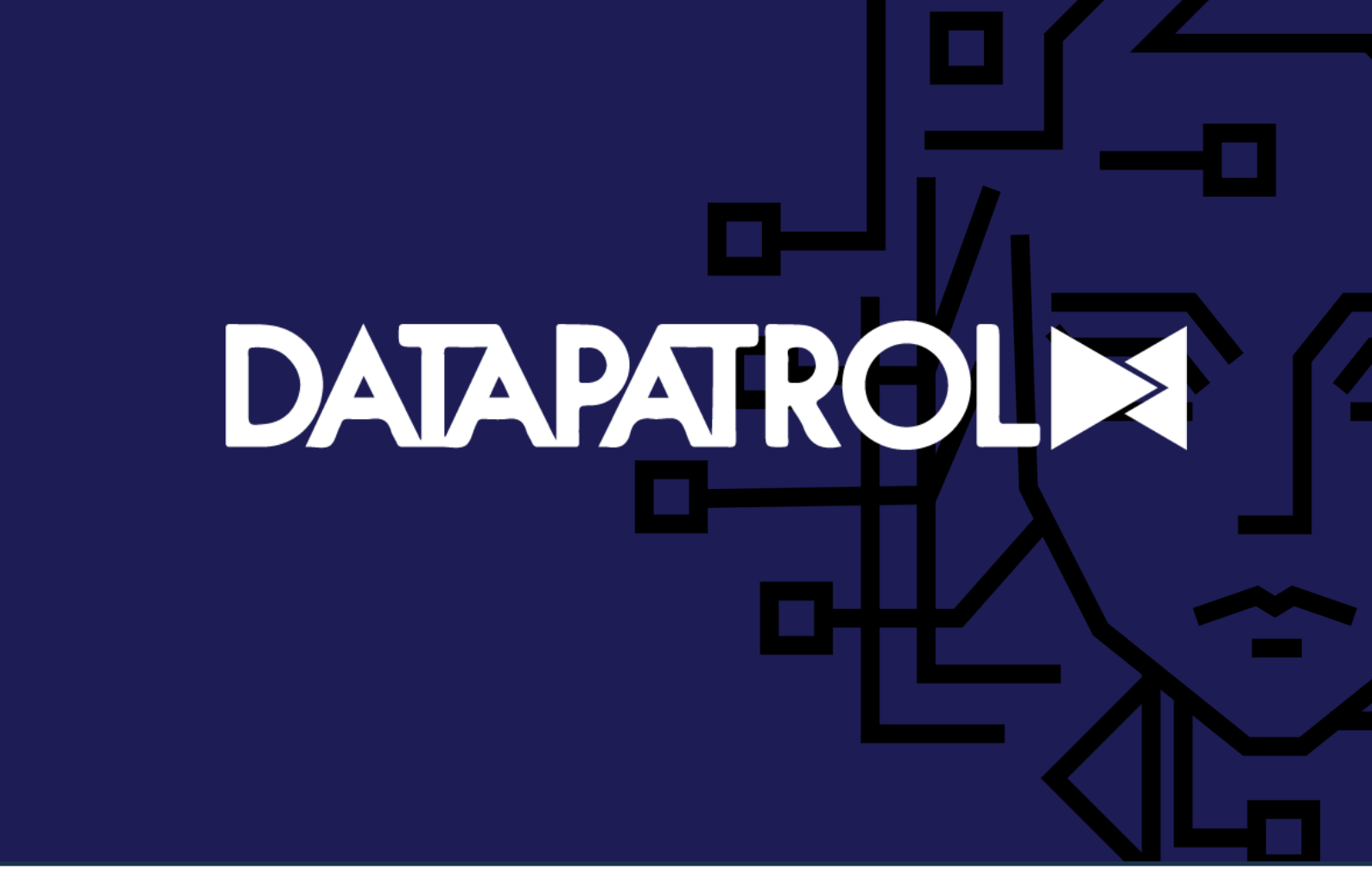


WebMark Integration Guide :

React & Angular

April 2025

1 Introduction

WebMark is a lightweight JavaScript SDK that integrates directly into your application's source code, regardless of the framework (React, Angular, Vue, etc.). It enables seamless watermark-based security for internal portals, helping companies protect sensitive information from screen capture.

