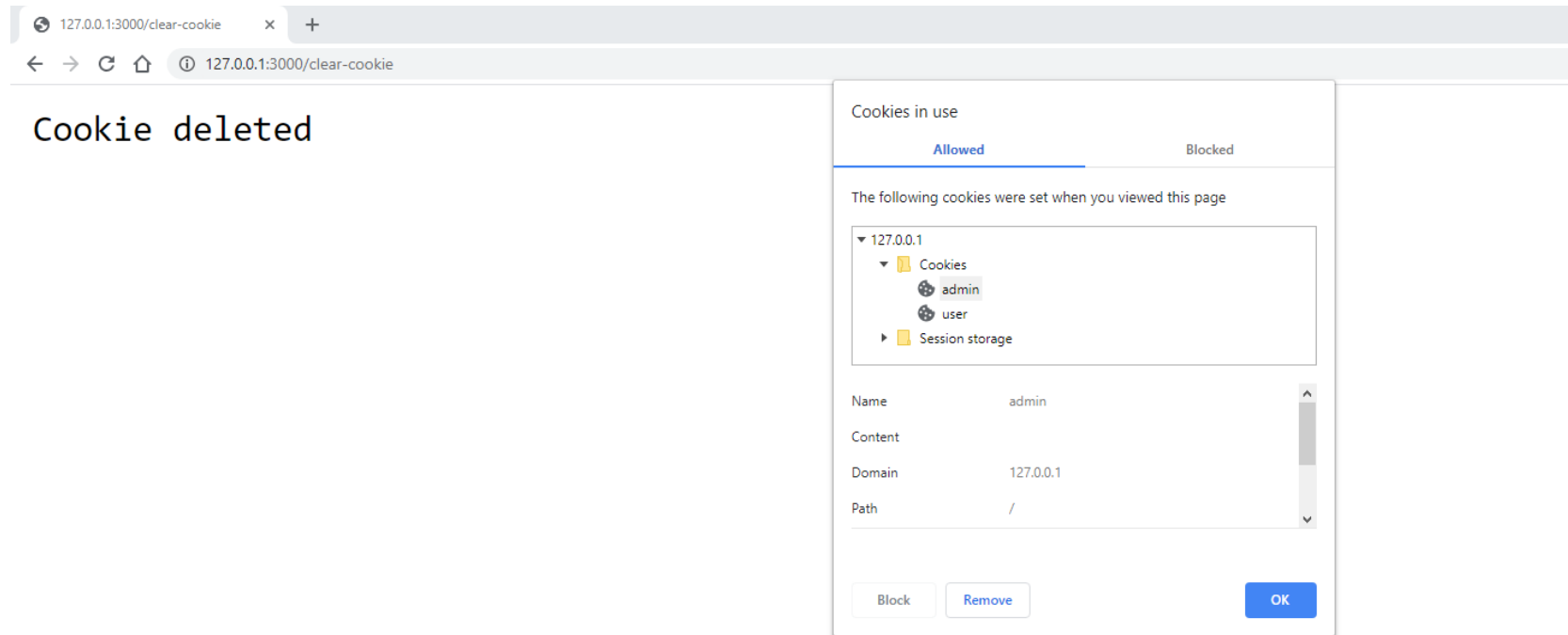
Response of Delete Cookie



Cookie deleted



Check Cookie is Deleted or not

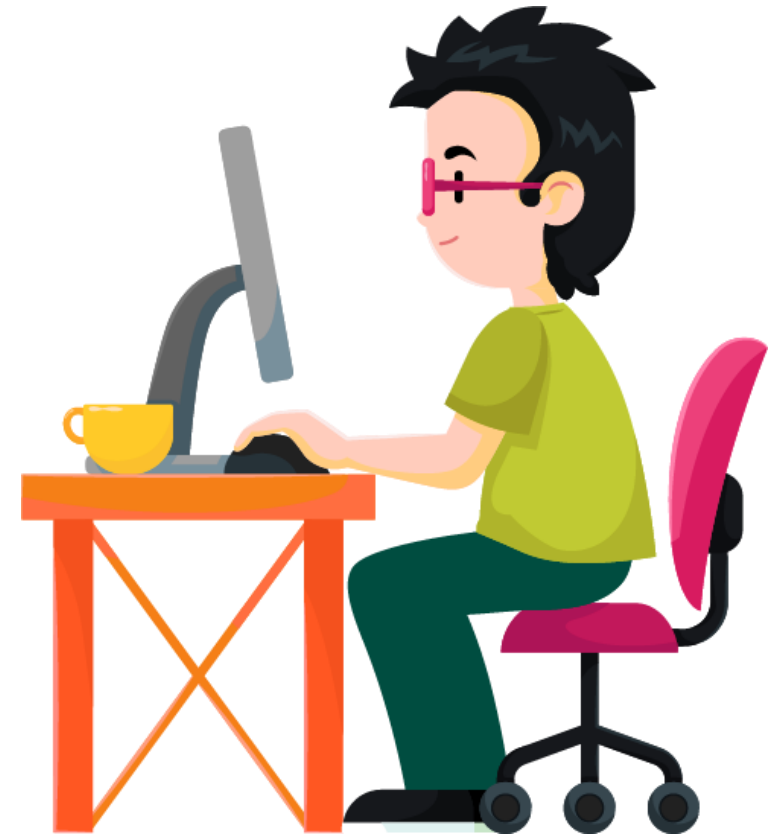


Task

- Take one Textbox and Get Color Name from the user and on submit button redirect on home page. On Homepage background color display as per the value.
- Counter Demo in Cookie. Store Current value in cookie on refresh increment value by 1.



Get Exclusive Video Tutorials



www.apptutorials.com

<https://www.youtube.com/user/Akashtips>





Get More Details

www.akashsir.com



If You Liked It !

Rating Us Now



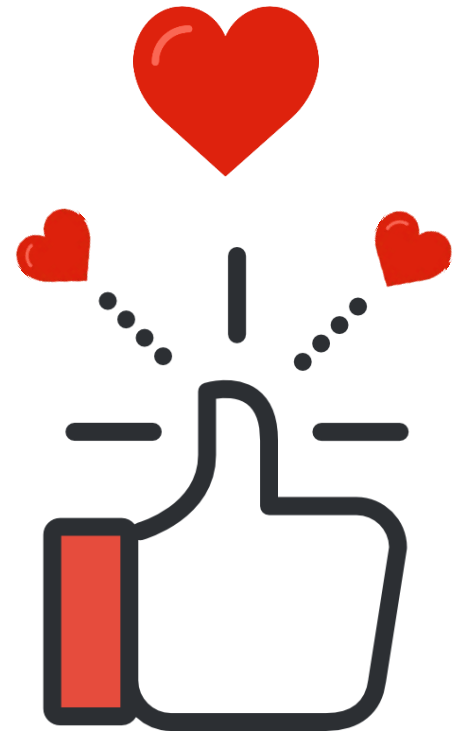
Just Dial

https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET



Sulekha

<https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad>



Connect With Me



Akash Padhiyar
#AkashSir

www.akashsir.com
www.akashtechlabs.com
www.akashpadhiyar.com
www.apptutorials.com

Social Info



Akash.padhiyar



Akashpadhiyar



Akash_padhiyar



+91 99786-21654



#Akashpadhiyar
#apptutorials