1. Calls Tab Enhancement

2. Lab Exercise Plan — LF Group HOL

3. test ddd

4. About

5. Aruba 660-330 Linxusme3s

6. Test Adobe cursor jumping on tables

7. MedScout issue with boling and colors

8. Cadent TOC issue

9. 5.4.1 DocumentSearchRequest Element Require

10. test copy paste

10.1. 5.4.1 DocumentRetrieveRequest Element Requirements

11. Dairy Comp page(VAS) with duplicated content (cloned with children) (cloned with children)

12. vas duplicate

13. Testing diacritics

14. CrossRiver test page 3

15. Test folder

15.1. testing missing TOC header

16. CrossRiver test page

17. CrossRiver test page 2

18. rețete

19. Test

20. Ashbyk img alt text dissapear

21. this is a h1 title

22. Baguette parce que car cela ouais

22.1. Test doc

23. Indenting

23.1. Another page

23.1.1. Advanced filtering

23.2. New page

24. test page Ocient

25. Reminder

25.1. Toxic Language

The toxic language guardrail detects harmful content including hate speech, threats, insults, and other communication that could damage your community or brand. Unlike simple keyword filters, this guardrail uses AI to understand context, tone, and intent.

knmknknkjn

Paste an image link or upload your image

Image URL

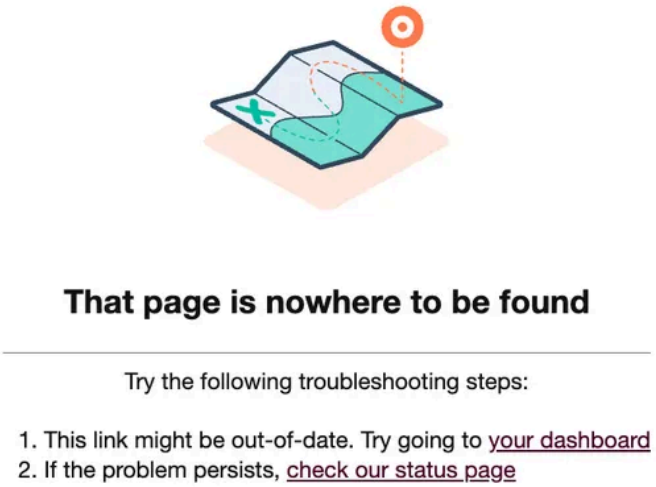
Add Image

When to Use This Guardrail

Paste an image link or upload your image

Image URL

Add Image



The key advantage over simple keyword filtering is that this guardrail understands nuance. Someone can be hostile without using profanity, and they can use strong language without being hostile. The guardrail analyzes intent and tone, not just word choice.

Understanding Sensitivity Levels

The sensitivity setting controls how strict the guardrail is. Think of it as adjusting the threshold for what constitutes a violation. This is one of the most important configuration choices you'll make because it fundamentally changes what content passes or fails.

Low sensitivity flags only severe violations like explicit threats and extreme hate speech. When you set sensitivity to low, you're saying that you expect strong opinions and robust disagreement, and you only want to block content that crosses a clear line into threatening or hateful territory. For example, "I strongly disagree with this approach and think it's misguided" would pass at low sensitivity, as would "This is a terrible idea." Only content like "I will find you and hurt you" or explicit hate speech would fail.

Low sensitivity works well for public forums where debate is expected, professional feedback environments where directness is valued, and technical communities where people discuss contentious topics. The tradeoff is that some content that makes people uncomfortable might still pass through.

Medium sensitivity is the default setting and represents a balanced approach. At this level, the guardrail flags clear violations including insults and hostile language while still allowing professional disagreement and constructive criticism. A message like "I disagree with your reasoning" would pass, but "You're an idiot" or "People like you are the problem" would fail.

Medium sensitivity works well for most applications including customer service systems, business communications, collaborative tools, and social platforms. It strikes a balance between allowing meaningful discourse and maintaining a respectful environment.

High sensitivity creates the strictest environment by flagging any potentially toxic content including mild rudeness and dismissive language. At this level, even content like "Whatever, dude" or "That's pretty dumb" would fail. Only respectful, neutral content passes at high sensitivity.

High sensitivity is appropriate for children's applications where you need maximum protection, educational platforms where you want to model respectful communication, safe spaces and support communities where people need to feel secure, and compliance-critical contexts where any potential issue needs to be caught.

Configuration Options

The toxic language guardrail accepts several configuration options that let you tune its behavior for your specific use case.

```
// 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
})
```

The sensitivity option is the most important and you'll use it frequently. The model, temperature, and maxTokens options are advanced settings that you typically won't need to change. The default model is optimized for guardrail tasks and provides the best balance of speed, accuracy, and cost. The default temperature of 0.1 ensures consistent results. The default maxTokens of 200 is sufficient for the explanation field.

Real-World Examples

Let's look at concrete examples of how different sensitivity levels handle various types of content. Understanding these patterns will help you choose the right sensitivity for your application.

Consider a message like "I disagree with your approach to this problem." This is professional disagreement and passes at all sensitivity levels. The language is neutral and respectful despite expressing disagreement.

Now consider "This is a terrible idea and shows poor judgment." This passes at low and medium sensitivity because while it's critical, it focuses on the idea rather than attacking the person. However, it might fail at high sensitivity because "terrible" and "poor judgment" could be seen as dismissive.

A message like "You don't know what you're talking about" fails at medium and high sensitivity because it attacks the person's competence directly. It might pass at low sensitivity since it doesn't contain explicit threats or hate speech, though it's borderline.

Content like "You're an idiot" or "People like you are the problem" fails at all sensitivity levels. These are clear personal attacks with no constructive value.

Finally, explicit threats like "I will find you and hurt you" fail at all sensitivity levels with maximum confidence. This is unambiguous toxic content.

Implementation Patterns

Here's how you'd typically use toxic language detection in different parts of your application.

For input validation, you check user messages before sending them to your AI or displaying them to other users:

```
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."}
```

For output validation, you check AI-generated responses before showing them to users:

```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
```

For handling ambiguous cases, you might implement a review queue for unsure results:

```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)
```

Performance Optimization

Since toxic language detection uses AI, it takes one to three seconds per check and consumes tokens. You can optimize performance by running a fast rule-based check first to catch obvious violations before making the expensive AI call.

```
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"
```

Security Best Practices

Never expose the reason field to end users. The reason explains why content failed validation, and exposing this information helps bad actors learn how to evade your guardrails. Instead, use generic error messages while logging the detailed reason internally for monitoring and improvement.

```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."}
```

Choosing the Right Sensitivity

Here's a decision framework for choosing sensitivity based on your application type. If you're building for children or vulnerable populations, always use high sensitivity. The potential harm from allowing toxic content through far outweighs the cost of false positives.

If you're building a customer-facing application like customer service, social media, or collaborative tools, medium sensitivity is usually appropriate. It catches clear violations while allowing professional disagreement.

If you're building for professional or technical audiences where robust debate is expected, consider low sensitivity. Technical forums, code review systems, and professional feedback tools benefit from allowing strong opinions.

You can also adjust sensitivity based on user context. Authenticated users with good history might get lower sensitivity while anonymous users get higher sensitivity. Users who identify as minors automatically get high sensitivity regardless of the default setting.

Next Steps

The toxic language guardrail is often used alongside other guardrails for comprehensive content validation. Consider combining it with biased language detection for a more complete content safety solution. You might also want to use contains string to quickly catch explicit forbidden terms before running the more expensive toxic language check.

For more detailed implementation guidance, see the best practices documentation which covers optimization strategies, error handling, and monitoring approaches.

```
asdasdasdasdasdasdasdasdasdsa
```

25.2. ASDASDSDA

ASDASDASDAS

ASDASDASDAS

> ASDSADASDSAD

25.3. Test numbering on images issue (cloned)

The Entity Bank Details (EBD) bundle integrates with the standard NetSuite Vendor Record to securely house all necessary information for submitting payments. Once installed, imported vendor and employee data are accessible via **Setup → Entity Bank Details → Overview**. Learn how to import, export, and update EBD for both vendors and employees in this guide.

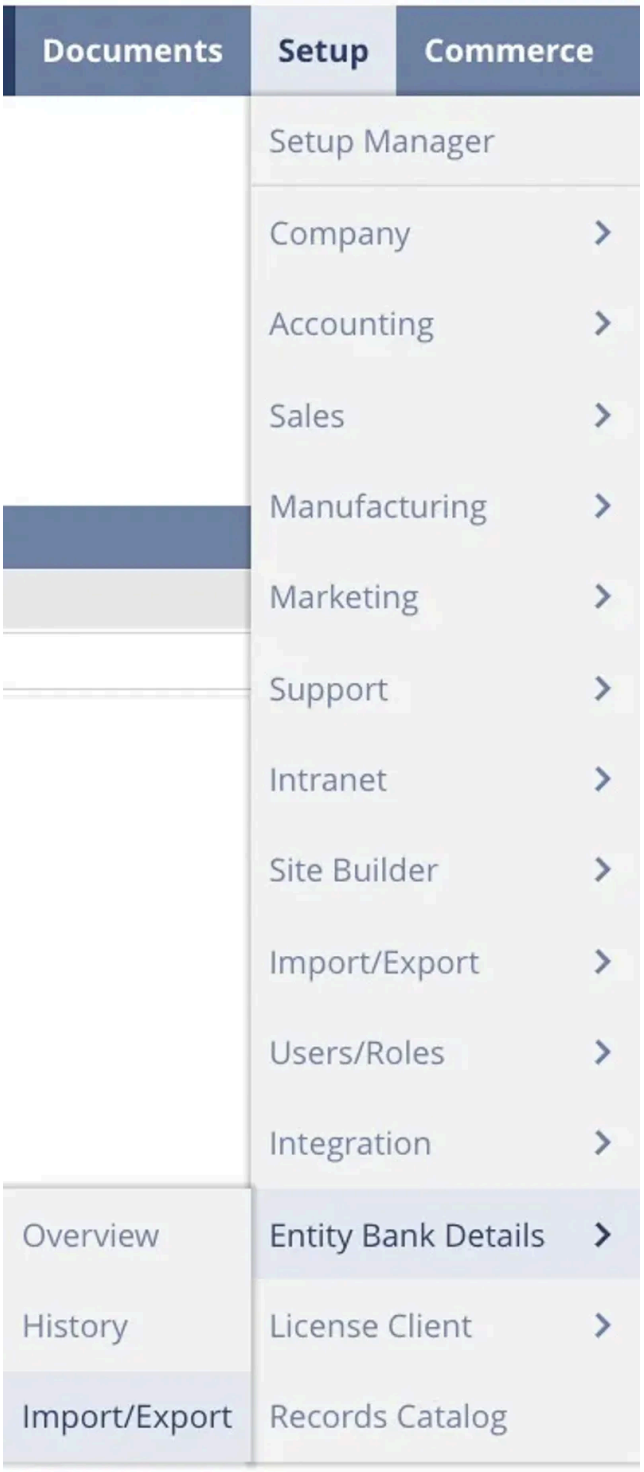
The process for importing and exporting records on the Vendor and Employee tabs is the same. The only difference is on the Vendor tab, which offers a specialized export utility for users with NetSuite's Electronic Bank Payments bundle, allowing them to conveniently transfer and re-upload data into EBD.

Import Entity Bank Data

The process for importing EBD data is the same for both Vendor Records and Employee Records.

1 Go to the Import/Export Tab

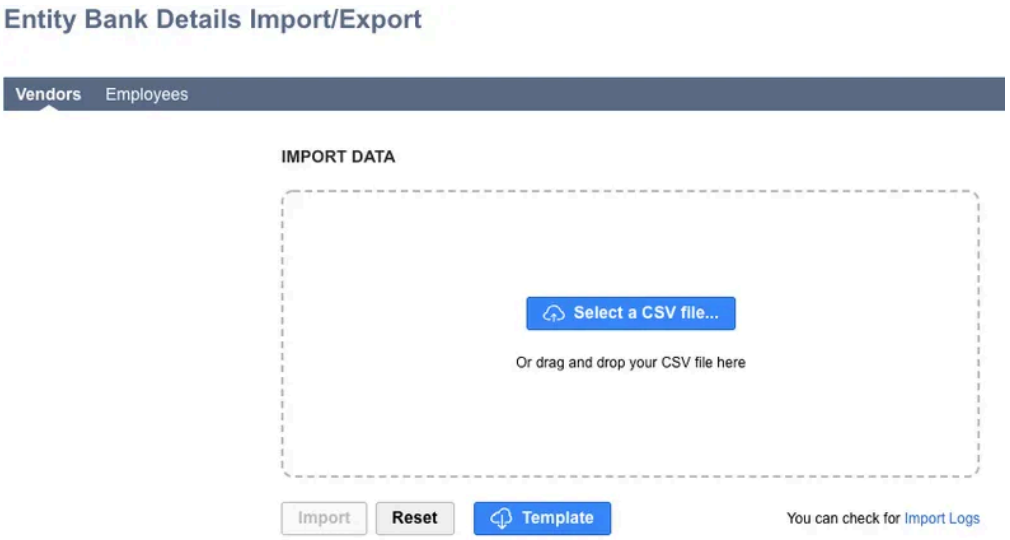
Go to **Setup → Entity Bank Details → Import/Export**



Go to Import/Export

2 Import CSV template

Select **Template** to download a blank CSV template. Use this CSV to fill out Bank Details for bulk upload.



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

> **NOTE:** Use this feature only to create new records. To update existing records, you must use the **Export CSV** feature, as it includes the necessary ID column, which is missing from the blank CSV template.

3 Format and Fill Out CSV

Please see the **Field Requirements for CSV** section of this page for a table containing all formatting requirements for the CSV file.

> **NOTE:** Confirm that you use the IBAN country code (e.g., CA).

> **TIP:** Have leading zeroes in your data? Apostrophes ['] in front of any leading zeroes or formatting cells as "plain text" will keep leading zeroes in Microsoft Excel or Google Sheets, but NOT Apple Numbers.

