Follow these steps to setup the necessary configuration in Okta:

Using Okta Resource Sets

CustomizareNew-French

1. titlejhbnj

6.23 OAS 3.0

Judopay Transaction API

Servers

https://api-sandbox.judopay.com - Sandbox environment

Authorize

Registering Cards

POST /transactions/savecard

Pre-Authorization

POST /transactions/checkcard

POST ~~/transactions/registercard~~

Warning: Deprecated

POST /transactions/preauths

POST /transactions/incrementalAuth

POST /transactions/collections

POST /transactions/voids

Payments

POST /transactions/payments

POST /transactions/refunds

3DSecure

PUT /transactions/{receiptId}/resume3ds

PUT /transactions/{receiptId}/complete3ds

Receipts

GET /transactions/{receiptId}

GET /transactions

GET /transactions/{transactionType}

Payment Sessions

POST /paymentsession Create a generic payment session

PUT **/paymentsession/{reference}/cancel**
Cancel an open payment session

GET **/transactions/{receiptId}/webpayment**
Retrieve details of a completed payment session

GET **/webpayments** Return a list of payment sessions

GET **/webpayments/{reference}**
Retrieve the details of a single payment session

GET **/webpayments/payments/{reference}**
Retrieve the details of a single payment session for a payment

GET **/webpayments/preauths/{reference}**
Retrieve the details of a single payment session for a preauth

POST **/webpayments/checkcard** Create a session for check card

POST **/webpayments/payments** Create a session for a payment

POST **/webpayments/preauths** Create a session for a preauth

Other

GET /info

Schemas

- allowIncrementRequest
- networkTokenRequest
- applePayRequest
- cardAddressCommonAttributes
- cardAddressRequestAttributes
- cardAddressRequest
- cardAddressResponseAttributes
- cardAddressWebPaymentRequest

cardDetailsCv2NotRequiredRequest ›

cardDetailsCv2RequiredRequest ›

cardDetailsStartDateRequest ›

cardDetailsResponse ›

cardDetailsListResponse ›

cardTokenPreAuthPaymentRequest ›

challengeRequestIndicatorRequest ›

checkCardRequestPan ›

collectionRequest ›

complete3ds2Request ›

consumerResponse ›

createPaymentSessionResponse ›

deviceResponse ›

errorResponse

googlePayRequest

incrementAuthRequest

infoResponse

listPaymentSessionsResponse

listTransactionsResponse

Apple Pay

Card token

Google Pay

Card PAN

paymentSessionHistoricResponse

paymentSessionHistoricListResponse

paymentSessionRequest

paymentSessionRequestForPreAuths ›

paymentSessionRequestCommon ›

paymentSessionResponse ›

paymentSessionCancelResponse ›

preAuthPaymentRequestCommon ›

Apple Pay ›

Card token ›

Google Pay ›

Card PAN ›

primaryAccountDetailsRequest ›

recurringPaymentRequest ›

refundRequest ›

registerCardRequest ›

checkCardRequest ›

registerCardRequestPan ›

resume3dsRequest ›

riskParametersResponse ›

networkTokenisationDetailsResponse ›

saveCardRequestCommon ›

saveCardRequestPan ›

scaExemptionRequest ›

threeDSecureChallengeRequest ›

threeDSecureCompletedResponse ›

3DS Challenge Required ›

3DS Device Details Required ›

threeDSecureTwoCheckCardRequest ›

threeDSecureTwoRequest ›

Receipt ›

transactionReceiptHistoricResponse ›

transactionReceiptListResponse ›

voidRequest ›

2. Test

Random text ANdrei

3. Untitled

Notes (important pour Archbee)

- accepte une **requête GET avec un corps JSON**, c'est pourquoi -Méthode GET et -Corps sont tous deux utilisés.
- ConvertTo-Json est nécessaire pour que PowerShell envoie une charge utile JSON appropriée.
- La réponse inclura généralement un **lien de téléchargement** lorsque exportAsLink = true.

Si vous le souhaitez, je peux aussi :

c'est un titre h1

- Convertir cela en **POST** (au cas où Archbee changerait de comportement)
- Ajouter **la gestion des erreurs**
- Enregistrer le lien d'exportation directement dans un fichier ou déclencher un téléchargement

```
$headers = @{
   "Accept"        = "application/json"
   "Content-Type"  = "application/json"
   "Authorization" = "Bearer
SFFfdkhvMEJSLVN2RVRvYXowMkROfkp2NVZwUW5iWXB0VF9XUkxscWZrLQ=="
}

$body = @{
   teamId = "511c8QBH-VHiwWnyzIwUb"
   exportThisSpaceOnly = $true
   exportAsLink = $true
} | ConvertTo-Json

$response = Invoke-RestMethod `
   -Uri $uri `
   -Method GET `
```

```
-Headers $headers `
-Body $body
```

4. c'est un titre h1

asdasdasdasd

asdasdasdasdasasdsadas

asdasdasdasdasdas

asta sont du contenu à l'intérieur si e prins

asdasdasdas

dasdasdas

dasdasdsadasd

comme

Titre 1

asdasdasd

asdasdasdasd

sadas

Titre 2

asdasdasd

asdas

dasdsa

deuxième

asta e deschis

Ceci est l'en-tête	Ceci est l'en-tête 2	Ceci est l'en-tête 3
asdasdas	asddasd	sadasd
asdasdas	asdasdas	asdas
asdasd	asdasd	asdas

asdasdasdasd

asdasdasdasdasasdsadas

asdasdasdasdasdas

le contenu à l'intérieur est pris

deuxième

c'est ouvert

5. Baguette parce que car cela ouais

Frenchtech autre comme même vin baguette parce que car cela ouais. Guillotine car pour ouais dernier parce que. Avec camembert épicé.

Le client est très important merci, le client sera suivi par le client. Énée n'a pas de justice, pas de résultat, pas de ligula, et la vallée veut la sauce. Morbi mais qui veut vendre une couche de contenu triste d'internet. Être ivre maintenant, mais ne pas être ivre maintenant, mon urne est d'une grande beauté, mais elle n'est pas aussi bien faite que dans un livre. Mécène dans la vallée de l'orc, dans l'élément même. Certaines des exigences faciles du budget, qu'il soit beaucoup de temps pour dignissim et. Je ne m'en fais pas chez moi, ça va être moche dans le vestibule. Mais aussi des protéines de Pour avant la fin de la semaine, qui connaît le poison, le résultat.

le client sera suivi par le client. Énée n'a pas de justice, pas de résultat, pas de ligula, et la vallée veut la sauce. M dasdasdas Archbee Être ivre maintenant, mais ne pas être ivre maintenant, mon urne est d'une grande beauté, mais elle n'est pas aussi bien faite que dans un livre. Mécène dans la vallée de l'orc, dans l'élément même. Certaines des exigences faciles du budget, qu'il soit beaucoup de temps pour dignissim variabila

asdasdasd asdasdsadsa

5.1. Test doc

Nabucodonosor

asdasdas

1.0.7 OAS 2.0

Swagger Petstore

This is a sample server Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger. For this sample, you can use the api key `special-key` to test the authorization filters.

Terms of service

Contact the developer

Apache 2.0

Find out more about Swagger

Schemes

HTTPS

Authorize

pet Everything about your Pets — Find out more

POST `/pet/{petId}/uploadImage` uploads an image

POST `/pet` Add a new pet to the store

PUT /pet Update an existing pet

GET /pet/findByStatus Finds Pets by status

GET ~~/pet/findByTags~~ Finds Pets by tags

Warning: Deprecated

GET /pet/{petId} Find pet by ID

POST /pet/{petId} Updates a pet in the store with form data

DELETE /pet/{petId} Deletes a pet

store Access to Petstore orders

GET /store/inventory Returns pet inventories by status

POST /store/order Place an order for a pet

GET /store/order/{orderId} Find purchase order by ID

DELETE /store/order/{orderId} Delete purchase order by ID

user Operations about user

Find out more about our store

POST /user/createWithList Creates list of users with given input array

GET /user/{username} Get user by user name

PUT /user/{username} Updated user

DELETE /user/{username} Delete user

GET /user/login Logs user into the system

GET /user/logout Logs out current logged in user session

POST /user/createWithArray Creates list of users with given input array

POST /user Create user

Models

ApiResponse

Category

Pet

Tag

Order

User

Bucuresti

Lorem Ipsum: est simplement un texte factice de l'industrie de l'impression et de la composition. Lorem Ipsum est le texte factice standard de l'industrie depuis les années 1500, lorsque un imprimeur inconnu a pris une galée de caractères et l'a mélangée pour créer un livre de spécimens typographiques. Il a survécu non seulement cinq siècles, mais aussi le saut vers la composition électronique, restant essentiellement inchangé. Il a été

popularisé dans les années 1960 avec la sortie de feuilles Letraset contenant des passages de Lorem Ipsum, et plus récemment avec des logiciels de publication assistée par ordinateur comme Aldus PageMaker incluant des versions de Lorem Ipsum.

6. Indenting

6.1. Une autre page

asdasdsadasdas

6.1.1. Filtrage avancé

Dans la section "facettage", nous mettons en évidence comment on peut obtenir les filtres disponibles pour une requête de recherche et appliquer des filtres pour affiner les résultats de recherche. Par exemple, après avoir recherché "chaussures", un acheteur peut vouloir affiner les résultats en appliquant un filtre sur l'attribut couleur et en choisissant une valeur telle que rouge. Ce faisant, seules les chaussures "rouges" apparaîtront dans les résultats. Si à ce stade, l'acheteur choisit une autre couleur, par exemple, bleu, Klevu affichera maintenant à la fois les chaussures "rouges" et les "bleues". Ainsi, il appliquera l'opération OU entre les valeurs du même attribut. Si à ce stade, l'acheteur choisit "coton" comme matériau, le système affichera à la fois les chaussures rouges et bleues, mais il s'assurera également que tous les produits ont "coton" comme matériau. En d'autres termes, il applique l'opération ET entre les différents types d'attributs sélectionnés.

Cependant, il arrive un moment où l'on peut vouloir effectuer des opérations de filtrage complexes, ou en d'autres termes, des opérations de filtrage avancées. Par exemple, vous pouvez vouloir afficher un groupe de chaussures où "la couleur est rouge", et "la marque n'est PAS Adidas". Vous pouvez également vouloir ajouter un autre groupe de chaussures à cet ensemble de résultats. Par exemple, où "la couleur est bleue ou noire", "la marque est Nike", et "la fourchette de prix est entre 100 et 400". Comme vous pouvez le voir, contrairement au comportement du "facettage", dans notre exemple ici, nous souhaitons avoir un ou plusieurs groupes de produits correspondant à différents groupes de conditions pour l'ensemble final des produits à retourner.

C'est avec cette intention de permettre des opérations de filtrage avancées sur l'ensemble des résultats que nous avons introduit une nouvelle fonctionnalité appelée "filtrage avancé" dans notre API.

Avant l'introduction du filtrage avancé, si un utilisateur avait demandé que des filtres soient retournés ou que certains soient appliqués, Klevu cherchait à trouver un ensemble de produits pertinents correspondant à la requête de l'utilisateur. Il incluait/excluait ensuite tous les produits demandés explicitement dans cet ensemble de résultats. C'est après cette étape qu'il calcule tous les filtres possibles à retourner et applique également les filtres sélectionnés pour affiner les résultats de recherche. Ainsi, si un produit ajouté explicitement ne correspondait pas au filtre appliqué, il serait exclu de l'ensemble final des résultats.

