運用ガイド

1. はじめに

本書は、Generative AI FW のシステム構築者、運用管理者のための説明書です。
Generative AI FW の運用方法について解説しています。導入される前に、必ずお読みください。
本書の内容に関しては、将来予告なしに変更することがあります。

1.1. 用語定義について

用語定義については「スタートアップマニュアル（概要編）」をご確認ください。

2. 前提条件

- 本書に記載の手順は全てサーバの管理者ユーザなどの**管理者権限を持つユーザで行う必要**があります。一般ユーザでしかログオンできない環境の場合は以下を実行し、管理者ユーザに昇格させてください。もしくはコマンド実行時に「sudo」を付けてください。

```
sudo -i
```

- 本書に記載の手順はGenerative AI FW のサーバにログオンしている状態である必要があります。

3. LLMの設定変更

本節では、LLMモデルを追加・削除する手順、LLMに関するチャット画面の設定を変更する手順について記載します。本製品では以下のLLMに対応しています。詳細は動作環境をご確認ください。

1. cotomi（cotomi v3、cotomiのファインチューニングモデル）
2. Azure OpenAI Service
3. OpenAIに準拠した外部環境のLLM

上記のうち、2、3についてはSaaSサービスを使用する場合、インターネット通信が必要な閉域環境下では使用できませんのでご注意ください。

詳細の手順は以下の通りです。

> - 以降の手順は全ての画面やAPIが使用されていないことを確認して実施してください。処理がエラーになる可能性があります。
> - cotomiはGenerative AI FWサーバ上で動作するcotomiを想定しています。
> - cotomiは同時に１つのモデルしか動作させることはできません。またcotomiセットアップ後はモデルを変更することができません。

3.1. LLMモデルの追加

3.1.1. Azure OpenAI Serviceの追加

1. /opt/nec/genai/config/genai_info.jsonを開き、llmに関する値を更新します。

```
vi /opt/nec/genai/config/genai_info.json
```

2. llmにAzure OpenAI Serviceで利用するLLMの定義を追加します。その際、既存のLLMの定義を消してしまうと使用できなくなるため消さないようご注意ください。定義の詳細は後述の「genai_info.jsonのLLM定義」をご確認ください。idにはAzure OpenAI Serviceにデプロイしたモデル名、baseにはAPIのエンドポイント、keyにはAPIキーを記載してください。

> モデルごとに回答のランダム性の上限を個別に指定できます。詳細は後述の「回答のランダム性の上限を指定」を参照してください。

例）cotomi v3に加えて、GPT-3.5 Turbo、GPT-4o、GPT-4o miniを追加する場合

```
"llm": [
        {
            "name": "cotomi v3",
            "id": "cotomi-v3.0",
            "description": "高速・高精度を両立するcotomiモデル",
            "enable": true,
            "order": 101,
            "base": "http://cotomi-v3:8000/v1"
        },
```

```
        {
            "name": "GPT-3.5 Turbo",
            "id": "gpt-35-turbo",
            "description": "Azure OpenAI Serviceの提供モデル",
            "enable": true,
            "order": 201,
            "base": "https://xxxx.openai.azure.com/",
            "key": "xxxx",
            "is_aoai": true
        },
        {
            "name": "GPT-4o",
            "id": "gpt-4o",
            "description": "Azure OpenAI Serviceの提供モデル（デプロイの種類：グローバル標準）",
            "enable": true,
            "order": 202,
            "base": "https://xxxx.openai.azure.com/",
            "key": "xxxx",
            "is_aoai": true
        },
        {
            "name": "GPT-4o mini",
            "id": "gpt-4o-mini",
            "description": "Azure OpenAI Serviceの提供モデル（デプロイの種類：グローバル標準）",
            "enable": true,
            "order": 203,
            "base": "https://xxxx.openai.azure.com/",
            "key": "xxxx",
            "is_aoai": true
        }
    ]
```

3. 画面情報を更新します。詳細は後述の「画面情報変更スクリプトの実行」を確認してください。

3.1.2. OpenAIに準拠した外部環境のLLMの追加

1. /opt/nec/genai/config/genai_info.jsonを開き、llmに関する値を更新します。

```
vi /opt/nec/genai/config/genai_info.json
```

2. llmに利用するLLMの定義を追加します。その際、既存のLLMの定義を消してしまうと使用できなくなるため消さないようご注意ください。定義の詳細は後述の「genai_info.jsonのLLM定義」をご確認ください。、baseにはAPIのエンドポイント、keyにはAPIキーを記載してください。

> モデルごとに回答のランダム性の上限を個別に指定できます。詳細は後述の「回答のランダム性の上限を指定」を参照してください。

例）cotomi v3に加えて、XXXXモデルを追加する場合

```
"llm": [
        {
            "name": "cotomi v3",
            "id": "cotomi-v3.0",
            "description": "高速・高精度を両立するcotomiモデル",
            "enable": true,
            "order": 101,
            "base": "http://cotomi-v3:8000/v1"
        },
        {
            "name": "XXXX",
            "id": "XXXX",
            "description": "OpenAIに準拠した外部環境のLLM",
            "enable": true,
            "order": 301,
            "base": "https://xxxx/",
            "key": "xxxx"
        }
    ]
```

> 外部LLMのChat Completions APIのエンドポイントが「https://<外部LLMのドメイン名>/<パス>/chat/completions」の場合、baseには「https://<外部LLMのドメイン名>/<パス>」を指定してください。例えば、 外部LLMのChat Completions APIのエンドポイントが https://example.com/foo/v1/chat/completions の場合の設定値は以下になります。
>
> “base” : “https://example.com/foo/v1/”

3. 画面情報を更新します。詳細は後述の「画面情報変更スクリプトの実行」を確認してください。

3.2. LLMモデルの削除

対応するLLMごとの削除手順は以下の通りです。

3.2.1. Azure OpenAI Serviceの削除

1. /opt/nec/genai/config/genai_info.jsonを開き、llmに関する値を更新します。

```
vi /opt/nec/genai/config/genai_info.json
```

2. 具体的には以下のllmの値から削除したLLMの記載を削除してください（以下はcotomi v3以外を削除した場合です）。なお、cotomi など他のLLMを追加している場合、その記載は残してください。

```
"llm": [
       {
           "name": "cotomi v3",
           "id": "cotomi-v3.0",
           "description": "高速・高精度を両立するcotomiモデル",
           "enable": true,
           "order": 101,
           "base": "http://cotomi-v3:8000/v1"
       }
   ]
```

3. 画面情報を更新します。詳細は後述の「画面情報変更スクリプトの実行」を確認してください。

3.2.2. OpenAI準拠の外部LLMの削除

1. /opt/nec/genai/config/genai_info.jsonを開き、llmに関する値を更新します。

```
vi /opt/nec/genai/config/genai_info.json
```

2. 具体的には以下のllmの値から削除したLLMの記載を削除してください（以下はcotomi v3以外を削除した場合です）。なお、cotomi など他のLLMを追加している場合、その記載は残してください。

```
"llm": [
       {
           "name": "cotomi v3",
           "id": "cotomi-v3.0",
           "description": "高速・高精度を両立するcotomiモデル",
           "enable": true,
           "order": 101,
           "base": "http://cotomi-v3:8000/v1"
       }
   ]
```

