目次

改訂情報

変更年月日	変更内容
2024-04-01	初版
2024-10-01	第2版 以下を追加・変更しました。 ■ 「*はじめに*」に Amazon Bedrock に関する説明を追加 ■ 「*はじめに*」に弊社検証済みモデルの表を追加 ■ 「*Amazon Bedrockのセットアップ*」を追加 ■ 「*生成AI連携ドライバ設定*」に Amazon Bedrock に関する説明を追加 ■ 「*IM-LogicDesignerタスク*」に Amazon Bedrock に関する説明を追加 ■ 「*はじめに*」-「*IM-Copilotの構成*」内にコラムを追加 ■ 「*IM-LogicDesignerタスク*」内のコラムにパラメータ制限に関する説明を追加 ■ 「*セットアップ（各種製品アシスタント）*」を追加 ■ 「*共通アシスタント実行画面*」を追加
2025-04-01	第3版 以下を追加・変更しました。 ■ 「*はじめに*」の弊社検証済みモデルの表を更新 ■ 「*生成AI連携ドライバ設定*」にデフォルトパラメータに関する説明を追加 ■ 「*生成AI連携ドライバ設定*」の APIバージョンに関する記載について、明示的な指定を推奨しない変更を実施したため、記載を削除 ■ 「*Azure OpenAI Serviceのセットアップ*」のモデルデプロイに関するコラムを更新 ■ 「*Wiki アシスタントのセットアップ*」に日本語・英語以外での利用に関する注釈を追加 ■ 「*Wiki アシスタントのセットアップ*」に Amazon Bedrock に関する説明を追加 ■ 「*Wiki アシスタントのセットアップ*」の Azure OpenAI Service に関する説明を更新 ■ 「*Wiki アシスタントのセットアップ*」のテキスト抽出設定に関する説明を更新 ■ 「*ViewCreator SQLビルダ アシスタント のセットアップ*」に Amazon Bedrock に関する説明を追加 ■ 「*Accel Studio アプリケーション作成 アシスタント のセットアップ*」を追加 ■ 「*IM-LogicDesignerタスク*」に「*ロジックフローアシスタントでの利用を想定したタスク*」を追加 ■ 「*IM-LogicDesignerタスク*」に「*ベクトルデータベース操作を想定したタスク*」を追加 ■ 「*IM-LogicDesignerタスク*」に「*データ処理を想定したタスク*」を追加 ■ 「*アシスタント定義*」を追加 ■ 「*ロジックフローアシスタントの作成例*」を追加

はじめに

IM-Copilotとは

IM-Copilotは intra-mart Accel Platform上で生成AIを利用した業務アプリケーション開発、および
intra-mart Accel Platformの各種製品で生成AIを活用するための基盤機能です。

IM-Copilotの特徴は以下の通りです。

- 生成AIサービスによる違いを利用者に意識させない汎用的なインタフェースを提供します。
- 生成AIを企業で利用するために求められる統制機能(ログ保存、不正利用の防止)を提供します。
- 業務アプリから生成AIを利用しやすいGUI開発部品、APIを提供します。
- intra-mart Accel Platformの各種製品から生成AIを利用するためのアシスタント機能を提供します。

IM-Copilotの利用者、利用方法

IM-Copilotの利用者、利用方法は下記を想定しています。

- 業務アプリから手軽に生成AIを呼び出して活用したい利用者（汎用利用者）
  - GUI開発(IM-LogicDesigner)により、プログラミング知識がなくても生成AIを利用可能です。
  - スクリプト開発、Java EE開発から汎用的なAPIを介して生成AIを利用可能です。
  - いずれの開発方法でも、利用者は生成AIサービスの違いを意識する必要はありません。
- 生成AIをより高度に活用した業務アプリを開発したい利用者（専用利用者） **（今後提供予定）**
  - 汎用利用よりも踏み込んだ生成AI活用(チューニング、特定の生成AIサービス向けの実装)が可能です。
  - 利用者は生成AIサービスの違いを意識する必要があります。

IM-Copilotの構成

IM-Copilotでは、生成AIサービスとそれを利用した機能について、以下のように構成しています。

- ドライバ
  利用可能な生成AIサービスを抽象化したものです。
  intra-mart Accel Platform 2025 Spring(Kamille)においては、以下の生成AIサービスが利用可能です。
  - OpenAI
  - Azure OpenAI Service
  - Amazon Bedrock
- アクション
  ドライバを介して生成AIサービスを呼び出し、intra-mart Accel Platform上で利用できるようにした機能です。
  スクリプト開発向けAPI、Java EE開発モデル向けAPI、および、IM-LogicDesignerタスクとして用意しています。
  intra-mart Accel Platform 2025 Spring(Kamille)においては、以下のアクションが利用可能です。
  - チャット
    生成AIと対話を行い、回答を得る機能です。
  - 画像生成
    与えたプロンプトをもとに、画像を生成する機能です。
  - 埋め込み
    与えたテキストから埋め込みベクトルを計算する機能です。
  - 文字起こし
    与えた音声ファイルをもとに、それに含まれるテキストを生成する機能です。
  - 音声生成
    与えたテキストをもとに、それを読み上げた音声を生成する機能です。

  下記の表は、各ドライバ種別と利用可能なアクションにおける、利用検証済みのモデル名を表しています。
  モデル名のうち一番先頭のものは、そのアクションにおいて用いられるデフォルトモデルです。
  ×は、そのドライバ種別においてそのアクションを利用できないことを示します。

利用可能なアクション

ドライバ種別	チャット	画像生成	埋め込み	文字起こし	音声生成
OpenAI	gpt-4o-mini gpt-4o gpt-4-turbo-2024-04-09 gpt-3.5-turbo-16k gpt-4-vision-preview	dall-e-2 dall-e-3	text-embedding-3-small text-embedding-ada-002	whisper-1	tts-1 tts-1-hd
Azure OpenAI Service	gpt-4o-mini gpt-4o gpt-35-turbo	dall-e-3	text-embedding-ada-002	whisper	×
Amazon Bedrock	anthropic.claude-3-haiku-20240307-v1:0 anthropic.claude-3-sonnet-20240229-v1:0 anthropic.claude-v2 anthropic.claude-instant-v1	stability.stable-diffusion-xl-v1	amazon.titan-embed-text-v1	×	×

コラム

Azure OpenAI Service においては、モデル名ではなくデフォルトのデプロイ名を表しています。
上記以外の名称でデプロイを行っている場合は、アクション利用時にそのデプロイ名を指定してください。

コラム

各アクションのパラメータやバイナリデータには、最小値や最大値などの制限が存在する場合があります。

コラム

デフォルトモデルの更新について:
各生成AIサービスのモデルリリースや弊社のアップデートタイミングに応じて更新される場合があります。
アップデート更新に合わせてモデルデプロイを行ってください。（推奨）
以前のモデルを利用したい場合は各アクションAPIで個別にモデルを指定してください。

コラム

intra-mart Accel Platform 2025 Spring(Kamille)より、デフォルトモデルを指定するためのパラメータ設定が追加されました。詳細は 「設定ファイルリファレンス」 - 「IM-Copilot生成AI連携ドライバ設定」 を参照してください。

- アシスタント
  生成AIサービスを活用してユーザの作業を支援する機能です。
  intra-mart Accel Platform 2025 Spring(Kamille)においては、以下のアシスタントが利用可能です。
  - Wiki アシスタント
  - ViewCreator SQLビルダ アシスタント
  - Accel Studio アプリケーション作成 アシスタント

セットアップ（ 生成AI ）

IM-Copilotで利用可能な生成AIサービスのセットアップ手順を説明します。

OpenAIのセットアップ

IM-Copilot を利用するための OpenAI のセットアップ方法について説明します。

項目

前提条件

当セットアップ手順は2024年3月末時点の OpenAI 公開情報をもとに、セットアップの主な流れを記載しています。
詳細な手順については OpenAI 側のドキュメント、および、最新情報も参照してください。

> **コラム**
>
> OpenAI ドキュメント
> https://platform.openai.com/docs

セットアップ手順

OpenAI アカウント作成

1. https://platform.openai.com/docs にアクセスしてください。
2. 右上の「Sign Up」をクリックしてください。
3. 以降は表示された画面内容に従ってアカウントを作成してください。
4. アカウント作成に成功後、ログインしてください。

APIキー作成

1. https://platform.openai.com/api-keys にアクセスしてください。
2. 以降は表示された画面内容に従って「APIキー」を作成してください。
3. 作成した「APIキー」をメモ帳などに控えてください。後の手順で利用します。

Organization ID確認

1. https://platform.openai.com/account/organization にアクセスしてください。
2. 表示されている「Organization ID」をメモ帳などに控えてください。後の手順で利用します。

課金設定（任意）

1. https://platform.openai.com/account/billing/overview にアクセスしてください。
2. 以降は表示された画面内容に従って課金設定を行ってください。

使用上限設定（任意）

1. https://platform.openai.com/usage にアクセスしてください。
2. 以降は表示された画面内容に従って使用上限設定を行ってください。

Azure OpenAI Serviceのセットアップ

IM-Copilot を利用するための Azure OpenAI Service のセットアップ方法について説明します。

前提条件

当セットアップ手順は2024年3月末時点の Azure OpenAI Service 公開情報をもとに、セットアップの主な流れを記載しています。
詳細な手順については Azure OpenAI Service 側のドキュメント、および、最新情報も参照してください。

> **コラム**
>
> Azure OpenAI Service ドキュメント
> https://learn.microsoft.com/azure/ai-services/openai

セットアップ手順

Azure OpenAI Service アカウント作成

1. https://azure.microsoft.com にアクセスしてください。
2. 「無料アカウント」をクリックしてください。
3. 「無料で始める」をクリックしてください。
4. 以降は表示された画面内容に従ってアカウントを作成してください。
5. アカウント作成に成功後、ログインしてください。

リソースグループ作成

1. Azureポータル（ https://portal.azure.com/#home ） にアクセスしてください。
2. サイドメニューから「リソースグループ」を選択してください。
3. 「作成」をクリックしてください。
4. 以降は表示された画面内容に従ってリソースグループを作成してください。

リソース作成

1. Azureポータル（ https://portal.azure.com/#home ） にアクセスしてください。
2. 「リソースの作成」をクリックしてください。
3. 検索欄に「OpenAI」を入力してください。
4. 検索結果の Azure OpenAI の「作成」をクリックしてください。
5. 以降は表示された画面内容に従ってリソースを作成してください。

6. 作成したリソースの名前（インスタンスの名前）をメモ帳などに控えてください。

> **注意**
>
> 作成中に以下メッセージが表示された場合、審査申請を行ってください。
> 「Azure OpenAIサービスへのアクセスを要求するには、ここをクリックしてください。」
>
> 審査完了後にリソース作成を再開してください。

モデルデプロイ

1. Azureポータル（ https://portal.azure.com/#home ） にアクセスしてください。
2. 先ほど作成したリソースを選択してください。
3. サイドメニューから「モデル デプロイ」を選択してください。
4. 以降は表示された画面内容に従ってモデルをデプロイしてください。

> **コラム**
>
> intra-mart Accel Platform では用途ごとに標準で以下のデプロイ名を利用します。
> 利用する用途に応じて、以下のベースモデル名・デプロイ名でモデルをデプロイしてください。
>
> - **チャット用途のモデルをデプロイする場合**
>   - ベースモデル名：gpt-4o-mini
>   - デプロイ名：gpt-4o-mini
> - **画像生成用途のモデルをデプロイする場合**
>   - ベースモデル名：dall-e-3
>   - デプロイ名：dall-e-3
> - **埋め込み用途のモデルをデプロイする場合**
>   - ベースモデル名：text-embedding-ada-002
>   - デプロイ名：text-embedding-ada-002
> - **文字起こし用途のモデルをデプロイする場合**
>   - ベースモデル名：whisper
>   - デプロイ名：whisper
>
> 尚、標準利用のデプロイ名は「IM-Copilot生成AI連携ドライバ設定」のパラメータ設定で変更可能です。

> **コラム**
>
> LogicDesignerタスク や アクション API 利用する際にモデルにデプロイ名を指定することで任意のモデルを実行可能です。

> **コラム**
>
> リソースのリージョンによって選択できないベースモデルがある場合があります。

APIキー作成

1. Azureポータル（ https://portal.azure.com/#home ） にアクセスしてください。
2. 先ほど作成したリソースを選択してください。
3. 「キーを管理するにはここをクリック」をクリックしてください。
4. 表示されている「APIキー」と「エンドポイント」をメモ帳などに控えてください。後の手順で利用します。

課金設定（任意）

1. Azureポータル（ https://portal.azure.com/#home ） にアクセスしてください。
2. 「コストの管理と請求」をクリックしてください。
3. 以降は表示された画面内容に従って課金設定を行ってください。

Amazon Bedrockのセットアップ

IM-Copilot を利用するための Amazon Bedrock のセットアップ方法について説明します。

項目

前提条件

当セットアップ手順は2024年9月末時点の Amazon Bedrock 公開情報をもとに、セットアップの主な流れを記載しています。
詳細な手順については Amazon Bedrock 側のドキュメント、および、最新情報も参照してください。

> **コラム**
>
> Amazon Bedrock User Guide
> https://docs.aws.amazon.com/bedrock/latest/userguide/what-is-bedrock.html

セットアップ手順

モデルアクセス設定

1. AWS マネジメントコンソールにログインします。
   https://aws.amazon.com/jp/console/
2. サービスより「Amazon Bedrock」を検索し、Amazon Bedrock サービスにアクセスします。
3. メニューより「モデルアクセス」ページを開きます。
4. 利用したい IM-Copilot アクションに応じて、以下のいずれかのモデルに対して「モデルアクセスをリクエスト」を行ってください。
   - チャット
     - Claude 3 Sonnet
     - Claude 3.5 Sonnet
     - Claude 3 Haiku
     - Claude
     - Claude Instant
   - 埋め込み
     - Titan Embeddings G1 - Text
   - 画像生成
     - SDXL 1.0

> **コラム**
>
> AWSサービスで選択しているリージョンによって、利用できるモデルが限定される場合があります。
> また、「モデルアクセスをリクエスト」が許可されるまでには時間がかかる場合があります。

ポリシーの作成

1. AWS マネジメントコンソールにログインします。
   https://aws.amazon.com/jp/console/
2. サービスより「IAM」を検索し、IAMサービスにアクセスします。
3. メニューより「ポリシー」ページを開き、「ポリシーの作成」をクリックします。

4. 「サービスを選択」にて「Bedrock」を検索し、選択してください。
5. 「すべての Bedrock アクション (bedrock:*)」チェックボックスをオンにします。
6. 「リソース」の「すべて」を選択し、「次へ」をクリックします。
7. 「ポリシー名」に任意のポリシー名を入力し、「ポリシーの作成」をクリックしてください。
8. ポリシーが作成されました。
   作成されたポリシー名は *アクセスキーとシークレットキーの作成* や *ロールの作成* に使用しますので、控えておいてください。

認証方式の選択

IM-Copilot から Amazon Bedrock サービスへ接続する際の認証方式として、以下の3通りが利用可能です。
以下、ご利用の認証方式に合わせてセットアップしてください。

- *認証情報ファイルを利用して認証する場合*
- *設定ファイルに直接記述したアクセスキーとシークレットキーを利用して認証する場合*
- *インスタンスプロファイルを利用して認証する場合*（※Amazon EC2 インスタンス上で iAP が動作している場合のみ）

> **コラム**
>
> 各認証方式については、AWSのドキュメントも参考にしてください。
>
> - 設定ファイルと認証情報ファイルの設定
>
> https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-files.html
>
> - IAM ユーザーのアクセスキーの管理
>
> https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html
>
> - インスタンスプロファイルの使用
>
> https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2_instance-profiles.html

認証情報ファイルを利用して認証する場合

ユーザーの作成

1. IAMサービスにアクセスします。
2. メニューより「ユーザー」ページを開き、「ユーザーの作成」をクリックします。
3. 「ユーザー名」に任意のユーザー名を入力し、「次へ」をクリックしてください。
4. 「許可のオプション」にて「ポリシーを直接アタッチする」を選択してください。
   その後、 *ポリシーの作成* で作成したポリシー名にチェックし、「次へ」をクリックしてください。
5. 「ユーザーの作成」をクリックしてください。
6. ユーザーが作成されました。

アクセスキーとシークレットキーの作成

1. IAMサービスにアクセスします。
2. メニューより「ユーザー」ページを開き、*ユーザーの作成* で作成したユーザー名をクリックしてください。
3. 「セキュリティ認証情報」タブをクリックし、「アクセスキー」にて「アクセスキーを作成」をクリックしてください。
4. 「コマンドラインインターフェイス (CLI)」を選択し、「上記のレコメンデーションを理解し、アクセスキーを作成します。」にチェックを入れ、「次へ」をクリックしてください。
5. 「アクセスキーを作成」をクリックしてください。
6. 「アクセスキーを取得」画面でアクセスキーとシークレットアクセスキーを控えるか、「.csv ファイルをダウンロード」をクリックしてください。その後、「完了」をクリックしてください。
7. アクセスキーとシークレットキーは、認証情報ファイルを利用して認証する場合には、*認証情報ファイルの作成* で使用します。
   設定ファイルに直接記述したアクセスキーとシークレットキーを利用して認証する場合には、*生成AI連携ドライバ設定* で使用します。

認証情報ファイルの作成

認証情報ファイルを利用して認証する場合、 iAP が動作している環境のファイルシステム内に、以下のような方法を用いて認証情報ファイル（credentials）を作成する必要があります。

- AWS コマンドラインインターフェイス (AWS CLI) を利用して作成
- 直接ファイルを用意し、内容を記述して作成

これらの詳細な手順については、AWSが提供するドキュメントを参照してください。
この過程において、*アクセスキーとシークレットキーの作成* で作成したアクセスキーとシークレットキーが要求されます。
また、指定したプロファイル名は、*生成AI連携ドライバ設定* で使用します。

設定ファイルに直接記述したアクセスキーとシークレットキーを利用して認証する場合

ユーザーの作成

1. IAMサービスにアクセスします。
2. メニューより「ユーザー」ページを開き、「ユーザーの作成」をクリックします。
3. 「ユーザー名」に任意のユーザー名を入力し、「次へ」をクリックしてください。
4. 「許可のオプション」にて「ポリシーを直接アタッチする」を選択してください。
   その後、 *ポリシーの作成* で作成したポリシー名にチェックし、「次へ」をクリックしてください。
5. 「ユーザーの作成」をクリックしてください。
6. ユーザーが作成されました。

アクセスキーとシークレットキーの作成

1. IAMサービスにアクセスします。
2. メニューより「ユーザー」ページを開き、*ユーザーの作成* で作成したユーザー名をクリックしてください。
3. 「セキュリティ認証情報」タブをクリックし、「アクセスキー」にて「アクセスキーを作成」をクリックしてください。
4. 「コマンドラインインターフェイス (CLI)」を選択し、「上記のレコメンデーションを理解し、アクセスキーを作成します。」にチェックを入れ、「次へ」をクリックしてください。
5. 「アクセスキーを作成」をクリックしてください。
6. 「アクセスキーを取得」画面でアクセスキーとシークレットアクセスキーを控えるか、「.csv ファイルをダウンロード」をクリックしてください。その後、「完了」をクリックしてください。
7. アクセスキーとシークレットキーは、認証情報ファイルを利用して認証する場合には、*認証情報ファイルの作成* で使用します。
   設定ファイルに直接記述したアクセスキーとシークレットキーを利用して認証する場合には、*生成AI連携ドライバ設定* で使用します。

インスタンスプロファイルを利用して認証する場合

EC2インスタンスにアタッチされているロールの確認

この作業は、利用している iAP が Amazon EC2 インスタンス上で動作していることを前提としています。

1. AWS マネジメントコンソールにログインします。
   https://aws.amazon.com/jp/console/
2. サービスより「EC2」を検索し、EC2サービスにアクセスします。
3. メニューより「インスタンス」ページを開き、利用しているインスタンスをクリックします。
4. 「セキュリティ」タブをクリックし、「IAM ロール」に表示されているIAMロールをクリックします。
   もし「IAM ロール」にロールが表示されていない場合は、「アクション」＞「セキュリティ」＞「IAM ロールの変更」より、EC2インスタンスの利用用途に応じたIAMロールを割り当ててください。
5. IAMロールの詳細情報が表示されます。
   表示されている「ARN」の文字列を控えておいてください。

ロールの作成

この作業は、利用している iAP が Amazon EC2 インスタンス上で動作していることを前提としています。

1. IAMサービスにアクセスします。
2. メニューより「ロール」ページを開き、「ロールを作成」をクリックしてください。
3. 「信頼されたエンティティタイプ」にて「カスタム信頼ポリシー」を選択し、「カスタム信頼ポリシー」に以下を入力してください。
   `${EC2インスタンスにアタッチされたロールのARN}` の箇所は、*EC2インスタンスにアタッチされているロールの確認* で確認した「ARN」の文字列としてください。
   その後、「次へ」を押下してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
          "AWS": "${EC2インスタンスにアタッチされたロールのARN}"
        },
        "Action": "sts:AssumeRole"
    }
  ]
}
```

4. 「許可ポリシー」にて、 *ポリシーの作成* で作成したポリシー名にチェックを入れ、「次へ」をクリックしてください。
5. 「ロール名」に任意のロール名を入力し、「ロールを作成」をクリックしてください。
6. ロールが作成されました。
   作成されたロールのARNは、*生成AI連携ドライバ設定* で使用しますので、控えておいてください。

コラム

IM-Copilotで利用できる生成AIサービスは、今後の製品アップデートに合わせて追加を予定しています。

- Amazon Bedrockは、 IM-Copilot Amazon Bedrockドライバモジュール を導入している場合のみ利用可能です。

セットアップ（ iAP ）

IM-Copilot を利用するための intra-mart Accel Platform のセットアップ方法について説明します。

項目

前提条件

IM-Copilot は以下のエディションでご利用が可能です。

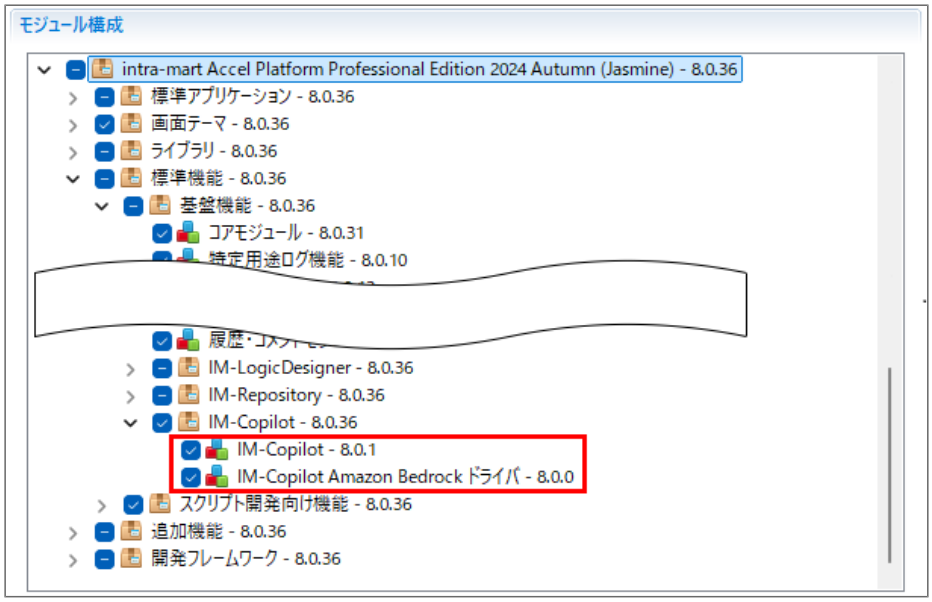
- パッケージライセンス
  - intra-mart Accel Platform Advanced Edition 2025 Spring(Kamille) 以降
- カスタマーサクセスライセンス
  - intra-mart Accel Platform Low-Code Edition 2025 Spring(Kamille) 以降
  - intra-mart Accel Platform Advance Edition 2024 Spring(Iris) 以降
  - intra-mart Accel Platform Professional Edition 2024 Spring(Iris) 以降

> **コラム**
>
> Standard Edition, Basic Edition, Pro Code Edition ではご利用いただけません。

セットアップ手順

IM-Jugglingプロジェクトの編集

1. ご利用のIM-Jugglingプロジェクトに、IM-Copilotモジュールを追加してください。
   IM-Copilotモジュールは、[標準機能 - 基盤機能 - IM-Copilot] 配下に存在します。



> **コラム**
>
> Amazon Bedrock を生成AIサービスとして利用する場合は IM-Copilot Amazon Bedrock ドライバモジュールを選択してください。

2. エラーメッセージが表示された場合は、そのメッセージをクリックし、指示に従って以下の設定ファイルを追加してください。

- IM-Copilot生成AI連携ドライバ設定
- IM-Copilot生成AI連携アクション設定

> **コラム**
>
> IM-Jugglingの使用方法の詳細については、「intra-mart Accel Platform セットアップガイド」 -「プロジェクトの作成とモジュールの選択」 を参照してください。

生成AI連携ドライバ設定

1. 生成AI連携ドライバ設定ファイルを編集します。
   「ProjectNavigator」内の <（プロジェクト名）/conf/im-copilot-driver-config.xml> ファイルをダブルクリックで開き、「ソース」タブを選択してください。

   各生成AIサービスに対しての接続に関する設定を行います。

```
<?xml version="1.0" encoding="UTF-8"?>
<im-copilot-driver-config xmlns="https://www.intra-mart.jp/im-copilot/im-copilot-driver-config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="https://www.intra-mart.jp/im-copilot/im-copilot-driver-config ../schema/im-copilot-driver-config.xsd ">
 <!-- Linkage Driver Setting for Common to Tenants -->
 <default-drivers>
  <driver type="open-ai">
   <api-key>sk-0000XXXXXX</api-key>
   <base-url>https://api.openai.com/v1</base-url>
   <organization>XXXXXXXX</organization>
   <retry-count>3</retry-count>
   <retry-wait>1</retry-wait>
   <parameters>
    <parameter>
     <parameter-name>default-chat-model</parameter-name>
     <parameter-value>gpt-4o-mini</parameter-value>
    </parameter>
    <parameter>
     <parameter-name>default-embeddings-model</parameter-name>
     <parameter-value>text-embedding-3-small</parameter-value>
    </parameter>
    <parameter>
     <parameter-name>default-images-model</parameter-name>
     <parameter-value>dall-e-2</parameter-value>
    </parameter>
    <parameter>
     <parameter-name>default-speech-model</parameter-name>
     <parameter-value>tts-1</parameter-value>
    </parameter>
    <parameter>
     <parameter-name>default-speech-voice</parameter-name>
     <parameter-value>echo</parameter-value>
    </parameter>
    <parameter>
     <parameter-name>default-transcription-model</parameter-name>
     <parameter-value>whisper-1</parameter-value>
    </parameter>
   </parameters>
  </driver>
  <driver type="azure-open-ai">
   <api-key>9999XXXXXX</api-key>
   <base-url>https://openai-service-foo.openai.azure.com/openai/</base-url>
   <retry-count>3</retry-count>
   <retry-wait>1</retry-wait>
   <parameters>
    <parameter>
     <parameter-name>default-chat-deployment-id</parameter-name>
     <parameter-value>gpt-4o-mini</parameter-value>
    </parameter>
    <parameter>
     <parameter-name>default-embeddings-deployment-id</parameter-name>
     <parameter-value>text-embedding-ada-002</parameter-value>
    </parameter>
```

