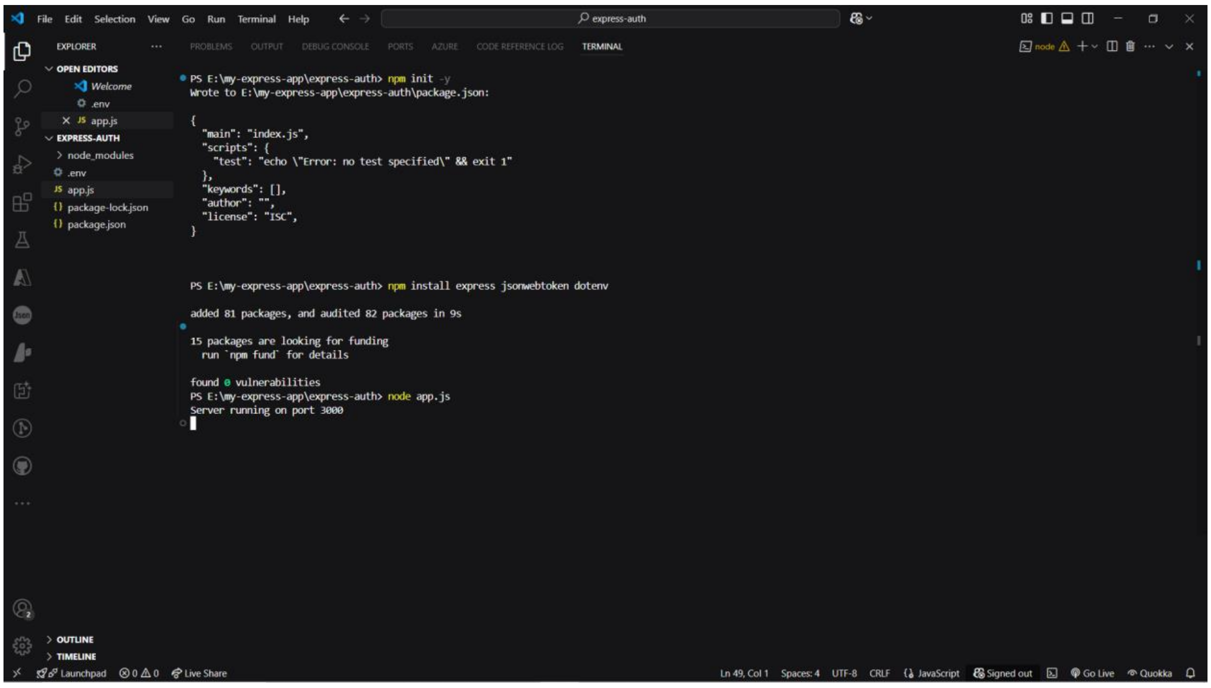
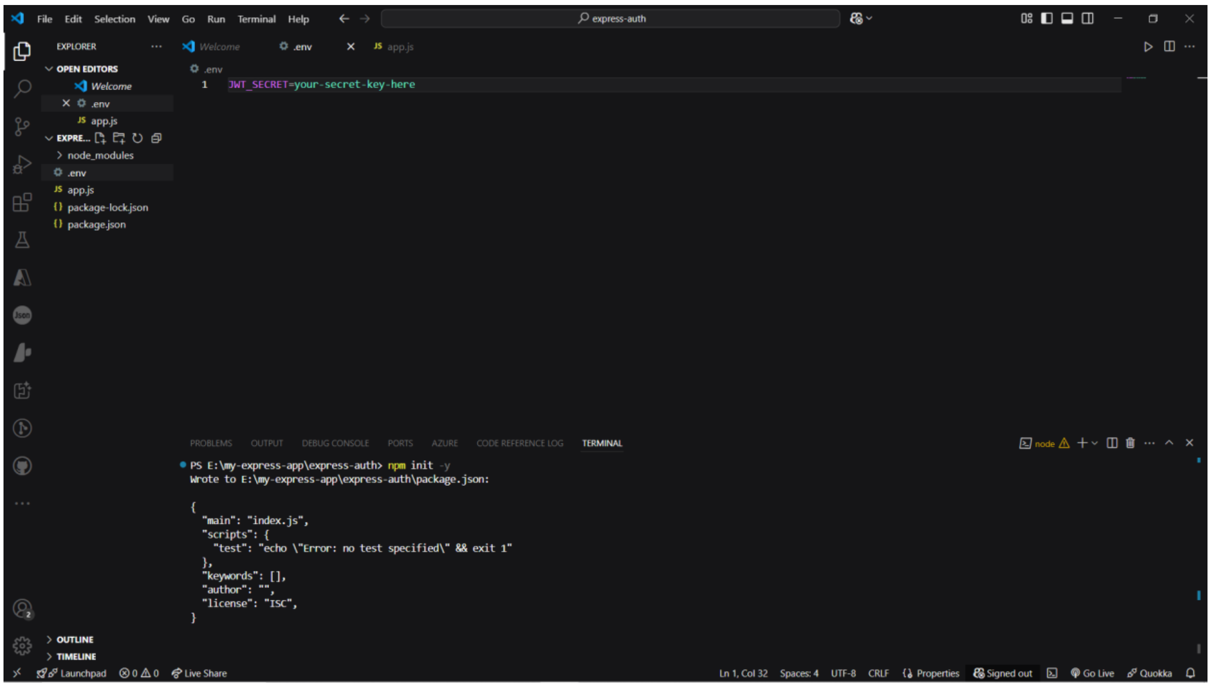
# 12 June, 2025

