SAP PRESS

Digital Payments with SAP®

Credit Card Processing, PSP Integration, and PCI Compliance

- Deploy the SAP digital payments add-on to process credit card transactions
- Configure the add-on to protect your SAP S/4HANA or SAP ERP system
- Integrate with payment service providers and stay PCI compliant

Saravana Kumar Kuppusamy

Rheinwerk Publishing

Contents

Chapter 4

Integration with Payment Service Providers

With the SAP BTP environment configured, the SAP digital payments add-on tenants provisioned, and key components activated, the next critical step is connecting the add-on to the payment service providers that will process actual customer payments. This chapter bridges the gap between the technical infrastructure and the external payment ecosystem, providing you the knowledge and hands-on procedures to integrate Stripe and other certified PSPs so that your organization can begin accepting and processing payments at scale.

This chapter provides comprehensive guidance for integrating payment service providers (PSPs) with the SAP digital payments add-on, using Stripe as the primary example. It covers the PSP integration architecture, detailed configuration steps using Stripe's certified adapter, and setup of payment methods including cards and digital wallet options. You will learn webhook configuration for real-time payment events, handling authorization, capture, and refund scenarios using actual Stripe implementation examples. The chapter includes production-tested configurations, troubleshoots common Stripe integration issues, and describes best practices for optimizing payment processing performance.

4.1 Understanding PSP Integration Architecture

With the SAP BTP environment configured (Chapter 2) and the digital payments add-on provisioned (Chapter 3), the next important step is connecting the add-on to PSPs. While Chapter 2 introduced the high-level architecture of PSP adapters, this section explains the technical patterns, communication flows, and integration approaches that enable the add-on to communicate with external payment processors.

Understanding these architectural patterns is important before starting the hands-on configuration steps; it helps implementation teams make informed decisions about PSP selection, troubleshoot integration issues effectively, and design robust payment flows that align with business requirements.

4.1.1 Payment Gateway Fundamentals

PSPs handle the technical complexity of payment processing between merchants and the card payment ecosystem. This section focuses on the gateway mechanisms and API architecture that enable PSP integration with the digital payments add-on. To accomplish this,

we first examine the authentication and security layer that protects API communication with PSPs. We then examine the request router and validator layer that ensures data integrity before transmission. Next, we explore the API endpoint layer and how different PSPs expose distinct endpoints for authorization, capture, refunds, and tokenization. Finally, we review the communication protocol requirements and regional API endpoints that ensure your payments infrastructure meets enterprise security and compliance standards.

API Gateway Architecture

Modern PSPs expose their payment processing capabilities through REST API gateways that provide standardized endpoints for payment operations. The digital payments add-on integrates with these gateways through PSP-specific adapters. Let's review the various technical layers that are part of the PSP infrastructure:

- **Authentication and security layer**
  PSPs implement different authentication approaches for API security. The digital payments add-on adapter framework abstracts these authentication differences. Each adapter manages its PSP's authentication requirements using credentials securely stored in SAP Credential Store. Table 4.1 details the primary authentication and security mechanisms as well as any alternate authentication mechanisms for your understanding. The digital payments add-on adapter framework abstracts these details from you, but it is important to understand these mechanisms. That way, you can answer any questions that arise from your internal security and compliance team when you plan to go live with the add-on and any of the PSPs listed here.

Payment Service Provider	Primary Auth Method	Security Mechanism	Alternate Mechanisms	Details
Stripe	API key authentication	HTTP authorization header	■ Strong Customer Authentication (3D Secure, CVV checks) ■ Dashboard multi-factor authentication (MFA) ■ Internal mutual TLS (mTLS)	■ Uses secret API keys passed as bearer tokens ■ Offers robust customer-facing authentication methods

Table 4.1: Payment Service Providers: API Authentication and Security Mechanisms

Payment Service Provider	Primary Auth Method	Security Mechanism	Alternate Mechanisms	Details
Novalnet	API key authentication	HTTP authorization header	■ Hosted payment pages ■ Strong Customer Authentication (two-factor authentication [2FA])	■ Uses specific activation and access keys ■ Offers hosted solutions for secure card data handling
Cybersource	Certificate-based authentication	mTLS	■ HTTP signature (API key alternative) ■ Payer authentication (3D Secure)	■ Client certificates used for server-to-server auth ■ Multiple options for merchant API access and customer verification
Worldpay/ Paymetric	Token-based authentication	OAuth 2.0/JSON Web Tokens (JWT)	■ API keys ■ Session keys ■ Transaction session keys	■ Uses OAuth-style JWT bearer tokens

Table 4.1: Payment Service Providers: API Authentication and Security Mechanisms (Cont.)

- **Request router and validator layer**
  This layer performs technical validation before forwarding requests to payment processing engines. Key functions include the following:
  - Merchant ID resolution: Maps the merchant account alias to the PSP's technical merchant identifier.
  - Schema validation: Verifies request payloads conform to the PSP's API schema.
  - Idempotency control: Detects duplicate requests to prevent accidental double-charging.
- **API endpoint layer**
  PSPs provide distinct endpoints for different payment operations. Let's compare endpoint patterns across two certified PSPs (see Table 4.2) to see the subtle differences in their API endpoint layers. The add-on adapter framework shields implementation teams from having to deal with these complexities; they can instead integrate using one unified API framework provided by the add-on, which accelerates the implementation schedule. Consumer applications only integrate with the add-on API endpoints; the add-on takes care of managing the integration differences with each of these PSP API endpoints.

Operation	Stripe V2	CyberSource
Create authorization	`POST /v1/payment_intents`	`POST /pts/v2/payments`
Capture payment	`POST /v1/payment_intents/{id}/capture` Where `{id}` is the transaction ID from the authorization request	`POST /pts/v2/payments/{id}/captures` Where `{id}` is the transaction ID from the authorization request
Refund ppayment	`POST /v1/refunds`	`POST /pts/v2/refunds`
Tokenize card	`POST /v1/payment_methods`	`POST /tms/v2/tokens`

Table 4.2: PSP API Endpoint Comparison

Regional API Endpoints

Global PSPs operate multiple regional API gateways for performance and data residency compliance. Stripe maintains a global endpoint (*api.stripe.com*) that automatically routes to regional data centers based on merchant account configuration. Cybersource provides distinct regional endpoints. Table 4.3 shows the regional endpoints available for widely used PSPs. It is important to understand these details so that you can choose the right PSP based on your company's global and regional policies and security requirements tied to payment processing.

PSP	Endpoint Strategy	Data Center (EU)	Data Center (US)
Stripe	Global endpoint with intelligent routing	Yes (via global endpoint)	Yes (via global endpoint)
Novalnet	Single primary data center location	Yes (primary data centers in Germany)	No (data is in EU)
Cybersource	Distinct regional endpoints required	Yes (via specific EU endpoint)	Yes (via specific NA endpoint)
Worldpay/Paymetric	Multiple regional endpoints/platforms	Yes (multiple EU endpoints exist)	Yes (multiple US endpoints exist)

Table 4.3: PSP Data Center Strategy

Canonical Request and Response Formats

All certified PSPs use JSON for request payloads and responses and have their own API specification. The add-on API specification defines the canonical request structure to be used. PSP adapters translate digital payments add-on canonical request structures into

individual PSP-specific formats. Implementation teams do not have to spend time trying to define a common canonical format for integrating with multiple PSPs; instead, they can leverage the add-on-provided canonical format to accelerate the implementation.

Let's look at an example digital payments add-on canonical request structure for a payment authorization of USD 15,000, shown in Listing 4.1.

```
json
{
  "AuthorizationByDigitalPaytSrvc": "DPA-AUTH-12345",
  "AmountInAuthorizationCurrency": "15000",
  "AuthorizationCurrency": "USD",
  "PaytCardByDigitalPaymentSrvc": "DPA-CARD-67890",
  "MerchantAccount": "merchant-alias-001"
}
```

Listing 4.1: Add-On: Example Canonical Request Structure

The PSP adapter translates this canonical JSON into the PSP-specific JSON format. For Stripe, this becomes what you see in Listing 4.2.

```
json
{
  "amount": 15000,
  "currency": "usd",
  "payment_method": "pm_1NAbCD2eZvKYlo2C",
  "capture_method": "manual",
  "metadata": {
    "dpa_auth_token": "DPA-AUTH-12345"
  }
}
```

Listing 4.2: Stripe PSP: JSON Example Request Payload

For another PSP, its JSON format would be different; however, implementation teams do not have to manage this complexity. Instead, they just define the routing conditions in the add-on configuration UI to route traffic to individual PSPs based on their company requirements. The add-on takes care of the mapping between canonical and PSP-specific formats.

Communication Protocol Requirements

It is recommended that all PSP communications use the following:

- Transport protocol: HTTPS (port 443)
- TLS version: 1.2 or higher (mandatory)
- Certificate validation: Required for production environments

For SAP S/4HANA and SAP ERP systems, these requirements are implemented through RFC destinations (Type G, HTTP connection to external server), configured in Transaction SM59. The DigiCert TLS RSA4096 Root G5 certificate must be installed in Trust Manager (Transaction STRUST). For other applications, you can refer to their administration and user guides for the configuration steps to enable the recommended communication protocols so that your payment processing will be secure and meet your enterprise standards.

4.1.2 API-Based Integration Patterns

Payment processing through the add-on follows industry-standard gateway integration patterns. Understanding these patterns is important for designing effective order-to-cash and invoice-to-cash processes. In this section, we examine the two-step *authorization and capture* pattern, which separates the fund reservation phase from the settlement phase, a typical requirement for scenarios in which the final invoice amount may differ from the initial order amount. We then explore the direct capture payment flow, which combines authorization and settlement into a single operation for use cases requiring immediate payment without preauthorization. Finally, we explore the three distinct Stripe API integration patterns that accommodate different payment collection scenarios, from standard add-on card registration to external collection with token preparation to centralized refund operations.

Two-Step Authorization and Capture

The most common pattern for business-to-business (B2B) and business-to-consumer (B2C) scenarios involves separating authorization from settlement. Figure 4.1 shows the two-step flow.

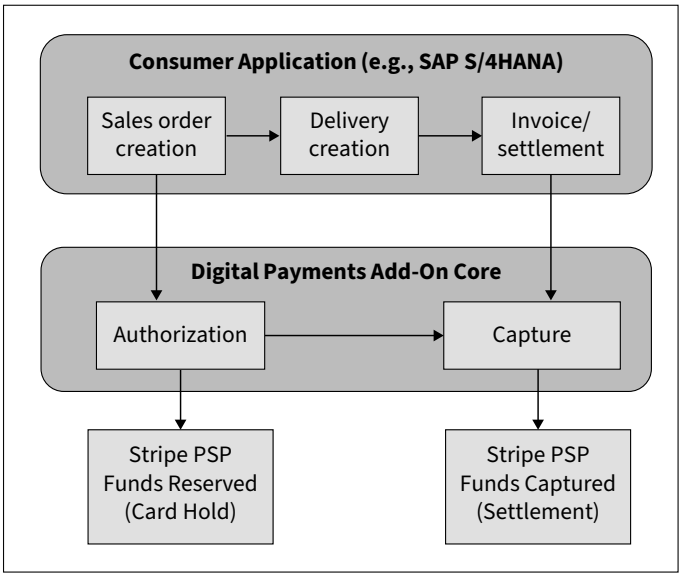

Figure 4.1: Authorization and Capture: SAP S/4HANA Two-Step Scenario

The following are the steps involved to authorize a $1,000 payment when placing a sales order:

1. A customer places an order totaling $1,000.
2. The consumer application calls the digital payments add-on */v1/authorizations* API.
3. The digital payments add-on routes the authorization request to the appropriate PSP adapter (e.g., Stripe).
4. The adapter translates the consumer application authorization request into a PSP-specific authorization request.
5. The PSP reserves funds on the customer's card (as a hold; not yet charged).
6. The PSP returns an authorization code (e.g., ch_3ABC...).
7. The digital payments add-on creates an internal authorization token and returns it to the consumer application.
8. The order proceeds to delivery processing.

In the capture phase (billing document settlement), if there are physical goods involved, capture happens after the goods are shipped. The following are the steps during the capture phase:

1. Goods are shipped, and an invoice is generated in the consumer application.
2. Finance runs the payment settlement job.
3. The consumer application calls the digital payments add-on */v1/charges* API with an authorization token.
4. The digital payments add-on retrieves the PSP authorization reference.
5. The adapter sends a capture request to the PSP.
6. The PSP transfers funds from the customer to the merchant account.
7. The settlement is posted in financial accounting.

This pattern is essential for scenarios in which the final invoice amount may differ from the initial order amount due to delivery variances, price adjustments, or partial shipments.

Direct Capture Payment Flow

For scenarios requiring immediate payment without preauthorization, the add-on supports direct capture. Common use cases are as follows:

- Subscription billing, in which cards are charged on renewal dates
- Invoice payment links sent to customers
- E-commerce checkout with immediate fulfillment

Figure 4.2 shows the simplified direct capture flow.

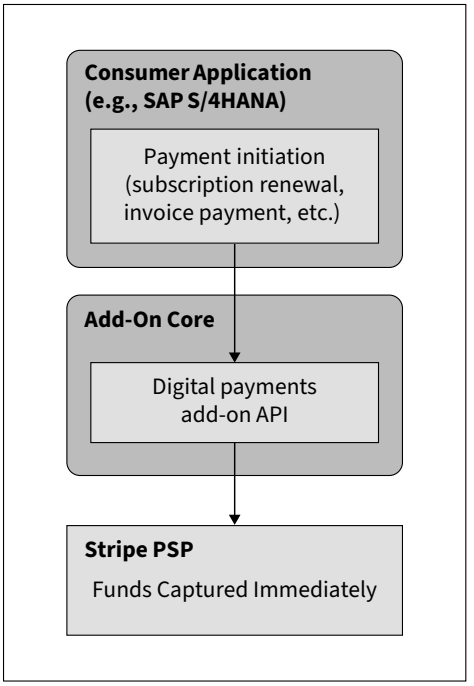

Figure 4.2: Direct Capture Flow

The key distinction is that a single API call performs both authorization and settlement simultaneously at the PSP.