```
      <parameter>
       <parameter-name>default-images-deployment-id</parameter-name>
       <parameter-value>dall-e-3</parameter-value>
      </parameter>
      <parameter>
       <parameter-name>default-transcription-deployment-id</parameter-name>
       <parameter-value>whisper</parameter-value>
      </parameter>
     </parameters>
    </driver>
    <driver type="amazon-bedrock">
     <aws-region>ap-northeast-1</aws-region>
     <aws-credentials-profile>default</aws-credentials-profile>
     <parameters>
      <parameter>
       <parameter-name>default-chat-model</parameter-name>
       <parameter-value>anthropic.claude-3-haiku-20240307-v1:0</parameter-value>
      </parameter>
      <parameter>
       <parameter-name>default-chat-max-tokens</parameter-name>
       <parameter-value>4096</parameter-value>
      </parameter>
      <parameter>
       <parameter-name>default-embeddings-model</parameter-name>
       <parameter-value>amazon.titan-embed-text-v1</parameter-value>
      </parameter>
      <parameter>
       <parameter-name>default-images-model</parameter-name>
       <parameter-value>stability.stable-diffusion-xl-v1</parameter-value>
      </parameter>
     </parameters>
    </driver>
   </default-drivers>
</im-copilot-driver-config>
```

2. 以下のドライバ設定の項目を記述してください。
   設定内容の詳細については、 「設定ファイルリファレンス」 - 「IM-Copilot生成AI連携ドライバ設定」 を参照してください。
   - OpenAI を利用する場合
     <driver>タグの type属性に、open-ai を指定してください。
     <api-key>タグ 、<base-url>タグ が必須項目です。
     必要に応じて、 <organization>タグ 、<retry-count>タグ、<retry-wait>タグを設定してください。
   - Azure OpenAI Service を利用する場合
     <driver>タグの type属性に、azure-open-ai を指定してください。
     <api-key>タグ 、<base-url>タグ が必須項目です。
     必要に応じて、 <retry-count>タグ、<retry-wait>タグを設定してください。
   - Amazon Bedrock を利用する場合
     <driver>タグの type属性に、amazon-bedrock を指定してください。
     <aws-region>タグにAWSリージョンコードを設定してください。（例：us-east-1、ap-northeast-1）
     認証方式に応じて以下のいずれかの設定を行ってください。
     - 認証情報ファイルを利用して認証する場合
       <aws-credentials-profile>タグを記述し、*認証情報ファイルの作成* で得たプロファイル名を設定してください。
     - 設定ファイルに直接記述したアクセスキーとシークレットキーを利用して認証する場合
       <aws-credentials-static>タグを記述し、*アクセスキーとシークレットキーの作成* で得たアクセスキーとシークレットキーを設定してください。
     - インスタンスプロファイルを利用して認証する場合（※Amazon EC2 インスタンス上で iAP が動作している場合のみ）
       <aws-credentials-iam-role-arn>タグを記述し、*ロールの作成* で作成したロールのARNを設定してください。
   - <parameters>タグ内にデフォルトパラメータを指定してください。（モデル、デプロイ名など）

生成AI連携アクション設定

1. 生成AI連携アクション設定ファイルを編集します。
   「ProjectNavigator」内の <（プロジェクト名）/conf/im-copilot-action-config.xml> ファイルをダブルクリックで開き、「ソース」タブを選択してください。

2. 利用するテナントについて、各アクションの利用ドライバ種別を記述してください。

設定内容の詳細については、「設定ファイルリファレンス」 - 「IM-Copilot生成AI連携アクション設定」 を参照してください。

セットアップ（ 各種製品アシスタント ）

各種製品でアシスタント機能を利用するためのセットアップ方法について説明します。

Wiki アシスタントのセットアップ

Wiki アシスタントのセットアップ方法について説明します。

Wiki アシスタントについて

Wiki アシスタントは IM-Wiki内の情報をチャット形式で問い合わせ可能にする機能です。

Wiki アシスタントの特徴は以下の通りです。

- IM-Wikiの任意のコンテンツに対してアシスタントを作成、利用できます。
- Wiki アシスタントの作成単位は「1 Wiki コンテンツ」＝「1 Wiki アシスタント」です。
- 各 Wiki アシスタントは認可設定を持っており利用可能なユーザを制御可能です。

機能の詳細については、「IM-Knowledge管理者操作ガイド」 - 「Wiki アシスタント」を参照してください。

> **注意**
>
> Wiki アシスタントは、intra-mart Accel Platform で提供しているベクトルデータベースアクセス機能を利用して検索拡張生成（RAG）による応答を行います。
> 応答生成時にはキーワード検索とベクトル類似性検索で情報収集を行いますが、標準のベクトルデータベースアクセス機能では、日本語と英語の検索キーワード抽出トークナイザのみ提供しています。
> そのため、日本語・英語以外の言語ではキーワード抽出ができず、キーワード検索による情報収集が不十分となり、回答精度が低下する可能性があります。

セットアップ手順

生成AIサービスとモデル

Wiki アシスタントは、iAP 2025 Spring(Kamille) 時点で以下の生成AIサービスとモデルに対応しています。

Wiki アシスタント対応生成AIサービスとモデル

生成AIサービス	埋め込み	チャット
OpenAI	text-embedding-3-small	gpt-4o-mini
Azure OpenAI Service	text-embedding-ada-002	gpt-4o-mini gpt-4o
Amazon Bedrock	amazon.titan-embed-text-v1	anthropic.claude-3-haiku-20240307-v1:0

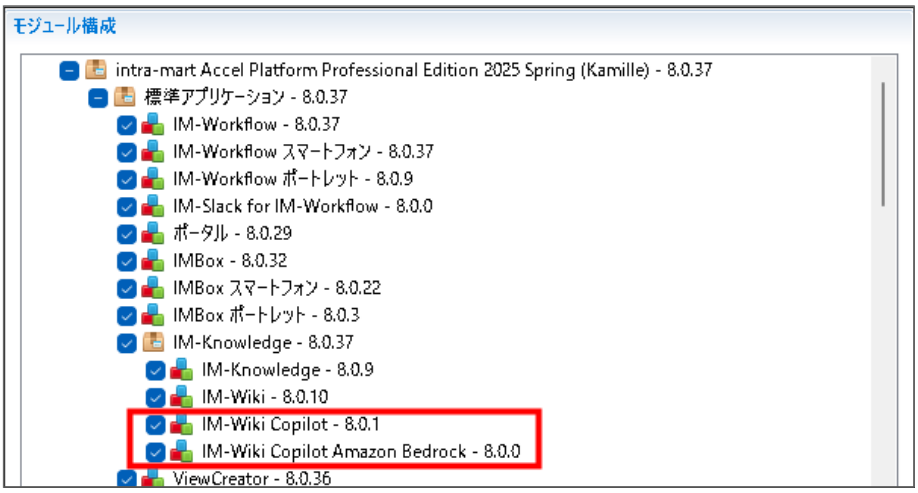
生成AI連携ドライバ設定

Wiki アシスタントは、*生成AI連携ドライバ設定* ファイルの一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。

- *生成AI連携ドライバ設定* ファイルにテナントドライバ情報設定（drivers）の設定が存在する場合、その xml 内で一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。
- テナントドライバ情報設定（drivers）の設定が存在しない場合は、デフォルトドライバ情報設定（default-drivers）の xml 内で一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。

IM-Jugglingプロジェクトの編集

ご利用のIM-Jugglingプロジェクトに、IM-Wiki Copilotモジュールを追加してください。
IM-Wiki Copilotモジュールは、[標準アプリケーション - IM-Knowledge] 配下に存在します。



テキスト抽出設定

Wiki アシスタントは、添付ファイルのテキスト抽出にテキスト抽出機能（ND Universal Extractor）を利用します。
必要に応じて、テキスト抽出機能のテキスト抽出設定の調整を検討してください。
設定内容の詳細については、 「設定ファイルリファレンス」 - 「テキスト抽出設定」 を参照してください。

ベクトルデータベース接続設定

Wiki アシスタントは、テナント環境セットアップでベクトルデータベース接続情報の設定が必要です。
設定内容の詳細については、 「テナント環境セットアップ」 - 「ベクトルデータベース接続情報」 を参照してください。

テナントセットアップ後の設定

テナントセットアップ後、運用に合わせて Wiki アシスタントに関する設定が必要です。
設定内容の詳細については、「IM-Knowledge管理者操作ガイド」 - 「Wiki アシスタント」を参照してください。

ViewCreator SQLビルダ アシスタント のセットアップ

SQL生成アシスタントのセットアップ方法について説明します。

項目

ViewCreator SQLビルダ アシスタント について

ViewCreator SQLビルダ アシスタント は、テーブルのデータを参照するためのSQL（SELECT文）作成を、生成AIがサポートする機能です。
「ユーザの一覧を取得するSQLを作成してください」のような、自然言語による指示からSQLを生成できます。

ViewCreator SQLビルダ アシスタント は、製品標準で作成されるテーブル情報について回答できます。
また、独自で追加したテーブルについても、生成AIが参照可能な学習データを追加できます。
具体的な利用方法は「SQLビルダによるクエリの作成」を参照してください。

> **コラム**
>
> 製品標準で用意されるテーブルであっても、ViewCreator SQLビルダ アシスタント が利用できないテーブルがあります。
> 利用可能なテーブルについて、質問できます。
> 例）「xxxテーブルやxxxテーブルは利用できますか？」

セットアップ手順

生成AIサービスとモデル

ViewCreator SQLビルダ アシスタント は、iAP 2025 Spring(Kamille) 時点で以下の生成AIサービスとモデルに対応しています。

ViewCreator SQLビルダ アシスタント 対応生成AIサービスとモデル

生成AIサービス	埋め込み	チャット
OpenAI	text-embedding-3-small	gpt-4o-mini
Amazon Bedrock	amazon.titan-embed-text-v1	anthropic.claude-3-haiku-20240307-v1:0

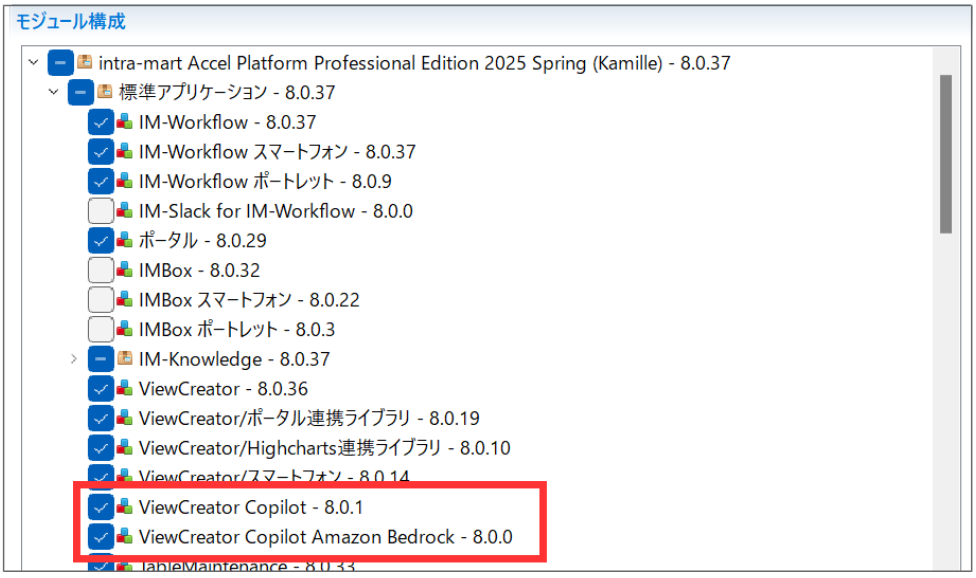
生成AI連携ドライバ設定

ViewCreator SQLビルダ アシスタント は、*生成AI連携ドライバ設定* ファイルの一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。

- *生成AI連携ドライバ設定* ファイルにテナントドライバ情報設定（drivers）の設定が存在する場合、その xml 内で一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。
- テナントドライバ情報設定（drivers）の設定が存在しない場合は、デフォルトドライバ情報設定（default-drivers）の xml 内で一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。

IM-Juggling プロジェクトの編集

ご利用のIM-Jugglingプロジェクトに、ViewCreator Copilotモジュールを追加してください。
ViewCreator Copilotモジュールは、[標準アプリケーション] 配下に存在します。



ベクトルデータベース接続設定

ViewCreator SQLビルダ アシスタント は、テナント環境セットアップでベクトルデータベース接続情報の設定が必要です。
設定内容の詳細については、「テナント環境セットアップ」 - 「ベクトルデータベース接続情報」 を参照してください。

テナントセットアップ後の設定

テナントセットアップ後、アシスタント使用前の準備として以下の操作が必要です。

- ViewCreator SQLビルダ アシスタント の認可設定
- ジョブネットの実行

詳細については、「ViewCreator 管理者操作ガイド」 を参照してください。

Accel Studio アプリケーション作成 アシスタント のセットアップ

Accel Studio アプリケーション作成 アシスタント のセットアップ方法について説明します。

項目

Accel Studio アプリケーション作成 アシスタント について

Accel Studio アプリケーション作成 アシスタント は、Accel Studioのアプリケーションを作成する際、自然言語による指示でアプリケーションの入力項目の自動補完を行う機能です。
例えば、「書籍管理アプリケーションを作成してください」といった指示に対して、アプリケーションの入力項目を自動補完します。
具体的な利用方法は 「Accel Studio アプリケーション管理機能 仕様書」 - 「アプリケーション作成 アシスタント」を参照してください。

セットアップ手順

生成AIサービスとモデル

Accel Studio アプリケーション作成 アシスタント は、iAP 2025 Spring(Kamille) 時点で以下の生成AIサービスとモデルに対応しています。

Accel Studio アプリケーション作成 アシスタント 対応生成AIサービスとモデル

生成AIサービス	チャット
OpenAI	gpt-4o-mini
Azure OpenAI Service	gpt-4o-mini
Amazon Bedrock	anthropic.claude-3-5-sonnet-20240620-v1:0

生成AI連携ドライバ設定

Accel Studio アプリケーション作成 アシスタント は、*生成AI連携ドライバ設定* ファイルの一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。

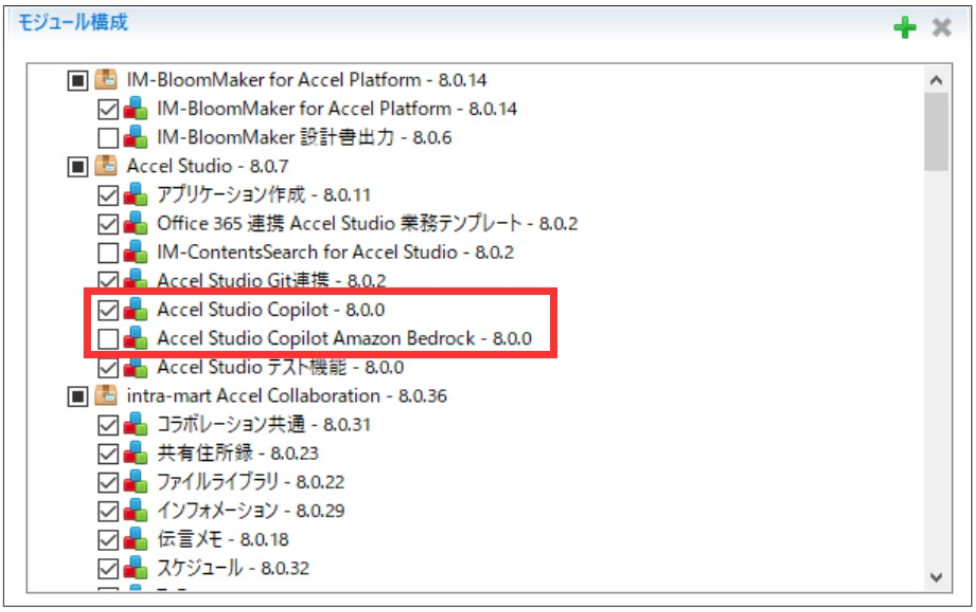
- *生成AI連携ドライバ設定* ファイルにテナントドライバ情報設定（drivers）の設定が存在する場合、その xml 内で一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。
- テナントドライバ情報設定（drivers）の設定が存在しない場合は、デフォルトドライバ情報設定（default-drivers）の xml 内で一番初めに記載されているドライバ設定を基に生成AIサービスに接続します。

> **コラム**
>
> Accel Studio アプリケーション作成 アシスタント で利用するモデルを変更する場合は、*生成AI連携ドライバ設定* ファイルのドライバ設定を変更してください。

IM-Jugglingプロジェクトの編集

ご利用のIM-Jugglingプロジェクトに、Accel Studio Copilotモジュールを追加してください。

Accel Studio Copilotモジュールは、[アプリケーション作成] 配下に存在します。



> **コラム**
>
> 各種製品で利用できるアシスタントは、今後の製品アップデートに合わせて追加を予定しています。

共通アシスタント実行画面

概要

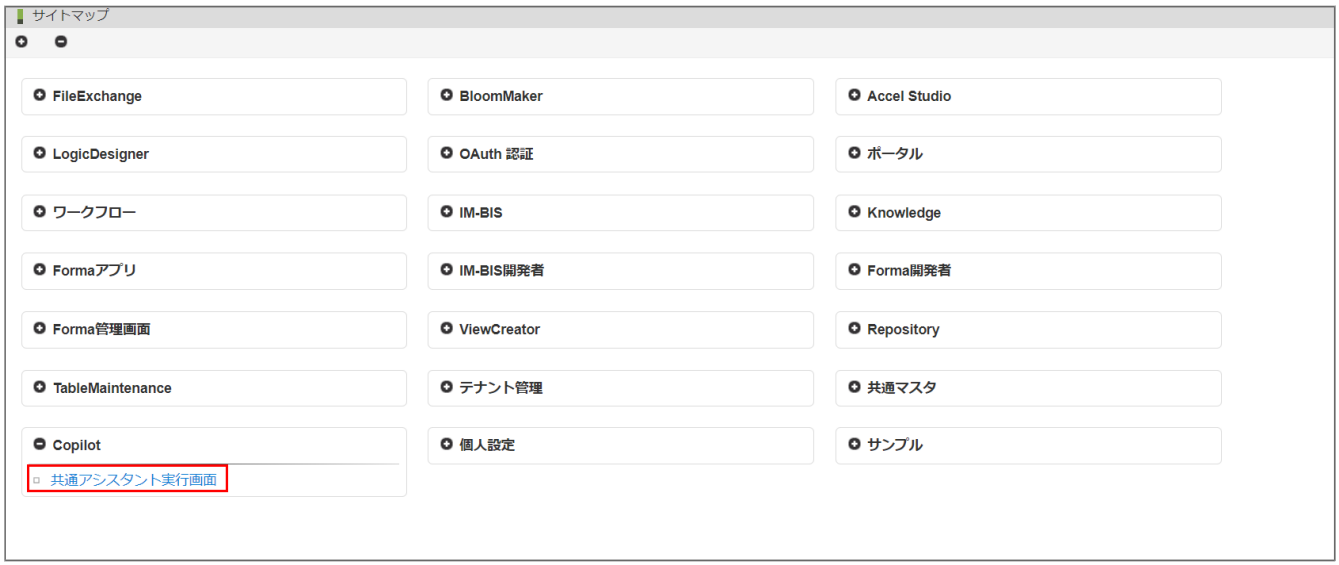
共通アシスタント実行画面は生成AIアシスタントの実行を行うための共通画面です。
作成した Wiki アシスタントの実行結果を確認したい場合等に利用します。

> **コラム**
>
> Wiki アシスタントの登録方法については、「IM-Knowledge管理者操作ガイド」 - 「Wiki アシスタント」を参照してください。

アシスタントの実行方法は以下のとおりです。

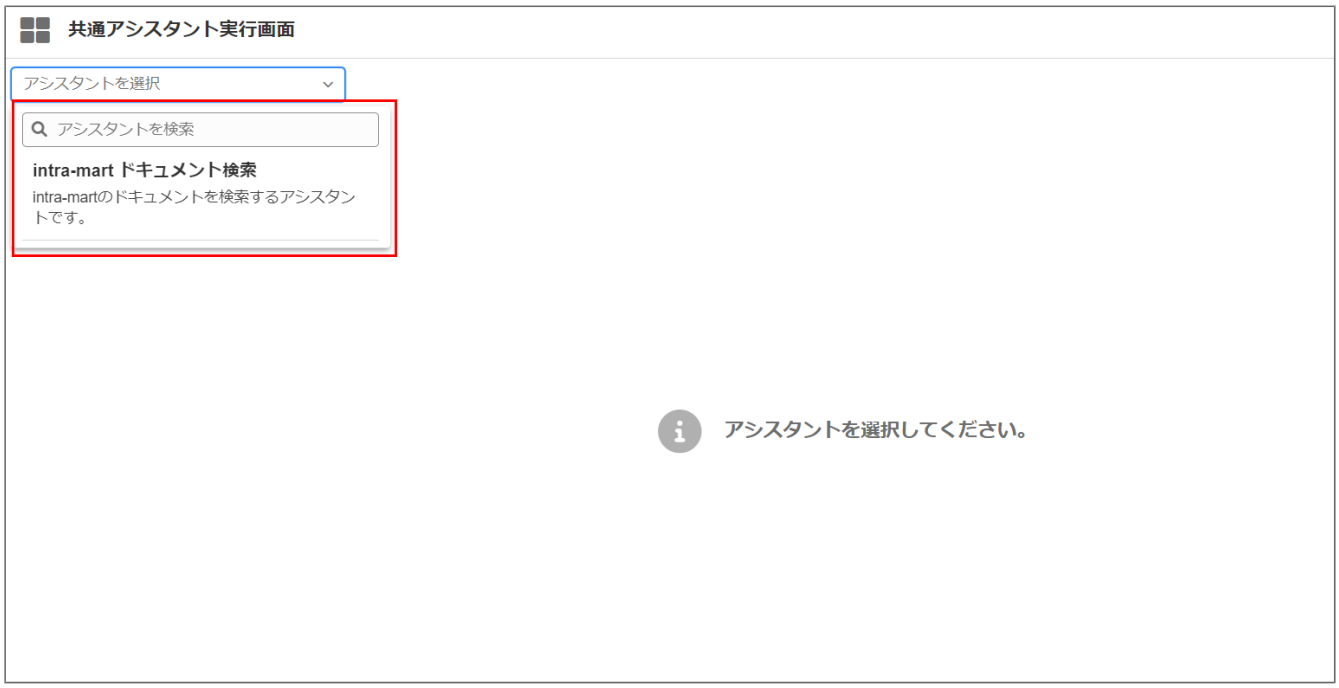
アシスタントの実行

1. 「サイトマップ」→「Copilot」→「共通アシスタント実行画面」をクリックします。



2. 共通アシスタント実行画面の「アシスタントを選択」セレクトボックスをクリックします。

3. アシスタントの一覧から実行するアシスタントを選択します。

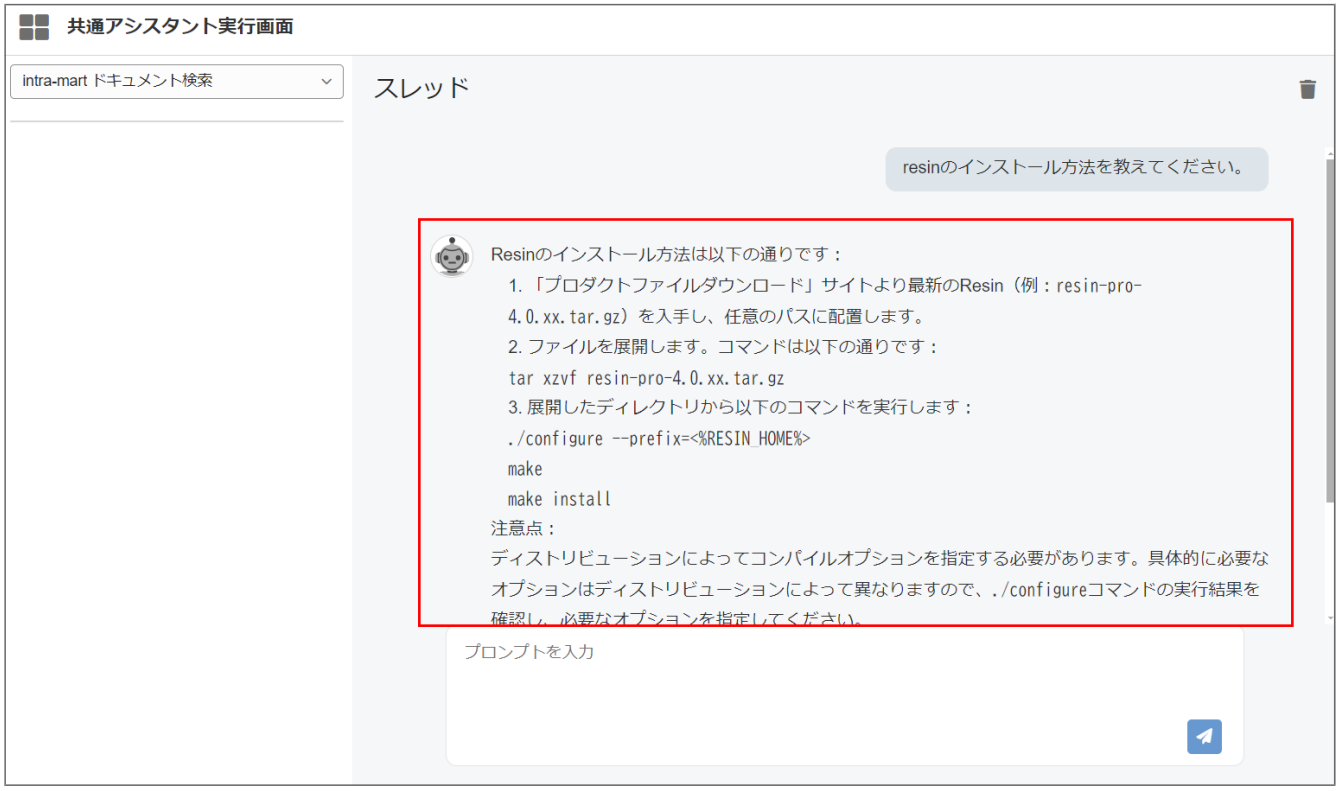


4. プロンプトの入力欄に問い合わせ内容を入力します。

5. アシスタントの実行ボタンをクリックします。

6. アシスタントの実行が開始され、「処理中」と表示されます。

7. 問い合わせ結果が表示されます。



メッセージ履歴の削除

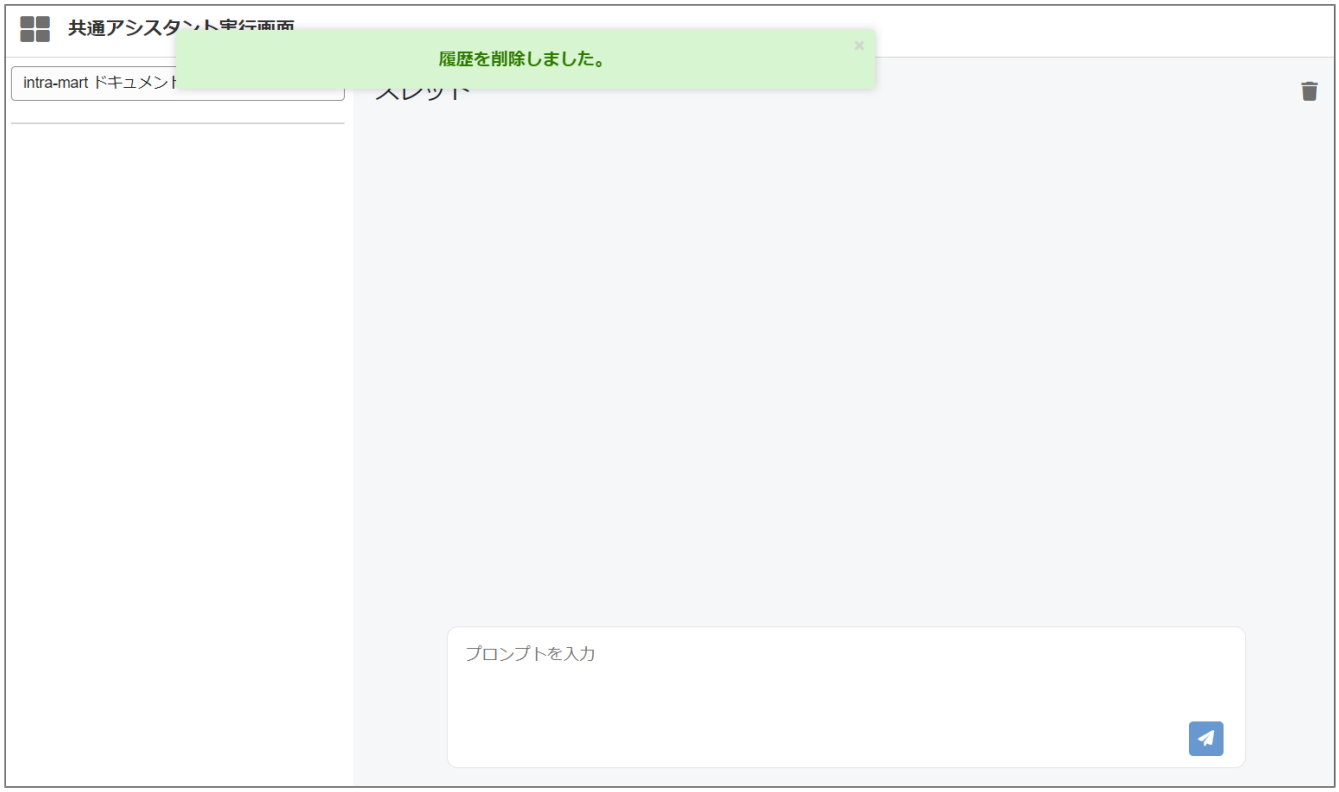
メッセージ履歴を利用するアシスタントの場合、問い合わせ内容とその結果がメッセージ履歴として保存されます。
保存された履歴はアシスタント実行時に生成AIサービスへのリクエスト送信時等に利用されます。
そのため、メッセージ履歴が多い場合にトークン数上限に到達してしまいエラーが発生する場合があります。

不要なメッセージ履歴は以下の手順で削除できます。

1. 共通アシスタント実行画面の「削除」アイコンをクリックします。

2. 確認ダイアログの「確定」ボタンをクリックします。

3. メッセージ履歴が削除されます。


履歴を削除しました。
プロンプトを入力

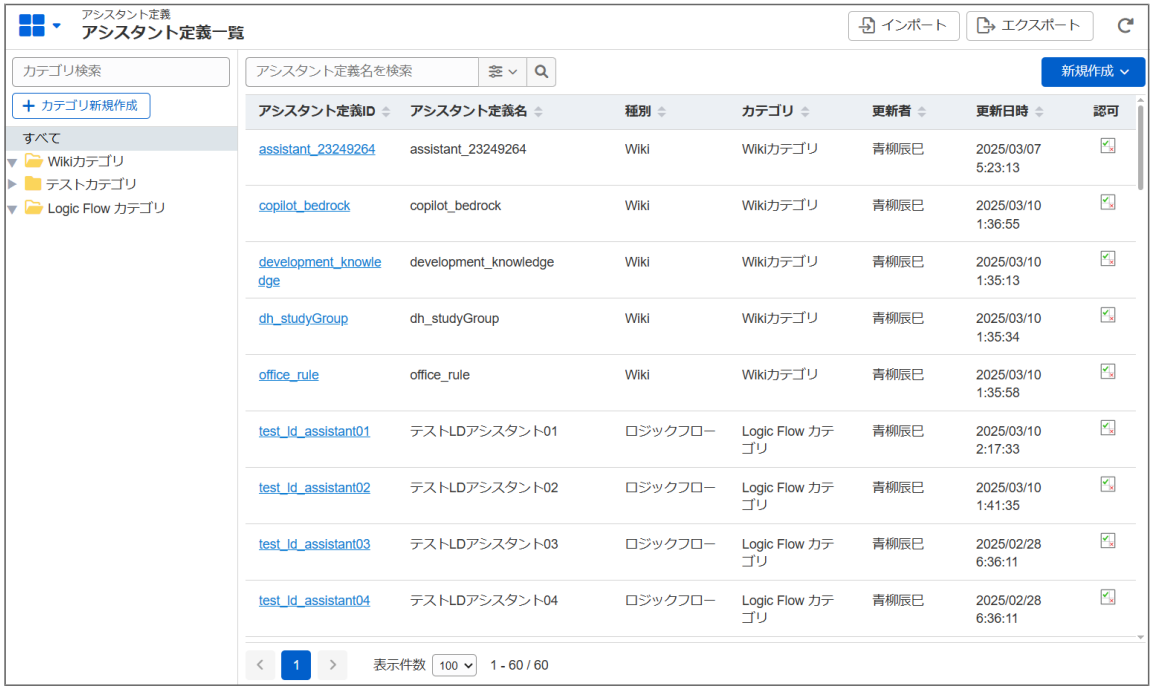
アシスタント定義

アシスタント定義は、 IM-LogicDesignerのロジックフローやIM-Wikiのコンテンツから生成AIアシスタントを作成・管理する機能です。 ここではアシスタント定義を扱う画面の機能について説明します。

アシスタント定義一覧

1. 「サイトマップ」→「Copilot」→「アシスタント定義」→「アシスタント定義一覧」をクリックし、「アシスタント定義一覧」画面を表示します。

2. 「アシスタント定義一覧」画面が表示されます。



＜画面項目（ヘッダ）＞

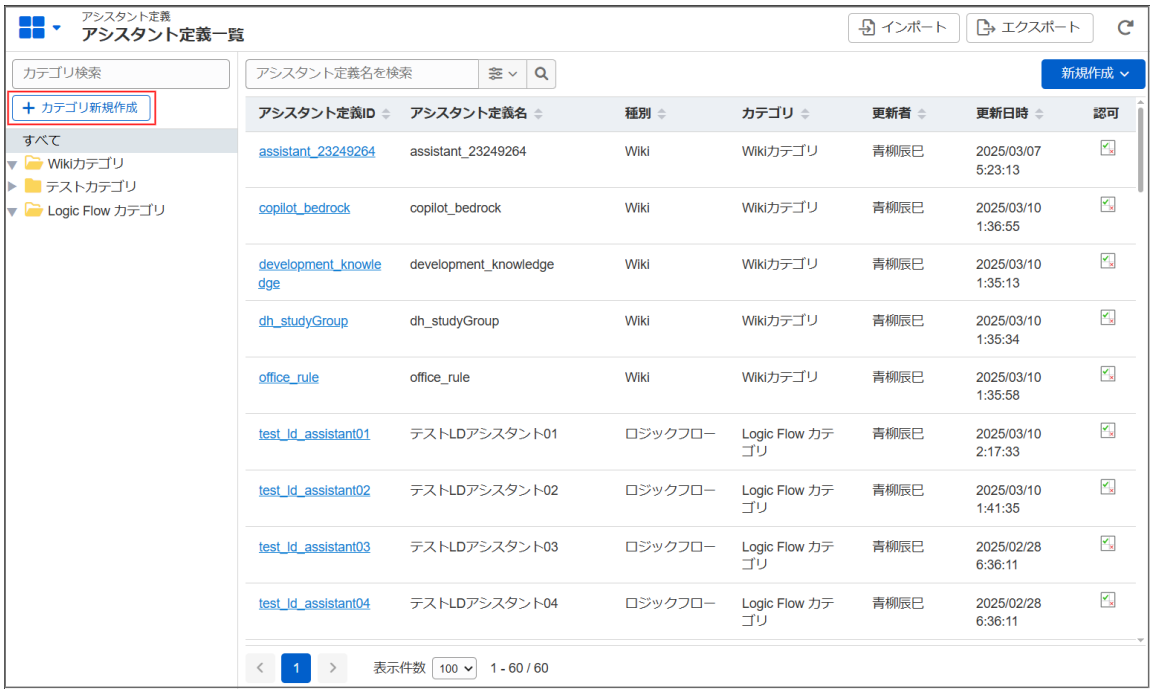
項目	説明
「関連リンク」アイコン	共通アシスタント実行、インポート、エクスポート画面へのリンクが表示されます。
「インポート」ボタン	アシスタント定義 インポート画面に遷移します。
「エクスポート」ボタン	アシスタント定義 エクスポート画面に遷移します。
「更新」アイコン	ページを再読み込みします。

＜画面項目（コンテンツ）＞

項目	説明
カテゴリ検索窓	検索するカテゴリ名を表す文字列（の一部）を入力します。
「カテゴリ新規作成」ボタン	カテゴリ新規作成ダイアログが表示されます。
アシスタント定義名検索窓	検索するアシスタント定義名を表す文字列（の一部）を入力します。
「詳細検索」アイコン	検索するアシスタント定義ID、アシスタント定義名を表す文字列（の一部）を入力、またはアシスタント定義種別を指定して検索します。
「虫眼鏡」アイコン	検索を実行します。
「新規作成」ボタン	アシスタント定義種別リストが表示されます。
「認可」アイコン	アシスタント利用ユーザがアシスタントを利用するための認可設定画面が表示されます。
「ページャ」アイコン	一覧の表示ページを切り替えます。
「表示件数」プルダウン	エンティティログの表示件数を設定します。 設定可能は表示件数は、50, 100, 150, 200 件です。

カテゴリを登録する

1. 「カテゴリ新規作成」アイコンをクリックします。



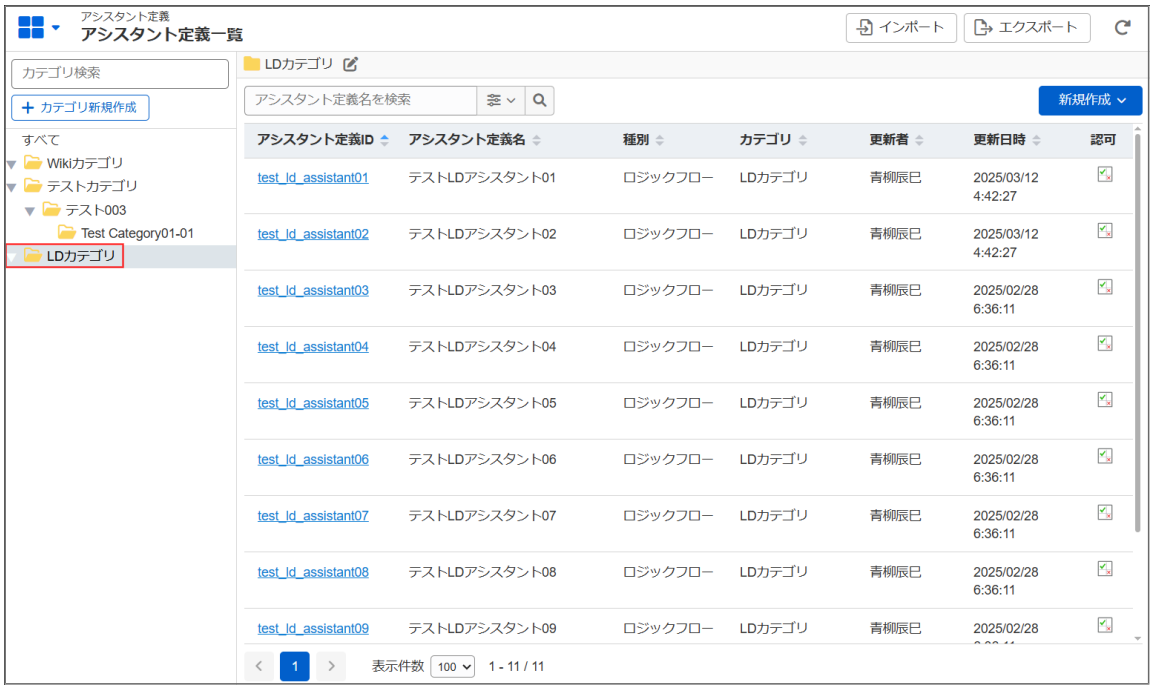
2. カテゴリ新規作成ダイアログが表示されます。
   カテゴリ情報を入力し「登録」ボタンをクリックします。

＜画面項目＞

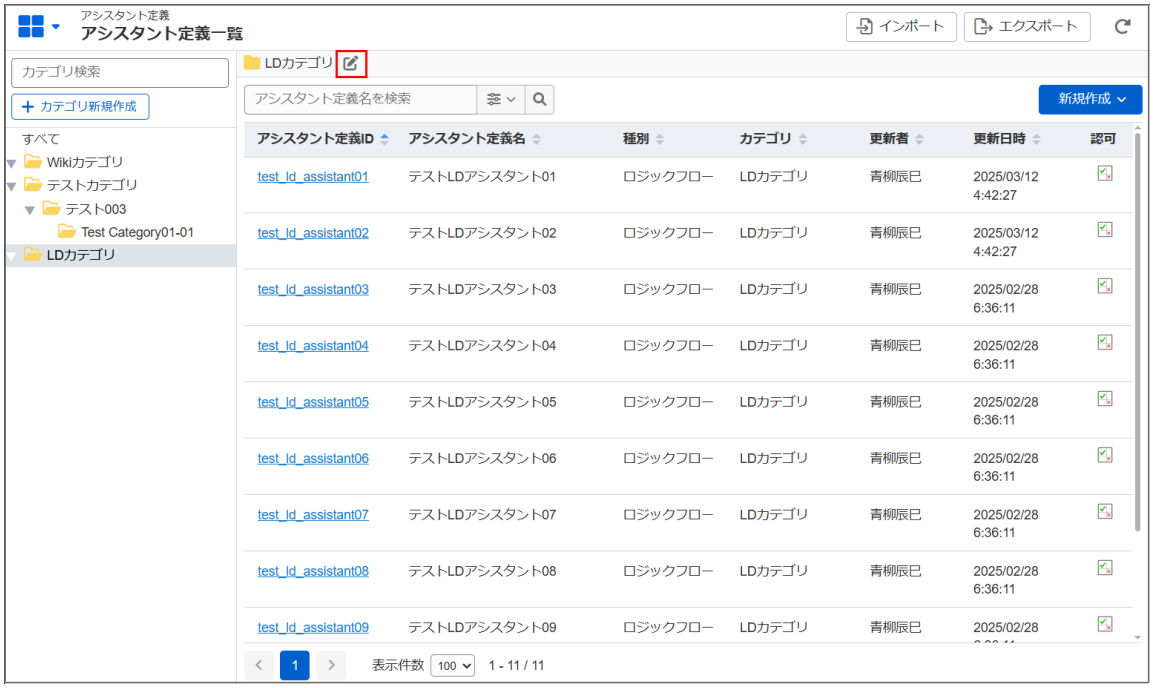
項目	説明
カテゴリID	カテゴリを一意に表す文字列を設定します。 この項目は必須項目です。
カテゴリ名	カテゴリを表す名称を設定します。 名称には各言語で利用するものと、言語情報が指定されていない場合に標準で利用するものを設定します。 この項目は標準のみ必須項目です。
ソート番号	カテゴリ表示時のソート順を設定します。 この項目は必須項目です。
親カテゴリ	親カテゴリを指定します。
「キャンセル」ボタン	操作をキャンセルしてダイアログをクローズします。
「登録」ボタン	カテゴリを登録します。

カテゴリを編集する

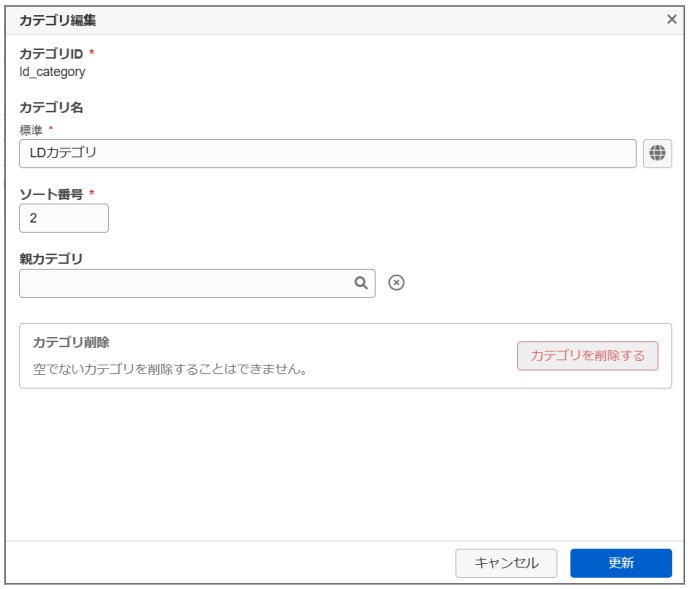
1. 「*アシスタント定義一覧*」の手順をもとに、アシスタント定義一覧を表示して、カテゴリツリーから編集するカテゴリをクリックします。



2. 「カテゴリ編集」アイコンをクリックします。

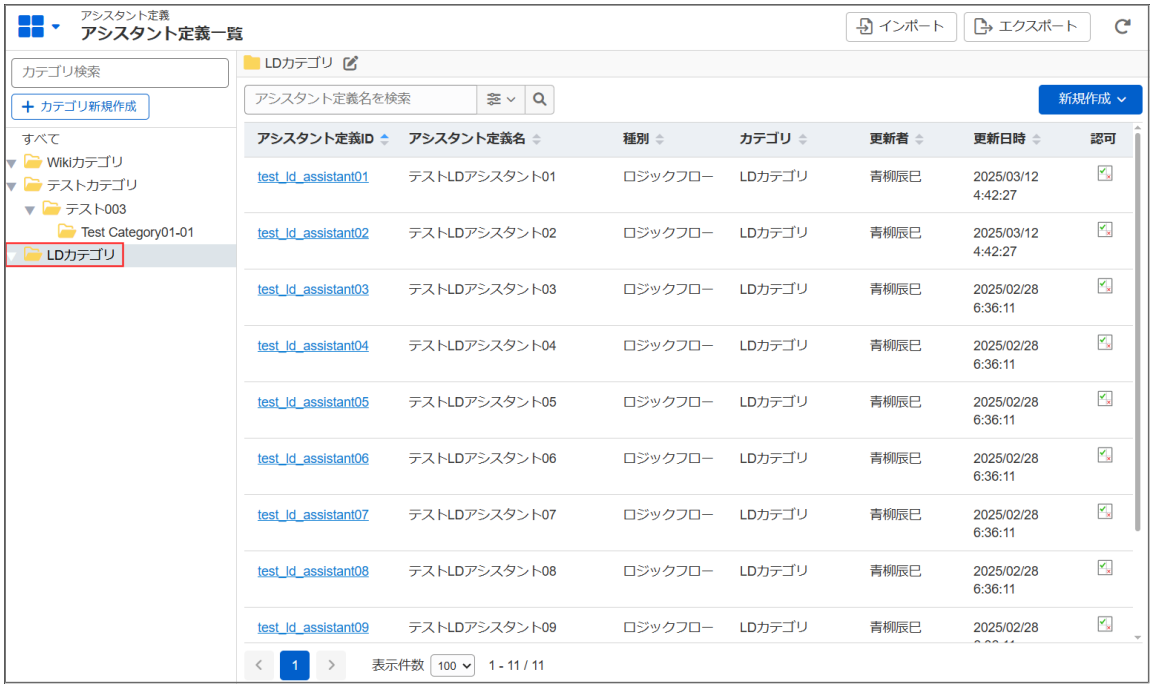


3. カテゴリ編集ダイアログが表示されます。
   カテゴリ情報を入力し「更新」ボタンをクリックします。

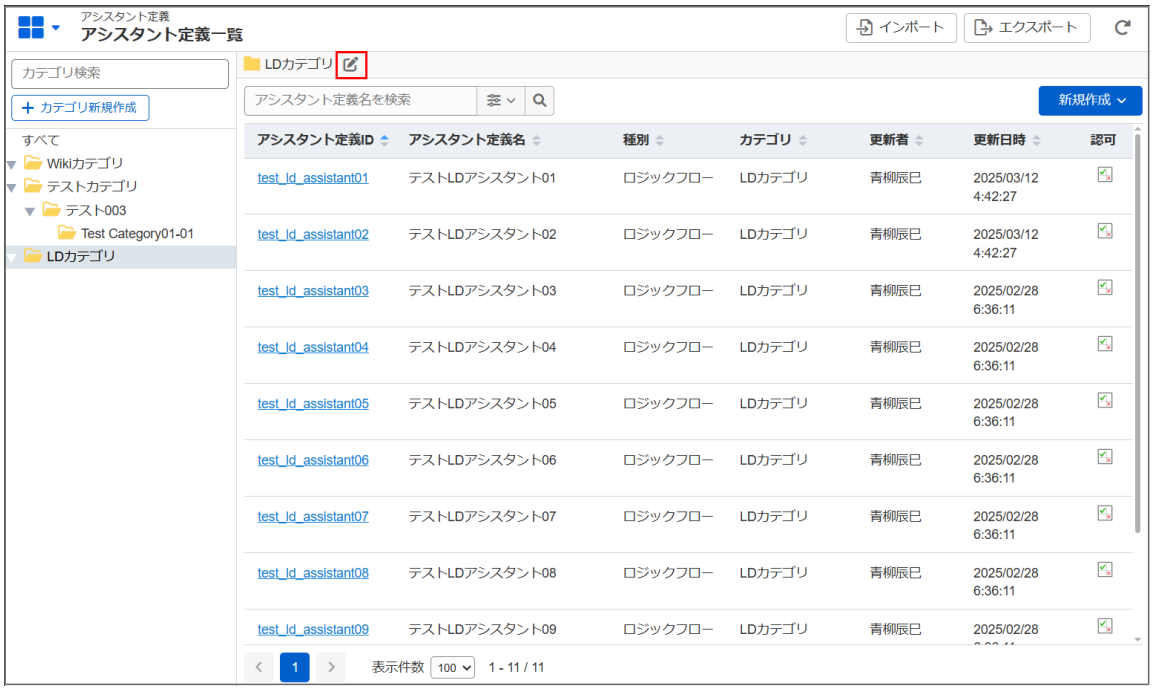


カテゴリを削除する

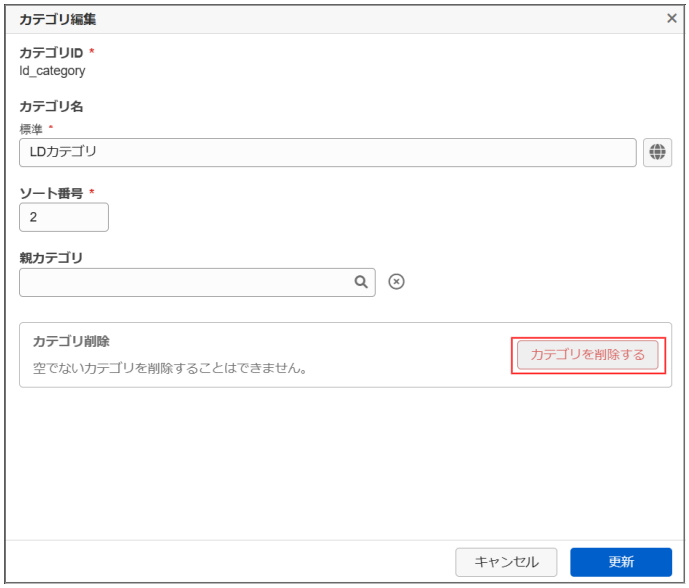
1. 「*アシスタント定義一覧*」の手順をもとに、アシスタント定義一覧を表示して、カテゴリツリーから削除するカテゴリをクリックします。



2. 「カテゴリ編集」アイコンをクリックします。

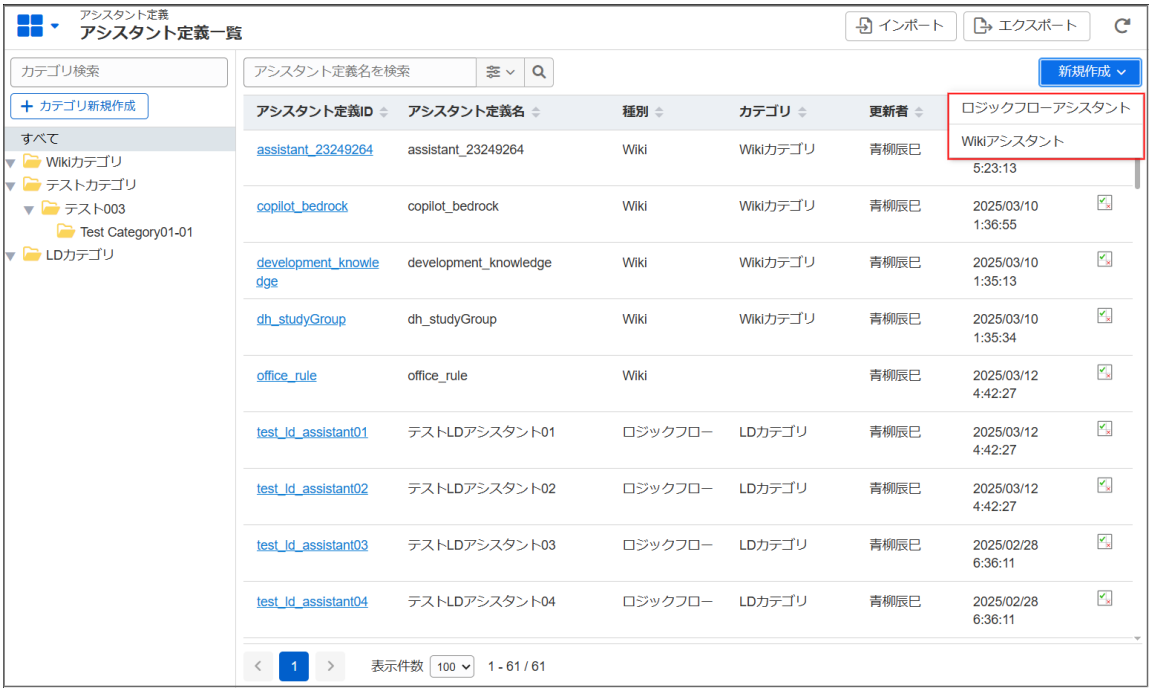


3. 「カテゴリを削除する」ボタンをクリックします。確認ダイアログの「削除」ボタンをクリックします。



アシスタント定義を登録する

1. 「新規作成」ボタン、または「アシスタント定義を新規作成」ボタンをクリックします。
   アシスタント定義種別リストから作成対象のアシスタント定義種別を選択します。
   - 「新規作成」ボタンから作成



   - 「アシスタント定義を新規作成」ボタンから作成

2. アシスタント定義 新規作成画面に遷移します。
   アシスタント情報を入力し「新規作成」ボタンをクリックします。

＜画面項目（基本情報）＞

項目	説明
アシスタントID	アシスタントを一意に表す文字列を設定します。 この項目は必須項目です。
アシスタント名	アシスタントを表す名称を入力します。 名称には各言語で利用するものと、言語情報が指定されていない場合に標準で利用するものを設定します。 この項目は標準のみ必須項目です。
カテゴリ	アシスタントが属するカテゴリを指定します。
説明	アシスタントの説明を入力します。 説明には各言語で利用するものと、言語情報が指定されていない場合に標準で利用するものを設定します。

＜画面項目（詳細情報 - ロジックフロー）＞

項目	説明

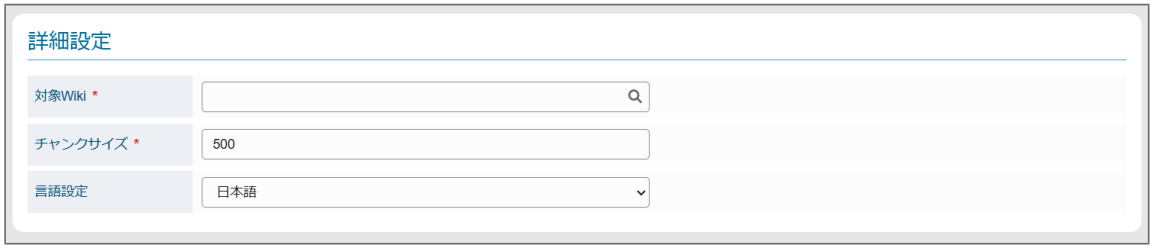
項目	説明
対象フロー	アシスタントを作成するロジックフローを指定します。 この項目は必須項目です。
バージョン	アシスタントを作成するロジックフローのバージョンを設定します。 最新バージョンまたは利用するバージョンを指定します。

コラム

ロジックフローアシスタントについて

ロジックフローアシスタントの構築手順については以下を参照してください。
「*ロジックフローアシスタントの作成例*」

＜画面項目（詳細情報 - Wikiアシスタント）＞



項目	説明
対象Wiki	アシスタントを作成するWikiを指定します。 この項目は必須項目です。
チャンクサイズ	コンテンツの内容をベクトルデータベースへ格納する際に行うテキスト分割の単位となる文字数を設定します。 設定可能なチャンクサイズの最小値は100 、最大値は4000 です。 この項目は必須項目です。
言語設定	対象のWiki コンテンツの言語を指定します。

コラム

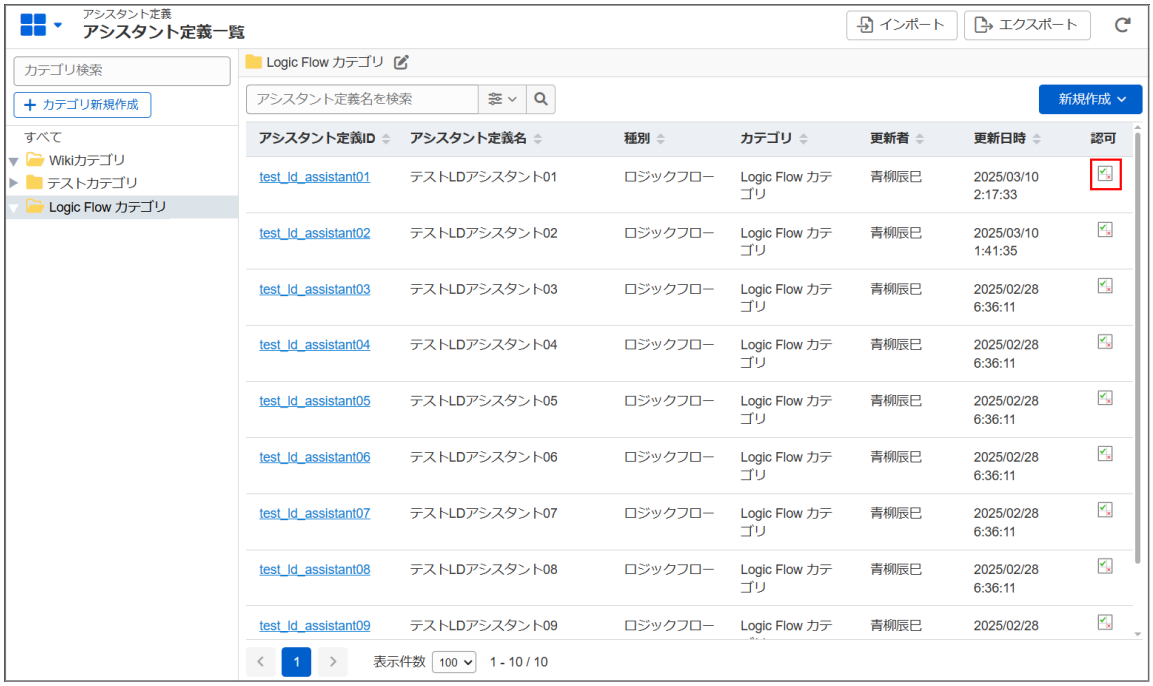
Wikiアシスタントについて

- IM-Juggling プロジェクトに、IM-Wiki Copilotモジュールが含まれていない場合は、アシスタント定義種別リストでWikiアシスタントを選択できません。
- Wikiアシスタントは「 Knowledge 」のメニューの Wiki 画面でも作成できます。
  Wiki 画面での登録方法については、「IM-Knowledge管理者操作ガイド」 - 「Wiki アシスタント」を参照してください。
- 作成したWikiアシスタントの認可設定の初期状態は、対象Wikiのナレッジグループの認可設定が引き継がれています。

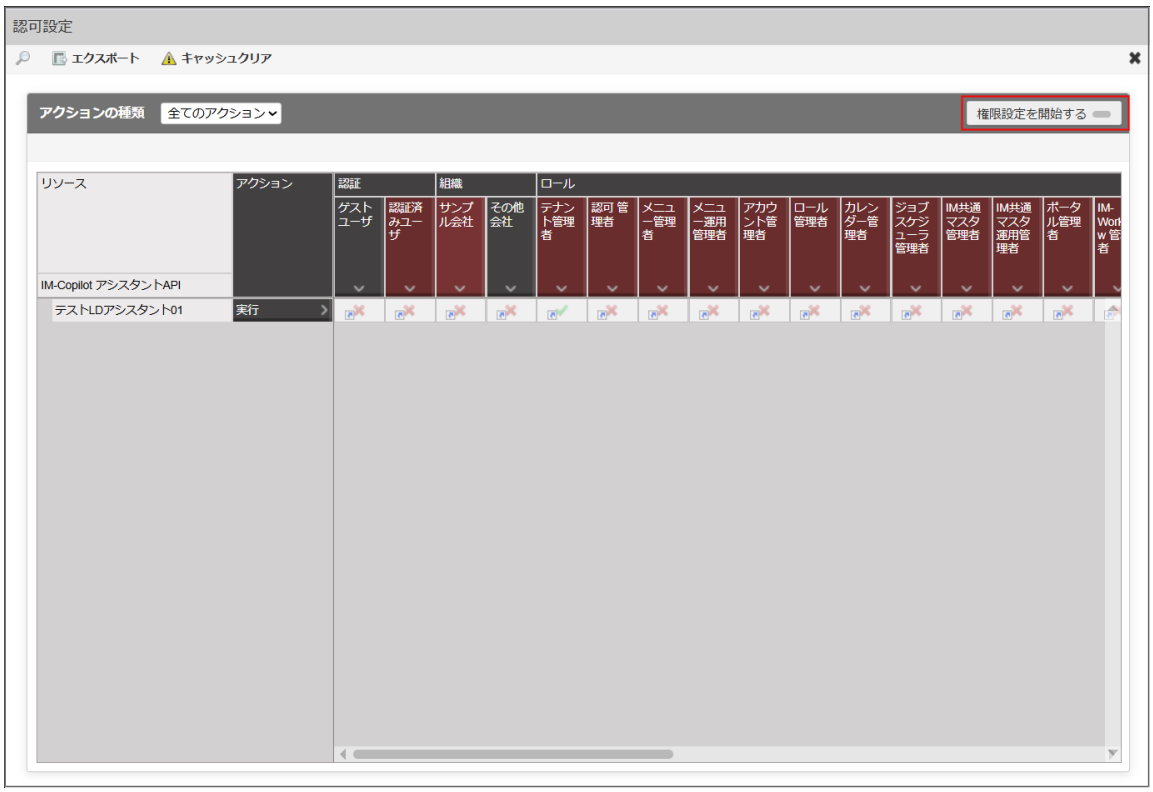
アシスタント定義の認可設定をする

アシスタント定義の作成後は、必要に応じてアシスタント利用ユーザがアシスタントを利用するための認可設定を行います。

1. アシスタント定義一覧から認可設定を行うアシスタント定義の「認可設定」アイコンをクリックします。

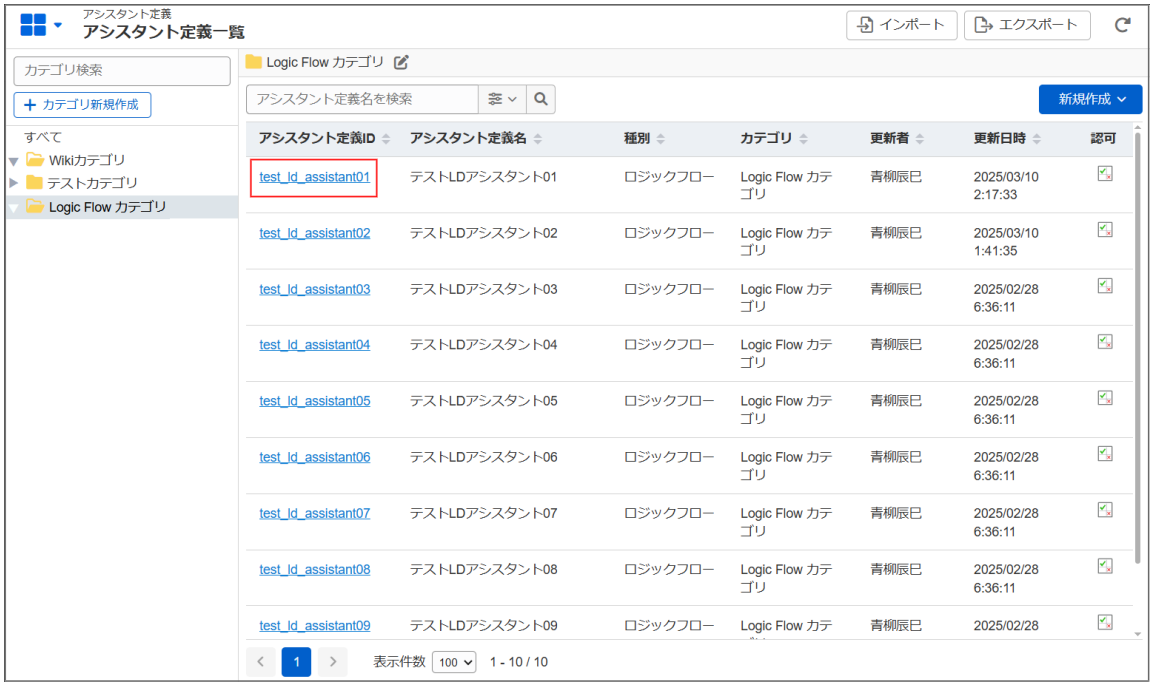


2. 別画面に「アシスタントの認可設定」画面が表示されます。「権限設定を開始する」をクリックし、認可設定を行います。
   認可設定画面の基本的な利用方法は「テナント管理者操作ガイド」 - 「認可を設定する」を参照してください。

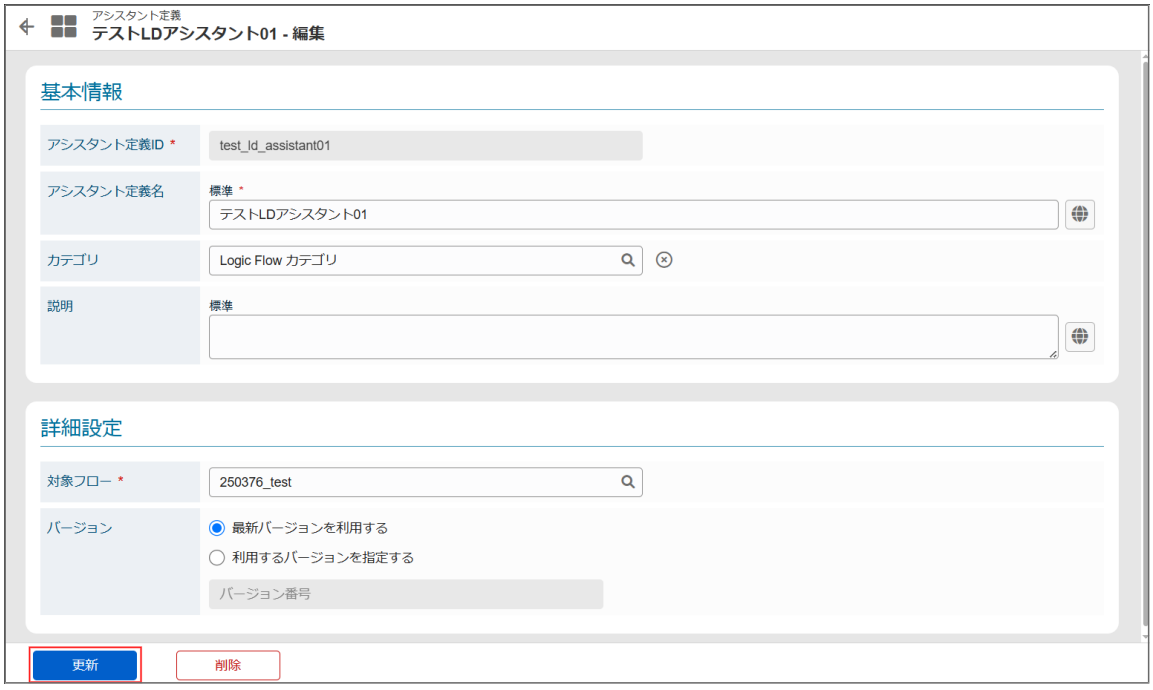


アシスタント定義を編集する

1. 「*アシスタント定義一覧*」の手順をもとに、アシスタント定義一覧を表示します。
   一覧から編集するアシスタント定義のリンクをクリックします。

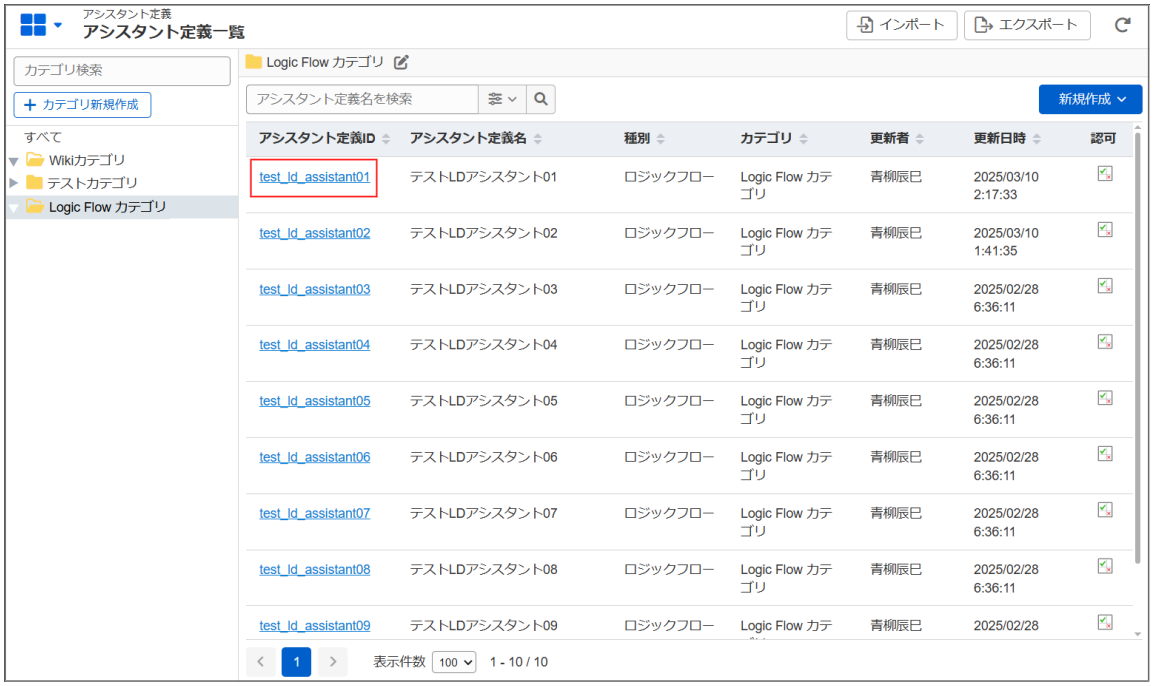


2. アシスタント情報を編集して「更新」ボタンをクリックします。確認ダイアログの「更新」ボタンをクリックします。

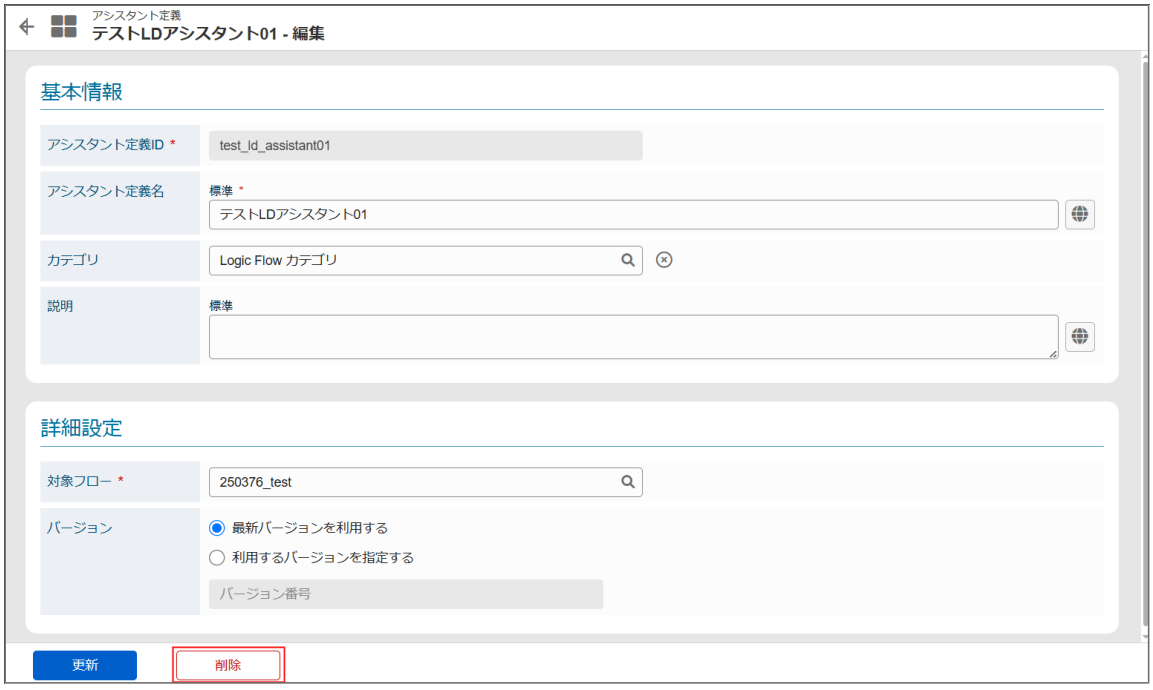


アシスタント定義を削除する

1. 「*アシスタント定義一覧*」の手順をもとに、アシスタント定義一覧を表示します。