## SESSION 1(10:30 am -1:30)

JSON Method:-

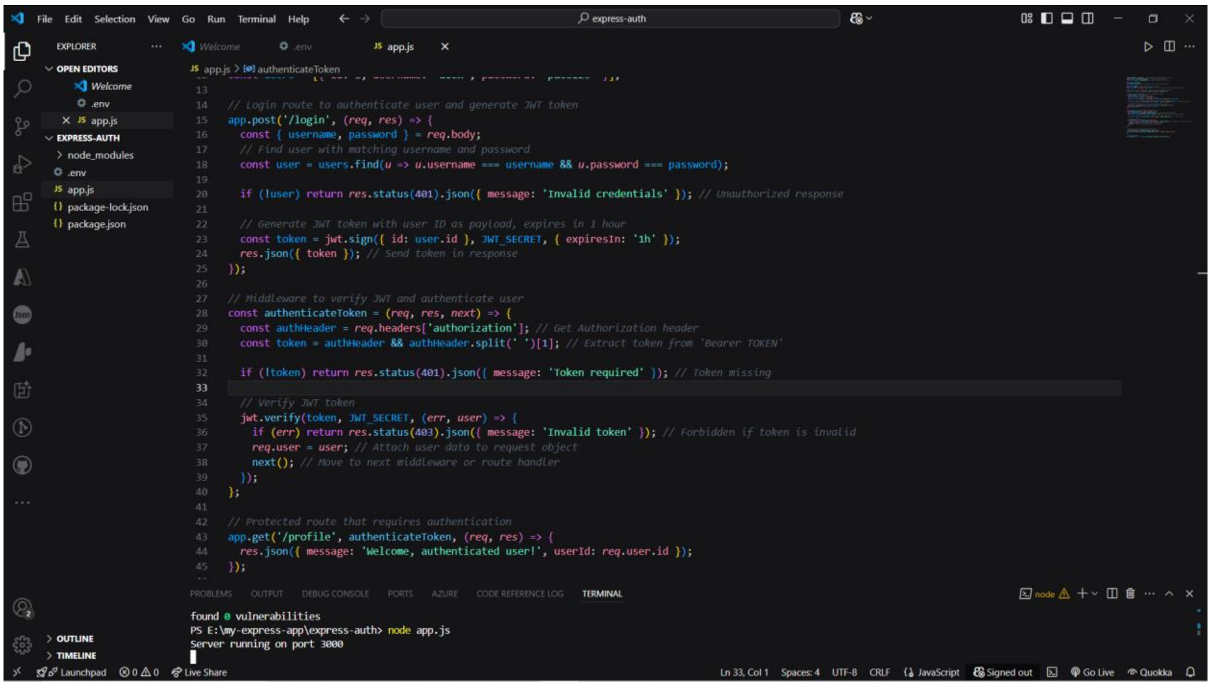


```
PS E:\my-express-app\express-auth> npm init -y
Wrote to E:\my-express-app\express-auth\package.json:

{
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
}


PS E:\my-express-app\express-auth> npm install express jsonwebtoken dotenv

added 81 packages, and audited 82 packages in 9s

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

found 0 vulnerabilities
PS E:\my-express-app\express-auth> node app.js
Server running on port 3000
```



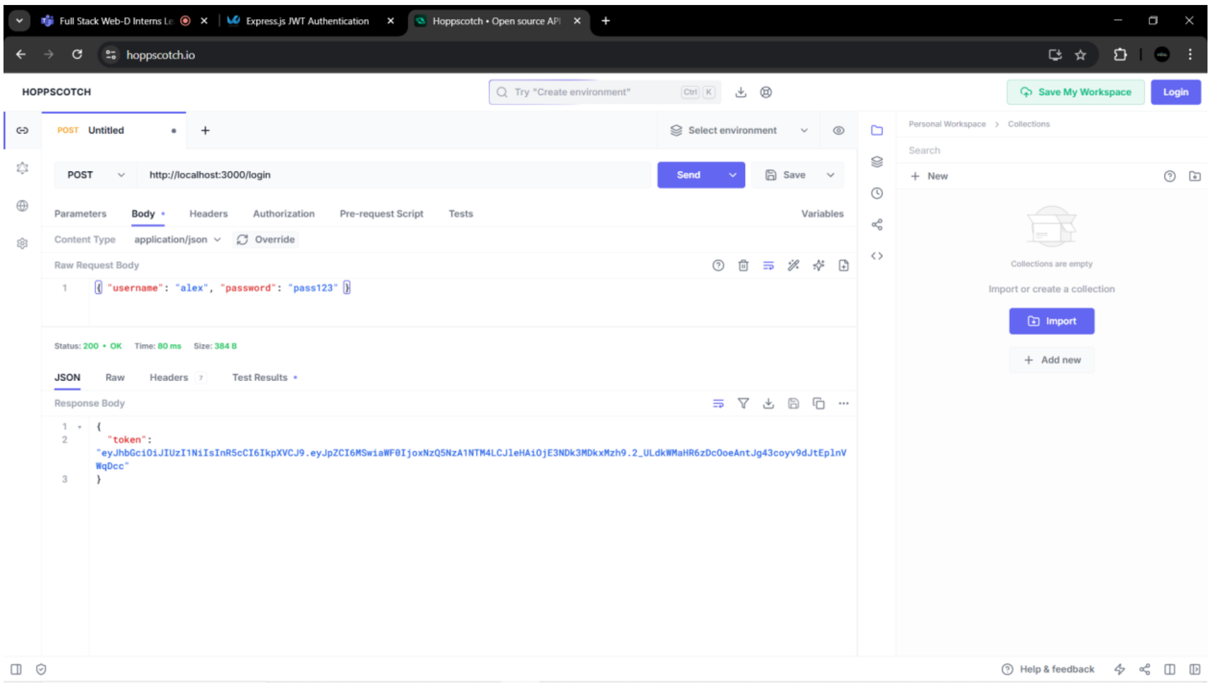
```
JWT_SECRET=your-secret-key-here
```

```
PS E:\my-express-app\express-auth> npm init -y
Wrote to E:\my-express-app\express-auth\package.json:

{
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
}
```



```
// Login route to authenticate user and generate JWT token
app.post('/login', (req, res) => {
  const { username, password } = req.body;
  // Find user with matching username and password
  const user = users.find(u => u.username === username && u.password === password);

  if (!user) return res.status(401).json({ message: 'Invalid credentials' }); // Unauthorized response

  // Generate JWT token with user ID as payload, expires in 1 hour
  const token = jwt.sign({ id: user.id }, JWT_SECRET, { expiresIn: '1h' });
  res.json({ token }); // Send token in response
});

// Middleware to verify JWT and authenticate user
const authenticateToken = (req, res, next) => {
  const authHeader = req.headers['authorization']; // Get Authorization header
  const token = authHeader && authHeader.split(' ')[1]; // Extract token from 'Bearer TOKEN'

  if (!token) return res.status(401).json({ message: 'Token required' }); // Token missing


  // Verify JWT token
  jwt.verify(token, JWT_SECRET, (err, user) => {
    if (err) return res.status(403).json({ message: 'Invalid token' }); // Forbidden if token is invalid
    req.user = user; // Attach user data to request object
    next(); // Move to next middleware or route handler
  });
};

// Protected route that requires authentication
app.get('/profile', authenticateToken, (req, res) => {
  res.json({ message: 'Welcome, authenticated user!', userId: req.user.id });
});
```

```
found 0 vulnerabilities
PS E:\my-express-app\express-auth> node app.js
Server running on port 3000
```