Stripe API Integration Patterns

The Stripe V2 adapter implements three distinct integration patterns depending on where the initial payment interaction occurs:

- **Pattern A: Standard digital payments add-on card registration**
  In this pattern, payment card registration is initiated from the consumer application (e.g., SAP S/4HANA's business partner maintenance). The Stripe V2 adapter receives the registration request, then returns the correct registration URL for the consumer application to securely capture the payment card details. Subsequently, the consumer application can leverage the poll API endpoint to poll the card details and get the digital payments add-on's internal token. This token is then stored in the application for future use, if customer consent to do so is obtained during card entry. Subsequent transactions from the same customer can reuse the digital payments add-on token, thereby simplifying payment integration requirements. Figure 4.3 depicts a simple logical flow to illustrate this pattern.

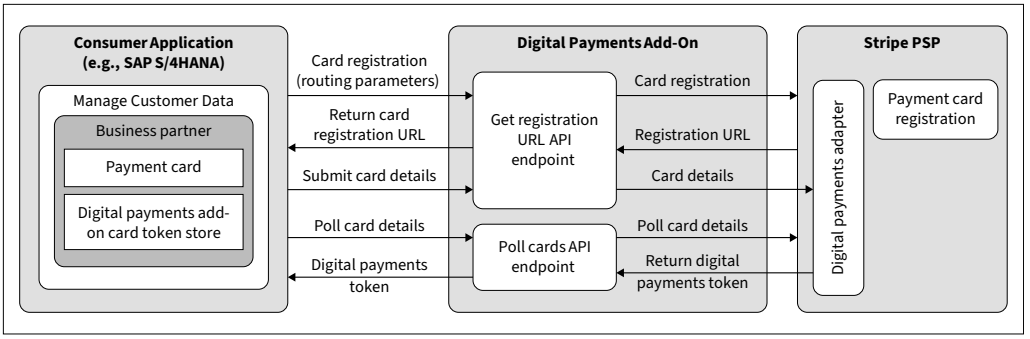

Figure 4.3: Standard Digital Payments Add-On Card Registration Pattern

- **Pattern B: External collection with token preparation**

  This pattern supports scenarios in which an external e-commerce storefront collects payment information by directly integrating with Stripe but needs to hand off the transaction to an SAP system (e.g., SAP S/4HANA) for order processing and fulfillment. The `getforpaymentcardwithauthorization` endpoint validates that the external payment intent exists and has status `requires_capture`, then wraps it in a digital payments add-on authorization token that SAP S/4HANA can use.

  After the e-commerce application sends the order information (along with the digital payments add-on authorization ID) to the SAP system, the order document follows the typical document flow steps as necessary to be fulfilled (including, for physical goods, shipment to the customer). After the fulfilment is complete, the SAP application triggers the end of business day settlement operation to capture the funds on the customer payment card. It is possible for the customer to request a refund if, say, the goods were damaged or lost in transit or for another reason; in that situation, the SAP application can trigger the refund endpoint as part of the credit memo settlement process. Figure 4.4 shows the step-by-step flow.

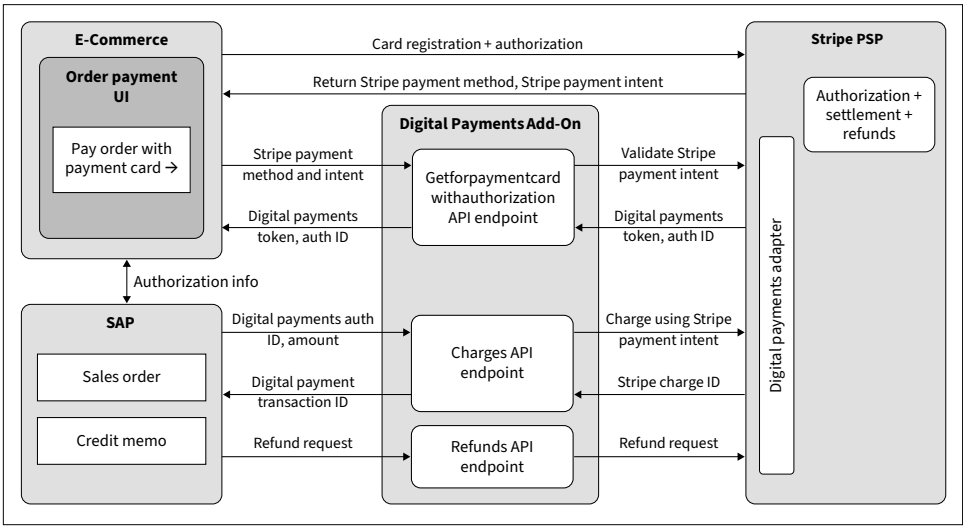

Figure 4.4: External Collection with Token Preparation

- **Pattern C: Direct capture token preparation**
  This pattern is used when payments are completed outside the digital payments add-on flow (such as in a mobile app or a self-service web application for your customer invoice payments) but your company wants to centralize refund processing through the digital payments add-on for consistent financial reconciliation. Figure 4.5 shows the step-by-step sequence of events that can facilitate centralizing refund processing for all payment transactions in your enterprise.

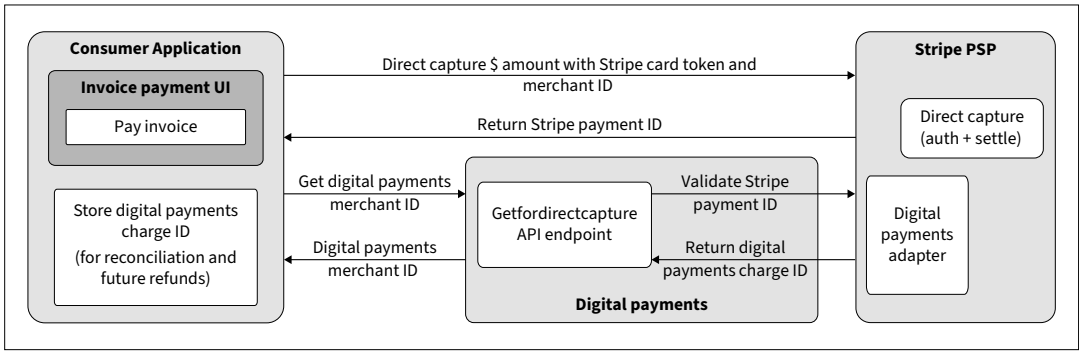

Figure 4.5: Centralized Refund Operations Through Digital Payments Add-On

The consumer applications, after they complete their payment flows outside of digital payments add-on, can integrate to the `getfordirectcapture` add-on API endpoint and get the digital payments add-on charge ID from the PSP. The digital payments add-on charge ID can then be used to centrally manage refund processing. As shown earlier in Figure 4.4, the consumer applications can send the digital payments add-on charge ID to the `refunds` API endpoint to trigger a refund (either partial or full, based on your company's needs). This pattern avoids the need for your implementation team to integrate all your company's applications with the digital payments add-on for all payment processing. Those applications can continue to integrate directly with the PSP and manage payment processing; just a minor enhancement is needed to get and store the digital payments add-on charge ID in the application. This digital payments add-on charge ID then can be used to manage refund processing as needed.

4.1.3 PSP Adapter Framework

As discussed in Chapter 2, PSP adapters serve as translation layers between the add-on's standardized API and each PSP's unique protocol. The adapter framework follows a plug-in architecture that allows different PSPs to integrate with the add-on without modifying the core service. Figure 4.6 illustrates the logical separation between the adapter framework and individual PSP implementations.

Each adapter implements a standard contract defined by the framework while maintaining complete independence in how it communicates with its target PSP. This design provides some architectural advantages:

- **Version independence**
  Adapters can be updated to support new PSP API versions without requiring changes to the digital payments add-on core service or other adapters. For example, Stripe's transition from the V1 to the V2 API was handled entirely within the Stripe adapter.
- **Error isolation**
  Failures in one adapter's communication with its PSP do not affect other adapters. If Stripe experiences an API outage, for example, transactions routed to Cybersource or Worldpay continue processing normally.
- **Independent certification**
  Each PSP can develop and certify its own adapter following SAP's adapter specification, enabling a growing ecosystem of payment providers without SAP needing to build every integration directly.

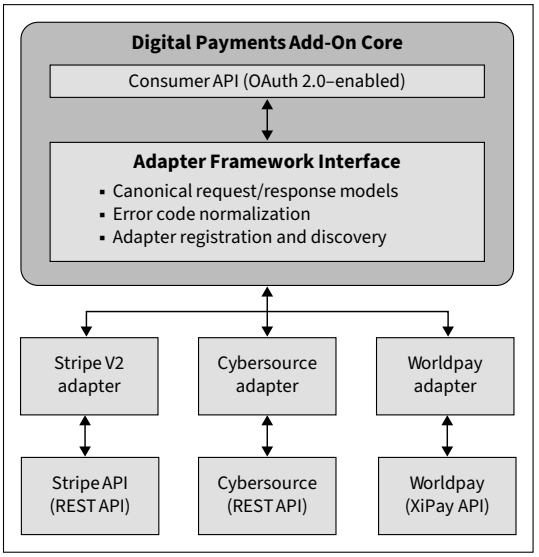

Figure 4.6: Adapter Framework: Logical Architecture

Adapters perform these essential functions:

- **Protocol and format translation**
  - Transform the digital payments add-on's canonical request structure into PSP-specific formats.
  - Convert PSP responses back into the add-on's standardized representation.
  - Handle PSP-specific API versioning (e.g., the Stripe PaymentIntents API vs. the legacy Charges API).
- **Secure communication**
  - Manage outbound HTTPS connections to PSP API endpoints.
  - Retrieve PSP API credentials from SAP Credential Store.
  - Handle TLS 1.2+ encryption.

- **PSP-specific processing**
  - Support unique PSP features (e.g., Level 2/3 data for corporate cards, 3D Secure authentication).
  - Manage PSP-specific metadata and custom fields.
  - Handle regional API variations and payment method availability.

Administrators can control which PSP adapters are available for payment routing through the PSP status management UI. Activating a PSP makes the adapter available to the routing engine. When an adapter is activated, the following is true:

- The routing engine includes this PSP when evaluating routing rules
- The adapter becomes eligible to process payment transactions
- Merchant account credentials must be configured

When an adapter is deactivated, the following is true:

- The routing engine excludes this PSP from consideration
- Existing merchant configurations remain but are not used
- No new transactions can be routed to this PSP

Activating an adapter does not automatically route transactions to it. You must also configure PSP determination rules that specify when to use a particular PSP, based on routing parameters such as company code, currency, country, or payment type.

Global organizations often require multiple merchant accounts to support different business units, brands, or geographic regions. The adapter framework supports multimerchant scenarios through the merchant mapping functionality. Figure 4.7 shows how merchant aliases flow through the system.

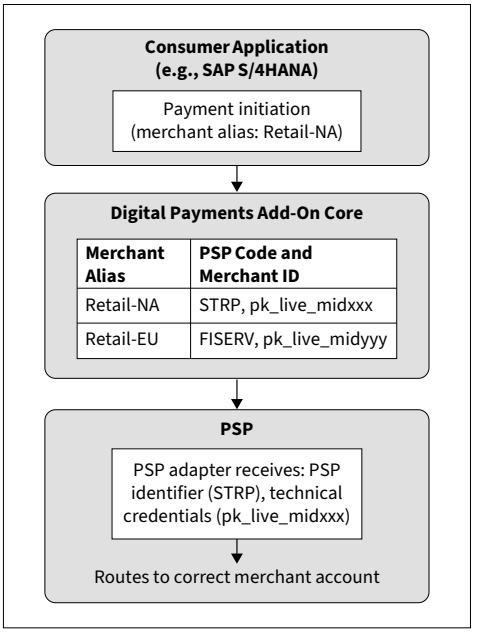

Figure 4.7: Add-On: Merchant Alias Mapping Mechanism

This merchant alias approach provides two key benefits:

- Consumer applications use logical names (e.g., virtual bank account IDs) rather than PSP-specific technical identifiers.
- Switching PSPs requires updating only the merchant mapping configuration, not the consumer application master data.

One of the critical adapter framework responsibilities is normalizing PSP-specific responses into the digital payments add-on's canonical format. Adapters translate PSP status codes into standardized digital payments add-on responses that consumer applications can handle consistently. Stripe PSP, as an example, uses codes like *card declined*, *insufficient funds*, and *expired card*, while other PSPs may use different terminology for the same conditions. The adapter framework maps these into digital payments add-on standard codes so that consumer applications can manage payment processing logic–related functions identically, regardless of which PSP processes the transaction.

Stripe is one of the most widely deployed PSP adapters for the digital payments add-on. PSP adapters follow versioning independent from the digital payments add-on core service. The Stripe adapter versioning demonstrates this:

- Stripe V1: Legacy adapter using the Stripe Charges API (deprecated)
- Stripe V2: Current production adapter using the Stripe PaymentIntents API (recommended)

Both versions can coexist within the same digital payments add-on tenant, differentiated by PSP code configuration. This enables phased migration strategies in which existing integrations continue operating on V1 while new implementations use V2.

Unlike the V1 adapter, which was tightly coupled to the digital payments add-on core, Stripe V2 operates as a fully independent microservice with its own dedicated infrastructure. Table 4.4 illustrates the Stripe V2 adapter's internal architecture.

Stripe V2: Four Key Internal Components	
Adapter API gateway: ■ Request validation ■ Rate limiting ■ Request/response logging	Stripe configuration service: ■ Merchant onboarding UI ■ API key management ■ Webhook endpoint registration
Payment processing engine: ■ Payment intent management ■ Payment method handling ■ 3D Secure orchestration ■ L2/L3 data processing	Webhook event handler: ■ Event validation (signature check) ■ Event processing queue ■ Retry logic

Table 4.4: Stripe V2 Adapter: Internal Architecture

The key architectural improvements in V2 are as follows:

- **Independent user interface services**
  Stripe V2 provides dedicated UI applications hosted at region-specific URLs (e.g., *us-prod-uiservices-dpa.sap.stripeconnectors.com*). These UIs handle merchant onboarding, configuration management, and advice scheduling without requiring access to the main digital payments add-on configuration interfaces.
- **Enhanced PaymentIntent API**
  V2 leverages Stripe's modern PaymentIntent API instead of the legacy Charge API used in V1. PaymentIntents provides better support for strong Customer Authentication (SCA) requirements under the Revised Payment Services Directive (PSD2), handles complex payment flows more gracefully, and offers improved idempotency controls.
- **Payment method support**
  V1 primarily supported credit/debit cards, but V2 natively integrates with Stripe's full payment method catalog, including the following:
  - Digital wallets (Apple Pay, Google Pay, Link) and buy now, pay later options
  - Bank transfers (ACH, SEPA, BACS, BECS, ACSS) and direct debit schemes
  - Regional payment methods (iDEAL, Bancontact, Giropay, EPS, P24, Alipay, WeChat Pay)
- **Separate webhook infrastructure**
  V2 implements a dedicated webhook event-processing system with signature verification, automatic retry logic, and event deduplication. This ensures that payment state changes (successful captures, failed authorizations, refund completions) are reliably propagated back to the digital payments add-on core service.