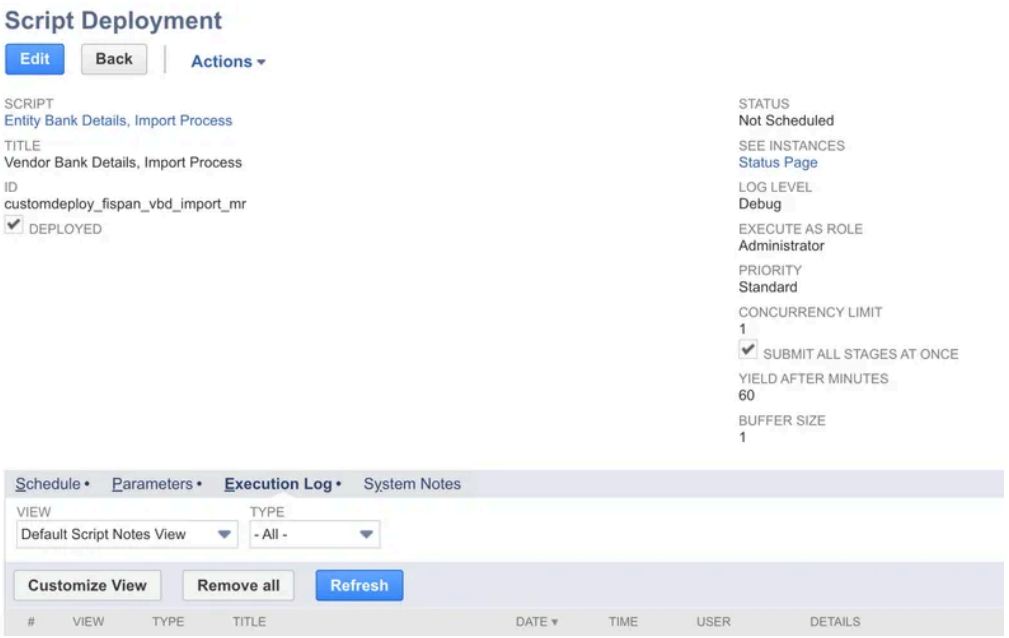
4 Upload the Formatted CSV

In the same Import/Export screen, upload the formatted CSV document.

Confirm successful upload by checking the **Import Logs** to ensure all records were uploaded successfully.

Import Data section

Select **Import logs → Execution logs** to confirm import completion.



Execution Logs

5 Check Information

Prior to sending payments, check the **Entity Bank Details** tab of the Vendor Record to ensure all information is correctly updated.

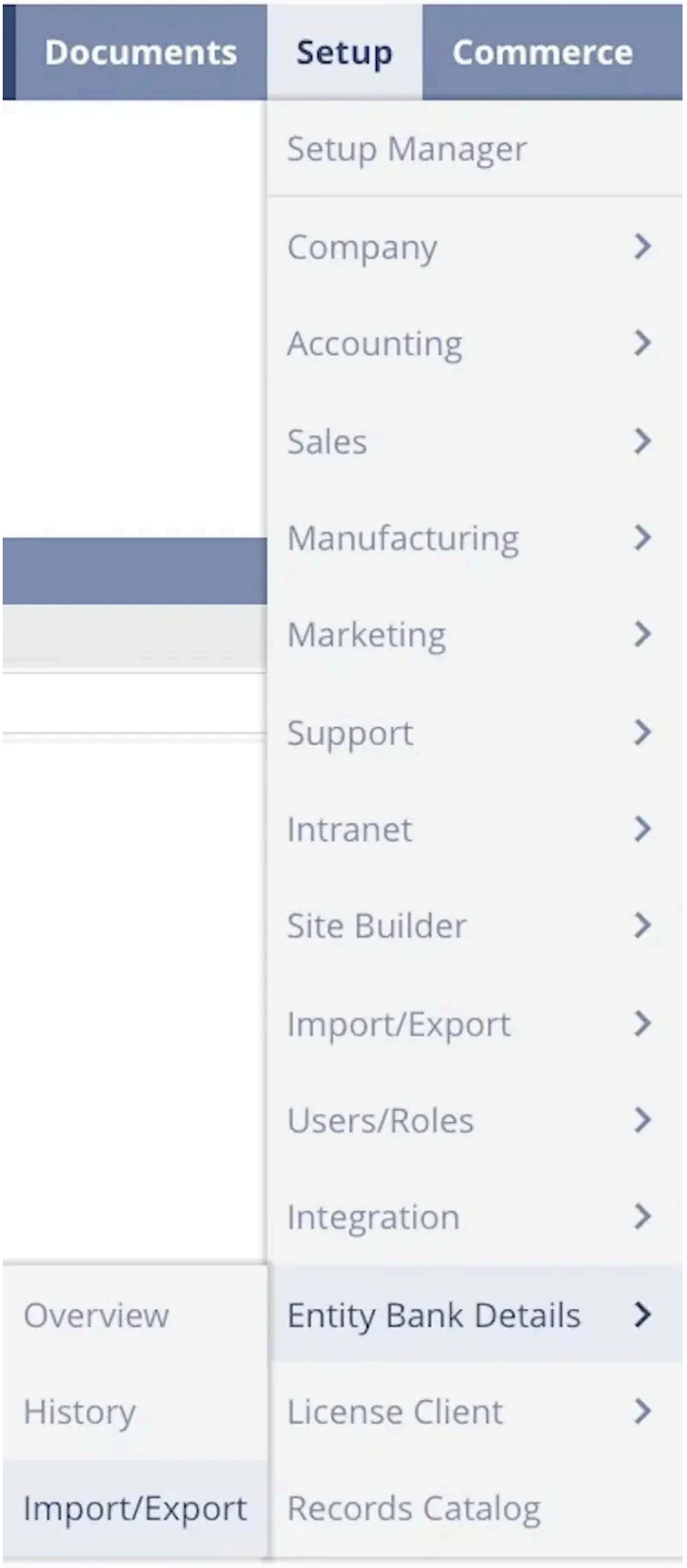
> **IMPORTANT:** All Entity Bank Details for both Vendors and Employees, alongside all History logs, are stored in your NetSuite environment. Please ensure you are regularly exporting your Entity Bank Detail records and your History logs to prevent any potential loss, such as through uninstallation of the bundle.

Export Entity Bank Data

The process for exporting EBD data is the same for both Vendor Records and Employee Records.

1 Go to the Import/Export Tab

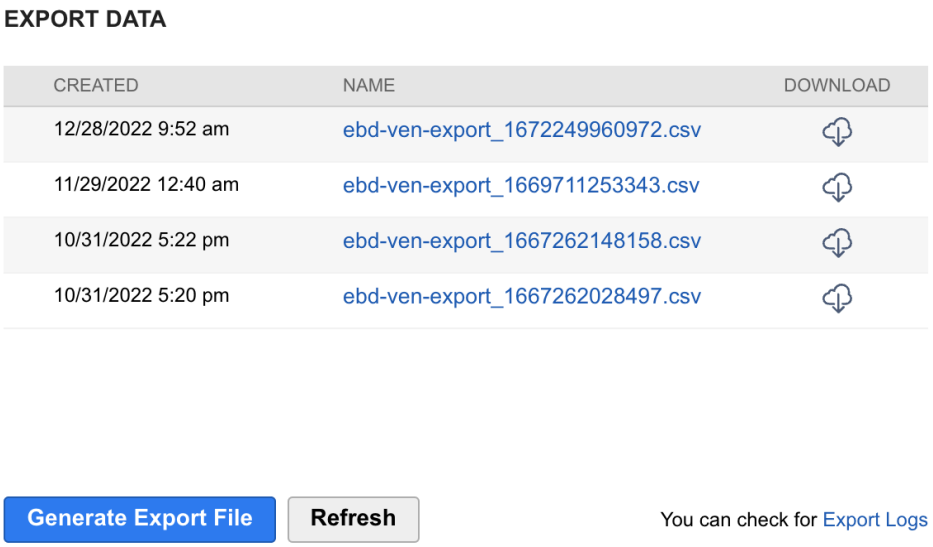
Go to **Setup → Entity Bank Details → Import/Export**



Go to Import/Export

2 Export CSV template

Click on the **Generate Export File** button to trigger a CSV download of all records kept in EBD.



Export Data section of the Import/Export page

The table will display a history of all exports created, alongside timestamps and buttons to download each of these files.

Export Logs contains more information about the export files generated. This can be found under the **Execution Log** tab. For each line item, it will report who initiated the export, the number of records exported in the one file, as well as the file name for that particular export.

Update Existing EBD Records

Use the Import/Export feature to bulk edit existing Entity Bank Details (EBD) for multiple Employee and Vendor Records at once.

1 Export CSV File

To update existing Entity Bank Details (EBD) records, you must first export a CSV file of those records.

The exported CSV file is crucial because it contains a unique ID column. This ID column is not present in the blank CSV import template and is necessary for the system to identify and update existing data.

2 Make edits to the CSV

Using your preferred application (such as Microsoft Excel or Google Sheets), open the exported CSV file to make your edits.

ebd-ven-export_

	id	vendor	label	primary	country	currency	method	bic	accountType	accountNumber	routingNumber
1	1	1305	US-DOMESTIC-USD	T	US	USD	DOMESTIC		CHECKING		

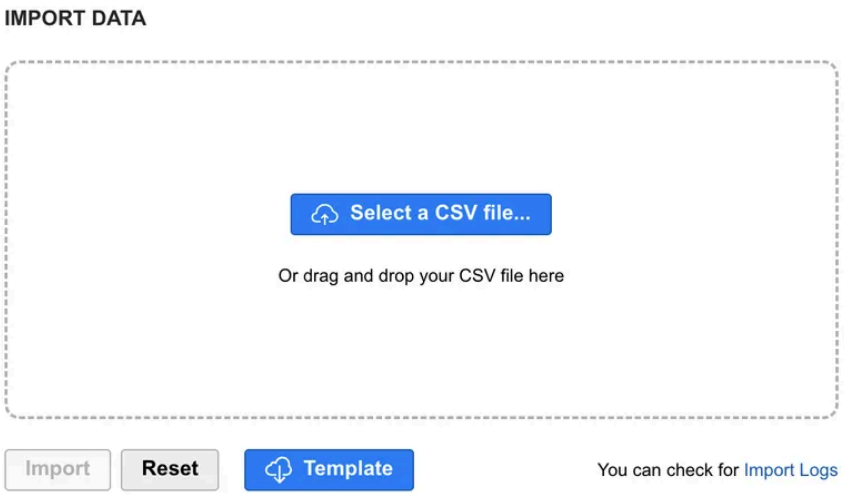
EBD Vendor Export CSV opened in Numbers. Note the ID column on the far left

Ensure the **ID** column on the far left remains untouched.

You can also add new records to the list; simply enter them as a new row at the bottom of the CSV file. The system will assign an ID to these new records upon import.

3 Import the edited Export file

After making edits to the previously exported file, you may now import the CSV file back into the Entity Bank Details bundle.



Import Data section. Click "Select a CSV file" or drag-and-drop a CSV file in this section to import your CSV file

Either **Select a CSV file** to upload, or drag and drop one into the **Import Data** section of the Import/Export page.

Troubleshooting Common Import Issues

Issue	Resolution
No success message appears in NetSuite after importing Entity Bank Data.	Check the import logs directly to verify the import status.
Records were successfully created but appear incomplete or inaccurate.	Records can be created even if they are missing required fields or have incorrect formatting. Validate the imported records for accuracy after every upload.
Duplicate bank records are created for a vendor/employee.	This happens if the same records are imported more than once. Ensure you are importing unique vendor or employee bank accounts with each import file.
The international payment method is incorrectly set or missing.	When importing, the payment method for all international wire accounts must be entered as INTERNATIONAL_WIRE in the EBD template. Additionally, ensure bank accounts for international wire countries are not set to the DOMESTIC payment method in the vendor's NetSuite profile.

FAQs

Is it possible to add additional fields?

No. NetSuite does not currently allow the addition of fields to the EBD record.

Why are the Bank Balance and Balance As Of fields in the Match Bank Data page blank after a successful import?

This is a NetSuite limitation. The system does not populate bank balance information because it relies on the usage of a bank feed, which is not integrated here.

Field Requirements for CSV

Make sure your CSV Import file follows these requirements.

IMPORTANT: Ensure any data with leading zeroes is preceded by an apostrophe `'` or formatted as plain text in Microsoft Excel or Google Sheets.

NOTE: If you are using payment methods that require mandatory country codes for Entity Bank Details, there may be additional columns that must be included in the import.

To ensure a successful import, we recommend first manually adding the Entity Bank Details to the ERP system. Then, download the template; it will already display the correct columns needed for entering any remaining details.