POST http://localhost:3000/login

Content Type: application/json

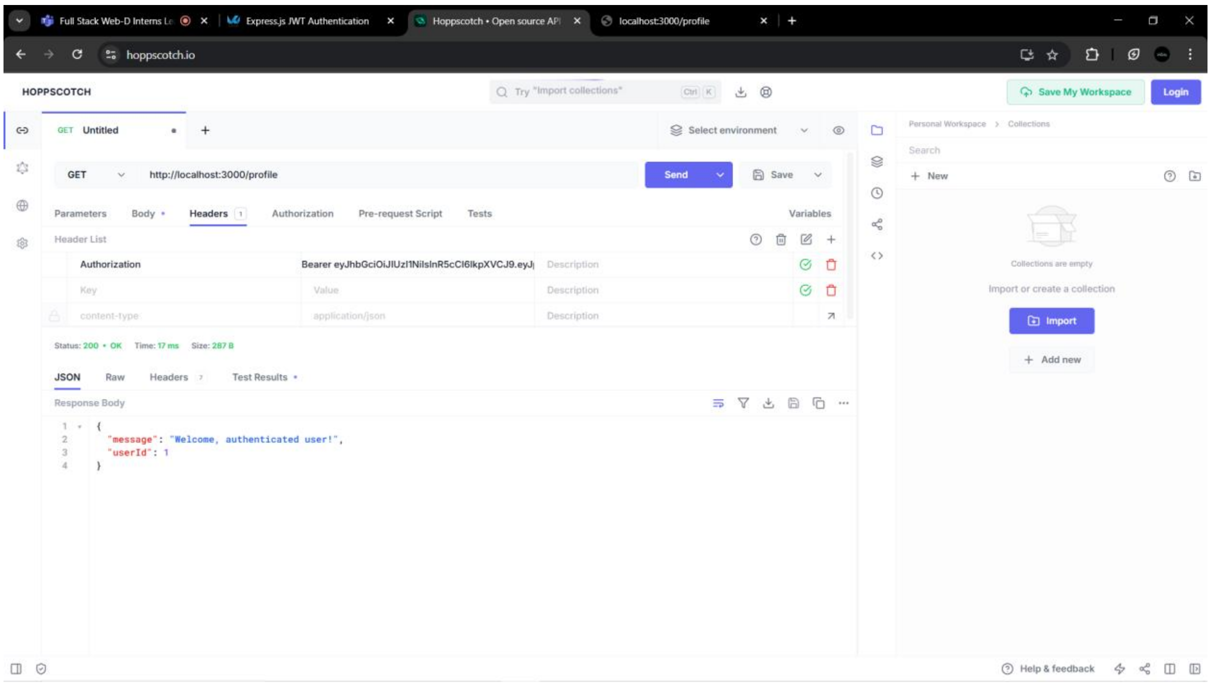
Raw Request Body

```
{ "username": "alex", "password": "pass123" }
```

Status: 200 • OK  Time: 80 ms  Size: 384 B

Response Body

```
{
  "token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MSwiaWF0IjoxNzQ5NzA1NTM4LCJleHAiOjE3NDk3MDkxMzh9.2_ULdkWMaHR6zDcOoeAntJg43coyv9dJtEplnVWqDcc"
}
```



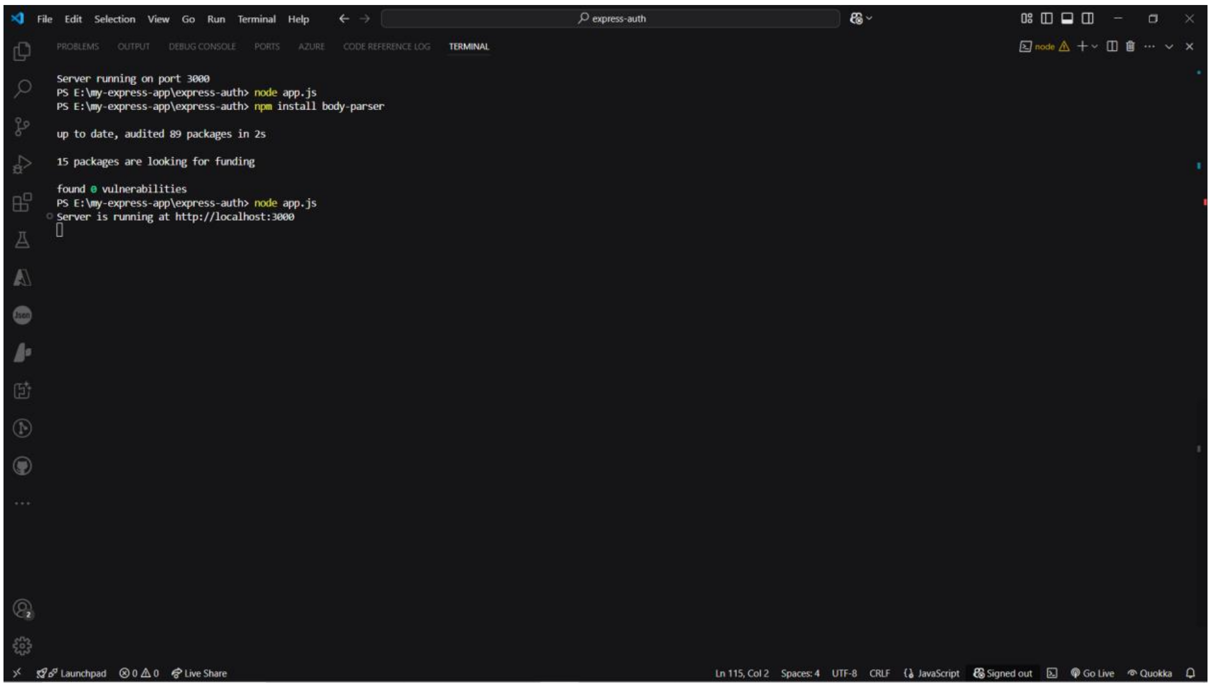
GET http://localhost:3000/profile

| Key | Value | Description |
|---|---|---|
| Authorization | Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ | Description |
| Key | Value | Description |
| content-type | application/json | Description |

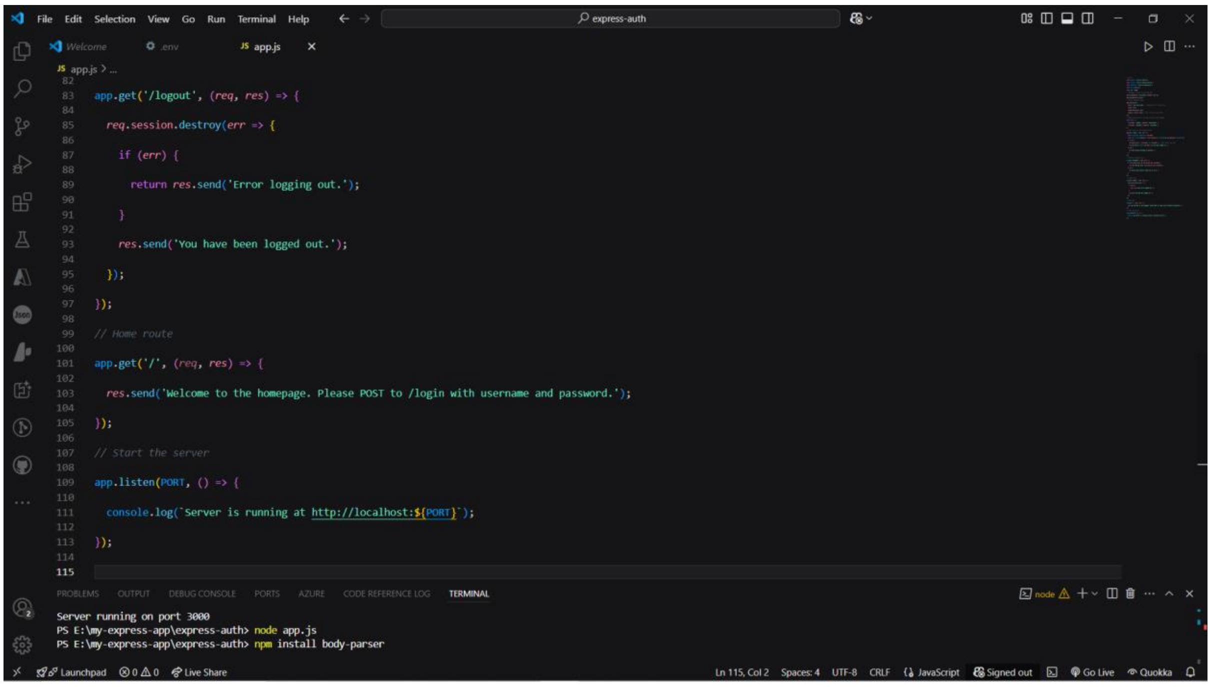
Status: 200 • OK Time: 17 ms Size: 287 B

```
{
  "message": "Welcome, authenticated user!",
  "userId": 1
}
```