3. 画面情報を更新します。詳細は後述の「画面情報変更スクリプトの実行」を確認してください。

3.3. 回答のランダム性の上限を指定

チャット画面では回答のランダム性の上限値は既定では2.0が設定されています。この上限値を変更したい場合は以下の手順に従い変更してください。

1. /opt/nec/genai/config/genai_info.jsonを開き、llmに関する値を更新します。

```
vi /opt/nec/genai/config/genai_info.json
```

2. 上限値を変更したいLLMモデルの設定に以下のようにconfigs項目を追加し、temperature_maxに値を指定してください。上限値はモデルごとに異なる値が指定できるようになっています。temperature_maxを記載しない場合は既定値で動作します。

```
"llm": [
       {
           "name": "cotomi v3",
           "id": "cotomi-v3.0",
           "description": "高速・高精度を両立するcotomiモデル",
           "enable": true,
           "order": 101,
           "base": "http://cotomi-v3:8000/v1",
           "configs": {
             "temperature_max": 1.0
           }
       }
   ]
```

設定名	説明	既定値
temperature_max	チャット画面で指定できる回答のランダム性の上限値。値の範囲は0.0～2.0まで0.1刻みで指定できます。 ただし、1.0を超えた場合はLLMからの回答がおかしくなることがあるので注意してください。	2.0

3. 画面情報を更新します。詳細は後述の「画面情報変更スクリプトの実行」を確認してください。

3.4. 画面情報変更スクリプトの実行

本節では画面情報変更スクリプトの実行方法について説明します。

1. /opt/nec/genai/config/genai_info.jsonを開き、llmに関する値を更新します。定義の詳細は後述の「genai_info.jsonのLLM定義」を確認してください。

```
vi /opt/nec/genai/config/genai_info.json
```

2. 画面情報を更新します。引数にKeycloakの管理者ユーザのパスワードを指定します。

```
cd /opt/nec/genai/operation
bash genai_info_update.sh <Keycloakの管理者ユーザのパスワード>
```

3. 成功すると以下が表示されます。

```
update info succeeded.
```

4. genaiサービスを再起動します。

```
systemctl restart genai
```

5. チャット画面の詳細設定から更新後のモデル一覧が選択できることを確認してください。選択できない場合は、genai_info.jsonが正しく定義されているか確認してください。

3.3.1. genai_info.jsonのLLM定義

genai_info.jsonのLLMのJson値に記載する定義は以下の通りです。なお、どの定義も必須項目のため省略できません。

キー名	説明	値
name	モデルに付与する一意の名前。重複不可。本設定値がチャット画面のモデル名に表示されます。	文字列（半角英数字、記号(-.)のみ）
id	モデルに付与する一意のID(モデル名)。重複不可。本設定値がAPIでのモデル名の指定値になります。また、Azure OpenAI Serviceではデプロイしたモデルの名前と一致させる必要があります。	文字列（半角英数字、記号(-.)のみ）
description	モデルの説明	文字列（日本語可）
enable	有効無効フラグ。必ずtrueに設定してください。	true
order	画面に表示するモデルの順番。一意の値を定義する必要があります。値が小さいほど上位に表示されます。	101以上の値
base	LLMにリクエストするエンドポイントを記載します。	HTTPSのURL
key	APIキー。APIキー認証が必要な場合は指定してください。	文字列
is_aoai	Azure OpenAI ServiceのLLMの判断フラグ。 Azure OpenAI Serviceの場合は必須。それ以外の場合は記載不要です。	true
configs	モデルごとの個別の設定値を記載。以下の設定値が指定できます。	-

	• temperature_max：チャット画面の回答のランダム性の上限値	

4. パスワード運用

4.1. パスワードポリシーの変更

Generative AI FWのパスワードポリシーの変更はKeycloakの管理画面から行います。

詳細な手順はKeycloakのドキュメント(Server Administration Guide)をご確認ください。

Keycloakの管理画面にログインする方法は「正常性確認ガイド」の「Keycloakの確認」をご確認ください。

4.2. 管理者によるログインパスワードの強制初期化

管理者はユーザがパスワードを忘れた時などにログインパスワードを強制的に初期化することができます。パスワードの初期化は管理ポータル画面から行います。詳細は「管理ポータル操作ガイド（基本操作編）」の「ユーザ登録・編集」の説明を確認してください。

なお、パスワードの初期化後は初期化したユーザに初期化パスワードを連絡してください。初期化したパスワードでログインしたのち、ユーザ側で再度パスワードを設定するよう促されます。

4.3. APIキーの確認・更新

Generative AI FW では、Keycloakによるユーザ認証だけでなく、API利用者に向けたAPIキーでの認証をサポートしています。APIキーの確認・更新は、管理ポータル画面から行います。詳細は「管理ポータル操作ガイド(設定画面編)」の「APIキー情報」の説明を確認してください。

4.3.1.APIキーを任意の値に設定

APIキーを画面から変更した場合、システム側でAPIキーが自動で採番されます。利用者側で任意のAPIキーを設定したい場合は以下の手順に従い、コマンドから変更してください。本手順は以下のユースケースで使用することを想定しています。

- 複数のGenerative AI FWサーバを同じAPIキーで統一したい
- 利用者側でAPIキーを決めたい

1. set_apikey.shを実行します。
   実行する際は「primary」「secondary」いずれか更新するキーを指定します。
   本手順ではプライマリーキーを更新する手順を記載しています。セカンダリーキーを更新する場合は適宜読み替えてください。

```
cd /opt/nec/genai/operation
bash set_apikey.sh primary
```

2. 実行すると設定するAPIキーの入力が求められます。任意のAPIキーを指定してください。

APIキーには半角英数字（全て小文字、文字数は32文字）で指定してください。それ以外の動作保証はしておりません。

```
Running in primary mode
Getting APISIX container IP address...
Retrieving configured API key list...
Checking keyids...
Using existing keyid for plain format: 015e2ba31dd44771a970cf13c1101574
Using existing keyid for bearer format: 16f365f3a99b4e7d9c105cf29ffc8d82
Using existing keyid for Bearer format: 27275f8170364dbdbc392598f3a6d05b
Please enter a new API key:
API key: <任意のAPIキーを入力>
```

3. メッセージに「xxxx API key settings completed successfuliy」 と表示されれば設定成功です(xxxxには引数で指定した primary もしくは、secondary のいずれかが入ります)。

```
Setting API keys...
Setting plain format API key...
plain format API key set successfully
Setting bearer format API key...
bearer format API key set successfully
Setting Bearer format API key...
Bearer format API key set successfully
primary API key settings completed successfully
```

4. 管理ポータルにログインし、設定画面でAPIキーが設定した内容に反映されていることを確認ください。

5. アクセス先・HTTPS証明書の更新

以下を行いたい場合に本手順を実施してください。

- サーバのホスト名を変更する場合
- HTTPSの証明書を正規の証明書に変更する場合
- ポートを変更する場合
- 証明書の有効期限切れなどにより証明書を更新したい場合

なお、下記手順後に画面やAPIのアクセス先が変わっているかを「正常性確認ガイド」に従って確認してください。

5.1. アクセス先のドメイン名の変更またはHTTPS証明書の更新をする場合

アクセス先のドメイン名またはHTTPS証明書の更新をする際の手順を説明します。変更するものによって手順変わりますのでご注意ください。

まず本手順の前提として、アクセス先のドメイン名を変更する場合はOS側の設定などは事前に完了させておく必要があります。ここではGenerative AI FW での変更手順のみを記載します。