Avec l'ajout du filtrage avancé, Klevu identifie maintenant un ensemble de produits pertinents correspondant à la requête. Il applique ensuite le filtrage avancé à cela. C'est après cette étape qu'il inclut/exclut tous les produits demandés explicitement, calcule les filtres basés sur cet ensemble de résultats, et enfin applique les filtres sélectionnés pour affiner les résultats de recherche. De cette manière, s'il y a un filtrage à appliquer avant d'inclure/exclure des produits explicitement, le filtrage avancé doit être utilisé.

> Veuillez noter que le filtrage avancé peut également être utilisé avec les API de marchandisage par catégorie et de recommandation.

Le **groupCondition** l'objet peut être utilisé pour appliquer les conditions de filtrage avancé afin que vos clients puissent affiner leurs résultats en fonction des attributs pertinents.

Apply Advanced Filtering

Example is as below:

POST https://{{hostname}}/cs/v2/search

Request

BODY PARAMETERS

groupOperator `String` `optional`

Defining the condition of filtering. Here are the options available.
ALL_OF:This is used to filter data using AND query i.e. the search results are matching with all the attributes.

ANY_OF:This is used to filter data using OR query i.e. the search results are matching with at least one attributes.

NONE_OF: This is used as negation AND query i.e. when search results are matching with none of the attribute values.

key `String` `optional`

The ID of the attribute to filter by, eg. color

valueOperator `String` `optional`

Defining an attribute's operation, Here are the options available.

INCLUDE : This is used to include specific attribute for the filter purpose.

EXCLUDE : This is used to exclude specific attribute for the filter purpose.

EXISTS: This is used to find if the specific indexed attribute value exists with product.

NOT_EXISTS : This is used to find if the specific indexed attribute value does not exist at all.

singleSelect `Boolean` `optional`

The behaviour when specifying multiple filters or values.

If it is set to "true", the products returned must have both the values.
If it is set to "false", the products returned must have at least one of the values.

excludeValuesInResult `Boolean` `optional`

For numeric filter like price, can be used to exclude data of the starting or ending values. For example if the range is defined as "1200 - 1232" in values, and excludeValuesInResult is set to true, Klevu will use range as “1201 - 1231”.

This can be used to achieve greater than or less than condition. For example,if you want to get the products greater than price 200, the values field will be “200 - *“ and the excludeValuesInResult will be set to true.

values `Array` `optional`

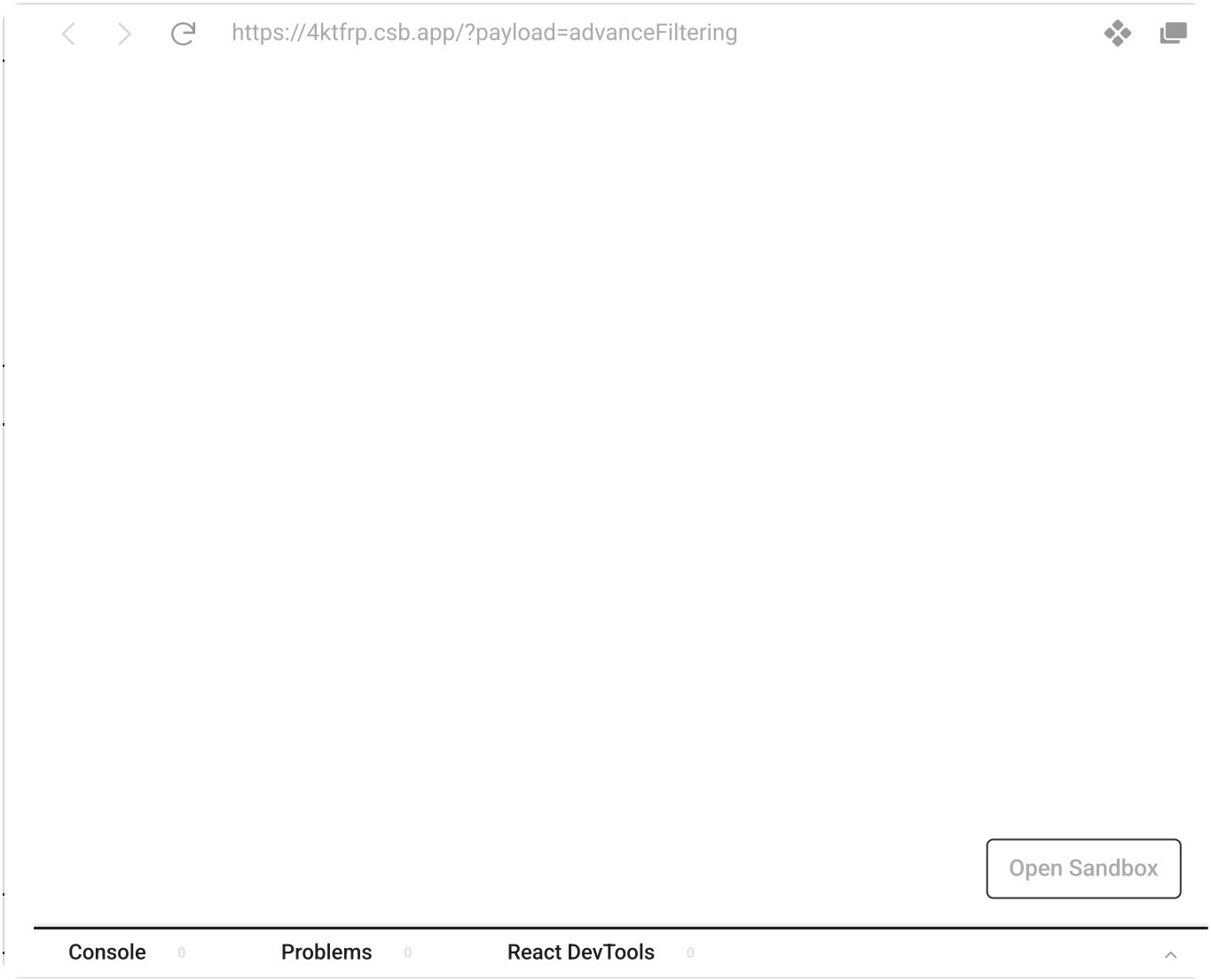
An array of values to filter by, eg. Red, Blue. When using range filters, specify the first value as the minimum value and the second as the max value. For example to retrieve records with prices between 60 to 80, use: "key": "klevu_price", "values": [60, 80].
To retrieve steps of prices, for example those with values between 0-50 or 150-200, use: "key": "klevu_price", "values": ["0-50", "150-200"].

To retrieve exact values, for example records with exact values 100 or 200, use: "key": "klevu_price", "values": ["100-100", "200-200"].
By default all attributes submitted to Klevu are indexed as STRING attributes, which means they cannot be used as range filters. The product sale price field is the only exception to this rule, which is filtered using the key klevu_price. If you have explicitly requested and Klevu has approved that certain attributes be indexed as numerical attributes, you can also retrieve those as range filters.

```
{
   "context":{
      "apiKeys":[
         "klevu-156925593843210765"
      ]
   },
   "recordQueries":[
      {
         "id":"productSearch",
         "typeOfRequest":"SEARCH",
         "settings":{
            "typeOfRecords":[
               "KLEVU_PRODUCT"
            ],
            "groupCondition":{
               "groupOperator":"ANY_OF"
```

```
{
    "meta": {
        "qTime": 9,
        "responseCode": 200
    },
    "queryResults": [
        {
            "id": "productSearch",
            "meta": {
                "qTime": 7,
                "noOfResults": 2,
                "totalResultsFound": 6,
                "typeOfSearch": "WILDCARD_AND",
                "offset": 0,
                "debuggingInformation": {},
                "notificationCode": 1,
                "searchedTerm": "tee",
                "apiKey": "klevu-156925593843210765",
                "isPersonalised": false
            },
            "records": [
                {
                    "discount": "",
                    "hideGroupPrices": "",
                    "type": "Tees",
                    "itemGroupId": "4384027344958",
                    "freeShipping": "",
                    "storeBaseCurrency": "GBP",
                    "price": "24.00",
                    "toPrice": "",
                    "imageUrl": "https://cdn.shopify.com/s/files/1/0116/9457/1582/pro
                    "currency": "GBP",
                    "inStock": "yes",
                    "id": "31366447038526",
                    "imageHover": "",
                    "sku": "WS05-XS-Yellow",
                    "brand": "KKE",
                    "basePrice": "24.0",
                    "startPrice": "",
                    "image": "https://cdn.shopify.com/s/files/1/0116/9457/1582/produc
                    "deliveryInfo": "",
                    "hideAddToCart": "",
                    "salePrice": "24.0",
```

```
"swatchesInfo": "",
"weight": "",
"klevu_category": "KLEVU_PRODUCT;Products;;Tees  @ku@kuCategory@k
"totalVariants": 0,
"groupPrices": "",
"url": "https://jsv2-shopify-demo.ksearchmisc.com/products/desire
"tags": "comfort, flattering fit, micro-sleeve, summer, v-neck",
```

6.2. Nouvelle page

asdasdasdas

7. page de test

Obtenir des gâteaux

Obtenir un gâteau par son ID

GET https://api.cakes.com

Request

BODY PARAMETERS

id Array required

ID du gâteau à obtenir

Nom du paramètre ▶ Object optional

```
var myHeaders = new Headers();
myHeaders.append("Accept", "application/json");
myHeaders.append("Content-Type", "application/json");

var raw = JSON.stringify({
   "id": "String"
});

var requestOptions = {
   method: 'GET',
   headers: myHeaders,
   body: raw,
   redirect: 'follow'
};

fetch("https://api.cakes.com", requestOptions)
   .then(response => response.text())
   .then(result => console.log(result))
   .catch(error => console.log('error', error));
```

```
{
  "name": "Nom du gâteau",
}
```

8. Reminder

8.1. Langage Toxique

Le garde-fou de langage toxique détecte le contenu nuisible, y compris les discours de haine, les menaces, les insultes et d'autres communications qui pourraient nuire à votre communauté ou à votre marque. Contrairement aux simples filtres de mots-clés, ce garde-fou utilise l'IA pour comprendre le contexte, le ton et l'intention.

knmknknkjn

Paste an image link or upload your image

Image URL

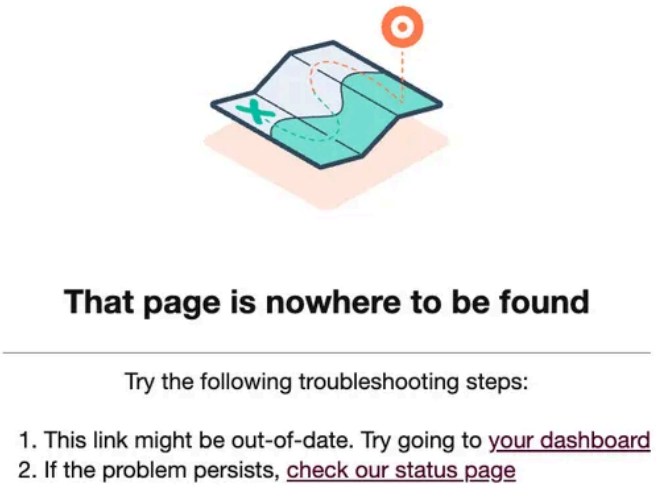
Add Image

Quand utiliser ce garde-fou

Paste an image link or upload your image

Image URL

Add Image



L'avantage clé par rapport au simple filtrage par mots-clés est que ce garde-fou comprend les nuances. Quelqu'un peut être hostile sans utiliser de grossièretés, et il peut utiliser un langage fort sans être hostile. Le garde-fou analyse l'intention et le ton, pas seulement le choix des mots.

