HELMET JS

What is Helmet.js?

Helmet.js is a Node.js middleware that secures your Express apps by setting various HTTP headers.

It helps prevent well-known web vulnerabilities by configuring HTTP headers appropriately.

```
const helmet = require('helmet');
const express = require('express');
const app = express();

app.use(helmet());
```

Key Features of Helmet.js

- DNS Prefetching Control: Disables browser DNS prefetching for security.
- HSTS: Enforces secure (HTTPS) connections to the server.
- XSS Filter: Enables Cross-Site Scripting (XSS) filter in browsers.
- Content Security Policy: Prevents cross-site scripting attacks by controlling resources the client can load.

```
app.use(helmet.hsts());
app.use(helmet.noSniff());
app.use(helmet.xssFilter());
```

How to Use Helmet.js in Your App

Install Helmet.js via npm. Step 2: Use it in your Express app by calling app.use(helmet()).

```
const express = require('express');
const helmet = require('helmet');
const app = express();

app.use(helmet());
```

Helmet.js is easy to integrate and requires minimal configuration.

Configuring Helmet.js for Specific Needs

Customization: Helmet.js allows you to configure each security middleware according to your app's needs.

```
app.use(helmet({
  contentSecurityPolicy: {
    directives: {
      defaultSrc: ["'self'"],
      scriptSrc: ["'self'", "'trusted-cdn.com'"]
    }
  },
  hsts: false,  // Disable HSTS
}));
```

Customization gives you flexibility in applying security headers based on

Common Use Cases for Helmet.js

- Preventing Clickjacking: Helmet's frameguard middleware helps prevent clickjacking attacks.
- Securing Sensitive Information: noSniff helps prevent browsers from trying to guess (sniff) the MIME type.
- Improving HTTPS Security: Enforce HTTPS connections using HSTS with Helmet.

```
app.use(helmet.frameguard({ action: 'deny' }));
app.use(helmet.noSniff());
app.use(helmet.hsts({ maxAge: 31536000 }));
```