Generative AI FW は既定では独自の自己証明書を使い、HTTPS通信を実現しています。正規の証明書など自前の証明書を使用したい場合は、以下のフォルダ配下に証明書ファイルを格納してください。

```
/opt/nec/genai/certs
```

1. 以下のコマンドを実行して新しいFQDNのドメイン名を確認します。

```
hostname -f
```

> ここで確認しているFQDNのドメイン名は外部PCからGenerative AI FW のサーバにアクセスする際の以下URLに使用するドメイン名になります。
>
> **https://<ドメイン名>**
>
> サーバ環境によっては実際のドメイン名とは一致しない値が返却されることがあります。その場合、コマンドで確認することは難しいため、適切なドメイン名をDNSなどで別途ご確認のうえ、以降の手順で指定してください。
>
> **なお、ドメイン名はIPアドレスには対応しておりません。**

2. /opt/nec/genai/config/genai.envを開きます。

```
vi /opt/nec/genai/config/genai.env
```

3. ドメイン名を変更する場合、「GENAI_DOMAIN」に変更後のFQDNのドメイン名を記載します。

> - ドメイン名は全て小文字で指定してください。
> - 指定するドメインの各ラベルの先頭に数値を指定しないでください。
>   各ラベルの先頭に数値が指定されている場合、Generative AI FWサービスが正常に動作しない場合があります。以下に問題ない設定例（OK）と問題ある設定例（NG）を示します。
>   例）
>   OK：test.example.com、test.example1.com
>
>   NG：test.1example.com、test.example.1com

```
# 外部公開するドメイン
GENAI_DOMAIN="<ドメイン名>"
```

4. 自前の証明書を使用する場合は証明書に関する以下の値を記載します。自前の証明書を使用しない場合は変更不要です。

```
# 証明書
GENAI_CERTIFICATE_FILE=<証明書のファイルパス>
GENAI_CERTIFICATE_KEY=<証明書のキーファイルパス>
```

具体的には以下のような指定になります。

```
# 証明書
GENAI_CERTIFICATE_FILE=/opt/nec/genai/certs/～.crt
GENAI_CERTIFICATE_KEY=/opt/nec/genai/certs/～.key
```

5. 以下のコマンドを実行します。なお、genai.envに証明書の記述をしなかった場合は自己証明書を新規に作成します。

```
cd /opt/nec/genai/operation
bash access_update.sh <Keycloakの管理者ユーザのパスワード>
```

6. 成功すると以下が表示されます。

```
access update succeeded
```

7. 自己証明書を作成した場合は以下に証明書を格納しています。APIなどを使用する際に利用してください。

```
/opt/nec/genai/certs/certificate.crt
```

8. genaiサービスを再起動します。

```
systemctl restart genai
```

5.2. ポート番号を変更する場合

まず本手順の前提として、ポート変更を変更する場合はOS側（ファイアウォールなどの設定も含む）の設定などは事前に完了させておく必要があります。ここではGenerative AI FW での変更手順のみを記載します。

ポート番号を初期値(443)から変更した場合、アクセス先のURLに「https://<Generative AI FW サーバのドメイン名>:<ポート番号>」のように明示的にポート番号を指定する必要があります。

1. /opt/nec/genai/config/genai.envを開きます。

```
vi /opt/nec/genai/config/genai.env
```

2. 以下の値を変更後のポート番号に修正します。

```
# genaiサービスの外部公開するポート番号
GENAI_PORT=443
```

3. 以下のコマンドを実行します。

```
cd /opt/nec/genai/operation
bash port_update.sh <Keycloakの管理者ユーザのパスワード>
```

4. 成功すると以下が表示されます。

```
port update succeeded.
```

5. genaiサービスを再起動します。

```
systemctl restart genai
```

6. サーバの起動・停止

6.1. サーバ起動

サーバが起動後にgenaiサービスが自動で起動します。

なお、LLMやエンベディングモデルは起動するまでに時間がかかることがあります。

6.2. サーバ停止

特に考慮することはありません。シャットダウン時にサービスは停止するため各コンテナが自動的に停止します。

画面やAPIの操作中にシャットダウンすることは推奨しておりません。

7. サービスの起動・停止

Generative AI FW はgenaiサービスですべてのコンポーネントの起動停止を管理しています。

- 画面やAPIの操作中にサービスの停止、再起動することは推奨しておりません。
- LLMやエンベディングモデルは起動するまでに時間がかかることがあります。

サービスの開始、停止、再起動は以下のように行います。

- サービスの開始

```
systemctl start genai
```

- サービスの停止

```
systemctl stop genai
```

- サービスの再起動

```
systemctl restart genai
```

8. ログ

Generative AI FW では稼働しているコンテナのログをjournalにログを出力後、fluentdを用いてファイルに出力しています。ログは「/opt/nec/genai/app_logs」配下に各モジュールごとに分かれて出力されます。

8.1. ログの採取

ログの採取は管理ポータル画面から行います。詳細は「管理ポータル操作ガイド（設定画面編）」をご確認ください。ログが正常に採取できない場合は後述の「新しいログが採取できない」を確認してください。

8.2. ログの保存設定

ログは初期値では当日のログとは別に過去のログが7日間ログが保存され、期間が過ぎたら古いログから削除されます。ログは日単位でファイルが分割（ローテーション）されます。保存期間の設定は後述の「APP_LOG_ROTATION_LIMIT」で変更できます。

保存期間を延ばす場合、追加でディスク容量が必要になります。ディスク容量の目安は「セットアップガイド」の「動作環境」をご確認ください。ディスク容量は7日単位で増やしてください。

8.3. ログの設定変更

ログの保存設定は/opt/nec/genai/config/genai.env の以下の設定で変更可能です。

以下に記載しない設定を変更することは推奨しておりません。

設定名	説明	既定値
APP_LOG_ROTATION_LIMIT	ログローテートするときの過去のログファイルの保持上限日数。上限を超えた場合は古いファイルから削除されます。値は必ず1以上の値を指定してください。	7

9. バックアップ・リストア

9.1. バックアップ・リストアの基本方針

バックアップをリストアする際には、Generative AI FWの構築を再度新規に実行し、そのうえで、バックアップデータを上書きしてリストアする方法を基本方針とします。

- バックアップ・リストア中はgenaiサービスが停止します。停止せずに行うことには対応しておりません。
- バックアップ・リストア前後でアクセス先・ポート番号は変更しないでください。変更している場合はリストア後に再度変更しなおす必要があります。
- バックアップ・リストア前後でLLMは変更しないでください。変更している場合はリストア後に再度変更しなおす必要があります。
- リストアする際は構築ガイドに記載する手順を事前に完了している必要があります。セットアップガイドの手順実施は必要がありません。
- バックアップ時点で、セットアップガイドの「プロキシ設定・証明書の設定」を実施済みの場合、リストア時に同設定が自動的に再適用されます。ただし、以下の注意事項がありますのでご確認ください。
  - **RHEL リポジトリへのアクセスが必要です。事前にセットアップガイドの「RHEL リポジトリの有効化」の手順を実施してください。**
  - **squidの設定ファイルが上書きされます。本製品以外の用途で Squid を使用している場合は、手順完了後にセットアップガイドの「Squid の設定更新」に従って設定を再適用してください。**
- **バックアップ・リストア前後でGenerative AI FWのバージョンは変更しないでください。変更する場合はアップデート後にバックアップデータを取得しなおしてください。**