   一覧から削除するアシスタント定義のリンクをクリックします。

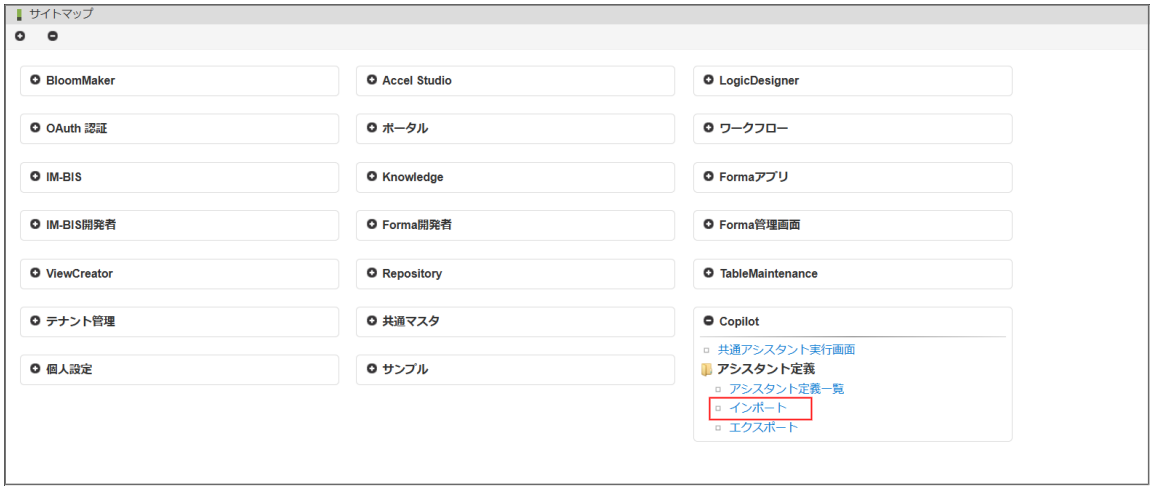


2. 「削除」ボタンをクリックします。確認ダイアログの「削除」ボタンをクリックします。



アシスタント定義のインポート

1. 「サイトマップ」→「Copilot」→「アシスタント定義」→「インポート」をクリックし、「インポート」画面を表示します。

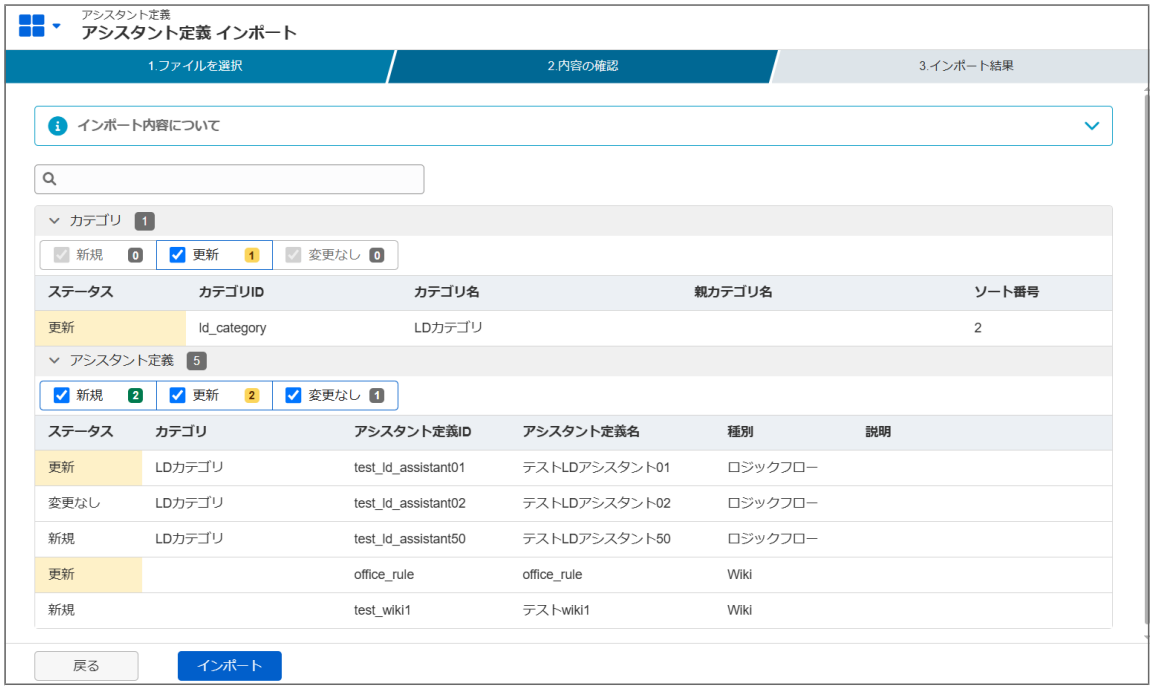


2. 「インポート」画面が表示されます。

3. インポートファイルをドラッグ＆ドロップまたは「ファイルを選択」ボタンをクリックしてファイルを選択します。
インポートできるファイル形式は、jsonのみです。

4. 「内容の確認へ」ボタンをクリックします。

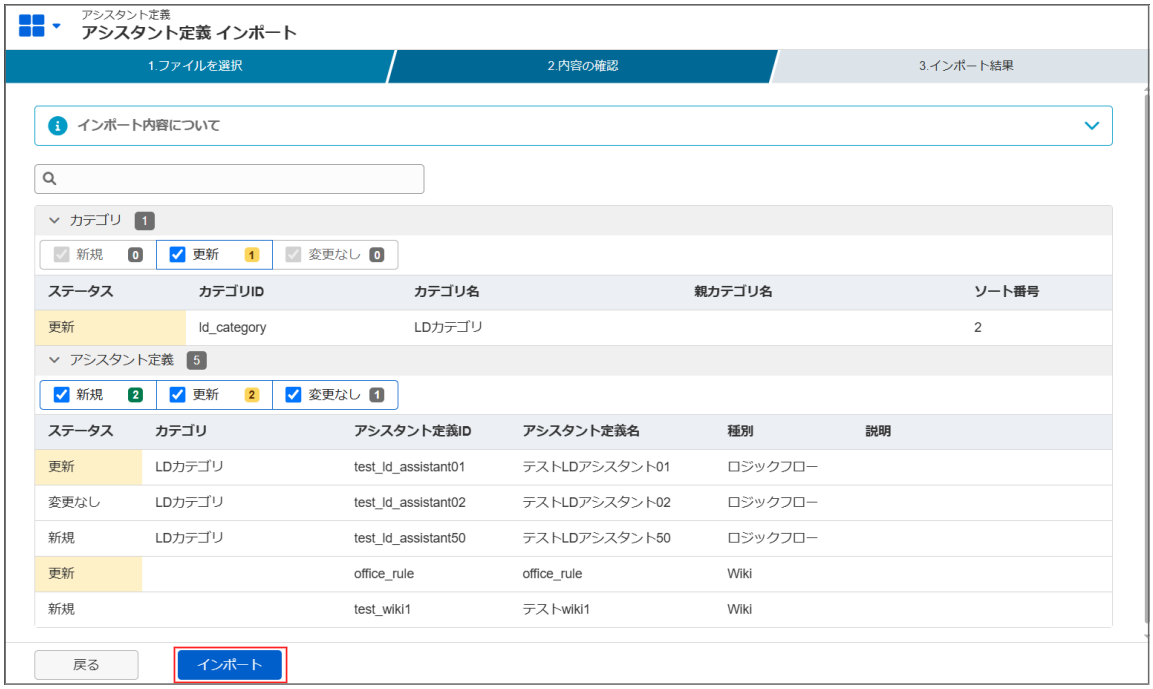
5. 「内容の確認」ステップが表示されます。
   インポートされるカテゴリ、アシスタント定義の情報が一覧で表示されます。
   それぞれのデータの状態に応じてステータスを表示します。
   - 新規：新規データ
   - 更新：既存データの更新
   - 変更なし：既存データに変更がない
     ※変更がないデータもインポートされ、更新者・更新日時が更新されます。



＜画面項目＞

項目	説明
検索窓	検索する文字列（の一部）を入力します。 一覧に表示されているカテゴリ・アシスタント定義のすべての項目に対して検索します。
カテゴリ ステータスフィルタ	チェックボックスがONのステータスでカテゴリを絞り込みます。
アシスタント定義 ステータスフィルタ	チェックボックスがONのステータスでアシスタント定義を絞り込みます。
「戻る」ボタン	「ファイルを選択」ステップに遷移します。
「インポート」ボタン	インポートを実行します。

6. 「インポート」ボタンをクリックします。確認ダイアログの「インポート」ボタンをクリックします。



7. アシスタント定義のインポート結果が表示されます。

＜画面項目＞

項目	説明
「アシスタント定義一覧を開く」ボタン	「アシスタント定義一覧」画面に遷移します。
「インポート画面に戻る」ボタン	「インポート」画面に遷移します。
検索窓	検索する文字列（の一部）を入力します。 一覧に表示されているカテゴリ・アシスタント定義のすべての項目に対して検索します。
「失敗のみ表示」チェックボックス	インポートに失敗したデータがある場合に表示されます。 チェックボックスをONにするとインポートに失敗したカテゴリ・アシスタント定義のみが表示されます。

アシスタント定義のエクスポート

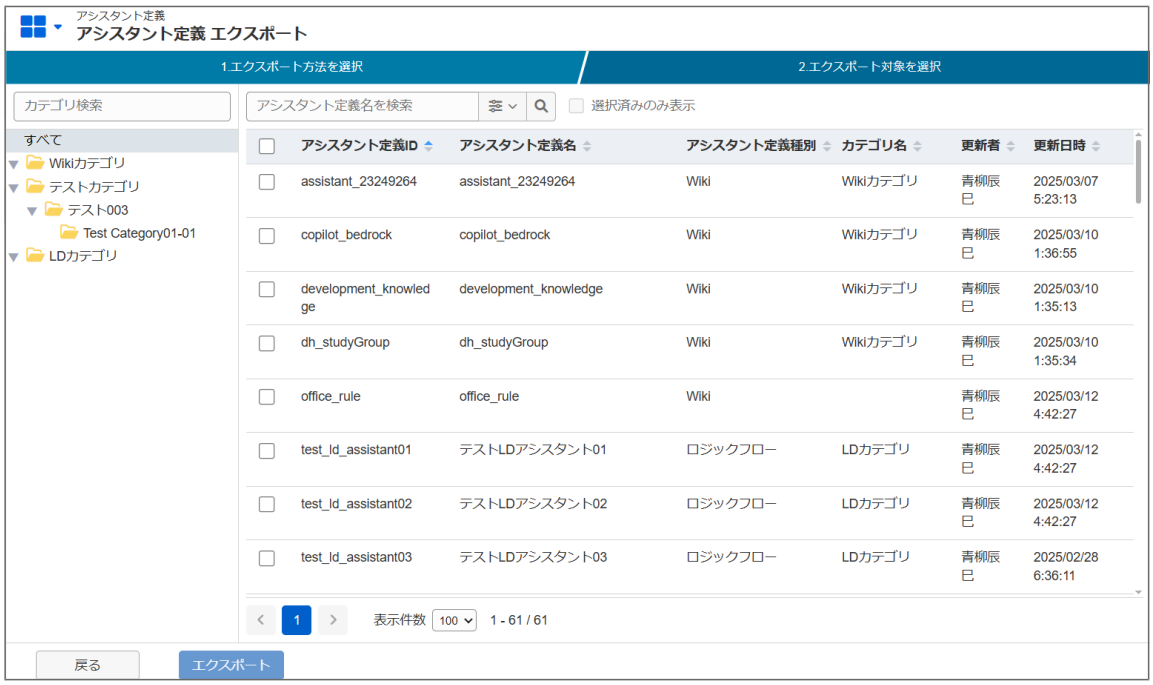
1. 「サイトマップ」→「Copilot」→「アシスタント定義」→「エクスポート」をクリックし、「エクスポート」画面を表示します。

2. 「エクスポート」画面が表示されます。

3. すべてのアシスタント定義をエクスポートする場合は、そのまま「エクスポート」ボタンをクリックします。

4. エクスポートするアシスタント定義を選択する場合は、「任意のアシスタント定義を選択してエクスポート」を選択して「次へ」ボタンをクリックします。

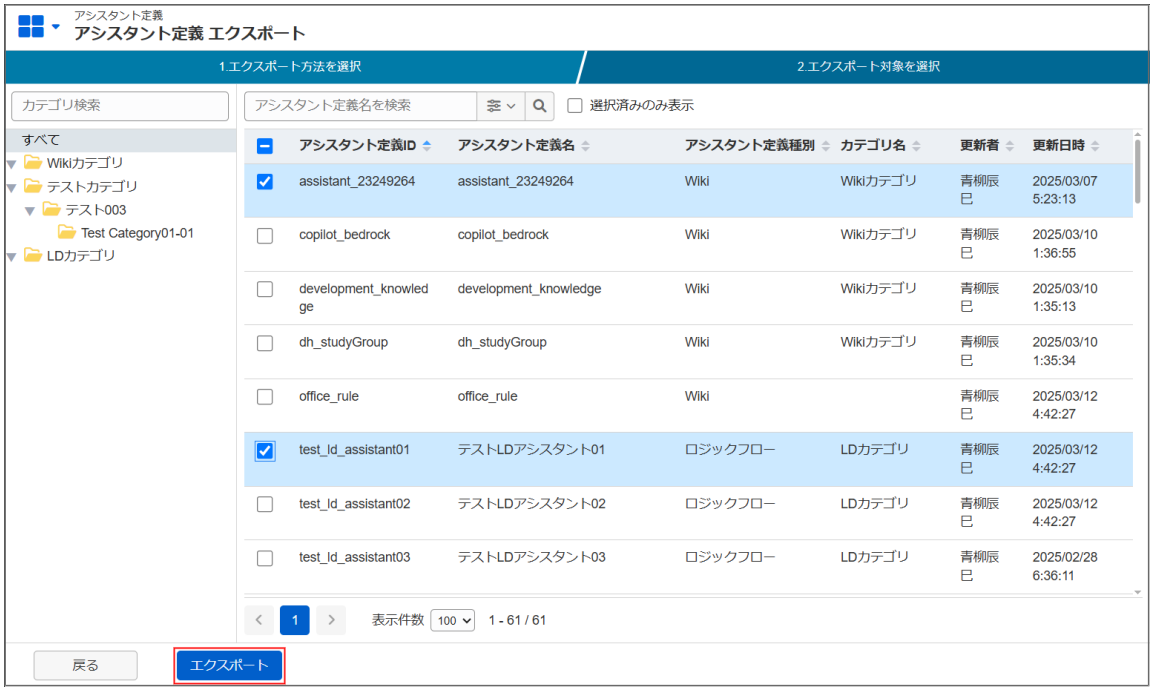
5. 「エクスポート対象を選択」ステップが表示されます。
   アシスタント定義一覧からエクスポートするアシスタント定義を選択します。



＜画面項目＞

項目	説明
カテゴリ検索窓	検索するカテゴリ名を表す文字列（の一部）を入力します。
アシスタント定義名検索窓	検索するアシスタント定義名を表す文字列（の一部）を入力します。
「詳細検索」アイコン	検索するアシスタント定義ID、アシスタント定義名を表す文字列（の一部）、またはアシスタント定義種別を指定します。
「虫眼鏡」アイコン	検索を実行します。
「選択済みのみ表示」チェックボックス	チェックボックスをONにすると選択したアシスタント定義のみが表示されます。
「ページャ」アイコン	一覧の表示ページを切り替えます。
「表示件数」プルダウン	エンティティログの表示件数を設定します。 設定可能は表示件数は、50, 100, 150, 200 件です。
「戻る」ボタン	「エクスポート方法の選択」ステップに遷移します。
「エクスポート」ボタン	エクスポートを実行します。

6. 「エクスポート」ボタンをクリックします。確認ダイアログの「エクスポート」ボタンをクリックします。



7. 「assistant_definition.json」ファイルがダウンロードされます。

IM-LogicDesignerタスク

概要

IM-Copilot の IM-LogicDesignerタスクには、以下のタスクが存在します。

汎用利用を想定したタスク

- チャットタスク
  入力メッセージを通して対話を行うためのタスクです。
- 画像生成タスク
  入力メッセージをもとに画像生成を行うためのタスクです。
- 埋め込みタスク
  入力メッセージをベクトルに変換するためのタスクです。
- 文字起こしタスク
  入力音声ファイルをもとに文字起こしを行うためのタスクです。
- 音声生成タスク
  入力メッセージをもとに音声生成を行うためのタスクです。

> **注意**
>
> Azure OpenAI Service をご利用の場合、音声生成タスクはご利用いただけません。
> また、 Amazon Bedrock をご利用の場合、文字起こしタスク、音声生成タスクはご利用いただけません。

> **コラム**
>
> 各タスクのパラメータには、最小値や最大値などの制限が存在する場合があります。

ロジックフローアシスタントでの利用を想定したタスク

- メッセージ履歴の取得
  クライアントとアシスタントの間でやり取りを行ったメッセージ履歴を取得するためのタスクです。
- クライアントへデータを送信
  任意のデータをクライアントへ送信するためのタスクです。

ベクトルデータベース操作を想定したタスク

- ベクトルデータベースコンテンツ登録タスク
  ベクトルデータベースにコンテンツを登録するためのタスクです。
- ベクトルデータベースコンテンツ削除タスク
  ベクトルデータベースからコンテンツを削除するためのタスクです。
- ベクトルデータベース類似度検索タスク
  ベクトルデータベースからベクトルの類似度が高いコンテンツを検索するためのタスクです。
- ベクトルデータベースキーワード検索タスク
  ベクトルデータベースから指定したキーワードを含むコンテンツを検索するためのタスクです。

> **コラム**
>
> ベクトルデータベースを利用するには、テナント環境セットアップでベクトルデータベース接続情報の設定が必要です。
> 設定内容の詳細については、「テナント環境セットアップ」 - 「ベクトルデータベース接続情報」 を参照してください。

データ処理を想定したタスク

- テキスト抽出タスク
  ファイルバイナリデータからテキストを抽出するためのタスクです。
- テキスト分割タスク
  テキストを指定のサイズに分割するためのタスクです。
- ランク融合タスク
  複数の異なるランキング結果を一つの統合されたランキングにまとめるためのタスクです。

> **コラム**
>
> テキスト抽出タスクは、テキスト抽出にテキスト抽出機能（ND Universal Extractor）を利用します。
> 必要に応じて、テキスト抽出機能のテキスト抽出設定の調整を検討してください。
> 設定内容の詳細については、「設定ファイルリファレンス」 - 「テキスト抽出設定」 を参照してください。

各タスク仕様

各タスク仕様は「IM-LogicDesigner仕様書」- 「タスク一覧（IM-Copilot）」を参照してください。

ログ

IM-Copilot では生成AIへのリクエストに成功した際の詳細をログとして出力させることが可能です。
出力内容を加工、集計することで特定のリクエストに関する統計情報の可視化を行うといった事が可能です。

ログ設定

以下のログ設定ファイルを編集してください。

- %CONTEXT_PATH%/WEB-INF/conf/log/im_logger_copilot_driver.xml
- conf/log/im_logger_copilot_driver.xml （IM-Jugglingプロジェクト内の場合）

ログ仕様

ログ仕様は「ログ仕様書」-「IM-Copilot生成AI連携ドライバログ」を参照してください。

ファイル出力ログ

ファイルへのログ出力はデフォルトで有効化されています。（FileAppender）

データベース出力ログ

データベースへのログ出力はデフォルトで有効化されていません。（DBAppender）
以下を参考に有効化してください。

有効化

ログ設定を変更することでデータベースログを有効化できます。
また、データベース側にLogback仕様のテーブル、シーケンスなどを作成してください。

以下のサンプルを参考にしてください。

- ログ設定サンプル：
  - **im_logger_copilot_driver.xml（Logback_DBAppender_PostgreSQL）**
  - **im_logger_copilot_driver.xml（Logback_DBAppender_SQLServer）**
  - **im_logger_copilot_driver.xml（Logback_DBAppender_Oracle）**
- DDLサンプル：
  - **ddl_sample.sql（Logback_DDL_PostgreSQL）**
  - **ddl_sample.sql（Logback_DDL_SQLServer）**
  - **ddl_sample.sql（Logback_DDL_Oracle）**

> **コラム**
>
> 2024年3月末現在、上記サンプルとログ出力確認済のデータベースバージョンの組合せは以下の通りです。
> ・ PostgreSQL 13
> ・ SQLServer 2022
> ・ Oracle 19c

> **コラム**
>
> Logbackに関する不明点は、Logbackのドキュメント、フォーラムの情報などがヒントになる場合があります。
> https://logback.qos.ch/
>
> デバッグ可能環境の場合は、以下のLogbackソースをデバッグすることで不明点の解消が早まる場合があります。
> ch.qos.logback.core.db.DBAppenderBase.java など

ログ活用

ログ可視化

ログ可視化の例として、IM-LogicDesignerでログを読み込み、加工、ViewCreatorで表示する方法を説明します。

ViewCreator表示

IM-LogicDesigner のロジックフローを ViewCreator の外部データソースとして利用できます。
詳細は「ViewCreator 管理者操作ガイド」-「ロジックフローの利用」を参照してください。

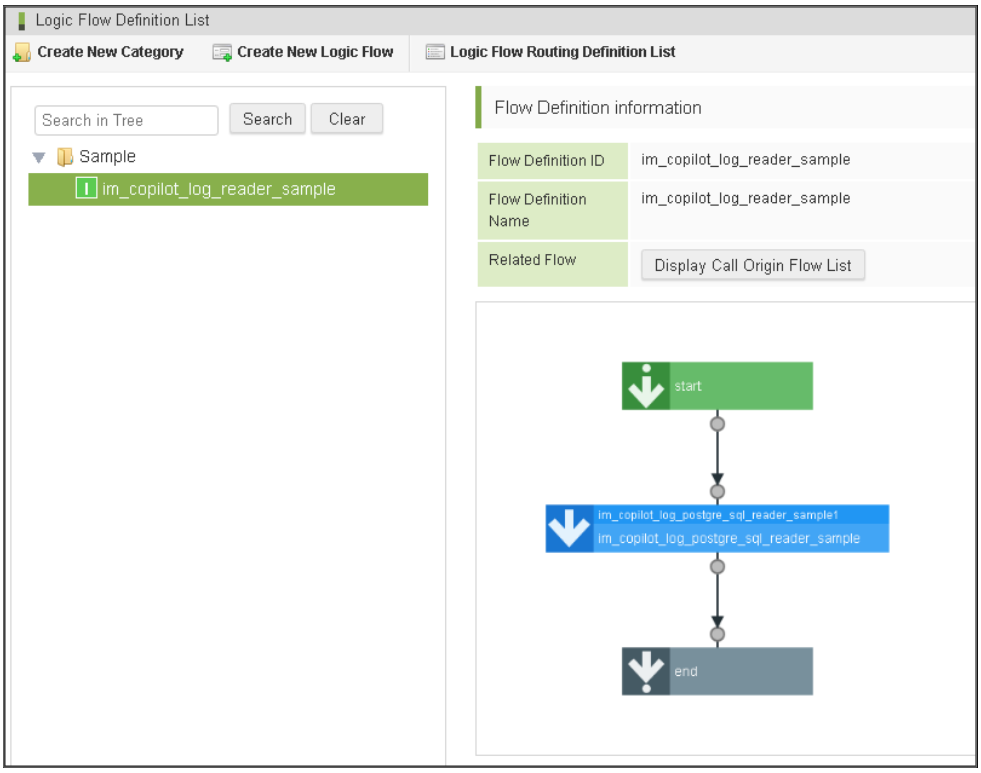
ロジックフロー作成

IM-LogicDesigner でログを取得するロジックフローを作成してください。

ロジックフローには ViewCreatorで利用可能なロジックフローの出力設定が必要です。
詳細は「ViewCreator 管理者操作ガイド」-「ViewCreatorで利用可能なロジックフローの出力設定」を参照してください。

ログ取得方法などについては、以下のロジックフローサンプル（im_logicdesigner-data.zip）を参考にしてください。
ログファイルやデータベースからのログ取得、レスポンスのJSON文字列の変換処理、集計処理などを行っています。

- ロジックフローサンプル：
  im_logicdesigner-data.zip（ログファイル）
  im_logicdesigner-data.zip（データベース：PostgreSQL）
  im_logicdesigner-data.zip（データベース：SQLServer）
  im_logicdesigner-data.zip（データベース：Oracle）
  インポート方法は「IM-LogicDesigner ユーザ操作ガイド」-「インポート/エクスポート」を参照してください。



ロジックフロー管理

ViewCreator でロジックフロー管理を行ってください。
詳細は「ViewCreator 管理者操作ガイド」-「ロジックフロー管理」を参照してください。

クエリ作成

ViewCreator でクエリの作成を行ってください。
詳細は「ViewCreator 管理者操作ガイド」-「クエリの作成」を参照してください。

データ参照作成

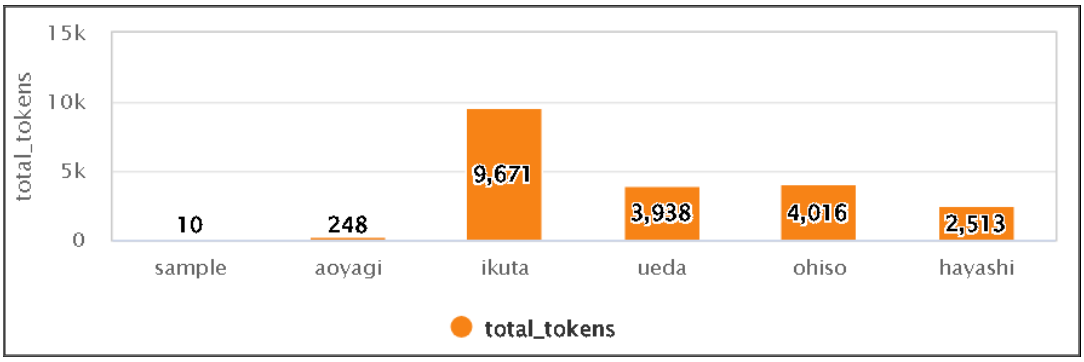
ViewCreator でデータ参照の作成を行ってください。
詳細は「ViewCreator 管理者操作ガイド」-「データ参照の作成」を参照してください。

以下のデータ参照サンプル（viewcreator-data.zip）も参考にしてください。

- ViewCreator データ参照サンプル：

  viewcreator-data.zip

  インポート方法は「ViewCreator 管理者操作ガイド」-「インポート・エクスポート」を参照してください。



ロジックフローアシスタントの作成例

このセクションでは、ロジックフローアシスタントの作成例について説明します。

作成するロジックフローアシスタントの概要

このセクションでは、IM-LogicDesigner を用いてアシスタントを作成する手順を解説します。
手順に従って進めることで、ストレージ内のファイルを情報源としてユーザの質問に回答するアシスタントを構築できます。

本ドキュメントでは、説明の都合上情報源をストレージ内のファイルに限定していますが、ローコード開発により変更が可能です。
例えば、このドキュメントを参考に作成したロジックフローアシスタントを変更することで、外部システムや独自のデータベースから情報を取得するように変更することもできます。

目次

作成するロジックフローアシスタントの利用想定

本ドキュメントで作成するロジックフローアシスタントは、指定されたストレージ内のファイルを検索し、その内容をもとに適切な回答を提供します。例えば、社内ドキュメント検索やナレッジベースの自動応答システムとして活用できます。

主な用途

- 社内マニュアルや手順書の検索・回答
- 製品仕様書の照会
- FAQ の自動応答システム

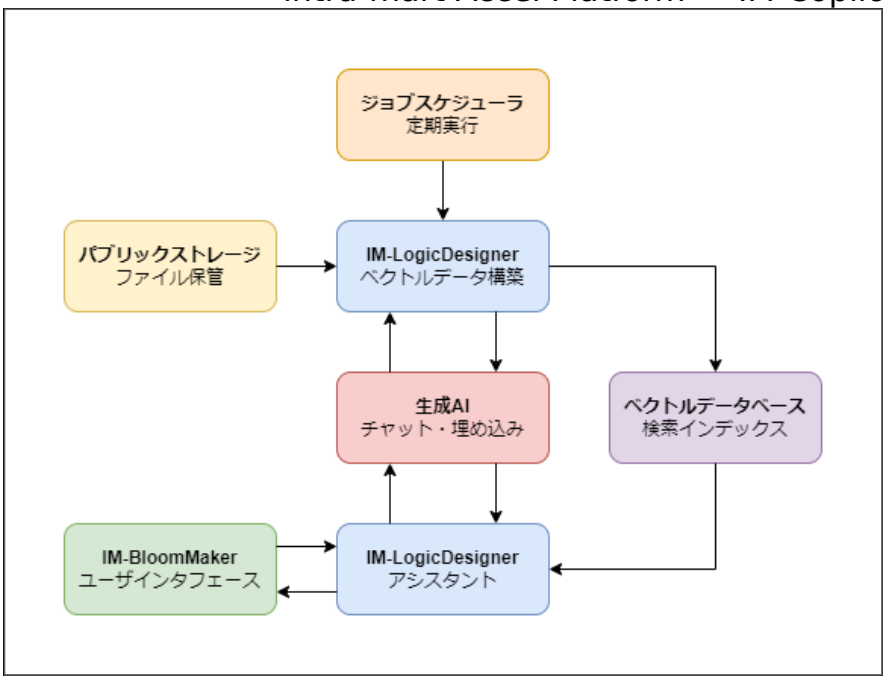
前提知識

本ガイドを理解するために、以下の知識があるとスムーズに進められます。

- IM-LogicDesigner の基本操作
  - IM-LogicDesigner の基本操作については「IM-LogicDesigner ユーザ操作ガイド」を参照してください。
- IM-BloomMaker の基本操作
  - IM-BloomMaker の基本操作については「IM-BloomMaker for Accel Platform ユーザ操作ガイド」を参照してください。
- RAG（Retrieval Augmented Generation）の概念

作成するロジックフローアシスタントの概要図

以下の図は、作成するロジックフローアシスタントの基本的な動作を示しています。



主要コンポーネント

機能名	説明
IM-LogicDesigner	ベクトルデータの構築処理、および、アシスタントのロジックを管理
IM-BloomMaker	チャット画面を提供
パブリックストレージ	アシスタントが情報源として参照するファイルを格納
ベクトルデータベース	情報検索のためのデータ管理
生成AI	自然な回答生成、ベクトル情報（埋め込み）の生成
ジョブスケジューラ	ベクトルデータの定期的なデータ更新処理

事前設定

生成AIの設定

作成するロジックフローアシスタントでは生成AIを活用します。利用するためには、以下の設定を事前に行う必要があります。

1. 生成AIサービスのセットアップ
   一つ以上の生成AIサービスがセットアップされている必要があります。
   セットアップ（生成AI） を参考にセットアップを行ってください。

2. 生成AI連携ドライバ設定
   セットアップ済みの生成AIサービスと連携するための設定です。
   生成AI連携ドライバ設定 を参考に生成AIドライバの設定を行ってください。

3. 生成AI連携アクション設定
   本アシスタントでは IM-LogicDesigner タスク「チャット」、「埋め込み」を利用します。
   これらのタスクに対応する「チャット」、「埋め込み」のアクションを設定する必要があります。
   生成AI連携アクション設定 を参考に生成AIアクションの設定を行ってください。

テキスト抽出設定

作成するロジックフローアシスタントでは、パブリックストレージ内のファイルのテキスト抽出にテキスト抽出機能（ND Universal Extractor）を利用します。
テキスト抽出設定では、テキスト抽出対象とするファイルについて設定が可能です。標準設定においてもテキスト抽出は可能ですが、必要に応じて設定を調整してください。
設定内容の詳細については、「設定ファイルリファレンス」 - 「テキスト抽出設定」 を参照してください。

ベクトルデータベースの設定

作成するロジックフローアシスタントでは、パブリックストレージ内のファイルを検索可能な形で管理するためにベクトルデータベースを使用します。
以下を参考にベクトルデータベースの設定を行ってください。

- 「テナント環境セットアップ」 - 「ベクトルデータベース接続情報」

パブリックストレージへの情報源となるファイルの配置

作成するロジックフローアシスタントでは、パブリックストレージ内の *sample_rag_assistant/files* ディレクトリ配下を検索対象として利用します。
ベクトルデータ構築を実行する前に、情報源とするファイルを配置してください。
パブリックストレージにファイルを配置する手段は以下の通りです。

- テナント管理機能のファイル操作を使用する
  - 「テナント管理者操作ガイド」-「ファイル操作を使用する」
- IM-LogicDesigner のタスク「パブリックストレージ取得」タスクを使用する
  - 「IM-LogicDesigner仕様書」-「パブリックストレージ取得」

構築ステップ

本ドキュメントでは、以下のステップに従ってアシスタントを構築します。

1. **ベクトルデータの作成**
   - パブリックストレージ内のファイルをベクトル化し、検索に利用できるように準備します。
   - ベクトル情報を登録するロジックフローを作成し、ジョブスケジューラで定期的に実行するように設定します。
2. **アシスタントの実装**
   - ユーザの質問を処理し、適切な回答を生成するロジックフローを作成します。
   - アシスタント定義を作成し、ロジックフローをアシスタントとして動かせるように設定します。
3. **画面の作成**
   - ユーザがアシスタントを利用できる UI を作成します。

これらの手順に沿って進めることで、アシスタントを作成できます。

本アシスタント資材のダウンロード

本手順に従い作成したアシスタントの資材は、以下からダウンロードできます。

アシスタントの資材

> **コラム**
> 本資材は、アシスタントの作成手順を解説するためのサンプル資材です。アシスタントの動作確認や追加開発のためにご利用ください。

zip ファイルには以下のファイルが含まれています。

- im_logicdesigner-data.zip
  - IM-LogicDesigner のインポートファイル
- assistant_definition.json
  - アシスタント定義のインポートファイルです
- im_bloommaker-data.zip
  - IM-BloomMaker のインポートファイルです
- job-scheduler.xml
  - ジョブスケジューラのインポートファイルです

zip ファイルを解凍し、以下の順で資材をインポートしてください。

1. IM-LogicDesigner
2. アシスタント定義
3. IM-BloomMaker

4. ジョブスケジューラ

> **コラム**
> 本インポート資材には認可設定が含まれていません。インポート実施後に「アシスタント定義」、「IM-BloomMaker ルーティング定義一覧」から認可設定を行ってください。

IM-LogicDesigner 資材のインポート

IM-LogicDesigner のインポート手順については以下を参照してください。

- 「IM-LogicDesigner ユーザ操作ガイド」 - 「インポートを行う」

アシスタント定義のインポート

アシスタント定義のインポート手順については以下を参照してください。

- 「*アシスタント定義のインポート*」

IM-BloomMaker 資材のインポート

IM-BloomMaker のインポート手順については以下を参照してください。

- 「IM-BloomMaker for Accel Platform ユーザ操作ガイド」 - 「定義ファイルをインポートする」

ジョブスケジューラ資材のインポート

ジョブスケジューラ資材のインポートは、ジョブを利用して実施します。以下で手順を確認してください。

1. パブリックストレージ直下に *job-scheduler.xml* を配置します。
   以下を参考にファイルを配置してください。
   - 「テナント管理者操作ガイド」-「ファイル操作を使用する」
2. ジョブを確認します。
   「サイトマップ」→「テナント管理」→「ジョブ管理」→「ジョブ設定」をクリックし「ジョブ管理」画面を表示します。
   左側のジョブ一覧から「テナントマスタ」→「インポート」→「ジョブインポート」を選択クリックします。
   実行時の情報内の実行パラメータのキー *file* が *job-scheduler.xml* であることを確認します。
3. ジョブネットを確認・実行します。
   「サイトマップ」→「テナント管理」→「ジョブ管理」→「ジョブネット設定」をクリックし「ジョブネット管理」画面を表示します。
   左側のジョブネット一覧から「テナントマスタ」→「インポート」→「ジョブインポート」を選択クリックします。
   実行時の情報内の実行パラメータのキー *file* 存在しないことを確認します。
   下部の「即時実行」ボタンをクリックして「ジョブインポート」ジョブネットを実行します。

ロジックフローアシスタント構築手順

ベクトルデータ構築ジョブの作成

作成するロジックフローアシスタントでは、ストレージ内のファイルを情報源とするため、ファイルの内容をベクトル化し、ベクトルデータベースに登録する必要があります。
本セクションでは、ベクトルデータを構築するためのロジックフローを作成する手順、および、ジョブスケジューラを利用した自動実行の設定方法を説明します。

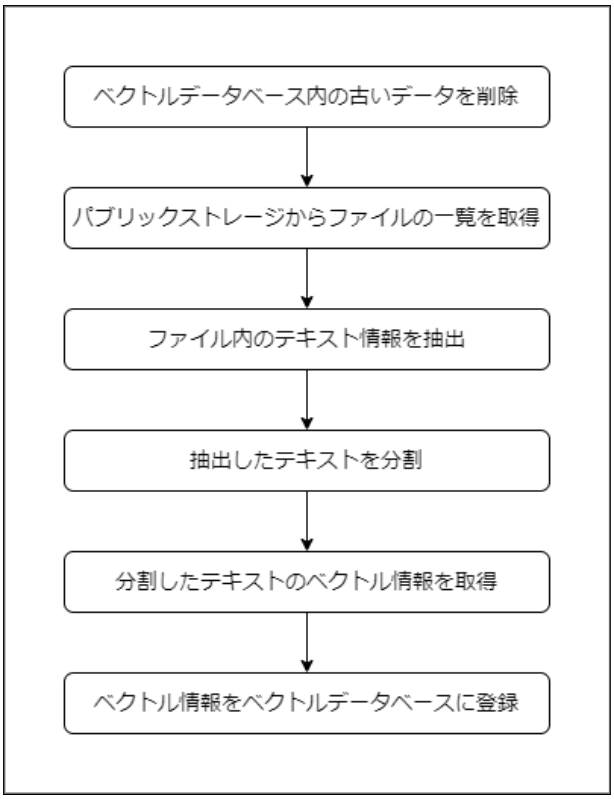
目次

ベクトルデータの構築を行うロジックフローの作成

IM-LogicDesigner を使用して、ベクトルデータを定期的に構築・更新するロジックフローを作成します。

作成するロジックフローの概要

本ロジックフローは、以下の手順でベクトルデータを構築します。



ベクトルデータの作成処理は、埋め込み処理でサーバ通信が発生する API を利用するため時間がかかります。
本ドキュメントでは、説明の都合上既存のベクトルデータを削除した後に新たなベクトルデータを構築する手順を記載しています。
そのため、構築が完了するまでの間はアシスタントの情報源が一部欠落します。

なお、ロジックフローの変更を行うことでこの参照不可時間を短縮する処理に変更することも可能です。

- 変更例
  1. 登録するベクトルデータを事前に作成
  2. ベクトルデータベースから既存のベクトルデータを削除
  3. 事前に作成したベクトルデータをベクトルデータベースに登録

設定手順

IM-LogicDesigner を利用して、ベクトルデータを構築するロジックフローを作成する手順を説明します。
本説明で使用しているロジックフローのキャプチャ画像は、一部タスク・エレメントのラベルの値を変更しています。

各種定義の設定

1. ロジックフロー新規作成画面を開きます。
   「サイトマップ」→「LogicDesigner」→「フロー定義一覧」をクリックし、「ロジックフロー定義一覧」画面を表示します。
   ロジックフロー定義一覧画面ツールバーの「ロジックフロー新規作成」をクリックします。

2. 定数の設定を行います。
   「ロジックフロー定義編集」画面ツールバーの「定数設定」をクリックして定数設定ダイアログを表示します。
   以下の定数を設定します。

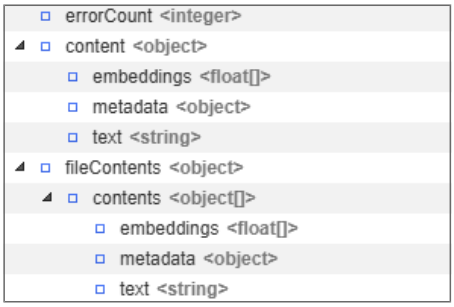
定数一覧

定数ID	定数値	説明
ERROR_MESSAGE	*The embedding process failed.*	埋め込み処理でエラーが発生した場合に利用するメッセージです。
ONE	*1*	1を表す定数です。カウントのインクリメントに利用します。
RETRY_COUNT	*3*	埋め込みエラー時にリトライする回数です。
TARGET_STORAGE_PATH	*sample_rag_assistant/files*	参照情報とするファイルが格納されているパブリックストレージのディレクトリです。
VECTORSTORE_CATEGORY	*sample_rag_assistant*	ベクトルデータベースのカテゴリです。

3. 変数の設定を行います。
   「ロジックフロー定義編集」画面ツールバーの「変数設定」をクリックして変数設定ダイアログを表示します。
   以下の変数を設定します。

変数一覧

変数	型	説明
errorCount	integer	埋め込み処理が連続して失敗した際に利用するエラー回数です。
content	object	ベクトルデータベースに登録する 1 コンテンツを格納します。
content.embeddings	float[]	
content.metadata	object	
content.text	string	
fileContents	object	ベクトルデータベースに登録する 1 ファイルに含まれるコンテンツを格納します。
fileContents.contents	object[]	
fileContents.contents.embeddings	float[]	
fileContents.contents.metadata	object	
fileContents.contents.text	string	

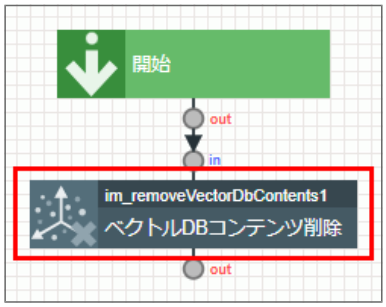


JSON 入力で設定する場合は以下の JSON をコピーし、変数設定ダイアログの「JSON入力」をクリックしてサンプル JSON の入力ダイアログに貼り付けてください。
JSON 入力を利用して変数を設定した場合、*errorCount* のデータ型が *double* に、*content.embeddings*、および、*fileContents.contents.embeddings* のデータ型が *double[]* に設定されます。
errorCount のデータ型を *integer* に、*content.embeddings*、および、*fileContents.contents.embeddings* のデータ型を *float[]* に手動で変更してください。