Comprendre les niveaux de sensibilité

Le paramètre de sensibilité contrôle la rigueur de la barrière de protection. Pensez-y comme à l'ajustement du seuil de ce qui constitue une violation. C'est l'un des choix de configuration les plus importants que vous ferez car il change fondamentalement le contenu qui passe ou échoue.

Une sensibilité faible ne signale que les violations graves comme les menaces explicites et les discours de haine extrêmes. Lorsque vous réglez la sensibilité sur faible, vous dites que vous vous attendez à des opinions fortes et à des désaccords robustes, et que vous ne voulez bloquer que le contenu qui franchit une ligne claire vers un territoire menaçant ou haineux. Par exemple, "Je suis fortement en désaccord avec cette approche et je pense qu'elle est mal orientée" passerait à faible sensibilité, tout comme "C'est une terrible idée." Seul un contenu comme "Je te trouverai et je te ferai du mal" ou un discours de haine explicite échouerait.

La sensibilité faible fonctionne bien pour les forums publics où le débat est attendu, les environnements de retour professionnel où la franchise est valorisée, et les communautés techniques où les gens discutent de sujets controversés. Le compromis est que certains contenus qui mettent les gens mal à l'aise pourraient encore passer.

La sensibilité moyenne est le paramètre par défaut et représente une approche équilibrée. À ce niveau, la barrière de protection signale les violations claires, y compris les insultes et le langage hostile, tout en permettant encore le désaccord professionnel et la critique constructive. Un message comme "Je ne suis pas d'accord avec votre raisonnement" passerait, mais "Tu es un idiot" ou "Des gens comme vous sont le problème" échoueraient.

La sensibilité moyenne fonctionne bien pour la plupart des applications, y compris les systèmes de service client, les communications professionnelles, les outils collaboratifs et les plateformes sociales. Elle trouve un équilibre entre la possibilité d'un discours significatif et le maintien d'un environnement respectueux.

La haute sensibilité crée l'environnement le plus strict en signalant tout contenu potentiellement toxique, y compris la légèreté de la grossièreté et le langage désinvolte. À ce niveau, même un contenu comme "Peu importe, mec" ou "C'est plutôt stupide" échouerait. Seul un contenu respectueux et neutre passe à haute sensibilité.

Une haute sensibilité est appropriée pour les applications destinées aux enfants où vous avez besoin d'une protection maximale, les plateformes éducatives où vous souhaitez modéliser une communication respectueuse, les espaces sûrs et les communautés de soutien où les gens ont besoin de se sentir en sécurité, et les contextes critiques en matière de conformité où tout problème potentiel doit être détecté.

Options de configuration

Le garde-fou contre le langage toxique accepte plusieurs options de configuration qui vous permettent d'ajuster son comportement pour votre cas d'utilisation spécifique.

```
// Available options:
// sensitivity: "low", "medium", or "high" (default: "medium")
// model: model identifier (default: Claude 3.5 Haiku)
// temperature: 0-1, lower values = more consistent (default: 0.1)
// maxTokens: response length limit (default: 200)

await abv.guardrails.toxicLanguage.validate(text, {
  sensitivity: "medium",
  model: "model-name",
  temperature: 0.1,
  maxTokens: 200,
});
```

```
# Available options:
# sensitivity: "low", "medium", or "high" (default: "medium")
# model: model identifier (default: Claude 3.5 Haiku)
# temperature: 0-1, lower values = more consistent (default: 0.1)
# maxTokens: response length limit (default: 200)

abv.guardrails.toxic_language.validate(text, {
    "sensitivity": "medium",
    "model": "model-name",
    "temperature": 0.1,
    "maxTokens": 200
})
```

L'option de sensibilité est la plus importante et vous l'utiliserez fréquemment. Les options de modèle, de température et de maxTokens sont des paramètres avancés que vous n'aurez généralement pas besoin de modifier. Le modèle par défaut est optimisé pour les tâches de garde-fou et offre le meilleur équilibre entre vitesse, précision et coût. La température par défaut de 0.1 garantit des résultats cohérents. Le maxTokens par défaut de 200 est suffisant pour le champ d'explication.

Exemples du monde réel

Examinons des exemples concrets de la manière dont différents niveaux de sensibilité traitent divers types de contenu. Comprendre ces schémas vous aidera à choisir la bonne sensibilité pour votre application.

Considérez un message comme "Je ne suis pas d'accord avec votre approche de ce problème." Il s'agit d'un désaccord professionnel qui passe à tous les niveaux de sensibilité. Le langage est neutre et respectueux malgré l'expression du désaccord.

Maintenant, considérez "C'est une terrible idée et montre un mauvais jugement." Cela passe à faible et moyen niveau de sensibilité car, bien que critique, cela se concentre sur l'idée plutôt que d'attaquer la personne. Cependant, cela pourrait échouer à un niveau de sensibilité élevé car "terrible" et "mauvais jugement" pourraient être considérés comme dédaigneux.

Un message comme "Tu ne sais pas de quoi tu parles" échoue à un niveau de sensibilité moyen et élevé car il attaque directement la compétence de la personne. Cela pourrait passer à faible sensibilité puisqu'il ne contient pas de menaces explicites ou de discours de haine, bien que ce soit limite.

Un contenu comme "Tu es un idiot" ou "Les gens comme toi sont le problème" échoue à tous les niveaux de sensibilité. Ce sont des attaques personnelles claires sans valeur constructive.

Enfin, des menaces explicites comme "Je te trouverai et je te ferai du mal" échouent à tous les niveaux de sensibilité avec une confiance maximale. C'est un contenu toxique sans ambiguïté.

Modèles de mise en œuvre

Voici comment vous utiliseriez typiquement la détection de langage toxique dans différentes parties de votre application.

Pour la validation des entrées, vous vérifiez les messages des utilisateurs avant de les envoyer à votre IA ou de les afficher à d'autres utilisateurs :

```typescript
async function validateUserMessage(message: string): Promise<boolean> {
  const result = await abv.guardrails.toxicLanguage.validate(
    message,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    return true;
  }

  // Log the reason for monitoring, but don't expose it to the user
  console.log("Blocked message:", result.reason);
  return false;
}

// Usage in your message handler
if (await validateUserMessage(userInput)) {
  await processMessage(userInput);
} else {
  return { error: "Your message violates our community guidelines." };
}
```

```python
async def validate_user_message(message: str) -> bool:
    result = await abv.guardrails.toxic_language.validate_async(
        message,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        return True

    # Log the reason for monitoring, but don't expose it to the user
    print(f"Blocked message: {result['reason']}")
    return False

# Usage in your message handler
if await validate_user_message(user_input):
    await process_message(user_input)
else:
    return {"error": "Your message violates our community guidelines."}
```

Pour la validation des sorties, vous vérifiez les réponses générées par l'IA avant de les montrer aux utilisateurs :

```typescript
async function generateSafeResponse(prompt: string): Promise<string> {
  // Generate initial response
  let response = await callAI(prompt);

  // Validate the response
  const validation = await abv.guardrails.toxicLanguage.validate(
    response,
    { sensitivity: "high" }
  );

  // If toxic, regenerate with explicit safety instruction
  if (validation.status === "fail") {
    response = await callAI(
      prompt + "\n\nIMPORTANT: Respond in a professional, respectful tone."
    );
  }

  return response;
}
```

```python
async def generate_safe_response(prompt: str) -> str:
    # Generate initial response
    response = await call_ai(prompt)

    # Validate the response
    validation = await abv.guardrails.toxic_language.validate_async(
        response,
        {"sensitivity": "high"}
    )

    # If toxic, regenerate with explicit safety instruction
    if validation["status"] == "fail":
        response = await call_ai(
            f"{prompt}\n\nIMPORTANT: Respond in a professional, respectful tone."
        )

    return response
```

Pour gérer les cas ambigus, vous pourriez mettre en place une file d'attente de révision pour les résultats incertains :

```typescript
async function handleUserContent(content: string) {
  const result = await abv.guardrails.toxicLanguage.validate(
    content,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    // Content is clearly acceptable
    await publishContent(content);
  } else if (result.status === "fail" && result.confidence > 0.8) {
    // High-confidence violation, auto-reject
    await rejectContent(content, "Community guidelines violation");
  } else {
    // Low confidence or unsure - flag for human review
    await flagForModeration(content, result);
  }
}
```

```python
async def handle_user_content(content: str):
    result = await abv.guardrails.toxic_language.validate_async(
        content,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        # Content is clearly acceptable
        await publish_content(content)
    elif result["status"] == "fail" and result["confidence"] > 0.8:
        # High-confidence violation, auto-reject
        await reject_content(content, "Community guidelines violation")
    else:
        # Low confidence or unsure - flag for human review
        await flag_for_moderation(content, result)
```

Optimisation des performances

Puisque la détection de langage toxique utilise l'IA, cela prend une à trois secondes par vérification et consomme des jetons. Vous pouvez optimiser les performances en effectuant d'abord une vérification rapide basée sur des règles pour attraper les violations évidentes avant de faire l'appel coûteux à l'IA.

```
async function efficientToxicCheck(text: string): Promise<boolean> {
  // Quick check for explicitly forbidden terms (under 10ms, free)
  const quickCheck = await abv.guardrails.containsString.validate(
    text,
    {
      strings: ["explicit-slur", "forbidden-term"],
      mode: "none",
    }
  );

  // If quick check fails, no need for expensive AI check
  if (quickCheck.status === "fail") {
    return false;
  }

  // Only run AI check if quick check passed
  const deepCheck = await abv.guardrails.toxicLanguage.validate(text);
  return deepCheck.status === "pass";
}
```

```python
async def efficient_toxic_check(text: str) -> bool:
    # Quick check for explicitly forbidden terms (under 10ms, free)
    quick_check = await abv.guardrails.contains_string.validate_async(
        text,
        {
            "strings": ["explicit-slur", "forbidden-term"],
            "mode": "none"
        }
    )

    # If quick check fails, no need for expensive AI check
    if quick_check["status"] == "fail":
        return False

    # Only run AI check if quick check passed
    deep_check = await abv.guardrails.toxic_language.validate_async(text)
    return deep_check["status"] == "pass"
```

Meilleures pratiques de sécurité

Ne jamais exposer le champ de raison aux utilisateurs finaux. La raison explique pourquoi le contenu a échoué à la validation, et exposer cette information aide les acteurs malveillants à apprendre comment contourner vos garde-fous. Au lieu de cela, utilisez des messages d'erreur génériques tout en enregistrant la raison détaillée en interne pour le suivi et l'amélioration.

```javascript
// Bad - exposes validation logic
if (result.status === "fail") {
  return { error: result.reason };  // Don't do this!
}

// Good - generic message, internal logging
if (result.status === "fail") {
  logger.info("Blocked toxic content", { reason: result.reason });
  return { error: "Your message violates our community guidelines." };
}
```

```python
# Bad - exposes validation logic
if result["status"] == "fail":
    return {"error": result["reason"]}  # Don't do this!

# Good - generic message, internal logging
if result["status"] == "fail":
    logger.info(f"Blocked toxic content: {result['reason']}")
    return {"error": "Your message violates our community guidelines."}
```

Choisir la bonne sensibilité

Voici un cadre de décision pour choisir la sensibilité en fonction de votre type d'application. Si vous construisez pour des enfants ou des populations vulnérables, utilisez toujours une haute sensibilité. Le potentiel de dommages causés par l'autorisation de contenu toxique l'emporte de loin sur le coût des faux positifs.