- リストアは基本方針の用途だけでなく、新規構築をせず設定をバックアップ時点に戻すロールバック用途にも対応しています。
  手順の詳細は後述の「リストアによるロールバック」を確認してください。

Generative AI FW では以下のコンポーネントの永続データとcotomi LLMを対象にバックアップ・リストアを行います。会話履歴やユーザ情報、管理ポータル画面の情報など本製品を使用するために必要な情報は全てデータベースに保存されているためバックアップ対象になります。

- PostgreSQL
- MongoDB
- Elasticsearch
- etcd
- Generative AI FWの設定情報
- cotomi LLM（cotomiのLLMを使用している場合のみ）
- 監査ログ
- proxy設定

以下の情報・設定はバックアップ・リストアでは復元されません。2については手動にて再度設定をして頂きますよう、お願いいたします。必要があれば個別にバックアップしてください。

1. ログ（アプリケーションログ）
2. Generative AI FW以外のサーバOS側の設定（podman、journalログの設定などを含む）

9.2. バックアップ手順

本手順ではGenerative AI FW のデータのバックアップを行います。

1. バックアップディレクトリを作成します。任意の場所を指定してください。なお、バックアップではバックアップディレクトリが存在しない場合、自動で作成するため本手順は必須ではありません。

```
cd /opt/nec/
mkdir -p backup/2025XXXX
```

2. バックアップ用のスクリプトを実行し、各コンポーネントのデータをバックアップします。
   スクリプト実行時の引数には作成したバックアップディレクトリを指定します。

```
cd /opt/nec/genai/operation
bash genai_backup.sh ../../backup/2025XXXX
```

バックアップフォルダは空フォルダを指定してください。バックアップデータがあるフォルダだと処理がエラーで中断します

3. バックアップに成功すると以下のメッセージが表示されます。バックアップ対象のバックアップが成功していることを確認してください。

```
・・・
PostgreSQL backup succeeded.
・・・
Elasticsearch backup succeeded.
・・・
MongoDB backup succeeded.
・・・
etcd backup succeeded.
・・・
certs folder, config folder, audit_logs folder and genai file backup succeeded.
・・・
all backup succeeded.
```

5. 必須ではありませんが、バックアップデータが正しくできているか確認することを推奨します。バックアップフォルダ配下には以下のフォルダが格納されています。各フォルダに容量があることを確認してください。

- postgres
- elasticsearch
- mongo
- etcd
- genai

9.3. リストア手順

本手順ではGenerative AI FW のデータのリストアを行います。以下の手順は基本方針に示す新規構築直後の環境へのリストアを行う時の手順です。ロールバック用途でのリストアは「リストアによるロールバック」の手順を参照してください。

1. バックアップデータのバージョンとリストア先環境のバージョンが一致しているか確認します。以下のファイルの中身が一致しているか確認してください。

> ⚠ バージョンが一致しない場合はリストアしないでください。動作保証しておりません。

インストールフォルダ配下

```
<インストールフォルダ>/setup/setup_init_version
```

バックアップディレクトリ配下

```
<バックアップディレクトリ>/genai/setup/setup_init_version
```

2. リストア用のスクリプトを実行し、各コンポーネントのデータをリストアします。
   スクリプト実行時の引数にはバックアップ時のディレクトリを指定します。

> ℹ リストア先の監査ログはすべて削除しバックアップデータに上書きします。現在の監査ログを残したい場合は「リストアによるロールバック」の手順を参照してください。

```
cd /opt/nec/genai/operation
bash genai_restore.sh ../../backup/2025XXXX
```

3. リストアに成功すると以下のメッセージが表示されます。バックアップ対象のリストアが成功していることを確認してください。

```
・・・
MongoDB restore succeeded.
・・・
PostgreSQL restore succeeded.
・・・
Elasticsearch restore succeeded.
・・・
etcd restore succeeded.
・・・
certs folder, config folder, audit_logs folder and genai file restore succeeded.
・・・
all restore succeeded.
```

9.4. リストアによるロールバック

本手順ではGenerative AI FW のデータのロールバックを行います。以下の手順はある時点の設定に戻すロールバック用途でリストアを行う時の手順です。基本方針に示す新規構築直後の環境へのリストアは「リストア手順」の手順を参照してください。

1. バックアップデータのバージョンとリストア先環境のバージョンが一致しているか確認します。以下のファイルの中身が一致しているか確認してください。

> ⚠ バージョンが一致しない場合はリストアしないでください。動作保証しておりません。

インストールフォルダ配下

```
<インストールフォルダ>/setup/setup_init_version
```

バックアップディレクトリ配下

```
<バックアップディレクトリ>/genai/setup/setup_init_version
```

2. リストア用のスクリプトを実行し、各コンポーネントのデータをリストアします。
   スクリプト実行時の引数にはバックアップ時のディレクトリと「--skip-audit-log」オプションを指定します。

> ℹ
> - 「--skip-audit-log」オプションを指定した場合、監査ログは以下の方針で処理されます
>   - リストア先の監査ログのまま変更なし（バックアップデータで上書きしない）
> - 「--delete-app-log」オプションを指定すると既存のアプリケーションログを削除します。

```
cd /opt/nec/genai/operation
bash genai_restore.sh ../../backup/2025XXXX --skip-audit-log
```

3. リストアに成功すると以下のメッセージが表示されます。バックアップ対象のリストアが成功していることを確認してください。

```
・・・
MongoDB restore succeeded.
・・・
PostgreSQL restore succeeded.
・・・
Elasticsearch restore succeeded.
・・・
etcd restore succeeded.
・・・
certs folder, config folder, audit_logs folder and genai file restore succeeded.
・・・
```

```
12 all restore succeeded.
```

10. 諸元

Generative AI FW の諸元について記載します。

10.1. LLMの諸元

本製品のLLM に対する諸元は以下の通りです。なお、LLM自体に関する諸元については別途LLM側の仕様書をご確認ください。なお、以下の多重度の諸元は利用する上での推奨値になります。超過しても動作はしますが、応答に時間がかかる可能性があります。

以下の入力文字数にはLLMへの質問文だけでなく、添付ファイルやURLの内容や検索結果の文字列も含まれます

項目	諸元
cotomi v3	• NVIDIA H100 NVLの場合、LLMと対話する推論API（チャット画面を含む）の多重度の最大は20（入力文字数:3万字）～100（入力文字数:2000字）です。入力文字数が多くなればなるほど推論時間が長くなるため、多重度は低くなります。 • NVIDIA L40Sの場合、LLMと対話する推論API（チャット画面を含む）の多重度の最大は10（入力文字数:3万字）～100（入力文字数:2000字）。入力文字数が多くなればなるほど推論時間が長くなるため、多重度は低くなります。 • NVIDIA RTX PRO 6000の場合、LLMと対話する推論API（チャット画面を含む）の多重度の最大は20（入力文字数:3万字）～100（入力文字数:2000字）です。入力文字数が多くなればなるほど推論時間が長くなるため、多重度は低くなります。 • 入力文字数の上限は20万文字です。ただし長文の場合、推論時間が長くなるため応答までに時間がかかります。多重で並列に処理することは推奨しておりません。