4.2 Setting Up PSP Accounts and API Credentials

This section provides practical, step-by-step guidance for creating PSP accounts and obtaining the API credentials required for adapter configuration. We'll use Stripe as the primary example (with production-tested procedures). You'll learn how to set up both test and production PSP environments, generate the necessary API keys, and prepare webhook endpoints for event handling.

4.2.1 PSP Account Creation

Before the digital payments add-on can process transactions through a PSP, you must establish a commercial relationship with that provider and create the necessary merchant accounts. This section covers the account creation process for Stripe and highlights the differences for other certified PSPs.

Before creating PSP accounts, ensure the prerequisites listed in Table 4.5 are met.

Category	Prerequisite
Business registration documents	PSPs require the following business verification documents: ■ Business registration certificate or articles of incorporation ■ Tax identification number (EIN in US, VAT number in EU) ■ Business bank account information for settlement ■ Authorized signatory identification documents
Commercial agreement	Review and accept the PSP's service agreement, including the following: ■ Transaction processing fees ■ Monthly minimum fees (if applicable) ■ Chargeback fees ■ Currency conversion fees for multicurrency processing
Technical contact information	Identify the team members who will receive the following: ■ API credential notifications ■ Webhook event notifications ■ Technical support communications ■ Security alerts

Table 4.5: PSP Account Setup: Prerequisites

Stripe Account Creation

Stripe offers a streamlined self-service onboarding process, as described in Table 4.6.

Step	Details
Initial registration	■ Navigate to *https://stripe.com* and select **Get Started** or **Sign up with Google.** ■ Enter your business email address and create a password. ■ Verify your email address by clicking the confirmation link sent to your inbox. ■ Complete your profile by providing the following: – Business legal name and address – Industry/business type – Website URL (if applicable) – Expected processing volume

Table 4.6: Stripe Account Creation Steps

Step	Details
Business verification	Stripe's automated verification typically requests the following business details: ■ Tax ID (EIN for US entities, VAT for European entities) ■ Business structure (LLC, corporation, partnership, sole proprietorship) ■ Date of incorporation ■ Legal name and date of birth of authorized representative ■ Government-issued ID verification (passport, driver's license) ■ Social Security Number or equivalent (for US entities)
Bank account details	■ Bank account number and routing number for settlement deposits ■ Account ownership verification (typically via microdeposit confirmation)

Table 4.6: Stripe Account Creation Steps (Cont.)

Stripe may request additional documentation for certain industries or higher-risk business models.

Next, you'll need to install the SAP Digital Payments Add-On V2 app. This step registers your Stripe account with the digital payments add-on ecosystem. You can install the SAP-specific app from Stripe's App Marketplace, as follows:

1. Navigate to the Stripe marketplace (*https://marketplace.stripe.com/*), as shown in Figure 4.8, and search for "SAP Digital Payments Add-On" in the search bar.

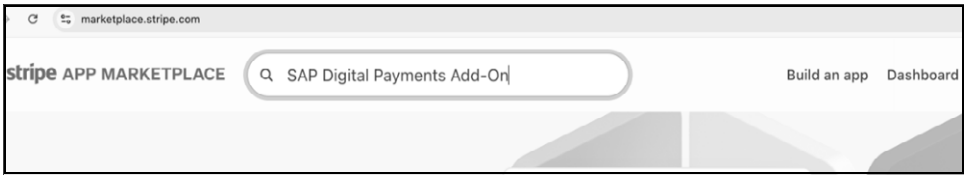

Figure 4.8: Stripe App Marketplace

2. Select the **SAP Digital Payments Add-On V2** app from the search results (see Figure 4.9).

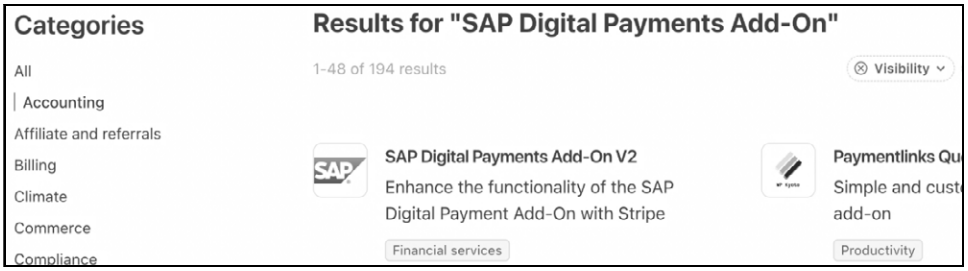

Figure 4.9: Add-On App

3. Select **Install app** (see Figure 4.10). When installing, you can choose the installation mode:
   - **Install in test mode**: For development and quality assurance activities (connects to your test subaccount)
   - **Install in live mode**: For production transaction processing (connects to your production subaccount)

   You must install the app separately in both test and live modes.

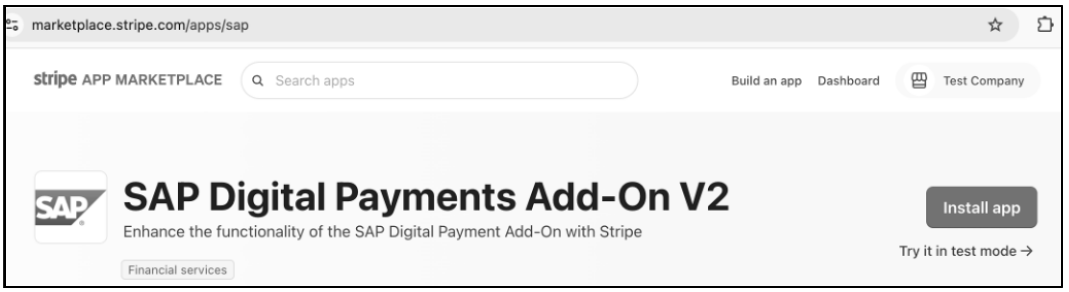

Figure 4.10: Add-On: Install App

4. In most implementations, you will install test mode first, validate the complete integration, then install live mode before production go-live. Review the app permissions (read and write access to payment methods, read and write access to payment intents, read access to charges and refunds, and webhook event subscription permissions), then click **Install app in test mode** (see Figure 4.11) to complete the installation.

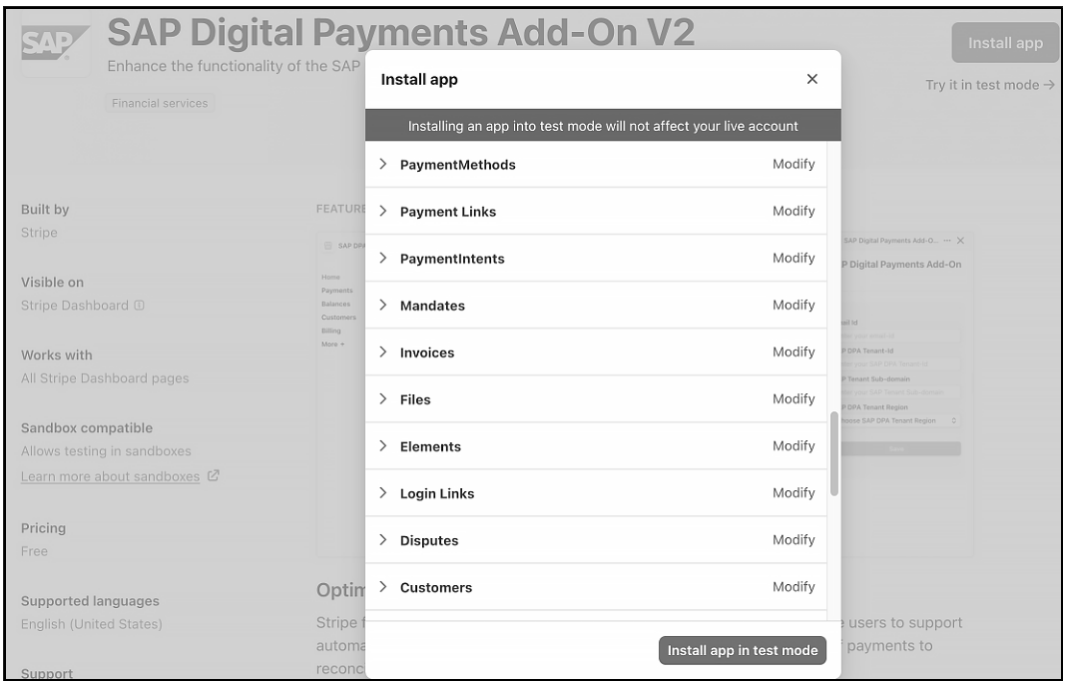

Figure 4.11: Add-On App Installation: Review Permissions

5. After installation, you will see a message prompting you to **Continue to app settings**, as shown in Figure 4.12.

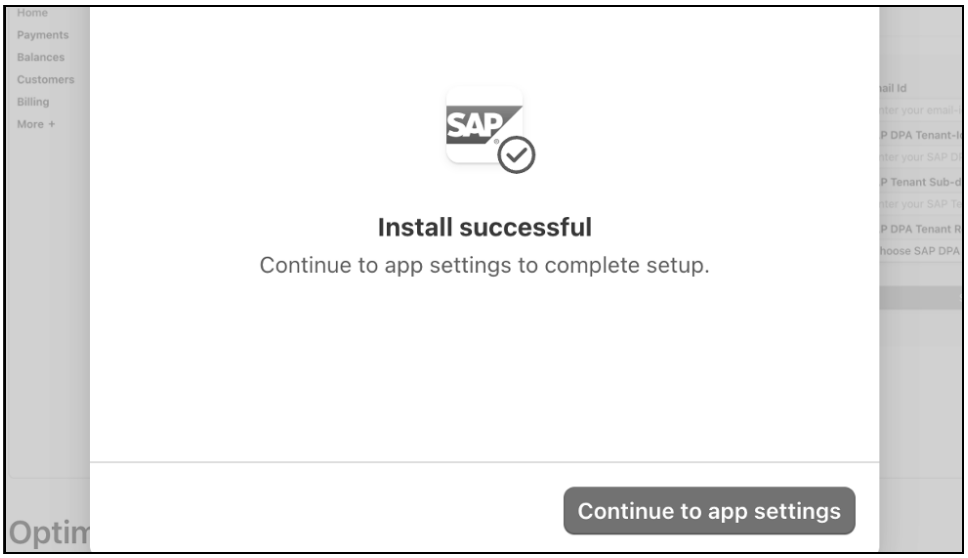

Figure 4.12: Postinstall Step: App Settings

6. The app displays your API credentials (publishable key and secret key), as shown in Figure 4.13. Store the keys in a safe place where you won't lose them. The best practice for security is to copy both keys to a secure password manager or encrypted vault. Do not store them in plaintext files or email them to team members.

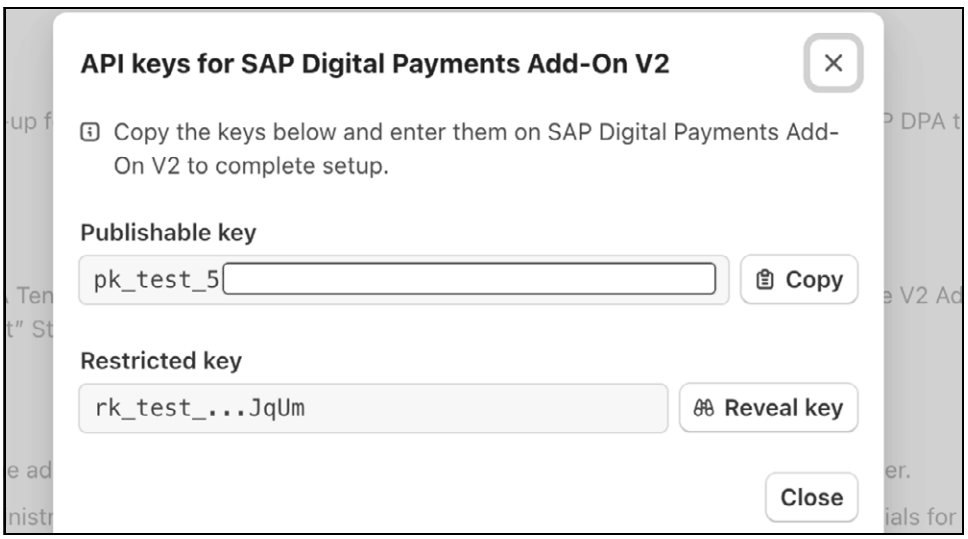

Figure 4.13: Add-On Installation: API Keys

7. Select **Close**. Figure 4.14 shows the Stripe dashboard view after successful app installation, where **SAP Digital Payments Add-On V2** appears in the installed apps list.

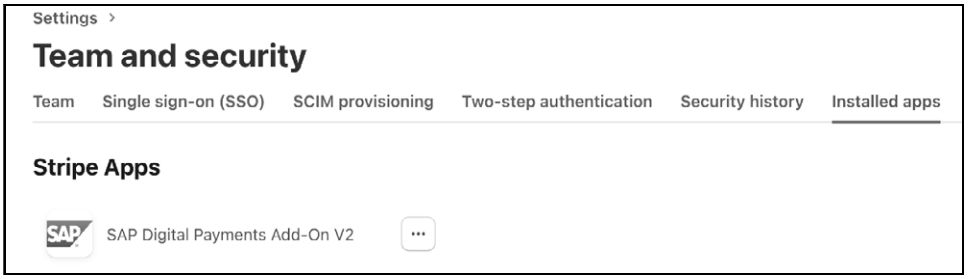

Figure 4.14: Stripe Dashboard: Add-On App Successful Installation

To retrieve or regenerate API keys after the initial installation, navigate to the **Apps** section in the Stripe dashboard, select **SAP Digital Payments Add-On V2**, and click **View API keys** (see Figure 4.15) to display the current credentials.