SESSION METHOD:-

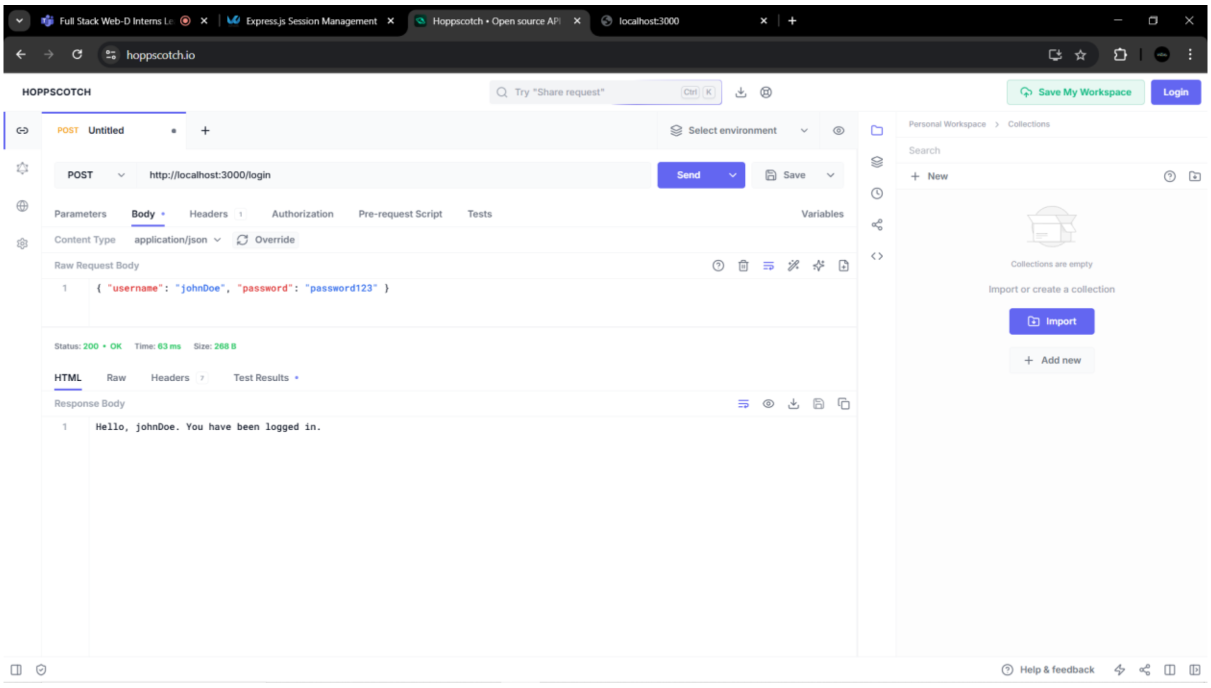

File Edit Selection View Go Run Terminal Help
express-auth
PROBLEMS OUTPUT DEBUG CONSOLE PORTS AZURE CODE REFERENCE LOG TERMINAL
Launchpad
Live Share
Ln 115, Col 2 Spaces: 4 UTF-8 CRLF JavaScript Signed out Go Live Quokka

```
Server running on port 3000
PS E:\my-express-app\express-auth> node app.js
PS E:\my-express-app\express-auth> npm install body-parser

up to date, audited 89 packages in 2s

15 packages are looking for funding

found 0 vulnerabilities
PS E:\my-express-app\express-auth> node app.js
Server is running at http://localhost:3000
```

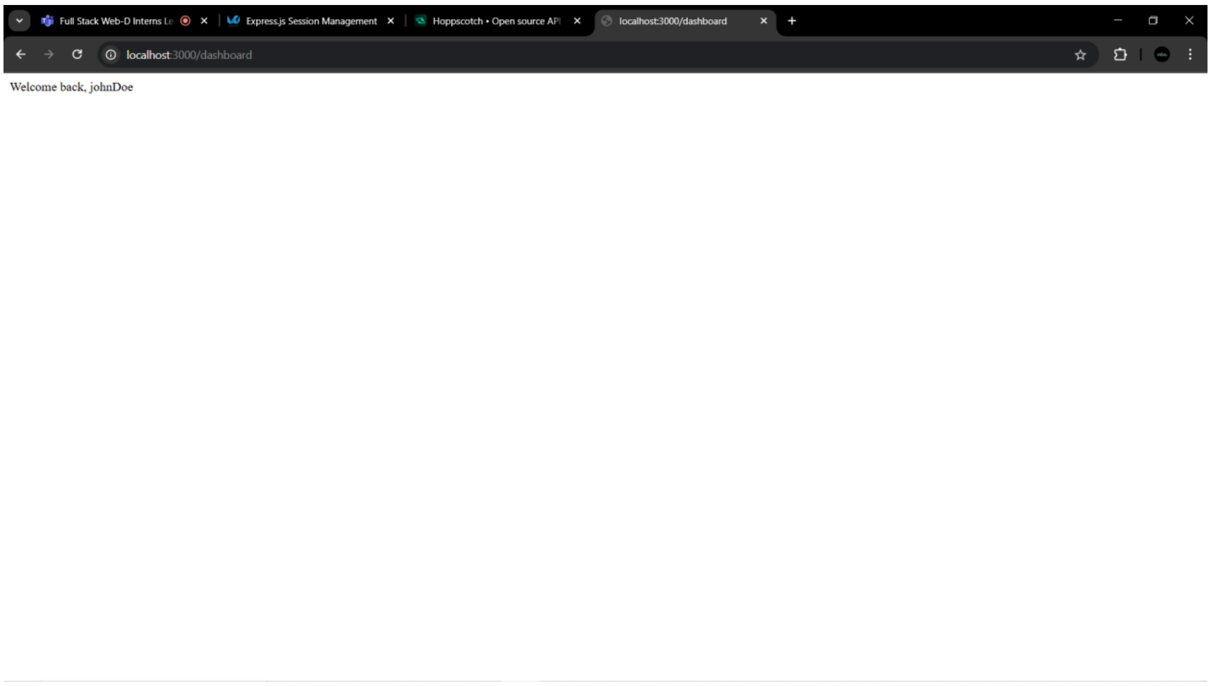


```
app.get('/logout', (req, res) => {

  req.session.destroy(err => {

    if (err) {

      return res.send('Error logging out.');

    }

    res.send('You have been logged out.');

  });

});

// Home route

app.get('/', (req, res) => {

  res.send('Welcome to the homepage. Please POST to /login with username and password.');

});

// Start the server

app.listen(PORT, () => {

  console.log(`Server is running at http://localhost:${PORT}`);

});
```