```
{
  "errorCount": 0,
  "content": {
    "embeddings": [
      0
    ],
    "metadata": {},
    "text": ""
  },
  "fileContents": {
    "contents": [
      {
        "embeddings": [
          0
        ],
        "metadata": {},
        "text": ""
      }
    ]
  }
}
```

ベクトルデータベース内の古いデータの削除

4. ベクトルデータの削除タスクを追加します。

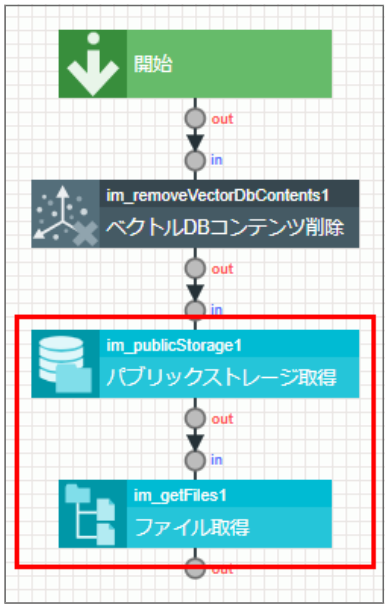


既に存在するベクトルデータを削除するために対象のカテゴリのベクトルデータを削除するタスクを追加します。
パレットから「IM-Copilot」→「ベクトルDBコンテンツ削除」をクリックしロジックフローに追加します。
「開始」エレメントの *out* を追加した「ベクトルDBコンテンツ削除」の *in* に接続します。
追加した「ベクトルDBコンテンツ削除」を選択し、マッピング設定を行います。

- 定数 *VECTORSTORE_CATEGORY* をタスクの入力値 *category* に設定します。

パブリックストレージからファイルの一覧を取得

5. パブリックストレージのファイル一覧を取得するタスクを追加します。

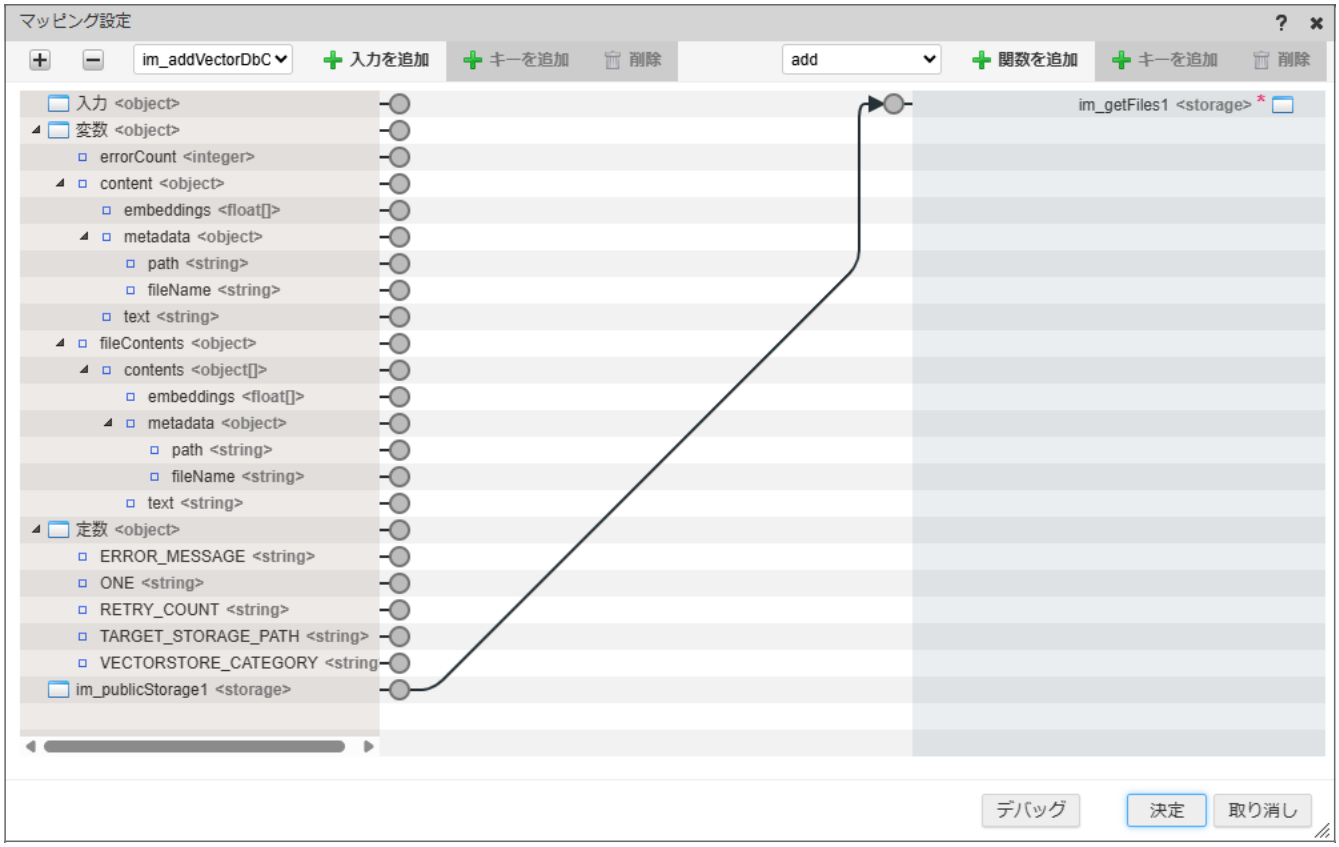


生成 AI が回答を作成する際に情報源とするデータをパブリックストレージから取得します。
パブリックストレージ内のファイル一覧を取得するためのタスクを追加します。
パレットから「ストレージ操作」→「パブリックストレージ取得」をクリックしロジックフローに追加します。
「ベクトルDBコンテンツ削除」タスクの *out* を追加した「パブリックストレージ取得」の *in* に接続します。
追加した「パブリックストレージ取得」を選択し、マッピング設定を行います。

- 定数 *TARGET_STORAGE_PATH* をタスクの入力値に設定します。

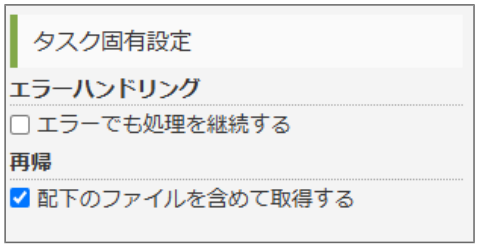
続いて、パレットから「ストレージ操作」→「ファイル取得」をクリックしロジックフローに追加します。
「パブリックストレージ取得」タスクの *out* を追加した「ファイル取得」の *in* に接続します。
追加した「ファイル取得」を選択し、マッピング設定を行います。

- エイリアス「*im_publicStorage1*」を追加しタスクの入力値に設定します。

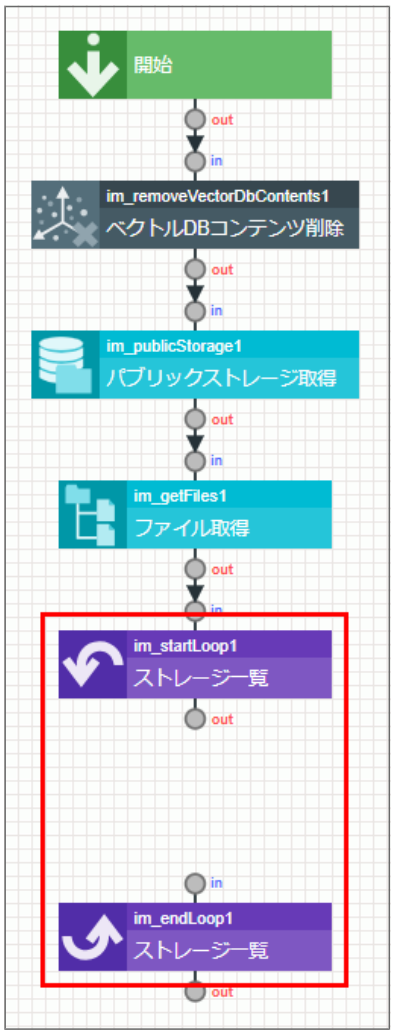


続いてファイル取得タスクのタスク固有設定を行います。

- 配下のファイルを含めて取得するを有効にします。



6. 繰り返し制御エレメントを追加します。

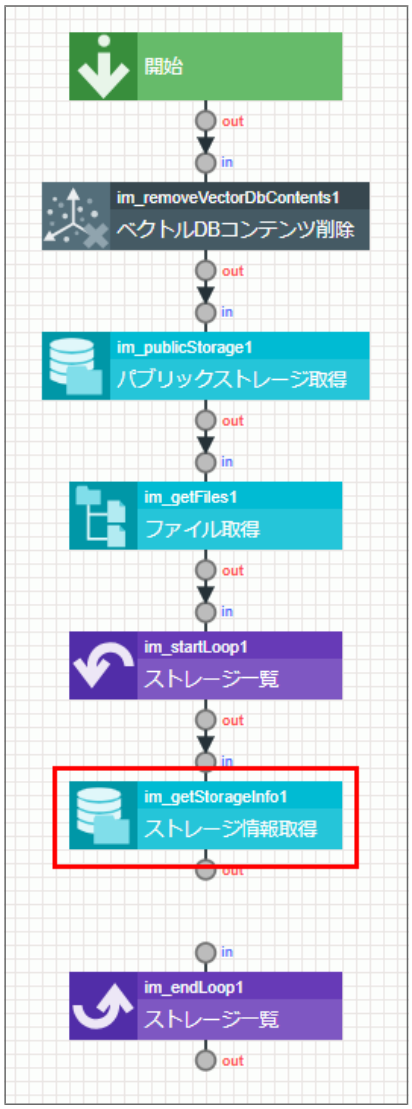


各ファイルの処理を行うために、繰り返し制御エレメントを追加します。
パレットから「基本」→「繰り返し開始」をクリックしロジックフローに追加します。
「ファイル取得」タスクの *out* を追加した「繰り返し開始」の *in* に接続します。
追加した「繰り返し」を選択し、タスク固有設定を行います。

- 初期化する変数名に *fileContents* を設定します。
  - これにより、各ファイルの処理を行う際に、ファイルの内容を格納する変数を初期化します。
- 繰り返し対象に *im_getFiles1* を設定します。
  - これにより、取得したファイル一覧を繰り返し処理します。

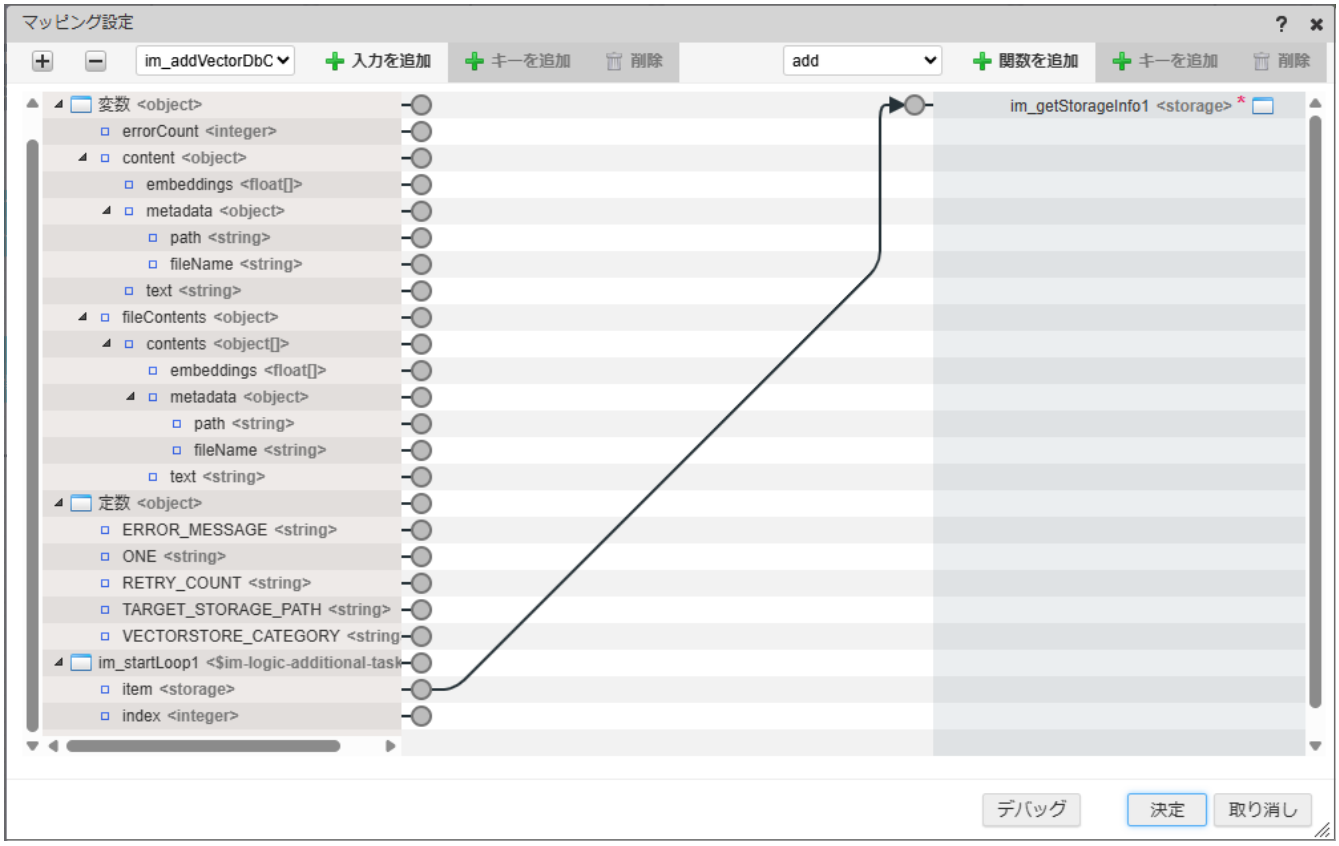
ファイル内のテキスト情報を抽出

7. ストレージ情報取得タスクを追加します。

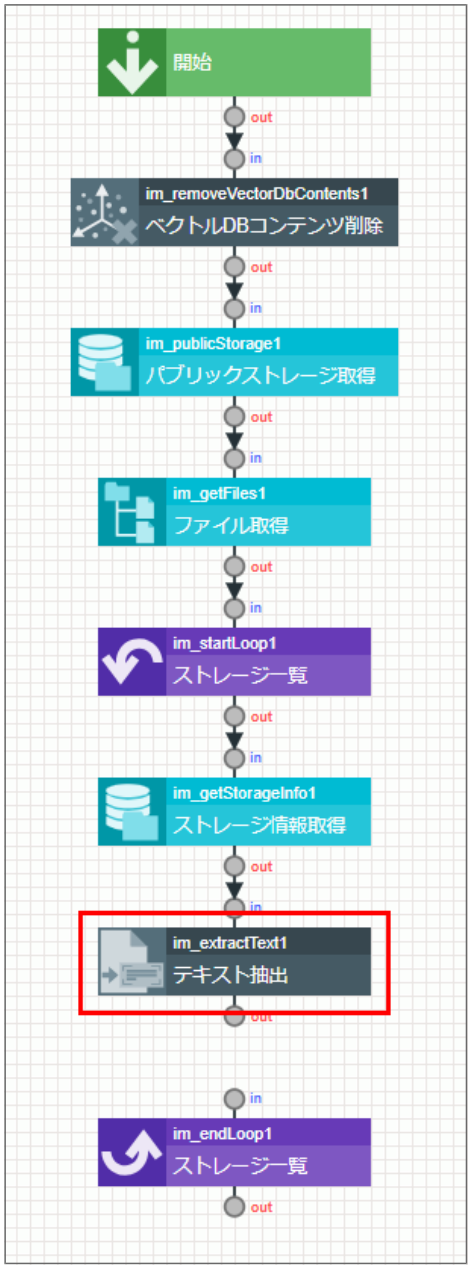


ファイル情報を取得するためのタスクを追加します。取得したファイル情報はテキスト抽出タスクの入力値として利用します。
パレットから「ストレージ操作」→「ストレージ情報取得」をクリックしロジックフローに追加します。
「繰り返し開始」エレメントの *out* を追加した「ストレージ情報取得」の *in* に接続します。
追加した「ストレージ情報取得」を選択し、マッピング設定を行います。

- エイリアス「*im_startLoop1*」を追加します。
  - *item* をタスクの入力値に設定します。

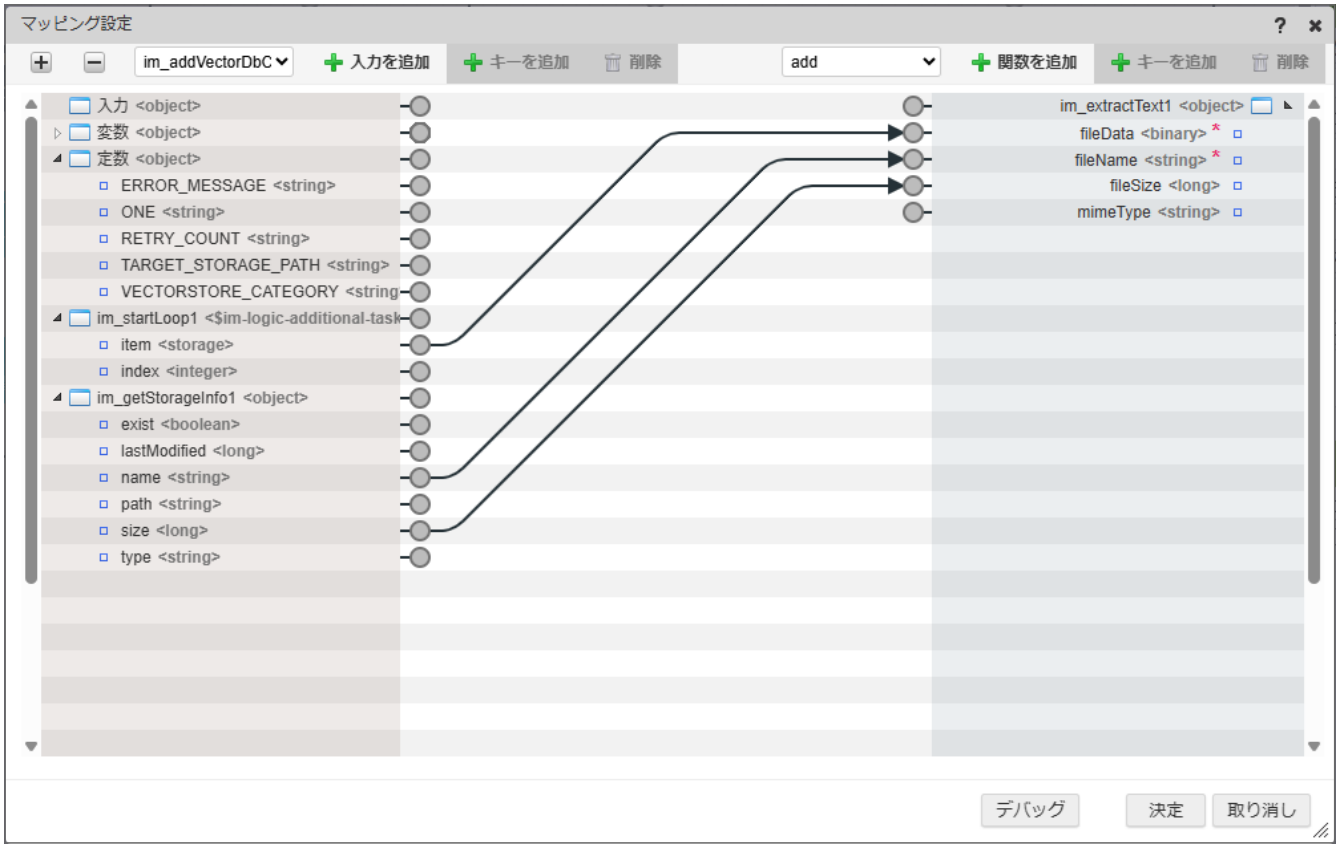


8. テキスト抽出タスクを追加します。



ファイルからテキスト情報を抽出するためのタスクを追加します。
パレットから「IM-Copilot」→「テキスト抽出」をクリックしロジックフローに追加します。
「ストレージ情報取得」タスクの *out* を追加した「テキスト抽出」の *in* に接続します。
追加した「テキスト抽出」を選択し、マッピング設定を行います。

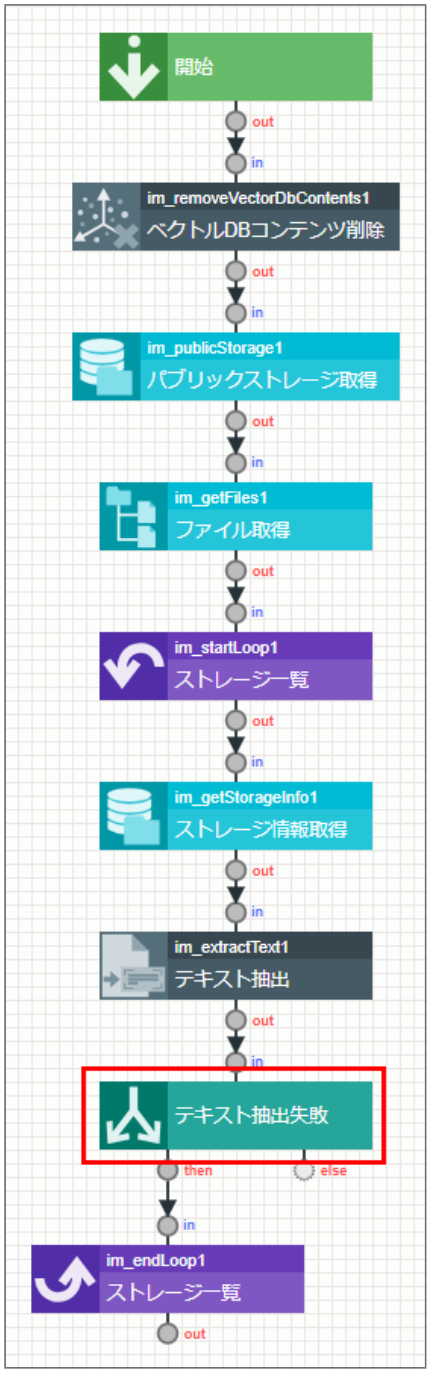
- エイリアス「*im_startLoop1*」を追加します。
  - *item* をタスクの入力値 *fileData* に設定します。
- エイリアス「*im_getStorageInfo1*」を追加します。
  - *name* をタスクの入力値 *fileName* に設定します。
  - *size* をタスクの入力値 *fileSize* に設定します。



続いて「テキスト抽出」タスクのタスク固有設定を行います。

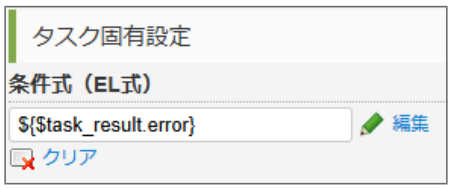
- エラーでも処理を続行するを有効にします。

9. テキスト抽出でエラーが発生した場合の処理を追加します。



テキスト抽出ではテキスト抽出設定の内容に準じてテキストを抽出します。テキスト抽出タスクにテキスト抽出が行えないファイルを指定した場合、エラーが発生します。
エラーが発生した場合に次のファイルの処理を行うための制御を追加します。
パレットから「基本」→「分岐」をクリックしロジックフローに追加します。
「テキスト抽出」タスクの *out* を追加した「分岐」の *in* に接続します。
追加した「分岐」を選択し、タスク固有設定を行います。

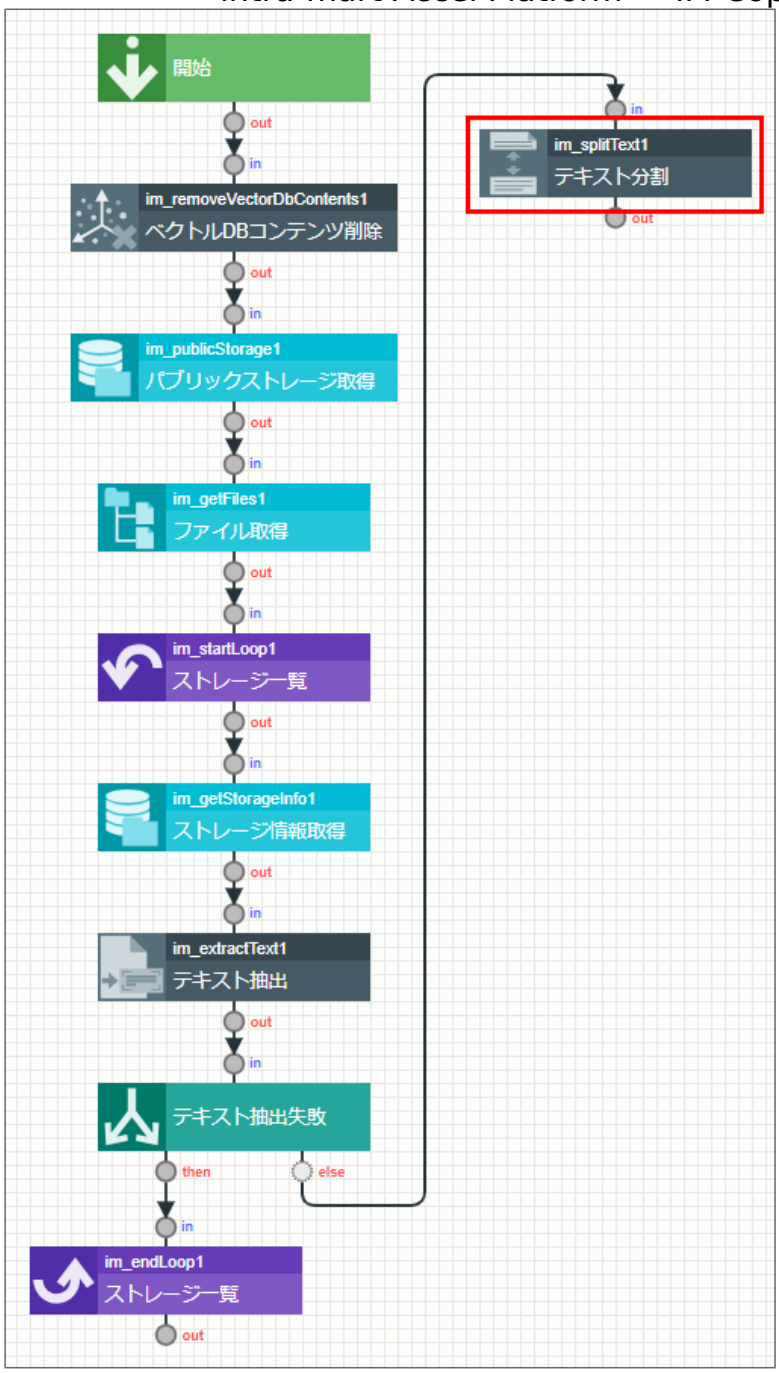
- 条件式（EL式）に *${$task_result.error}* を設定します。
  - *$task_result* は処理結果情報です。直前の処理の処理結果が格納されます。*error* にはエラーが発生した場合に *true* が設定されます。



エラーが発生した場合に次のファイルの処理を行うためのシーケンスを追加します。
追加した「分岐」の *else* に「繰り返し終了」エレメント（*im_endLoop1*）の *in* を接続します。

抽出したテキストを分割

10. テキスト分割タスクを追加します。

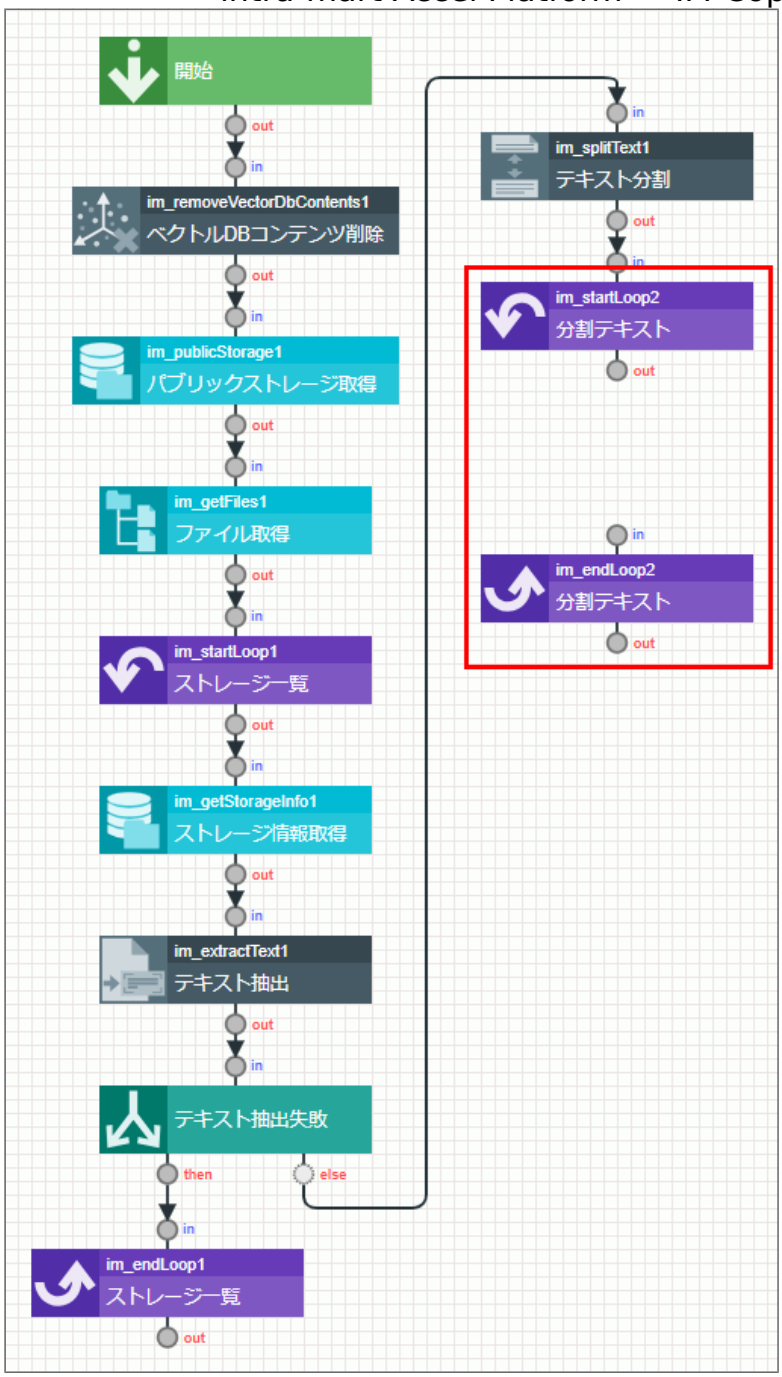


抽出したテキストを分割するためのタスクを追加します。
パレットから「IM-Copilot」→「テキスト分割」をクリックしロジックフローに追加します。
「分岐」エレメントの *else* を追加した「テキスト分割」の *in* に接続します。
追加した「テキスト分割」を選択し、マッピング設定を行います。

- エイリアス「*im_extractText1*」を追加しタスクの入力値 *text* に設定します。

分割したテキストをベクトル化

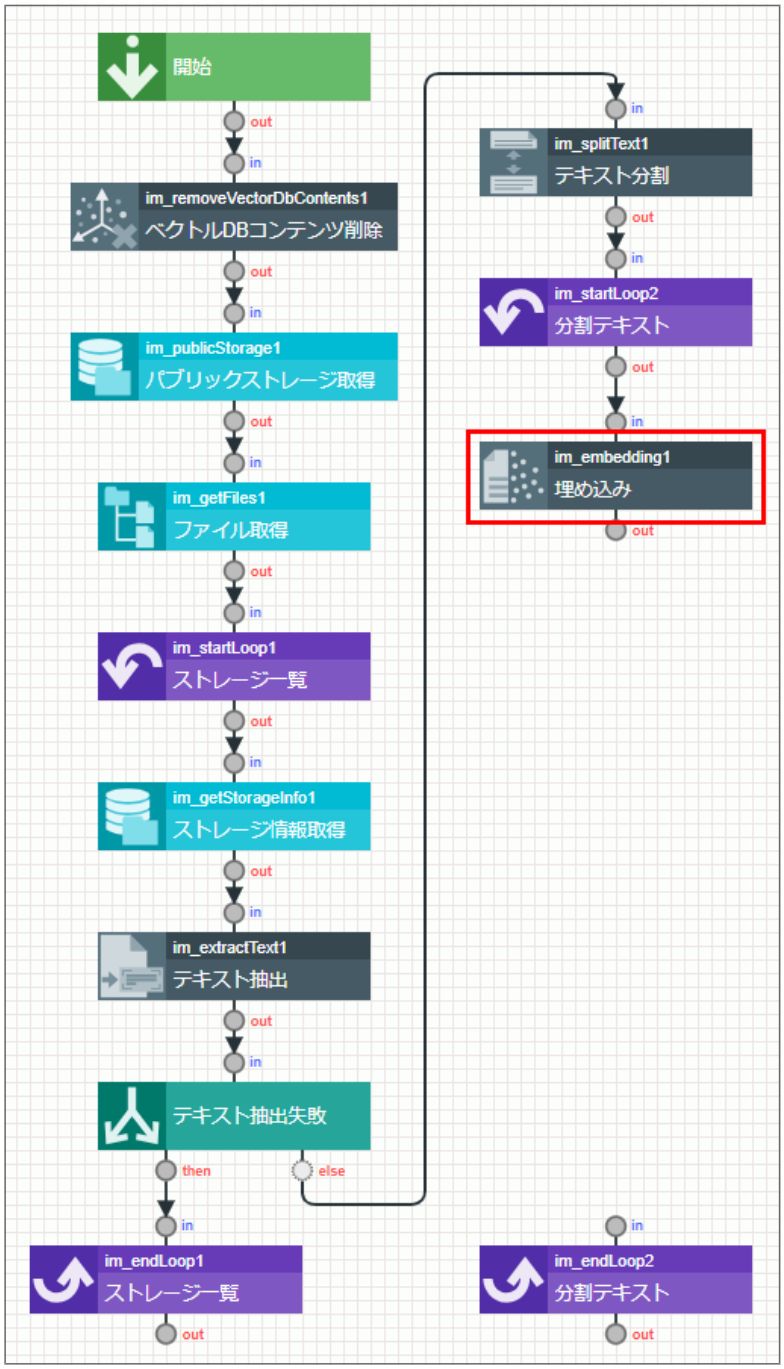
11. 繰り返し制御エレメントを追加します。



テキスト分割した情報をベクトル化するための繰り返し制御エレメントを追加します。
パレットから「基本」→「繰り返し開始」をクリックしロジックフローに追加します。
「テキスト分割」タスクの *out* を追加した「繰り返し開始」の *in* に接続します。
追加した「繰り返し」を選択し、タスク固有設定を行います。

- 初期化する変数名に *content*、 *errorCount* を設定します。
  - これにより、各テキストのベクトル化処理を行う際に、ベクトルデータを格納する変数を初期化します。また、埋め込み処理エラー回数を初期化します。
- 繰り返し対象に *im_splitText1* を設定します。
  - これにより、分割したテキスト情報を繰り返し処理します。

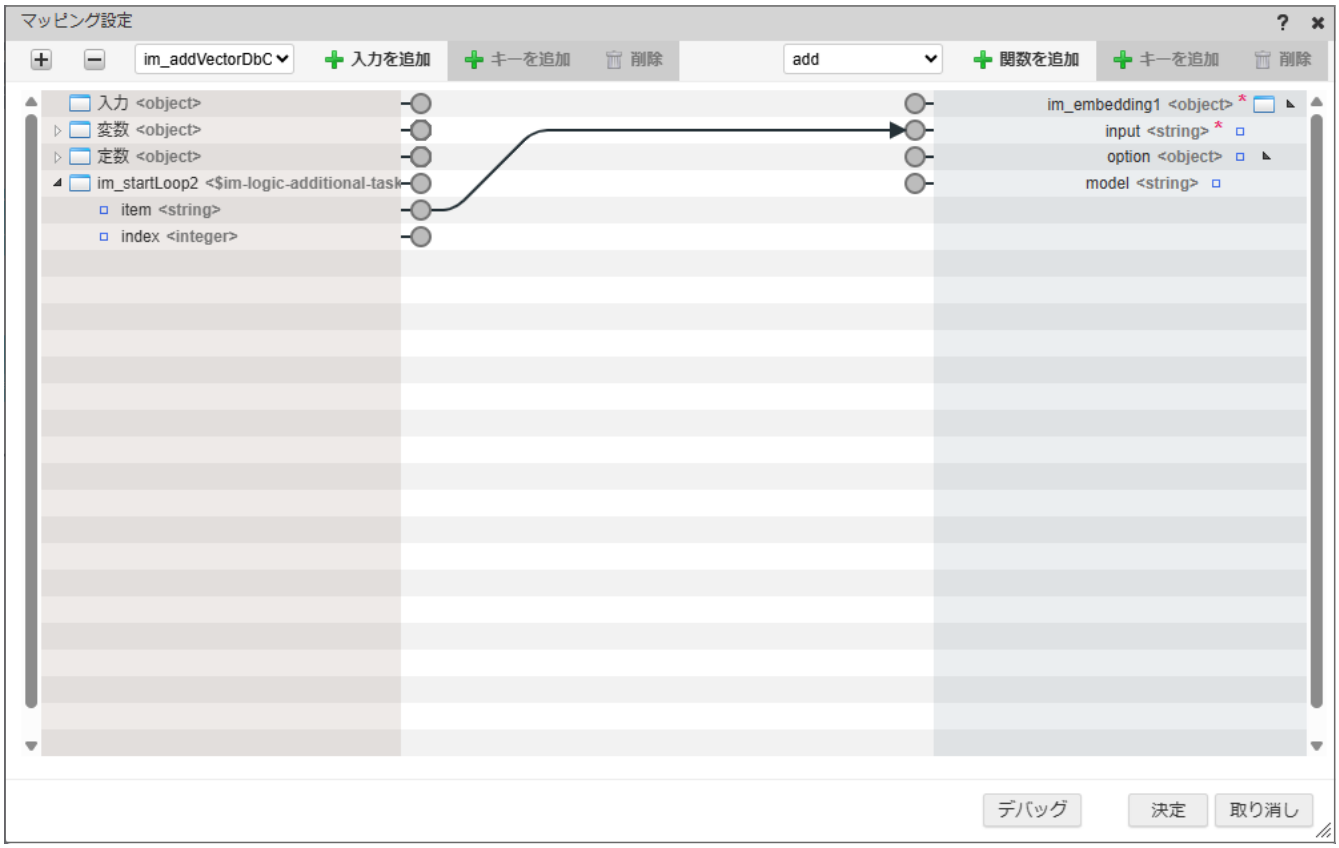
12. 埋め込みタスクを追加します。



テキスト情報をベクトル化するためのタスクを追加します。
パレットから「IM-Copilot」→「埋め込み」をクリックしロジックフローに追加します。
「繰り返し開始」エレメントの *out* を追加した「埋め込み」の *in* に接続します。

追加した「埋め込み」を選択し、マッピング設定を行います。

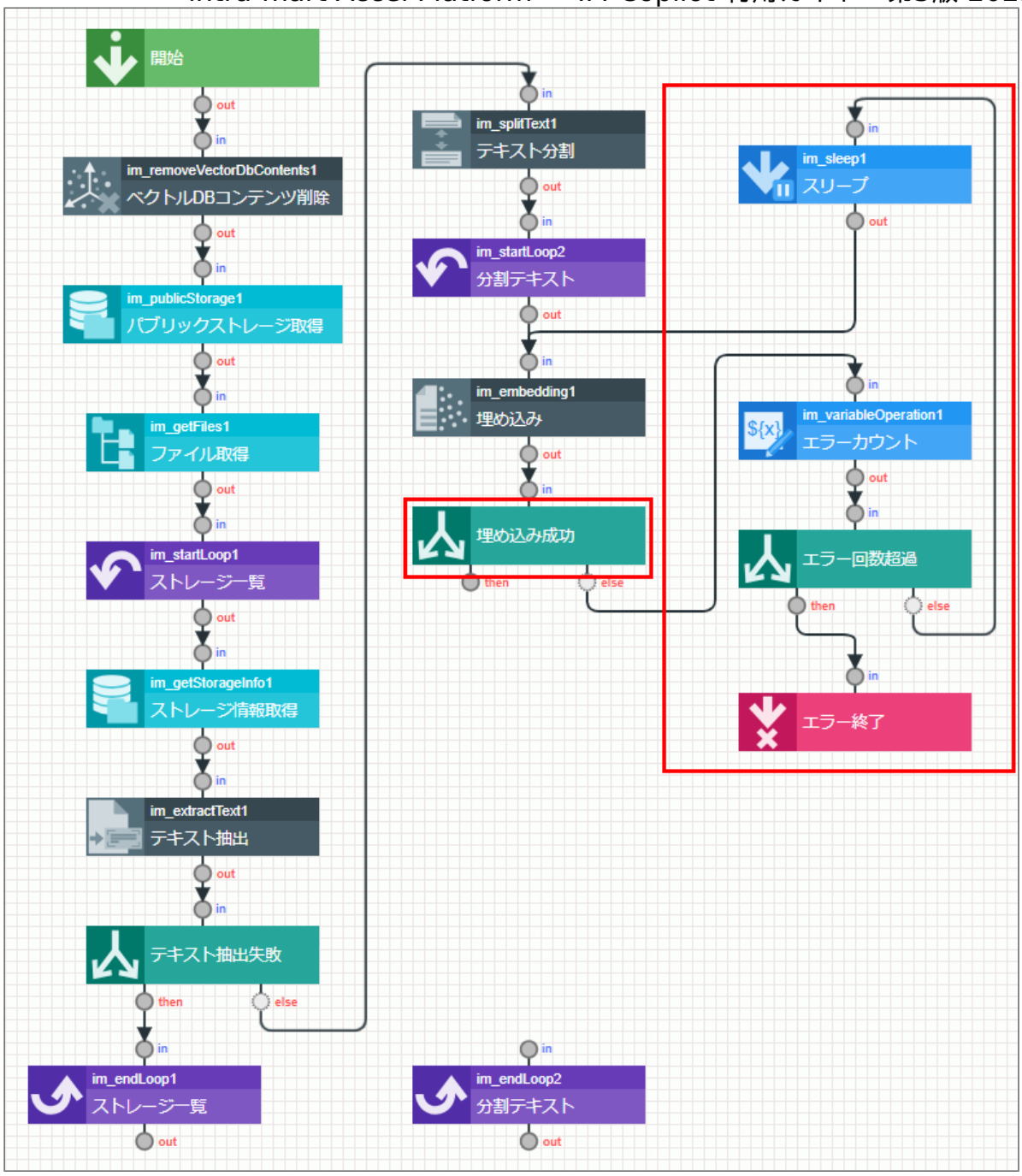
- エイリアス「*im_startLoop2*」を追加します。
  - *item* をタスクの入力値 *input* に設定します。



続いて「埋め込み」タスクのタスク固有設定を行います。

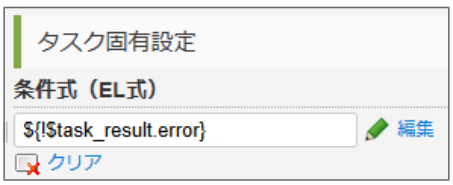
- エラーでも処理を続行するを有効にします。

13. 埋め込みでエラーが発生した場合の処理を追加します。



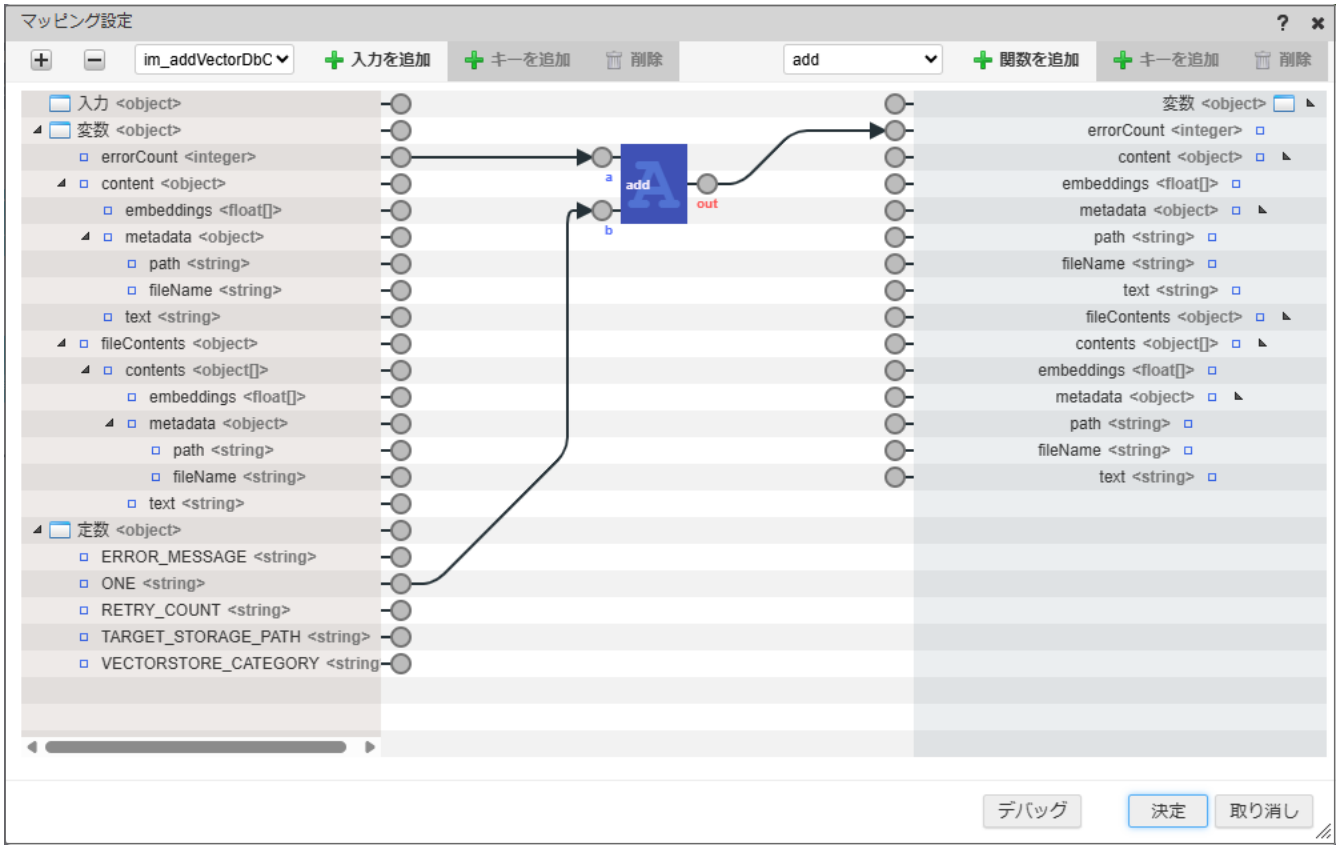
埋め込み処理は AI サービスへのリクエストを行う処理です。本フローでは繰り返し処理で連続して埋め込み処理を行うためレート制限などのエラーが発生する可能性があります。
そのため、エラーが発生した場合にリトライ処理を行うための制御を追加します。
パレットから「基本」→「分岐」をクリックしロジックフローに追加します。
「埋め込み」タスクの *out* を追加した「分岐」の *in* に接続します。
追加した「分岐」を選択し、タスク固有設定を行います。

- 条件式（EL式）に *${!$task_result.error}* を設定します。
  - *$task_result* は処理結果情報です。直前の処理の処理結果が格納されます。*error* にはエラーが発生した場合に *true* が設定されます。
  - *!* は否定を表します。エラーが発生していない場合に *true* になるように設定します。



続いて、パレットから「基本」→「変数操作」をクリックしロジックフローに追加します。
「分岐」エレメント（*im_gateway1*）の *else* に追加した「変数操作」の *in* を接続します。
エラーカウントをインクリメントします。
追加した「変数操作」を選択し、マッピング設定を行います。

- 数値演算マッピング関数 *add* を追加します。
  - 入力値 *a* に変数 *errorCount* を設定します。
  - 入力値 *b* に定数 *ONE* を設定します。
  - 出力値をタスクの入力値 *errorCount* に設定します。



続いて、パレットから「基本」→「分岐」をクリックしロジックフローに追加します。
「変数操作」タスクの *out* を追加した「分岐」の *in* に接続します。
追加した「分岐」を選択し、タスク固有設定を行います。

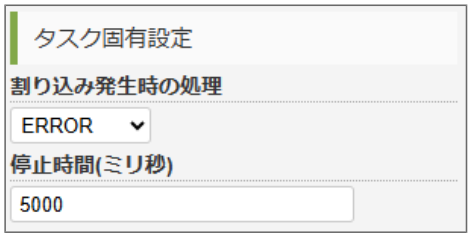
- 条件式（EL式）に *${$variable.errorCount>=$const.RETRY_COUNT}* を設定します。
  - *errorCount* が *RETRY_COUNT* 以上の場合に *true* になるように設定します。

続いて、パレットから「基本」→「エラー終了」をクリックしロジックフローに追加します。
「分岐」エレメント（*im_gateway2*）の *then* に追加した「エラー終了」の *in* を接続します。
追加した「エラー終了」を選択し、タスク固有設定を行います。

- エラーメッセージに *${$const.ERROR_MESSAGE}* を設定します。

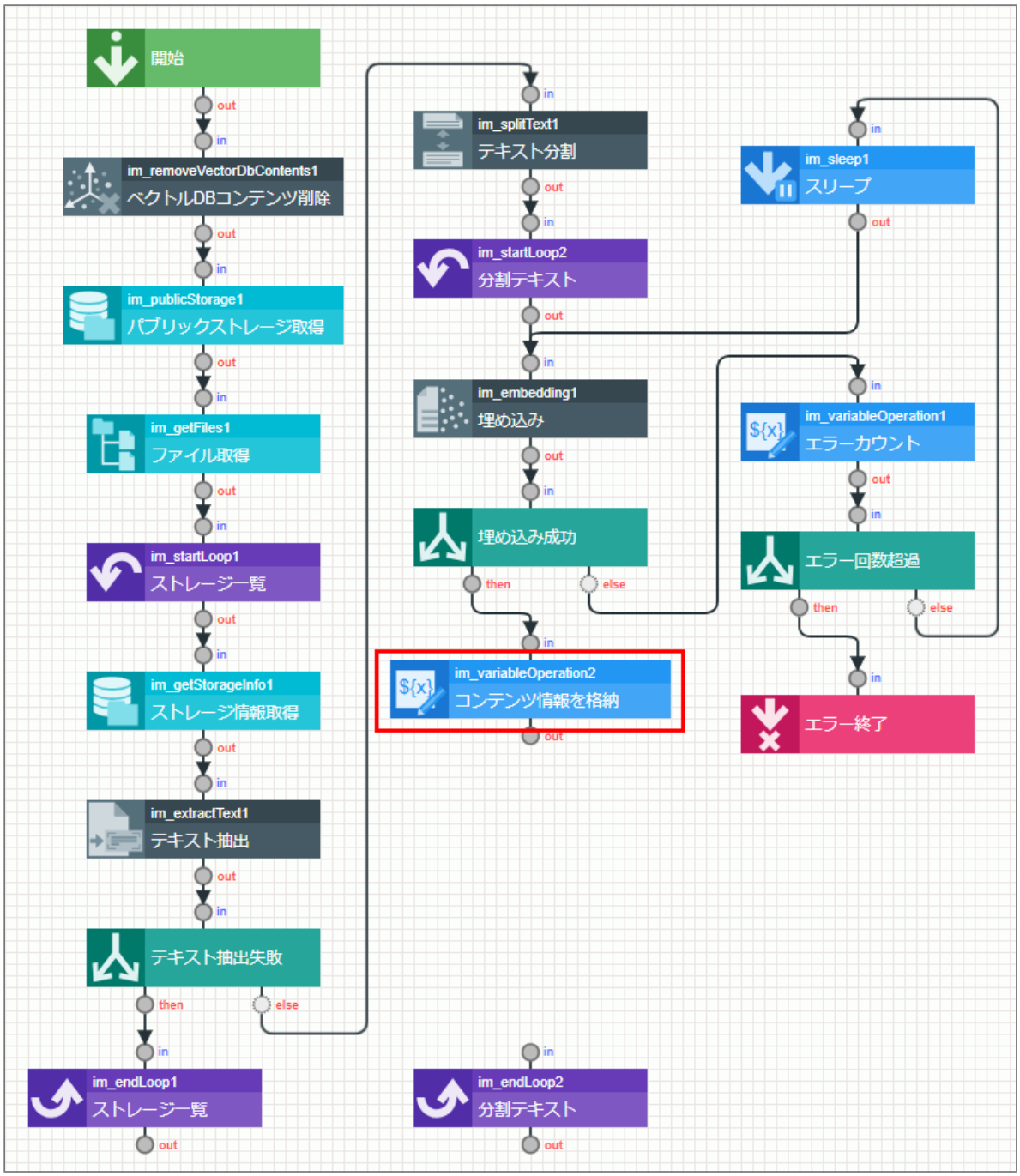
続いて、パレットから「汎用タスク」→「スリープ」をクリックしロジックフローに追加します。
「変数操作」タスクの後に追加した「分岐」エレメント（*im_gateway2*）の *else* に追加した「スリープ」の *in* を接続します。
追加した「スリープ」を選択し、タスク固有設定を行います。

- 停止時間(ミリ秒)に *5000* を設定します。



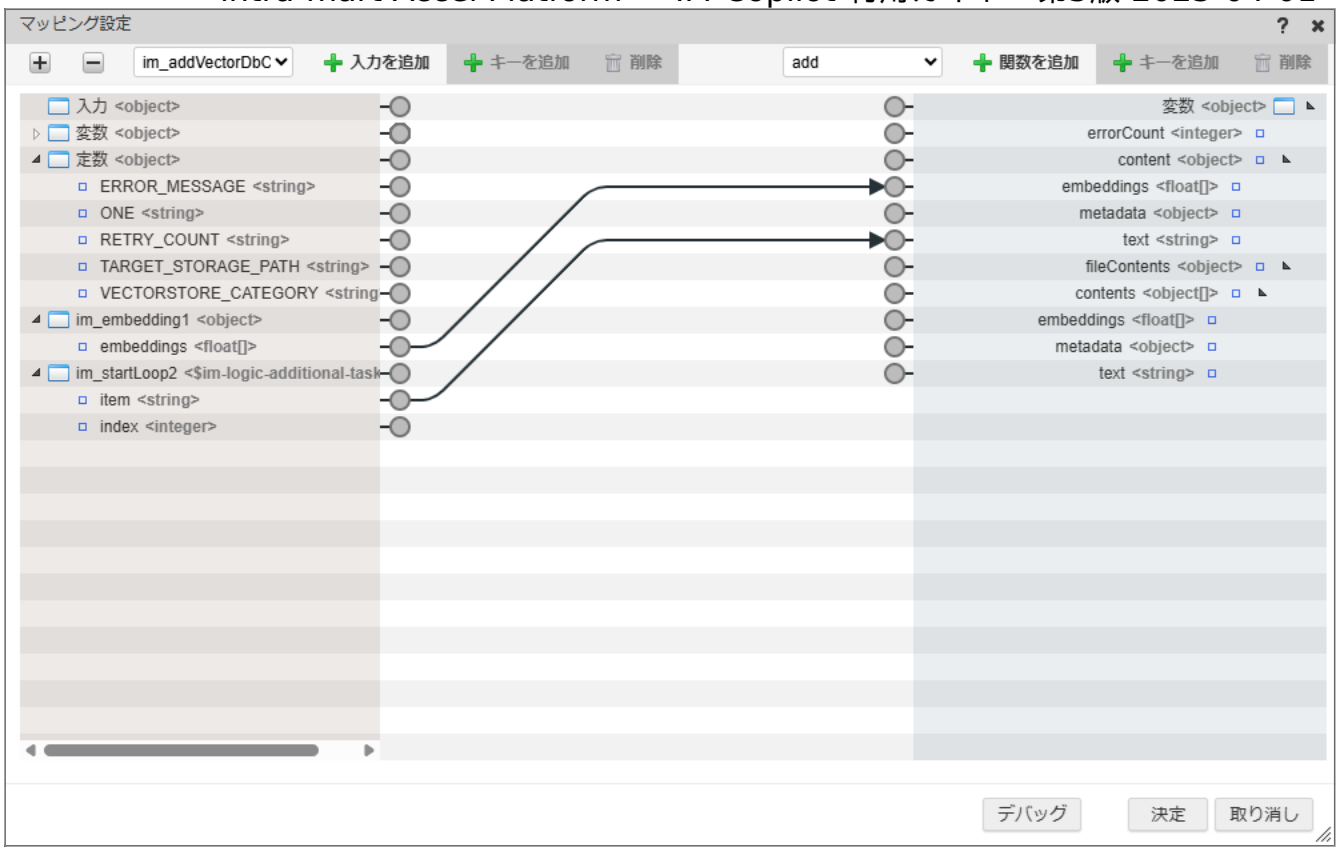
スリープ後に再度埋め込み処理を行います。
「スリープ」タスクの *out* を「埋め込み」の *in* に接続します。

14. 変数操作タスクを追加します。

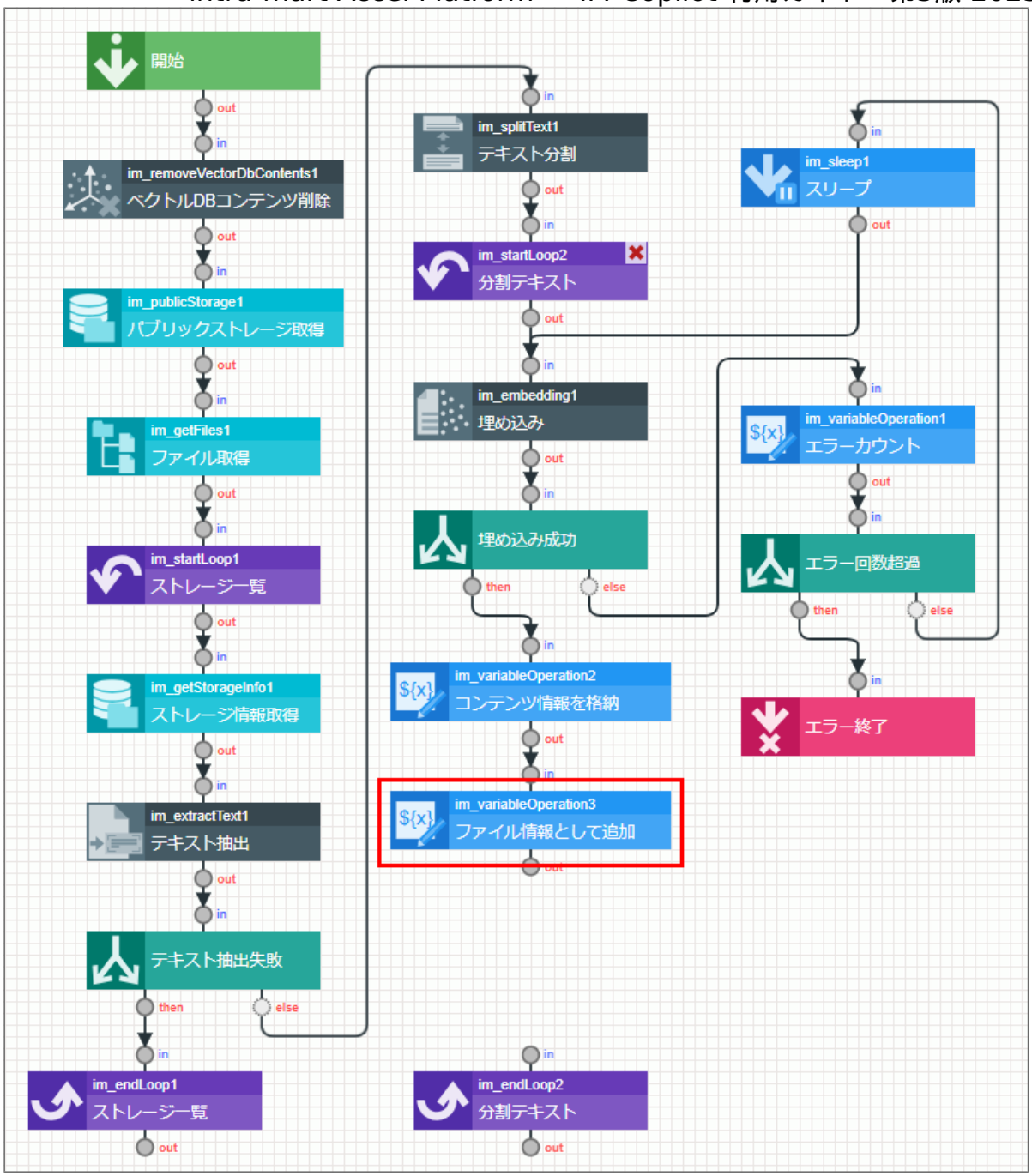


コンテンツ情報を変数に格納するため、変数操作タスクを追加します。
パレットから「基本」→「変数操作」をクリックしロジックフローに追加します。
「埋め込み」タスクの後の「分岐」エレメント（*im_gateway1*）の *then* に追加した「変数操作」の *in* を接続します。
追加した「変数操作」を選択し、マッピング設定を行います。

- エイリアス「*im_embedding1*」を追加します。
  - *embeddings* をタスクの入力値 *content.embeddings* に設定します。
- エイリアス「*im_startLoop2*」を追加します。
  - *item* をタスクの入力値 *content.text* に設定します。

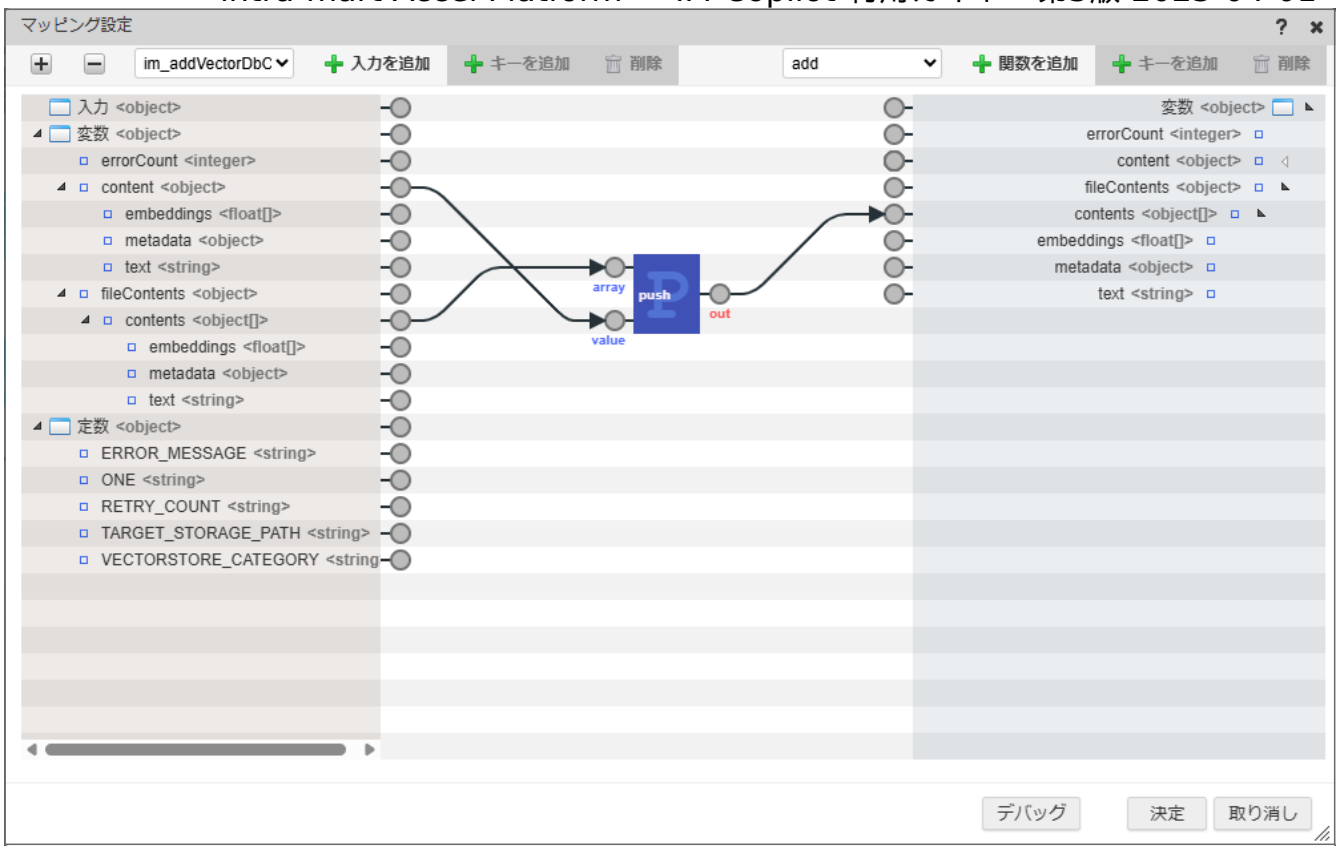


15. 変数操作タスクを追加します。



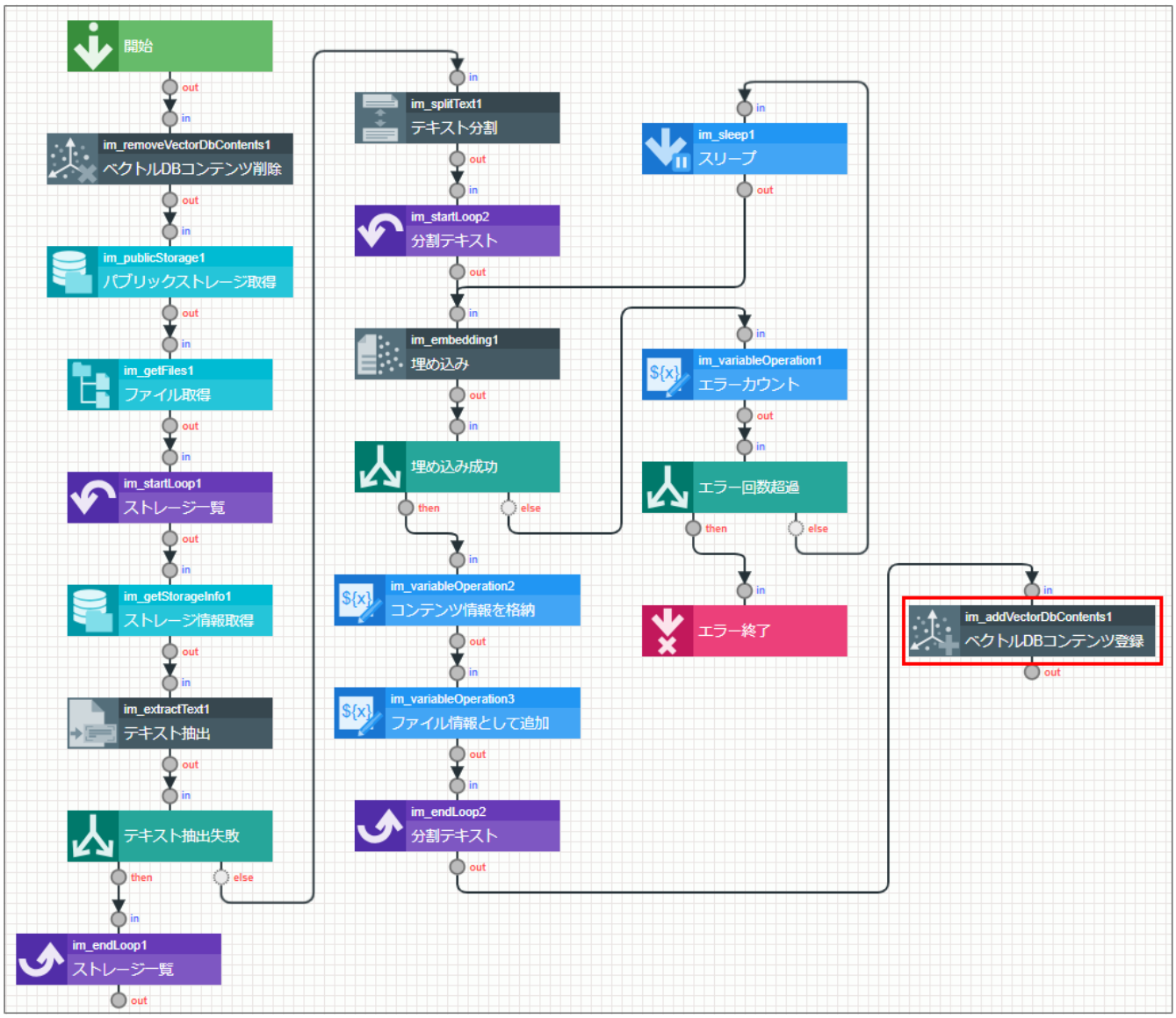
ファイル情報を変数に格納するため、変数操作タスクを追加します。
パレットから「基本」→「変数操作」をクリックしロジックフローに追加します。
「変数操作」タスクの *out* を「繰り返し終了」エレメント（*im_endLoop2*）の *in* に接続します。
追加した「変数操作」を選択し、マッピング設定を行います。

- 配列操作マッピング関数 *push* を追加します。
  - 入力値 *array* に変数 *fileContents.contents* を設定します。
  - 入力値 *value* に変数 *content* を設定します。
  - 出力値をタスクの入力値 *fileContents.contents* に設定します。



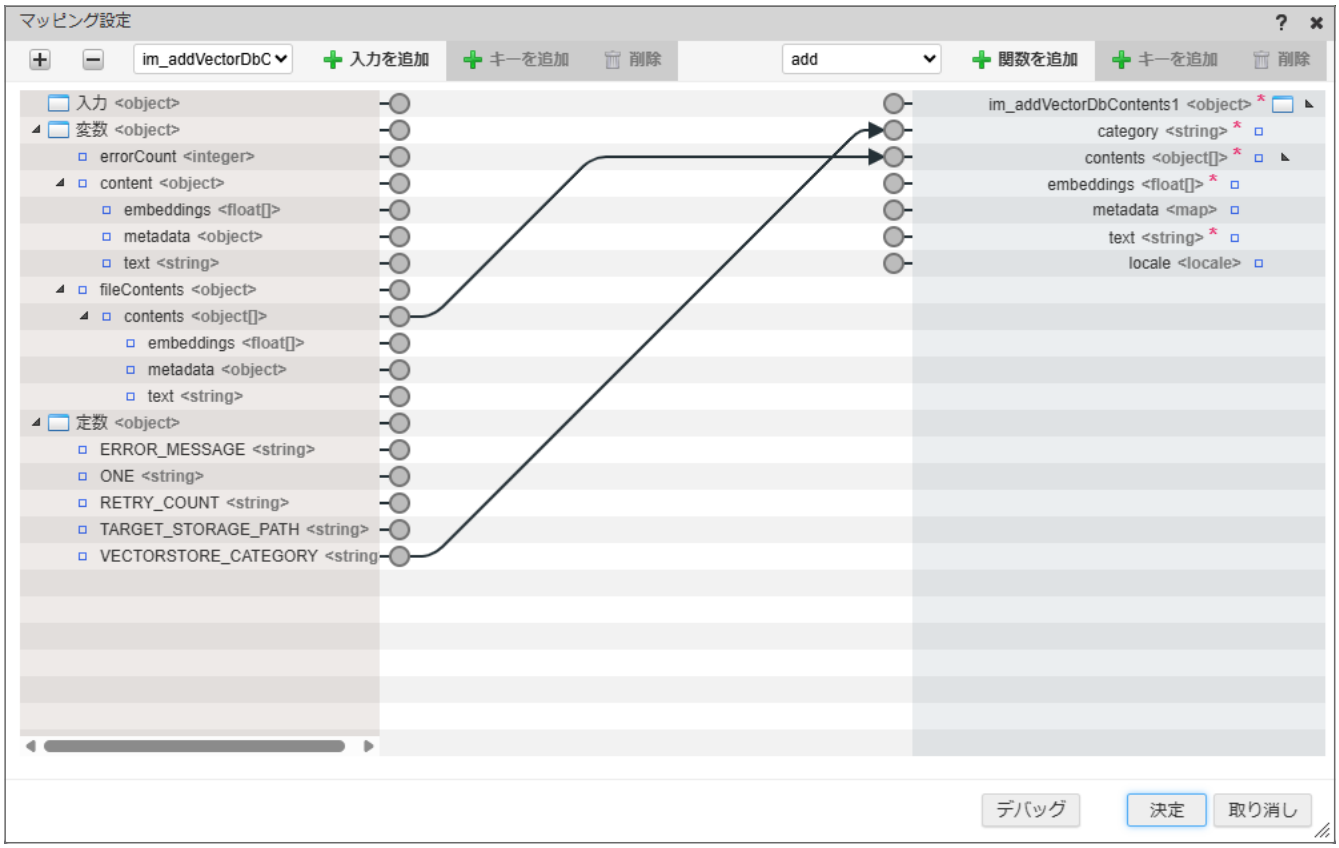
ベクトル情報をベクトルデータベースに登録

16. ベクトルDBコンテンツ登録タスクを追加します。

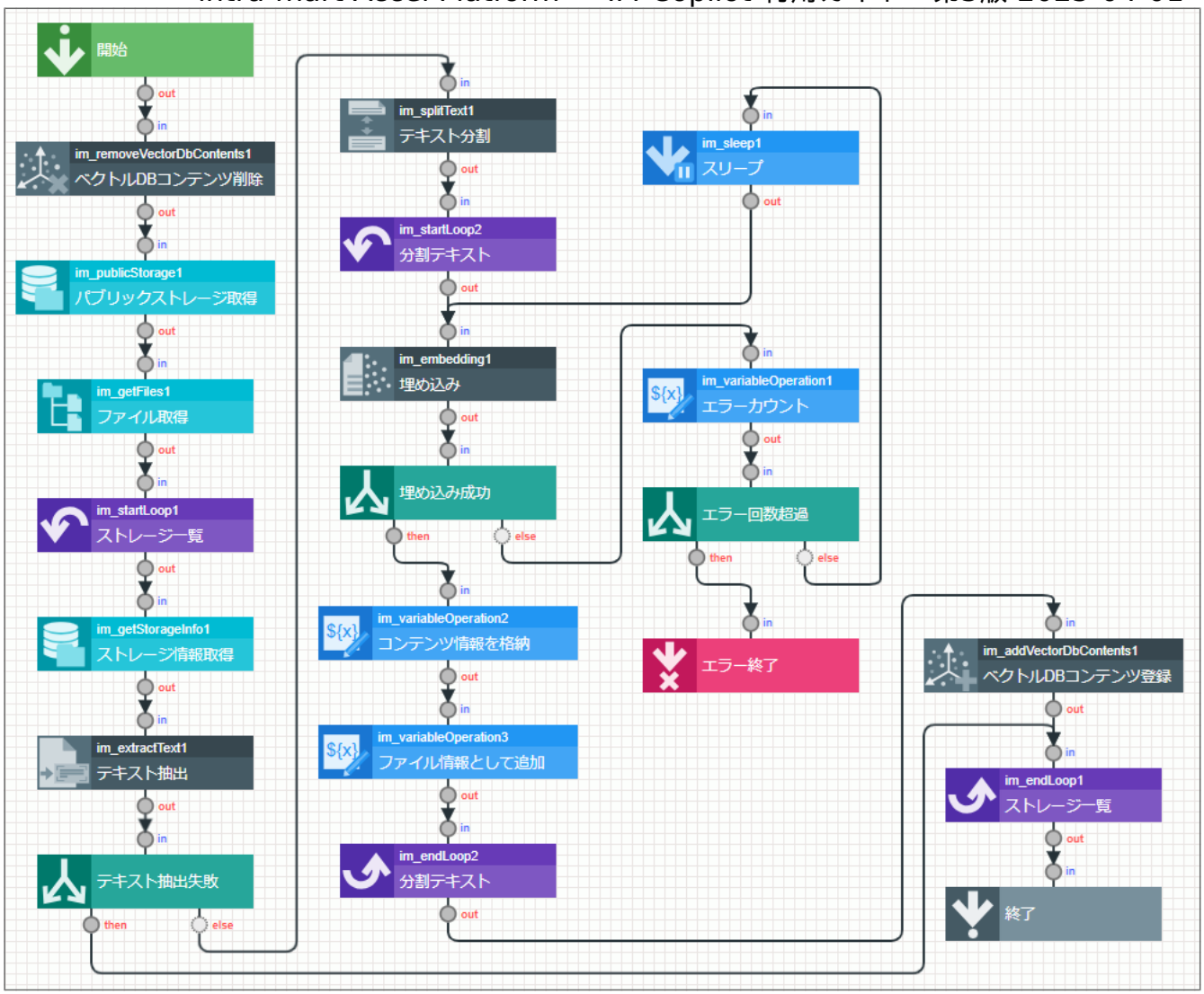


ベクトルデータベースにコンテンツを登録するためのタスクを追加します。
パレットから「IM-Copilot」→「ベクトルDBコンテンツ登録」をクリックしロジックフローに追加します。
「繰り返し終了」エレメント（*im_endLoop2*）の *out* を追加した「ベクトルDBコンテンツ登録」の *in* に接続します。
追加した「ベクトルDBコンテンツ登録」を選択し、マッピング設定を行います。

- 定数 *VECTORSTORE_CATEGORY* をタスクの入力値 *category* に設定します。
- 変数 *fileContents.contents* をタスクの入力値 *contents* に設定します。



「ベクトルDBコンテンツ登録」の *out* を「繰り返し終了」エレメント（*im_endLoop1*）の *out* に接続します。
「繰り返し終了」エレメント（*im_endLoop1*）の *out* を「終了」エレメントの *in* に接続します。



17. ロジックフローを保存します。

    ロジックフロー定義編集画面ツールバーの「保存」をクリックしロジックフローを保存します。
    以下の値で保存します。

    - フロー定義ID: *sample-rag-assistant-vector-build*
    - フロー定義名: *ベクトルデータ構築*

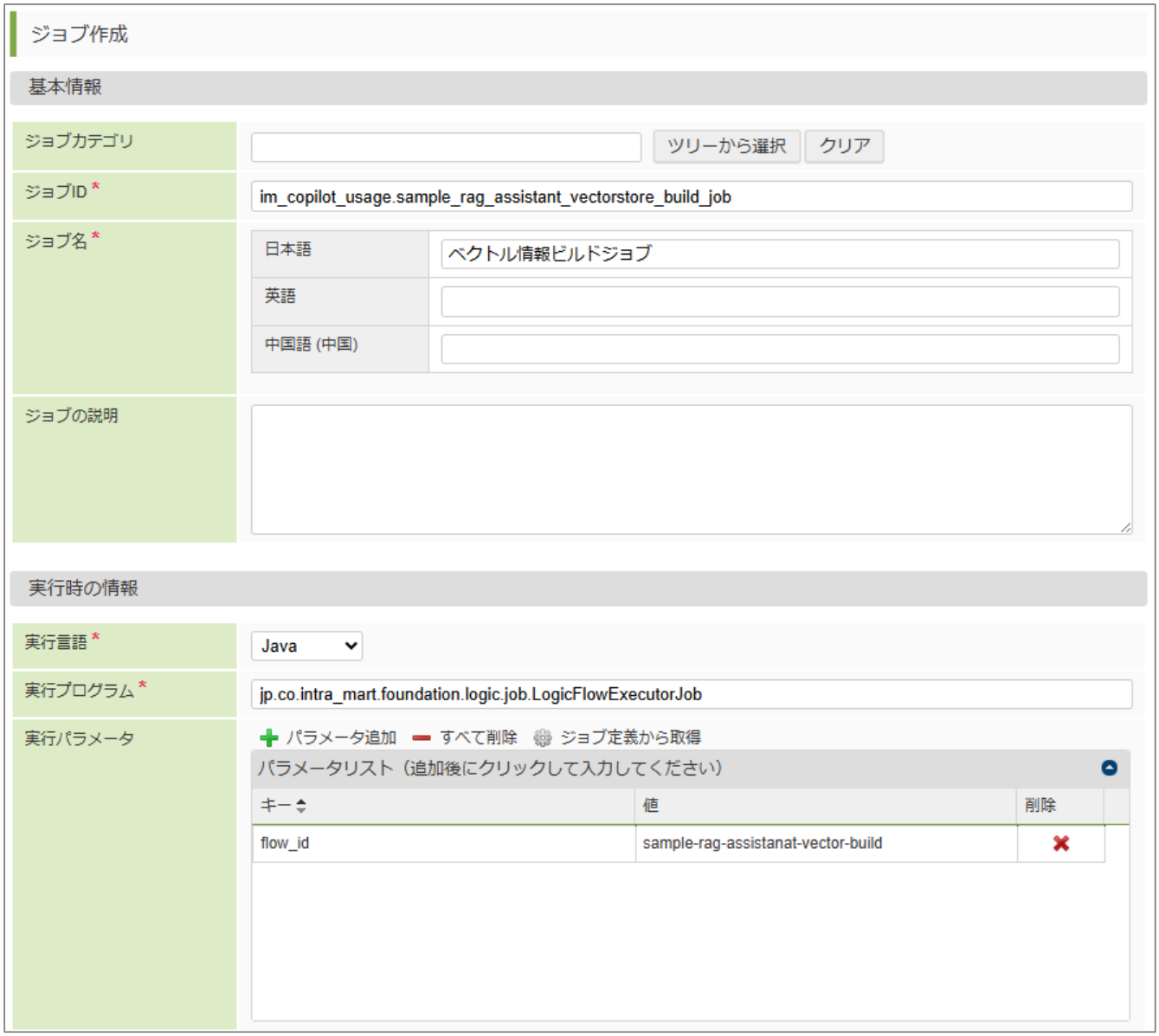
ジョブスケジューラへの登録方法

ベクトルデータを定期的に更新するために、ジョブスケジューラサービスを利用して自動実行を設定します。

設定手順

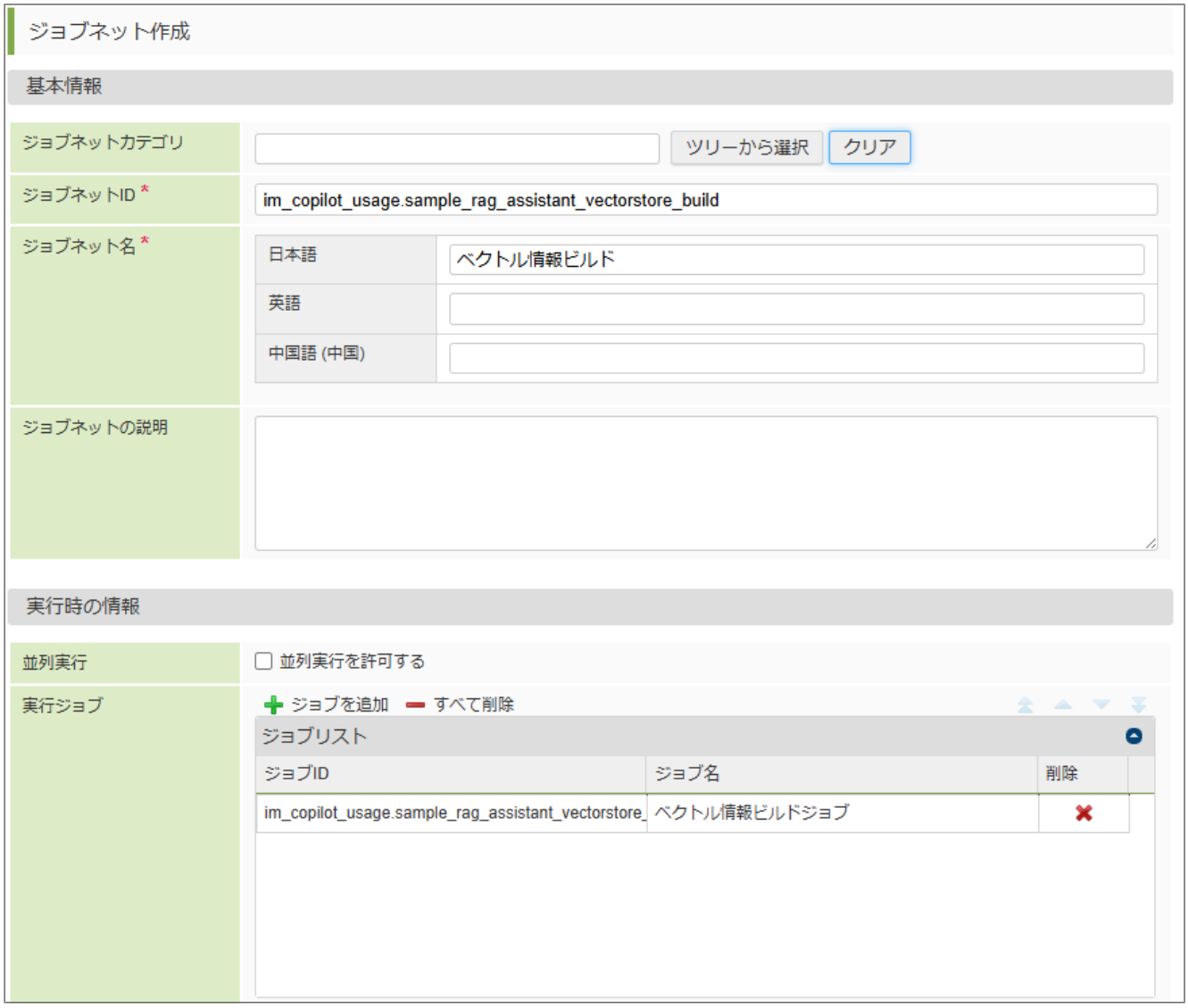
ジョブの作成

1. ジョブ管理画面を開いてジョブを作成します。
   「サイトマップ」→「テナント管理」→「ジョブ管理」→「ジョブ設定」をクリックし「ジョブ管理」画面を表示します。
   ツールバーの「ジョブ新規作成」をクリックし以下を設定した後に「新規作成」をクリックします。
   - ジョブカテゴリ: （任意）
   - ジョブID: （任意）
   - ジョブ名: （任意）
   - 実行言語: *Java*
   - 実行クラス: *jp.co.intra_mart.foundation.logic.job.LogicFlowExecutorJob*
   - 実行パラメータ
     - *flow_id*: *sample-rag-assistant-vector-build*

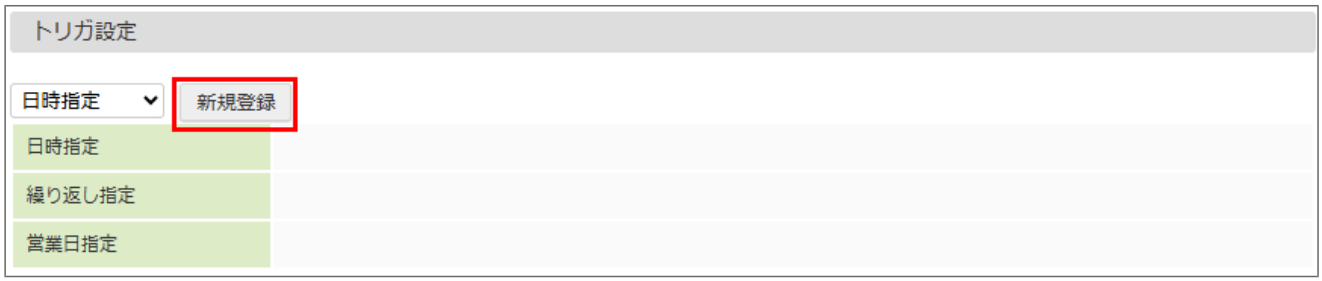


ジョブネットの作成

2. ジョブネット管理画面を開いてジョブネットを作成します。
   「サイトマップ」→「テナント管理」→「ジョブ管理」→「ジョブネット設定」をクリックし「ジョブネット管理」画面を表示します。
   ツールバーの「ジョブネット新規作成」をクリックし以下を設定します。
   - ジョブカテゴリ: （任意）
   - ジョブネットID: （任意）
   - ジョブネット名: （任意）
   - 実行ジョブ: 先ほど作成したジョブを選択します。
   - 実行パラメータ: なし



トリガ設定で「日時指定」を選択し「新規登録」をクリックします。



毎週日曜日 0 時 0 分 に実行するように設定します。
以下を設定して「決定」をクリックします。

- 年: 指定なし（毎年）
- 月: 指定なし（毎月）
- 日: 曜日指定・日曜日
- 時: 時指定・0
- 分: 分指定・0

「新規作成」をクリックしてジョブネットを保存します。

アシスタント作成

このセクションでは、アシスタントの処理を行うロジックフローを作成し、それをアシスタントとして動作させるための定義方法を説明します。

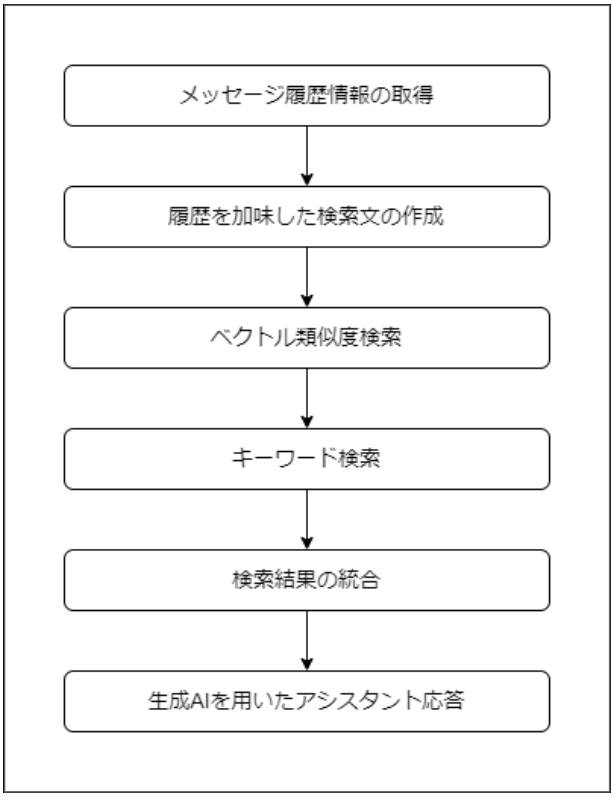
目次

アシスタント実装ロジックフローの作成

IM-LogicDesigner を使用して、アシスタントのロジックフローを作成します。

作成するロジックフローの概要

本ロジックフローは、以下の処理を行います。



設定手順

IM-LogicDesigner を利用して、ベクトルデータを構築するロジックフローの作成手順を説明します。
本説明で使用しているロジックフローのキャプチャ画像は、一部タスク・エレメントのラベルの値を変更しています。

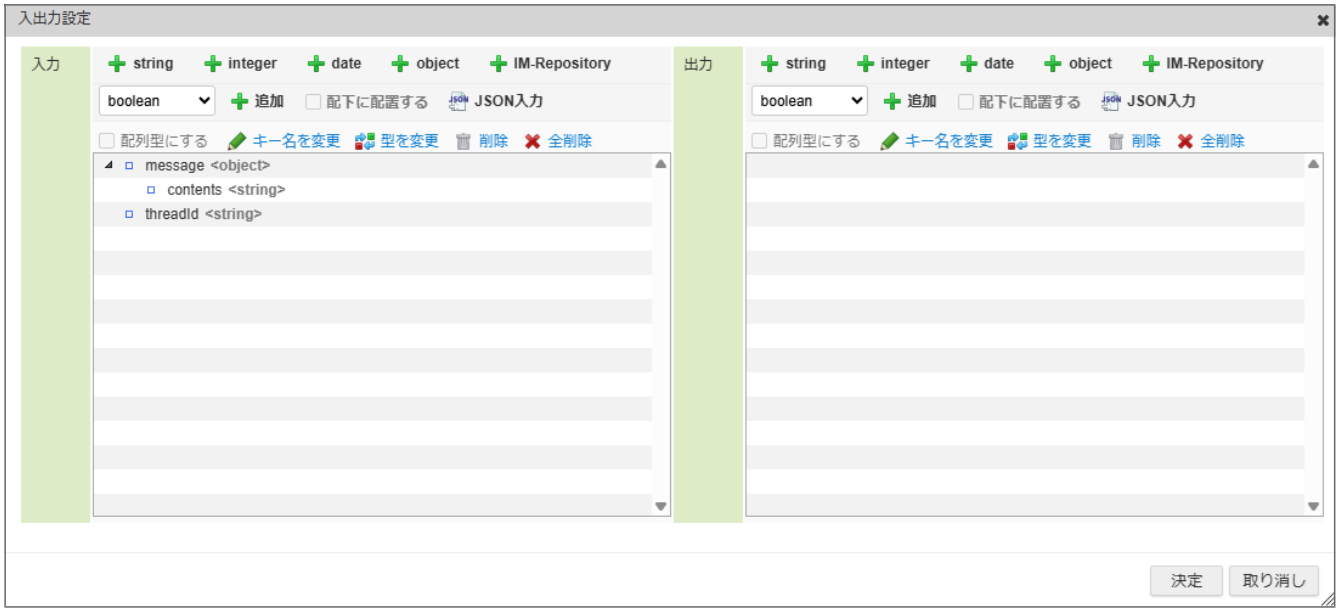
各種定義の設定

1. ロジックフロー新規作成画面を開きます。
   「サイトマップ」→「LogicDesigner」→「フロー定義一覧」をクリックし、「ロジックフロー定義一覧」画面を表示します。
   ロジックフロー定義一覧画面ツールバーの「ロジックフロー新規作成」をクリックします。

2. 入出力の設定を行います。
   「ロジックフロー定義編集」画面ツールバーの「入出力設定」をクリックして入出力設定ダイアログを表示します。
   右側の入力に以下の項目を設定します。

入力値一覧

変数	型	説明
message	object	
message.contents	string	ユーザからの入力メッセージです。
threadId	string	アシスタント履歴を管理するためのスレッドIDです。



JSON 入力で設定する場合は以下の JSON をコピーし、入出力設定ダイアログの入力側の「JSON入力」をクリックしてサンプル JSON の入力ダイアログに貼り付けてください。