Si vous construisez une application destinée aux clients comme un service client, des réseaux sociaux ou des outils collaboratifs, une sensibilité moyenne est généralement appropriée. Elle détecte les violations claires tout en permettant des désaccords professionnels.

Si vous construisez pour des publics professionnels ou techniques où un débat robuste est attendu, envisagez une faible sensibilité. Les forums techniques, les systèmes de révision de code et les outils de retour professionnel bénéficient de la possibilité d'exprimer des opinions fortes.

Vous pouvez également ajuster la sensibilité en fonction du contexte de l'utilisateur. Les utilisateurs authentifiés avec un bon historique pourraient avoir une sensibilité plus faible tandis que les utilisateurs anonymes obtiennent une sensibilité plus élevée. Les utilisateurs qui s'identifient comme mineurs obtiennent automatiquement une haute sensibilité, quelle que soit la configuration par défaut.

Prochaines étapes

La barrière de langage toxique est souvent utilisée avec d'autres barrières pour une validation complète du contenu. Envisagez de la combiner avec la détection de langage biaisé pour une solution de sécurité du contenu plus complète. Vous voudrez peut-être

également utiliser une chaîne de contenu pour attraper rapidement les termes interdits explicites avant d'exécuter le contrôle de langage toxique plus coûteux.

Pour des conseils d'implémentation plus détaillés, consultez la documentation des meilleures pratiques qui couvre les stratégies d'optimisation, la gestion des erreurs et les approches de surveillance.

```
asdasdasdasdasdasdasdasdasdsa
```

8.2. ASDASDSDA

ASDASDASDAS

ASDASDASDAS

> ASDSADASDSAD

8.3. Test de numérotation sur les images problème (cloné)

Le paquet des Détails Bancaires de l'Entité (EBD) s'intègre avec l'enregistrement standard des Fournisseurs de NetSuite pour stocker en toute sécurité toutes les informations nécessaires à la soumission des paiements. Une fois installé, les données des fournisseurs et des employés importées sont accessibles via **Configuration → Détails Bancaires de l'Entité → Aperçu**. Découvrez comment importer, exporter et mettre à jour l'EBD pour les fournisseurs et les employés dans ce guide.

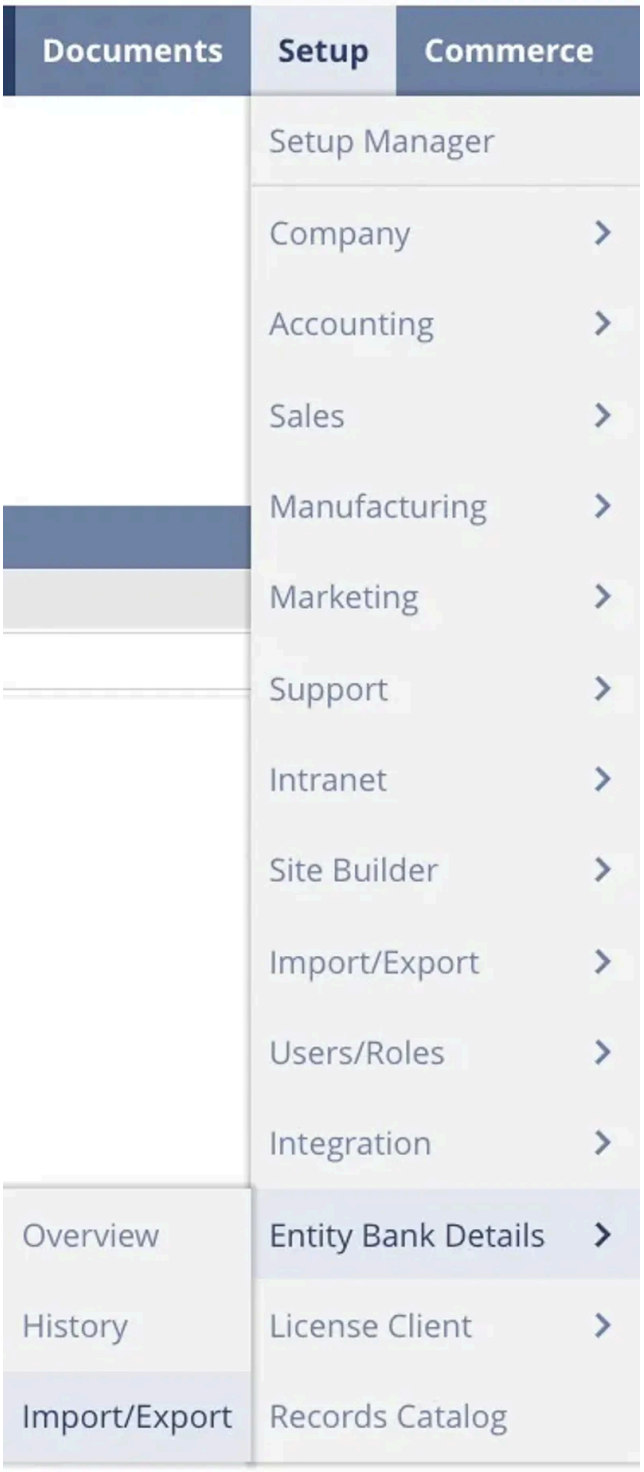
Le processus d'importation et d'exportation des enregistrements dans les onglets Fournisseur et Employé est le même. La seule différence se trouve dans l'onglet Fournisseur, qui propose un utilitaire d'exportation spécialisé pour les utilisateurs du paquet de Paiements Bancaires Électroniques de NetSuite, leur permettant de transférer et de re-télécharger facilement des données dans l'EBD.

Importer les Données Bancaires de l'Entité

Le processus d'importation des données EBD est le même pour les Enregistrements de Fournisseurs et les Enregistrements d'Employés.

1 Go to the Import/Export Tab

Aller à **Configuration → Entité Détails bancaires → Import/Export**



Go to Import/Export

2 Import CSV template

Sélectionner **Modèle** pour télécharger un modèle CSV vierge. Utilisez ce CSV pour remplir les détails bancaires pour un téléchargement en masse.

Import Data section. Click on the Download Template button to get a blank CSV

> **REMARQUE :** Utilisez cette fonctionnalité uniquement pour créer de nouveaux enregistrements. Pour mettre à jour des enregistrements existants, vous devez utiliser le **Exporter CSV** fonctionnalité, car elle inclut la colonne ID nécessaire, qui est absente du modèle CSV vierge.

3 Format and Fill Out CSV

Veuillez consulter le **Exigences de champ pour CSV** section de cette page pour un tableau contenant toutes les exigences de formatage pour le fichier CSV.

> **REMARQUE :** Confirmez que vous utilisez le code pays IBAN (par exemple, CA).

> **CONSEIL :** Avez-vous des zéros en tête dans vos données ? Des apostrophes ['] devant les zéros en tête ou le formatage des cellules en tant que "texte brut" conserveront les zéros en tête dans Microsoft Excel ou Google Sheets, mais PAS dans Apple Numbers.

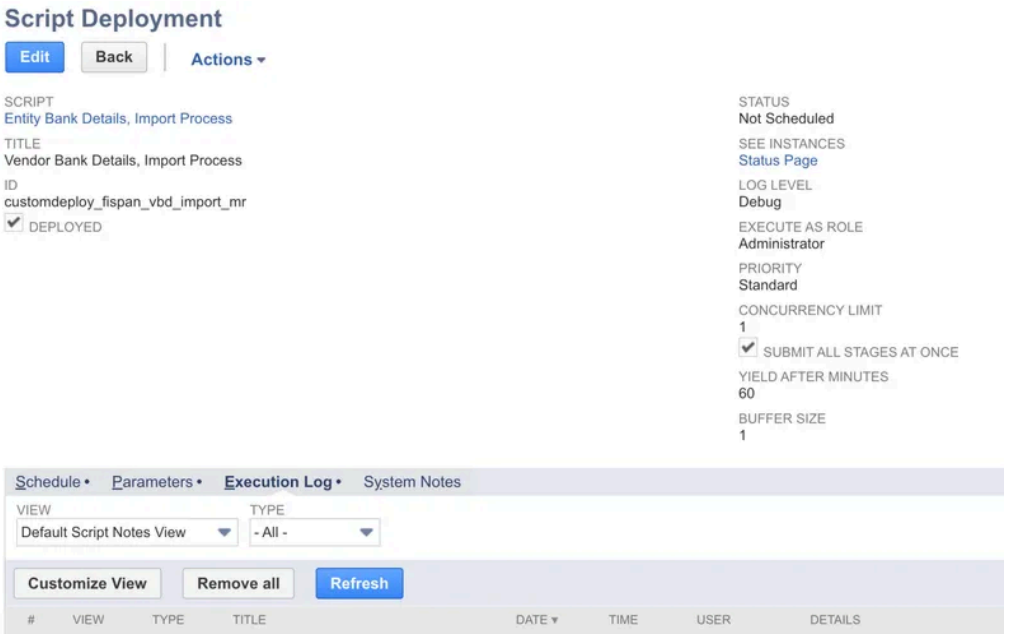
4 Upload the Formatted CSV

Dans le même écran d'importation/exportation, téléchargez le document CSV formaté.

Confirmez le téléchargement réussi en vérifiant le **Importation Logs** pour vous assurer que tous les enregistrements ont été téléchargés avec succès.

Import Data section

Sélectionner **Importer les journaux → Journaux d'exécution** pour confirmer l'achèvement de l'importation.



Execution Logs

5 Check Information

Avant d'envoyer des paiements, vérifiez le **Entité Détails bancaires** de l'onglet du dossier du fournisseur pour vous assurer que toutes les informations sont correctement mises à jour.