10.2. 機能および API の諸元

提供する機能および API に関する諸元は以下の通りです。

機能	諸元
拡張対話（ファイル添付）機能	添付可能なファイル容量の上限は 30MB とします。
インデックス管理機能	1回に登録できる文書の上限はAPIのタイムアウト時間に依存します。1回で登録する文書の合計容量は100MB程度を上限とします。
インデックス管理機能	作成できるインデックス数の上限は 1 万件とします。
インデックス管理機能	1 インデックスあたりの文書登録数の上限は 1 万件とします。
インデックス管理機能	全インデックスでの登録文書総数の上限は50万件とします。上限を超過した場合でも登録は可能ですが動作保証しておりません。
チャット画面	画面のログインの維持時間は最大で 1 日です。

管理ポータル画面	
API機能全般	API実行時のタイムアウトまでの時間は最大１時間です。
管理ポータル画面	登録ユーザ数の上限は1万です。上限を超過した場合でも登録は可能ですが動作保証しておりません。
管理ポータル画面	グループごとのテンプレートの登録上限数は100です。上限を超過した場合、新規の登録ができなくなります。 また、全グループでのテンプレートの合計登録上限数は1000です。上限を超過した場合でも登録は可能ですが動作保証しておりません。
管理ポータル画面	ユーザの一括更新・一括削除、ならびにグループの一括追加・一括削除において、1回の処理で扱える上限は1000です。 上限を超過した場合でも登録は可能ですが動作保証しておりません。
会話履歴	対話時の会話履歴の最大保存期間は90日(1日12時間稼働を想定)です。期間を超えた会話履歴は自動的に削除されます。 最大保存期間の設定を変更する場合は後述の「会話履歴の最大保存期間を変更したい」をご確認ください。
推論を用いない回答根拠の確認機能	回答、参照文書それぞれの全体の文章の長さは5万字、文章内の文章数（文の数）は250までを上限とします。
推論を用いない回答根拠の確認機能	回答、参照文書それぞれの1文の長さは512～1024文字を上限とします（判定上限の文字数は文章によって異なります）
推論を用いない回答根拠の確認機能	1分間に処理できるリクエスト数は約20件が目安になります。ただし、マシンスペックや入力値などにより多少前後する可能性があります。
推論を用いた回答根拠の確認機能	回答文章の長さの上限は1万字、回答文章、参照文章の長さの合計は１0万字を上限とします。
カスタム認証	Active Directory 同期コマンドの最大処理数はユーザ数が１万、グループ数は1000になります。上限を超過した場合でも同期は可能ですが動作保証しておりません。

11. 監査ログ

Generative AI FWでは、監査目的に以下のログをサーバ上に保存しています。

- LLMとの対話履歴のログ（チャット画面、推論系 APIの場合のみ）
- 管理ポータル画面の操作ログ
- 各画面のアクセスログ（ログイン、ログアウト）

> - 対話履歴のログはOpenAI準拠のAPIでは記録されません。
> - ベクトル管理APIなど一部APIでは操作ログは記録されません。

11.1. 監査ログ出力場所

各監査ログの出力場所は下記の表のようになっています。

監査ログ種別	出力先
対話履歴ログ	/opt/nec/genai/audit_logs/api/chat/audit_chat_api.log
管理ポータル画面操作ログ	/opt/nec/genai/audit_logs/operation/admin/audit_ui_management.log
アクセスログ	/opt/nec/genai/audit_logs/keycloak/events.log

11.2. 監査ログフォーマット

各監査ログのフォーマットは下記のような形式になっています。

11.2.1. 対話履歴ログ

対話履歴ログは1行ごとに対話の内容を記録しており、CSV形式で各列ごとに以下の情報を出力します。以下の表は出力順で記載しています。

列名	説明
時間	監査ログの出力時間を表します。タイムゾーンはUTCになります。
リクエストID	一回の会話に一意なIDを表します。UUIDで表記します。
ユーザID	チャット画面の場合はメールアドレスが出力されます。カスタム認証利用時はユーザを一意に識別する値が出力されます(認証方式に依存)。 APIの場合は「x-nec-genai-client-id」ヘッダーで指定した値になります。
メールアドレス	チャット画面、APIの場合どちらもユーザを識別するメールアドレスが出力されます。カスタム認証利用時にユーザIDではユーザが判別できない場合のみ本項目を確認してください。それ以外では参照不要です。
APIタイプ	会話の種別を表します。以下の内容が出力されます。 • 一般対話：v1/chat • テンプレート対話：v1/templatechat • 検索対話：v1/searchchat • 拡張対話：v1/exchat
ステータスコード	HTTPのステータスコードを表します。 成功の場合は200、失敗の場合はそれ以外が出力されます。
質問内容	会話時の質問内容が出力されます。
回答内容	LLMからの回答内容が出力されます。
モデル	使用したLLMのモデル名が出力されます。
検索文書情報	検索対話や拡張対話時に参照した文書やURLの内容が出力されます。それ以外では出力されません。
リクエストパラメータ	リクエストパラメータの詳細が出力されます。

エラー情報	エラーの場合のみ、エラーメッセージなどの詳細情報を出力します。

11.2.2. 管理ポータル画面操作ログ

管理ポータル画面操作ログは1行ごとに管理ポータルの操作の内容を記録しており、CSV形式で各列ごとに以下の情報を出力します。以下の表は出力順で記載しています。

列名	説明
時間	監査ログの出力時間を表します。タイムゾーンはUTCになります。
リクエストID	一回の操作に一意なIDを表します。UUIDで表記します。
ユーザID	メールアドレスが出力されます。カスタム認証利用時はユーザを一意に識別する値が出力されます(連携先に依存)。
ステータスコード	操作実行時の結果です。200~399であれば、成功、400以上であれば画面操作でエラーが発生しています。
リクエストURLのパス	操作した画面を表しています。 /users/ : ユーザー覧画面 /users/user_xxx : ユーザ編集画面（xxxはユーザID) /users/multi : ユーザー括削除 /users/sync : ユーザー括更新 /users/multi/progress : ユーザー括更新・削除状況取得 /groups/ : グループ画面 /groups/group_xxx : グループ編集画面（xxxはグループID) /groups/multi : グループ一括追加・削除 /group/multi/progress : グループ一括操作状況取得 /vectorsearch/indexes/ : インデックス画面 /vectorsearch/indexes/xxx/documents : 文書登録画面(xxxはインデックス名) /templates/ : テンプレート画面 /templates/template_xxx : テンプレート画面(xxxはテンプレートID) /options/ : 設定画面 /options/entailment : 設定画面での回答根拠設定変更 /options/guardrail : 設定画面でのAIガードレール設定変更 /system/logs: ログ取得
リクエストメソッド	各画面で行える操作は以下の通りです。 GET : 一覧を取得