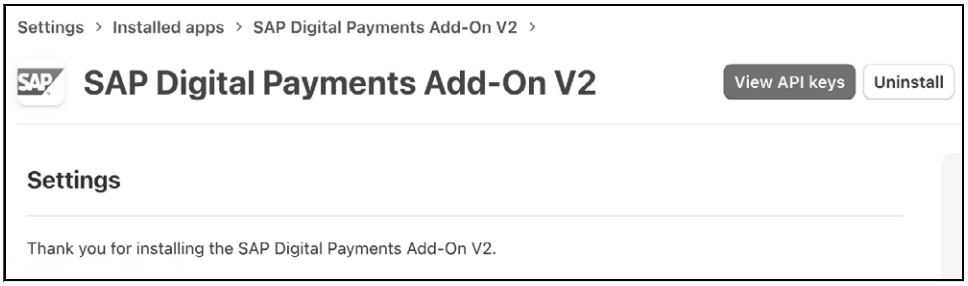

Figure 4.15: Postinstall: View API Keys

Large companies often require multiple Stripe accounts for different business units or geographic regions. The following are the most common multimerchant scenarios:

- Brand separation: Separate merchant accounts for different customer-facing brands to maintain distinct settlement reporting.
- Geographic separation: Different accounts for US versus European operations to comply with data residency requirements.
- Business unit independence: Separate profit and loss tracking for divisions or subsidiaries.

When you see a need to create multiple Stripe merchant accounts, follow this checklist:

- Use a different email address for each account (Stripe requires a unique email per account).
- Complete the business verification process for each entity.
- Install the SAP Digital Payments Add-On V2 app in each account separately.
- Configure separate merchant mappings in the digital payments add-on.
- Configure PSP determination parameters in the digital payments add-on configuration UI.

Account Creation for Other Certified PSPs

Unlike Stripe's self-service model, Cybersource requires assisted onboarding. The typical steps to work through are as follows:

- Contact the Cybersource sales team or your acquiring bank partner.
- Complete a merchant application with business verification documents.
- Sign the merchant service agreement.
- Receive your merchant ID assignment (typically within 10–15 business days).
- Gain access to your Cybersource Business Center portal credentials.
- For digital payments add-on integration, do the following:
  - Request the tokenization feature activation on your account.
  - Collect your SAP BTP tenant ID.
  - Email the merchant ID and tenant ID mapping to *GDLSAPDPA@visa.com*, clearly indicating if this is for the test or production environment.

Worldpay integration (formerly Paymetric) requires multiple steps, as follows:

- Establish a Worldpay merchant account by visiting *https://www.sap.com/products/crm/partners/worldpay-uk-limited-worldpay-b2b-payments-formerly-paymetric.html* and click **Request a Quote** to initiate a quote for a contract.
- Contract with WorldPay for XiSecure tokenization services.
- Onboard with the token exchange service if multi-PSP token portability is required.
- Request 3DS Flex onboarding if using 3D Secure authentication.
- Receive your credential package, including the following:
  - Merchant account ID
  - Web Services API (WSA) user name and password
  - TMS profile name and merchant identifier
  - Shared key for webhook signature validation

The following is a quick checklist for setting up a PayPal Braintree account; this is for reference only as the onboarding steps vary. Braintree follows PayPal's standard merchant onboarding:

- Create a PayPal Business account at paypal.com.
- Navigate to the **Braintree** section within the PayPal Business dashboard.
- Complete the Braintree-specific verification (additional underwriting beyond PayPal).
- Receive your Braintree merchant ID and API credentials.
- Generate sandbox credentials for testing.

The Novalnet account setup checklist is EU-region focused, and the steps are as follows:

- Complete a merchant application with European business verification.

- Select the desired payment methods from the available options.
- Configure settlement currency preferences.
- Receive API credentials and project ID assignments.

4.2.2 API Keys and Webhooks Overview

PSP API credentials consist of multiple components, each serving a specific security and operational purpose. Again, we'll split our focus between Stripe and other PSPs.

Stripe API Credential Types

Stripe uses a dual-key architecture for API authentication. Table 4.7 lists key features of the publishable key that you can use as a best practices reference.

Characteristic	Description
Visibility	Can be embedded in client-side code (JavaScript, mobile apps)
Permissions	Limited to creating tokenized payment methods and setup intents
Security level	Safe to expose in frontend applications
Usage in digital payments add-on	Used as the `MerchantAccount` parameter in digital payments add-on API calls
Rotation	Can be regenerated without affecting existing tokens

Table 4.7: Publishable Key Characteristics

Table 4.8 lists key features of the secret key that you can use as a best practices reference.

Characteristic	Description
Visibility	Must be kept strictly confidential, never exposed in client-side code
Permissions	Full account access including charges, refunds, and customer data access
Security level	Equivalent to root credentials, must be stored encrypted
Usage in digital payments add-on	Used by Stripe adapter to authenticate backend API calls
Rotation	Requires updating the adapter configuration

Table 4.8: Secret Key Characteristics

Credential Types for Other PSPs

The credential types for other certified PSPs are as follows:

- **Cybersource**
  Cybersource uses an authentication model based on merchant profiles. Cybersource credentials are not created through self-service;l they are generated by the account manager or requested through the Cybersource Business Center portal with appropriate administrative permissions. You can obtain more details by visiting the Cybersource developer website at *https://developer.cybersource.com/technology-partners/sap-dpa.html*.
- **Worldpay/Paymetric**
  The Worldpay integration requires multiple credential sets due to its multicomponent architecture. You can obtain more details by visiting the Worldpay website at *https://docs.paymetric.info/sap-digital-payments-add-on-pr.htm*.
- **Novalnet**
  Novalnet's payment module for the SAP digital payments add-on provides a unified dashboard that acts as a central platform to accept digital payments, process payment transactions in a secure manner, and manage payments. You can obtain more details by visiting the Novalnet website at *https://www.novalnet.com/integration/sap-digital-payment-module/*.

4.2.3 Test and Production Environments Setup

All certified PSPs provide separate test and production environments to enable safe implementation and testing without risking live transactions or customer data. This section covers separation requirements for environments and best practices.

Stripe Environments

Stripe implements environment separation through "modes" rather than separate systems (see Table 4.9).

Test Mode Characteristics	Live Mode Characteristics
■ API endpoint: Same base URL (api.stripe.com) for both modes. ■ Transaction processing: No real card networks involved; uses Stripe's test card numbers. ■ Settlement: No actual funds transfer; simulated settlement for testing. ■ Data isolation: Separation from live mode data; test transactions never appear in live reporting.	■ API endpoint: Same base URL (api.stripe.com). ■ Transaction processing: Real card network communication with actual authorizations and declines. ■ Settlement: Actual funds transferred to business bank account (typically T+2 business days). ■ Production data: All transactions appear in live reporting and reconciliation.

Table 4.9: Stripe Test Mode Versus Live Mode

The Stripe dashboard includes a mode toggle in the top-left corner (see Figure 4.16). This toggle controls which environment you're viewing and configuring. When installing the SAP Digital Payments Add-On V2 app, you must explicitly choose which mode to install it in; the credentials generated are specific to the selected mode.

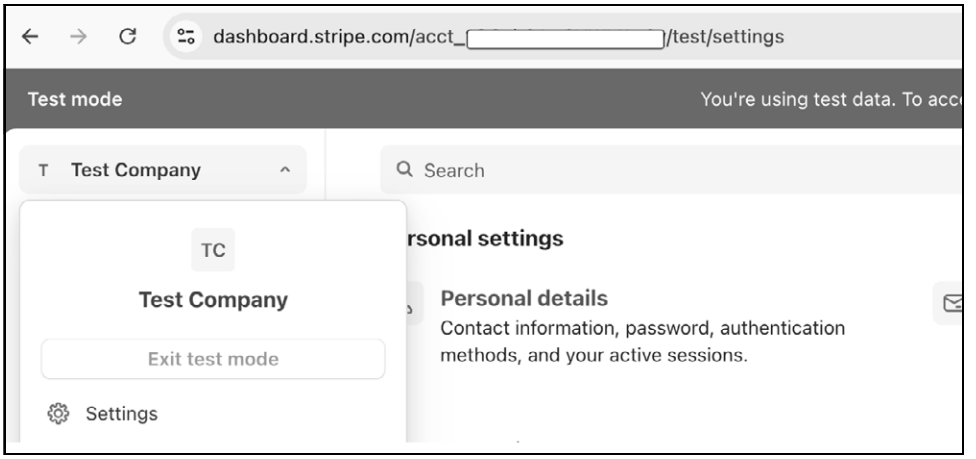

Figure 4.16: Test and Live Mode Toggle

Your SAP BTP landscape includes separate subaccounts for test and production. These must be mapped to corresponding PSP environments. Table 4.10 shows the correct mapping.

SAP BTP Subaccount	Subscription Plan	Stripe Environment	Stripe Credentials	Consumer Application
DP Test	SAP digital payments add-on demo	Test mode	pk_test_..., sk_test_...	Development and quality assurance SAP S/4HANA systems
DP Production	SAP digital payments add-on	Live mode	pk_live_..., sk_live_...	Production SAP S/4HANA system

Table 4.10: PSP Environment Mapping to SAP BTP Subaccounts

Do not configure live mode PSP credentials in the DP Test subaccount or test mode credentials in DP Production. Mixing environments can result in test transactions being charged to real cards or production transactions failing in test mode.

After installing the app, you must connect your SAP BTP tenant to the Stripe adapter, as follows:

1. From the Stripe dashboard, navigate to **Apps • SAP Digital Payments Add-On V2**, then click the app to open the onboarding interface. A window opens as shown in Figure 4.17.

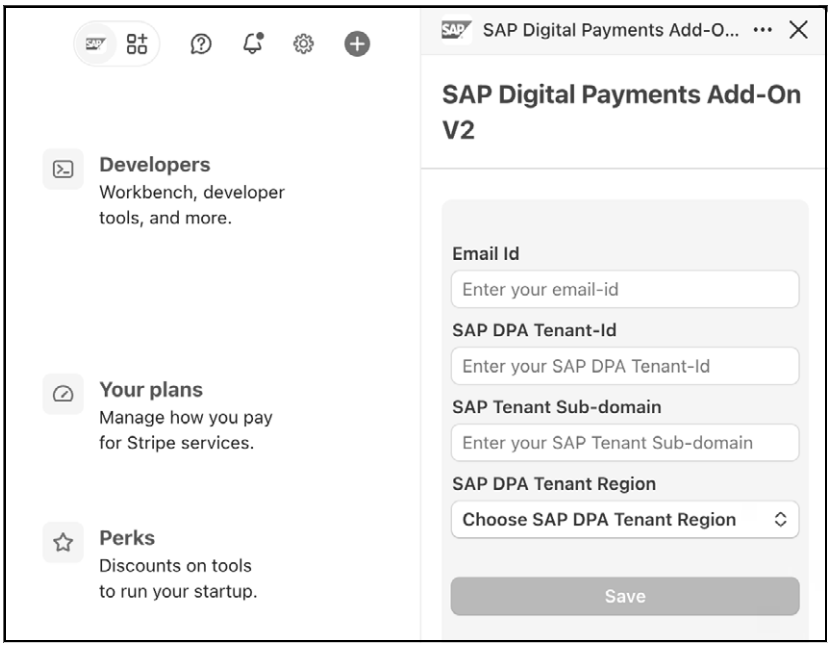

Figure 4.17: Stripe Dashboard: Add-On Onboarding UI

2. Enter the necessary information; see Table 4.11 for a guide.

Field	Description
Email	Your administrator email for account verification and recovery
SAP DPA Tenant-Id	Found in your SAP BTP subaccount URL (the subdomain portion)
SAP Tenant Sub-domain	The subdomain you created in Chapter 3
SAP DPA Tenant Region	Select from dropdown: ■ **US10 Test** (for US East Virginia test environment) ■ **EU10 Test** (for Europe Frankfurt test environment)

Table 4.11: Add-On Onboarding in Stripe

3. Click **Save** to create the connection.

You will receive an email notification from *no-reply@verificationemail.com* with your user name and temporary password. These credentials are distinct from your Stripe login and are used specifically to access the Stripe configuration UI (for onboarding merchants) and the Stripe advice UI (for scheduling payment advice retrieval).

Maintaining strict separation between the test and production environments prevents critical issues. Table 4.12 lists the advantages of environment isolation.

Category	Advantages
Data contamination prevention	■ Test transaction tokens never mix with production payment records. ■ Customer card data registered in production never appears in test systems. ■ Financial reconciliation remains clean, with no test data in production reports.
Testing safety	■ Developers can safely test refund scenarios without affecting real customer payments. ■ Quality assurance teams can simulate payment failures without impacting production authorization rates. ■ Integration testing can use high transaction volumes without triggering PSP fraud alerts.
Compliance separation	■ The PCI DSS audit scope clearly distinguishes test versus production systems. ■ Production credential access logs are separate from test environment access logs. ■ Security incident response procedures can isolate test environment compromises.

Table 4.12: Environment Isolation: Advantages

Other PSP Environment Models

Cybersource uses separate test and production instances; that is test and production are completely isolated systems, as follows:

- Test environment: Separate URL (*apitest.cybersource.com*) with distinct test merchant IDs.
- Production environment: Production URL (*api.cybersource.com*) with live merchant IDs.

Worldpay, on the other hand, has separate merchant IDs and credentials for each environment:

- Certification environment: For integration development and testing.
- Production environment: For live transaction processing.

4.3 Configuring PSP Adapter in the Add-On

With PSP accounts created and API credentials obtained, the next step is configuring the PSP adapter within your digital payments add-on tenant. This section provides details for activating the Stripe V2 adapter, onboarding merchant accounts, and establishing the technical connection between your SAP BTP tenant and Stripe's payment infrastructure. Stripe

serves as the primary example; configuration differences will be highlighted for other certified PSPs.

4.3.1 Adapter Installation

PSP adapter installation in the digital payments add-on context refers to *activation* rather than deploying new software components. Adapters are predeployed within your SAP BTP tenant when you subscribe to the add-on service. Activation makes the adapter available to the routing engine for transaction processing.