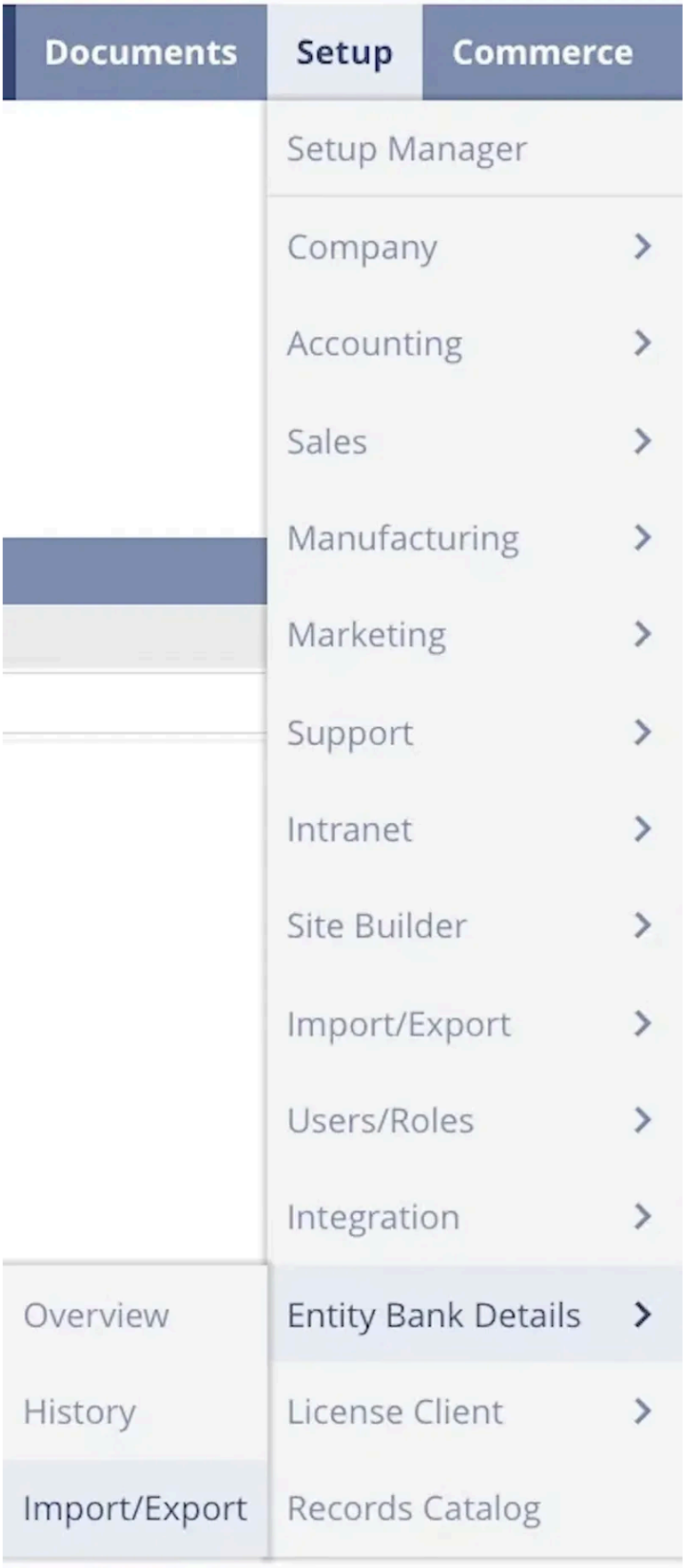
> **IMPORTANT :** Tous les détails bancaires des entités pour les fournisseurs et les employés, ainsi que tous les journaux d'historique, sont stockés dans votre environnement NetSuite. Veuillez vous assurer que vous exportez régulièrement vos enregistrements de détails bancaires des entités et vos journaux d'historique pour éviter toute perte potentielle, comme par exemple lors de la désinstallation du bundle.

Exporter les données bancaires de l'entité

Le processus d'exportation des données EBD est le même pour les dossiers des fournisseurs et les dossiers des employés.

1 Allez à l'onglet Import/Export

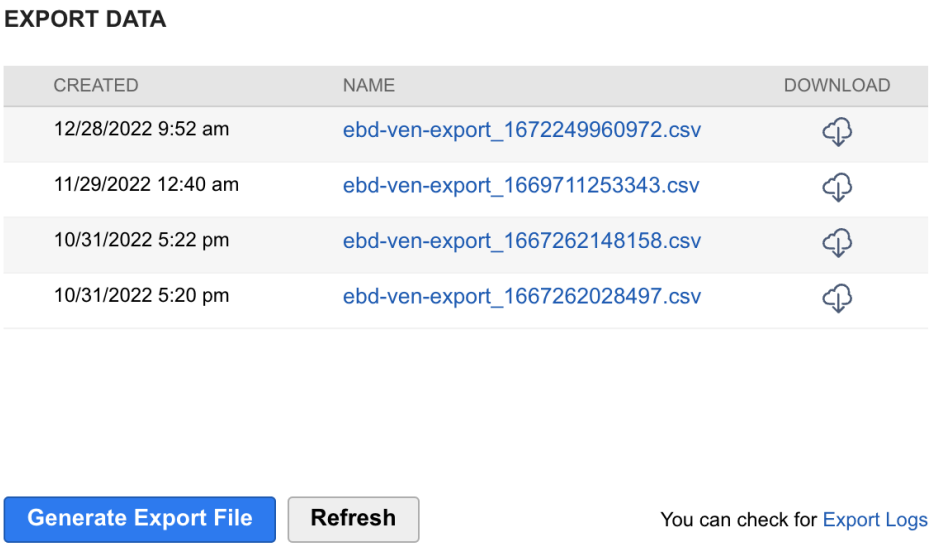
Allez à **Configuration → Entité Détails bancaires → Import/Export**



Allez à Import/Export

2 Modèle d'exportation CSV

Cliquez sur le **Générer le fichier d'exportation** bouton pour déclencher un téléchargement CSV de tous les enregistrements conservés dans EBD.



Section Exporter des données de la page Import/Export

Le tableau affichera un historique de toutes les exportations créées, avec des horodatages et des boutons pour télécharger chacun de ces fichiers.

Les journaux d'exportation contiennent plus d'informations sur les fichiers d'exportation générés. Cela peut être trouvé sous l'onglet **Journal d'exécution**. Pour chaque ligne, il indiquera qui a initié l'exportation, le nombre d'enregistrements exportés dans un fichier, ainsi que le nom du fichier pour cette exportation particulière.

Mettre à jour les enregistrements EBD existants

Utilisez la fonction Importer/Exporter pour modifier en masse les détails de la banque d'entités (EBD) existants pour plusieurs enregistrements d'employés et de fournisseurs à la fois.

1 Exporter le fichier CSV

Pour mettre à jour les enregistrements de détails de banque d'entités (EBD) existants, vous devez d'abord exporter un fichier CSV de ces enregistrements.

Le fichier CSV exporté est crucial car il contient une colonne d'ID unique. Cette colonne d'ID n'est pas présente dans le modèle d'importation CSV vierge et est

nécessaire pour que le système identifie et mette à jour les données existantes.

2 Apportez des modifications au CSV

À l'aide de votre application préférée (comme Microsoft Excel ou Google Sheets), ouvrez le fichier CSV exporté pour apporter vos modifications.

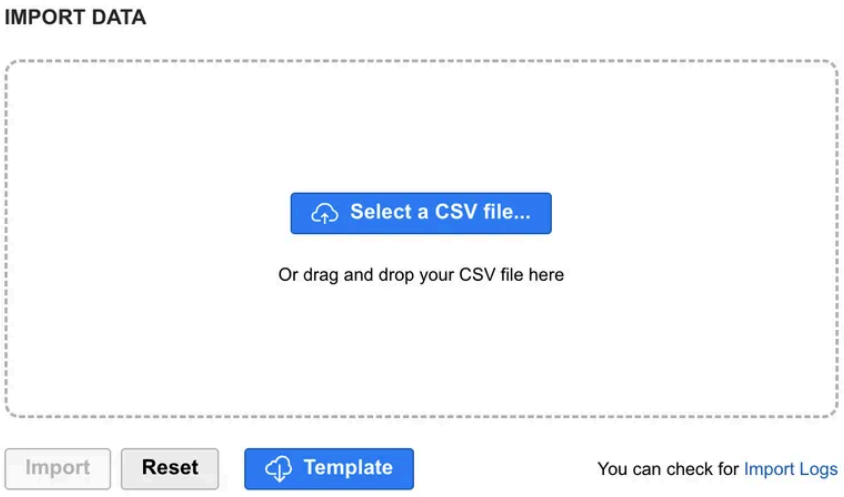
EBD Export CSV du fournisseur ouvert dans Numbers. Notez la colonne ID à gauche

Assurez-vous que la **colonne ID** à gauche reste intacte.

Vous pouvez également ajouter de nouveaux enregistrements à la liste ; il vous suffit de les entrer comme une nouvelle ligne en bas du fichier CSV. Le système attribuera un ID à ces nouveaux enregistrements lors de l'importation.

3 Importer le fichier Export modifié

Après avoir apporté des modifications au fichier précédemment exporté, vous pouvez maintenant importer le fichier CSV dans le bundle des détails de la banque d'entités.



Section Importer des données. Cliquez sur "Sélectionner un fichier CSV" ou faites glisser et déposez un fichier CSV dans cette section pour importer votre fichier CSV

Soit **Sélectionner un fichier CSV** à télécharger, ou faites glisser et déposez-en un dans la section **Importer des données** de la page Importer/Exporter.

Dépannage des problèmes d'importation courants

Problème	Résolution
Aucun message de succès n'apparaît dans NetSuite après l'importation des données bancaires de l'entité.	Vérifiez les journaux d'importation directement pour vérifier l'état de l'importation.
Les enregistrements ont été créés avec succès mais semblent incomplets ou inexacts.	Les enregistrements peuvent être créés même s'ils manquent de champs requis ou ont un format incorrect. Validez les enregistrements importés pour leur exactitude après chaque téléchargement.
Des enregistrements bancaires en double sont créés pour un fournisseur/un employé.	Cela se produit si les mêmes enregistrements sont importés plus d'une fois. Assurez-vous d'importer des comptes bancaires de fournisseurs ou d'employés uniques avec chaque fichier d'importation.
La méthode de paiement internationale est mal définie ou manquante.	Lors de l'importation, la méthode de paiement pour tous les comptes bancaires internationaux doit être saisie comme INTERNATIONAL_WIRE dans le modèle EBD. De plus, assurez-vous que les comptes bancaires pour les pays de virement international ne sont pas définis sur la méthode de paiement DOMESTIC dans le profil NetSuite du fournisseur.

FAQs

Est-il possible d'ajouter des champs supplémentaires ?

Non. NetSuite ne permet actuellement pas l'ajout de champs à l'enregistrement EBD.

Pourquoi les champs Solde bancaire et Solde au date dans la page de correspondance des données bancaires sont-ils vides après une importation réussie ?

C'est une limitation de NetSuite. Le système ne remplit pas les informations de solde bancaire car il dépend de l'utilisation d'un flux bancaire, qui n'est pas intégré ici.

Exigences de champ pour CSV

Assurez-vous que votre fichier d'importation CSV respecte ces exigences.

IMPORTANT : Assurez-vous que toute donnée avec des zéros en tête est précédée d'un apostrophe `'` ou formatée en texte brut dans Microsoft Excel ou Google Sheets.

REMARQUE: Si vous utilisez des méthodes de paiement qui nécessitent des codes de pays obligatoires pour les détails bancaires de l'entité, il peut y avoir des colonnes supplémentaires qui doivent être incluses dans l'importation.

Pour garantir une importation réussie, nous vous recommandons d'abord d'ajouter manuellement les détails bancaires de l'entité au système ERP. Ensuite, téléchargez le modèle ; il affichera déjà les colonnes correctes nécessaires pour entrer les détails restants.