This guide provides step-by-step instructions for integrating WebMark into applications built with React Library and Angular framework.

2 React Integration

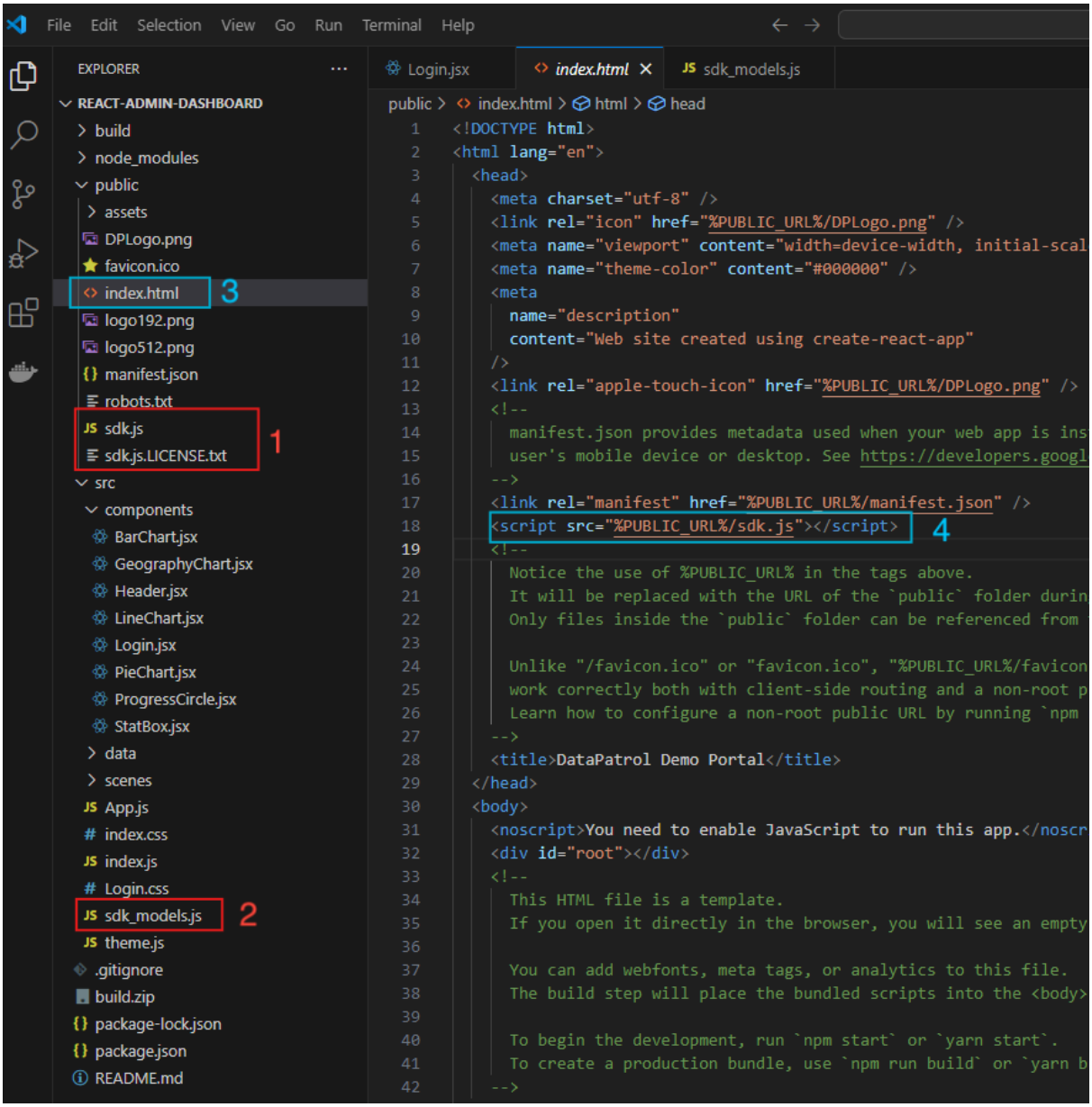
2.1 Initialization

During the implementation, 3 files will be provided. WebMark_SDK.js, sdk.js.LICENSE.txt and sdk_models.js. To integrate the SDK into your project, follow these steps:

- **Placement of the SDK:** Begin by placing the obfuscated SDK file and License in the Public folder of your project. This ensures that the SDK is easily accessible to your application during runtime.
- **Placement of the Models File:** Place the sdk_models.js file anywhere inside the project (inside “src“ folder).
- **Referencing the SDK:** Next, you will need to reference the SDK within your index.html file. Open the index.html file and locate the <head></head> section.
- **Adding the Script Tag:** Insert the following script tag within the <head> section to link the SDK: <script src="%PUBLIC_URL%/sdk.js"></script>. This line of code tells the browser to load the SDK from the specified location, allowing your application to utilize its functionalities.

By following these steps, you ensure that the WebMark SDK is properly integrated into your project.

NOTE: Please refrain from modifying the files provided, as any changes will invalidate them.



2.2 Invoking the SDK

DataPatrol SDK provides two essential functions to manage watermarks in your project:

ApplyWatermark: Adds a watermark to the application interface.
RemoveWatermark: Clears the watermark, typically used during logout.

These functions ensure enhanced security by visually marking the user's session. It's best to invoke these functions at key moments:

ApplyWatermark: During app initialization (e.g., in App.js or the layout component).
RemoveWatermark: During user logout to clean up the session.

Start by importing the necessary modules and setting up constants for **token** and **appInfo**. These provide the foundation for interacting with the SDK.

Make sure to create a global variable to hold our API response for later on.

```
import {
  DataPatrolUserInfo,
  DataPatrolAppInfo,
  DataPatrolLogInfo,
  DataPatrolLogLevel,
  DataPatrolFeatureType,
  DataPatrolSecurityAlertType
} from "./sdk_models";

const token = "AD0920E1173E4E2920E111B7B9920E11C4BCD8920E115F8C50"; // Your unique token
const appInfo = new DataPatrolAppInfo(
  "229994940",          // Unique App ID provided by the customer
  "Customer A Web App", // Website Name
  "3.0.0.0"             // Website Version
);
var handler = '' // Global variable to hold the response value from our API
```

- **Apply Watermark Function:**

This function is responsible for applying a watermark to the application. This watermark ensures accountability by associating the user session with specific user details.

Note: Developers must provide the username or email of the logged-in user when invoking this function.

The ApplyWatermark function accepts the following five parameters:

  - API URL: The URL endpoint provided by the customer. This endpoint is used to retrieve the watermarking policy that governs how the watermark is applied.
  - Token: A unique authentication token supplied by DataPatrol. This token is used to validate the request and ensure only authorized users can apply the watermark.
  - User Information (UserInfo): The user object, encapsulated in a UserInfo instance, must contain the username or email of the logged-in user. This personalizes the watermark for the active session and links the user to their activity.
  - Application Information (AppInfo): This includes:
    - Unique App ID: A unique identifier for the application, provided by the customer. This helps differentiate users in the admin panel if multiple applications are being managed.

- Website Name: The name of the website or application as supplied by the customer. This is useful for distinguishing users across multiple websites.
- Website Version: The version of the application, provided by the customer, to track which deployment is being used.

- Website Behavior: If true, the website will only load when the watermark has finished loading.

```
const applyWatermark = useCallback(async (user)=>{
  const userInfo = new DataPatrolUserInfo(user);
  if(window.DataPatrolSdk){
    try {
      handler =  await window.DataPatrolWebSdk.applyWatermark(
                "http://172.178.79.50:4000/int/v1/policy",
                token,
                userInfo,
                appInfo,
                true
              );
    } catch (error) {
      //Error Handling can be added here if needed
    }
  } else {
    //Error handling can be added here if needed
  }
}, []);
```

- **Remove Watermark function:**

```
const removeWatermark = useCallback(async () => {
  if (window.DataPatrolWebSdk) {
    await window.DataPatrolWebSdk.removeWatermark(handler);
  } else {
    //Error handling can be added here if needed
  }
}, []);
```