Steps for accessing the PSP status management UI are as follows:

1. Log into the SAP BTP cockpit (*https://cockpit.btp.cloud.sap*), navigate to your DP Test subaccount, and select **Services • Instances and Subscriptions** from the left navigation pane. Under **Subscriptions**, locate the **SAP digital payments add-on Demo** subscription, then click **…** and select the **Go to Application** link, as shown in Figure 4.18.

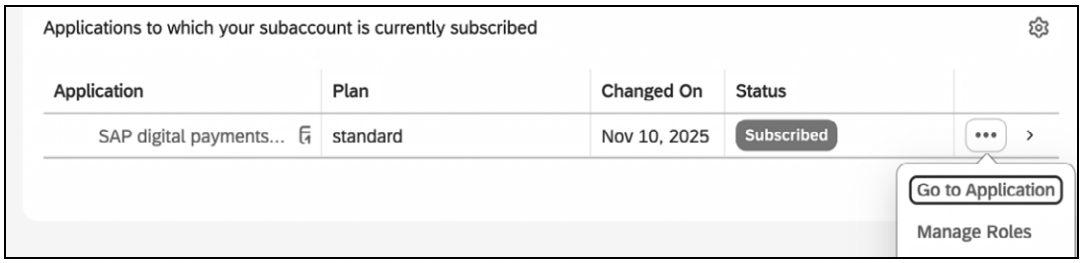

Figure 4.18: Add-On Subaccount: Go to Application Navigation

2. You'll be directed to the base application URL. The format depends on your SAP BTP region:
   - US10 Test:
     https://{subdomain}.test-digitalpayments-sap.cfapps.us10.hana.ondemand.com
   - EU10 Test:
     https://{subdomain}.demo-digitalpayments-sap.cfapps.eu10.hana.ondemand.com

   Here, *{subdomain}* is the subdomain you specified in Chapter 3 (e.g., *yourcompany-dp-test*). Figure 4.19 shows an example base application URL if your SAP BTP region is US10.

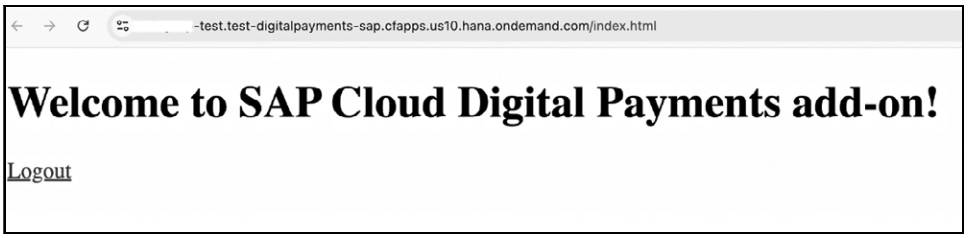

Figure 4.19: Add-On Subaccount: Base Application URL

3. Append */pspStatus/index.html* to the base URL to access the PSP status management UI, as shown in Figure 4.20.

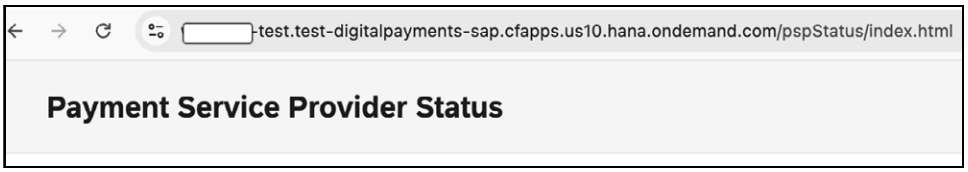

Figure 4.20: Add-On Subaccount: PSP Status Configuration UI

When accessing the PSP status UI, you may be prompted to authenticate through your identity provider, and your user must be assigned to a role collection containing the `DigitalPaymentsAdministrator` role. The UI displays all available adapters for your tenant. Figure 4.21 shows the typical view with multiple PSP adapters listed.

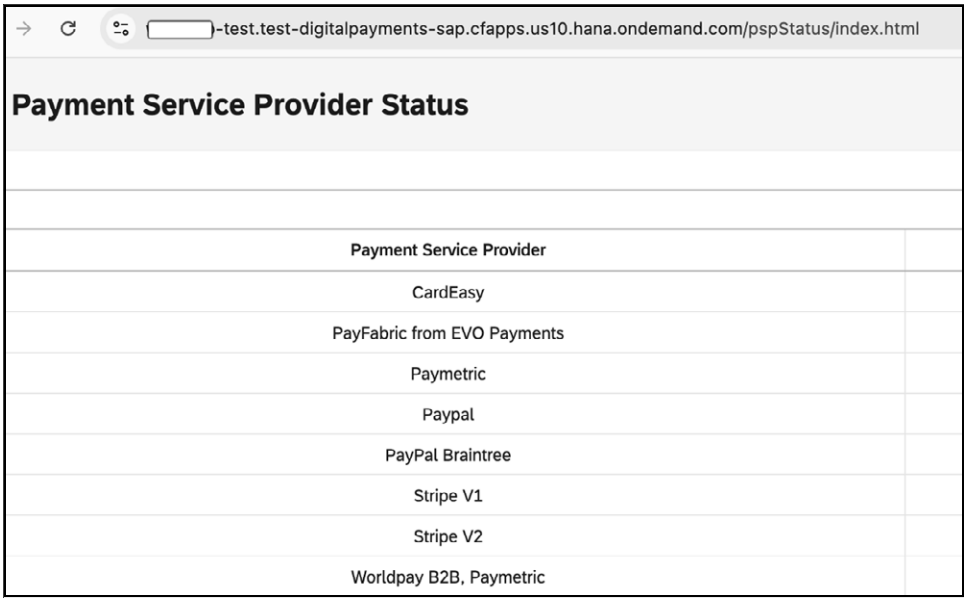

Figure 4.21: PSP Status: Configuration UI

Locate the row for Stripe V2 (initially shows as **Inactive**), click the **Activate** button, and confirm the activation in the dialog that appears, in which the status should be updated to **Active** (Figure 4.22).

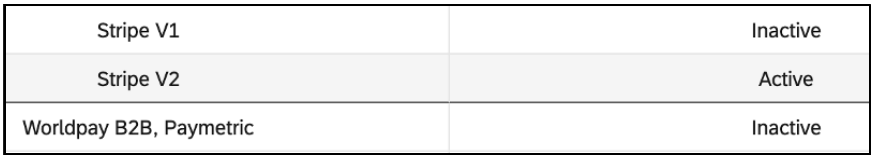

Figure 4.22: Add-On Subaccount: Stripe Adapter Activation

The activated status confirms that the routing engine will now consider Stripe when evaluating PSP determination rules, the Stripe adapter is available for merchant configuration, and transactions can be routed to Stripe once merchant accounts are onboarded. Activating the adapter does not automatically route transactions to Stripe. You must still configure PSP determination rules and merchant account mappings with your API credentials.

The production adapter activation follows an identical process with few environment-specific differences. Navigate to your DP Production subaccount, access the SAP digital payments add-on subscription, use the production base URL, append */pspStatus/index.html*, and activate Stripe V2 using the same procedure (see Figure 4.23).

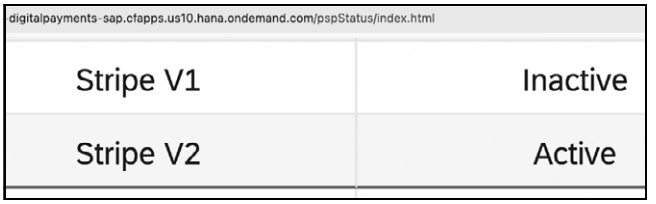

Figure 4.23: Add-On Production Subaccount: Stripe V2 Adapter Activation

Activating Multiple PSP Adapters

If your organization uses multiple PSPs, activate each adapter separately. You must repeat the activation process for each required PSP, and each adapter requires its own merchant account onboarding. Make sure that you configure routing rules to direct transactions to the appropriate PSP based on the transaction characteristics.

4.3.2 Configuration Parameters

After activating the Stripe adapter, you must onboard your merchant account(s) by configuring the API credentials you obtained (covered in Section 4.2.2). This establishes the actual connection between the digital payments add-on and your Stripe account, enabling payment processing. Stripe provides a dedicated web application for merchant configuration separate from the digital payments add-on configuration UIs. This is different from other PSPs, which use the standard digital payments add-on merchant mapping interface. We begin by accessing the Stripe configuration UI using region-specific URLs and authenticating with credentials generated during the tenant connection process. We then configure the merchant account by entering publishable and secret API keys, selecting the default currency for settlement, and optionally enabling Level 2/3 data transmission for optimized interchange rates. Next, we create merchant aliases that map Stripe technical merchant IDs to user-friendly identifiers used by consumer applications, enabling multimerchant scenarios across different business units and currencies. Finally, we configure PSP determination rules that route transactions to the appropriate merchant based on company code, currency, payment method, and other routing parameters, ensuring payments are processed by the correct merchant account.

Accessing the Stripe Configuration UI

The Stripe configuration UI application requires separate authentication credentials generated during the tenant connection process. Based on your SAP BTP region and environment, construct the appropriate URL. Table 4.13 shows example URLs for US and Europe that you can use as a reference.

SAP BTP Region and Environment	Stripe Configuration UI URL
US East (VA) test	*https://us-test-uiservices-dpa.sap.stripeconnectors.com/stripeConfiguration/index.html?subdomain={subdomain}*
US East (VA) production	*https://us-prod-uiservices-dpa.sap.stripeconnectors.com/stripeConfiguration/index.html?subdomain={subdomain}*
Europe (Frankfurt) test	*https://eu-demo-uiservices-dpa.sap.stripeconnectors.com/stripeConfiguration/index.html?subdomain={subdomain}*
Europe (Frankfurt) production	*https://eu-prod-uiservices-dpa.sap.stripeconnectors.com/stripeConfiguration/index.html?subdomain={subdomain}*

Table 4.13: SAP BTP Region and Environment: Stripe Configuration UI URLs

Replace *{subdomain}* with your SAP BTP subaccount subdomain (e.g., *yourcompany-dp-test*). For example, for the US10 test region, *https://us-test-uiservices-dpa.sap.stripeconnectors.com/stripeConfiguration/index.html?subdomain=yourcompany-dp-test.*

You will have access to the Stripe configuration UI for merchant onboarding. The merchant onboarding process connects your Stripe merchant account to the digital payments add-on tenant. You can configure multiple merchant IDs for a given digital payments add-on tenant. Figure 4.24 shows the Stripe configuration home page with existing example merchant configurations.

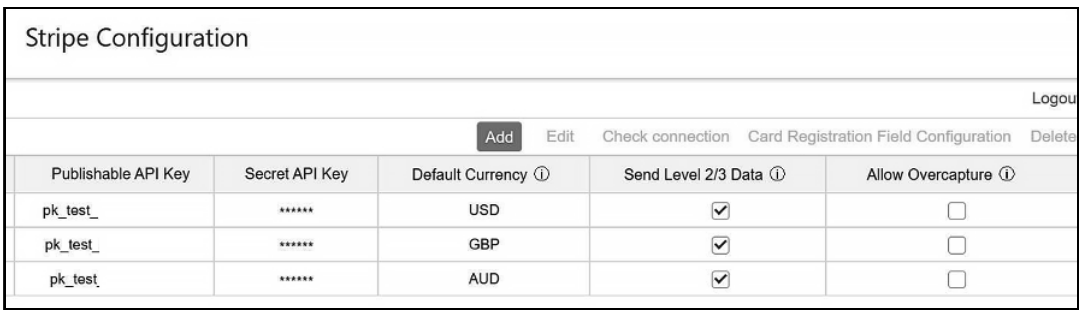

Figure 4.24: Stripe Merchant Configuration

You can add new merchants by clicking the **Add** button in the top-right corner. A configuration form appears, requesting the following information:

- **Publishable API Key**
  Enter the publishable key (obtained in Section 4.2.2).
- **Secret API Key**
  Enter the secret key (obtained in Section 4.2.2).
- **Default Currency**
  Select from the dropdown list of ISO currency codes; this represents the primary currency for this merchant account. This should match the settlement currency configured in your Stripe dashboard account settings.
- **Send Level 2/3 Data**
  Check this checkbox if you want to send Level 2/3 data to Stripe during payment capture. Level 2/3 data includes line-item details, tax amounts, and customer purchase order numbers and is required for optimized interchange rates on corporate and purchasing cards.

After you add a new merchant account, you can select the newly onboarded merchant from the list and click the **Check Connection** button at the top of the page. The system performs a live API call to Stripe using your credentials and a successful connection shows a message: **Connection to Stripe merchant account successful**.

Troubleshooting Merchant Onboarding Issues

The following are some possible causes if you observe errors such as invalid API credentials, and troubleshooting steps to follow:

- Credentials copied incorrectly: Recopy from Stripe dashboard, ensure no leading/trailing spaces.
- Wrong mode credentials: Verify test keys (pk_test_/sk_test_) are used in test environment, live keys (pk_live_/sk_live_) in production.
- Keys from wrong Stripe account: Confirm you're using credentials from the account where the SAP Digital Payments Add-On V2 app is installed.
- Keys revoked or rolled: Generate new keys from **Stripe Dashboard • Apps • SAP DPA V2 • App Settings**.

Another potential issue is that the merchant saves successfully but the check connection fails. Possible causes and solutions are as follows:

- Network connectivity: Verify outbound HTTPS (port 443) from SAP BTP to *api.stripe.com*.
- Stripe account restricted: Check the Stripe dashboard for account verification issues.
- API key permissions insufficient: Ensure the SAP Digital Payments Add-On V2 app installation granted all required permissions.
- Regional endpoint mismatch: Verify that your Stripe account region matches your SAP BTP region.

If you can't access the Stripe configuration UI, some possible causes and solutions are as follows:

- Incorrect subdomain in URL: Verify the subdomain matches your SAP BTP subaccount exactly.
- Wrong region prefix: Ensure *us-test*, *us-prod*, *eu-demo*, or *eu-prod* matches your environment.
- Authentication credentials not received: Check your spam folder for email from *noreply@verificationemail.com*.
- Tenant not onboarded: Verify that you completed the tenant connection as described in Section 4.2.3.

Configuring Merchant Aliases

After onboarding merchants in the Stripe configuration UI, you must create merchant aliases in the digital payments add-on merchant mapping interface to make these merchants available to consumer applications. When using multiple merchant IDs, you must define merchant ID aliases for both payment card payments and external payments. This is important for generating unique digital payment transaction advice for each merchant ID.