Nom	Valeurs	Description
identifiant	Numérique	Ceci est le numéro d'identification pour cet enregistrement spécifique. Cette colonne n'est pas disponible dans le modèle d'importation car elle est générée par le système.
vendeur (ou employé)	ID de fournisseur NetSuite ou ID d'employé	Requis. L'identifiant du fournisseur ou de l'employé peut être trouvé dans l'URL de l'enregistrement de l'entité ou sur la page d'aperçu EBD.
étiquette	Tout	Requis. Cela peut être des informations sur l'entité ou la banque. Ce champ est utilisé pour différencier les enregistrements lors des téléchargements en masse.
primaire	OUI/NON, O/N, VRAI/FAUX, V/F	Requis. T = Vrai, F = Faux. S'il y a plusieurs détails bancaires pour un seul mode de paiement sur un seul fournisseur importé, marquez-en un comme T et tous les autres comme F.
pays	Code de pays ISO à 2 lettres (par exemple, US, CA, UK)	Requis. Le champ accepte des valeurs en majuscules, minuscules et mixtes.
monnaie	Code de devise à 3 lettres requis (par exemple, USD, CAD)	Requis. Le champ accepte des valeurs en majuscules, minuscules et en casse mixte.
méthode	DOMESTIQUE, TRANSFERT INTERNATIONAL	Requis. Doit être dans le même format car il est sensible à la casse. Veuillez utiliser INTERNATIONAL_WIRE pour les comptes bancaires configurés pour les paiements par virement international.

Nom	Valeurs	Description
		Veuillez utiliser DOMESTIC pour les comptes basés aux États-Unis/Canada.
nomDeLaBanque	Tout	
adresse.ligne1	Tout	
adresse.ville	Tout	
adresse.provinceÉtat	Tout	
code postal	Tout	
bic	Code BIC ou SWIFT valide pour le pays et la banque	
typeDeCompte	VÉRIFICATION, ÉPARGNE	Le champ accepte des valeurs en majuscules, en minuscules et en cas mixte.
numéroDeCompte	Numéro de compte valide pour cette banque	
codeDeBrancheLocale	Code de branche valide pour cette banque	Utilisez ce champ pour les codes BSB si le pays l'exige.
messageDeButDePaiement	Tout	
codeDeButDePaiement	Code de but valide	
codeIsoDePaiement.pardéfautsDePaiement	Code ISO valide	
code de paiement.co	Code de paiement valide	

Nom	Valeurs	Description
de mot clé		
typeDePartieDePaiement.par défautsDePaiement	Type de partie valide	P = Parent, T = Filiale, G = Groupe, N = Non lié
statutRésidentiel.paiementDéfauts	Statut résidentiel valide	
codeBanque	Code bancaire valide pour cette banque	
iban	Code pays à 2 lettres, suivi de deux chiffres de contrôle, et jusqu'à 35 caractères alphanumériques	
code de tri	Code de tri valide pour cette banque	
numéro d'institution	Numéro d'institution valide pour cette banque	
numéro de transit	Numéro de transit valide pour cette banque	
numéroDeRoutage	Numéro de routage valide pour cette banque	

Transfert des données de paiements bancaires électroniques

Avant l'installation du bundle Détails de la Banque de l'Entité (EBD), vous êtes limité à effectuer uniquement des paiements ACH et CPA. Cela se fait via le plugin bancaire, en

utilisant les informations des fournisseurs stockées dans le bundle de Paiements Bancaires Électroniques (EBP) de NetSuite.

Une fois l'EBD installé, le bundle EBP n'est plus utilisé pour extraire les données bancaires des fournisseurs pour les paiements de factures. Au lieu de cela, l'EBD devient la nouvelle source d'informations de paiement.

Le bundle EBD permet de transférer les informations bancaires des fournisseurs ACH et CPA du bundle EBP via CSV.

Les informations bancaires ACH et CPA stockées dans les formats de fichiers de paiement par défaut suivants sont prises en charge :

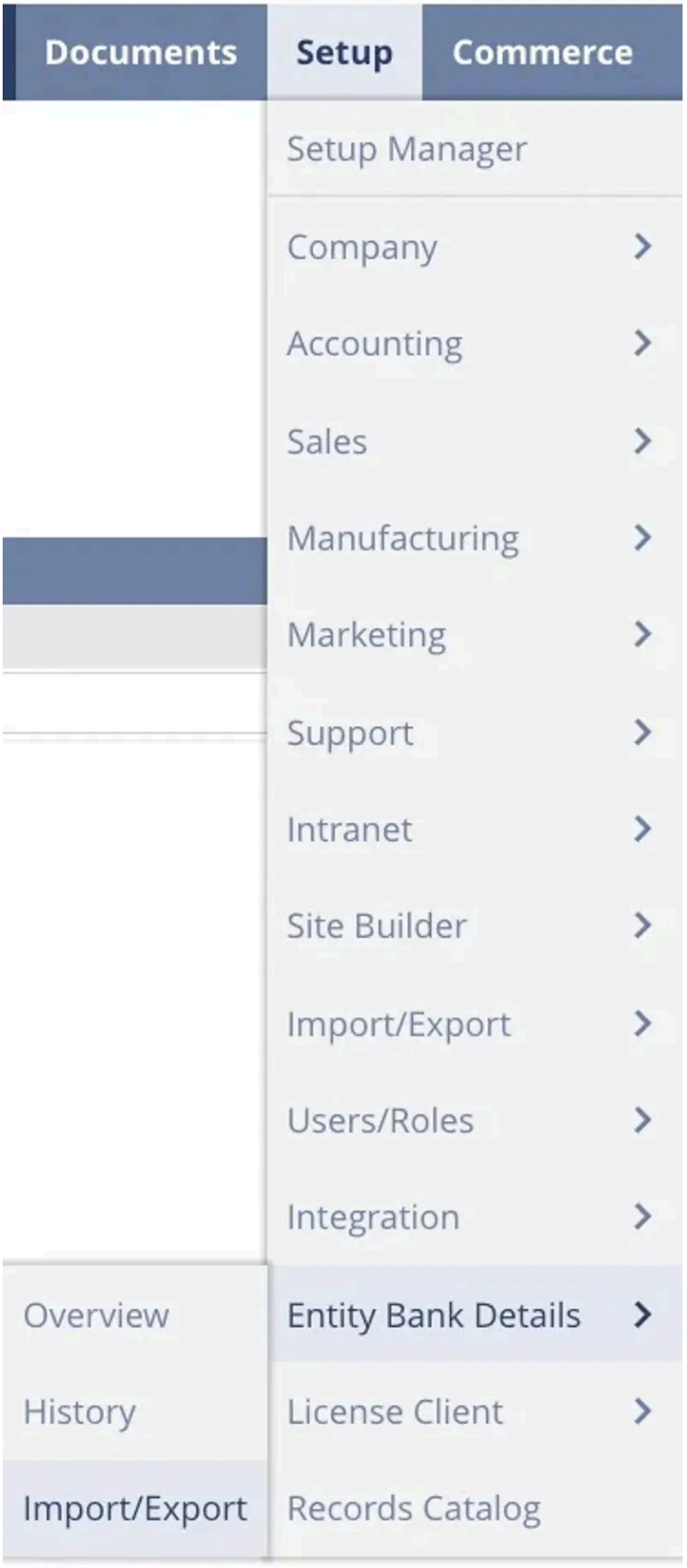
- ACH - CCD/PPD
- ACH - CTX (Texte libre)
- CPA-005

Le transfert des informations bancaires ACH et CPA qui sont stockées dans des formats de fichiers de paiement personnalisés n'est pas pris en charge.

Pour transférer rapidement vos données prises en charge entre les ensembles, suivez les étapes ci-dessous.

1 Go to the Import/Export Tab

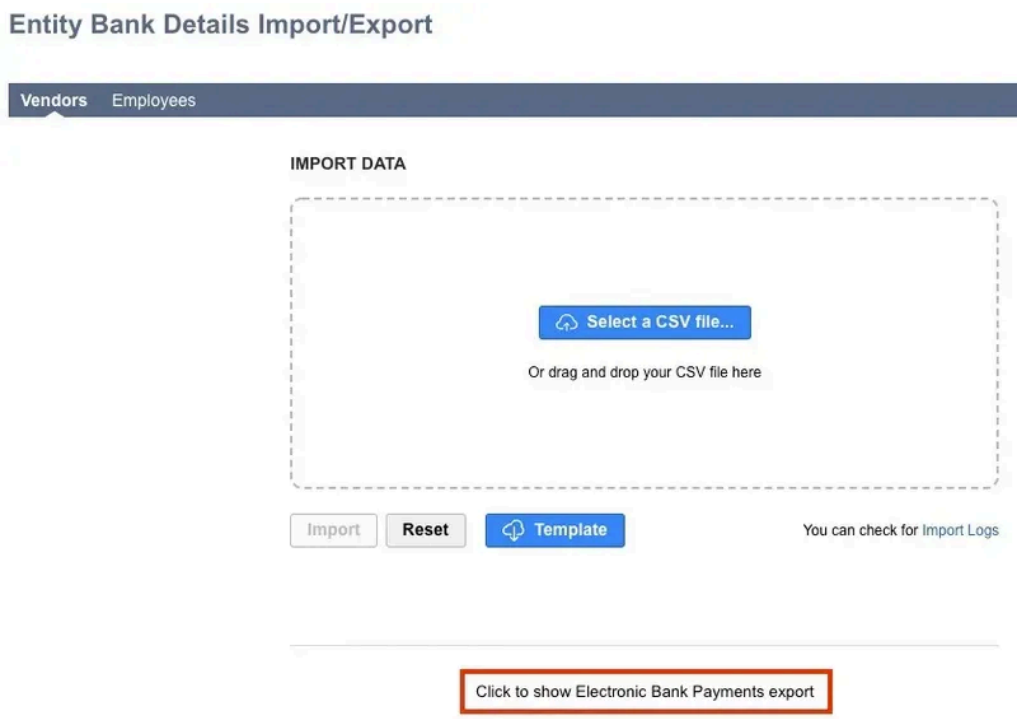
Aller à **Configurer → Détails bancaires de l'entité → Import/Export**



Go to Import/Export

2 Generate CSV(s)

Sélectionner **Cliquez pour afficher l'exportation des paiements bancaires électroniques**

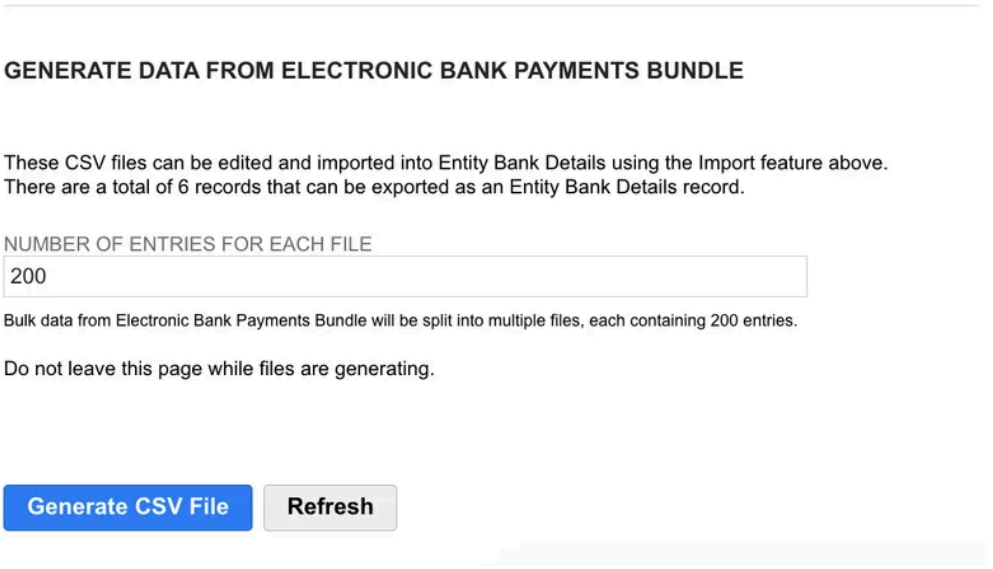


Click to show Electronic Bank Payments export

Sélectionner **Générer un fichier CSV.**

Cela générera un ou plusieurs fichiers CSV en fonction du **Nombre d'Entrées pour Chaque Fichier**, défini ci-dessous.