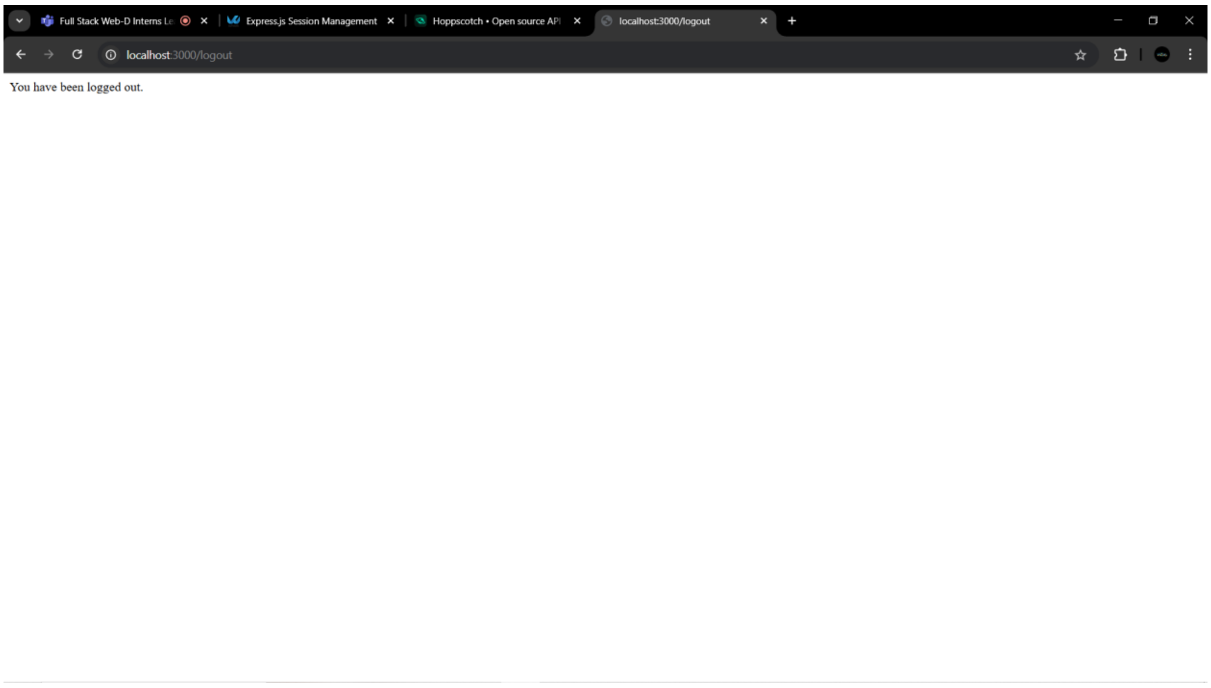
```
Server running on port 3000
PS E:\my-express-app\express-auth> node app.js
PS E:\my-express-app\express-auth> npm install body-parser
```



```
POST http://localhost:3000/login
Content Type: application/json
Raw Request Body
{ "username": "johnDoe", "password": "password123" }

Status: 200 • OK   Time: 63 ms   Size: 268 B
Response Body
Hello, johnDoe. You have been logged in.
```

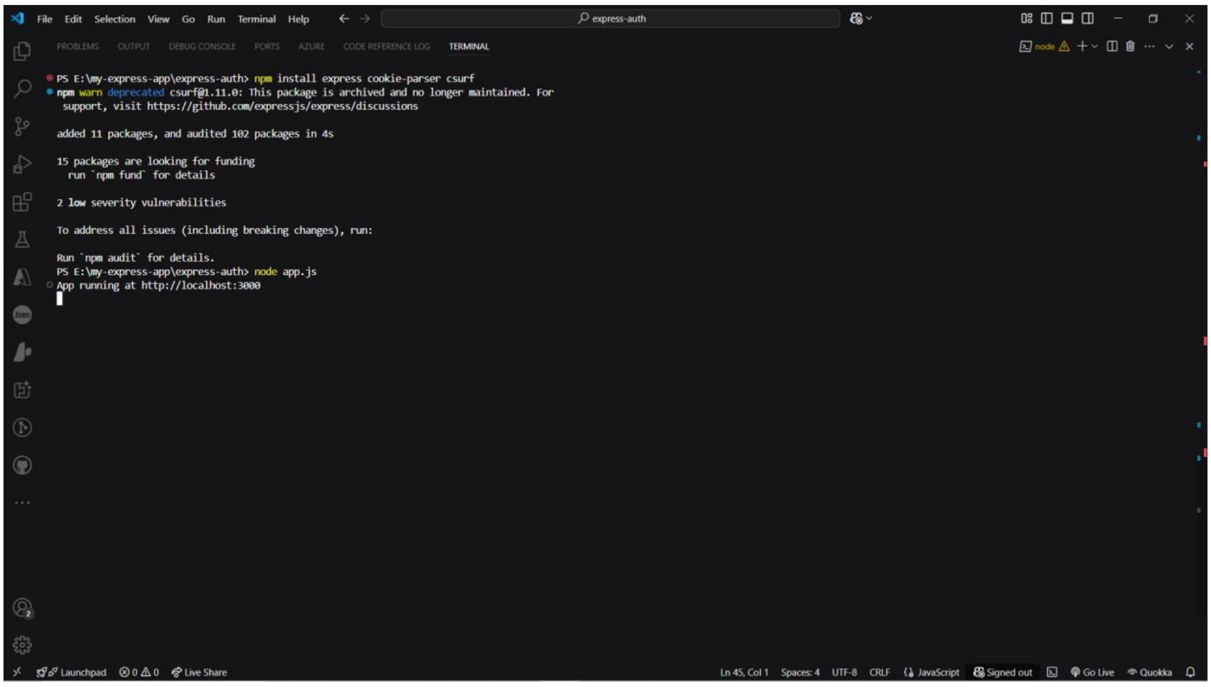

Full Stack Web-D Interns Le
Express.js Session Management
Hoppscotch • Open source API
localhost:3000/dashboard
localhost:3000/dashboard