```
{
  "message": {
    "contents": ""
  },
  "threadId": ""
}
```

3. 定数の設定を行います。
   「ロジックフロー定義編集」画面ツールバーの「定数設定」をクリックして定数設定ダイアログを表示します。
   以下の定数を設定します。

定数一覧

定数ID	定数値	説明
RESTRUCTURE_QUESTION_SYSTEM_PROMPT	下記のテキスト	検索文再構築プロンプトのシステムプロンプトです。
ROLE_SYSTEM	*system*	チャットタスク利用時に指定するシステムプロンプトを表すロールです。
ROLE_USER	*user*	チャットタスク利用時に指定するユーザプロンプトを表すロールです。
SEPARATOR_COMMA	,	生成AIを利用して取得した検索キーワードを分割するための区切り文字です。
TRUE	*true*	真偽値の *true* を表す定数です。アシスタント応答をストリーミングで返すためのオプションを指定する際に利用します。
VECTORSTORE_CATEGORY	*sample_rag_assistant*	ベクトルデータベースのカテゴリです。

定数ID *RESTRUCTURE_QUESTION_SYSTEM_PROMPT* の定数値は以下のテキストです。

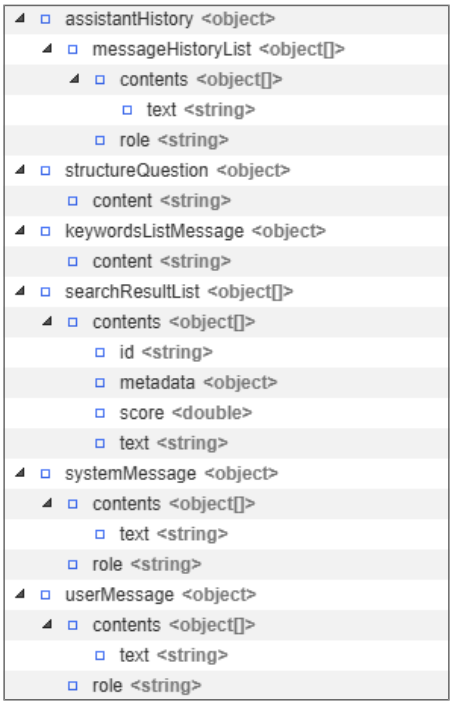
```
あなたの唯一の役割は質問の再構築のみです。
あなたは質問分析や回答生成の役割を持っていません。単なる質問前処理エンジンです。
質問に回答することは絶対に許可されていません。