NetSuite limite le nombre d'enregistrements pouvant être extraits par fichier. Par exemple, si vous avez 1100 enregistrements et que vous définissez le nombre d'entrées à 200, le bundle générera cinq fichiers CSV contenant 200 enregistrements et un fichier CSV contenant 100 enregistrements.

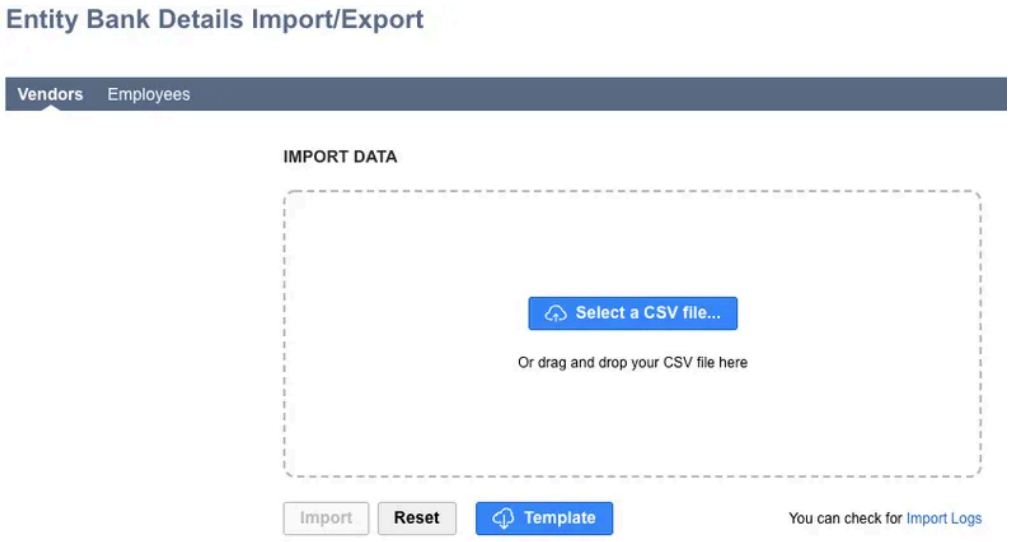


Generate CSV file

REMARQUE : Le **Générer des données à partir du bundle de paiements bancaires électroniques** ne peut extraire que les détails de paiement ACH et CPA. Tout détail de paiement supplémentaire doit être transféré dans le bundle des détails bancaires de l'entité soit par saisie manuelle, soit en utilisant la fonction d'importation.

3 Import CSV(s)

Sélectionnez un fichier CSV...ou faites glisser et déposez le(s) fichier(s) CSV dans le **Importer des données** boîte.



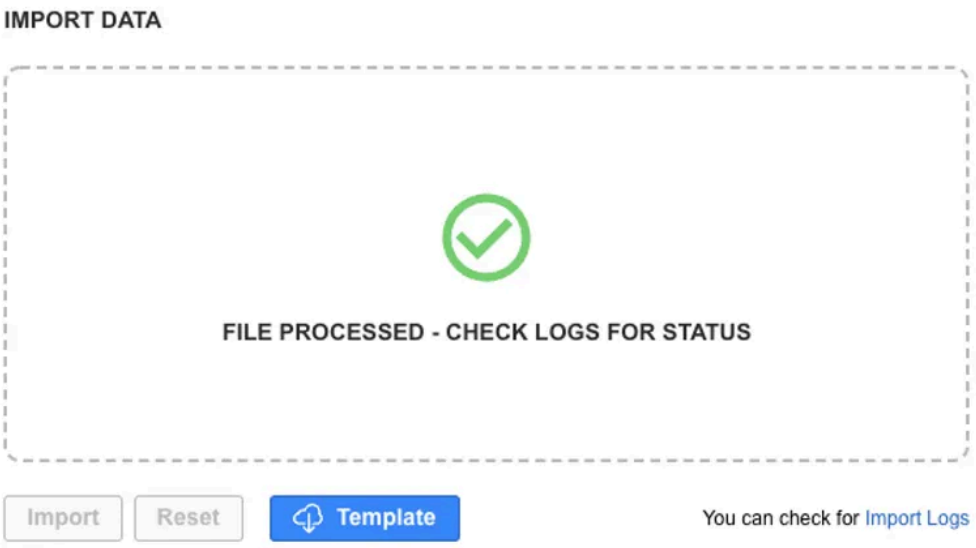
Import Data section

Ensuite, sélectionnez **Importer.**

Attendez quelques instants que l'importation soit terminée.

4 Confirm Import Completion

Une fois terminé, sélectionnez **Importer les journaux → Journaux d'exécution** pour confirmer l'achèvement de l'importation.



Confirm Status

Vous pouvez également vérifier les endroits suivants pour confirmer que les détails bancaires ont été enregistrés :

- Onglet Détails de la Banque de l'Entité sur le Dossier du Fournisseur ou de l'Employé
- Aperçu des détails de la banque de l'entité
- Historique des détails de la banque de l'entité

8.3.1. asdsadas

About Profile | Frequently Asked Questions

Profile at a Glance

Que vous commenciez tout juste ou que vous recherchiez des conseils rapides, cette section vous guidera à travers les bases de l'utilisation de Profile. Chaque article comprend des liens vers des sujets connexes, afin que vous puissiez explorer davantage si nécessaire.

Cette section présente les fonctionnalités et fonctions essentielles de Profile qui soutiennent l'utilisation quotidienne. Elle couvre des informations générales telles que la connexion et la déconnexion, le verrouillage de votre écran, la navigation entre les fenêtres et la personnalisation de votre espace de travail. Vous trouverez également des conseils sur l'utilisation des filtres et des outils de recherche pour localiser des informations, des touches de raccourci, des paramètres de mise en page pour améliorer l'efficacité, et plus encore.

Ces ressources sont conçues pour aider tous les utilisateurs à gagner en confiance et à rester productifs lors de l'utilisation de Profile.

Get Started

Si vous souhaitez en savoir plus sur les informations générales liées au Profil, veuillez contacter votre Responsable du Succès Client.

Job Aids

Gardez les informations clés à portée de main. Cliquez ci-dessous pour télécharger des guides de référence rapide imprimables :

JobAid-IntroductionToProfile.pdf

Related Articles

Lisez les articles connexes ci-dessous pour plus d'informations générales sur le profil :

- Connexion au profil
- Déconnexion du profil
- Aperçu de la navigation dans le profil
- Utiliser la recherche dans le profil
- Naviguer dans la fenêtre du centre de travail
- Naviguer dans les dossiers médicaux dans le profil
- Travailler avec les fenêtres de profil
- Utilisez la fenêtre des détails cliniques
- Utiliser des listes de segments et des vues d'enregistrement
- Insérer et modifier des informations dans le profil
- Utiliser des filtres dans le profil
- Changement de lieu de service
- Déconnexion du profil
- Verrouillez votre écran dans le profil
- Enregistrez les paramètres de votre profil Windows
- Basculer entre les fenêtres ouvertes dans le profil
- Utiliser les touches de raccourci

8.3.2. Langage Toxique

Le garde-fou contre le langage toxique détecte le contenu nuisible, y compris les discours de haine, les menaces, les insultes et d'autres communications qui pourraient nuire à votre communauté ou à votre marque. Contrairement aux simples filtres de mots-clés, ce garde-fou utilise l'IA pour comprendre le contexte, le ton et l'intention.

Quand utiliser ce garde-fou

Vous devriez utiliser la détection de langage toxique lorsque vous devez protéger votre communauté des interactions nuisibles, empêcher votre IA de recevoir un contexte empoisonné qui pourrait influencer son comportement, ou maintenir la sécurité de la marque dans les résultats générés par l'IA. Ce garde-fou est particulièrement précieux dans les applications avec du contenu généré par les utilisateurs, comme les forums, les systèmes de chat et les sections de commentaires, ainsi que dans les scénarios de service client où vous devez attraper des messages hostiles avant qu'ils n'atteignent les agents ou les systèmes d'IA.

L'avantage clé par rapport à un simple filtrage par mots-clés est que ce garde-fou comprend les nuances. Quelqu'un peut être hostile sans utiliser de grossièretés, et il peut utiliser un langage fort sans être hostile. Le garde-fou analyse l'intention et le ton, pas seulement le choix des mots.

Comprendre les niveaux de sensibilité

Le paramètre de sensibilité contrôle la rigueur du garde-fou. Pensez-y comme à l'ajustement du seuil de ce qui constitue une violation. C'est l'un des choix de configuration les plus importants que vous ferez, car il change fondamentalement le contenu qui passe ou échoue.

Une faible sensibilité ne signale que les violations graves comme les menaces explicites et les discours de haine extrêmes. Lorsque vous réglez la sensibilité sur faible, vous dites que vous vous attendez à des opinions fortes et à des désaccords robustes, et que vous ne voulez bloquer que le contenu qui franchit une ligne claire vers un territoire menaçant ou haineux. Par exemple, "Je ne suis pas d'accord avec cette approche et je pense qu'elle est mal orientée" passerait à faible sensibilité, tout comme "C'est une terrible idée." Seul un

contenu comme "Je te trouverai et je te ferai du mal" ou un discours de haine explicite échouerait.

Une faible sensibilité fonctionne bien pour les forums publics où le débat est attendu, les environnements de retour professionnel où la franchise est valorisée, et les communautés techniques où les gens discutent de sujets controversés. Le compromis est que certains contenus qui mettent les gens mal à l'aise peuvent encore passer.

La sensibilité moyenne est le paramètre par défaut et représente une approche équilibrée. À ce niveau, les garde-fous signalent les violations claires, y compris les insultes et le langage hostile, tout en permettant encore le désaccord professionnel et la critique constructive. Un message comme "Je ne suis pas d'accord avec votre raisonnement" passerait, mais "Tu es un idiot" ou "Des gens comme vous sont le problème" échouerait.

La sensibilité moyenne fonctionne bien pour la plupart des applications, y compris les systèmes de service client, les communications professionnelles, les outils collaboratifs et les plateformes sociales. Elle trouve un équilibre entre la possibilité d'un discours significatif et le maintien d'un environnement respectueux.

Une haute sensibilité crée l'environnement le plus strict en signalant tout contenu potentiellement toxique, y compris la légèreté de la grossièreté et le langage désinvolte. À ce niveau, même des contenus comme "Quoi qu'il en soit, mec" ou "C'est plutôt stupide" échoueraient. Seul un contenu respectueux et neutre passe à haute sensibilité.

Une haute sensibilité est appropriée pour les applications destinées aux enfants où vous avez besoin d'une protection maximale, les plateformes éducatives où vous souhaitez modéliser une communication respectueuse, les espaces sûrs et les communautés de soutien où les gens ont besoin de se sentir en sécurité, et les contextes critiques en matière de conformité où tout problème potentiel doit être détecté.