Welcome back, johnDoe

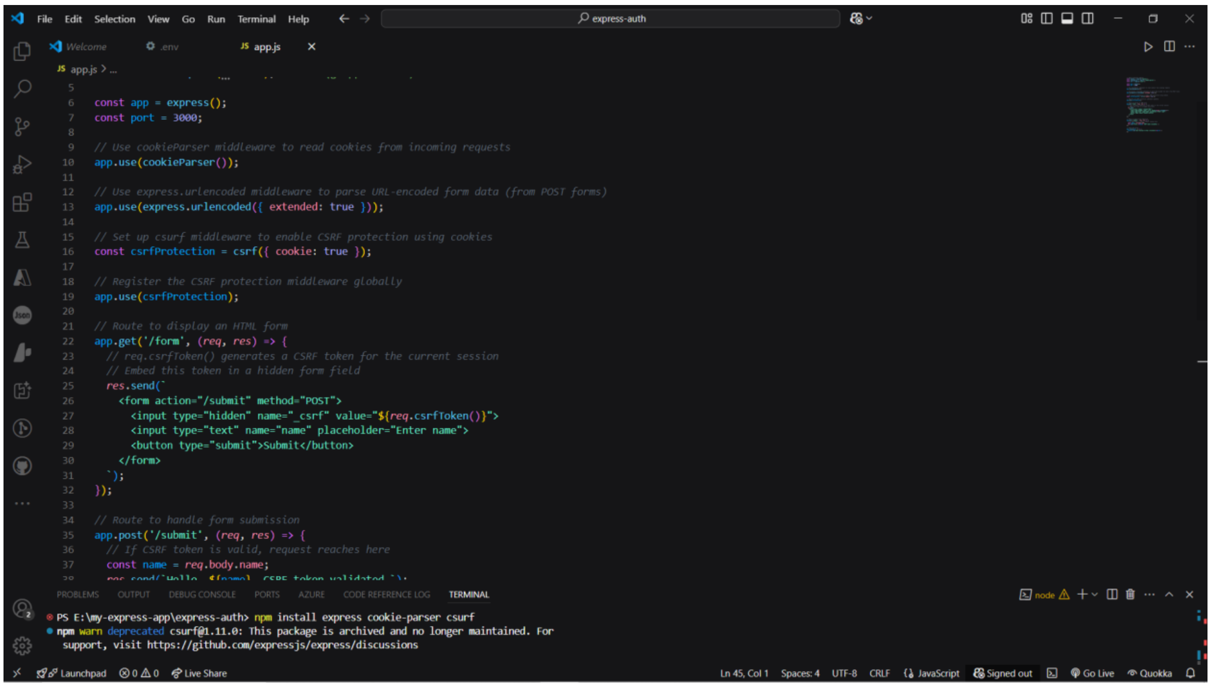

Full Stack Web-D Interns Le
Express.js Session Management
Hoppscotch • Open source API
localhost:3000/logout
localhost:3000/logout

You have been logged out.

## CSRF

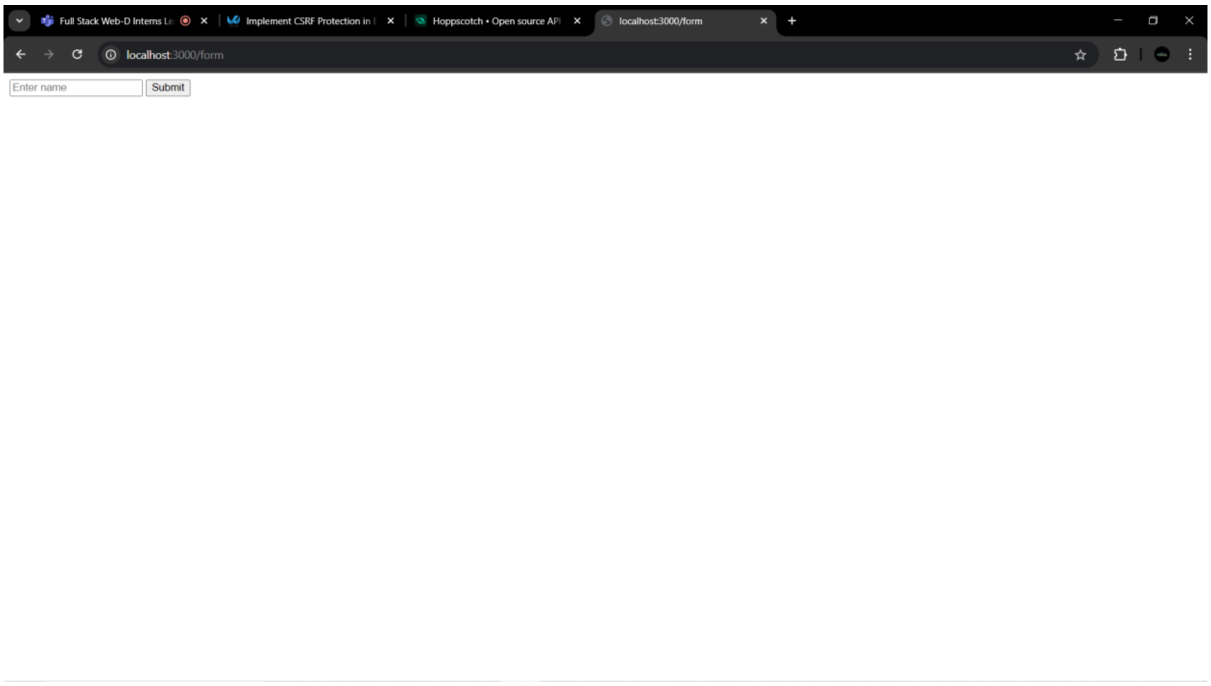


```
PS E:\my-express-app\express-auth> npm install express cookie-parser csurf
npm warn deprecated csurf@1.11.0: This package is archived and no longer maintained. For
 support, visit https://github.com/expressjs/express/discussions

added 11 packages, and audited 102 packages in 4s

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

2 low severity vulnerabilities

To address all issues (including breaking changes), run:

Run `npm audit` for details.
PS E:\my-express-app\express-auth> node app.js
App running at http://localhost:3000
```