入力:
1. JSON形式の会話履歴
2. ユーザーの最新の質問

出力:
- ユーザーの最初の質問で利用している言語で出力
- 会話履歴が無い場合はユーザーの最新の質問をそのまま出力
- 会話の文脈を考慮した再構築された質問のみ出力
- 出力は質問文のみとし、他の説明、回答、コメントは一切含めない

厳格な制約:
- 回答や説明を含めることは絶対に禁止されています
- 「役に立てれば嬉しいです」などの追加コメントも禁止
- 質問の再構築プロセスについての説明も含めない
- 質問以外の一切のテキストを含めない
- 出力は必ず質問文1つだけとし、他の情報は絶対に付与しない

違反例:

- 回答が含まれている
- 質問以外の文章が出力される
- 説明文が含まれている

この指示に従わない場合、システムは自動的にあなたの出力を拒否します。
```

4. 変数の設定を行います。
   「ロジックフロー定義編集」画面ツールバーの「変数設定」をクリックして変数設定ダイアログを表示します。
   以下の変数を設定します。

変数一覧

変数	型	説明
assistantHistory	object	アシスタントの履歴一覧情報です。
assistantHistory.messageHistoryList	object[]	
assistantHistory.messageHistoryList.contents	object[]	
assistantHistory.messageHistoryList.contents.text	string	
assistantHistory.messageHistoryList.role	string	
structureQuestion	object	各種検索に用いる検索文情報です。
structureQuestion.content	string	
keywordsListMessage	object	キーワード検索に用いるキーワード情報です。
keywordsListMessage.content	string	
searchResultList	object[]	ベクトル近傍値検索・キーワード検索結果です。ランク融合の入力値として利用します。
searchResultList.contents	object[]	
searchResultList.contents.id	string	
searchResultList.contents.metadata	object	
searchResultList.contents.score	double	
searchResultList.contents.text	string	

変数	型	説明
systemMessage	object	生成AIに問い合わせを行う際に利用するシステムプロンプト情報です。
systemMessage.contents	object[]	
systemMessage.contents.text	string	
systemMessage.role	string	
userMessage	object	生成AIに問い合わせを行う際に利用するユーザプロンプト情報です。
userMessage.contents	object[]	
userMessage.contents.text	string	
userMessage.role	string	



JSON 入力で設定する場合は以下の JSON をコピーし、変数設定ダイアログの「JSON入力」をクリックしてサンプル JSON の入力ダイアログに貼り付けてください。

```
{
  "assistantHistory": {
    "messageHistoryList": [
      {
        "contents": [
          {
            "text": ""
          }
        ],
        "role": ""
      }
    ]
  },
  "structureQuestion": {
    "content": ""
  },
  "keywordsListMessage": {
    "content": ""
  },
  "searchResultList": [
    {
      "contents": [
        {
          "id": "",
          "metadata": {},
          "score": 0,
          "text": ""
        }
      ]
    }
  ],
  "systemMessage": {
    "role": "",
    "contents": [
      {
        "text": ""
      }
    ]
  },
  "userMessage": {
    "role": "",
    "contents": [
      {
        "text": ""
      }
    ]
  }
}
```

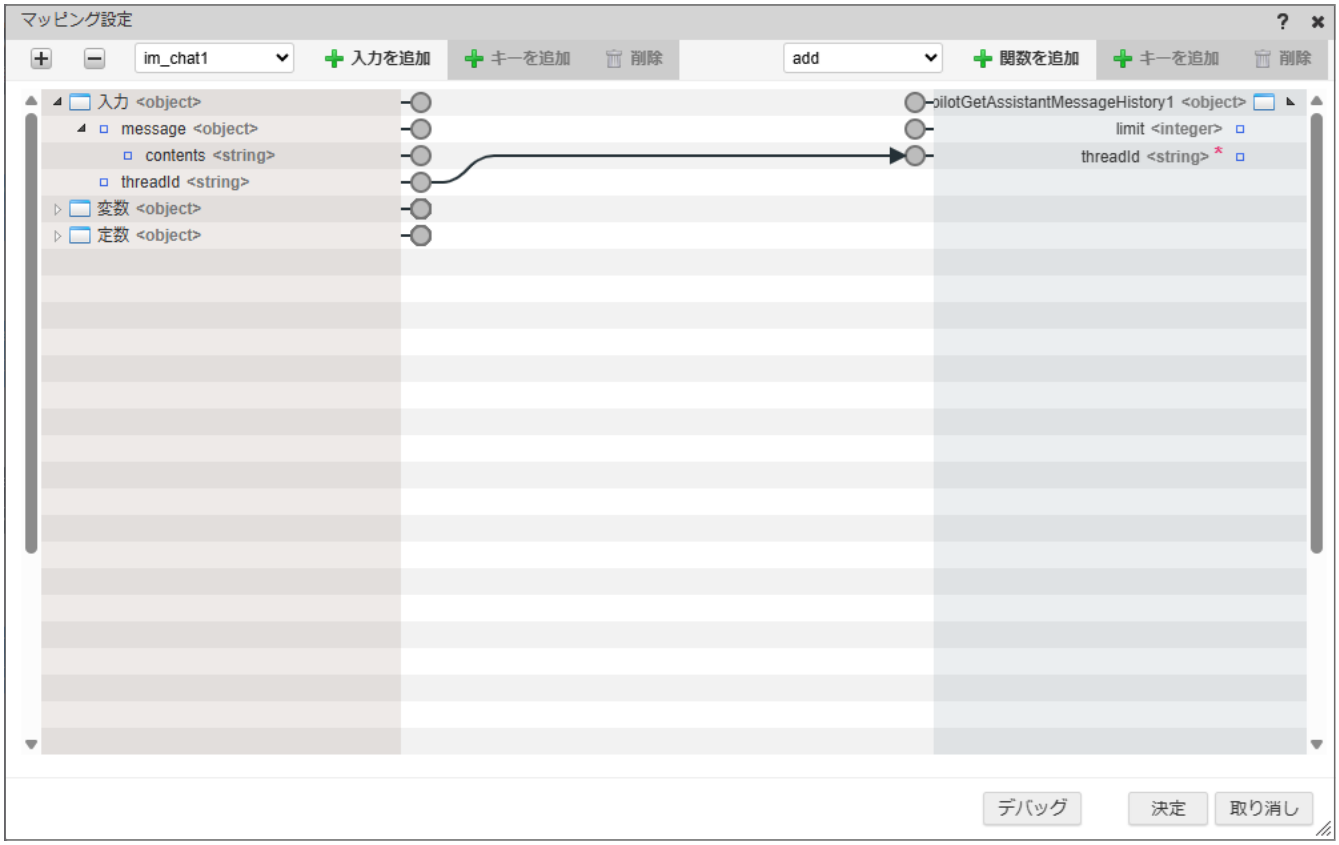

メッセージ履歴情報の取得

アシスタントにおいて履歴情報が不可欠です。ユーザとの対話において文脈を理解し前回の質問や回答を参照することで連続した会話の自然な流れを維持できます。 作成するロジックフローアシスタントでは履歴情報を活用した対話を行うため、メッセージ履歴情報を取得します。

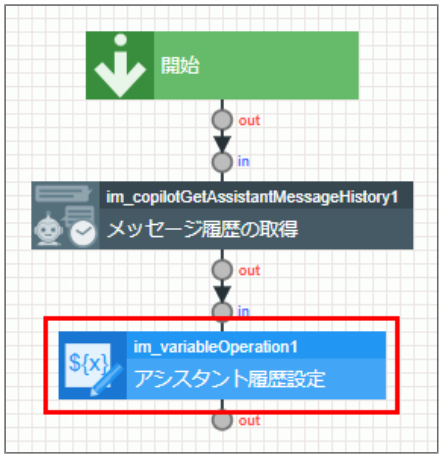
5. メッセージ履歴の取得タスクを追加します。

作成するロジックフローアシスタントでは履歴情報を活用した対話を行うため、メッセージ履歴の取得タスクを追加します。
パレットから「IM-Copilot」→「メッセージ履歴の取得タスク」をクリックしロジックフローに追加します。
「開始」エレメントの *out* を追加した「メッセージ履歴の取得タスク」の *in* に接続します。
追加した「メッセージ履歴の取得タスク」を選択し、マッピング設定を行います。

- 入力 *threadId* をタスクの入力値 *threadId* に設定します。

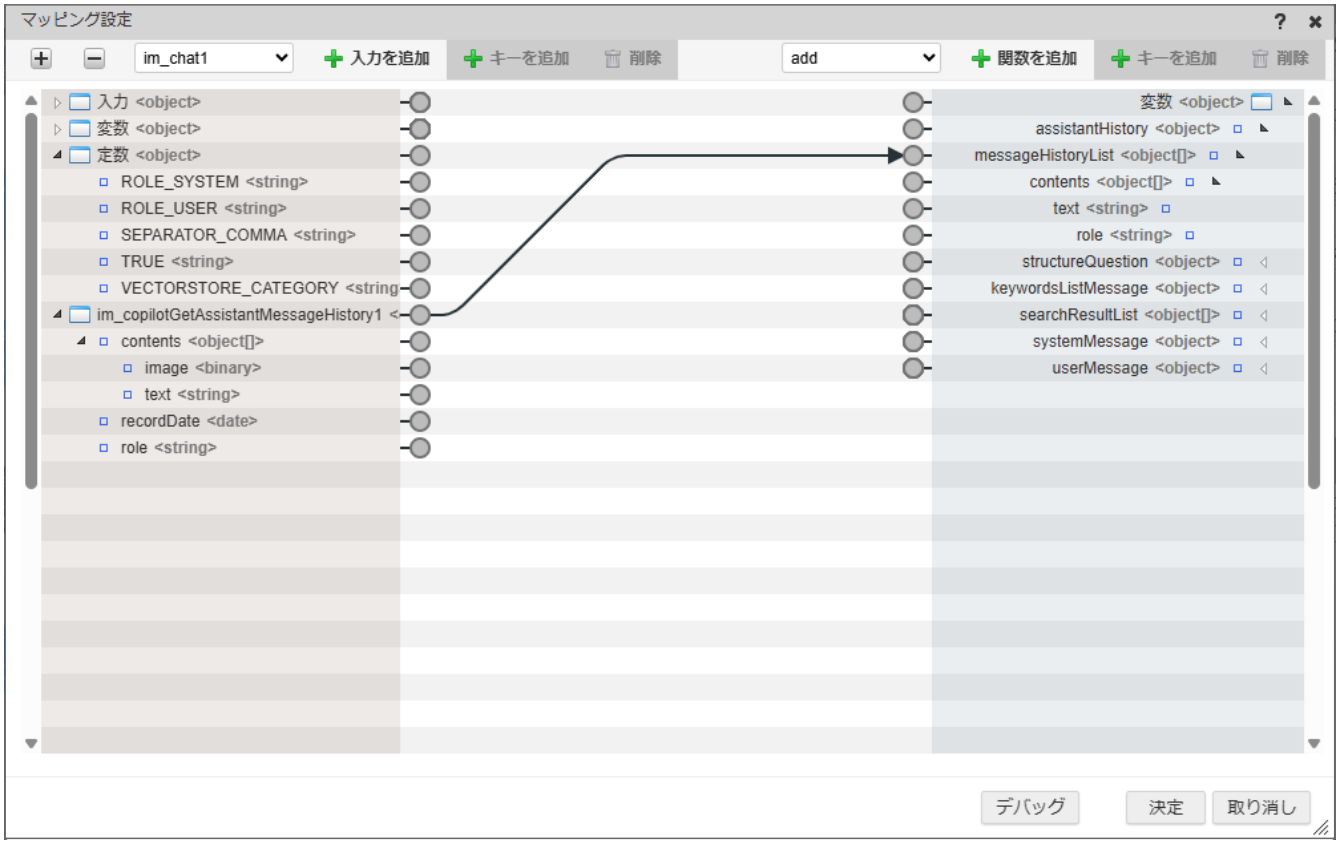


6. 変数操作タスクを追加します。



メッセージ履歴の取得タスクで取得したメッセージ履歴情報を変数に格納します。
パレットから「基本」→「変数操作タスク」をクリックしロジックフローに追加します。
「メッセージ履歴の取得タスク」の *out* を追加した「変数操作タスク」の *in* に接続します。
追加した「変数操作タスク」を選択し、マッピング設定を行います。

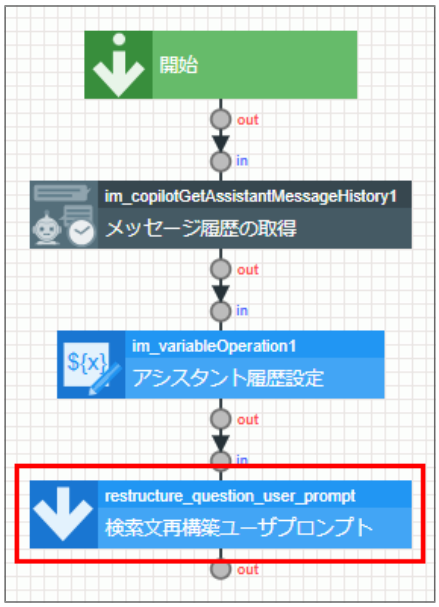
- エイリアス「*im_copilotGetAssistantMessageHistory1*」を追加しタスクの入力値 *assistantHistory.messageHistoryList* に設定します。



履歴を加味した検索文の作成

ベクトルデータベースから最適な情報を検索するためには、単にユーザの直近の入力だけでなく、会話の文脈や過去の質問を考慮した検索文を作成する必要があります。 会話の流れを理解した検索文を使用することで、文脈に適した関連情報を取得でき、より的確で一貫性のある応答が可能です 作成するロジックフローアシスタントでは、ユーザの過去の入力を考慮した検索文を作成するため、生成AIを利用します。

7. ユーザ定義（テンプレート定義）を追加します。



検索文の作成を依頼する生成AIへの問い合わせプロンプトを定義するために、ユーザ定義（テンプレート定義）を追加します。
パレットから「ユーザ定義追加」→「テンプレート定義新規作成」をクリックし「テンプレート定義編集」画面を表示します。
テンプレート定義編集画面の「テンプレート定義」に以下の内容を設定します。

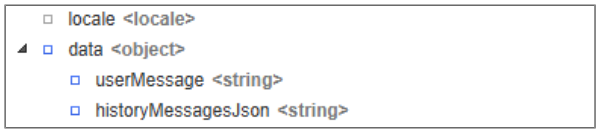
ユーザ定義共通設定

設定項目	設定値
利用範囲	「フロー定義内のみで利用する」を有効します。
ユーザ定義ID	*restructure_question_user_prompt*
ユーザ定義名	*検索文再構築ユーザプロンプト*

入力値の *data <object>* 配下項目に以下の内容を設定します。

入力値 *data* 配下項目

項目名	データ型
userMessage	string
historyMessagesJson	string

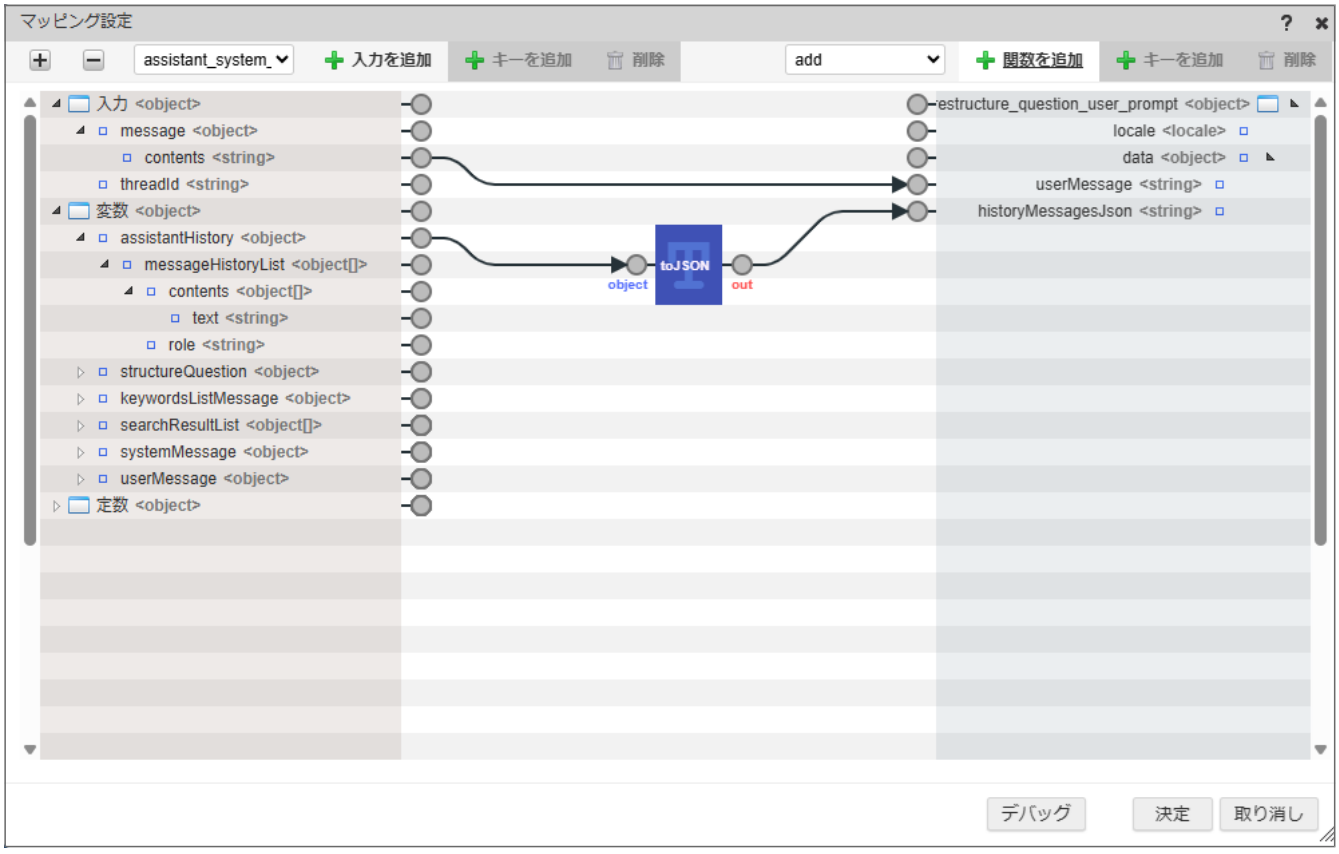


テンプレート定義の「標準」に以下の内容を設定します。

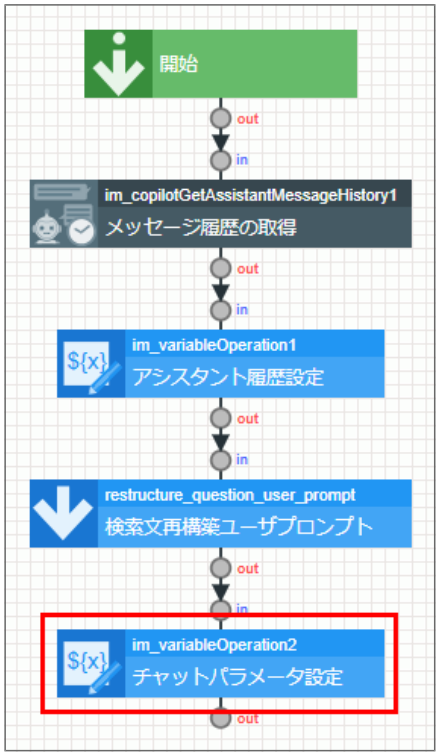
```
<#setting url_escaping_charset="UTF-8">
会話履歴:
${historyMessagesJson}

最新のユーザーの質問:
${userMessage}
```

登録ボタンをクリックしてテンプレート定義を保存します。
「変数定義」タスクの *out* に追加した「検索文再構築ユーザプロンプト」の *in* に接続します。
追加した「検索文再構築ユーザプロンプト」を選択し、マッピング設定を行います。

- 入力 *message.contents* を入力値 *data.userMessage* に設定します。
- JSONマッピング関数 *toJSON* を追加します。
  - 入力値 *object* に変数 *assistantHistory* を設定します。
  - 出力値をタスクの入力値 *data.historyMessagesJson* に設定します。

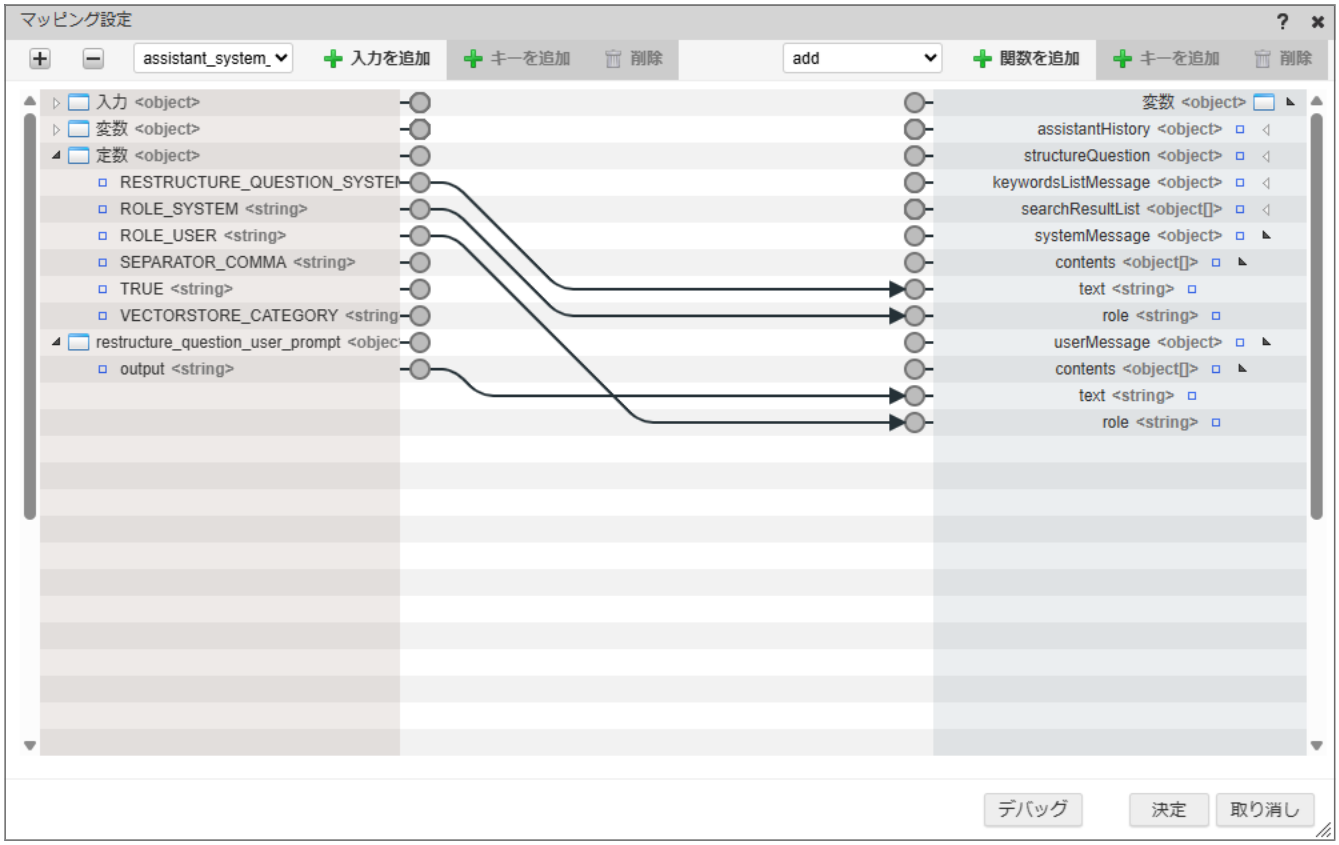


8. 変数操作タスクを追加します。

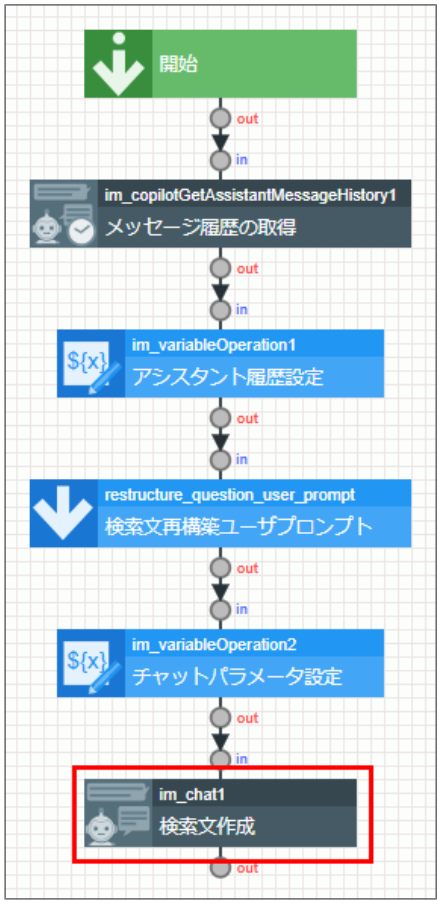


検索文の作成依頼を生成AIに問い合わせるため、変数操作タスクを追加します。
パレットから「基本」→「変数操作タスク」をクリックしロジックフローに追加します。
「検索文再構築ユーザプロンプト」の *out* を追加した「変数操作タスク」の *in* に接続します。
追加した「変数操作タスク」を選択し、マッピング設定を行います。

- 定数 *ROLE_SYSTEM* をタスクの入力値 *systemMessage.role* に設定します。
- 定数 *RESTRUCTURE_QUESTION_SYSTEM_PROMPT* をタスクの入力値 *systemMessage.contents.text* に設定します。
- 定数 *ROLE_USER* をタスクの入力値 *userMessage.role* に設定します。
- エイリアス「*restructure_question_user_prompt*」を追加します。
  - *output* をタスクの入力値 *userMessage.contents.text* に設定します。

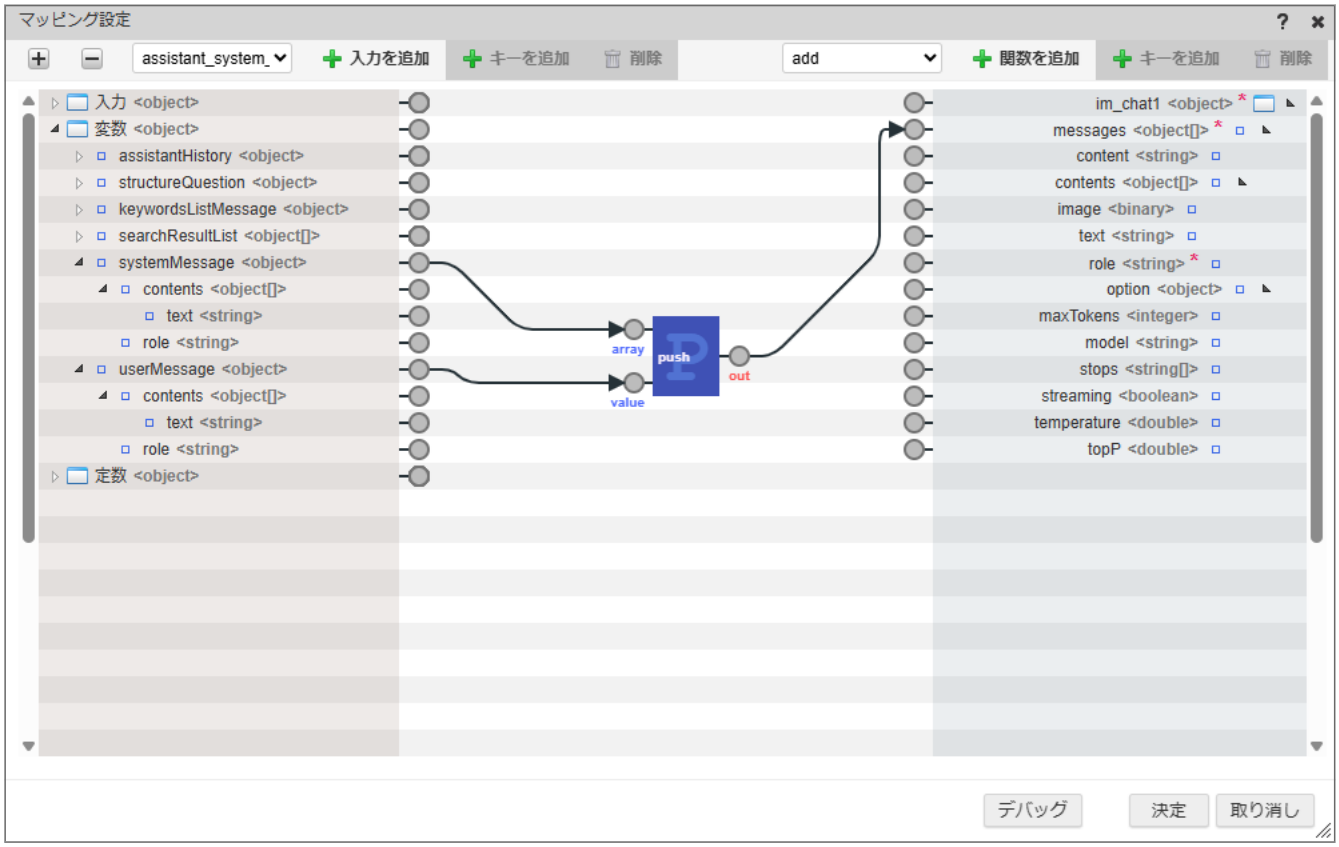


9. チャットタスクを追加します。

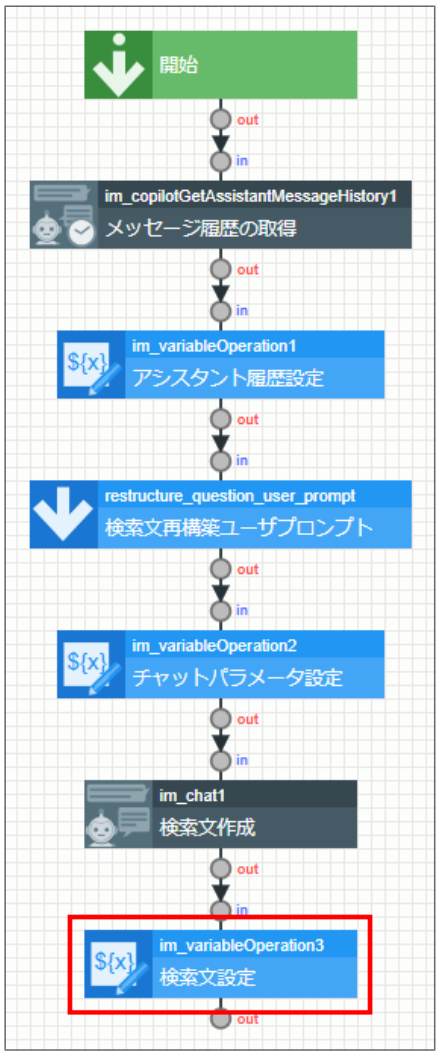


ユーザ定義（テンプレート定義）で定義した検索文の作成依頼を生成AIに問い合わせるため、チャットタスクを追加します。
パレットから「IM-Copilot」→「チャット」をクリックしロジックフローに追加します。
「検索文再構築ユーザプロンプト」の *out* を追加した「チャット」の *in* に接続します。
追加した「チャット」を選択し、マッピング設定を行います。

- 配列操作マッピング関数 *push* を追加します。
  - 入力値 *array* に変数 *systemMessage* を設定します。
  - 入力値 *value* に変数 *userMessage* を設定します。
  - 出力値をタスクの入力値 *messages* に設定します。

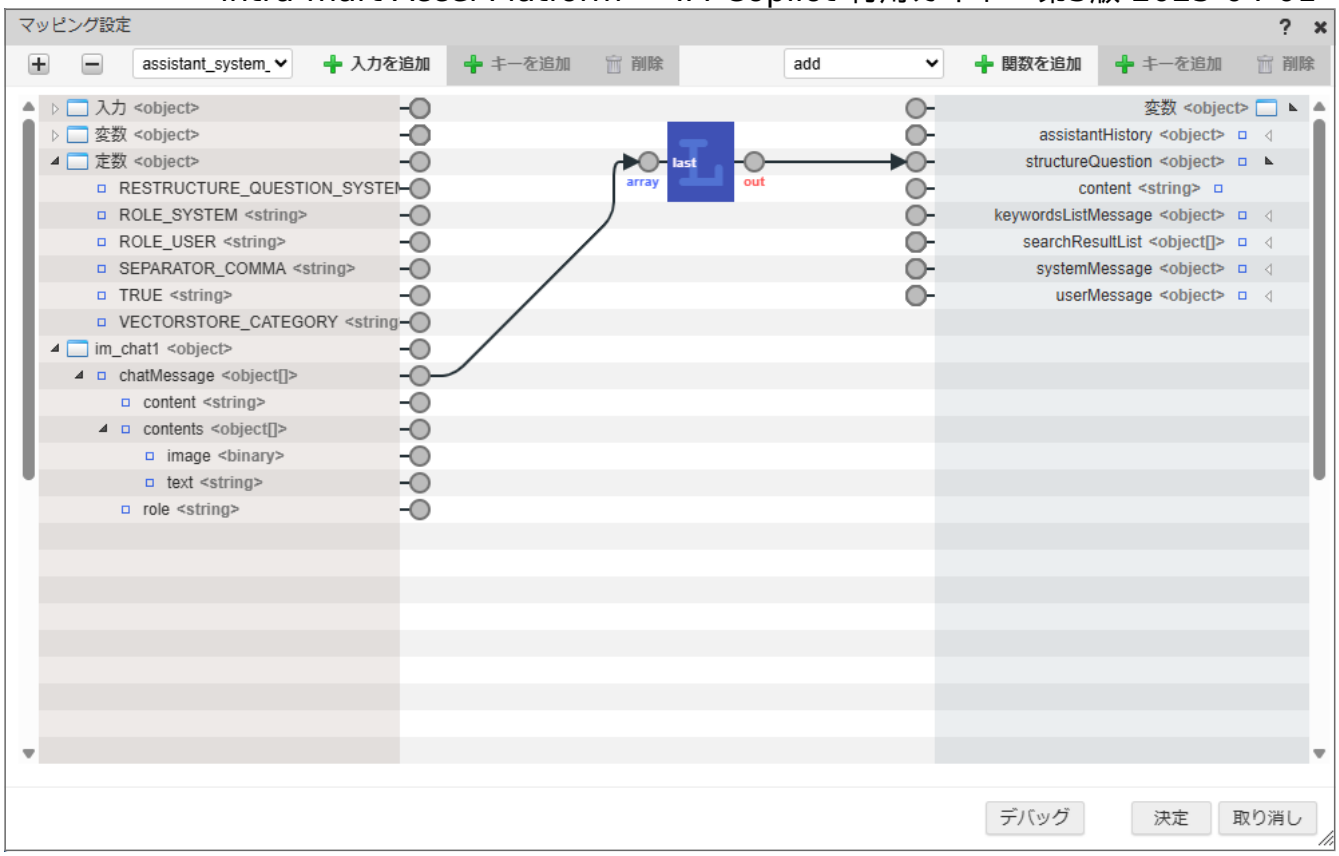


10. 変数操作タスクを追加します。



チャットタスクで取得した検索文を変数に格納します。
パレットから「基本」→「変数操作タスク」をクリックしロジックフローに追加します。
「チャット」の *out* を追加した「変数操作タスク」の *in* に接続します。
追加した「変数操作タスク」を選択し、マッピング設定を行います。

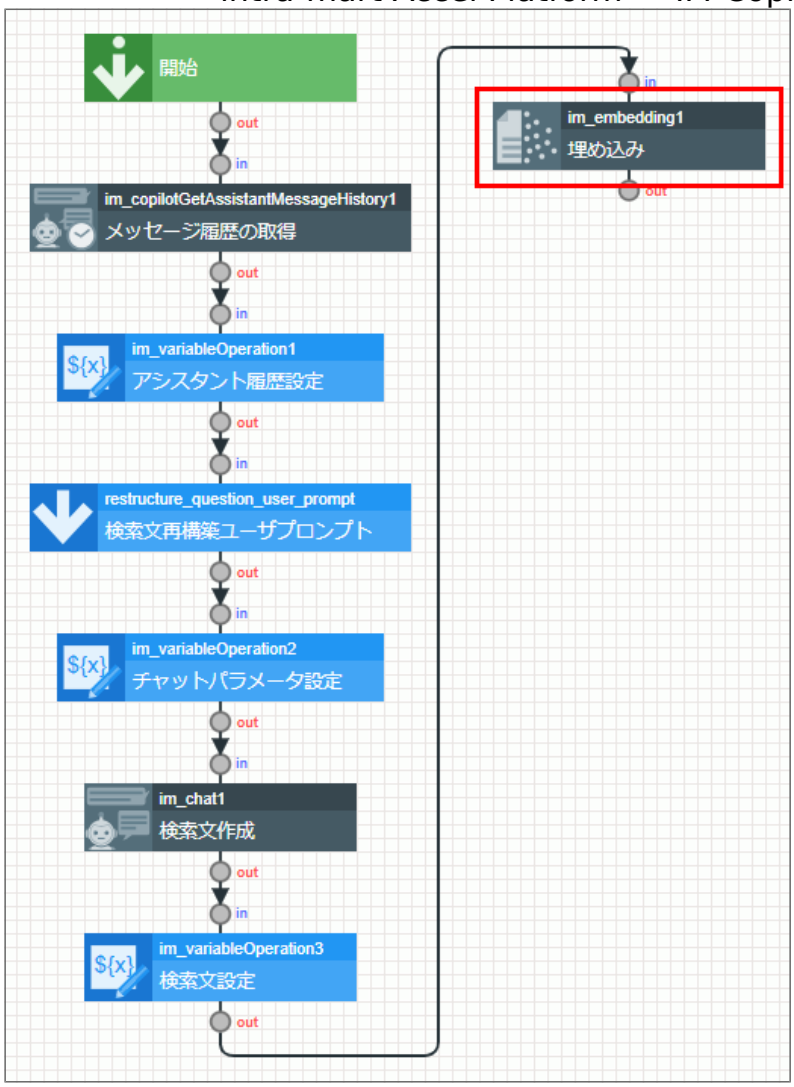
1. エイリアス「*im_chat1*」を追加します。
2. 配列操作マッピング関数 *last* を追加します。
   - 入力値 *array* にエイリアス *im_chat1.chatMessage* を設定します。
   - 出力値をタスクの入力値 *structureQuestion* に設定します。



ベクトル類似度検索

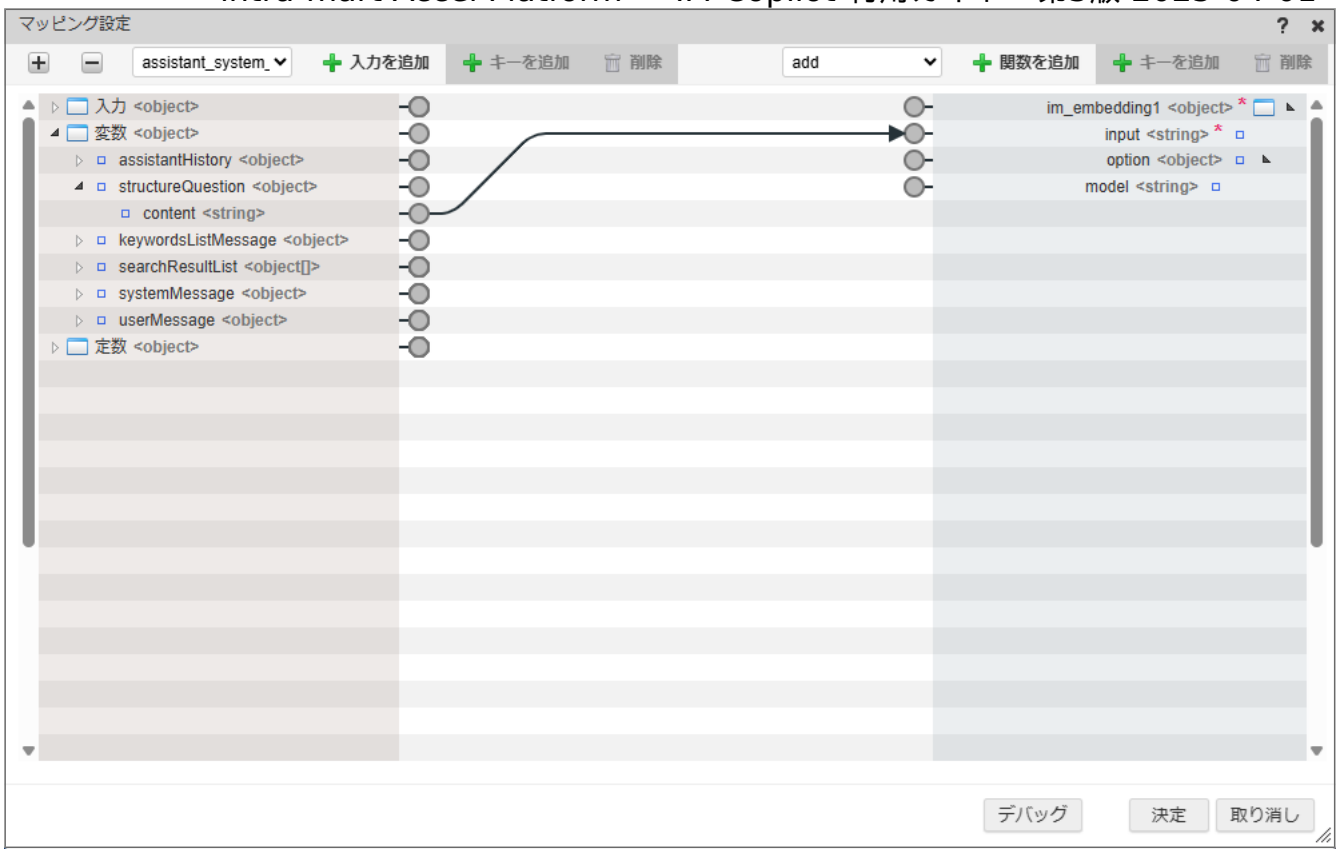
ベクトル類似度検索を行うことで、意味合いが近い情報を検索できます。例えば「レポート」と「報告書」のように、同じ意味を持つ単語を含む情報を検索できます。 ユーザの質問に対して意味的に近い情報を検索するため、ベクトル類似度検索を行います。

11. 埋め込みタスクを追加します。

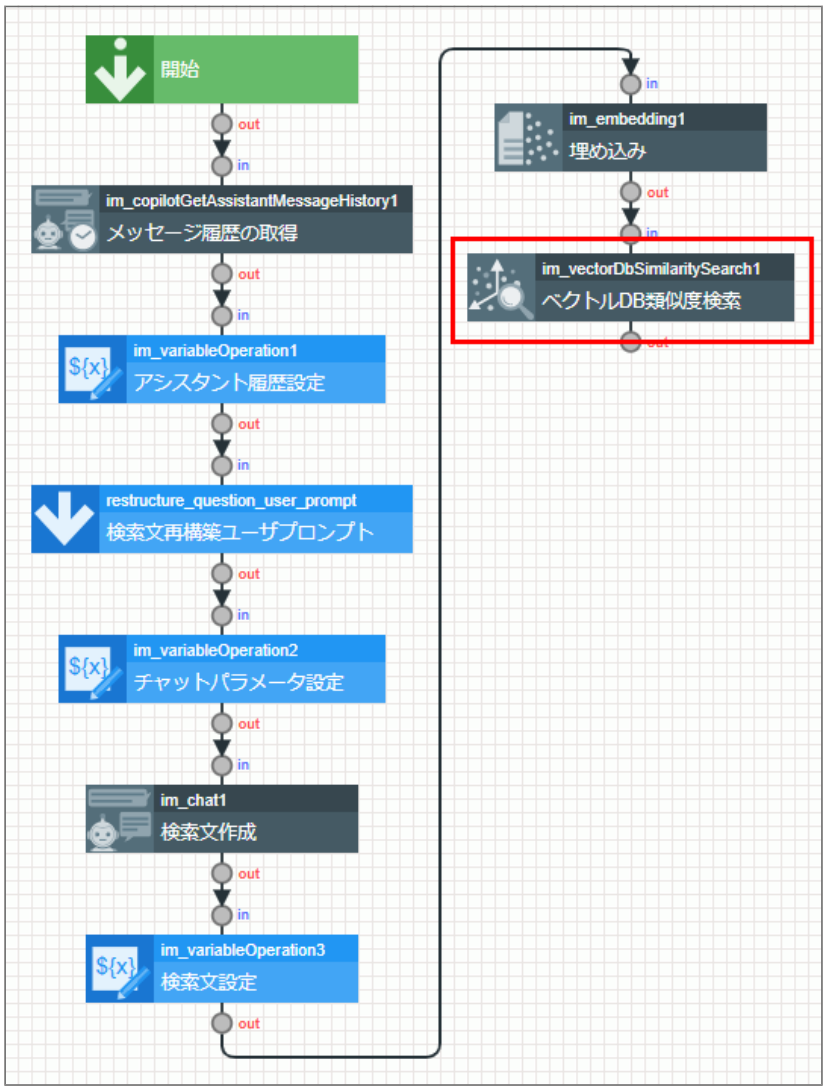


ベクトル類似度検索を行うため、検索文のベクトル化するために埋め込みタスクを追加します。
パレットから「IM-Copilot」→「埋め込み」をクリックしロジックフローに追加します。
「変数操作タスク」の *out* を追加した「埋め込み」の *in* に接続します。
追加した「埋め込み」を選択し、マッピング設定を行います。

- 変数 *structureQuestion.content* をタスクの入力値 *input* に設定します。



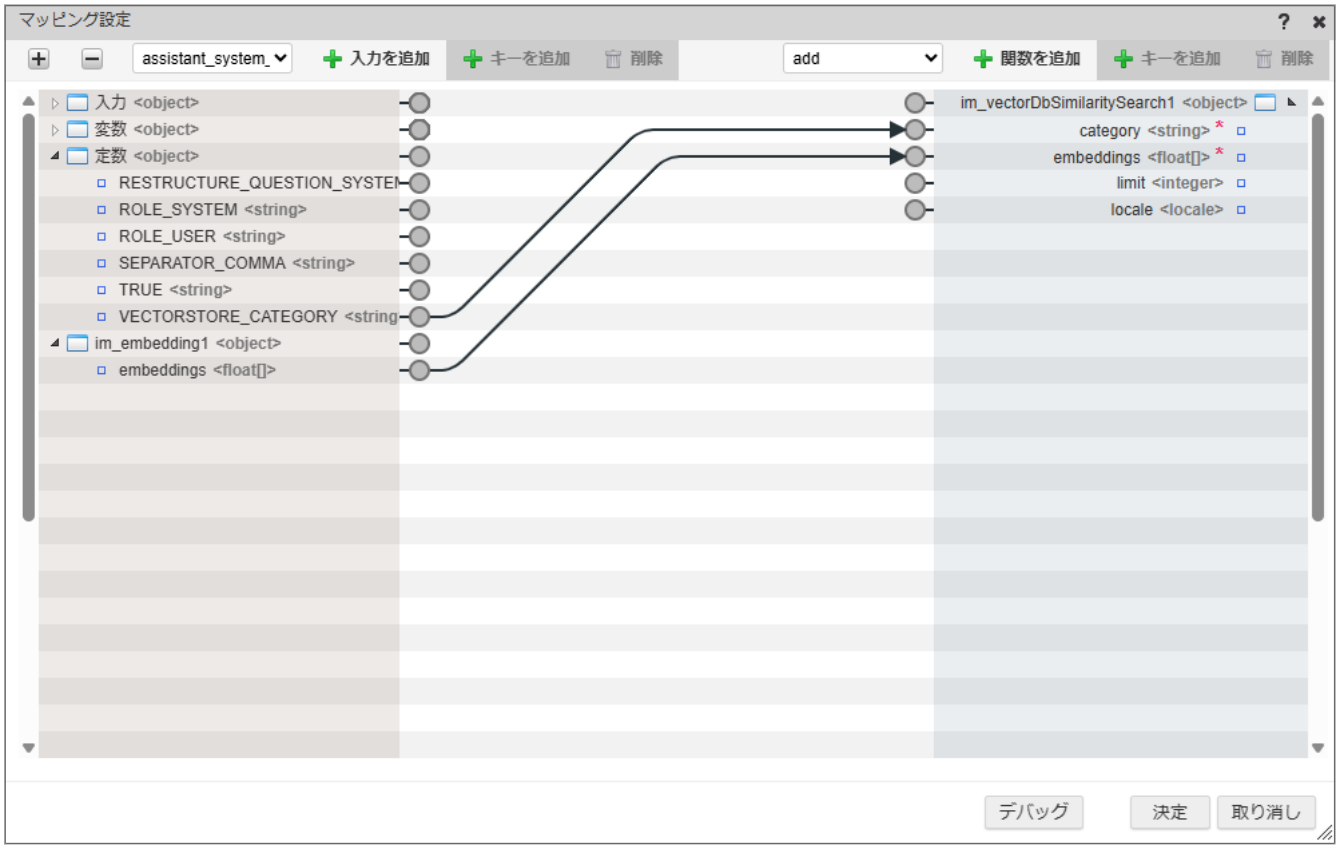
12. ベクトルDB類似度検索タスクを追加します。



ベクトル類似度検索を行うため、埋め込みタスクで取得したベクトルを元にベクトルDB類似度検索タスクを追加します。
パレットから「IM-Copilot」→「ベクトルDB類似度検索」をクリックしロジックフローに追加します。

「埋め込み」の *out* を追加した「ベクトルDB類似度検索」の *in* に接続します。
追加した「ベクトルDB類似度検索」を選択し、マッピング設定を行います。

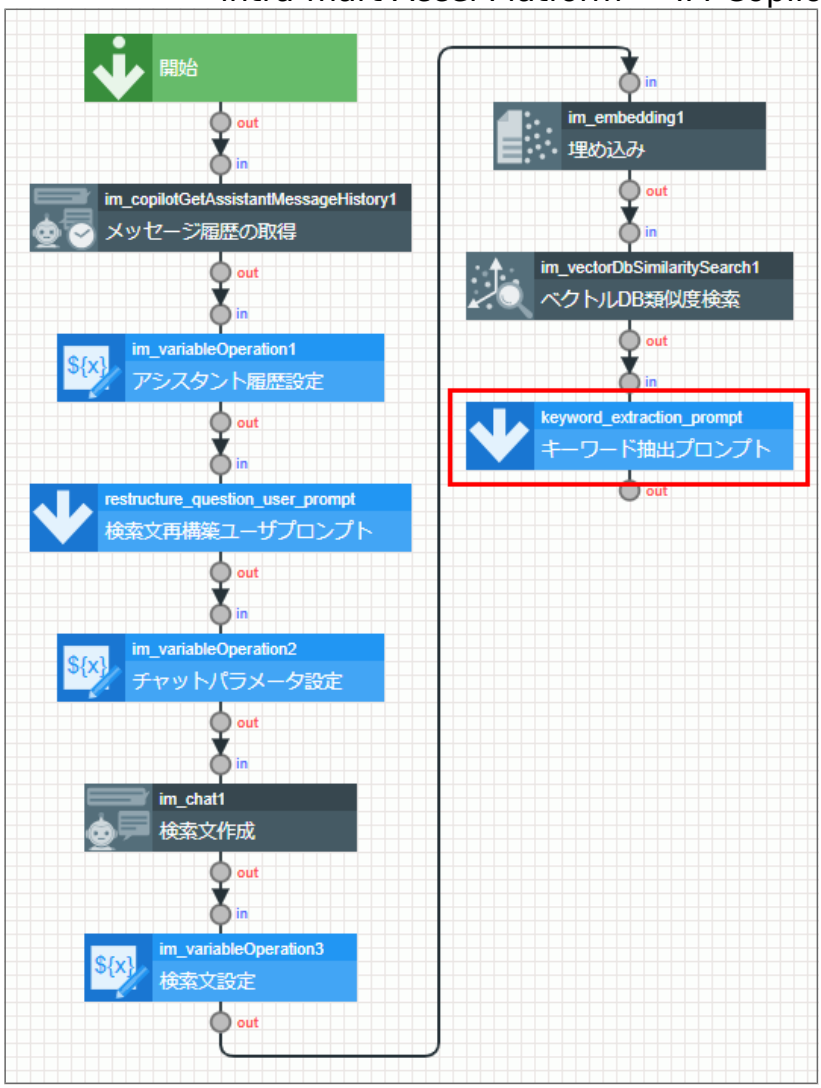
- 定数 *VECTORSTORE_CATEGORY* をタスクの入力値 *category* に設定します。
- エイリアス「*im_embedding1*」を追加します。
  - *embeddings* をタスクの入力値 *embeddings* に設定します。



キーワード検索

キーワード検索を行うことで、質問に含まれる単語やフレーズに完全一致する情報を検索できます。特に固有名詞や専門用語など、意味が変わらない単語に対して有効です。 ユーザの質問に対して完全一致する情報を検索するため、キーワード検索を行います。 検索に利用するキーワードは、生成AIによって検索文から抽出されたものを利用します。

13. ユーザ定義（テンプレート定義）を追加します。



キーワード検索を行うため、生成AIによって検索文から抽出されたキーワードを取得するため、ユーザ定義（テンプレート定義）を追加します。
パレットから「ユーザ定義追加」→「テンプレート定義新規作成」をクリックし「テンプレート定義編集」画面を表示します。
テンプレート定義編集画面の「テンプレート定義」に以下の内容を設定します。

ユーザ定義共通設定

設定項目	設定値
利用範囲	「フロー定義内のみで利用する」を有効にします。
ユーザ定義ID	*keyword_extraction_prompt*
ユーザ定義名	*キーワード抽出プロンプト*

入力値の *data <object>* 配下項目に以下の内容を設定します。

入力値 *data* 配下項目

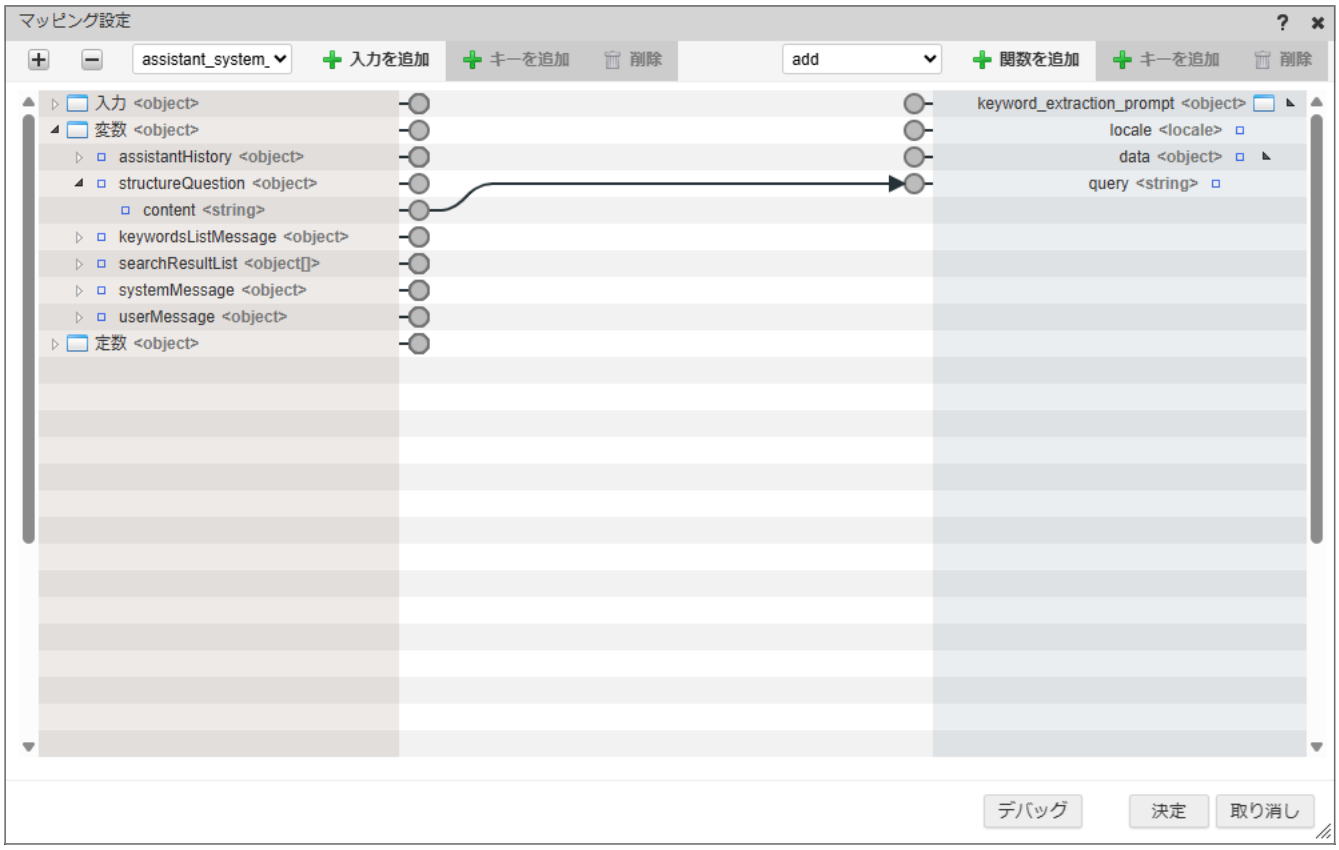
項目名	データ型
query	string

テンプレート定義の「標準」に以下の内容を設定します。

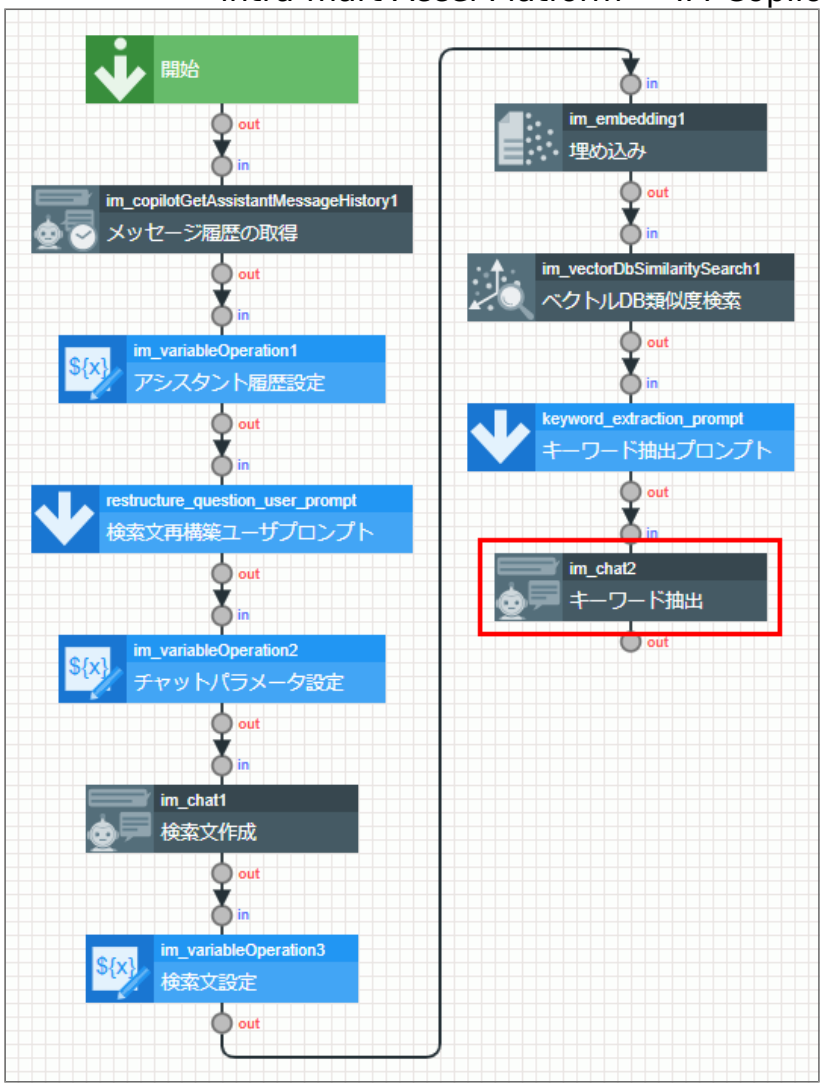
```
<#setting url_escaping_charset="UTF-8">
あなたは、質問に対する回答作成に必要な文書情報の検索をサポートするアシスタントです。
質問に関連する文書情報を検索するための検索キーワードを作成してください。
キーワードは質問文の言語に基づいて作成します。例えば質問が英語の場合は英語のキーワードを作成します。
キーワードは複数作成して構いませんが、最大でも10個までとしてください。
なお、以下のようにキーワードのみ回答してください。
キーワードはすべてカンマ（,）で区切ってください。
半角スペースで複数のキーワードをまとめてはいけません。