Name	Values	Description
id	Numeric	This is the ID number for this specific record. This column is not available on the Import Template as it is system-generated.
vendor (or employee)	NetSuite Vendor ID or Employee ID	Required. The Vendor or Employee ID can be found in the URL of the entity record or on the EBD Overview page.
label	Anything	Required. This can be entity or bank details. This field is used to differentiate the records when doing mass uploads.
primary	YES/NO, Y/N, TRUE/FALSE, T/F	Required. T = True, F = False. If there are multiple bank details for a single payment method on a single vendor being imported, mark one as T and all others F.
country	2-letter ISO country code (e.g., US, CA, UK)	Required. Field accepts upper, lower, and mixed case values.
currency	3-letter currency code required (e.g., USD, CAD)	Required. Field accepts upper, lower, and mixed case values.
method	DOMESTIC, INTERNATIONAL_WIRE	Required. Has to be in the same format as it is case sensitive. Please use INTERNATIONAL_WIRE for bank accounts set up for international wire payments. Please use DOMESTIC for US/Canada-based accounts.
bankName	Anything	
address.line1	Anything	
address.city	Anything	

Name	Values	Description
address.stateProvince	Anything	
address.postalCode	Anything	
bic	Valid BIC or SWIFT code for the country and bank	
accountType	CHECKING, SAVINGS	Field accepts upper, lower, and mixed case values.
accountNumber	Valid account number for this bank	
localBranchCode	Valid branch code for this bank	Use this field for BSB codes if the country requires it.
paymentDefaults.purposeMessage	Anything	
paymentDefaults.purposeCode	Valid purpose code	
paymentDefaults.paymentIsoCode	Valid ISO code	
paymentDefaults.paymentCodeword	Valid payment code	
paymentDefaults.pay	Valid party type	P = Parent, T = Subsidiary, G = Group, N = Non-related

Name	Values	Description
mentPartyType		
paymentDefaults.residentialStatus	Valid residential status	
bankCode	Valid bank code for this bank	
iban	2-letter country code, followed by two check digits, and up to 35 alphanumeric characters	
sortCode	Valid sort code for this bank	
institutionNumber	Valid institution number for this bank	
transitNumber	Valid transit number for this bank	
routingNumber	Valid routing number for this bank	

Transferring Electronic Bank Payments Data

Before the Entity Bank Details (EBD) bundle is installed, you are limited to making only ACH and CPA payments. This is done through the bank plugin, using vendor information stored in NetSuite's Electronic Bank Payments (EBP) bundle.

Once EBD is installed, the EBP bundle is no longer used to pull vendor banking data for bill payments. Instead, EBD becomes the new source of payment information.

The EBD bundle provides the ability to transfer ACH and CPA vendor banking information from the EBP bundle via CSV.

ACH and CPA banking information stored in the following default Payment File Formats are supported:

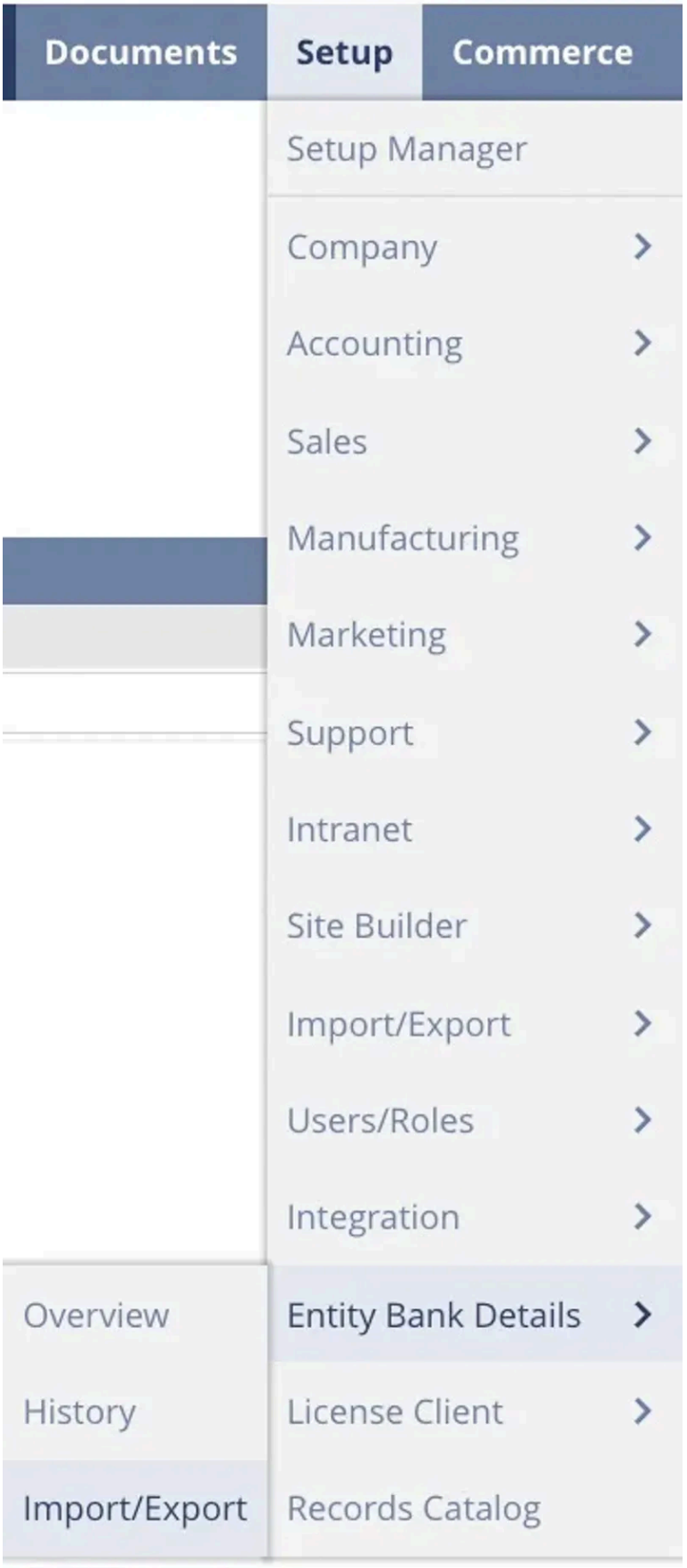
- ACH - CCD/PPD
- ACH - CTX (Free Text)
- CPA-005

Transferring ACH and CPA banking information that is stored in custom Payment File Formats is unsupported.

To quickly transfer your supported data across bundles, follow the steps below.

1 Go to the Import/Export Tab

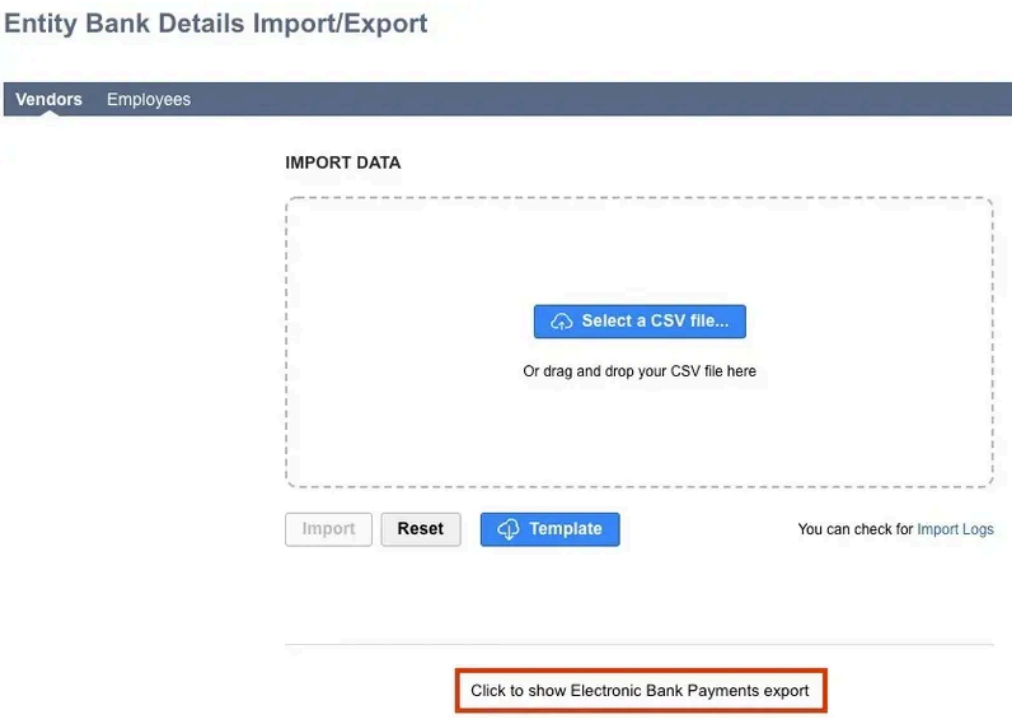
Go to **Set Up → Entity Bank Details → Import/Export**



Go to Import/Export

2 Generate CSV(s)

Select **Click to show Electronic Bank Payments export**

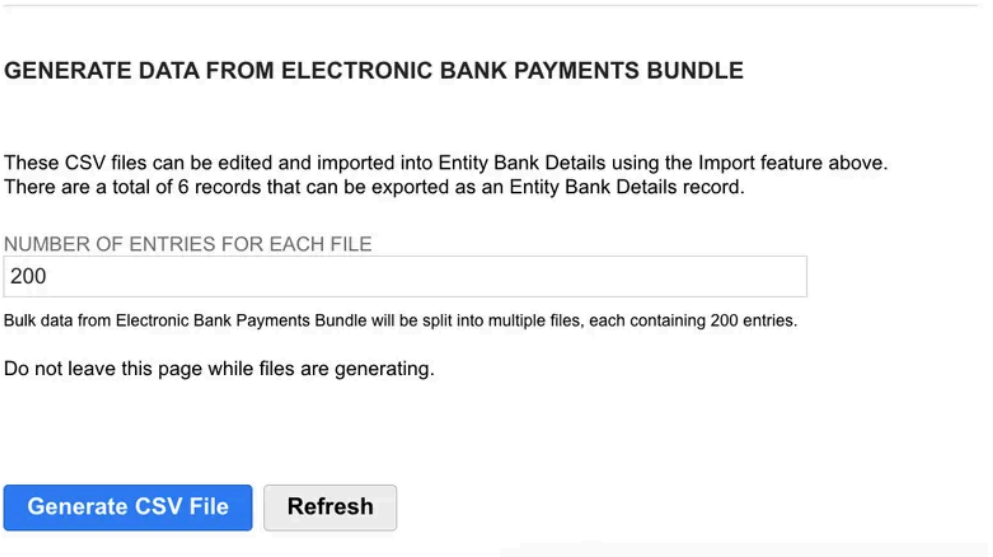


Click to show Electronic Bank Payments export

Select **Generate CSV File.**

This will generate a CSV file(s) dependent on the **Number of Entries for Each File**, set below.

NetSuite limits the number of records that can be pulled per file. For example, if you have 1100 records and set the number of entries to 200, the bundle will generate five CSVs containing 200 records and one CSV containing 100 records.



Generate CSV file

> **NOTE:** The **Generate Data from Electronic Bank Payments Bundle** can only extract ACH and CPA payment details. Any additional payment details must be transferred into the Entity Bank Details bundle either via manual entry or using the Import feature.

3 Import CSV(s)

Select a CSV file... or drag and drop the CSV file(s) into the **Import Data** box.

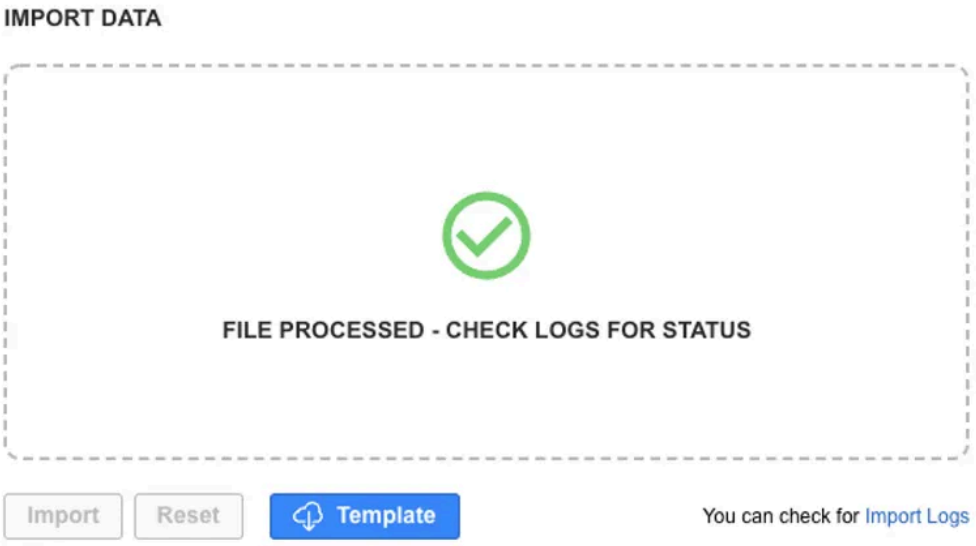
Import Data section

Then, select **Import.**

Wait a few moments for the import to complete.

4 Confirm Import Completion

Once complete, select **Import logs → Execution logs** to confirm import completion.



Confirm Status

You can also check the following places to confirm bank details have been saved:

- Entity Bank Details tab on the Vendor or Employee Record
- Entity Bank Details Overview page
- Entity Bank Details History page

25.3.1. asdsadas

25.3.2. Toxic Language

The toxic language guardrail detects harmful content including hate speech, threats, insults, and other communication that could damage your community or brand. Unlike simple keyword filters, this guardrail uses AI to understand context, tone, and intent.

When to Use This Guardrail

You should use toxic language detection when you need to protect your community from harmful interactions, prevent your AI from receiving poisoned context that could influence its behavior, or maintain brand safety in AI-generated outputs. This guardrail is particularly valuable in applications with user-generated content like forums, chat systems, and comment sections, as well as in customer service scenarios where you need to catch hostile messages before they reach agents or AI systems.

The key advantage over simple keyword filtering is that this guardrail understands nuance. Someone can be hostile without using profanity, and they can use strong language without being hostile. The guardrail analyzes intent and tone, not just word choice.

Understanding Sensitivity Levels

The sensitivity setting controls how strict the guardrail is. Think of it as adjusting the threshold for what constitutes a violation. This is one of the most important configuration choices you'll make because it fundamentally changes what content passes or fails.

Low sensitivity flags only severe violations like explicit threats and extreme hate speech. When you set sensitivity to low, you're saying that you expect strong opinions and robust disagreement, and you only want to block content that crosses a clear line into threatening or hateful territory. For example, "I strongly disagree with this approach and think it's misguided" would pass at low sensitivity, as would "This is a terrible idea." Only content like "I will find you and hurt you" or explicit hate speech would fail.

Low sensitivity works well for public forums where debate is expected, professional feedback environments where directness is valued, and technical communities where people discuss contentious topics. The tradeoff is that some content that makes people uncomfortable might still pass through.