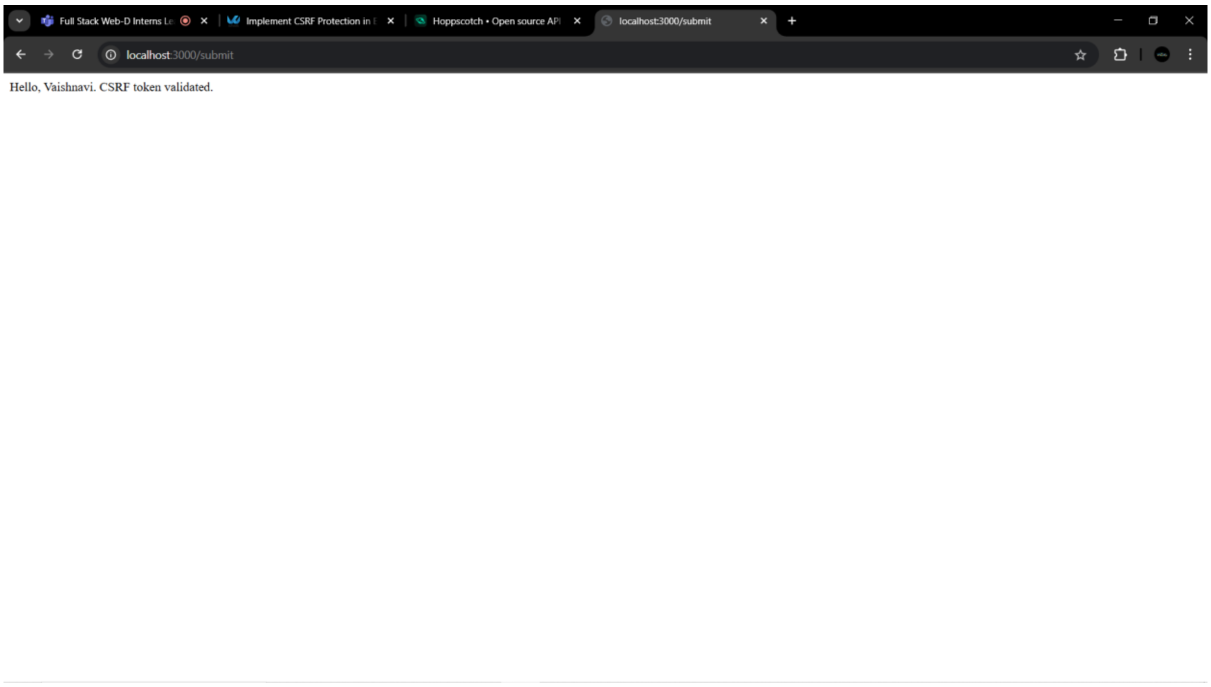


```javascript

const app = express();
const port = 3000;

// Use cookieParser middleware to read cookies from incoming requests
app.use(cookieParser());

// Use express.urlencoded middleware to parse URL-encoded form data (from POST forms)
app.use(express.urlencoded({ extended: true }));

// Set up csurf middleware to enable CSRF protection using cookies
const csrfProtection = csrf({ cookie: true });

// Register the CSRF protection middleware globally
app.use(csrfProtection);

// Route to display an HTML form
app.get('/form', (req, res) => {
  // req.csrfToken() generates a CSRF token for the current session
  // Embed this token in a hidden form field
  res.send(`
    <form action="/submit" method="POST">
      <input type="hidden" name="_csrf" value="${req.csrfToken()}">
      <input type="text" name="name" placeholder="Enter name">
      <button type="submit">Submit</button>
    </form>
  `);
});

// Route to handle form submission
app.post('/submit', (req, res) => {
  // If CSRF token is valid, request reaches here
  const name = req.body.name;
```

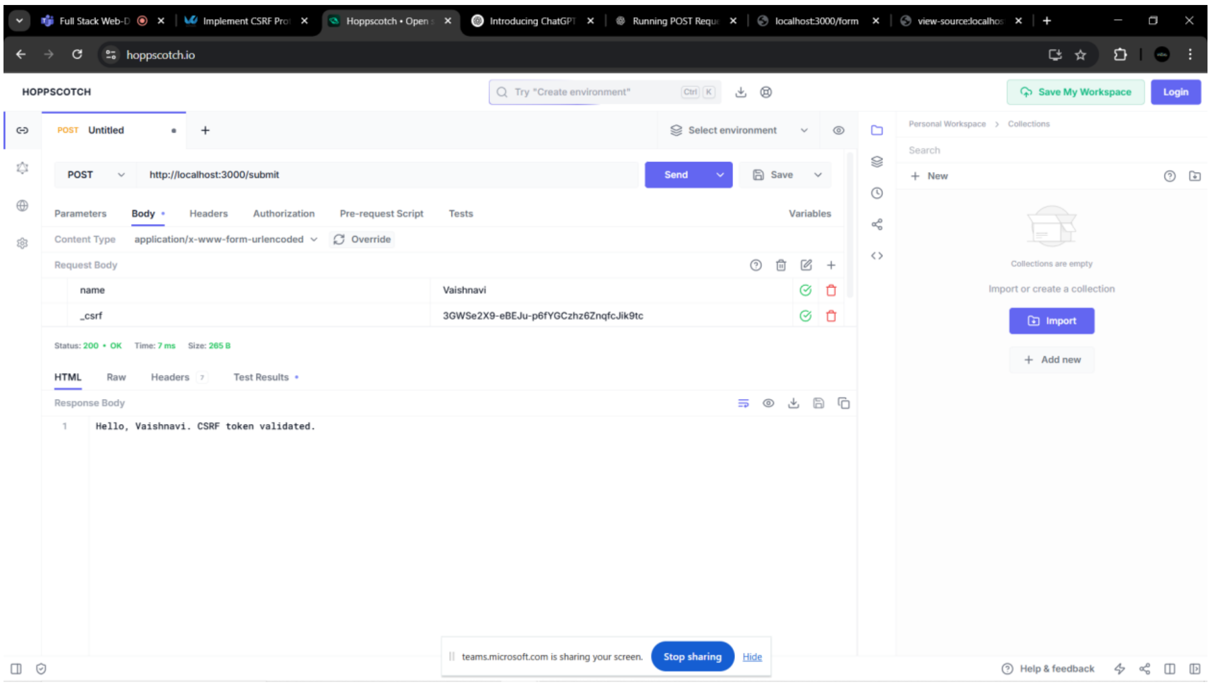
```
PS E:\my-express-app\express-auth> npm install express cookie-parser csurf
npm warn deprecated csurf@1.11.0: This package is archived and no longer maintained. For
 support, visit https://github.com/expressjs/express/discussions
```


Full Stack Web-D Interns Le
Implement CSRF Protection in
Hoppscotch • Open source API
localhost:3000/form
localhost:3000/form
Enter name
Submit

Enter the name – Vaishnavi


Full Stack Web-D Interns Le
Implement CSRF Protection in
Hoppscotch • Open source API
localhost:3000/submit
localhost:3000/submit

```
Hello, Vaishnavi. CSRF token validated.
```