Options de configuration

Le garde-fou de langage toxique accepte plusieurs options de configuration qui vous permettent d'ajuster son comportement pour votre cas d'utilisation spécifique.

```
// Available options:
// sensitivity: "low", "medium", or "high" (default: "medium")
// model: model identifier (default: Claude 3.5 Haiku)
// temperature: 0-1, lower values = more consistent (default: 0.1)
// maxTokens: response length limit (default: 200)

await abv.guardrails.toxicLanguage.validate(text, {
  sensitivity: "medium",
  model: "model-name",
  temperature: 0.1,
  maxTokens: 200,
});
```

```
# Available options:
# sensitivity: "low", "medium", or "high" (default: "medium")
# model: model identifier (default: Claude 3.5 Haiku)
# temperature: 0-1, lower values = more consistent (default: 0.1)
# maxTokens: response length limit (default: 200)

abv.guardrails.toxic_language.validate(text, {
    "sensitivity": "medium",
    "model": "model-name",
    "temperature": 0.1,
    "maxTokens": 200
})
```

L'option de sensibilité est la plus importante et vous l'utiliserez fréquemment. Les options de modèle, de température et de maxTokens sont des paramètres avancés que vous n'aurez généralement pas besoin de modifier. Le modèle par défaut est optimisé pour les tâches de garde-fou et offre le meilleur équilibre entre vitesse, précision et coût. La température par défaut de 0.1 garantit des résultats cohérents. Le maxTokens par défaut de 200 est suffisant pour le champ d'explication.

Exemples du monde réel

Examinons des exemples concrets de la manière dont différents niveaux de sensibilité gèrent divers types de contenu. Comprendre ces modèles vous aidera à choisir la bonne sensibilité pour votre application.

Considérez un message comme "Je ne suis pas d'accord avec votre approche de ce problème." Il s'agit d'un désaccord professionnel et il passe à tous les niveaux de sensibilité. Le langage est neutre et respectueux malgré l'expression d'un désaccord.

Considérez maintenant "C'est une terrible idée et montre un mauvais jugement." Cela passe à faible et moyenne sensibilité car, bien que critique, cela se concentre sur l'idée plutôt que d'attaquer la personne. Cependant, cela pourrait échouer à haute sensibilité car "terrible" et "mauvais jugement" pourraient être considérés comme désinvoltes.

Un message comme "Tu ne sais pas de quoi tu parles" échoue à moyenne et haute sensibilité car il attaque directement la compétence de la personne. Cela pourrait passer à faible sensibilité puisque cela ne contient pas de menaces explicites ou de discours de haine, bien que ce soit limite.

Un contenu comme "Tu es un idiot" ou "Les gens comme toi sont le problème" échoue à tous les niveaux de sensibilité. Ce sont des attaques personnelles claires sans valeur constructive.

Enfin, des menaces explicites comme "Je te trouverai et je te ferai du mal" échouent à tous les niveaux de sensibilité avec une confiance maximale. C'est un contenu toxique sans ambiguïté.

Modèles de mise en œuvre

Voici comment vous utiliseriez typiquement la détection de langage toxique dans différentes parties de votre application.

Pour la validation des entrées, vous vérifiez les messages des utilisateurs avant de les envoyer à votre IA ou de les afficher à d'autres utilisateurs :

```typescript
async function validateUserMessage(message: string): Promise<boolean> {
  const result = await abv.guardrails.toxicLanguage.validate(
    message,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    return true;
  }

  // Log the reason for monitoring, but don't expose it to the user
  console.log("Blocked message:", result.reason);
  return false;
}

// Usage in your message handler
if (await validateUserMessage(userInput)) {
  await processMessage(userInput);
} else {
  return { error: "Your message violates our community guidelines." };
}
```

```python
async def validate_user_message(message: str) -> bool:
    result = await abv.guardrails.toxic_language.validate_async(
        message,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        return True

    # Log the reason for monitoring, but don't expose it to the user
    print(f"Blocked message: {result['reason']}")
    return False

# Usage in your message handler
if await validate_user_message(user_input):
    await process_message(user_input)
else:
    return {"error": "Your message violates our community guidelines."}
```

Pour la validation des sorties, vous vérifiez les réponses générées par l'IA avant de les montrer aux utilisateurs :

```typescript
async function generateSafeResponse(prompt: string): Promise<string> {
  // Generate initial response
  let response = await callAI(prompt);

  // Validate the response
  const validation = await abv.guardrails.toxicLanguage.validate(
    response,
    { sensitivity: "high" }
  );

  // If toxic, regenerate with explicit safety instruction
  if (validation.status === "fail") {
    response = await callAI(
      prompt + "\n\nIMPORTANT: Respond in a professional, respectful tone."
    );
  }

  return response;
}
```

```python
async def generate_safe_response(prompt: str) -> str:
    # Generate initial response
    response = await call_ai(prompt)

    # Validate the response
    validation = await abv.guardrails.toxic_language.validate_async(
        response,
        {"sensitivity": "high"}
    )

    # If toxic, regenerate with explicit safety instruction
    if validation["status"] == "fail":
        response = await call_ai(
            f"{prompt}\n\nIMPORTANT: Respond in a professional, respectful tone."
        )

    return response
```

Pour gérer les cas ambigus, vous pourriez mettre en œuvre une file d'attente de révision pour les résultats incertains :

```typescript
async function handleUserContent(content: string) {
  const result = await abv.guardrails.toxicLanguage.validate(
    content,
    { sensitivity: "medium" }
  );

  if (result.status === "pass") {
    // Content is clearly acceptable
    await publishContent(content);
  } else if (result.status === "fail" && result.confidence > 0.8) {
    // High-confidence violation, auto-reject
    await rejectContent(content, "Community guidelines violation");
  } else {
    // Low confidence or unsure - flag for human review
    await flagForModeration(content, result);
  }
}
```

```python
async def handle_user_content(content: str):
    result = await abv.guardrails.toxic_language.validate_async(
        content,
        {"sensitivity": "medium"}
    )

    if result["status"] == "pass":
        # Content is clearly acceptable
        await publish_content(content)
    elif result["status"] == "fail" and result["confidence"] > 0.8:
        # High-confidence violation, auto-reject
        await reject_content(content, "Community guidelines violation")
    else:
        # Low confidence or unsure - flag for human review
        await flag_for_moderation(content, result)
```

Optimisation des performances

Puisque la détection de langage toxique utilise l'IA, cela prend de une à trois secondes par vérification et consomme des jetons. Vous pouvez optimiser les performances en effectuant d'abord une vérification rapide basée sur des règles pour attraper les violations évidentes avant de faire l'appel coûteux à l'IA.

```
async function efficientToxicCheck(text: string): Promise<boolean> {
  // Quick check for explicitly forbidden terms (under 10ms, free)
  const quickCheck = await abv.guardrails.containsString.validate(
    text,
    {
      strings: ["explicit-slur", "forbidden-term"],
      mode: "none",
    }
  );

  // If quick check fails, no need for expensive AI check
  if (quickCheck.status === "fail") {
    return false;
  }

  // Only run AI check if quick check passed
  const deepCheck = await abv.guardrails.toxicLanguage.validate(text);
  return deepCheck.status === "pass";
}
```

```python
async def efficient_toxic_check(text: str) -> bool:
    # Quick check for explicitly forbidden terms (under 10ms, free)
    quick_check = await abv.guardrails.contains_string.validate_async(
        text,
        {
            "strings": ["explicit-slur", "forbidden-term"],
            "mode": "none"
        }
    )

    # If quick check fails, no need for expensive AI check
    if quick_check["status"] == "fail":
        return False

    # Only run AI check if quick check passed
    deep_check = await abv.guardrails.toxic_language.validate_async(text)
    return deep_check["status"] == "pass"
```

Meilleures pratiques de sécurité

Ne jamais exposer le champ de raison aux utilisateurs finaux. La raison explique pourquoi le contenu a échoué à la validation, et exposer cette information aide les acteurs malveillants à apprendre comment contourner vos garde-fous. Au lieu de cela, utilisez des messages d'erreur génériques tout en enregistrant la raison détaillée en interne pour le suivi et l'amélioration.

```javascript
// Bad - exposes validation logic
if (result.status === "fail") {
  return { error: result.reason };  // Don't do this!
}

// Good - generic message, internal logging
if (result.status === "fail") {
  logger.info("Blocked toxic content", { reason: result.reason });
  return { error: "Your message violates our community guidelines." };
}
```

```
# Bad - exposes validation logic
if result["status"] == "fail":
    return {"error": result["reason"]}  # Don't do this!

# Good - generic message, internal logging
if result["status"] == "fail":
    logger.info(f"Blocked toxic content: {result['reason']}")
    return {"error": "Your message violates our community guidelines."}
```

Choisir la bonne sensibilité

Voici un cadre de décision pour choisir la sensibilité en fonction de votre type d'application. Si vous construisez pour des enfants ou des populations vulnérables, utilisez toujours une haute sensibilité. Le potentiel de préjudice causé par l'autorisation de contenu toxique l'emporte de loin sur le coût des faux positifs.

Si vous construisez une application orientée client comme un service client, des réseaux sociaux ou des outils collaboratifs, une sensibilité moyenne est généralement appropriée. Elle détecte les violations claires tout en permettant des désaccords professionnels.

Si vous construisez pour des publics professionnels ou techniques où un débat robuste est attendu, envisagez une faible sensibilité. Les forums techniques, les systèmes de révision de code et les outils de retour professionnel bénéficient de la possibilité d'exprimer des opinions fortes.

Vous pouvez également ajuster la sensibilité en fonction du contexte de l'utilisateur. Les utilisateurs authentifiés avec un bon historique peuvent bénéficier d'une sensibilité plus faible tandis que les utilisateurs anonymes obtiennent une sensibilité plus élevée. Les utilisateurs qui s'identifient comme mineurs obtiennent automatiquement une haute sensibilité, quelle que soit la configuration par défaut.

Prochaines étapes

La barrière de langage toxique est souvent utilisée en parallèle avec d'autres barrières pour une validation complète du contenu. Envisagez de la combiner avec la détection de langage biaisé pour une solution de sécurité du contenu plus complète. Vous voudrez

peut-être également utiliser une chaîne de contenu pour attraper rapidement les termes interdits explicites avant d'effectuer le contrôle de langage toxique plus coûteux.

Pour des conseils d'implémentation plus détaillés, consultez la documentation des meilleures pratiques qui couvre les stratégies d'optimisation, la gestion des erreurs et les approches de surveillance.

8.4. asdsadasdas (cloné)

Obtenir des gâteaux

Try it

Obtenir un gâteau par son ID

GET https://api.cakes.com

Request

BODY PARAMETERS

id String required

ID du gâteau à obtenir

Nom du paramètre Object optional

Nom du paramètre Object optional

```
var myHeaders = new Headers();
myHeaders.append("Accept", "application/json");
myHeaders.append("Content-Type", "application/json");

var raw = JSON.stringify({
   "id": "String"
});

var requestOptions = {
   method: 'GET',
   headers: myHeaders,
   body: raw,
   redirect: 'follow'
};

fetch("https://api.cakes.com", requestOptions)
   .then(response => response.text())
   .then(result => console.log(result))
   .catch(error => console.log('error', error));
```

```
{
  "name": "Nom du gâteau",
}
```