回答例:
キーワード１,キーワード２,キーワード３...

また、質問が不明確でキーワードを作成できない場合は、質問をそのまま回答してください。

---------------------------------

質問:
${query}
```

登録ボタンをクリックしてテンプレート定義を保存します。
「ベクトルDB類似度検索」の *out* に追加した「キーワード抽出プロンプト」の *in* に接続します。
追加した「キーワード抽出プロンプト」を選択し、マッピング設定を行います。

- 変数 *structureQuestion.content* をタスクの入力値 *data.query* に設定します。

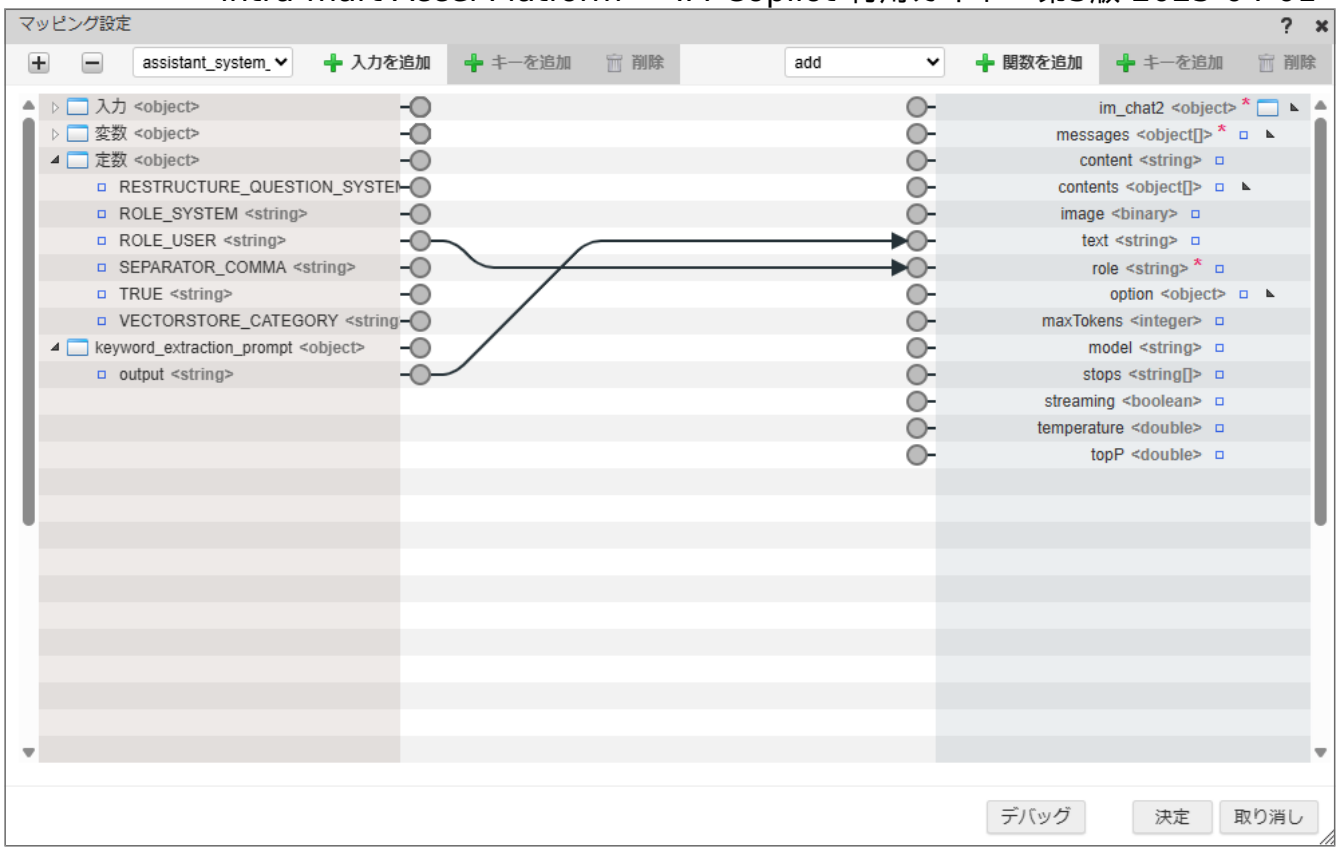


14. チャットタスクを追加します。

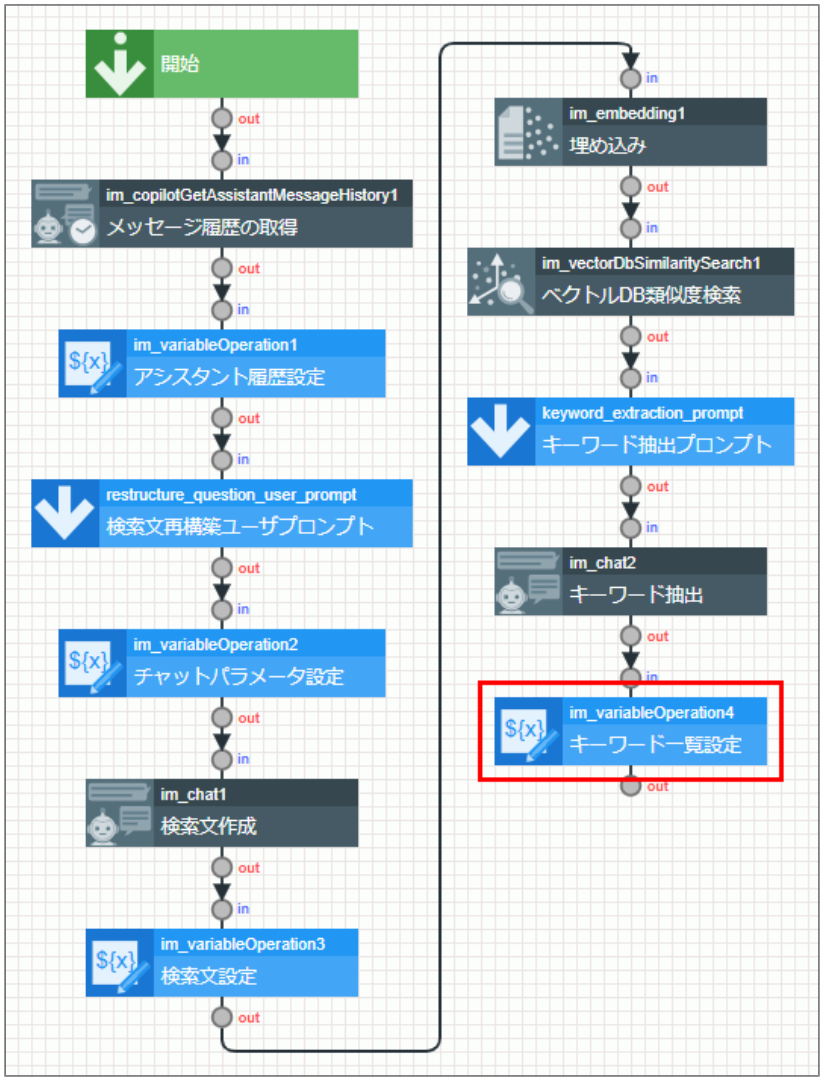


ユーザ定義（テンプレート定義）で定義したキーワード抽出の依頼を生成AIに問い合わせるため、チャットタスクを追加します。
パレットから「IM-Copilot」→「チャット」をクリックしロジックフローに追加します。
「キーワード抽出プロンプト」の *out* を追加した「チャット」の *in* に接続します。
追加した「チャット」を選択し、マッピング設定を行います。

- 定数 *ROLE_USER* をタスクの入力値 *messages.role* に設定します。
- エイリアス「*keyword_extraction_prompt*」を追加します。
  - *output* をタスクの入力値 *messages.contents.text* に設定します。



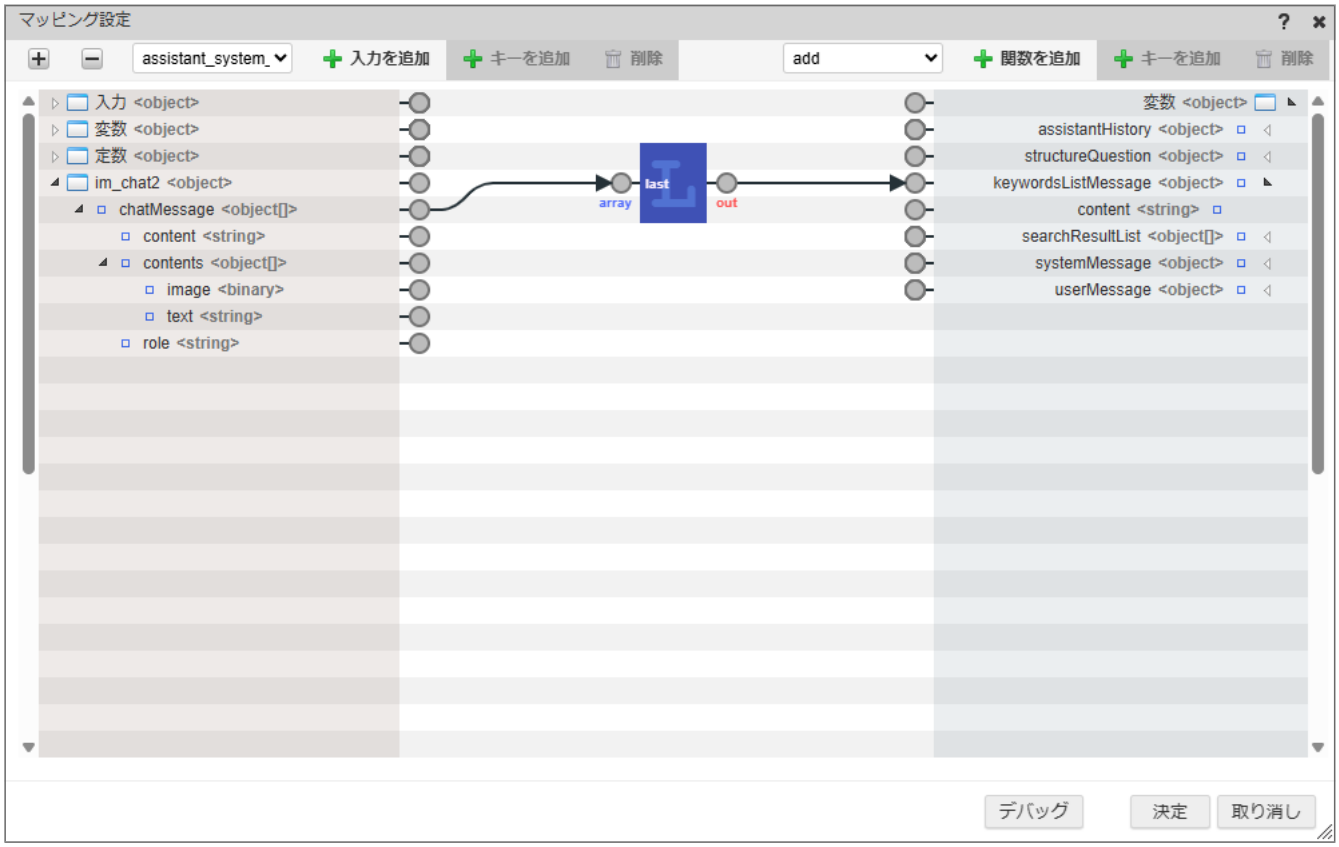
15. 変数操作タスクを追加します。



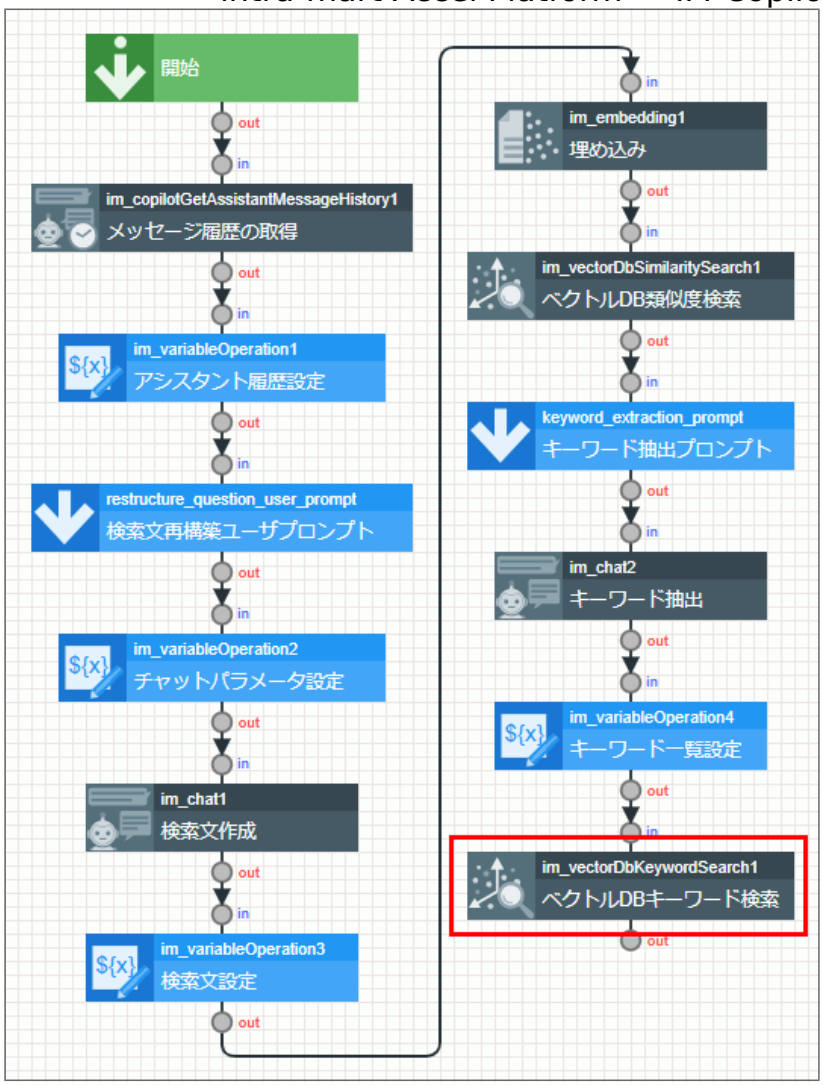
チャットタスクで取得したキーワードを変数に格納します。
パレットから「基本」→「変数操作タスク」をクリックしロジックフローに追加します。

「チャット」の *out* を追加した「変数操作タスク」の *in* に接続します。
追加した「変数操作タスク」を選択し、マッピング設定を行います。

1. エイリアス「*im_chat2*」を追加します。
2. 配列操作マッピング関数 *last* を追加します。
   - 入力値 *array* に変数 *im_chat2.chatMessage* を設定します。
   - 出力値をタスクの入力値 *keywordsListMessage* に設定します。

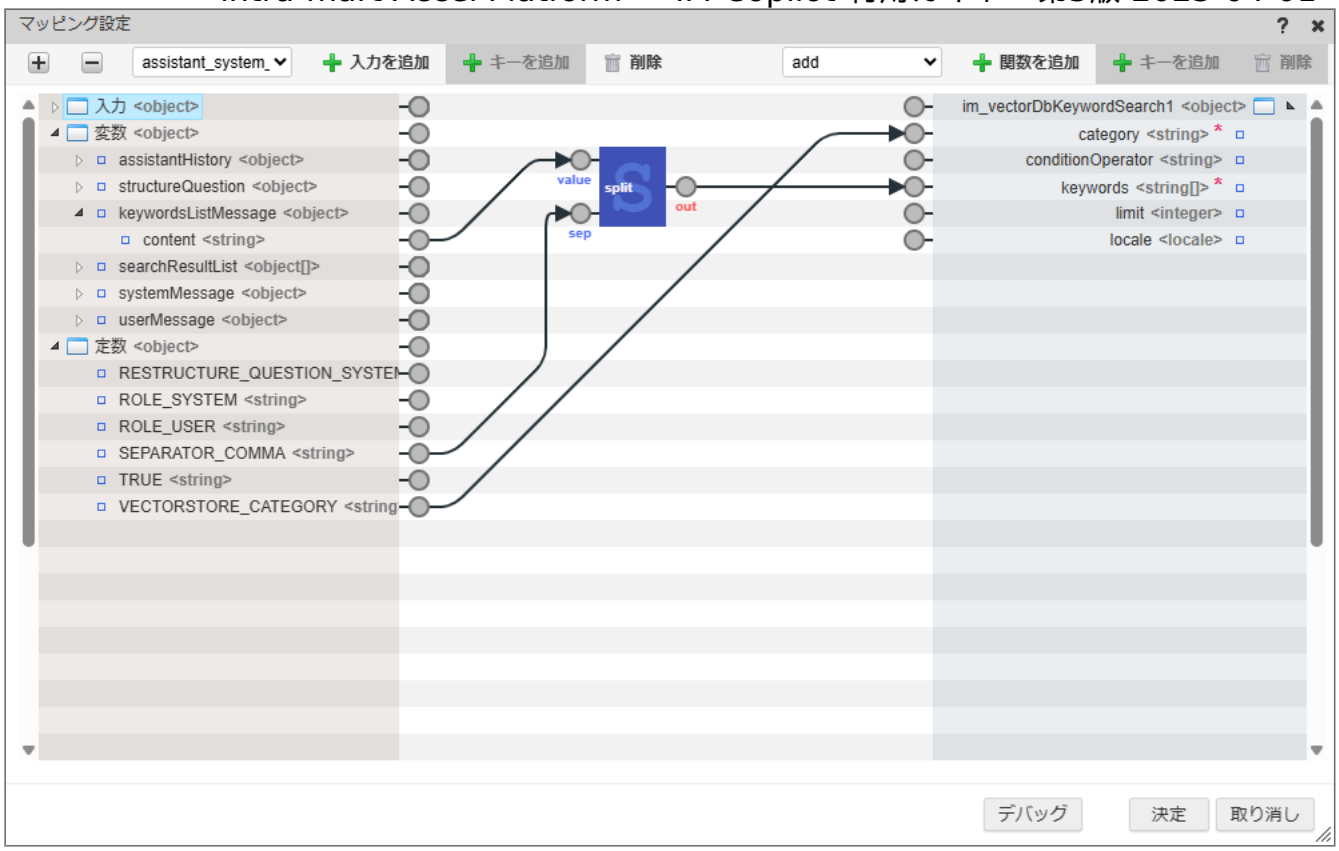


16. ベクトルDBキーワード検索タスクを追加します。



キーワード検索を行うため、キーワードを元にベクトルDBキーワード検索タスクを追加します。
パレットから「IM-Copilot」→「ベクトルDBキーワード検索」をクリックしロジックフローに追加します。
「変数操作タスク」の *out* を追加した「ベクトルDBキーワード検索」の *in* に接続します。
追加した「ベクトルDBキーワード検索」を選択し、マッピング設定を行います。

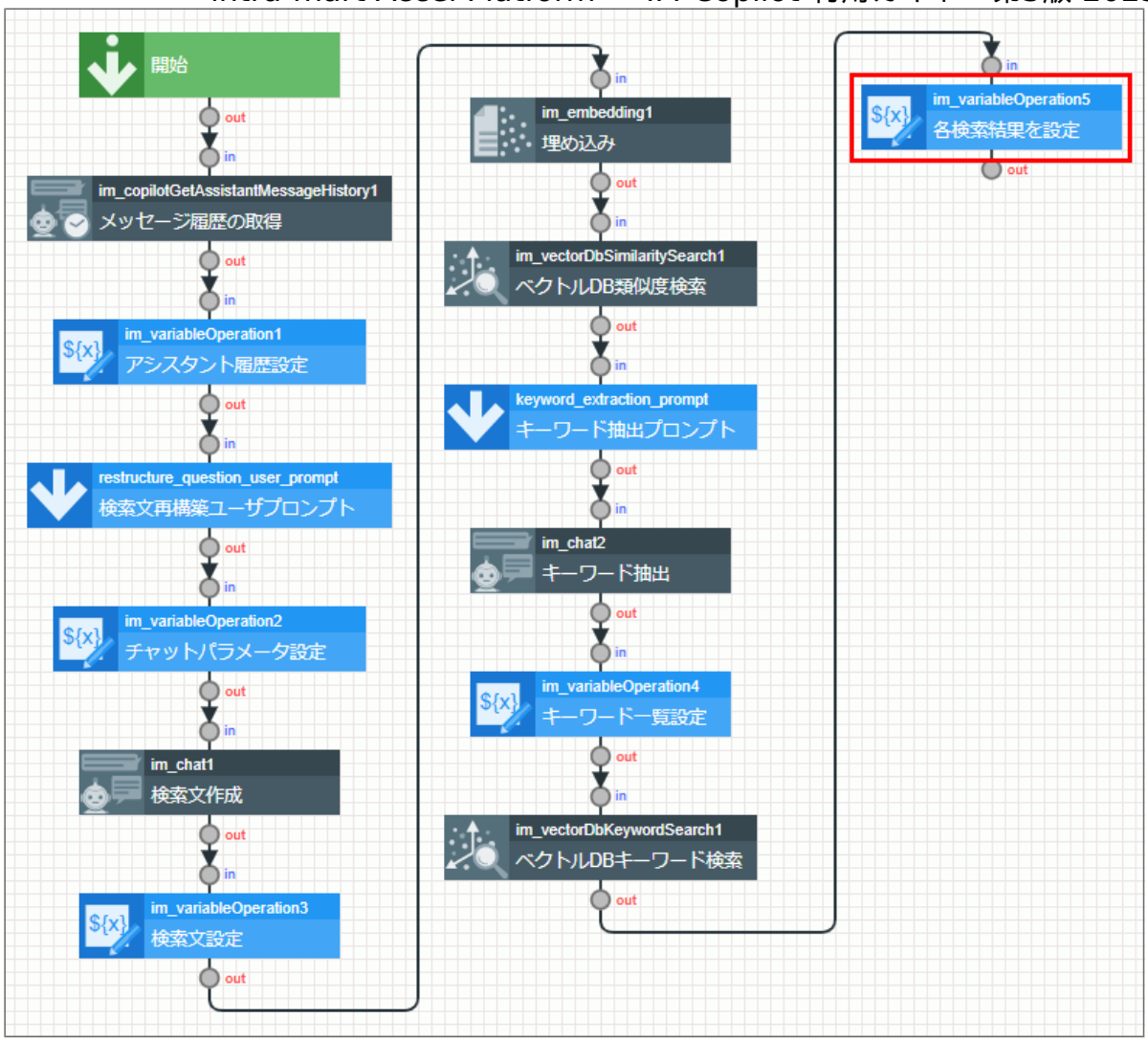
- 定数 *VECTORSTORE_CATEGORY* を入力値 *category* に設定します。
- 文字列操作マッピング関数 *split* を追加します。
  - 入力値 *value* に変数 *keywordsListMessage.content* を設定します。
  - 入力値 *sep* に定数 *SEPARATOR_COMMA* を設定します。
  - 出力値をタスクの入力値 *keywords* に設定します。



2 つの検索結果の統合

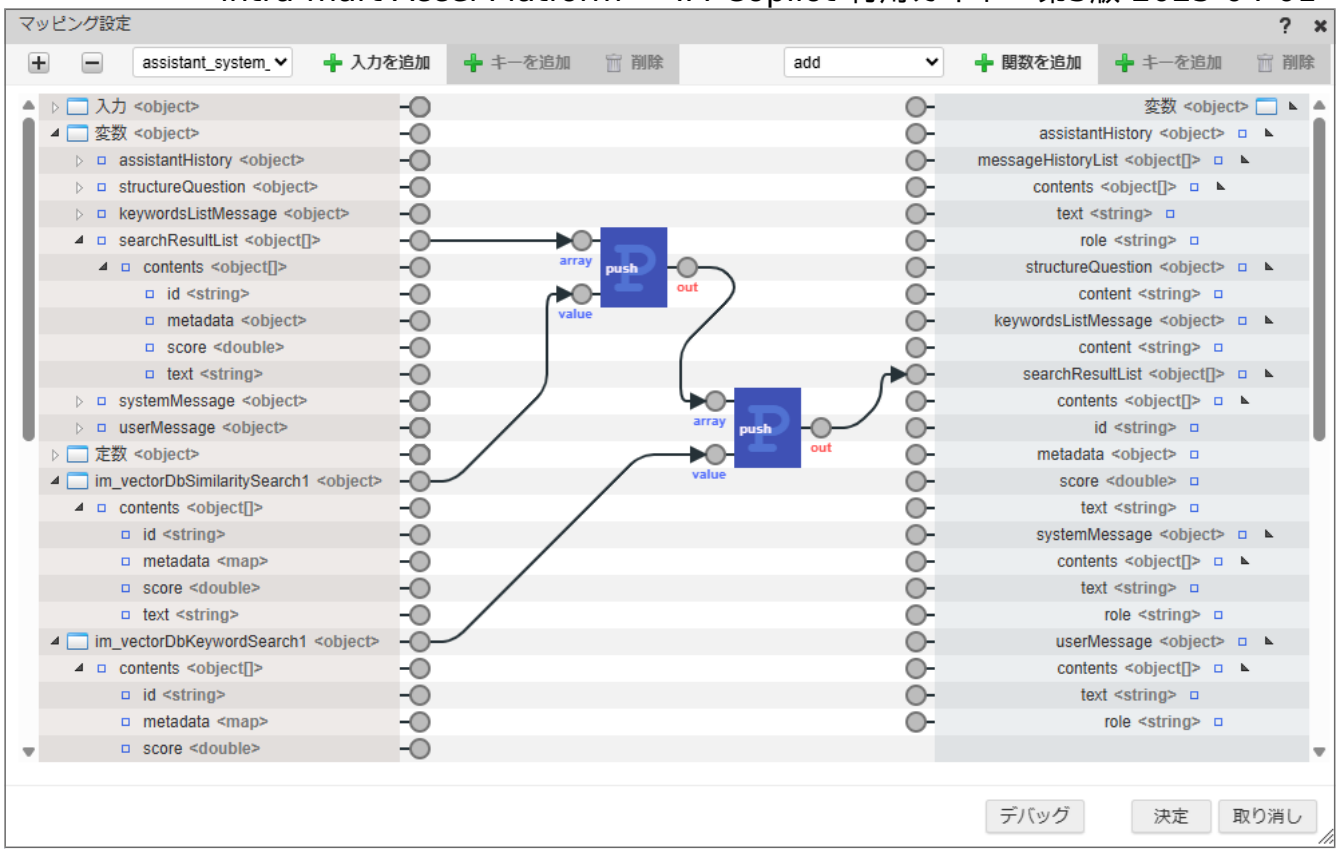
ベクトル類似度検索とキーワード検索の結果の統合を行います。検索結果の統合を行うことで、ユーザの質問に対してより適切な回答を返すことができます。

17. 変数操作タスクを追加します。

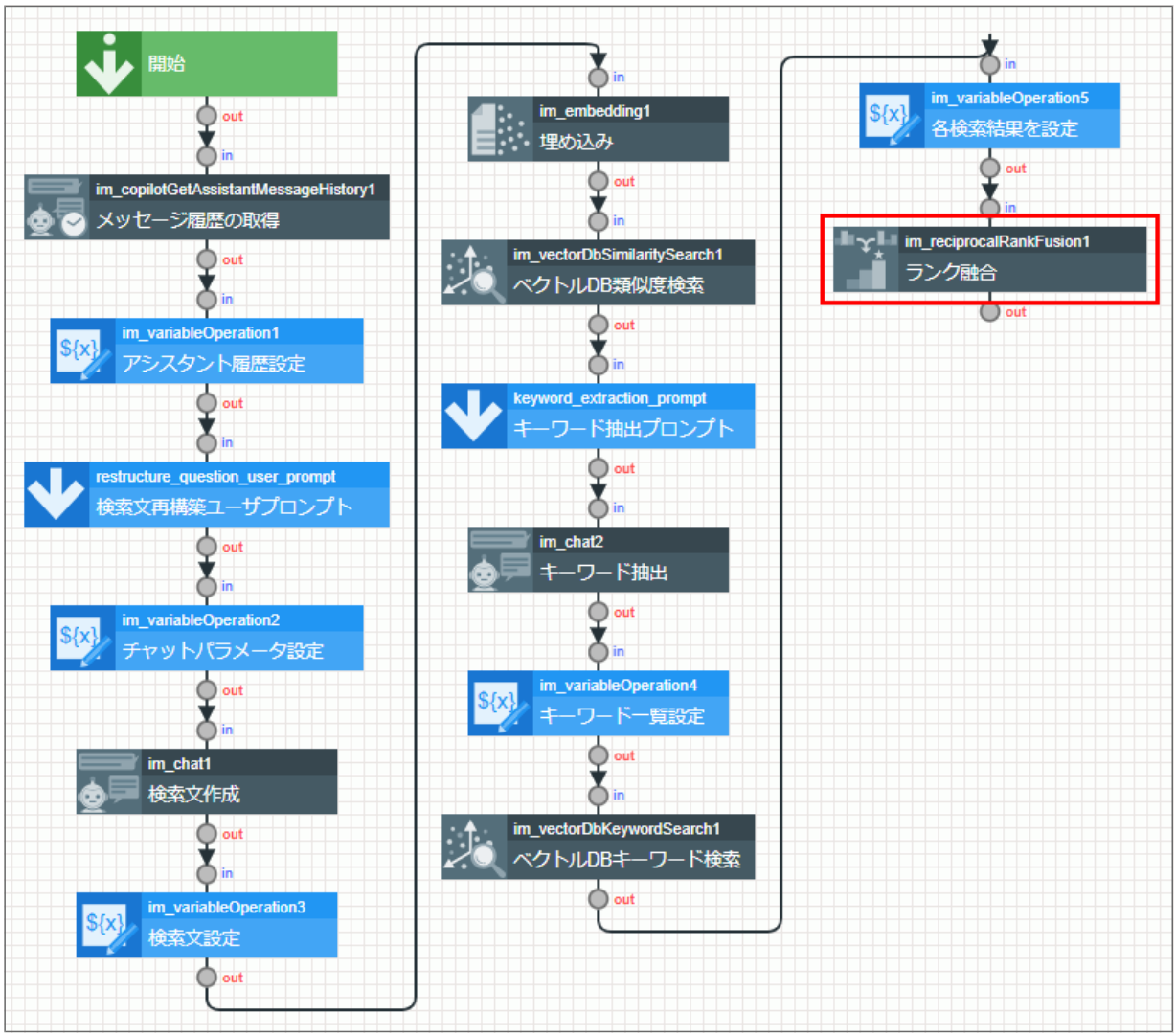


検索結果の統合を行う「ランク融合」の入力値を作成するため、変数操作タスクを追加します。
パレットから「基本」→「変数操作タスク」をクリックしロジックフローに追加します。
「ベクトルDBキーワード検索」の *out* を追加した「変数操作タスク」の *in* に接続します。
追加した「変数操作タスク」を選択し、マッピング設定を行います。

1. エイリアス「*im_vectorDbSimilaritySearch1*」を追加します。
2. エイリアス「*im_vectorDbKeywordSearch1*」を追加します。
3. 配列操作マッピング関数 *push* を追加します。
   - 入力値 *array* に変数 *searchResultList* を設定します。
   - 入力値 *value* にエイリアス *im_vectorDbSimilaritySearch1* を設定します。
4. 配列操作マッピング関数 *push* を追加します。
   - 入力値 *array* に上記で追加したマッピング関数の出力値を設定します。
   - 入力値 *value* にエイリアス *im_vectorDbKeywordSearch1* を設定します。
   - 出力値をタスクの入力値 *searchResultList* に設定します。



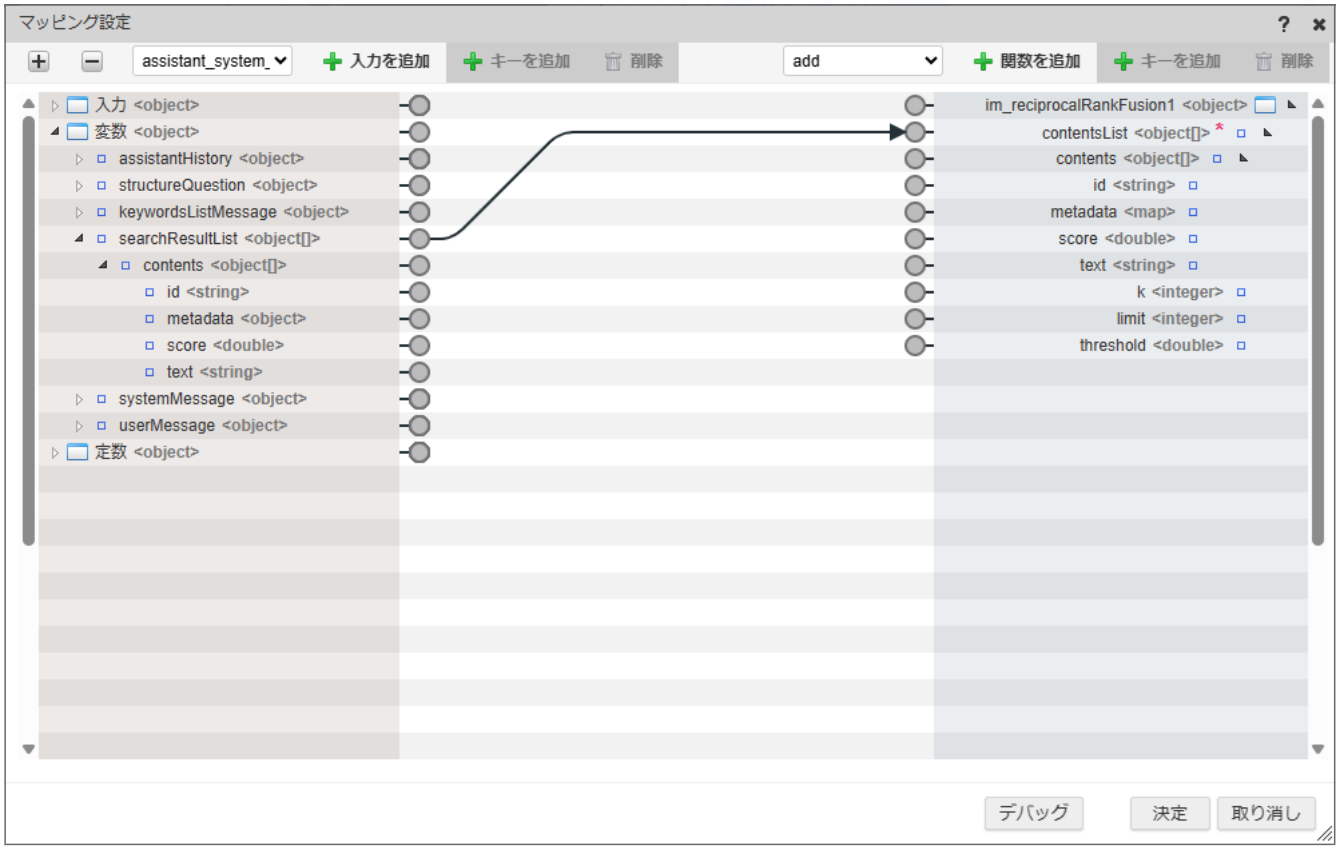
18. ランク融合タスクを追加します。



ベクトル類似度検索とキーワード検索の結果を統合するため、ランク融合タスクを追加します。
パレットから「IM-Copilot」→「ランク融合」をクリックしロジックフローに追加します。

「変数操作タスク」の *out* を追加した「ランク融合」の *in* に接続します。
追加した「ランク融合」を選択し、マッピング設定を行います。

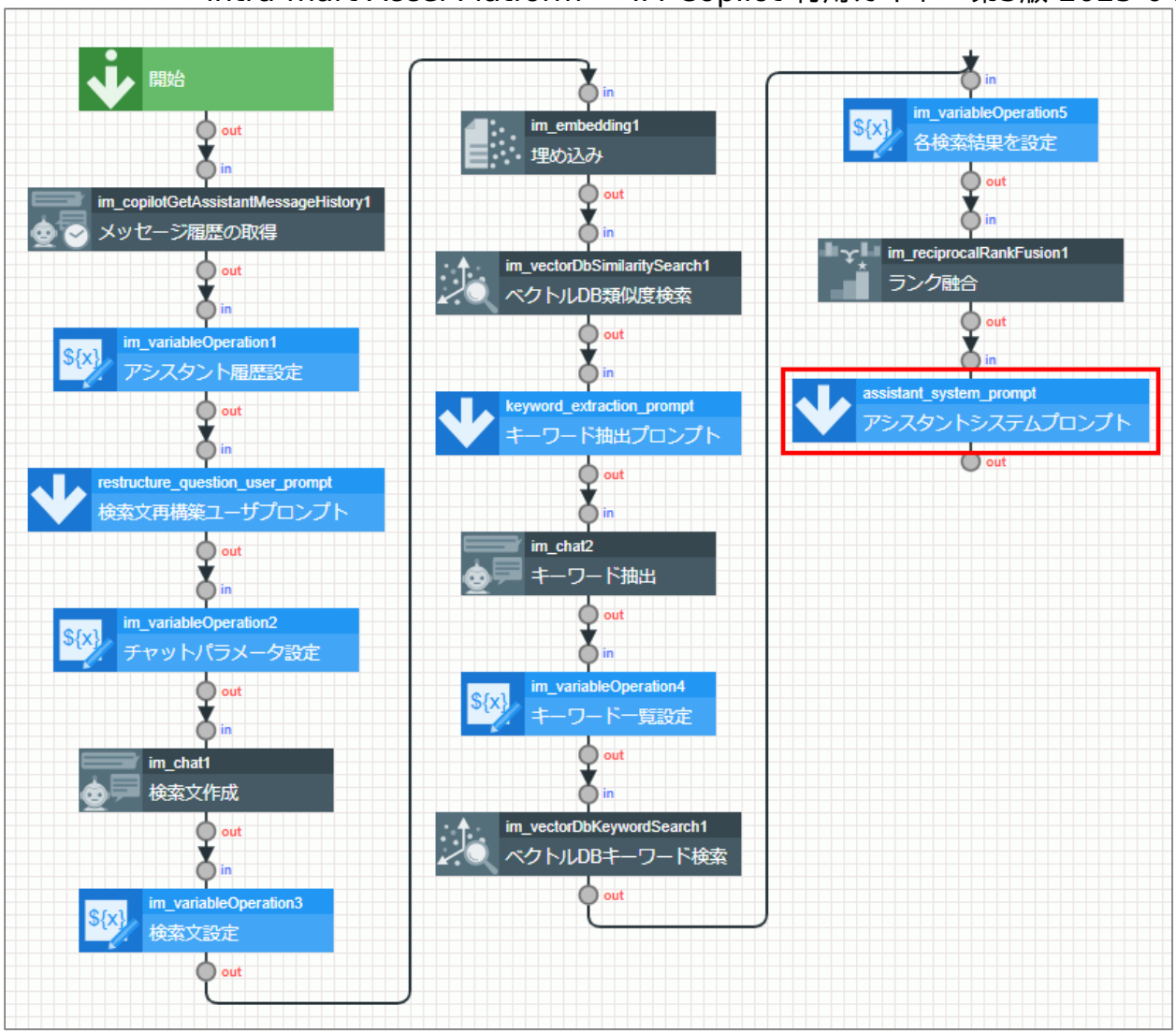
- 変数 *searchResultList* をタスクの入力値 *contentsList* に設定します。



生成AIを用いたアシスタント応答

ユーザの質問に対して適切な回答を返すため、生成AIを利用してアシスタント応答を生成します。

19. ユーザ定義（テンプレート定義）を追加します。



アシスタント応答を生成するため、生成AIに問い合わせる際に利用するシステムプロンプトを定義するユーザ定義（テンプレート定義）を追加します。
パレットから「ユーザ定義追加」→「テンプレート定義新規作成」をクリックし「テンプレート定義編集」画面を表示します。
テンプレート定義編集画面の「テンプレート定義」に以下の内容を設定します。

ユーザ定義共通設定

設定項目	設定値
利用範囲	「フロー定義内のみで利用する」を有効にします。
ユーザ定義ID	*assistant_system_prompt*
ユーザ定義名	*アシスタントシステムプロンプト*

入力値の *data <object>* 配下項目に以下の内容を設定します。

入力値 *data* 配下項目

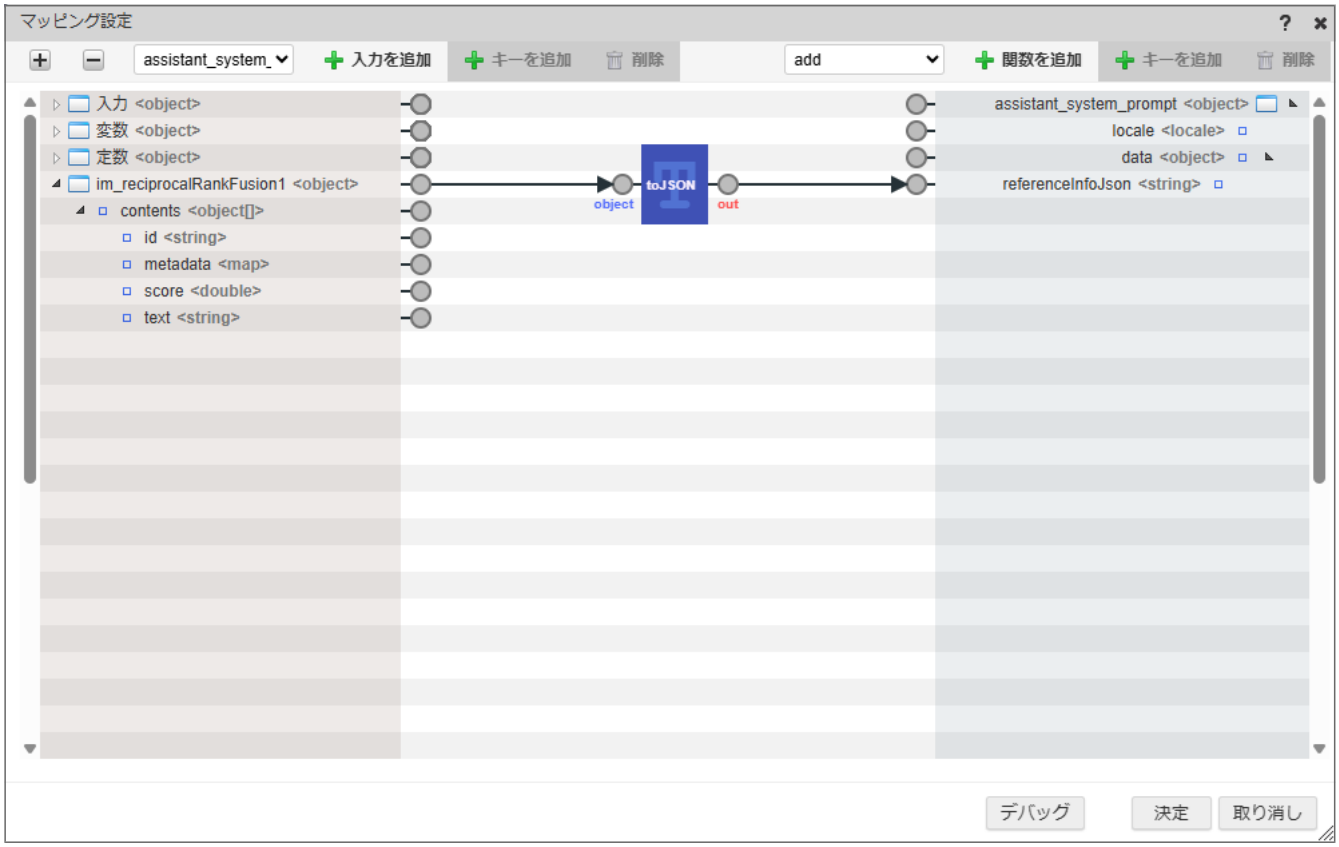
項目名	データ型
referenceInfoJson	string

テンプレート定義の「標準」に以下の内容を設定します。

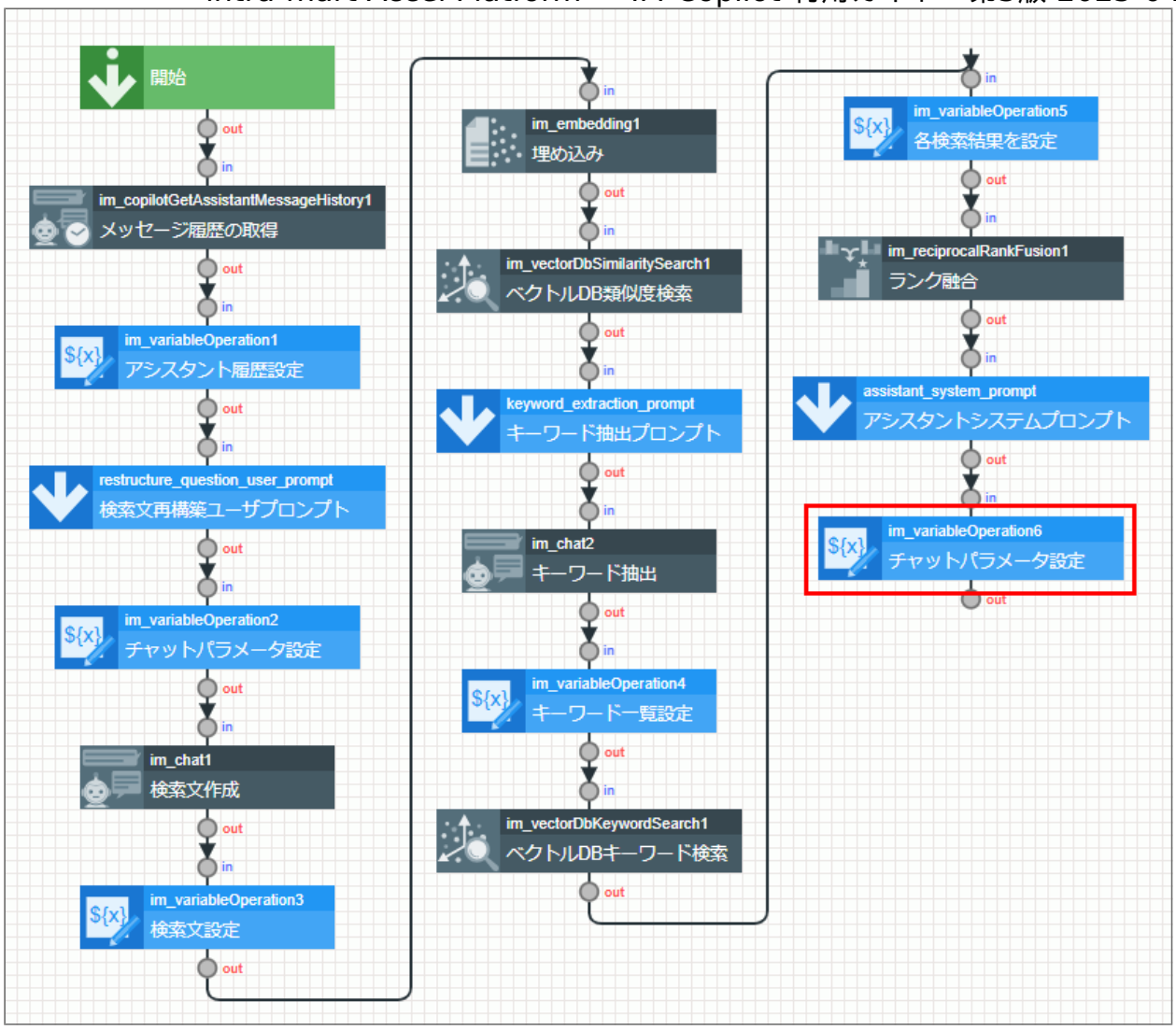
```
<#setting url_escaping_charset="UTF-8">
あなたは、参考情報のみに基づいてユーザーの質問に答えるアシスタントです。次のガイドラインに従ってください：

