How to get started with a private deployment

Any company can create a Genesys Cloud Embedded Framework integration for private use in their Genesys Cloud organization.

- Development
- Deployment
- Permissions
- Genesys Cloud organization settings

Development

In your code, use exposed methods to tell Genesys Cloud what to do for specific events and actions. For example, if agents will write interaction logs, tell Genesys Cloud what do with the interaction log information and where to associate it.

1. Create a framework.js file that implements the methods and actions that you want to use.

   When you create the framework.js file, include only the methods and actions that you want to use in your implementation of Genesys Cloud Embeddable Framework. If you include methods and actions that you do not use in your implementation, the framework could produce unwanted behavior. For more information, see Configuration and methods and Actions.

2. (Optional) If you want to implement the `getAuthToken` method, your administrator must create an OAuth client in your Genesys Cloud organization.

   This OAuth client authenticates your external system with Genesys Cloud. You will use the Client ID from this OAuth client in step 3 under **Development**. For more information, see getAuthToken and the Genesys Cloud Embeddable Framework tab in Create an OAuth client (Resource Center).

3. (Optional) Include the **Client ID** from your OAuth client.

   a. Get the **Client ID** from the OAuth client that your administrator created in your Genesys Cloud organization.
   b. Add the **Client ID** to the appropriate Genesys Cloud region under the `clientIds` configuration in your framework.js file.

   For example, if you create an OAuth client in a North American Genesys Cloud organization, then in your framework.js file, add the **Client ID** as the value for `mypurecloud.com`. For more information, see clientIds.

4. Create an iframe and add attributes that grant the iframe access to cameras, microphones, and autoplay.

   > **Note**: For WebRTC phones to function correctly, you must add the recommended attributes to your iframe.

   ```
   <iframe allow="camera *; microphone *; autoplay *" src="https://apps.mypurecloud.com/crm/index.html?crm=framework-l
   ```

5. (Optional) Create a second iframe to embed the Interactions window.

   The Interactions window contains scripts or conversations for digital interactions (such as chat, message, or email interactions). By default, the Interactions window is a separate pop-up window that opens when agents click **Chat**, **Script**, and other icons on interactions in the client. With a second iframe in your web app and configuration in your framework.js file, you can set this Interactions window to be an embedded window rather than a separate pop-up window.

   ```
   <iframe allow="camera *; microphone *; autoplay *" src="https://apps.mypurecloud.com/crm/interaction.html"></iframe
   ```

   For more information, see embeddedInteractionWindow on the settings page and the category `interactionWindow` for Notification event on the subscribe page.

Deployment

When you are ready to release your implementation, update the URLs and upload the framework.js file.

1. Update the initial required iframe URL in your code so the URL no longer points to your local host. See step 4 under **Development**.

   ```
   <iframe allow="camera *; microphone *; autoplay *" src="https://apps.mypurecloud.com/crm/embeddableFramework.html">
   ```

   > **Important**: If you implemented any of the following properties, add the corresponding query parameter to the updated iframe URL.

Property	Query parameter	Notes
clientId	`?enableFrameworkClientId=true`	See step 2 under **Development**.
dedicatedLoginWindow	`?dedicatedLoginWindow=true`	See settings.
provider	`?provider={single sign-on identity provider name}`	See sso object.
orgName	`?org={Genesys Cloud org name}`	If you use the `orgName` query parameter, you must also use the `provider` query parameter. See sso object.

2. Give your administrator the updated URL to add to the **Authorized redirect URIs** field in the OAuth client in your Genesys Cloud organization. See step 2 under **Development**.

3. Give your administrator the framework.js file to upload to the Genesys Cloud Embeddable Framework integration in your Genesys Cloud organization.

   For more information, see Add a Private Genesys Cloud Embeddable Framework integration (Resource Center).

   Administrators can replace uploaded framework.js files at any time. They can also add a maximum of five different implementations of the integration to a single Genesys Cloud organization. If they add more than one Private Genesys Cloud Embeddable Framework integration, when users log in or clear their cache, the integration prompts them to select which implementation of the integration to use.

   > **Tip**: You could use one implementation as a development environment and another as a production environment. Or you can create different implementations (with different framework.js files) for different third-party products, such as one implementation for Salesforce and another implementation for Zendesk.

Permissions

Your administrator needs to grant permissions in your Genesys Cloud organization to agents who will use the Genesys Cloud Embeddable Framework integration. For a list of the permissions, see Administrator requirements for the Genesys Cloud embedded clients (Resource Center).

Genesys Cloud organization settings

If the **Restrict Genesys Cloud Embedding** setting is enabled in your Genesys Cloud organization, tell your Genesys Cloud administrator to add the parent domain to **Allowed Embeddable Domain(s)**. Otherwise, your implementation of the Genesys Cloud Embeddable Framework will fail to embed. For more information, see Manage Genesys Cloud embedding with the Genesys Cloud embedded clients (Resource Center).