Medium sensitivity is the default setting and represents a balanced approach. At this level, the guardrail flags clear violations including insults and hostile language while still allowing professional disagreement and constructive criticism. A message like "I disagree with your reasoning" would pass, but "You're an idiot" or "People like you are the problem" would fail.

Medium sensitivity works well for most applications including customer service systems, business communications, collaborative tools, and social platforms. It strikes a balance between allowing meaningful discourse and maintaining a respectful environment.

High sensitivity creates the strictest environment by flagging any potentially toxic content including mild rudeness and dismissive language. At this level, even content like "Whatever, dude" or "That's pretty dumb" would fail. Only respectful, neutral content passes at high sensitivity.

High sensitivity is appropriate for children's applications where you need maximum protection, educational platforms where you want to model respectful communication, safe spaces and support communities where people need to feel secure, and compliance-critical contexts where any potential issue needs to be caught.

Configuration Options

The toxic language guardrail accepts several configuration options that let you tune its behavior for your specific use case.

```
// 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
})
```

The sensitivity option is the most important and you'll use it frequently. The model, temperature, and maxTokens options are advanced settings that you typically won't need to change. The default model is optimized for guardrail tasks and provides the best balance of speed, accuracy, and cost. The default temperature of 0.1 ensures consistent results. The default maxTokens of 200 is sufficient for the explanation field.

Real-World Examples

Let's look at concrete examples of how different sensitivity levels handle various types of content. Understanding these patterns will help you choose the right sensitivity for your application.

Consider a message like "I disagree with your approach to this problem." This is professional disagreement and passes at all sensitivity levels. The language is neutral and respectful despite expressing disagreement.

Now consider "This is a terrible idea and shows poor judgment." This passes at low and medium sensitivity because while it's critical, it focuses on the idea rather than attacking the person. However, it might fail at high sensitivity because "terrible" and "poor judgment" could be seen as dismissive.

A message like "You don't know what you're talking about" fails at medium and high sensitivity because it attacks the person's competence directly. It might pass at low sensitivity since it doesn't contain explicit threats or hate speech, though it's borderline.

Content like "You're an idiot" or "People like you are the problem" fails at all sensitivity levels. These are clear personal attacks with no constructive value.

Finally, explicit threats like "I will find you and hurt you" fail at all sensitivity levels with maximum confidence. This is unambiguous toxic content.

Implementation Patterns

Here's how you'd typically use toxic language detection in different parts of your application.

For input validation, you check user messages before sending them to your AI or displaying them to other users:

```
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."}
```

For output validation, you check AI-generated responses before showing them to users:

```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
```

For handling ambiguous cases, you might implement a review queue for unsure results:

```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)
```

Performance Optimization

Since toxic language detection uses AI, it takes one to three seconds per check and consumes tokens. You can optimize performance by running a fast rule-based check first to catch obvious violations before making the expensive AI call.

```
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"
```

Security Best Practices

Never expose the reason field to end users. The reason explains why content failed validation, and exposing this information helps bad actors learn how to evade your guardrails. Instead, use generic error messages while logging the detailed reason internally for monitoring and improvement.

```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."}
```

Choosing the Right Sensitivity

Here's a decision framework for choosing sensitivity based on your application type. If you're building for children or vulnerable populations, always use high sensitivity. The potential harm from allowing toxic content through far outweighs the cost of false positives.

If you're building a customer-facing application like customer service, social media, or collaborative tools, medium sensitivity is usually appropriate. It catches clear violations while allowing professional disagreement.

If you're building for professional or technical audiences where robust debate is expected, consider low sensitivity. Technical forums, code review systems, and professional feedback tools benefit from allowing strong opinions.

You can also adjust sensitivity based on user context. Authenticated users with good history might get lower sensitivity while anonymous users get higher sensitivity. Users who identify as minors automatically get high sensitivity regardless of the default setting.

Next Steps

The toxic language guardrail is often used alongside other guardrails for comprehensive content validation. Consider combining it with biased language detection for a more complete content safety solution. You might also want to use contains string to quickly catch explicit forbidden terms before running the more expensive toxic language check.

For more detailed implementation guidance, see the best practices documentation which covers optimization strategies, error handling, and monitoring approaches.

25.4. asdsadasdas (cloned)

Get Cakes

Try it

Get a cake by its ID

GET https://api.cakes.com

Request

BODY PARAMETERS

id `String` `required`

ID of the cake to get

Parameter name `Object` `optional`

Parameter name `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": "Cake's name",
}
```

26. Virtual Account Number API Specification

26.1. Create an Automatcher (Receivables) Account

POST https://api.mpay.com.au/receivables/v1/create Try it

Production URL

Creates a new virtual account number. These account numbers can be used along with the returned BSB to receive payments from any bank in Australia.

Payments to an invalid bank-account number, i.e. one that has not been created, will be returned by Monoova to the remitting institution. Each account number has a check digit to minimize common payer errors such as adjacent transposition errors or single-digit typos.

An optional unique ID can be linked to each account number for your reference. If a unique ID is not provided, Monoova will generate and return one automatically.

If isActive is omitted, it will default to false. This can be changed to true via the /receivables/v1/status endpoint.

For the creation of batches of account numbers, please see /receivables/v1/batchCreate.

Body Parameters

body automatcherCreateBody_V1 *

Responses

200 successful validation

automatcherStatusResponse_V1

```
curl --request POST \
     --url https://api.mpay.com.au/receivables/v1/create \
     --header 'accept: application/json'
```

```
// successful validation
{}
```

26.2. Get Account Status By Bank Account

GET https://api.mpay.com.au/receivables/v1/statusByBank... Try it

Production URL

This API is used for checking the status of a bank account using the bank account number. This API will only return statuses for virtual accounts belonging to the parent mAccount.

Path Parameters

bankAccountNumber string *

Query Parameters

bsb string

Responses

200 successful validation

automatcherStatusResponse_V1

```
curl --request GET \
     --url https://api.mpay.com.au/receivables/v1/statusByBankAccount/{bankAccountNumber}
     --header 'accept: application/json'
```

```
// successful validation
{}
```

26.3. Batch Create

26.4. Set Account Status

26.5. Batch Status

26.6. Get Account Status By ClientUniqueId

26.7. securitySchemes

26.8. schemas

26.9. schemas

26.10. schemas

26.11. schemas

26.12. schemas

26.13. schemas

27. Judopay Transaction API

27.1. Registering Cards

27.1.1. /transactions/savecard

27.2. Pre-Authorization

27.2.1. /transactions/checkcard

27.2.2. /transactions/registercard

27.2.3. /transactions/preauths

27.2.4. /transactions/incrementalAuth

27.2.5. /transactions/collections

27.2.6. /transactions/voids

27.3. Payments

27.3.1. /transactions/payments

27.3.2. /transactions/refunds

27.4. 3DSecure

27.4.1. /transactions/{receiptId}/resume3ds

27.4.2. /transactions/{receiptId}/complete3ds

27.5. Receipts

27.5.1. /transactions/{receiptId}

27.5.2. /transactions

27.5.3. /transactions/{transactionType}

GET https://api-sandbox.judopay.com/transactions/{transactionType} Try it

Sandbox environment

Return a list of transactions of the given transactionType associated with the given query. Please note: The records returned by this endpoint should not be relied upon for real-time processing.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Path Parameters

- **transactionType** string *

Header Parameters

- **Api-Version** string *

Query Parameters

- **pageSize** integer — 10
- **offset** integer
- **sort** string — time-descending
- **from** string
- **to** string
- **yourPaymentReference** string

- yourConsumerReference string

Responses

- 200 Successful response
  - listTransactionsResponse
    - resultCount number
    - pageSize integer
    - offset integer
    - results transactionReceiptListResponse[]
    - sort string
- 401 Unauthorized
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer
- 403 Forbidden if token used does not have required permission
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer

```
curl --request GET \
     --url 'https://api-sandbox.judopay.com/transactions/{transactionType}?pageSize=10&o
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

27.6. Payment Sessions

27.6.1. Create a generic payment session

POST https://api-sandbox.judopay.com/paymentsession Try it

Sandbox environment

Create a payment session without specifying a payment operation. webPaymentOperation will be 0.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Header Parameters

> Api-Version string *

Body Parameters

- **body** paymentSessionRequestForPreAuths * — All Of: paymentSession...
  - **judoId** string *
  - **yourConsumerReference** string *
  - **yourPaymentReference** string *
  - **yourPaymentMetaData** object *
  - **currency** string *
  - **amount** number<float>
  - **cardAddress** cardAddressWebPaymentRequest — All Of: cardAddressReq...
  - **expiryDate** string<date-time> — mm/dd/yyyy, --:-- -
  - **isPayByLink** boolean
  - **isJudoAccept** boolean
  - **successUrl** string
  - **cancelUrl** string
  - **emailAddress** string
  - **mobileNumber** string
  - **phoneCountryCode** string — 44
  - **threeDSecure** object
  - **hideBillingInfo** boolean
  - **hideReviewInfo** boolean
  - **disableNetworkTokenisation** boolean
  - **primaryAccountDetails** object

- **allowIncrement** boolean

Responses

- 200 Successful response
  - paymentSessionResponse
    - **expiryDate** string<date-time>
    - **reference** string
- 400 Bad request (Api-Version header missing, or request body attributes invalid)
- 401 Unauthorized
  - errorResponse
    - **requestId** string
    - **message** string
    - **code** integer
    - **category** integer
- 403 Forbidden if token used does not have required permission
  - errorResponse
    - **requestId** string
    - **message** string
    - **code** integer
    - **category** integer

```
curl --request POST \
     --url https://api-sandbox.judopay.com/paymentsession \
     --header 'accept: application/json' \
     --header 'content-type: application/json' \
     --header 'api-version: 6.23' \
     --data '{
     "judoId": "100100100",
     "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
     "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
     "yourPaymentMetaData": "{\"internalLocationRef\":\"Example\",\"internalId\":99}",
     "currency": "GBP",
     "cardAddress": {
     "address1": "CardHolder House",
     "address2": "1 CardHolder Street",
     "town": "CardHolder Town",
     "postCode": "AB1 2CD",
     "state": "FL",
     "countryCode": "826"
     },
     "phoneCountryCode": "44",
     "threeDSecure": {
     "challengeRequestIndicator": "noPreference"
     },
     "hideBillingInfo": true,
     "hideReviewInfo": true,
     "primaryAccountDetails": {
     "name": "Doe",
     "accountNumber": "12345678",
     "dateOfBirth": "1980-01-31",
     "postCode": "AB1 2CD"
     }
     }'
```

```
// Successful response
{
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```

27.6.2. Cancel an open payment session

PUT https://api-sandbox.judopay.com/paymentsession/{reference}/canc Try it

Sandbox environment

Update the status of an Open payment session to Cancelled, preventing it from being used in the future.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Path Parameters

reference string *

Header Parameters

Api-Version string *

Body Parameters

body object *

Responses

200 Successful response

paymentSessionCancelResponse

reference string

status string

400 Bad request (Api-Version header missing, or payment session is not in Open status)

errorResponse

- requestId string
- message string
- code integer
- category integer

401 Unauthorized

errorResponse

- requestId string
- message string
- code integer
- category integer

403 Forbidden if token used does not have required permission

errorResponse

- requestId string
- message string
- code integer
- category integer

404 Payment session reference not found

errorResponse

- requestId string
- message string
- code integer
- category integer

```
curl --request PUT \
     --url https://api-sandbox.judopay.com/paymentsession/{reference}/cancel \
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg",
  "status": "Cancelled"
}
```

27.6.3. Retrieve details of a completed payment session

GET https://api-sandbox.judopay.com/transactions/{receiptId}/webpaym Try it

Sandbox environment

Retrieve details of a completed payment session matching the given receiptId.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Path Parameters

> **receiptId** string *

Header Parameters

> **Api-Version** string *

Responses

v ● 200 Successful response

paymentSessionHistoricResponse

- amount number<float>
- cardAddress object — All Of: cardAddressRes...
- clientIpAddress string
- clientUserAgent string
- companyName string
- currency string
- expiryDate string<date-time>
- judoId string
- paymentCancelUrl string
- paymentSuccessUrl string
- reference string
- allowedCardTyes integer[]
- response object
- status string<Open | Success | Expired …
- transactionType string<Payment | PreAuth…
- yourConsumerReference string
- yourPaymentMetaData object
- yourPaymentReference string
- receipt transactionReceiptHistoricRespo…
- portalUserRecId integer
- webPaymentOperation integer

- isPayByLink boolean
- isJudoAccept boolean
- isThreeDSecureTwo boolean
- mobileNumber string
- phoneCountryCode string
- emailAddress string
- noOfAuthAttempts integer
- shortReference string

- 400 Bad request (Api-Version header missing)
- 401 Unauthorized

errorResponse

- requestId string
- message string
- code integer
- category integer

- 403 Forbidden if token used does not have required permission

errorResponse

- requestId string
- message string
- code integer
- category integer

- 404 ReceiptId not found, or not associated with a payment session, or doesn't match authentication details.