3 Angular Integration

3.1 Initialization

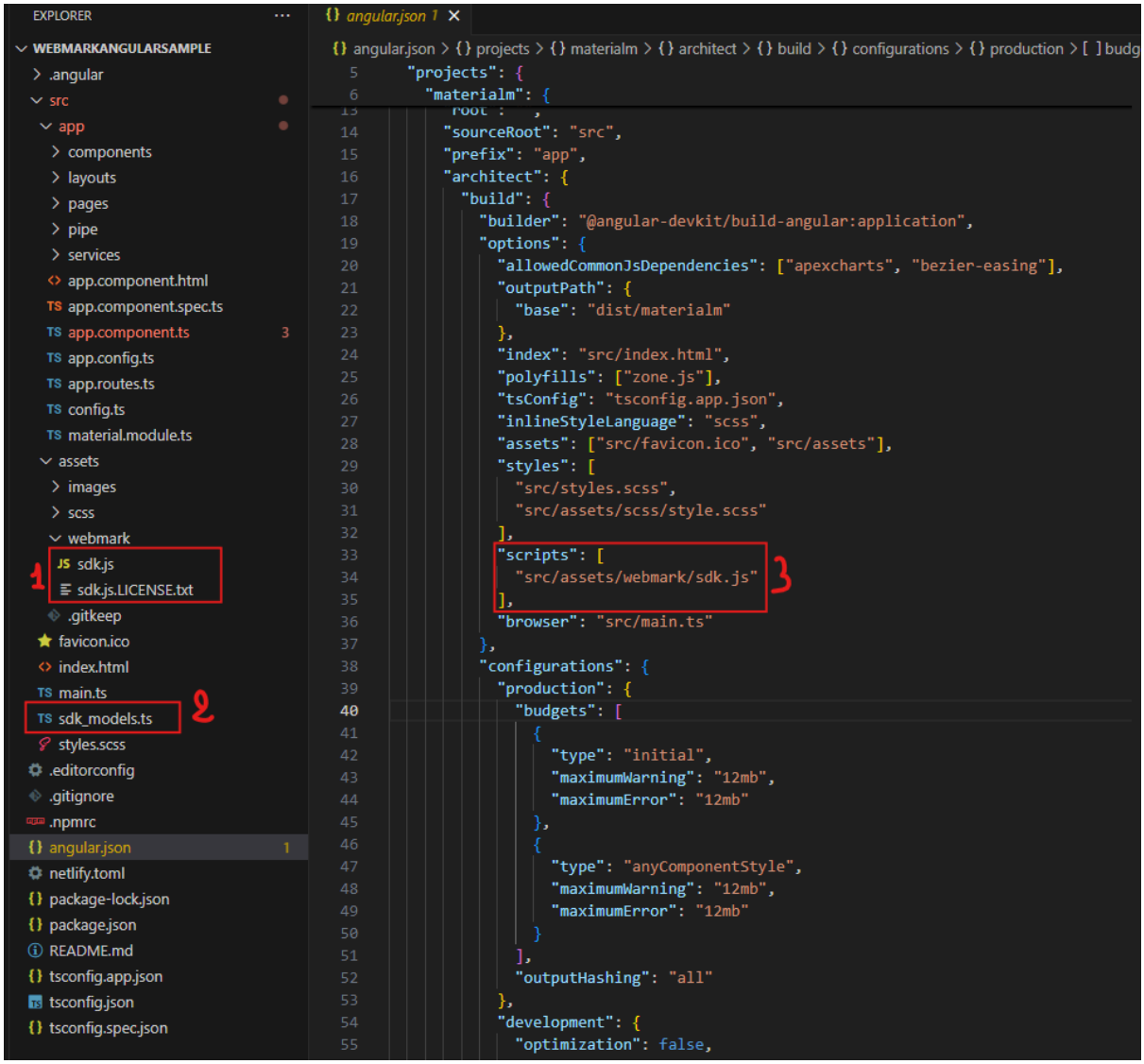
During the implementation, same as React and for every other frameworks, 3 files will be provided: WebMark_SDK.js, sdk.js.LICENSE.txt, and sdk_models.js. To integrate the SDK into your Angular project, follow these steps:

- **Placement of the SDK**: Place the obfuscated SDK file and license (WebMark_SDK.js and sdk.js.LICENSE.txt) in the src/assets folder. This ensures the SDK is accessible at runtime.
- **Placement of the Models File**: Place sdk_models.js inside the src/app folder or any preferred location within src.
- **Referencing the SDK**: Open the angular.json file. Locate the "scripts" array under the architect > build > options section of your project configuration.
- **Adding the Script Reference**: Add the path to the SDK file like this:
  "scripts": [

    "src/assets/webmark/sdk.js"

  ],
- **Using the SDK**: You can now import and use sdk_models.js within your Angular components or services.

NOTE: Do not modify any of the files provided. Altering them will invalidate the SDK and compromise its integrity.



3.2 Invoking the SDK:

Similar to React, the DataPatrol SDK offers the same functionalities to apply and retrieve watermarks.

It is recommended to place the applyWatermark function in the app.component.ts file, as it requires a user parameter and can be called during initialization.

```
export async function applyWatermarkInternal(userName: string) {
  var token: string = "AD0920E1173E4E2920E111B7B9920E11C4BCD8920E115F8C50";
  var appInfo: DataPatrolAppInfo = new DataPatrolAppInfo("229994940", "Customer A Web App", "3.0.0.0");
  var userInfo: DataPatrolUserInfo = new DataPatrolUserInfo(userName);

  if (DataPatrolWebSdk) {
    DataPatrolWebSdk.eventEmitter.on('securityAlert', (dataPatrolSecurityAlertTypeEventArgs: any) => {
      if (dataPatrolSecurityAlertTypeEventArgs.DataPatrolSecurityAlertType == DataPatrolSecurityAlertType.DevToolsDetected) {
        window.alert("DevTool detected" + dataPatrolSecurityAlertTypeEventArgs.DataPatrolSecurityAlertType);
      } else if (dataPatrolSecurityAlertTypeEventArgs.DataPatrolSecurityAlertType === DataPatrolSecurityAlertType.AdBlockerDetected) {
        window.alert("AdBlocker detected" + dataPatrolSecurityAlertTypeEventArgs.DataPatrolSecurityAlertType);
      }
    });

    try {
      AppComponent.handler = await DataPatrolWebSdk.applyWatermark("https://dp-agent.datapatrol.local:447/int/v1/policy", token, userInfo, appInfo);
    } catch (error) {
      console.error("Catch " + error);
    }
  } else {
    window.alert("SDK not Loaded");
  }
}
```

The response from the applyWatermark function must be stored in a handler to enable watermark removal later. In this case, the handler is assigned to a static property handler of the AppComponent class. This makes it accessible throughout the application for managing the watermark lifecycle.

3.3 Removing the Watermark:

In the sign-out process, the watermark is removed using the removeWatermark function from the DataPatrolWebSdk. This function is called with the handler that was previously stored in the AppComponent class during the watermark application. Before attempting to remove the watermark, the code checks if the SDK is loaded.

```
async signout() {
  localStorage.removeItem("userName");

  if (DataPatrolWebSdk) {
    await DataPatrolWebSdk.removeWatermark(AppComponent.handler);
  } else {
    window.alert("SDK not Loaded");
  }

  this.router.navigate(['authentication/login']);
}
```

4 Security Measures and Useful Information

DataPatrol WebMark is designed to be irremovable by end users, with several protective measures in place:

- **DevTools Locking:** Access to DevTools is fully restricted across all browsers, preventing use of shortcuts or manual access.
- **AdBlock Detection:** The website detects adblockers instantly, requiring users to disable them to proceed.
- **Watermark Layer Protection:** If DevTools access is achieved, the watermark's tag and styling change dynamically, with continuous verification to prevent tampering.
- **Policy Updates:** Any changes to watermark policies are automatically applied upon page reload or navigation.
- **Number of Watermarks:** Up to four distinct text watermarks can be applied for enhanced security.
- **API URL:**
    - For the policy, the API URL must have this format: http://your-api/int/v1/policy.
    - For the Logs, the API URL must have this format: http://your-api/int/v1/log.