Follow the steps mentioned in Section 4.3.1, in which we walked through accessing the PSP status to get the base application URL of your add-on. Navigate to *{baseURL}/merchantMapping/index.html* to access the merchant mapping configuration UI. An example URL for US10 looks like this: *https://yourcompany-dp-test.test-digitalpayments-sap.cfapps.us10.hana.ondemand.com/merchantMapping/index.html*.

To create the merchant alias, follow these steps:

1. Before creating merchant aliases, it is a good practice to create a simple table like the one in Table 4.14. Here you can gather all the merchant accounts, with their corresponding technical merchant IDs and the user-friendly merchant aliases that will be configured and used in the consumer applications. We have gathered information to configure merchant aliases for a company that has three business units, one in North America (BU1), one in Europe (BU2), and one in Australia (BU3). We have also assumed that for BU2 we would like to support two different currencies: GBP and EUR.

PSP	Technical Merchant ID	Currency	Merchant Alias
Stripe V2	pk_test_1xxx…	USD	DPBU1NA01
Stripe V2	Pk_test_2xxx…	GBP	DPBU2EUR01
Stripe V2	Pk_test_3xxx…	EUR	DPBU2EUR02
Stripe V2	Pk_test_4xxx…	AUD	DPBU3APAC01

Table 4.14: Merchant Aliases: Example Table to Gather Required Details

2. In the merchant mapping UI, click **New Alias** and enter the following configuration:
   - **Payment Service Provider**: Stripe V2
   - **Merchant**: pk_test_1xxx… (publishable key from Stripe configuration UI)
   - **Merchant Alias**: DPBU1NA01 (user-friendly identifier used by consumer applications)
3. Click **Save**. Repeat the steps for all the four merchant alias entries. Figure 4.25 shows the merchant mapping UI with multiple aliases configured.

Payment Service Provider	Merchant	Merchant Alias
Stripe V2	pk_test_...	DPBU1NA01
Stripe V2	pk_test_...	DPBU2EUR01
Stripe V2	pk_test_...	DPBU2EUR02
Stripe V2	pk_test_...	DPBU3APAC01

Figure 4.25: Merchant Mapping: Example Company

These aliases are then used in PSP determination rules, virtual bank account configuration, payment advice reconciliation processes, and so on.

Configuring PSP Determination Rules

After merchant aliases are created, you must configure routing rules that direct transactions to the appropriate PSP and merchant. PSP determination is typically needed in these scenarios:

- Register a payment card: Identify the PSP to be used to tokenize the payment card.
- Authorize a payment card: Identify the PSP to be used to perform an authorization.
- Refund a payment: Determine the PSP's merchant account in certain refund cases in which the merchant account is not yet known.

Follow the steps mentioned in Section 4.3.1 to access the PSP status and get the base application URL of your add-on. Navigate to {*baseURL*}/ *pspDetermination /index.html* to access the PSP determination UI. An example URL for US10 looks like this: *https://yourcompany-dp-test.test-digitalpayments-sap.cfapps.us10.hana.ondemand.com/pspDetermination /index.html*.

Before configuring the routing rules, it is important to understand the mandatory and optional parameters that are supported and the sequencing and prioritization of the routing conditions (see Figure 4.26):

- **Sequence Number**
  This field determines the priority and sequence in which the routing condition is determined at runtime. In Figure 4.26, you can see that there are four routing conditions with sequence numbers 1 through 4. At runtime, the routing rules will be evaluated from 1

through 4. If you want to change the sequence of the routing conditions, you can use the **Move Up** or **Move Down** option.

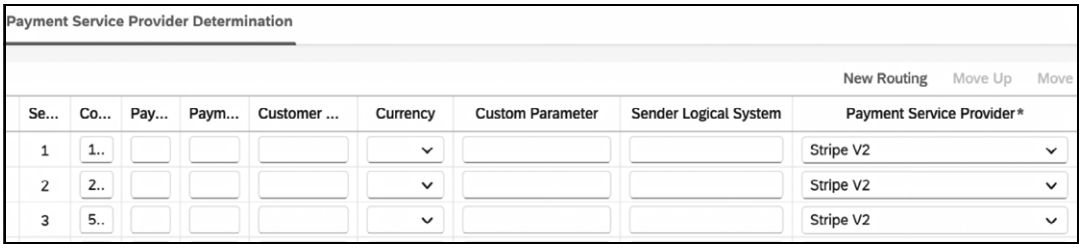

Figure 4.26: Routing Conditions: Mandatory and Optional Parameters

- **Company Code**
  Company code typically refers to your company's legal entity, which is legally registered to operate your business in a certain region. When an example organization has three business units, its corresponding company codes are configured in the consumer application as 1000 (BU1), 2000 (BU2), and 3000 (BU3). The company code is an optional parameter, but in most implementations you will see that this field is used to enforce a clear separation between legal entities and their corresponding payment transactions and bank accounts.
- **Payment Method**
  This is an optional parameter; you can choose **Credit Card (CC)** or **External Payment (EP)**. We will discuss payment methods and their significance further in Section 4.3.3.
- **Customer country/Region**
  This is an optional parameter in which you can specify two-character ISO country codes (e.g., US, DE, GB).
- **Currency**
  This is an optional parameter in which you can specify three-character ISO currency codes (e.g., USD, EUR, GBP, and AUD). Figure 4.27 shows an example configuration in which a company with three business units wants to support four currencies for payment transactions using Stripe as the PSP.

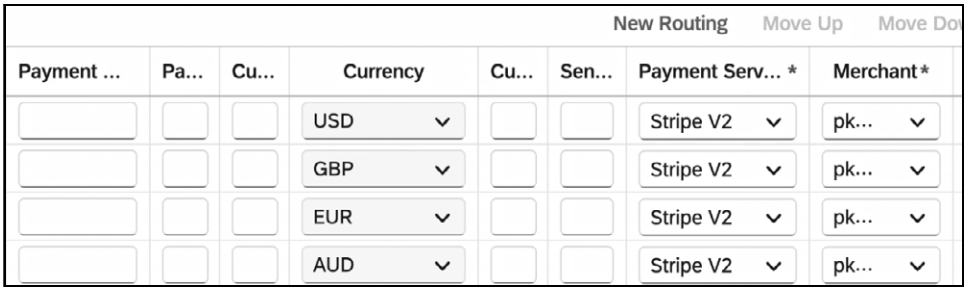

Figure 4.27: PSP Determination: Currency Codes Usage

- **Custom Parameter**
  This is an optional, user-defined routing parameter that can be used to specify routing conditions in special cases in which your implementation demands that a user-defined parameter be used to route payment transactions, rather than the standard parameters.
- **Sender Logical System**
  Sender Logical System serves as a technical identifier for the consumer application system and may function as a routing parameter. The value format is contingent upon the specific consumer application. Details for deriving the sender logical systems are as follows:
  - **SAP S/4HANA**
    The sender logical system is constructed by concatenating the backend system's installation number with the logical system ID. To locate the installation number, navigate to **System • Status** and reference the **Installation Number** field. For the logical system ID, use Transaction SE16, enter table `T000`, select **Table Contents**, then select **Execute**. Double-click the relevant client; the logical system ID can be found in the **LOGSYS** field.
  - **SAP S/4HANA Cloud Public Edition**
    In this case, the sender logical system corresponds to the logical system ID. Access your system with administrator credentials, search for "Communication Arrangements", and select the arrangement associated with the SAP digital payments add-on integration (scenario ID SAP_COM_0216). The logical system ID is listed in the **Own SAP Cloud System** field within the **Common Data** section.
- **Payment Service Provider**
  This and the corresponding technical merchant ID are mandatory parameters in the routing conditions. You will see that only PSPs whose adapters have been already activated are visible in the payment service provider determination user interface. Make sure you choose the correct technical merchant ID along with the PSP so that the transactions are routed to the correct merchant account for processing.

The table entries are evaluated using these guidelines:

- Rows are checked from top to bottom, one at a time.
- The first row that matches is used to select the PSP or merchant account.
- Every parameter listed in a row must match exactly for that row to be chosen.
- Empty cells act as wildcards and will match anything. However, a cell containing a space character does not count as empty.
- If none of the rows match, an error message will indicate that no PSP was found.

Only fill in a field if you want it to influence PSP selection; otherwise, leave it blank. It's best to include a final catchall row at the bottom with your default PSP and all other fields left empty. This ensures that transactions go through even if nothing else matches.

4.3.3 Payment Method Setup

With merchant accounts onboarded, the next configuration step defines which payment methods are available for transaction processing. The Stripe V2 adapter supports over 40 payment methods through Stripe's Payment Elements framework, but not all methods are automatically enabled.

The following payment methods are available by default after merchant onboarding (no additional configuration required):

- Credit cards: Visa, Mastercard, American Express, Discover, JCB, Diners Club, UnionPay
- Debit cards: All major debit card networks
- Digital wallets: Apple Pay, Google Pay, Link (Stripe's native wallet)

These card-based methods work immediately after merchant onboarding because they use the standard card-processing flow.

The SAP digital payments add-on enables support for *external payments*, which refers to transactions handled through external channels such as bank debits or redirects instead of payment cards. You can process external payments using either a one-step process (*direct capture*), in which funds are charged straight to the payer's account without prior authorization, or a two-step process (*manual capture* or separate authorization and capture), in which the payment is first authorized before being charged.

The Stripe V2 connector supports the following business processes:

- **External payment direct capture**
  When a user places an order and opts to pay through an external payment channel, approval for the transaction must be secured from the PSP. The consumer application submits a direct capture request to the SAP digital payments add-on. Depending on the selected payment type, the appropriate PSP is identified, and the initiation request is transmitted in the required format. The user then authorizes the direct capture using the PSP's interface, which is called by the consumer application. Upon receiving authorization, the execution request for direct capture is issued, and the resulting outcome is returned to the consumer application through the SAP digital payments add-on.
- **External payment capture with authorization**
  This process has the following elements:
  - **External payment authorization**
    If a user chooses to pay for an order via an external payment channel, transaction approval from the PSP is necessary. The consumer application dispatches an authorization request to the SAP digital payments add-on. Following this, the corresponding PSP is selected according to payment type, and the initiation request is sent in the prescribed format. Approval is granted by the user through the PSP's interface, accessed by the consumer application. The execution request for authorization is sent

upon approval, and the result is relayed back to the consumer application via the SAP digital payments add-on.

- **External payment authorization cancellation**
  Should a user cancel an order that has already received payment approval, the consumer application issues an authorization cancellation request to the SAP digital payments add-on. Based on payment type, the relevant PSP is identified, and the request is sent in the correct format. The PSP processes the cancellation and communicates the result to the consumer application through the SAP digital payments add-on. If the cancellation cannot be executed, the consumer application is notified accordingly.
- **External payment settlement**
  The consumer application periodically initiates settlement runs to finalize authorized external payments. The authorization data is sent to the SAP digital payments add-on, which identifies the relevant PSP and forwards the settlement request in the appropriate format. The PSP completes the settlement process and returns the outcome to the consumer application via the SAP digital payments add-on.

- **External payment refund**
  In instances in which a settled external payment must be refunded—such as product returns—the consumer application requests a refund, citing reference information for the original payment. The suitable PSP is determined, and the refund data is transmitted in the required format. The PSP processes the refund and returns the result to the consumer application through the SAP digital payments add-on.
- **Digital payment advice**
  To obtain the details of processed digital payments, the consumer application regularly executes reports requesting advice information from the SAP digital payments add-on. The add-on collects advice details from all applicable PSPs, consolidates the information, and delivers it to the consumer application.
- **External payment authorization preparation**
  If the consumer application sends an external payment authorization request directly to the PSP, the PSP processes and returns the result straight to the consumer application. To facilitate subsequent processing within other systems, the consumer application communicates the external payment authorization to the SAP digital payments add-on via a preparation request. The authorization is recorded by the add-on, and an authorization reference is supplied to the consumer application.
- **External payment direct capture preparation**
  For external payments captured directly at the PSP, the PSP processes the capture and returns the result directly to the consumer application. To support further system integration, the consumer application submits the capture information to the SAP digital payments add-on through a direct capture preparation request. The add-on stores the direct capture, providing a corresponding payment reference to the consumer application.

4.4 Payment Methods and Currency Configuration

Once the adapter is connected and the merchant account is onboarded, the next important step is defining how customers can pay. The Stripe V2 adapter for the SAP digital payments add-on supports a wide range of payment methods beyond standard credit cards, including digital wallets; buy now, pay later (BNPL) options; and other payment schemes. This section guides you through the supported payment methods, the architectural differences between direct capture and authorization, and the specific configuration required to handle multicurrency settlements and Level 2/3 data processing. We first review the complete catalog of Stripe payment methods, from card-based options to bank redirects to direct debits, and explain which methods support two-step authorization/capture versus direct capture only; this distinction determines your order-to-cash process design. We then configure multicurrency support by creating distinct merchant aliases for each currency you need to process, enabling your global organization to settle funds into different bank accounts without conversion fees. Next, we walk through how to enable Level 2/3 data transmission to reduce interchange costs on corporate and purchasing cards. Finally, we activate specific payment methods in both the Stripe dashboard and the payment page configuration UI to make them available to customers.

4.4.1 Stripe Payment Methods Overview

The Stripe V2 adapter leverages Stripe's PaymentIntents API architecture. This allows it to support payment methods that require complex authentication flows (like 3D Secure) or asynchronous confirmation (like bank redirects). However, not all payment methods behave the same way within the order-to-cash process. It is critical to distinguish between methods that support authorization and capture (two steps) and those that require direct capture (one step), as follows:

- **Card-based payments (two steps or one step)**
  These are the most flexible methods. They support preauthorization (holding funds before shipping), which is the standard requirement for SAP S/4HANA sales order processing. This includes the following payment methods:
  - Credit/debit cards: Visa, Mastercard, American Express, Discover, Diners Club, JCB, and UnionPay
  - Digital wallets: Apple Pay and Google Pay (treated as *wallet cards* in the digital payments add-on)
- **BNPL**
  These payment methods allow customers to finance purchases. The integration behavior varies by provider:
  - Klarna/Afterpay/Clearpay: Support authorization and capture. You can authorize the funds at order creation and capture them upon delivery.
  - Affirm: Typically supports direct capture only in this integration.