```
curl --request GET \
     --url https://api-sandbox.judopay.com/transactions/{receiptId}/webpayment \
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "amount": 10.99,
  "cardAddress": {},
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
  "paymentCancelUrl": "https://my.web.site/cancel",
  "paymentSuccessUrl": "https://my.web.site/success",
  "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw",
  "allowedCardTyes": [
    0
  ],
  "response": {
    "postUrl": "https://pay-sandbox.judopay.com/v2",
    "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
  },
  "status": "Success",
  "transactionType": "PreAuth",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentMetaData": {},
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "receipt": {
    "receiptId": "1001131610340495360",
    "originalReceiptId": "1001124307998347264",
    "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
    "type": "PreAuth",
    "createdAt": "2025-02-05T16:28:32.8596+00:00",
    "result": "Success",
    "message": "AuthCode: 123456",
    "judoId": 100100100,
    "merchantName": "Test Merchant",
    "appearsOnStatementAs": "APL*/TestMerchant       ",
    "originalAmount": "10.99",
    "amountCollected": "1.99",
    "netAmount": "1.00",
    "amount": "1.00",
    "currency": "GBP",
    "recurringPaymentType": "MIT",
    "acquirerTransactionId": "33666277256892153705",
```

```
"externalBankResponseCode": "A",
"authCode": "123456",
"postCodeCheckResult": "Passed",
"walletType": 1,
"acquirer": "MyBank",
"webPaymentReference": "5QcAAAMAAAASAAAADAAAAHaKm0p0p-Ew6VrBhPxjxvMLinJOMZEJWG_K7kJ5
"noOfAuthAttempts": 1,
"paymentNetworkTransactionId": "123456789012345",
"allowIncrement": true,
"isIncrementalAuth": true,
"disableNetworkTokenisation": true,
"cardDetails": {
  "cardLastfour": "1234",
  "endDate": "1225",
  "cardToken": "Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr",
  "cardType": 2,
  "startDate": "0121",
  "cardScheme": "Mastercard",
  "cardFunding": "Credit",
```

27.6.4. Return a list of payment sessions

GET https://api-sandbox.judopay.com/webpayments Try it

Sandbox environment

Return a list of payment sessions matching the given query.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Header Parameters

- **Api-Version** string *

Query Parameters

- **pageSize** integer — 10
- **offset** integer
- **sort** string — time-descending

Responses

- 200 Successful response
  - listPaymentSessionsResponse
    - **resultCount** number
    - **pageSize** integer
    - **offset** integer
    - **results** paymentSessionHistoricListRespon…

- **400** Bad request (Api-Version header missing)
- **401** Unauthorized
  - errorResponse
    - **requestId** string
    - **message** string
    - **code** integer
    - **category** integer
- **403** Forbidden if token used does not have required permission
  - errorResponse
    - **requestId** string
    - **message** string
    - **code** integer
    - **category** integer

```
curl --request GET \
     --url 'https://api-sandbox.judopay.com/webpayments?pageSize=10&offset=integer&sort=
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "resultCount": 135,
  "pageSize": 10,
  "offset": 0,
  "results": [
    {
      "amount": 10.99,
      "cardAddress": {},
      "clientIpAddress": "1.2.3.4",
      "clientUserAgent": "Mozilla/5.0",
      "companyName": "Test Merchant",
      "currency": "GBP",
      "expiryDate": "2026-02-05T16:28:32.8596+00:00",
      "judoId": "100100100",
      "paymentCancelUrl": "https://my.web.site/cancel",
      "paymentSuccessUrl": "https://my.web.site/success",
      "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqy
      "allowedCardTyes": [
        0
      ],
      "status": "Open",
      "transactionType": "PreAuth",
      "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
      "yourPaymentMetaData": {},
      "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
      "webPaymentOperation": 0,
      "isPayByLink": true,
      "isJudoAccept": false,
      "tradingName": "Test Merchant"
    }
  ]
}
```

27.6.5. Retrieve the details of a single payment session

GET https://api-sandbox.judopay.com/webpayments/{reference} Try it

Sandbox environment

Return the details of a single payment session with the given reference.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Path Parameters

> reference string *

Header Parameters

> Api-Version string *

Responses

200 Successful response

paymentSessionHistoricResponse

- amount number<float>
- cardAddress object — All Of: cardAddressRes...
- clientIpAddress string
- clientUserAgent string
- companyName string
- currency string
- expiryDate string<date-time>
- judoId string
- paymentCancelUrl string
- paymentSuccessUrl string
- reference string
- allowedCardTyes integer[]
- response object
- status string<Open | Success | Expired …
- transactionType string<Payment | PreAuth…
- yourConsumerReference string
- yourPaymentMetaData object
- yourPaymentReference string
- receipt transactionReceiptHistoricRespo…
- portalUserRecId integer
- webPaymentOperation integer

- isPayByLink boolean
- isJudoAccept boolean
- isThreeDSecureTwo boolean
- mobileNumber string
- phoneCountryCode string
- emailAddress string
- noOfAuthAttempts integer
- shortReference string

● 400 Bad request (Api-Version header missing)

● 401 Unauthorized

errorResponse

- requestId string
- message string
- code integer
- category integer

● 403 Forbidden if token used does not have required permission

errorResponse

- requestId string
- message string
- code integer
- category integer

● 404 Reference not found, or doesn't match authentication details.

```
curl --request GET \
     --url https://api-sandbox.judopay.com/webpayments/{reference} \
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "amount": 10.99,
  "cardAddress": {},
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
  "paymentCancelUrl": "https://my.web.site/cancel",
  "paymentSuccessUrl": "https://my.web.site/success",
  "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw",
  "allowedCardTyes": [
    0
  ],
  "response": {
    "postUrl": "https://pay-sandbox.judopay.com/v2",
    "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
  },
  "status": "Success",
  "transactionType": "PreAuth",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentMetaData": {},
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "receipt": {
    "receiptId": "1001131610340495360",
    "originalReceiptId": "1001124307998347264",
    "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
    "type": "PreAuth",
    "createdAt": "2025-02-05T16:28:32.8596+00:00",
    "result": "Success",
    "message": "AuthCode: 123456",
    "judoId": 100100100,
    "merchantName": "Test Merchant",
    "appearsOnStatementAs": "APL*/TestMerchant       ",
    "originalAmount": "10.99",
    "amountCollected": "1.99",
    "netAmount": "1.00",
    "amount": "1.00",
    "currency": "GBP",
    "recurringPaymentType": "MIT",
    "acquirerTransactionId": "33666277256892153705",
```

```
"externalBankResponseCode": "A",
"authCode": "123456",
"postCodeCheckResult": "Passed",
"walletType": 1,
"acquirer": "MyBank",
"webPaymentReference": "5QcAAAMAAAASAAAADAAAAHaKm0p0p-Ew6VrBhPxjxvMLinJOMZEJWG_K7kJ5
"noOfAuthAttempts": 1,
"paymentNetworkTransactionId": "123456789012345",
"allowIncrement": true,
"isIncrementalAuth": true,
"disableNetworkTokenisation": true,
"cardDetails": {
  "cardLastfour": "1234",
  "endDate": "1225",
  "cardToken": "Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr",
  "cardType": 2,
  "startDate": "0121",
  "cardScheme": "Mastercard",
  "cardFunding": "Credit",
```

27.6.6. Retrieve the details of a single payment session for a payment

GET https://api-sandbox.judopay.com/webpayments/payments/{referenc Try it

Sandbox environment

Return the details of a single payment session associated with a payment with the given reference.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Path Parameters

> reference string *

Header Parameters

> Api-Version string *

Responses

200 Successful response

paymentSessionHistoricResponse

- amount number<float>
- cardAddress object — All Of: cardAddressRes...
- clientIpAddress string
- clientUserAgent string
- companyName string
- currency string
- expiryDate string<date-time>
- judoId string
- paymentCancelUrl string
- paymentSuccessUrl string
- reference string
- allowedCardTyes integer[]
- response object
- status string<Open | Success | Expired …
- transactionType string<Payment | PreAuth…
- yourConsumerReference string
- yourPaymentMetaData object
- yourPaymentReference string
- receipt transactionReceiptHistoricRespo…
- portalUserRecId integer
- webPaymentOperation integer

- isPayByLink boolean
- isJudoAccept boolean
- isThreeDSecureTwo boolean
- mobileNumber string
- phoneCountryCode string
- emailAddress string
- noOfAuthAttempts integer
- shortReference string

● 400 Bad request (Api-Version header missing)

● 401 Unauthorized

errorResponse

- requestId string
- message string
- code integer
- category integer

● 403 Forbidden if token used does not have required permission

errorResponse

- requestId string
- message string
- code integer
- category integer

● 404 Reference not found, or doesn't match authentication details, or not for a payment

```
curl --request GET \
     --url https://api-sandbox.judopay.com/webpayments/payments/{reference} \
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "amount": 10.99,
  "cardAddress": {},
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
  "paymentCancelUrl": "https://my.web.site/cancel",
  "paymentSuccessUrl": "https://my.web.site/success",
  "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw",
  "allowedCardTyes": [
    0
  ],
  "response": {
    "postUrl": "https://pay-sandbox.judopay.com/v2",
    "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
  },
  "status": "Success",
  "transactionType": "PreAuth",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentMetaData": {},
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "receipt": {
    "receiptId": "1001131610340495360",
    "originalReceiptId": "1001124307998347264",
    "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
    "type": "PreAuth",
    "createdAt": "2025-02-05T16:28:32.8596+00:00",
    "result": "Success",
    "message": "AuthCode: 123456",
    "judoId": 100100100,
    "merchantName": "Test Merchant",
    "appearsOnStatementAs": "APL*/TestMerchant       ",
    "originalAmount": "10.99",
    "amountCollected": "1.99",
    "netAmount": "1.00",
    "amount": "1.00",
    "currency": "GBP",
    "recurringPaymentType": "MIT",
    "acquirerTransactionId": "33666277256892153705",
```

```
"externalBankResponseCode": "A",
"authCode": "123456",
"postCodeCheckResult": "Passed",
"walletType": 1,
"acquirer": "MyBank",
"webPaymentReference": "5QcAAAMAAAASAAAADAAAAHaKm0p0p-Ew6VrBhPxjxvMLinJOMZEJWG_K7kJ5
"noOfAuthAttempts": 1,
"paymentNetworkTransactionId": "123456789012345",
"allowIncrement": true,
"isIncrementalAuth": true,
"disableNetworkTokenisation": true,
"cardDetails": {
  "cardLastfour": "1234",
  "endDate": "1225",
  "cardToken": "Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr",
  "cardType": 2,
  "startDate": "0121",
  "cardScheme": "Mastercard",
  "cardFunding": "Credit",
```

27.6.7. Retrieve the details of a single payment session for a preauth

GET https://api-sandbox.judopay.com/webpayments/preauths/{reference

Try it

Sandbox environment

Return the details of a single payment session associated with a pre-authorisation with the given reference.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Path Parameters

> reference string *

Header Parameters

> Api-Version string *

Responses

200 Successful response

paymentSessionHistoricResponse

- amount number<float>
- cardAddress object — All Of: cardAddressRes...
- clientIpAddress string
- clientUserAgent string
- companyName string
- currency string
- expiryDate string<date-time>
- judoId string
- paymentCancelUrl string
- paymentSuccessUrl string
- reference string
- allowedCardTyes integer[]
- response object
- status string<Open | Success | Expired …
- transactionType string<Payment | PreAuth…
- yourConsumerReference string
- yourPaymentMetaData object
- yourPaymentReference string
- receipt transactionReceiptHistoricRespo…
- portalUserRecId integer
- webPaymentOperation integer

- isPayByLink boolean
- isJudoAccept boolean
- isThreeDSecureTwo boolean
- mobileNumber string
- phoneCountryCode string
- emailAddress string
- noOfAuthAttempts integer
- shortReference string

● 400 Bad request (Api-Version header missing)

● 401 Unauthorized

errorResponse

- requestId string
- message string
- code integer
- category integer

● 403 Forbidden if token used does not have required permission

errorResponse

- requestId string
- message string
- code integer
- category integer

● 404 Reference not found, or doesn't match authentication details, or not for a preauth

```
curl --request GET \
     --url https://api-sandbox.judopay.com/webpayments/preauths/{reference} \
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "amount": 10.99,
  "cardAddress": {},
  "clientIpAddress": "1.2.3.4",
  "clientUserAgent": "Mozilla/5.0",
  "companyName": "Test Merchant",
  "currency": "GBP",
  "expiryDate": "2026-02-05T16:28:32.8596+00:00",
  "judoId": "100100100",
  "paymentCancelUrl": "https://my.web.site/cancel",
  "paymentSuccessUrl": "https://my.web.site/success",
  "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw",
  "allowedCardTyes": [
    0
  ],
  "response": {
    "postUrl": "https://pay-sandbox.judopay.com/v2",
    "reference": "5QcAAAIAAAAMAAAADwAAAPwlgcQCLU6sLrNhi8kt1vEoVLwiFJc2EKTqT7JXb2xeQRqyYw
  },
  "status": "Success",
  "transactionType": "PreAuth",
  "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
  "yourPaymentMetaData": {},
  "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
  "receipt": {
    "receiptId": "1001131610340495360",
    "originalReceiptId": "1001124307998347264",
    "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
    "type": "PreAuth",
    "createdAt": "2025-02-05T16:28:32.8596+00:00",
    "result": "Success",
    "message": "AuthCode: 123456",
    "judoId": 100100100,
    "merchantName": "Test Merchant",
    "appearsOnStatementAs": "APL*/TestMerchant       ",
    "originalAmount": "10.99",
    "amountCollected": "1.99",
    "netAmount": "1.00",
    "amount": "1.00",
    "currency": "GBP",
    "recurringPaymentType": "MIT",
    "acquirerTransactionId": "33666277256892153705",
```

