チャット部品組込みガイド（RAG）

はじめに

本書は、既存のHTMLにチャット部品(以降、「チャットUI」と記載)を組み込むための手順を示したガイドです。
チャット部品を用いることで、Webブラウザでチャットを行うための部品を組み込み、コーディングレスで本サービスが提供するAPIを利用できます。

本書の対象読者は以下を想定しています。

- 自製のWebサイトにGenerative AI FWと連携するチャットボット機能を追加したいWebシステム開発者

> - 本ガイドに記載する「https://<サーバのドメイン名>/」はGenerative AI FWがインストールされているサーバのドメイン名に置き換えてAPIを実行してください。
> - 本サービスではHTTPSに既定では自己証明書を使用しています。そのためAPI利用時に考慮が必要です。詳細は「スタートアップマニュアル（導入準備編）」をご確認ください。

本ガイドの流れ

本書は、事前にインデックスに登録した文書に基づいた会話(RAG)を行うチャットUIを、既存のWebサイトへの組み込む手順を説明します。
以下の手順に沿って説明します。

1. 前提条件の確認
2. WebバックエンドにAPI Keyを返却するAPIを実装
3. Webフロントエンドへのチャットの組み込み

本ガイドを実施することで下記の画面のようにチャットUIをWebサイトへ組み込むことができます。

チャットUIの組み込みイメージ

前提条件の確認

認証情報の準備

組み込むにあたり必要となる、以下の情報を準備します。

- API Key
  - 「セットアップガイド」の「APIキーの更新」にて生成されたAPI Key
- チャットボットモジュールURL
  - https://<サーバのドメイン名>/js/bundle.js

また、「管理ポータル操作ガイド（インデックス・文書管理編）」に従い、管理ポータルにアクセスし、インデックス管理の機能によりインデックスを作成、チャットUIにより検索する文書を登録してください。文書を登録したインデックス名をご準備ください。

- インデックス名

Webバックエンド

API Keyを取得する処理を組み込むためのWebバックエンドが必要です。バックエンドのない静的なWebサイトには組み込むことができません。

WebバックエンドにAPI Keyを返却するAPIを実装

連携元システムのWebバックエンドに、本サービスのAPIの呼び出しに必要なAPI Keyを返却するAPIを実装します。以下はNode.jsを用いた場合の実装例です。以降では、そのAPIのURLを

`http://localhost/token`とします。

```
const express = require('express')
const app = express()
const port = 80

app.get('/token', (req, res) => {
  res.send('bearer <API Keyの値>')
})

app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

Webフロントエンドへのチャットui組み込み

Webフロントエンドにチャットuiを組み込むために、WebページにHTMLに対して、チャットuiをロードするための<script>タグとチャットuiを表示するための タグを組み込みます。

以下はscriptタグに組み込むコード例です。`'http://localhost/token'`の箇所は「Webバックエンドにapi Keyを返却するapiを実装」の手順にて実装したapi Key取得apiのものに置き換えてください。

modelオプションで対話するllmを指定することができます。指定可能な値は、管理ポータルの基本情報ページにおける「利用可能なllm一覧」から確認することができます。

templateidオプション(任意)で、テンプレートを指定することができます。システムプロンプトを調整したい場合に本オプションを指定します。テンプレートの作成方法、および、テンプレートidの取得方法は「管理ポータル操作ガイド（テンプレート機能編)」を参照ください。

```
<head>
  . . .
  <script defer type="module">
    import {mountChatbot} from '<チャットボットモジュールURLの値>'

    fetch('http://localhost/token').then((response) => {
        return response.text()
    }).then((token) => {
      const option = {
          url: 'https://<サーバのドメイン名>/genai-api/v1/searchchat',
          model: 'cotomi-v3.0',
          index: '<インデックス名の値>',
          templateId: 'template_aaaaaaaa-bbbb-cccc-eeee-ffffffffffff'
      }
      mountChatbot('chatbot', token, option)
    })
  </script>
</head>
<body>
  . . .
  <div id="chatbot"></div>
</body>
```