	POST ： 追加 PUT ： 更新・編集 DELETE ： 削除
リクエスト詳細情報	画面操作に対しての詳細情報を表しています。

リクエストURLのパスとリクエストメソッドの組み合わせでそれぞれ何の操作を表しているかは以下の通りです。

	GET	POST	PUT	DELETE
/users/	ユーザー一覧取得	ユーザ追加	なし	なし
/users/user_xxx	ユーザ情報取得	なし	ユーザ情報編集	ユーザ削除
/users/multi	なし	なし	なし	ユーザー括削除
/users/sync	なし	ユーザー括更新	なし	なし
/users/multi/progress	ユーザー括登録・削除状況取得	なし	なし	なし
/groups/	グループ一覧取得	グループ追加	なし	なし
/groups/group_xxx	グループ情報取得	なし	グループ情報編集	グループ削除
/groups/multi	なし	グループ一括追加	なし	グループ一括削除
/groups/multi/progress	グループ一括操作状況取得	なし	なし	なし
/vectorsearch/indexes/	インデックス一覧取得	インデックス追加	なし	なし
/vectorsearch/indexes/xxx	なし	なし	インデックス編集	インデックス削除
/vectorsearch/indexes/xxx/documents	登録文書一覧取得	文書登録	なし	登録文書削除
/templates/	テンプレート一覧取得	テンプレート追加	なし	テンプレート削除
/templates/template_xx	テンプレート情報取得	なし	テンプレート情報編集	なし
/options/	設定取得	なし	なし	なし
/options/entailment	なし	なし	回答根拠設定変更	なし
/options/guardrail	なし	なし	AIガードレール設定変更	なし

/info/apikey/manage	APIキー情報取得	なし	APIキー更新	なし
/system/logs	ログ取得	なし	なし	なし

11.2.3. アクセスログ

下記のようにJSONL形式でログが出力されます。

```
{"timestamp":"2026-01-15T00:03:19.148354903Z","sequence":2824,"loggerClassName":"org.jboss.logging.Logger",
"loggerName":"org.keycloak.events","level":"DEBUG","message":"type=¥"REFRESH_TOKEN¥",
realmId=¥"95c15878-320a-4cd5-b483-d5e068ef38e5¥", realmName=¥"GenaiRealm¥", clientId=¥"admin_client¥",
userId=¥"0eac2d8a-c5cb-49ca-a97b-ebf6c7b2d1a4¥", sessionId=¥"4a85532b-8c67-e510-fb6b-8fa75dbebe16¥",
ipAddress=¥"xxxx¥", token_id=¥"onrtrt:6d581122-c107-ab94-26fe-e6dd44f49041¥",
grant_type=¥"refresh_token¥", refresh_token_type=¥"Refresh¥", access_token_expiration_time=¥"300¥",
updated_refresh_token_id=¥"b66aad21-d344-e1db-0717-e8936aabfa5d¥", scope=¥"openid email profile¥",
age_of_refresh_token=¥"302¥", refresh_token_id=¥"c6fe3a99-885b-f72a-931c-a109fc5b6be0¥",
refresh_token_sub=¥"0eac2d8a-c5cb-49ca-a97b-ebf6c7b2d1a4¥", client_auth_method=¥"client-secret¥"",
"threadName":"executor-thread-6","threadId":16969,"mdc":{},"ndc":"","hostName":"116cc0ff6b9c","processName":
"/usr/lib/jvm/java-21-openjdk-21.0.9.0.10-1.el9.x86_64/bin/java","processId":1}
{"timestamp":"2026-01-15T00:06:08.453552522Z","sequence":2825,"loggerClassName":"org.jboss.logging.Logger",
"loggerName":"org.keycloak.events","level":"DEBUG","message":"type=¥"LOGOUT¥",
realmId=¥"95c15878-320a-4cd5-b483-d5e068ef38e5¥", realmName=¥"GenaiRealm¥", clientId=¥"admin_client¥",
userId=¥"0eac2d8a-c5cb-49ca-a97b-ebf6c7b2d1a4¥", sessionId=¥"4a85532b-8c67-e510-fb6b-8fa75dbebe16¥",
ipAddress=¥"xxxx¥", redirect_uri=¥"https://xxxx/admin/¥",
authSessionParentId=¥"4a85532b-8c67-e510-fb6b-8fa75dbebe16¥", authSessionTabId=¥"0cBlxlbm4EU¥"",
"threadName":"executor-thread-6","threadId":16969,"mdc":{},"ndc":"","hostName":"116cc0ff6b9c","processName"
:"/usr/lib/jvm/java-21-openjdk-21.0.9.0.10-1.el9.x86_64/bin/java","processId":1}
{"timestamp":"2026-01-15T00:06:20.677244951Z","sequence":2826,"loggerClassName":"org.jboss.logging.Logger",
"loggerName":"org.keycloak.events","level":"DEBUG","message":"type=¥"LOGIN¥",
realmId=¥"95c15878-320a-4cd5-b483-d5e068ef38e5¥", realmName=¥"GenaiRealm¥", clientId=¥"admin_client¥",
userId=¥"0eac2d8a-c5cb-49ca-a97b-ebf6c7b2d1a4¥", sessionId=¥"b28ad269-3a24-0292-fdd2-1213d327bf9a¥",
ipAddress=¥"xxxx¥", auth_method=¥"openid-connect¥", auth_type=¥"code¥", response_type=¥"code¥",
redirect_uri=¥"https://xxxx/admin/¥", consent=¥"no_consent_required¥",
code_id=¥"b28ad269-3a24-0292-fdd2-1213d327bf9a¥", username=¥"admin@example.com¥", response_mode=¥"fragment¥",
authSessionParentId=¥"b28ad269-3a24-0292-fdd2-1213d327bf9a¥", authSessionTabId=¥"3w35bhRm3e8¥"",
"threadName":"executor-thread-6","threadId":16969,"mdc":{},"ndc":"","hostName":"116cc0ff6b9c","processName":
"/usr/lib/jvm/java-21-openjdk-21.0.9.0.10-1.el9.x86_64/bin/java","processId":1}
```

ログイン時のログを見やすく成形すると下記の通りです。

```
{
  "timestamp": "2026-01-15T00:06:20.677244951Z",
  "sequence": 2826,
  "loggerClassName": "org.jboss.logging.Logger",
  "loggerName": "org.keycloak.events",
  "level": "DEBUG",
  "message": "
     type=¥"LOGIN¥",
     realmId=¥"95c15878-320a-4cd5-b483-d5e068ef38e5¥",
     realmName=¥"GenaiRealm¥",
     clientId=¥"admin_client¥",
     userId=¥"0eac2d8a-c5cb-49ca-a97b-ebf6c7b2d1a4¥",
     sessionId=¥"b28ad269-3a24-0292-fdd2-1213d327bf9a¥",
     ipAddress=¥"xxxx¥",
     auth_method=¥"openid-connect¥",
     auth_type=¥"code¥",
     response_type=¥"code¥",
     redirect_uri=¥"https://xxxx/admin/¥",
     consent=¥"no_consent_required¥",
     code_id=¥"b28ad269-3a24-0292-fdd2-1213d327bf9a¥",
     username=¥"admin@example.com¥",
     response_mode=¥"fragment¥",
     authSessionParentId=¥"b28ad269-3a24-0292-fdd2-1213d327bf9a¥",
     authSessionTabId=¥"3w35bhRm3e8¥"
  ",
  "threadName": "executor-thread-6",
  "threadId": 16969,
  "mdc": {},
  "ndc": "",
  "hostName": "116cc0ff6b9c",
  "processName": "/usr/lib/jvm/java-21-openjdk-21.0.9.0.10-1.el9.x86_64/bin/java",
  "processId": 1
}
```