```
"externalBankResponseCode": "A",
"authCode": "123456",
"postCodeCheckResult": "Passed",
"walletType": 1,
"acquirer": "MyBank",
"webPaymentReference": "5QcAAAMAAAASAAAADAAAAHaKm0p0p-Ew6VrBhPxjxvMLinJOMZEJWG_K7kJ5
"noOfAuthAttempts": 1,
"paymentNetworkTransactionId": "123456789012345",
"allowIncrement": true,
"isIncrementalAuth": true,
"disableNetworkTokenisation": true,
"cardDetails": {
  "cardLastfour": "1234",
  "endDate": "1225",
  "cardToken": "Ck3AeNlBfjvs9d61MNZiG0gtCvijqvKr",
  "cardType": 2,
  "startDate": "0121",
  "cardScheme": "Mastercard",
  "cardFunding": "Credit",
```

27.6.8. Create a session for check card

POST https://api-sandbox.judopay.com/webpayments/checkcard Try it

Sandbox environment

Create a session for a check card operation (zero amount pre-authorisation). transactionType will be CHECKCARD, webPaymentOperation will be 1.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Header Parameters

> **Api-Version** string *

Body Parameters

- **body** paymentSessionRequest * — All Of: paymentSession...
  - **judoId** string *
  - **yourConsumerReference** string *
  - **yourPaymentReference** string *
  - **yourPaymentMetaData** object
  - **currency** string
  - **amount** number<float>
  - **cardAddress** cardAddressWebPaymentRequest — All Of: cardAddressReq...
  - **expiryDate** string<date-time> — mm/dd/yyyy, --:--
  - **isPayByLink** boolean
  - **isJudoAccept** boolean
  - **successUrl** string
  - **cancelUrl** string
  - **emailAddress** string
  - **mobileNumber** string
  - **phoneCountryCode** string — 44
  - **threeDSecure** object
  - **hideBillingInfo** boolean
  - **hideReviewInfo** boolean
  - **disableNetworkTokenisation** boolean
  - **primaryAccountDetails** object

Responses

- 200 Successful response
  - createPaymentSessionResponse
    - payByLinkUrl string
    - postUrl string
    - reference string
- 400 Bad request (Api-Version header missing, or request body attributes invalid)
- 401 Unauthorized
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer
- 403 Forbidden if token used does not have required permission
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer

```
curl --request POST \
     --url https://api-sandbox.judopay.com/webpayments/checkcard \
     --header 'accept: application/json' \
     --header 'content-type: application/json' \
     --header 'api-version: 6.23' \
     --data '{
     "judoId": "100100100",
     "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
     "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
     "cardAddress": {
     "postCode": "AB1 2CD"
     },
     "phoneCountryCode": "44",
     "threeDSecure": {
     "challengeRequestIndicator": "noPreference"
     },
     "hideBillingInfo": true,
     "hideReviewInfo": true,
     "primaryAccountDetails": {
     "name": "Doe",
     "accountNumber": "12345678",
     "dateOfBirth": "1980-01-31",
     "postCode": "AB1 2CD"
     }
     }'
```

```
// Successful response
{
  "payByLinkUrl": "https://pay-sandbox.judopay.com/ABC123",
  "postUrl": "https://pay-sandbox.judopay.com/v2",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```

27.6.9. Create a session for a payment

POST https://api-sandbox.judopay.com/webpayments/payments Try it

Sandbox environment

Create a session for a payment operation. transactionType will be Payment, webPaymentOperation will be 0.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Header Parameters

> Api-Version string *

Body Parameters

- **body** paymentSessionRequest * — All Of: paymentSession...
  - **judoId** string *
  - **yourConsumerReference** string *
  - **yourPaymentReference** string *
  - **yourPaymentMetaData** object
  - **currency** string
  - **amount** number<float>
  - **cardAddress** cardAddressWebPaymentRequest — All Of: cardAddressReq...
  - **expiryDate** string<date-time> — mm/dd/yyyy, --:-- -
  - **isPayByLink** boolean
  - **isJudoAccept** boolean
  - **successUrl** string
  - **cancelUrl** string
  - **emailAddress** string
  - **mobileNumber** string
  - **phoneCountryCode** string — 44
  - **threeDSecure** object
  - **hideBillingInfo** boolean
  - **hideReviewInfo** boolean
  - **disableNetworkTokenisation** boolean
  - **primaryAccountDetails** object

Responses

- 200 Successful response
  - createPaymentSessionResponse
    - payByLinkUrl string
    - postUrl string
    - reference string
- 400 Bad request (Api-Version header missing, or request body attributes invalid)
- 401 Unauthorized
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer
- 403 Forbidden if token used does not have required permission
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer

```
curl --request POST \
     --url https://api-sandbox.judopay.com/webpayments/payments \
     --header 'accept: application/json' \
     --header 'content-type: application/json' \
     --header 'api-version: 6.23' \
     --data '{
     "judoId": "100100100",
     "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
     "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
     "cardAddress": {
     "postCode": "AB1 2CD"
     },
     "phoneCountryCode": "44",
     "threeDSecure": {
     "challengeRequestIndicator": "noPreference"
     },
     "hideBillingInfo": true,
     "hideReviewInfo": true,
     "primaryAccountDetails": {
     "name": "Doe",
     "accountNumber": "12345678",
     "dateOfBirth": "1980-01-31",
     "postCode": "AB1 2CD"
     }
     }'
```

```
// Successful response
{
  "payByLinkUrl": "https://pay-sandbox.judopay.com/ABC123",
  "postUrl": "https://pay-sandbox.judopay.com/v2",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```

27.6.10. Create a session for a preauth

POST https://api-sandbox.judopay.com/webpayments/preauths Try it

Sandbox environment

Create a session for a pre-authorisation. transactionType will be PREAUTH, webPaymentOperation will be 0.

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Header Parameters

> Api-Version string *

Body Parameters

- **body** paymentSessionRequestForPreAuths * — All Of: paymentSession...
  - **judoId** string *
  - **yourConsumerReference** string *
  - **yourPaymentReference** string *
  - **yourPaymentMetaData** object *
  - **currency** string *
  - **amount** number<float>
  - **cardAddress** cardAddressWebPaymentRequest — All Of: cardAddressReq...
  - **expiryDate** string<date-time> — mm/dd/yyyy, --:-- -
  - **isPayByLink** boolean
  - **isJudoAccept** boolean
  - **successUrl** string
  - **cancelUrl** string
  - **emailAddress** string
  - **mobileNumber** string
  - **phoneCountryCode** string — 44
  - **threeDSecure** object
  - **hideBillingInfo** boolean
  - **hideReviewInfo** boolean
  - **disableNetworkTokenisation** boolean
  - **primaryAccountDetails** object

- allowIncrement boolean

Responses

- 200 Successful response
  - createPaymentSessionResponse
    - payByLinkUrl string
    - postUrl string
    - reference string
- 400 Bad request (Api-Version header missing, or request body attributes invalid)
- 401 Unauthorized
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer
- 403 Forbidden if token used does not have required permission
  - errorResponse
    - requestId string
    - message string
    - code integer
    - category integer

```
curl --request POST \
     --url https://api-sandbox.judopay.com/webpayments/preauths \
     --header 'accept: application/json' \
     --header 'content-type: application/json' \
     --header 'api-version: 6.23' \
     --data '{
     "judoId": "100100100",
     "yourConsumerReference": "2b45fd3f-cee5-4e7e-874f-28051db65408",
     "yourPaymentReference": "6482c678-cad3-4efd-b081-aeae7a89a134",
     "yourPaymentMetaData": "{\"internalLocationRef\":\"Example\",\"internalId\":99}",
     "currency": "GBP",
     "cardAddress": {
     "address1": "CardHolder House",
     "address2": "1 CardHolder Street",
     "town": "CardHolder Town",
     "postCode": "AB1 2CD",
     "state": "FL",
     "countryCode": "826"
     },
     "phoneCountryCode": "44",
     "threeDSecure": {
     "challengeRequestIndicator": "noPreference"
     },
     "hideBillingInfo": true,
     "hideReviewInfo": true,
     "primaryAccountDetails": {
     "name": "Doe",
     "accountNumber": "12345678",
     "dateOfBirth": "1980-01-31",
     "postCode": "AB1 2CD"
     }
     }'
```

```
// Successful response
{
  "payByLinkUrl": "https://pay-sandbox.judopay.com/ABC123",
  "postUrl": "https://pay-sandbox.judopay.com/v2",
  "reference": "5QcAAAQAAAAPAAAACAAAABtGgvhBrF9BHTN7nqn1e0J4hVVmi-y27dGPjWBMtls3Gj_XDg"
}
```

27.7. Other

27.7.1. /info

GET https://api-sandbox.judopay.com/info Try it

Sandbox environment

Obtain information about service, including latest supported Api-Version

Credentials

HTTP Basic TokenSecretAuth

This is the typical authorisation scenario to use, specify your token in Username and secret in Password.

Sent as Base64 encoded string representing token

in Authorization header.

username : password

Header Parameters

Api-Version string *

Responses

200 Successful response

infoResponse

version string

400 Bad request (Api-Version header missing)

401 Unauthorized

errorResponse

- requestId string
- message string
- code integer
- category integer

```curl
curl --request GET \
     --url https://api-sandbox.judopay.com/info \
     --header 'accept: application/json' \
     --header 'api-version: 6.23'
```

```
// Successful response
{
  "version": "6.23.0.0"
}
```

27.8. Models

27.8.1. Model: allowIncrementRequest

Model Parameters

- allowIncrementRequest
  - allowIncrement boolean


```
{
  "allowIncrement": false
}
```

27.8.2. Model: networkTokenRequest

Model Parameters

networkTokenRequest

- disableNetworkTokenisation boolean


```
{
  "disableNetworkTokenisation": false
}
```

27.8.3. Model: applePayRequest

Model Parameters

applePayRequest

- pkPayment object *
- cardAddress cardAddressRequestAttributes — All Of cardAddressCom...


```
{
  "pkPayment": {
    "token": {
      "paymentMethod": {
        "displayName": "",
        "network": "",
        "type": ""
      },
      "paymentData": {
        "data": "",
        "signature": "",
        "header": {
          "publicKeyHash": "",
          "ephemeralPublicKey": "",
          "transactionId": ""
        },
        "version": ""
      }
    }
  },
  "cardAddress": {}
}
```

27.8.4. Model: cardAddressCommonAttributes

Model Parameters

- cardAddressCommonAttributes
  - address1 string
  - address2 string
  - town string
  - postCode string *
  - state string

```
{
  "address1": "",
  "address2": "",
  "town": "",
  "postCode": "",
  "state": ""
}
```

Card holder address.

27.8.5. Model: cardAddressRequestAttributes

Model Parameters

cardAddressRequestAttributes

All Of
cardAddressCom...

- address1 string
- address2 string
- town string
- postCode string *
- state string
- countryCode string


```
{
  "address1": "",
  "address2": "",
  "town": "",
  "postCode": "",
  "state": "",
  "countryCode": ""
}
```

27.8.6. Model: cardAddressRequest

27.8.7. Model: cardAddressResponseAttributes

27.8.8. Model: cardAddressWebPaymentRequest

27.8.9. Model: cardDetailsCv2NotRequiredRequest

27.8.10. Model: cardDetailsCv2RequiredRequest

27.8.11. Model: cardDetailsStartDateRequest

27.8.12. Model: cardDetailsResponse

27.8.13. Model: cardDetailsListResponse

27.8.14. Model: cardTokenPreAuthPaymentRequest

27.8.15. Model: challengeRequestIndicatorRequest

27.8.16. Model: checkCardRequestPan

Model Parameters

checkCardRequestPan

All Of
cardDetailsCv2N...

- cardNumber string *
- cv2 string
- expiryDate string *
- startDate string
- yourConsumerReference string *
- yourPaymentReference string
- clientDetails object
- judoId string
- currency string
- initialRecurringPayment boolean
- disableNetworkTokenisation boolean
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string
- threeDSecure object
- threeDSecureMpi object
- cardAddress cardAddressRequestAttributes

All Of
cardAddressCom...

- primaryAccountDetails object

27.8.17. Model: collectionRequest

Model Parameters

collectionRequest

- receiptId string *
- amount number<float>
- yourPaymentMetaData object
- yourPaymentReference string *


```
{
  "receiptId": "",
  "amount": "",
  "yourPaymentMetaData": {},
  "yourPaymentReference": ""
}
```

27.8.18. Model: complete3ds2Request

Model Parameters

complete3ds2Request

All Of
object +

- cv2 string
- version string *
- primaryAccountDetails object


```
{
  "cv2": "",
  "version": "",
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.19. Model: consumerResponse

Model Parameters

consumerResponse

- yourConsumerReference string

```
{
  "yourConsumerReference": ""
}
```

27.8.20. Model: createPaymentSessionResponse

Model Parameters

createPaymentSessionResponse

- payByLinkUrl string
- postUrl string
- reference string


```
{
  "payByLinkUrl": "",
  "postUrl": "",
  "reference": ""
}
```

27.8.21. Model: deviceResponse

Model Parameters

deviceResponse

identifier string


```
{
  "identifier": ""
}
```


[Conditional] If clientDetails was provided containing a kdeviceid

27.8.22. Model: errorResponse

Model Parameters

errorResponse

- requestId string
- message string
- code integer
- category integer


```
{
  "requestId": "",
  "message": "",
  "code": 0,
  "category": 0
}
```

27.8.23. Model: googlePayRequest

Model Parameters

googlePayRequest

- **googlePayWallet** object *
- **cardAddress** cardAddressRequestAttributes — All Of: cardAddressCom...


```
{
  "googlePayWallet": {
    "token": "",
    "cardNetwork": "",
    "cardDetails": ""
  },
  "cardAddress": {}
}
```

27.8.24. Model: incrementAuthRequest

Model Parameters

incrementAuthRequest

- receiptId string *
- amount number<float> *
- yourPaymentMetaData object
- yourPaymentReference string *


```
{
  "receiptId": "",
  "amount": "",
  "yourPaymentMetaData": {},
  "yourPaymentReference": ""
}
```

27.8.25. Model: infoResponse

Model Parameters

infoResponse

- version string


```
{
  "version": ""
}
```

27.8.26. Model: listPaymentSessionsResponse

Model Parameters

- listPaymentSessionsResponse
  - resultCount number
  - pageSize integer
  - offset integer
  - results paymentSessionHistoricListResponse…

```
{
  "resultCount": 0,
  "pageSize": 0,
  "offset": 0,
  "results": [
    {
      "amount": "",
      "cardAddress": {},
      "clientIpAddress": "",
      "clientUserAgent": "",
      "companyName": "",
      "currency": "",
      "expiryDate": "",
      "judoId": "",
      "paymentCancelUrl": "",
      "paymentSuccessUrl": "",
      "reference": "",
      "allowedCardTyes": [
        0
      ],
      "status": "",
      "transactionType": "",
      "yourConsumerReference": "",
      "yourPaymentMetaData": {},
      "yourPaymentReference": "",
      "webPaymentOperation": 0,
      "isPayByLink": false,
      "isJudoAccept": false,
      "tradingName": ""
    }
  ]
}
```