- **Bank redirects and real-time payments (direct capture only)**
  These methods require the customer to log into their bank environment to approve the payment. Because the funds are pushed immediately, they do not support preauthorization. If you use these methods, your SAP process must handle immediate settlement documentation. These methods include the following:
  - European local methods: iDEAL (Netherlands), Bancontact (Belgium), Giropay (Germany), EPS (Austria), P24 (Poland)
  - Asian wallets: Alipay and WeChat Pay
- **Direct debits (asynchronous)**
  - Schemes: ACH (USA), SEPA (Europe), BACS (UK), BECS (Australia), and ACSS (Canada).
  - Behavior: Unlike cards, these methods have a settlement delay. The status may remain Pending for several days before turning to Success or Failed.

Use Table 4.15 to plan which digital payments add-on scenario to use for each method.

Payment Method	Support
Credit/debit cards (Code: CC)	Supports direct capture and separate auth/capture.
Apple Pay / Google Pay (Code: WC)	Supports direct capture and separate auth/capture.
PayPal (Code: PP)	Supports direct capture and separate auth/capture.
Klarna (Code: KL)	Supports direct capture and separate auth/capture.
Afterpay/Clearpay (Code: A2)	Supports direct capture and separate auth/capture.
iDEAL (Code: ID)	Supports direct capture only.
Bancontact (Code: BA)	Supports direct capture only.
Giropay (Code: G1)	Supports direct capture only.
Alipay (Code: AP)	Supports direct capture only.
ACH/eCheck (Code: AC)	Supports direct capture only.

Table 4.15: Payment Methods and Support for Two-Step and One-Step Payments Process

4.4.2 Currency Support and Multimerchant Strategy

Global organizations often require multiple currencies (e.g., USD, EUR, and GBP) to settle funds into different bank accounts without incurring conversion fees. The SAP digital payments add-on handles this through merchant aliases.

When you onboard a merchant in the Stripe configuration UI, you will typically select a default currency (e.g., USD). This selection tells the adapter which currency to use for reporting and default settlement behavior for that specific configuration entry.

If your consumer application (e.g., SAP S/4HANA) needs to process payments in multiple currencies, you cannot simply use one generic merchant alias. You must create a distinct configuration for each currency, even if they all point to the same underlying Stripe account.

Stripe treats certain currencies (like JPY, KRW, or CL) as zero-decimal currencies. While the digital payments add-on core handles the normalization of amounts (e.g., converting 1,000 JPY to the correct minor unit integer for Stripe), you should ensure your SAP S/4HANA currency decimal configurations (Transaction OY04) match the ISO standards to prevent settlement errors.

4.4.3 Level 2 and Level 3 Data Configuration

For B2B transactions, passing additional data fields to the card issuer can significantly reduce interchange fees. This is known as Level 2 (L2) and Level 3 (L3) data processing:

- Level 2 data includes the tax amount and customer reference code.
- Level 3 data includes line-item details such as product codes, unit quantities, unit of measure, and discount amounts.

The Stripe V2 adapter has a built-in capability to map SAP billing data to Stripe's L2/L3 fields:

- Open the Stripe configuration UI (refer to Section 4.3.2 for detailed, step-by-step instructions to navigate to the Stripe configuration UI).
- Locate your merchant configuration.
- Check the box labeled **Send Level 2/3 Data**.
- Click **Save**.

Note that this feature only works if the upstream consumer application (e.g., SAP S/4HANA) sends the line-item data in the payload to the digital payments add-on. If SAP S/4HANA sends a summary total only, the adapter cannot fabricate L3 data. Also make sure that your Stripe account is configured to accept L3 data.

4.4.4 Mail Order/Telephone Order and Offline Mandates Support

Many companies operate call centers in which agents take card details over the phone (a mail order/telephone order [MOTO] process) or process offline mandates for recurring billing. The Stripe V2 adapter includes specific support for these scenarios through virtual terminals.

MOTO transactions are exempt from SCA because the cardholder is not present to perform biometric or one-time passcode (OTP) verification. To enable this in the digital payments add-on, perform the following tasks:

- Enable the feature with Stripe: You can contact Stripe Support to enable MOTO processing on your Stripe account. By default, Stripe may decline high volumes of MOTO transactions as fraud risks.

- Configure the digital payments add-on: When the call center agent enters the card in the SAP system, the digital payments add-on adapter flags the transaction as MOTO.
- Virtual terminal: The adapter utilizes a specialized card registration virtual terminal that allows the agent to manually key in the primary account number (PAN) and card verification code (CVC) without triggering 3D Secure.

For recurring payments (like subscriptions), you may collect a physical signature from a customer authorizing you to debit their account. This is an offline mandate, and the following should be noted:

- The adapter supports flagging these transactions, so Stripe knows that a physical agreement exists.
- This is important for SEPA Direct Debit to avoid chargebacks.

4.4.5 Enable Specific Methods in the Add-On

Although the Stripe adapter technically supports the methods listed in Section 4.4.1, they will not appear on your checkout page until you explicitly configure them in the SAP digital payments add-on. This is done using the Payment Page Configuration app. Begin by navigating to the Payment Page Configuration interface and selecting or creating a configuration for your region. You will then activate individual payment methods, assigning each method to the Stripe V2 adapter, selecting whether to use authorization or direct capture transaction types, and assigning merchant aliases and weighting values to control the visual display order. Finally, you will enable those same payment methods in the Stripe dashboard, ensuring that both the add-on configuration and the PSP account are aligned before customers can use these payment options.

Let's look at these processes in more detail.

Enabling Payment Methods

Follow these steps to navigate to the user interface where you can enable the payment types and transaction types and assign them to the correct PSP:

1. Log into the SAP BTP cockpit (*https://cockpit.btp.cloud.sap*), navigate to your DP Test subaccount, and select **Services • Instances and Subscriptions** from the left navigation pane. Under **Subscriptions,** locate the SAP digital payments add-on demo subscription, click **…**, and select the **Go to Application** link as shown in Figure 4.28.

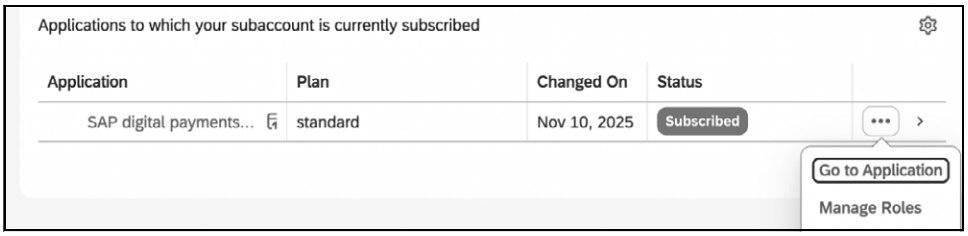

Figure 4.28: DP Test Subaccount: Application Link Navigation

2. You'll be directed to the base application URL. The format depends on your SAP BTP region:
   - US10 Test:
     https://{subdomain}.test-digitalpayments-sap.cfapps.us10.hana.ondemand .com
   - EU10 Test:
     https://{subdomain}.demo-digitalpayments-sap.cfapps.eu10.hana.ondeman .com

   Here, *{subdomain}* is the subdomain you specified in Chapter 3 (e.g., *yourcompany-dp-test*). Figure 4.29 shows an example base application URL if your SAP BTP region is US10.

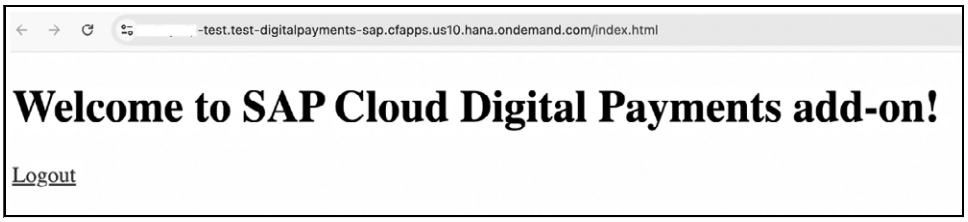

Figure 4.29: Add-On Application: Home Page

3. Append */paymentPageConfiguration/index.html* to the base URL to access the PSP status management UI, as shown in Figure 4.30.

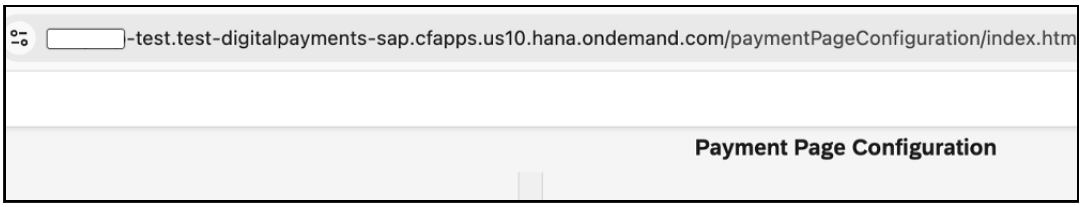

Figure 4.30: Add-On: Payment Page Configuration URL

4. Select an existing configuration or click **Create** to start a new one. Figure 4.31 shows an example existing configuration for the EMEA region.

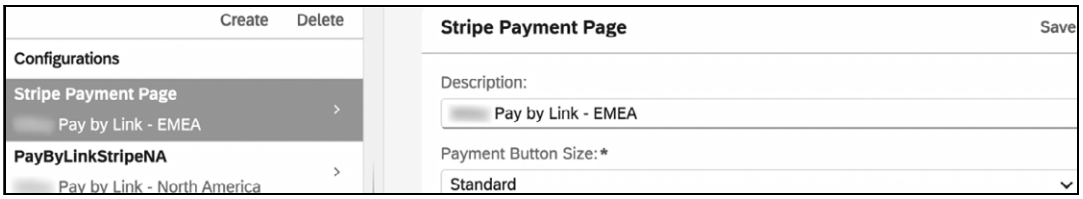

Figure 4.31: Payment by Link

5. You will see a list of generic **Digital Payment Types** (e.g., payment card, PayPal, Klarna, Apple Pay). Check the **Active** box for the desired method; for this example, this will be Klarna. In the **Payment Service Provider** column, select **Stripe V2** (see Figure 4.32).

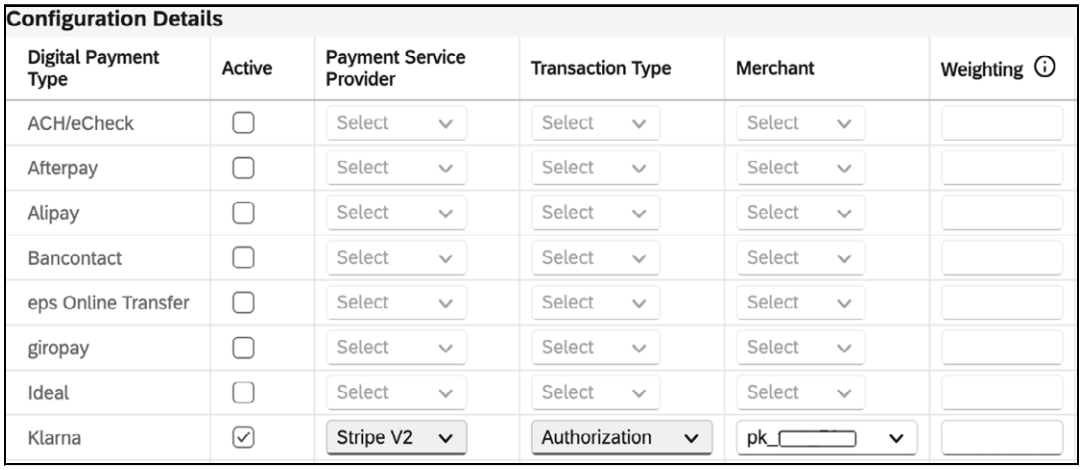

Figure 4.32: Payment Type Activation: Klarna Example

6. Select either **Authorization** (to reserve funds) or **Direct Capture** (for immediate charge) as the **Transaction Type**, as shown in Figure 4.33.

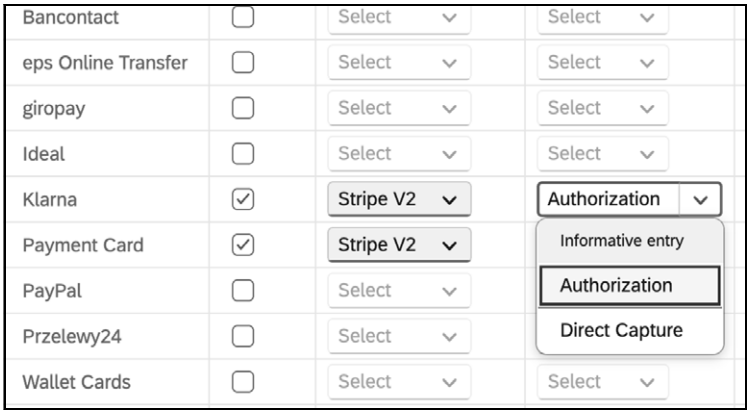

Figure 4.33: Payment Page Configuration: Transaction Types

7. Assign the **Merchant**. Select the merchant alias to be used and, for the weighting, assign a number (0–99) in the **Weighting** column (see Figure 4.34). This determines the visual sort order of the buttons on the checkout page (higher numbers appear first).
8. Now you are ready to process the Klarna payment type.