下記のようにtypeでログイン、ログアウトを判別可能です。

ℹ ログアウトに関しては、ブラウザを閉じるなど明確にログアウトしない場合は記録されません。

```
type=¥"LOGOUT¥"
```

下記のようにclientIdでチャットUI、管理ポータルか判別可能です。(chat_clientはチャットUI、admin_clientは管理ポータル)

```
clientId=¥"admin_client¥"
```

usernameで誰がログインしたか判別できます。

- デフォルトの認証方式の場合は、メールアドレスで出力されます

```
username=¥"admin@example.com¥"
```

- IdP連携を行っている場合は、連携先IdPのユーザのオブジェクトIDで出力されます

```
username=¥"11111111-2222-3333-4444-555555555555¥"
```

- ユーザID認証の場合は、ユーザIDで出力されます

```
username=¥"user-001¥"
```

なお、旧バージョン(V2.1.2以前)のアクセスログはプレーンテキスト（パターン整形、key="value" をカンマ区切り）で出力されます。現行版の JSONL 形式とは表現のみが異なり、type、clientId、username などのイベント属性の名称と意味は同一です。必要に応じて相互に読み替えて参照してください。

```
2025-05-22 07:53:36,361 40d23a68c34b /usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64/bin/java[1] DEBUG
[org.key.events] (executor-thread-38) type="LOGIN",..., clientId="chat_client",
userId="cfcaf59c-2d83-4bab-8d69-b350e92c366e", sessionId="8e38196f-911a-42e9-aa7d-f86ff1f3fec3",
ipAddress="xxxx",..., redirect_uri="https://xxxx/",..., username="admin@example.com",...
2025-}05-22 } 07:53:36,691 40d23a68c34b /usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64/bin/java[1] DEBUG
[org.key.events] (executor-thread-38) type="CODE_TO_TOKEN",...
2025-05-22 07:53:36,783 40d23a68c34b /usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64/bin/java[1] DEBUG
[org.key.events] (executor-thread-37) type="INTROSPECT_TOKEN",...
2025-05-22 07:53:36,783 40d23a68c34b /usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64/bin/java[1] DEBUG
[org.key.events] (executor-thread-38) type="INTROSPECT_TOKEN",...
2025-05-22 07:53:36,783 40d23a68c34b /usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64/bin/java[1] DEBUG
[org.key.events] (executor-thread-39) type="INTROSPECT_TOKEN",...
2025-05-22 07:53:40,615 40d23a68c34b /usr/lib/jvm/java-21-openjdk-21.0.7.0.6-1.el9.x86_64/bin/java[1] DEBUG
[org.key.events] (executor-thread-39) type="LOGOUT",..., clientId="chat_client",
userId="cfcaf59c-2d83-4bab-8d69-b350e92c366e", sessionId="8e38196f-911a-42e9-aa7d-f86ff1f3fec3",
ipAddress="xxxx", redirect_uri="https://xxxx", ....
```

11.3. 監査ログの保存設定

監査ログは初期値では30日間ログが保存され、期間が過ぎたら古いログから削除されます。保存期間の設定は後述の「AUDIT_LOG_ROTATION_INTERVAL」で変更できます。

保存期間を延ばす場合、追加でディスク容量が必要になります。ディスク容量の目安は「セットアップガイド」の「動作環境」をご確認ください。ディスク容量は30日単位で増やしてください。

対話履歴のログと管理ポータル画面操作ログは日単位でファイルが分割（ローテーション）されます。

アクセスログは日単位に加え、ファイルサイズが上限を超えた場合にもファイルが分割されます。

分割された古いログは以下のファイル名で保存されます。

監査ログ種別	ファイル名
対話履歴ログ	audit_chat_api.log.yyyy-mm-dd.gz
管理ポータル画面操作ログ	audit_ui_management.log.yyyy-mm-dd.gz
アクセスログ	events.log.yyyy-mm-dd（日付での分割の場合） events.log.yyyy-mm-dd.n（ファイルサイズでの分割の場合）

11.4 監査ログの設定変更

監査ログの保存設定は/opt/nec/genai/config/genai.env の以下の設定で変更可能です。

以下に記載しない設定を変更することは推奨しておりません。

設定名	説明	既定値
AUDIT_LOG_ROTATION_LIMIT	ログローテートするときのログファイルの保持上限日数。上限を超えた場合は古いファイルから削除されます。値は必ず1以上の値を指定してください。	30

変更手順は以下の通りです。

1. /opt/nec/genai/config/genai.envを開きます。

```
vi /opt/nec/genai/config/genai.env
```

2. 上記設定を変更します。
3. genaiサービスを再起動します。

```
systemctl restart genai
```

12. トラブルシューティング

12.1. サービスは起動するがAPIや画面の機能が動作しない

各機能が動作しない場合、以下の操作をして改善するかご確認ください。

12.1.1. genaiサービスを再起動する

手順は本ガイドの「サービスの起動・停止」をご確認ください。

12.1.2. 各モジュールの稼働状況を確認する

以下の手順を実施して、各コンテナの稼働状況を確認してください。

1. podman ps でsetupコンテナを除く各コンテナ正常に動作していることを確認してください。

```
podman ps
```

   podman ps では以下に示すコンテナイメージ(IMAGE列、NAMES列)のSTATUSが実行中（Up XX minutesなど）になっていることを確認してください（時間表記は経過時間によって異なります）。IMAGE列の数字については動作環境によって一致しないことがあります。また、※に示すcotomiのIMAGE名は稼働させているモデルによって名前が異なりますのでご注意ください(XXXの部分)。

IMAGE列	NAMES列
docker.io/apache/apisix:3.14.1-debian	internal_apisix
quay.io/coreos/etcd:v3.6.6	etcd
docker.io/library/postgres:16.11	postgres
localhost/genai-ai-orchestrator:1.5.0	genai-ai-orchestrator
localhost/genai-admin-backend:1.5.0	genai-admin-backend
localhost/genai-vector-db:1.5.0	genai-vector-db
docker.io/mongodb/mongodb-community-server:7.0.26-ubuntu2204	mongo-primary
docker.io/mongodb/mongodb-community-server:7.0.26-ubuntu2204	mongo-secondary
docker.io/mongodb/mongodb-community-server:7.0.26-ubuntu2204	mongo-arbiter
docker.io/elastic/elasticsearch:8.19.7	es01
docker.io/library/nginx:1.29.3-alpine-slim	genai-chat-ui

docker.io/library/nginx:1.29.3-alpine-slim	genai-admin-frontend
docker.io/vllm/vllm-openai:v0.11.2	embedding
localhost/vllm-openai:v0.11.2-cotomi-fp8-v3.0.1	cotomi-v3
docker.io/apache/apisix:3.14.1-debian	apisix
docker.io/keycloak/keycloak:26.4.6	keycloak
localhost/genai-explainer-api:1.5.0	genai-explainer-api
localhost/genai-llm-explainer:1.5.0	genai-llm-explainer
localhost/genai-guardrail-server:1.5.0	genai-guardrail-server

12.2. 会話履歴の最大保存期間を変更したい

対話機能の会話履歴には保存期間に上限があります。既定の保存期間は「セットアップガイド」の動作環境、最大の保存期間は本ガイドの「諸元」を確認してください。

削除された会話履歴などMongoDBデータベース内の不要なディスク容量は「会話履歴の最大保存期間を変更したい」に記載する「MONGO_CAPPED_SIZE」の設定だけでは削除されません。削除するための手順については別途お問い合わせください。