27.8.27. Model: listTransactionsResponse

Model Parameters

- listTransactionsResponse
  - resultCount number
  - pageSize integer
  - offset integer
  - results transactionReceiptListResponse[]
  - sort string

27.8.28. Model: paymentRequestApplePay

Model Parameters

paymentRequestApplePay — All Of: applePayRequest +

- pkPayment object *
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...
- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object

```
{
  "pkPayment": {
    "token": {
      "paymentMethod": {
        "displayName": "",
        "network": "",
        "type": ""
      },
      "paymentData": {
        "data": "",
        "signature": "",
        "header": {
          "publicKeyHash": "",
          "ephemeralPublicKey": "",
          "transactionId": ""
        },
        "version": ""
      }
    }
  },
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "webPaymentReference": "",
  "clientDetails": {
    "key": "",
    "value": ""
  },
  "initialRecurringPayment": false,
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.29. Model: paymentRequestCardToken

Model Parameters

paymentRequestCardToken — All Of: cardTokenPreAut...

- cardToken string *
- cv2 string
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...
- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- acceptHeaders string
- userAgent string
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string

- **threeDSecure** object
- **threeDSecureMpi** object
- **recurringPayment** boolean
- **recurringPaymentType** string<RECURRING | MI…
- **relatedReceiptId** string
- **relatedPaymentNetworkTransactionId** string
- **disableNetworkTokenisation** boolean

```
{
  "cardToken": "",
  "cv2": "",
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": ""
```

27.8.30. Model: paymentRequestGooglePay

Model Parameters

paymentRequestGooglePay

All Of
googlePayReque...

- googlePayWallet object *
- cardAddress cardAddressRequestAttributes

All Of
cardAddressCom...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- acceptHeaders string
- userAgent string
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string
- threeDSecure object

- **threeDSecureMpi** `object`

```
{
  "googlePayWallet": {
    "token": "",
    "cardNetwork": "",
    "cardDetails": ""
  },
  "cardAddress": {},
  "judoId": ""
```

27.8.31. Model: paymentRequestPan

Model Parameters

paymentRequestPan

All Of
cardDetailsCv2N...

- cardNumber string *
- cv2 string
- expiryDate string *
- startDate string
- cardAddress cardAddressRequestAttributes

All Of
cardAddressCom...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- acceptHeaders string
- userAgent string
- cardHolderName string
- emailAddress string
- shippingAddress object

- **mobileNumber** string
- **phoneCountryCode** string
- **threeDSecure** object
- **threeDSecureMpi** object
- **recurringPayment** boolean
- **recurringPaymentType** string<RECURRING | MI…
- **relatedReceiptId** string
- **relatedPaymentNetworkTransactionId** string
- **disableNetworkTokenisation** boolean

```
{
  "cardNumber": "",
  "cv2": "",
  "expiryDate": "",
  "startDate": "",
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": ""
```

27.8.32. Model: paymentSessionHistoricResponse

Model Parameters

paymentSessionHistoricResponse

- amount number<float>
- cardAddress object — All Of: cardAddressRes...
- clientIpAddress string
- clientUserAgent string
- companyName string
- currency string
- expiryDate string<date-time>
- judoId string
- paymentCancelUrl string
- paymentSuccessUrl string
- reference string
- allowedCardTyes integer[]
- response object
- status string<Open | Success | Expired | …
- transactionType string<Payment | PreAuth |…
- yourConsumerReference string
- yourPaymentMetaData object
- yourPaymentReference string
- receipt transactionReceiptHistoricResponse
- portalUserRecId integer

- **webPaymentOperation** integer
- **isPayByLink** boolean
- **isJudoAccept** boolean
- **isThreeDSecureTwo** boolean
- **mobileNumber** string
- **phoneCountryCode** string
- **emailAddress** string
- **noOfAuthAttempts** integer
- **shortReference** string

```
{
  "amount": "",
  "cardAddress": {},
  "clientIpAddress": "",
  "clientUserAgent": "",
  "companyName": "",
  "currency": "",
  "expiryDate": "",
  "judoId": "",
  "paymentCancelUrl": "",
  "paymentSuccessUrl": "",
  "reference": "",
  "allowedCardTyes": [
    0
  ],
  "response": {
    "postUrl": "",
    "reference": ""
  },
  "status": "",
  "transactionType": "",
  "yourConsumerReference": "",
  "yourPaymentMetaData": {},
  "yourPaymentReference": "",
  "receipt": {
    "receiptId": "",
    "originalReceiptId": "",
    "yourPaymentReference": "",
    "type": "",
    "createdAt": "",
    "result": "",
    "message": "",
    "judoId": 0,
    "merchantName": "",
    "appearsOnStatementAs": "",
    "originalAmount": "",
    "amountCollected": "",
    "netAmount": "",
    "amount": "",
    "currency": "",
    "recurringPaymentType": "",
    "acquirerTransactionId": "",
    "externalBankResponseCode": "",
```

```
"authCode": "",
"postCodeCheckResult": "",
"walletType": 0,
"acquirer": "",
"webPaymentReference": "",
"noOfAuthAttempts": 0,
"paymentNetworkTransactionId": "",
"allowIncrement": false,
"isIncrementalAuth": false,
"disableNetworkTokenisation": false,
"cardDetails": {
  "cardLastfour": "",
  "endDate": "",
  "cardToken": "",
  "cardType": 0,
  "startDate": "",
  "cardScheme": "",
  "cardFunding": "",
```

27.8.33. Model: paymentSessionHistoricListResponse

Model Parameters

paymentSessionHistoricListResponse

- amount number<float>
- cardAddress object — All Of: cardAddressRes...
- clientIpAddress string
- clientUserAgent string
- companyName string
- currency string
- expiryDate string<date-time>
- judoId string
- paymentCancelUrl string
- paymentSuccessUrl string
- reference string
- allowedCardTyes integer[]
- status string<Open | Success | Expired | …
- transactionType string<Payment | PreAuth |…
- yourConsumerReference string
- yourPaymentMetaData object
- yourPaymentReference string
- webPaymentOperation integer
- isPayByLink boolean
- isJudoAccept boolean

> **tradingName** string


```
{
  "amount": "",
  "cardAddress": {},
  "clientIpAddress": "",
  "clientUserAgent": "",
  "companyName": "",
  "currency": "",
  "expiryDate": "",
  "judoId": "",
  "paymentCancelUrl": "",
  "paymentSuccessUrl": "",
  "reference": "",
  "allowedCardTyes": [
    0
  ],
  "status": "",
  "transactionType": "",
  "yourConsumerReference": "",
  "yourPaymentMetaData": {},
  "yourPaymentReference": "",
  "webPaymentOperation": 0,
  "isPayByLink": false,
  "isJudoAccept": false,
  "tradingName": ""
}
```

27.8.34. Model: paymentSessionRequest

Model Parameters

paymentSessionRequest — All Of: paymentSession...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object
- currency string
- amount number<float>
- cardAddress cardAddressWebPaymentRequest — All Of: cardAddressReq...
- expiryDate string<date-time>
- isPayByLink boolean
- isJudoAccept boolean
- successUrl string
- cancelUrl string
- emailAddress string
- mobileNumber string
- phoneCountryCode string
- threeDSecure object
- hideBillingInfo boolean
- hideReviewInfo boolean
- disableNetworkTokenisation boolean
- primaryAccountDetails object

```
{
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "cardAddress": {},
  "expiryDate": "",
  "isPayByLink": false,
  "isJudoAccept": false,
  "successUrl": "",
  "cancelUrl": "",
  "emailAddress": "",
  "mobileNumber": "",
  "phoneCountryCode": "44",
  "threeDSecure": {
    "challengeRequestIndicator": "noPreference",
    "scaExemption": ""
  },
  "hideBillingInfo": true,
  "hideReviewInfo": true,
  "disableNetworkTokenisation": false,
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.35. Model: paymentSessionRequestForPreAuths

Model Parameters

paymentSessionRequestForPreAuths — All Of: paymentSession...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float>
- cardAddress cardAddressWebPaymentRequest — All Of: cardAddressReq...
- expiryDate string<date-time>
- isPayByLink boolean
- isJudoAccept boolean
- successUrl string
- cancelUrl string
- emailAddress string
- mobileNumber string
- phoneCountryCode string
- threeDSecure object
- hideBillingInfo boolean
- hideReviewInfo boolean
- disableNetworkTokenisation boolean
- primaryAccountDetails object

> **allowIncrement** boolean


```
{
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "cardAddress": {},
  "expiryDate": "",
  "isPayByLink": false,
  "isJudoAccept": false,
  "successUrl": "",
  "cancelUrl": "",
  "emailAddress": "",
  "mobileNumber": "",
  "phoneCountryCode": "44",
  "threeDSecure": {
    "challengeRequestIndicator": "noPreference",
    "scaExemption": ""
  },
  "hideBillingInfo": true,
  "hideReviewInfo": true,
  "disableNetworkTokenisation": false,
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  },
  "allowIncrement": false
}
```

27.8.36. Model: paymentSessionRequestCommon

Model Parameters

paymentSessionRequestCommon

All Of
object +

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object
- currency string


```
{
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": ""
}
```

27.8.37. Model: paymentSessionResponse

Model Parameters

paymentSessionResponse

- expiryDate string<date-time>
- reference string


```
{
  "expiryDate": "",
  "reference": ""
}
```

27.8.38. Model: paymentSessionCancelResponse

Model Parameters

- paymentSessionCancelResponse
  - reference string
  - status string


```
{
  "reference": "",
  "status": ""
}
```

27.8.39. Model: preAuthPaymentRequestCommon

Model Parameters

preAuthPaymentRequestCommon

All Of
paymentSession...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object
- currency string
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object

```
{
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "webPaymentReference": "",
  "clientDetails": {
    "key": "",
    "value": ""
  },
  "initialRecurringPayment": false,
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.40. Model: preAuthRequestApplePay

Model Parameters

preAuthRequestApplePay — All Of: applePayRequest +

- pkPayment object *
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...
- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- allowIncrement boolean

```
{
  "pkPayment": {
    "token": {
      "paymentMethod": {
        "displayName": "",
        "network": "",
        "type": ""
      },
      "paymentData": {
        "data": "",
        "signature": "",
        "header": {
          "publicKeyHash": "",
          "ephemeralPublicKey": "",
          "transactionId": ""
        },
        "version": ""
      }
    }
  },
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "webPaymentReference": "",
  "clientDetails": {
    "key": "",
    "value": ""
  },
  "initialRecurringPayment": false,
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  },
  "allowIncrement": false
}
```

27.8.41. Model: preAuthRequestCardToken

Model Parameters

preAuthRequestCardToken — All Of: cardTokenPreAut...

- cardToken string *
- cv2 string
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...
- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- acceptHeaders string
- userAgent string
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string

- **threeDSecure** object
- **threeDSecureMpi** object
- **recurringPayment** boolean
- **recurringPaymentType** string<RECURRING | MI…
- **relatedReceiptId** string
- **relatedPaymentNetworkTransactionId** string
- **allowIncrement** boolean
- **disableNetworkTokenisation** boolean

```
{
  "cardToken": "",
  "cv2": "",
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "webPaymentReference": ""
```

27.8.42. Model: preAuthRequestGooglePay

Model Parameters

preAuthRequestGooglePay

All Of
googlePayReque...

- googlePayWallet object *
- cardAddress cardAddressRequestAttributes

All Of
cardAddressCom...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- acceptHeaders string
- userAgent string
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string
- threeDSecure object

- **threeDSecureMpi** `object`
- **allowIncrement** `boolean`

```
{
  "googlePayWallet": {
    "token": "",
    "cardNetwork": "",
    "cardDetails": ""
  },
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": ""
```

27.8.43. Model: preAuthRequestPan

Model Parameters

preAuthRequestPan

All Of
cardDetailsCv2N...

- cardNumber string *
- cv2 string
- expiryDate string *
- startDate string
- cardAddress cardAddressRequestAttributes

All Of
cardAddressCom...

- judoId string *
- yourConsumerReference string *
- yourPaymentReference string *
- yourPaymentMetaData object *
- currency string *
- amount number<float> *
- webPaymentReference string
- clientDetails object
- initialRecurringPayment boolean
- primaryAccountDetails object
- acceptHeaders string
- userAgent string
- cardHolderName string
- emailAddress string
- shippingAddress object

- **mobileNumber** string
- **phoneCountryCode** string
- **threeDSecure** object
- **threeDSecureMpi** object
- **recurringPayment** boolean
- **recurringPaymentType** string<RECURRING | MI…
- **relatedReceiptId** string
- **relatedPaymentNetworkTransactionId** string
- **allowIncrement** boolean
- **disableNetworkTokenisation** boolean

```
{
  "cardNumber": "",
  "cv2": "",
  "expiryDate": "",
  "startDate": "",
  "cardAddress": {},
  "judoId": "",
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "yourPaymentMetaData": {},
  "currency": "",
  "amount": "",
  "webPaymentReference": ""
```

27.8.44. Model: primaryAccountDetailsRequest

Model Parameters

primaryAccountDetailsRequest

- primaryAccountDetails object


```
{
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.45. Model: recurringPaymentRequest

Model Parameters

recurringPaymentRequest

- recurringPayment boolean
- recurringPaymentType string<RECURRING | MI…
- relatedReceiptId string
- relatedPaymentNetworkTransactionId string


```
{
  "recurringPayment": false,
  "recurringPaymentType": "",
  "relatedReceiptId": "",
  "relatedPaymentNetworkTransactionId": ""
}
```

27.8.46. Model: refundRequest

Model Parameters

refundRequest

- receiptId string *
- amount number<float>
- yourPaymentReference string *


```
{
  "receiptId": "",
  "amount": "",
  "yourPaymentReference": ""
}
```