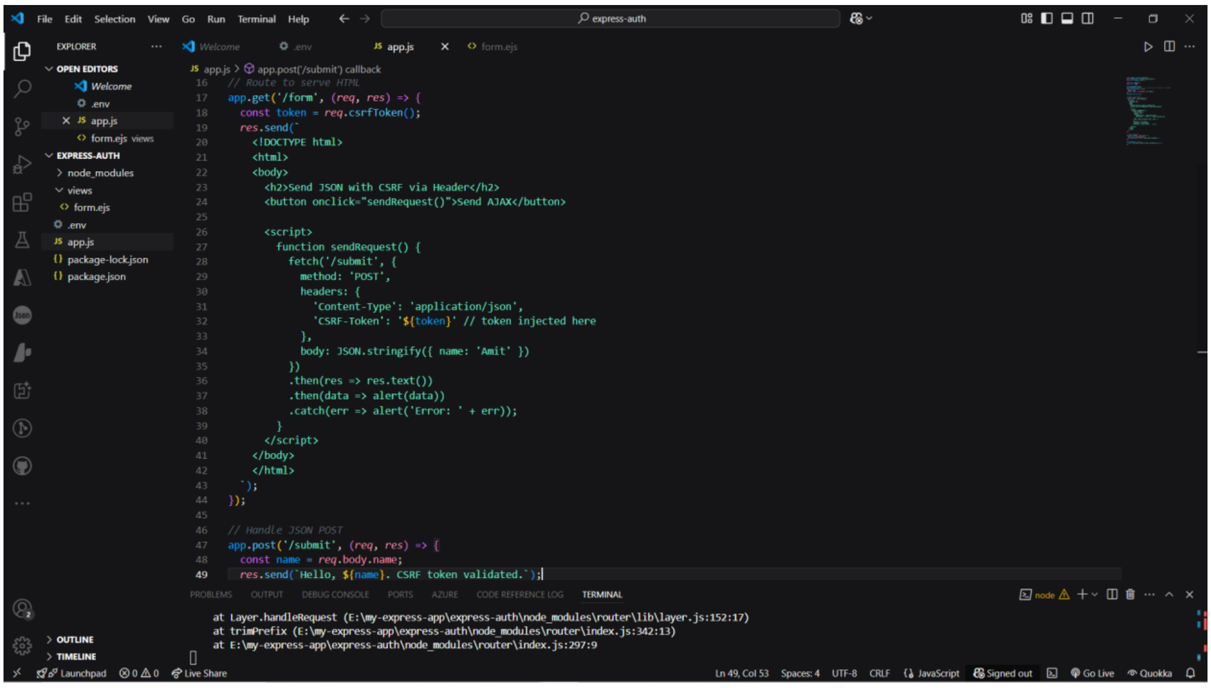
POST http://localhost:3000/submit

Content Type application/x-www-form-urlencoded

| | |
|---|---|
| name | Vaishnavi |
| _csrf | 3GWSe2X9-eBEJu-p6fYGCzhz6ZnqfcJik9tc |

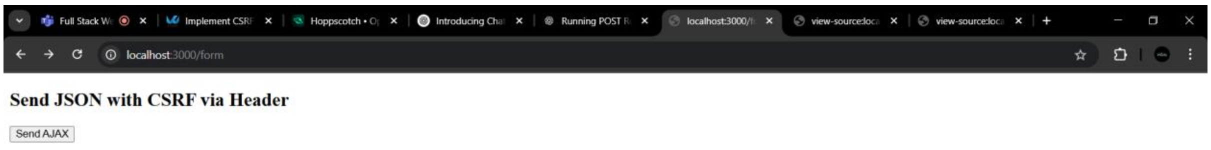
Status: 200 • OK Time: 7 ms Size: 265 B

```
Hello, Vaishnavi. CSRF token validated.
```

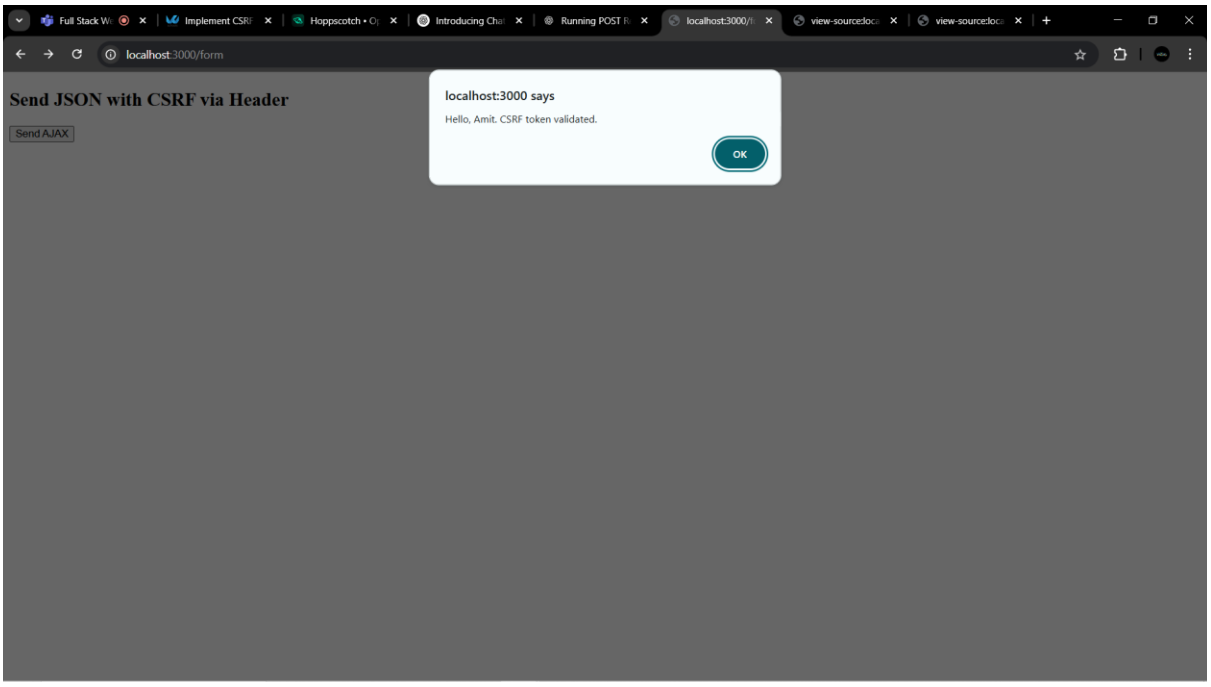


```
// Route to serve HTML
app.get('/form', (req, res) => {
  const token = req.csrfToken();
  res.send(`
    <!DOCTYPE html>
    <html>
    <body>
      <h2>Send JSON with CSRF via Header</h2>
      <button onclick="sendRequest()">Send AJAX</button>

      <script>
        function sendRequest() {
          fetch('/submit', {
            method: 'POST',
            headers: {
              'Content-Type': 'application/json',
              'CSRF-Token': '${token}' // token injected here
            },
            body: JSON.stringify({ name: 'Amit' })
          })
          .then(res => res.text())
          .then(data => alert(data))
          .catch(err => alert('Error: ' + err));
        }
      </script>
    </body>
    </html>
  `);
});

// Handle JSON POST
app.post('/submit', (req, res) => {
  const name = req.body.name;
  res.send(`Hello, ${name}. CSRF token validated.`);
```

```
    at Layer.handleRequest (E:\my-express-app\express-auth\node_modules\router\lib\layer.js:152:17)
    at trimPrefix (E:\my-express-app\express-auth\node_modules\router\index.js:342:13)
    at E:\my-express-app\express-auth\node_modules\router\index.js:297:9
```



Send JSON with CSRF via Header



Send JSON with CSRF via Header

localhost:3000 says

Hello, Amit. CSRF token validated.

## SESSION 2(4:30 – 6:30)

Working on the project :- book-library-system