保存期間を変えたい場合は以下の手順に従い、変更してください。

1. /opt/nec/genai/config/genai.envを開きます。

```
vi /opt/nec/genai/config/genai.env
```

2. MONGO_CAPPED_SIZEの値を下記の算出式に合わせて必要な容量を記載してください。単位はByteで、既定では66GBに設定されています。

```
MONGO_CAPPED_SIZE=70866960384
```

必要な容量は以下の式で算出してください。上記の値を増やす場合、動作環境のディスク容量もopt配下に追加が必要になります。
会話履歴の1日分(2000文字程度の会話を1日12時間稼働させた場合)の保存に必要な容量は2.2GB(2362232013)になります。

MONGO_CAPPED_SIZE = 2362232013 × <保存期間の日数>

ただし、2000文字以上で頻繁に会話する、拡張対話でファイルやURL、Web検索を使う対話を多く行う場合はより多くの容量が必要になります。その場合は1日分の容量を余裕を持った値に設定してください。例えば一番保存に容量が必要な画像ファイルを入力とした対話を頻繁に行う場合（500KB程度の容量の画像ファイルを多重度50で1日6時間稼働させた場合）の1日の保存に必要な容量は54.2GB(58196806861)になります。

MONGO_CAPPED_SIZEを小さい値に変更した場合、その時点での超過データは次回の会話履歴登録時に古い会話履歴から削除されます。また設定変更だけではディスク容量は削減されません。後述の「MongoDBのディスク使用量削減」をご確認ください。

3. genaiサービスを再開します。

```
systemctl restart genai
```

12.3. MongoDBのメモリ使用量の変更

MongoDBはキャッシュ機能を持っており、キャッシュはメモリを消費します。デフォルトではキャッシュサイズを1GBに制限していますが、キャッシュサイズを変更したい場合は以下を実施してください。

1. genai.envを修正します。

```
vi /opt/nec/genai/config/genai.env
```

2. MONGO_CACHE_SIZEにMongoDBのキャッシュサイズをGB単位で指定します。
   指定可能な値の最小値は0.25，最大値は(サーバのメモリサイズ - 1GB)/2です。

```
MONGO_CACHE_SIZE=1
```

3. サービスを再起動します。

```
systemctl restart genai
```

12.4. MongoDBのディスク使用量削減

会話履歴などの登録データを削除してもMongoDBが利用するディスク容量は削減されません。
MongoDBのディスク使用量が肥大化してきた場合は以下を実行してMongoDBのディスク使用量を削減してください。なお、以下の操作は上から順に実施してください。

- 実行中はLLMとの対話機能や会話履歴に関する操作は行うことができません。処理が完了するまで操作が実行されないため、応答がない場合があります。
- 処理完了するまでに時間がかかる可能性があります。
- 以下の実行コマンドでは改行表示の都合上「¥」が含まれますが、実行時は削除して一行にして実行してください。

```
source /opt/nec/genai/config/genai.env
podman exec mongo-secondary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-db;' --eval ' db.runCommand({compact: "history", force: true});'
podman exec mongo-secondary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-admin-backend;' --eval ' db.runCommand({compact: "vectorsearch", force: true});'
podman exec mongo-secondary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-admin-backend;' --eval ' db.runCommand({compact: "user_info", force: true});'
podman exec mongo-secondary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-admin-backend;' --eval ' db.runCommand({compact: "resources", force: true});'
podman exec mongo-primary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'rs.stepDown();'
podman exec mongo-primary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-db;' --eval ' db.runCommand({compact: "history", force: true});'
podman exec mongo-primary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-admin-backend;' --eval ' db.runCommand({compact: "vectorsearch", force: true});'
podman exec mongo-primary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-admin-backend;' --eval ' db.runCommand({compact: "user_info", force: true});'
podman exec mongo-primary mongosh -u ${MONGO_ADMIN_USERNAME} -p ${MONGO_ADMIN_PASSWORD} ¥
--eval 'use genai-admin-backend;' --eval ' db.runCommand({compact: "resources", force: true});'
```

12.5. PostgreSQLのディスク使用量削減

ユーザの削除をしてもPostgreSQLが利用するディスク容量は削減されません。
PostgreSQLのディスク使用量が肥大化してきた場合、以下を実行してPostgreSQLのディスク使用量を削減してください。なお、以下の操作は上から順に実施してください。

- 実行中はユーザの追加・編集・削除に関する操作は行うことができません。処理が完了するまで操作が実行されないため、応答がない場合があります。
- 処理完了するまでに時間がかかる可能性があります。

```
source /opt/nec/genai/config/genai.env
podman exec -u ${POSTGRES_ADMIN_USER} ${POSTGRES_ADMIN_PASS} bash -c  "psql -d keycloak -c 'VACUUM FULL;'"
```

12.6. Elasticsearchのディスク使用量削減

インデックス削除やインデックスに登録した文書を削除した場合、Elasticsearchが利用するディスク容量は削減されますが、関連データは全て削除されません。

Elasticsearchのディスク使用量が肥大化してきた場合、以下を実行してElasticsearchのディスク使用量を削減してください。なお、以下の操作は上から順に実施してください。

実行中でもインデックスに関する操作は行うことができます。

1. 下記を実行します。

以下の実行コマンドでは改行表示の都合上「¥」が含まれますが、実行時は削除して一行にして実行してください。

```
source /opt/nec/genai/config/genai.env
podman exec -i es01 curl --noproxy localhost -s -k -u elastic:$ELASTIC_SEARCH_PASSWORD ¥
-XGET https://localhost:9200/_cat/indices?v ¥
| awk 'NR>1{print $3}' ¥
| xargs -n1 -I{} podman exec -i es01 curl -s -k -u elastic:$ELASTIC_SEARCH_PASSWORD ¥
-XPOST https://localhost:9200/{}/_forcemerge
```

2. 次に下記を実行します。

```
podman exec -it es01  curl  -s -k -u elastic:$ELASTIC_SEARCH_PASSWORD -XPOST https://localhost:9200/_forcemerge
```

12.7. 新しいログが採取できない

ログを採取した際に新しいログが出力されない場合、ログを出力するサービスが正常に動作していない可能性があります。以下のサービス再起動を実施後、再度ログを採取し新しいログが出力されるか確認してください。

```
systemctl restart fluentd
```

12.8. LLM のトークン数上限を変更する

LLMは128Kトークンまで対応しておりますが、チャット画面などの対話機能で長文のリクエストを行った場合、LLMに負荷がかかるため、他のリクエストの処理時間も遅くなってしまいます。以下の設定を変更することでトークン数の上限を変更することが可能です。設定したトークン数を超過した場合、LLMに推論させることなくエラーになるため、LLMへの負荷を軽減させることができます。

設定名	説明	初期値
MAX_MODEL_LEN	LLMのトークン数の上限。日本語の場合、1トークンあたり2文字程度になります。値の範囲は1K-128Kになります。Kは単位で1024を表します。 本項目のトークン数はリクエスト時の入力だけでなくLLMの回答と合わせたトークン数になります。入力文字数の最大は128Kの場合で最大20万字、32Kで最大5万字程度を目安にしてください。	128K

変更手順は以下の通りです。

1. genai.envを修正します。

```
vi /opt/nec/genai/config/genai.env
```

2. MAX_MODEL_LENに変更したいトークン数を指定します。

```
MAX_MODEL_LEN=128K
```

3. サービスを再起動します。

```
systemctl restart genai
```