27.8.47. Model: registerCardRequest

Model Parameters

registerCardRequest — All Of: object +

- yourConsumerReference string *
- yourPaymentReference string
- clientDetails object
- judoId string
- currency string
- initialRecurringPayment boolean
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string
- threeDSecure object
- threeDSecureMpi object
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...
- primaryAccountDetails object

27.8.48. Model: checkCardRequest

Model Parameters

checkCardRequest — All Of: object +

- yourConsumerReference string *
- yourPaymentReference string
- clientDetails object
- judoId string
- currency string
- initialRecurringPayment boolean
- disableNetworkTokenisation boolean
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string
- threeDSecure object
- threeDSecureMpi object
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...
- primaryAccountDetails object

```
{
  "yourConsumerReference": "",
  "yourPaymentReference": "",
  "clientDetails": {
    "key": "",
    "value": ""
  },
  "judoId": "",
  "currency": "",
  "initialRecurringPayment": false,
  "disableNetworkTokenisation": false,
  "cardHolderName": "",
  "emailAddress": "",
  "shippingAddress": {
    "address1": "",
    "address2": "",
    "town": "",
    "postCode": "",
    "countryCode": "",
    "isBillingAddress": false
  },
  "mobileNumber": "",
  "phoneCountryCode": "44",
  "threeDSecure": {
    "authenticationSource": "",
    "methodNotificationUrl": "https://api.judopay.com/order/3ds/methodNotification",
    "challengeNotificationUrl": "https://api.judopay.com/order/3ds/challengeNotification"
  },
  "threeDSecureMpi": {
    "dsTransId": "",
    "cavv": "",
    "eci": "",
    "threeDSecureVersion": ""
  },
  "cardAddress": {},
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.49. Model: registerCardRequestPan

Model Parameters

registerCardRequestPan

All Of
cardDetailsCv2N...

- cardNumber string *
- cv2 string
- expiryDate string *
- startDate string
- yourConsumerReference string *
- yourPaymentReference string
- clientDetails object
- judoId string
- currency string
- initialRecurringPayment boolean
- cardHolderName string
- emailAddress string
- shippingAddress object
- mobileNumber string
- phoneCountryCode string
- threeDSecure object
- threeDSecureMpi object
- cardAddress cardAddressRequestAttributes

All Of
cardAddressCom...

- primaryAccountDetails object

```
{
  "recordNumber": ""
```

27.8.50. Model: resume3dsRequest

Model Parameters

resume3dsRequest

All Of
object +

- cv2 string
- threeDSecure object *
- primaryAccountDetails object


```
{
  "cv2": "",
  "threeDSecure": {
    "methodCompletion": ""
  },
  "primaryAccountDetails": {
    "name": "",
    "accountNumber": "",
    "dateOfBirth": "",
    "postCode": ""
  }
}
```

27.8.51. Model: riskParametersResponse

Model Parameters

riskParametersResponse

- **postCodeCheck** string<PASSED | FAILED | U…
- **cv2Check** string<PASSED | FAILED | NOT_PR…
- **merchantSuggestion** string


```
{
  "postCodeCheck": "",
  "cv2Check": "",
  "merchantSuggestion": ""
}
```

27.8.52. Model: networkTokenisationDetailsResponse

Model Parameters

networkTokenisationDetailsResponse

- networkTokenProvisioned boolean
- networkTokenUsed boolean
- virtualPan object


```
{
  "networkTokenProvisioned": false,
  "networkTokenUsed": false,
  "virtualPan": {
    "lastFour": "",
    "expiryDate": ""
  }
}
```

27.8.53. Model: saveCardRequestCommon

Model Parameters

saveCardRequestCommon — All Of: object +

- yourConsumerReference string *
- judoId string
- currency string
- cardHolderName string
- disableNetworkTokenisation boolean
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...


```
{
  "yourConsumerReference": "",
  "judoId": "",
  "currency": "",
  "cardHolderName": "",
  "disableNetworkTokenisation": false,
  "cardAddress": {}
}
```

27.8.54. Model: saveCardRequestPan

Model Parameters

saveCardRequestPan — All Of: object +

- cardNumber string *
- expiryDate string *
- startDate string
- yourConsumerReference string *
- judoId string
- currency string
- cardHolderName string
- disableNetworkTokenisation boolean
- cardAddress cardAddressRequestAttributes — All Of: cardAddressCom...

```
{
  "cardNumber": "",
  "expiryDate": "",
  "startDate": "",
  "yourConsumerReference": "",
  "judoId": "",
  "currency": "",
  "cardHolderName": "",
  "disableNetworkTokenisation": false,
  "cardAddress": {}
}
```

27.8.55. Model: scaExemptionRequest

Model Parameters

scaExemptionRequest

Example

transactionRiskAnalysis

```
{}
```

Indicates reason why challenge may not be necessary (this may be over-ruled by the issuer). Should not be specified in the same request as challengeRequestIndicator

27.8.56. Model: threeDSecureChallengeRequest

Model Parameters

- threeDSecureChallengeRequest
  - acceptHeaders string
  - userAgent string


```
{
  "acceptHeaders": "text/html",
  "userAgent": ""
}
```

27.8.57. Model: threeDSecureCompletedResponse

Model Parameters

- threeDSecureCompletedResponse
  - attempted boolean
  - result string
  - eci string
  - challengeRequestIndicator string<NoPreferen…
  - scaExemption string<TrustedBeneficiary | …


```
{
  "attempted": false,
  "result": "",
  "eci": "",
  "challengeRequestIndicator": "",
  "scaExemption": ""
}
```


[Conditional] Only returned for transactions using 3DS

27.8.58. Model: threeDSecureTwoChallengeRequiredResp

Model Parameters

threeDSecureTwoChallengeRequiredResponse

- challengeUrl string
- cReq string
- version string
- receiptId string
- result string
- message string
- md string


```
{
  "challengeUrl": "",
  "cReq": "",
  "version": "",
  "receiptId": "",
  "result": "",
  "message": "",
  "md": ""
}
```

27.8.59. Model: threeDSecureTwoDeviceDetailsRequiredR

Model Parameters

threeDSecureTwoDeviceDetailsRequiredRespon…

- **methodUrl** string
- **version** string
- **receiptId** string
- **result** string
- **message** string
- **md** string


```
{
  "methodUrl": "",
  "version": "",
  "receiptId": "",
  "result": "",
  "message": "",
  "md": ""
}
```

27.8.60. Model: threeDSecureTwoCheckCardRequest

Model Parameters

- threeDSecureTwoCheckCardRequest
  - cardHolderName string
  - emailAddress string
  - shippingAddress object
  - mobileNumber string
  - phoneCountryCode string
  - threeDSecure object
  - threeDSecureMpi object

```
{
  "cardHolderName": "",
  "emailAddress": "",
  "shippingAddress": {
    "address1": "",
    "address2": "",
    "town": "",
    "postCode": "",
    "countryCode": "",
    "isBillingAddress": false
  },
  "mobileNumber": "",
  "phoneCountryCode": "44",
  "threeDSecure": {
    "authenticationSource": "",
    "methodNotificationUrl": "https://api.judopay.com/order/3ds/methodNotification",
    "challengeNotificationUrl": "https://api.judopay.com/order/3ds/challengeNotification"
  },
  "threeDSecureMpi": {
    "dsTransId": "",
    "cavv": "",
    "eci": "",
    "threeDSecureVersion": ""
  }
}
```

27.8.61. Model: threeDSecureTwoRequest

Model Parameters

- threeDSecureTwoRequest
  - cardHolderName string
  - emailAddress string
  - shippingAddress object
  - mobileNumber string
  - phoneCountryCode string
  - threeDSecure object
  - threeDSecureMpi object

```
{
  "cardHolderName": "",
  "emailAddress": "",
  "shippingAddress": {
    "address1": "",
    "address2": "",
    "town": "",
    "postCode": "",
    "countryCode": "",
    "isBillingAddress": false
  },
  "mobileNumber": "",
  "phoneCountryCode": "44",
  "threeDSecure": {
    "authenticationSource": "",
    "methodNotificationUrl": "https://api.judopay.com/order/3ds/methodNotification",
    "challengeNotificationUrl": "https://api.judopay.com/order/3ds/challengeNotification"
    "challengeRequestIndicator": "noPreference",
    "scaExemption": ""
  },
  "threeDSecureMpi": {
    "dsTransId": "",
    "cavv": "",
    "eci": "",
    "threeDSecureVersion": ""
  }
}
```

27.8.62. Model: transactionReceiptResponse

Model Parameters

transactionReceiptResponse

- receiptId string
- originalReceiptId string
- yourPaymentReference string
- type string<Payment | Refund | PreAuth |…
- createdAt string<date-time>
- result string
- message string
- judoId integer
- merchantName string
- appearsOnStatementAs string
- originalAmount string
- amountCollected string
- netAmount string
- amount string
- currency string
- recurringPaymentType string
- acquirerTransactionId string
- externalBankResponseCode string
- authCode string
- walletType integer

- **riskScore** integer
- **paymentNetworkTransactionId** string
- **allowIncrement** boolean
- **isIncrementalAuth** boolean
- **emailAddress** string
- **cardDetails** cardDetailsResponse
- **billingAddress** cardAddressResponseAttributes — All Of: cardAddressCom...
- **consumer** consumerResponse
- **device** deviceResponse
- **yourPaymentMetaData** object
- **threeDSecure** threeDSecureCompletedResponse
- **risks** riskParametersResponse
- **disableNetworkTokenisation** boolean
- **networkTokenisationDetails** networkTokenisati…

```
{
  "receiptId": "",
  "originalReceiptId": "",
  "yourPaymentReference": "",
  "type": "",
  "createdAt": "",
  "result": "",
  "message": "",
  "judoId": 0,
  "merchantName": "",
  "appearsOnStatementAs": "",
  "originalAmount": "",
  "amountCollected": "",
  "netAmount": "",
  "amount": "",
  "currency": "",
  "recurringPaymentType": "",
  "acquirerTransactionId": "",
  "externalBankResponseCode": "",
  "authCode": "",
  "walletType": 0,
  "riskScore": 0,
  "paymentNetworkTransactionId": "",
  "allowIncrement": false,
  "isIncrementalAuth": false
```

27.8.63. Model: transactionReceiptHistoricResponse

Model Parameters

transactionReceiptHistoricResponse

- receiptId string
- originalReceiptId string
- yourPaymentReference string
- type string<Payment | Refund | PreAuth |…
- createdAt string<date-time>
- result string
- message string
- judoId integer
- merchantName string
- appearsOnStatementAs string
- originalAmount string
- amountCollected string
- netAmount string
- amount string
- currency string
- recurringPaymentType string
- acquirerTransactionId string
- externalBankResponseCode string
- authCode string
- postCodeCheckResult string

- **walletType** integer
- **acquirer** string
- **webPaymentReference** string
- **noOfAuthAttempts** integer
- **paymentNetworkTransactionId** string
- **allowIncrement** boolean
- **isIncrementalAuth** boolean
- **disableNetworkTokenisation** boolean
- **cardDetails** cardDetailsResponse
- **billingAddress** cardAddressResponseAttributes — All Of: cardAddressCom...
- **consumer** consumerResponse
- **device** deviceResponse
- **yourPaymentMetaData** object
- **threeDSecure** threeDSecureCompletedResponse
- **risks** riskParametersResponse
- **networkTokenisationDetails** networkTokenisati…

```
{
  "receiptId": "",
  "originalReceiptId": "",
  "yourPaymentReference": "",
  "type": "",
  "createdAt": "",
  "result": "",
  "message": "",
  "judoId": 0,
  "merchantName": "",
  "appearsOnStatementAs": "",
  "originalAmount": "",
  "amountCollected": "",
  "netAmount": "",
  "amount": "",
  "currency": "",
  "recurringPaymentType": "",
  "acquirerTransactionId": "",
  "externalBankResponseCode": "",
  "authCode": "",
  "postCodeCheckResult": "",
  "walletType": 0,
  "acquirer": "",
  "webPaymentReference": "",
  "noOfAuthAttempts": 0,
  "paymentNetworkTransactionId": "",
  "allowIncrement": false
```

27.8.64. Model: transactionReceiptListResponse

Model Parameters

transactionReceiptListResponse

- **receiptId** string
- **originalReceiptId** string
- **yourPaymentReference** string
- **type** string<Payment | Refund | PreAuth |…
- **createdAt** string<date-time>
- **result** string
- **message** string
- **judoId** integer
- **appearsOnStatementAs** string
- **originalAmount** string
- **amountCollected** string
- **netAmount** string
- **amount** string
- **currency** string
- **postCodeCheckResult** string
- **walletType** integer
- **allowIncrement** boolean
- **isIncrementalAuth** boolean
- **cardDetails** cardDetailsListResponse
- **consumer** consumerResponse

- **yourPaymentMetaData** object
- **threeDSecure** threeDSecureCompletedResponse


```
{
  "receiptId": "",
  "originalReceiptId": "",
  "yourPaymentReference": "",
  "type": "",
  "createdAt": "",
  "result": "",
  "message": "",
  "judoId": 0,
  "appearsOnStatementAs": "",
  "originalAmount": "",
  "amountCollected": "",
  "netAmount": "",
  "amount": "",
  "currency": "",
  "postCodeCheckResult": "",
  "walletType": 0,
  "allowIncrement": false,
  "isIncrementalAuth": false,
  "cardDetails": {
    "cardLastfour": "",
    "endDate": "",
    "cardType": 0
  },
  "consumer": {
    "yourConsumerReference": ""
  },
  "yourPaymentMetaData": {},
  "threeDSecure": {
    "attempted": false,
    "result": "",
    "eci": "",
    "challengeRequestIndicator": "",
    "scaExemption": ""
  }
}
```

27.8.65. Model: voidRequest

Model Parameters

voidRequest

- receiptId string *
- amount number<float>
- yourPaymentReference string


```
{
  "receiptId": "",
  "amount": "",
  "yourPaymentReference": ""
}
```