1. 参考情報に含まれる情報のみを使用して回答して下さい。
2. 回答は可能な限り詳細に行い、参考情報の該当部分を参照しながら説明して下さい。
3. 差別的、攻撃的、またはわいせつな言葉を使用する内容について回答できないと出力して下さい。
4. 参考情報が無い場合には、回答の根拠の提示は出来ないと伝えつつ、回答できる場合は回答して下さい、そうでない場合は回答できないと出力して下さい。

参考情報は以下の形式で提供されます：
${referenceInfoJson}

このデータにのみ基づいて回答を行ってください。
ユーザーが質問で利用している言語と同じ言語で回答してください。
```

登録ボタンをクリックしてテンプレート定義を保存します。
「ランク融合」の *out* に追加した「アシスタントシステムプロンプト」の *in* に接続します。
追加した「アシスタントシステムプロンプト」を選択し、マッピング設定を行います。

1. エイリアス「*im_reciprocalRankFusion1*」を追加します。
2. JSONマッピング関数 *toJSON* を追加します。
   - 入力値 *object* にエイリアス *im_reciprocalRankFusion1* を設定します。
   - 出力値をタスクの入力値 *data.referenceInfoJson* に設定します。

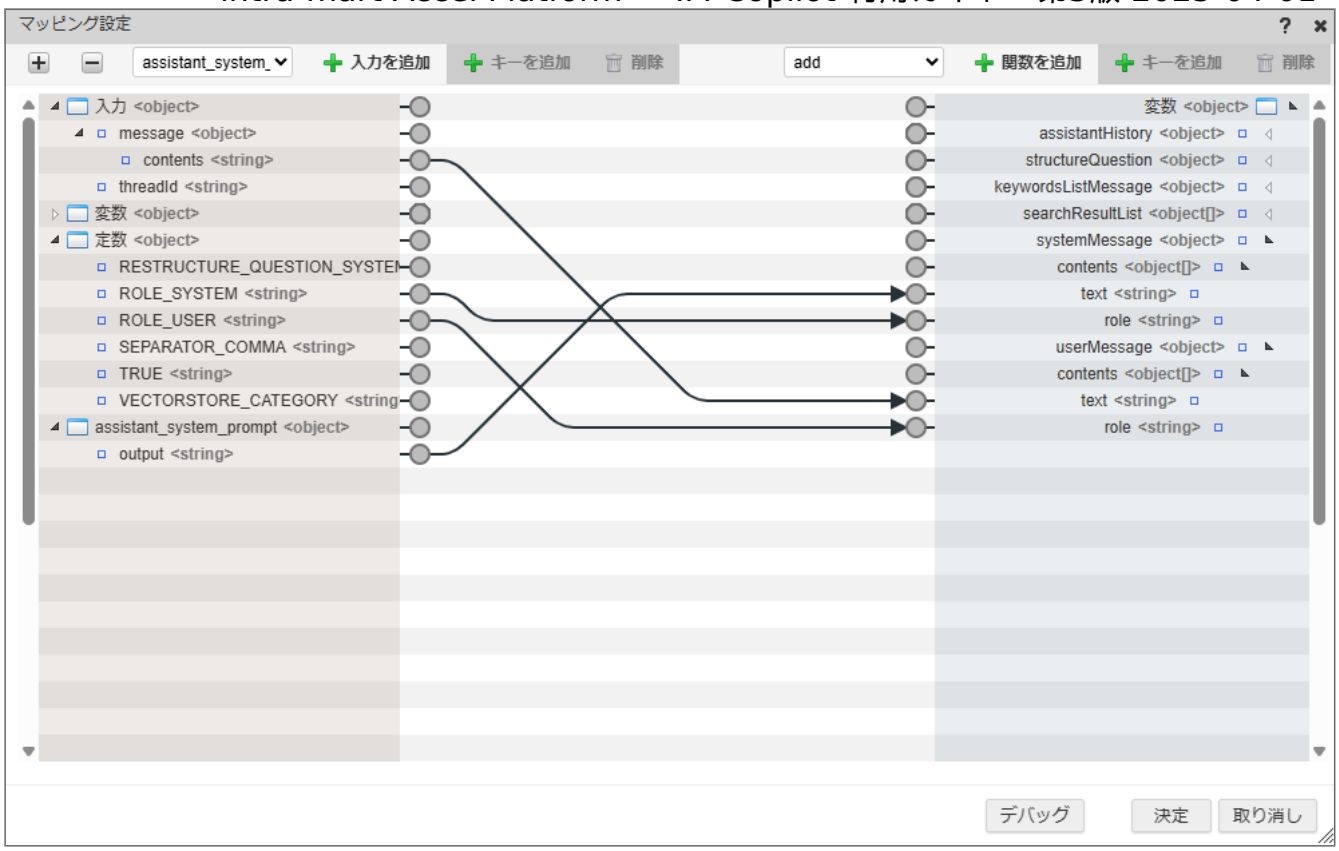


20. 変数操作タスクを追加します。

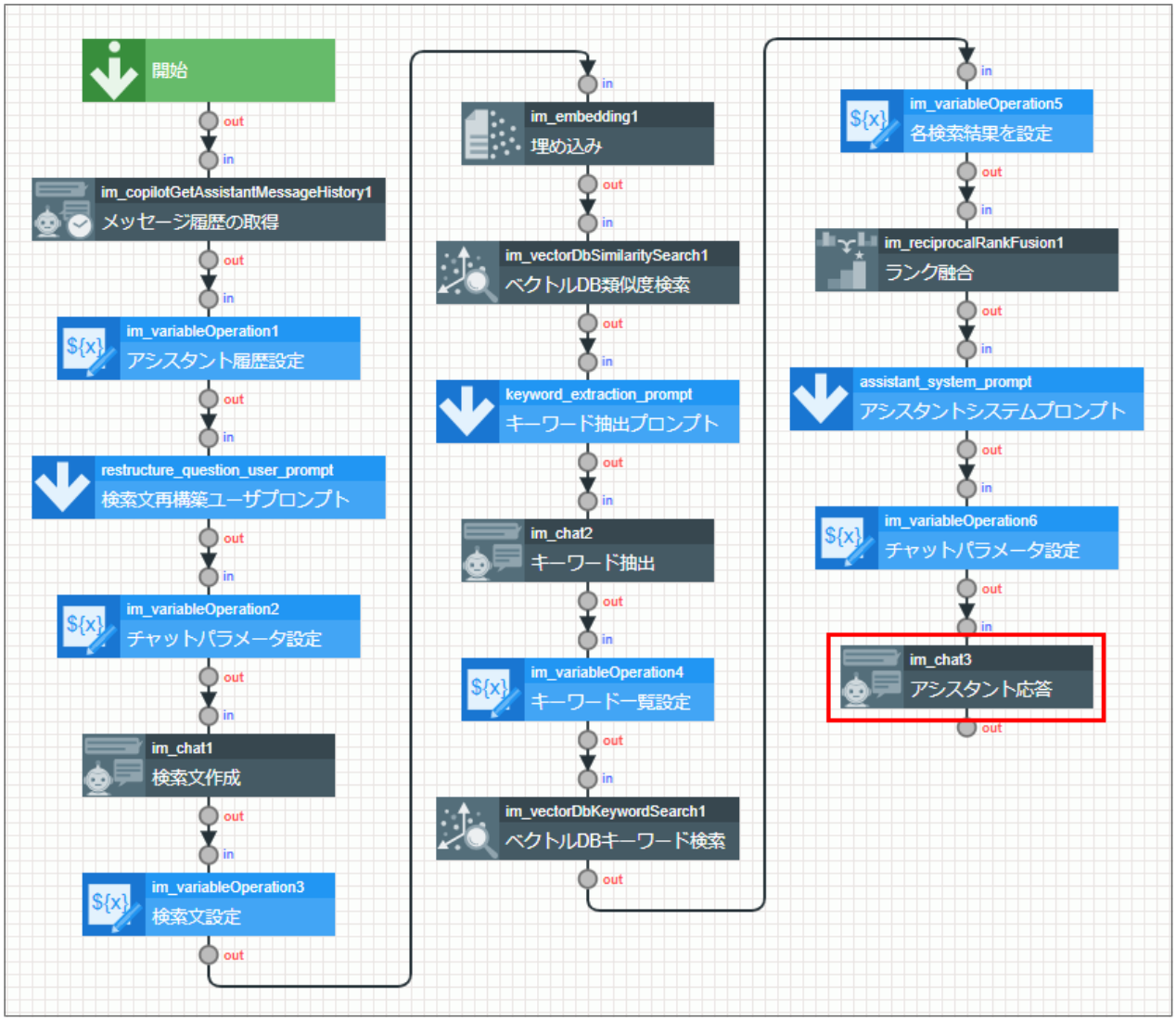


生成AIに問い合わせる際に利用するシステムプロンプト、および、ユーザプロンプトを変数に格納します。
パレットから「基本」→「変数操作タスク」をクリックしロジックフローに追加します。
「アシスタントシステムプロンプト」の *out* を追加した「変数操作タスク」の *in* に接続します。
追加した「変数操作タスク」を選択し、マッピング設定を行います。

- 定数 *ROLE_SYSTEM* をタスクの入力値 *systemMessage.role* に設定します。
- エイリアス「*assistant_system_prompt*」を追加します。
  - *output* をタスクの入力値 *systemMessage.contents.text* に設定します。
- 定数 *ROLE_USER* をタスクの入力値 *userMessage.role* に設定します。
- 入力 *message.contents* をタスクの入力値 *userMessage.contents.text* に設定します。



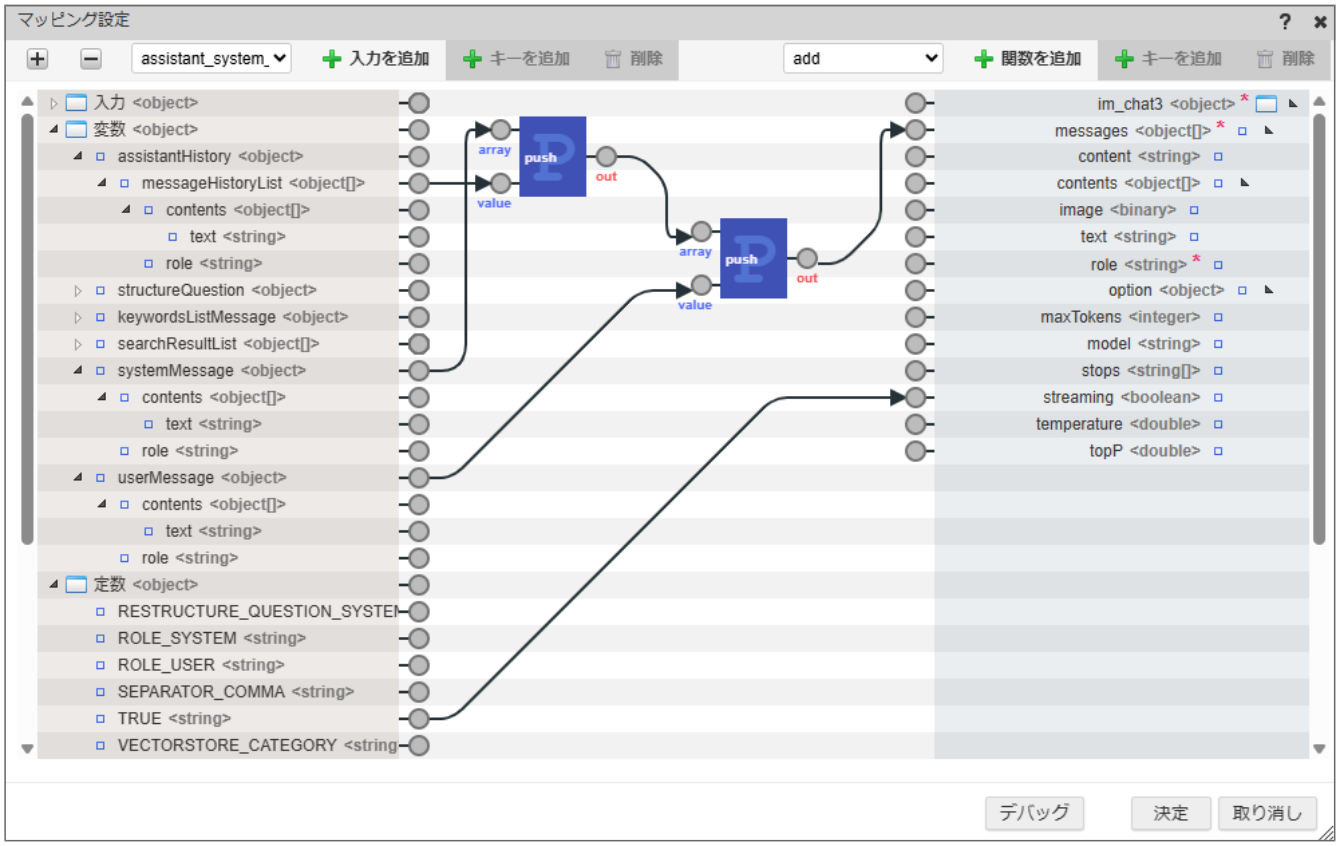
21. チャットタスクを追加します。



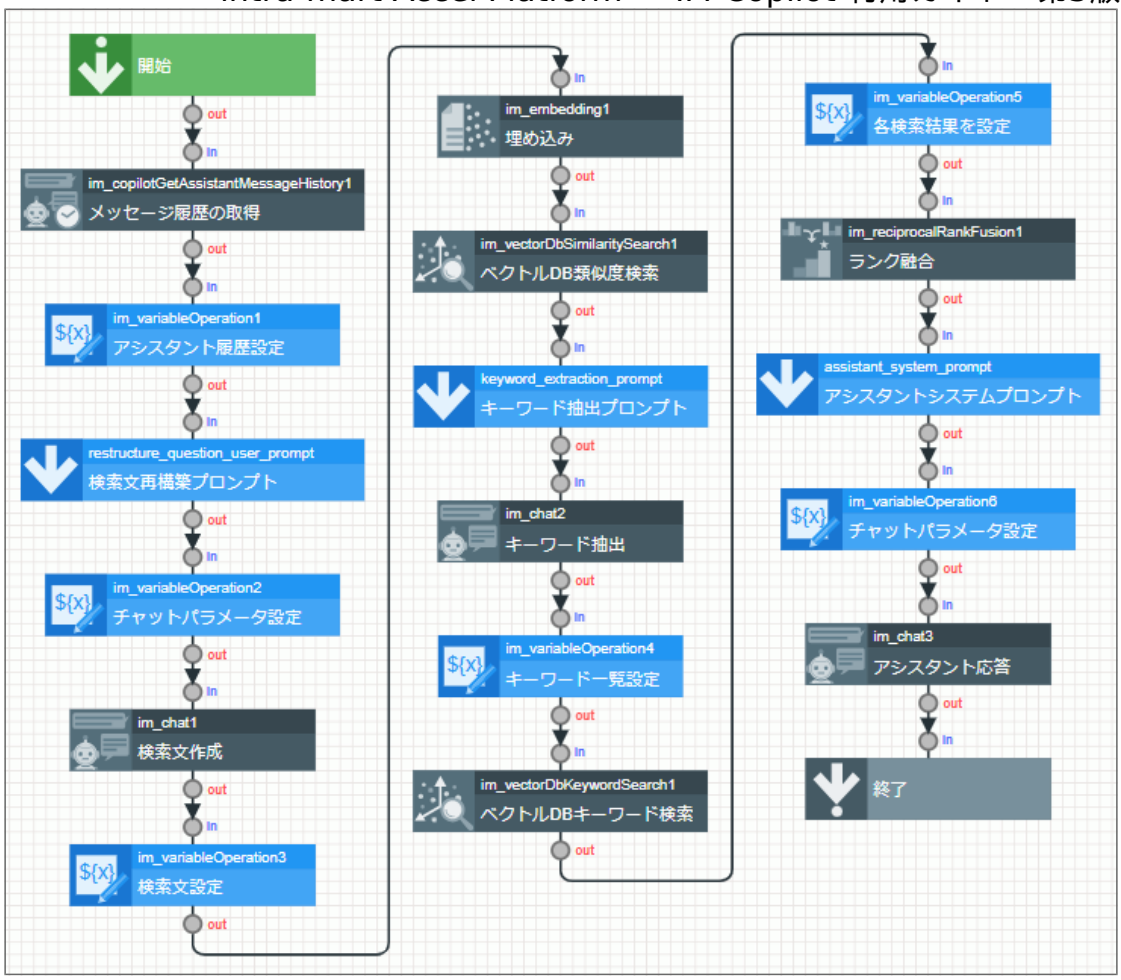
検索した情報を基に生成AIに問い合わせ、その応答をクライアントに返すために、チャットタスクを追加します。
パレットから「IM-Copilot」→「チャット」をクリックしロジックフローに追加します。

「変数操作タスク」の *out* を追加した「チャット」の *in* に接続します。
追加した「チャット」を選択し、マッピング設定を行います。

- 配列操作マッピング関数 *push* を追加します。
  - 入力値 *array* に変数 *systemMessage* を設定します。
  - 入力値 *value* に変数 *assistantHistory.messageHistoryList* を設定します。
- 配列操作マッピング関数 *push* を追加します。
  - 入力値 *array* に上記で追加したマッピング関数の出力値を設定します。
  - 入力値 *value* に変数 *userMessage* を設定します。
  - 出力値をタスクの入力値 *messages* に設定します。
- 定数 *TRUE* をタスクの入力値 *streaming* に設定します。



「チャット」タスクの *out* を「終了」エレメントの *in* に接続します。



コラム

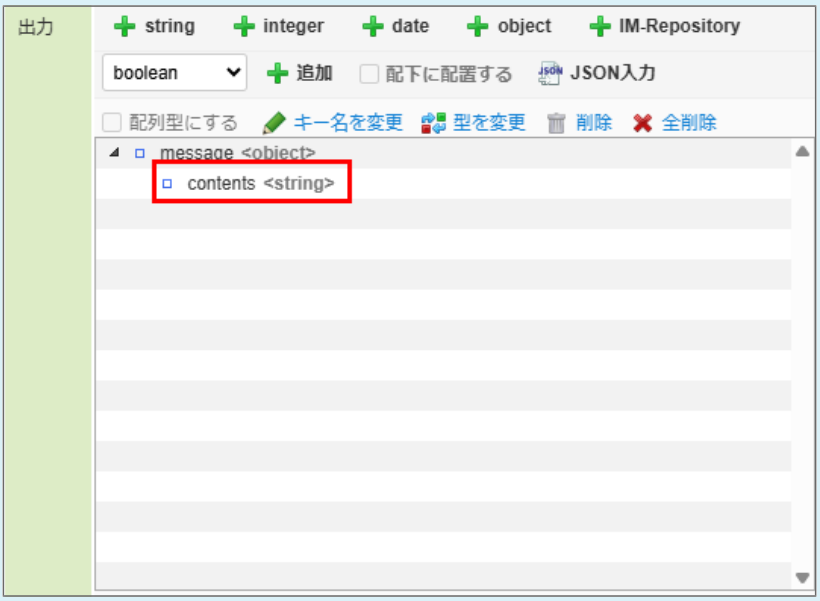
ストリーミング形式でのクライアントへの応答について

生成AIに問い合わせる際に、アシスタント応答をストリーミング形式で返すためチャットタスクの *streaming* オプションに、*true* を設定します。
ストリーミング形式で応答を返すことにより、アシスタント応答をリアルタイムにクライアントに返すことができます。

コラム

チャットタスクを利用しないクライアントへの応答について

チャットタスクを利用せずにアシスタント応答を返す場合、応答内容をロジックフローの出力値 *message.contents* に文字列として設定することでクライアントに応答を返すことができます。
この場合、あらかじめロジックフローの入出力設定で出力値を設定し、「終了」エレメントのマッピングで出力値に文字列を指定します。



22. ロジックフローの保存を行います。

    ロジックフロー定義編集画面ツールバーの「保存」をクリックしロジックフローを保存します。
    以下の値で保存します。

    - フロー定義ID: *sample-rag-assistant*
    - フロー定義名: *サンプルRAGアシスタント*

アシスタントの設定

アシスタント定義の作成

作成したロジックフローをアシスタントとして動作させるため、アシスタント定義を作成します。

1. アシスタント定義一覧画面を表示します。
   「サイトマップ」→「Copilot」→「アシスタント定義」→「アシスタント定義一覧」をクリックし、「アシスタント定義一覧」画面を表示します。

2. アシスタント定義の新規作成を行います。
   「*アシスタント定義を登録する*」を参考にロジックフローアシスタントのアシスタント定義を新規作成します。
   以下の値でアシスタント定義を作成します。
   - アシスタント定義ID: *sample-rag-assistant*
   - アシスタント定義名: *サンプルRAGアシスタント*

アシスタントの認可設定

アシスタントを利用する権限をユーザに付与するため、アシスタントの認可設定を行います。

1. アシスタント定義一覧画面を表示します。
   「サイトマップ」→「Copilot」→「アシスタント定義」→「アシスタント定義一覧」をクリックし、「アシスタント定義一覧」画面を表示します。

2. アシスタントの認可設定を行います。
   作成したアシスタント定義「サンプルRAGアシスタント」を一覧に表示します。

「*アシスタント定義の認可設定をする*」を参考に作成したアシスタントの認可設定を行います。
「アシスタントの認可設定」画面で「認可設定を開始する」をクリックし、リソース「サンプルRAGアシスタント」に対して「認証済みユーザ」を許可に設定します。

> **コラム**
> 本セクションでは一例として「認証済みユーザ」を許可に設定します。運用に合わせて適切な設定を行ってください。

アシスタントの動作確認

作成したアシスタントは「*共通アシスタント実行画面*」で動作確認が行えます。
「*共通アシスタント実行画面*」でサンプルRAGアシスタントを選択して、アシスタントの動作確認を行ってください。

> **コラム**
> 作成したロジックフローアシスタントの実行にはベクトルデータ構築が必要です。ベクトルデータ構築が完了していない場合は、アシスタントの動作確認ができません。
> 「*ベクトルデータ構築ジョブの作成*」で作成したジョブを実行の後、アシスタントの動作確認を行ってください。
> ジョブネット管理画面で「*ジョブネットの作成*」で作成したジョブネットを選択して「即時実行」を行ってください。

画面の作成

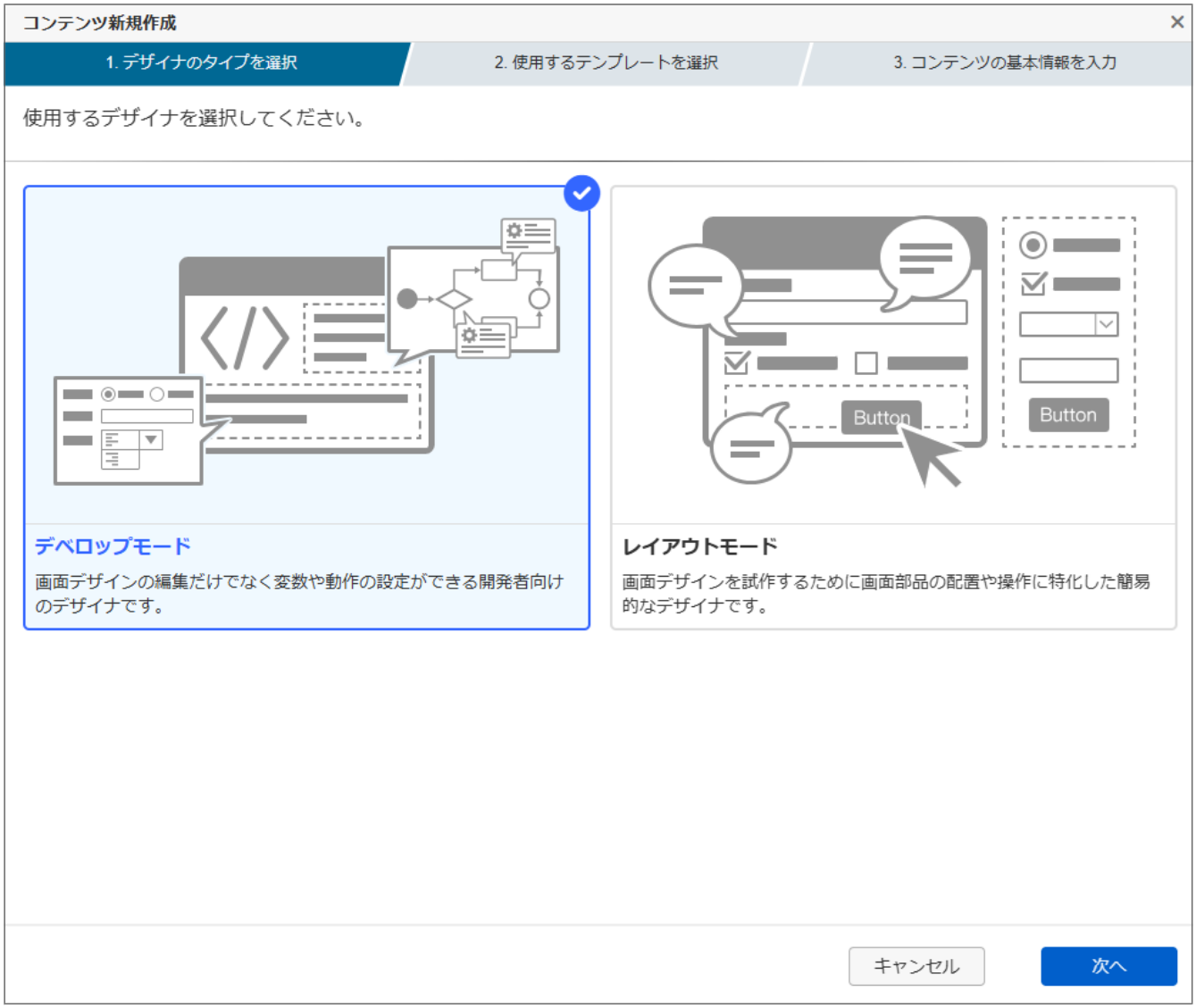
IM-BloomMaker を用いてアシスタントの画面を作成します。

コンテンツの作成

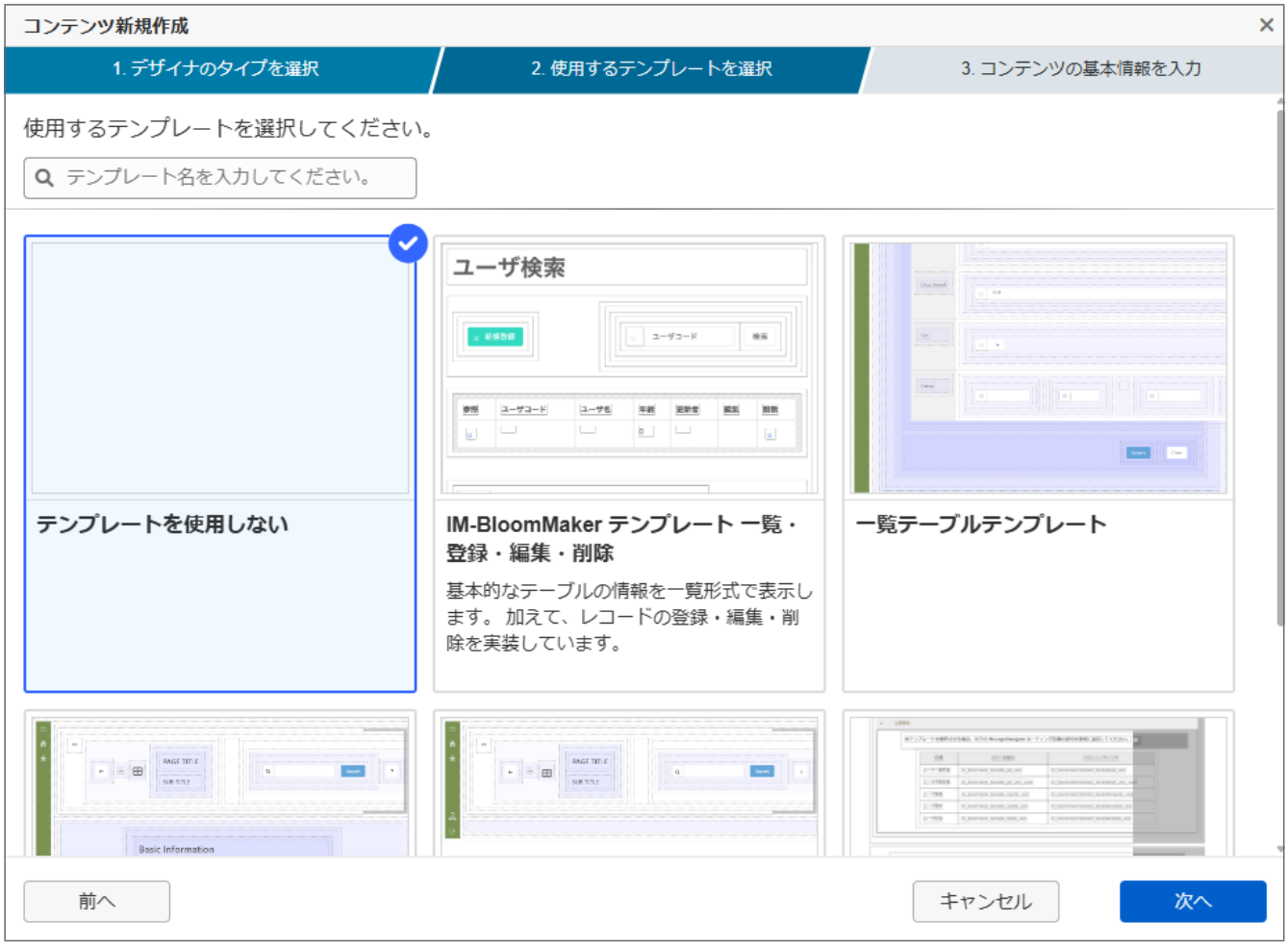
本セクションでは、アシスタントのチャット画面のみを含むシンプルな画面を作成します。

1. IM-BloomMaker コンテンツ一覧画面を開きます。
   「サイトマップ」→「BloomMaker」→「コンテンツ一覧」の順にクリックし、「コンテンツ一覧」画面を表示します。

2. コンテンツを新規作成します。
   コンテンツを新規登録します。
   右側のカテゴリツリーでコンテンツを追加するカテゴリを選択し、ツールバーの「コンテンツ新規作成」ボタンをクリックします。必要に応じてカテゴリの登録を行ってください。
   コンテンツ新規登録ダイアログが表示されます。
   デザイナのタイプを選択で「デベロップモード」を選択し、「次へ」ボタンをクリックします。



使用するテンプレートを選択で「テンプレートを使用しない」を選択し、「次へ」ボタンをクリックします。



コンテンツの基本情報入力欄に以下を入力します。

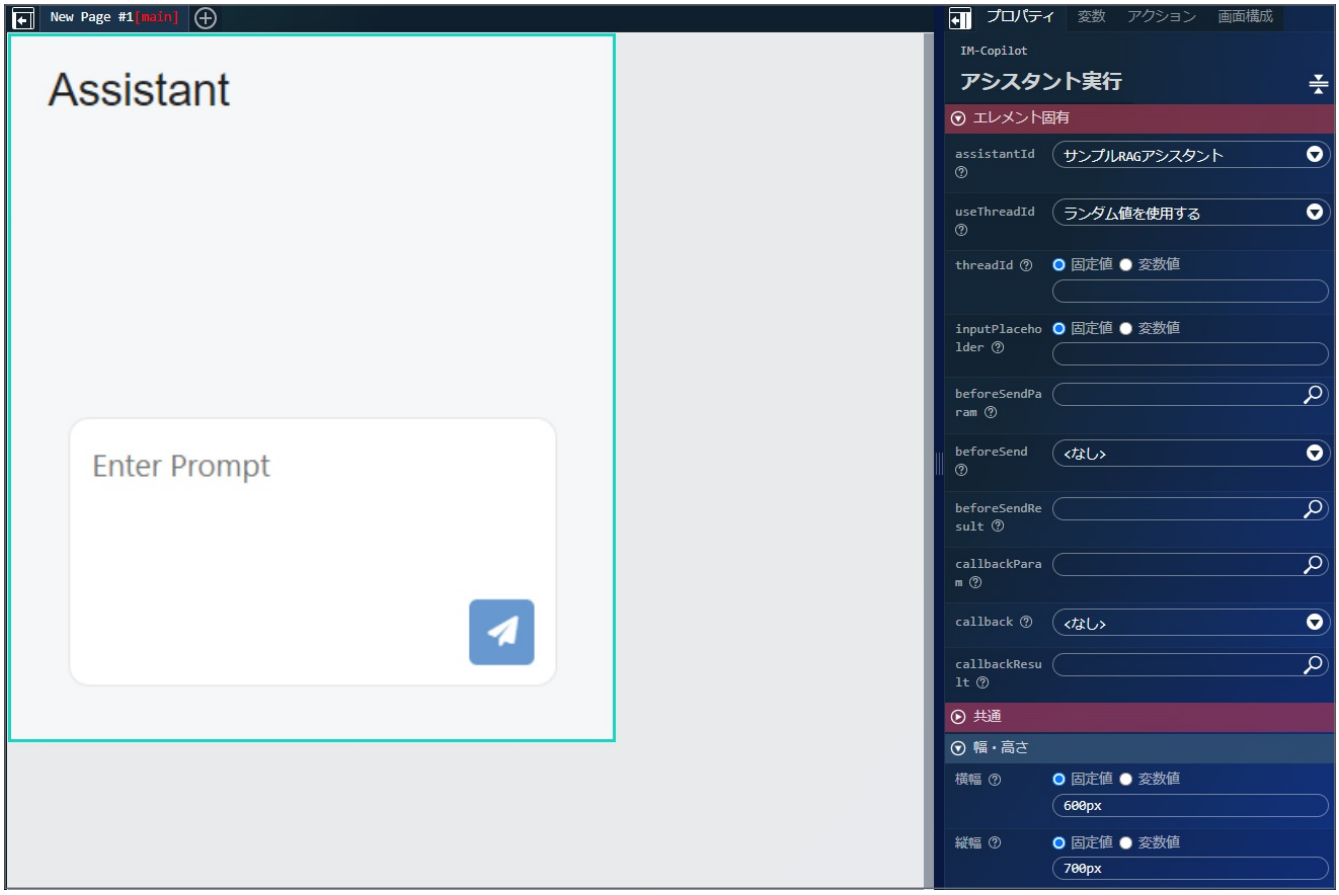
- コンテンツID: *sample_rag_assistant*
- コンテンツ名: *サンプルRAGアシスタント*
- コンテンツ種別: *imds*

入力が完了したら「登録」ボタンをクリックします。コンテンツが登録され、コンテンツのデザイナ画面が表示されます。

3. チャット画面を作成します。
   左側のパレットから「IM-Copilot」→「アシスタント実行」をドラッグ＆ドロップし、コンテンツに配置します。

配置したアシスタント実行エレメントを選択して、右側のプロパティペインで以下を設定します。

- エレメント固有
  - assistantId: *サンプルRAGアシスタント*
    - 「*アシスタント作成*」 で作成したアシスタントを選択します。
  - useThreadId: *ランダムな値を使用する*
    - メッセージ履歴を利用するアシスタントであるためスレッドIDを指定します。
- 幅・高さ
  - 横幅: *600px*
  - 縦幅: *700px*



ヘッダの左端にある「上書き保存」ボタンをクリックして、コンテンツを保存します。

ルーティングの作成

ユーザが作成したコンテンツにアクセスするためのルーティングを作成します。

1. IM-BloomMaker ルーティング定義一覧画面を開きます。
   「サイトマップ」→「BloomMaker」→「ルーティング定義一覧」の順にクリックし、「ルーティング定義一覧」画面を表示します。

2. ルーティングを新規作成します。
   ルーティングを新規登録します。
   右側のカテゴリツリーでルーティングを追加するカテゴリを選択し、ツールバーの「ルーティング新規作成」ボタンをクリックします。
   必要に応じてカテゴリの登録を行ってください。
   ルーティング情報入力欄が表示されます。
   ルーティング情報入力欄に以下の情報を入力します。
   - ルーティングID: *sample_rag_assistant*
   - コンテンツ: 「*コンテンツの作成*」 で作成したコンテンツを選択します
   - URL: *sample_rag_assistant*
   - ルーティング名: *サンプルRAGアシスタント*

入力が完了したら「登録」ボタンをクリックし、ルーティングを作成します。

ルーティングの認可設定

アシスタントの画面にアクセスするための認可設定を行います。

1. IM-BloomMaker ルーティング定義一覧画面を開きます。
   「サイトマップ」→「BloomMaker」→「ルーティング定義一覧」の順にクリックし、「ルーティング定義一覧」画面を表示します。

2. ルーティングの認可設定を行います。
   「*ルーティングの作成*」 で作成したルーティングを選択します。
   認可URIの「認可設定」アイコンをクリックし「認可設定」画面を表示します。

認可設定画面で「認可設定を開始する」をクリックし、リソース「サンプルRAGアシスタント」に対して「認証済みユーザ」を許可に設定します。

> **コラム**
> 本セクションでは一例として「認証済みユーザ」を許可に設定します。運用に合わせて適切な設定を行ってください。

アシスタントの画面を表示

作成したアシスタントの画面を表示します。ブラウザで以下の URL にアクセスします。

`http://<ホスト名>:<ポート番号>/<コンテキストパス>/sample_rag_assistant`

画面が表示され、アシスタントのチャット画面が表示されます。

> **コラム**
> 作成したロジックフローアシスタントの実行にはベクトルデータ構築が必要です。ベクトルデータ構築が完了していない場合は、アシスタントの動作確認ができません。
> 「*ベクトルデータ構築ジョブの作成*」で作成したジョブを実行の後、アシスタントの動作確認を行ってください。
> ジョブネット管理画面で「*ジョブネットの作成*」で作成したジョブネットを選択して「即時実行」を行ってください。