Sawport Webview Widget Angular Documentation

Overview

The **Sawport Webview Widget** is an Angular component that securely embeds the Sawport web application inside an iframe. This widget uses JSON Web Tokens (JWT) to authenticate users and grant access to the webview.

Features

- Secure authentication using JWT.
- Seamless embedding using an `iframe`.
- Supports camera, microphone, and screen sharing.
- Lightweight and easy to integrate into Angular projects.

Prerequisites

Before integrating the widget, ensure you have:

1. **Angular 15+** installed.
2. **NPM package** `jose` for JWT signing:

   ```
   npm install jose
   ```

3. **A valid** `WEBVIEW_JWT_KEY` for signing JWTs securely.

Installation

Step 1: Create the Component

Create a new component called `WebWidgetComponent` and paste the following code:

```
import { Component, Input, OnInit } from '@angular/core';
import { SignJWT } from 'jose';
import { CommonModule } from '@angular/common';
import { DomSanitizer, SafeResourceUrl } from '@angular/platform-browser';

const WEBVIEW_URL = 'https://mobile.sawport.com';
const WEBVIEW_JWT_KEY = 'WEBVIEW_JWT_KEY';
const WEBVIEW_JWT_ALG = 'HS256';

@Component({
  selector: 'app-web-widget',
  standalone: true,
  imports: [CommonModule],
  template: `
    <iframe
      *ngIf="safeUrl"
      [src]="safeUrl"
      width="350px"
      height="650px"
      frameborder="0"
      allowfullscreen
      allow="camera; microphone; fullscreen; display-capture"
    ></iframe>
  `,
  styles: [
    `
      iframe {
        border: none;
        width: 350px;
        height: 650px;
      }
```

```
    `,
  ],
})
export class WebWidgetComponent implements OnInit {
  @Input() user!: { email: string; mobile: string; name: string };
  signedParams: string = '';
  safeUrl: SafeResourceUrl | null = null;
  WEBVIEW_URL = WEBVIEW_URL;
  WEBVIEW_JWT_KEY = WEBVIEW_JWT_KEY; // Use an environment variable in production
  WEBVIEW_JWT_ALG = WEBVIEW_JWT_ALG; // Algorithm

  constructor(private sanitizer: DomSanitizer) {} // Inject DomSanitizer

  async signParams() {
    try {
      const secret = new TextEncoder().encode(this.WEBVIEW_JWT_KEY);
      const token = await new SignJWT({
        email: this.user?.email,
        mobile: this.user?.mobile,
        name: this.user?.name,
      })
        .setProtectedHeader({ alg: this.WEBVIEW_JWT_ALG })
        .sign(secret);

      this.signedParams = token;
      this.safeUrl = this.sanitizer.bypassSecurityTrustResourceUrl(
        `${this.WEBVIEW_URL}?token=${this.signedParams}`
      );
    } catch (error) {
      console.error('Error signing JWT:', error);
      this.signedParams = '';
      this.safeUrl = null;
    }
  }

  ngOnInit() {
    this.signParams();
  }
}
```

Step 2: Use the Component

Add the `app-web-widget` selector inside your main application template:

```
<app-web-widget [user]="{ email: 'user@example.com', mobile: '+1234567890', name: 'John Doe' }"></app-web-widget>
```

Security Considerations

- **Use environment variables** for `WEBVIEW_JWT_KEY` instead of hardcoding it.
- **Restrict JWT expiration time** for security.
- **Ensure the webview URL (`WEBVIEW_URL`) is trusted** before embedding.

Troubleshooting

Issue: NG0904: Unsafe value used in a resource URL context

Solution: Ensure you are using Angular's `DomSanitizer` to trust the resource URL.

Issue: JWT not signing correctly

Solution: Check if the `WEBVIEW_JWT_KEY` is correctly set and is a valid secret.

Conclusion

The **Sawport Webview Widget** enables seamless authentication and embedding of the Sawport web app in Angular applications. By using JWT for authentication and Angular's `DomSanitizer` for security, this component ensures a smooth and secure experience.

For further assistance, contact the **Sawport Development Team**.