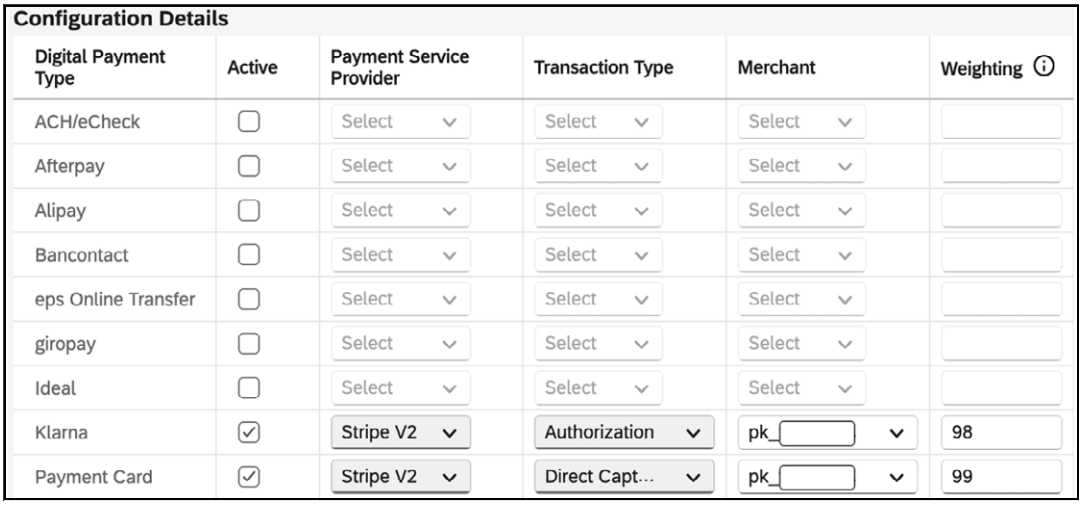

Figure 4.34: Payment Page Configuration: Merchant Alias and Weighting Assignment

Enabling Payment Methods in Stripe Dashboard

Although the Stripe adapter supports multiple payment methods, they must be explicitly enabled in your Stripe account before becoming available for transaction processing, as follows:

1. Log into the Stripe dashboard (*dashboard.stripe.com*) and ensure you're in the correct mode using the mode toggle: either test mode (test subaccount integration) or live mode (production subaccount integration). Navigate to **Settings • Payments** (see Figure 4.35).

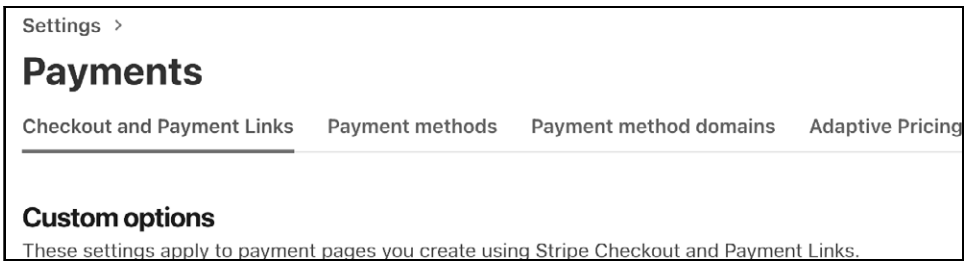

Figure 4.35: Stripe Payment Methods: Navigation UI

2. Review the currently enabled methods. The **Payment methods** page displays four sections (see Figure 4.36):
   - **All**: These are payment methods you can enable based on your account country and business type.
   - **Enabled:** These are the payment methods currently active for your account.
   - **Disabled**: These are the payment methods currently inactive for your account.
   - **Requires action**: Some payment methods may require follow-up actions. If so, they will be listed here.

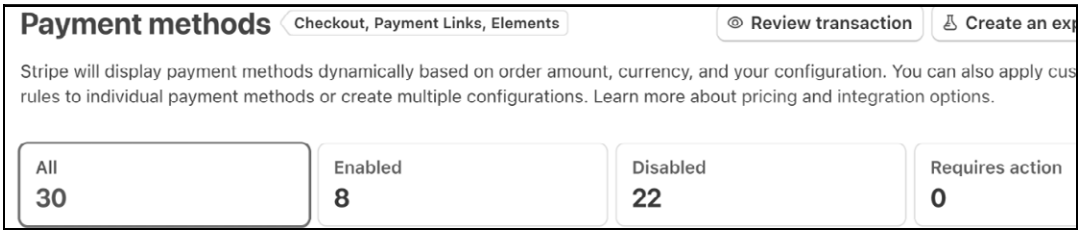

Figure 4.36: Stripe Payment Methods UI

3. Most new Stripe accounts have some payment methods enabled by default (e.g., card payments, Apple Pay, Link). Figure 4.37 shows an example Stripe account.

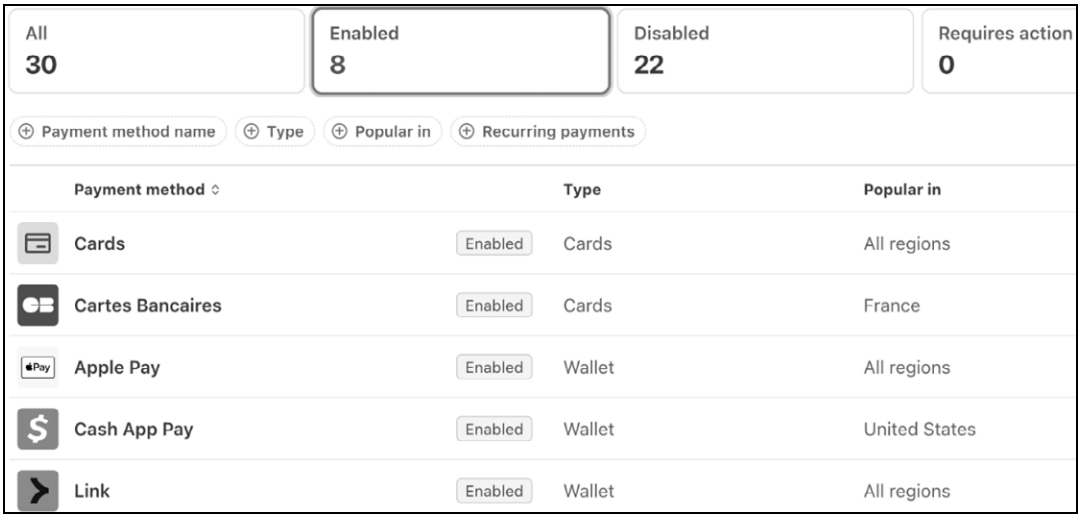

Figure 4.37: Default Payment Methods: Stripe Payments

4. To enable additional wallet payments, scroll to the **Disabled** section (Figure 4.38) and locate them (e.g., Alipay, Amazon Pay). Select Alipay from the list of **Payment method**s.

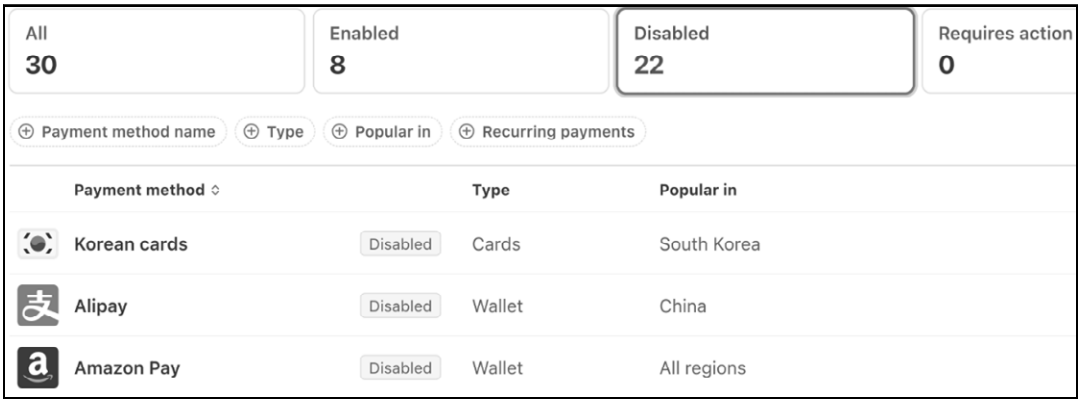

Figure 4.38: Stripe Payment Methods: Inactive Example

5. To enable Alipay as a new payment method. Click the **Turn on** button (see Figure 4.39). Once turned on, the payment method immediately moves to the **Enabled** section and becomes available for processing.

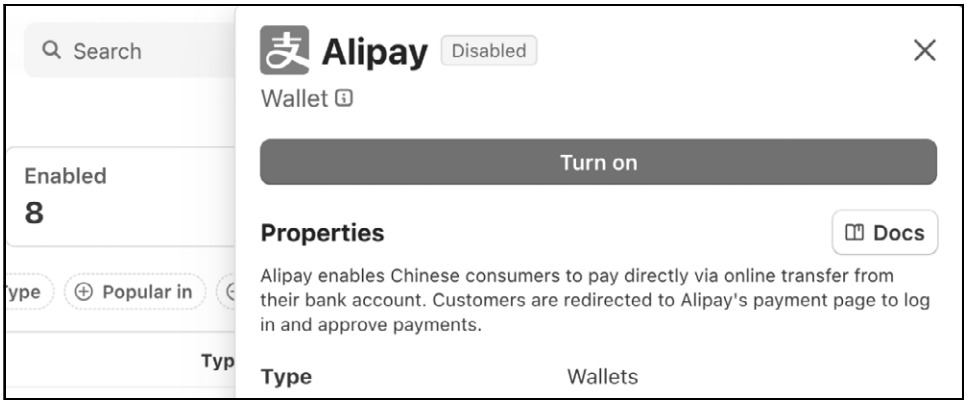

Figure 4.39: Alipay Payment Method: Enablement

6. Let's look at another example, for SEPA Direct Debit (for European Economic Area [EEA] merchants). Like you did for Alipay, locate **SEPA Direct Debit** in the **Disabled** payments method section and click **Turn on**. You will see that it is now enabled (see Figure 4.40).

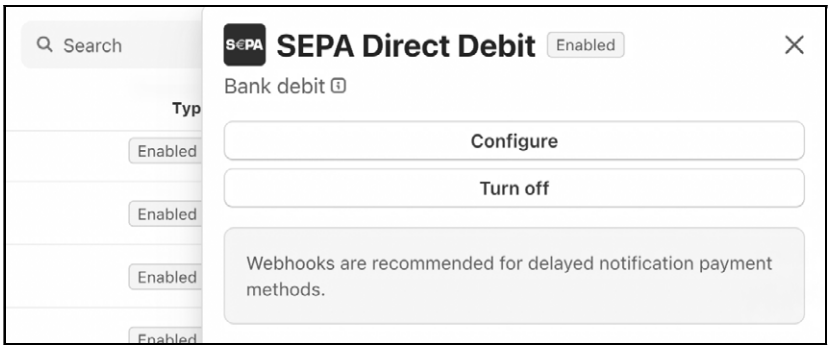

Figure 4.40: SEPA Direct Debit Enrollment

7. If required in your jurisdiction, configure your creditor **Identifier** (see Figure 4.41) by selecting the **Configure** button.

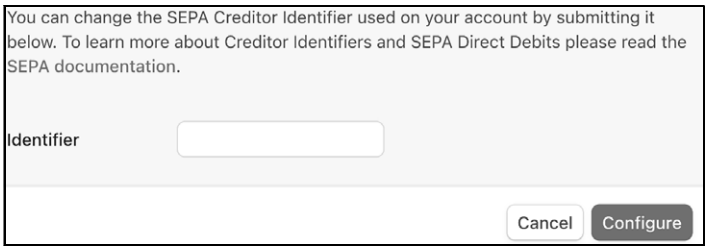

Figure 4.41: SEPA Direct Debit: Configure Creditor Identifier UI

4.5 Webhook Setup and Event Management

Payment processing communication is typically asynchronous. While the SAP digital payments add-on initiates transactions, the final status is often determined later by the PSP. This is particularly true for redirect methods (like 3D Secure or iDEAL) in which the customer must perform actions on a banking page. To handle these updates, the Stripe V2 adapter utilizes webhooks. A *webhook* is a notification sent from Stripe to the digital payments add-on whenever an event occurs (e.g., `payment_intent.succeeded`).

The Stripe V2 adapter utilizes a thin event architecture to ensure performance and reliability, as follows:

- Notification: Stripe sends a lightweight notification to the adapter containing just the event ID and object ID.
- Verification: The adapter validates the signature to ensure the request is genuine.
- Retrieval: The adapter calls back to the Stripe API to fetch the full, latest status of the object. This prevents race conditions that might cause a webhook to contain stale data.
- Normalization: The status is mapped to SAP standard codes (e.g., P for pending or S for success) and pushed to the digital payments add-on core.

For the integration to function, the adapter must subscribe to specific Stripe events. The SAP Digital Payments Add-On V2 app automatically registers these during installation, but you should verify they are active. Table 4.16 lists the critical events.

Event Name	Process	Description
`payment_intent.succeeded`	Authorization/ capture	Confirms funds are authorized or captured. Updates the digital payments add-on token status to **Success**.
`payment_intent.payment_failed`	Exception handling	Signals a decline. The adapter maps the Stripe decline code to an SAP error code.
`payment_intent.requires_action`	3D Secure	Signals that the customer must perform SCA.
`charge.refunded`	Refund	Confirms a refund has processed. Essential for updating financial accounting.
`payment_method.attached`	Card registration	Confirms a new card/wallet has been saved to a customer profile.
`mandate.updated`	SEPA/direct debit	Tracks changes to the digital mandate required for bank debits.

Table 4.16: Stripe PSP: Webhook Events List

When the SAP Digital Payments Add-On V2 app is installed, it programmatically creates a webhook endpoint in your Stripe account that points to the digital payments add-on infrastructure. You can verify that the webhook is active and working by following this verification procedure:

1. Log into the Stripe dashboard (ensure you are in the correct mode, either test or live) and navigate to **Settings • Developers • Webhooks**.
2. Look for an endpoint URL that matches the digital payments add-on connector domain:
   - US Region: *https://us-prod-gateway.sap.stripeconnectors.com*
   - EU Region: *https://eu-prod-gateway.sap.stripeconnectors.com*
3. Validate the status; ensure that it is **Enabled**.
4. Check the event subscriptions. Click the endpoint and ensure the events listed in Table 4.16 are present.

In payment processing, network timeouts can cause uncertainty about whether the charge request failed or the response was just lost. If the consumer application retries a charge request that succeeded the first time, the customer could be double-charged. Stripe supports idempotency keys to solve this. An *idempotency key* is a unique value generated by the client (in this case, the digital payments add-on adapter) and sent with the API request.

The digital payments add-on handles idempotency as follows:

- Key generation: When a consumer application initiates a payment, the digital payments add-on adapter generates a universally unique identifier (UUID) for that specific transaction attempt.
- Stripe check: When Stripe receives the request, it checks if it has already seen a request with that specific idempotency dey in the last 24 hours.
- Replay: If Stripe recognizes the key, it does not create a new charge. Instead, it simply replays the original response (e.g., **Success** or **Decline**).

This mechanism ensures that even if the digital payments add-on encounters a network error and automatically retries the API call, the customer is never charged twice for the same order.

The digital payments add-on can periodically fetch a settlement file from Stripe, which consumer applications can use to clear open items in accounts receivable. The Stripe V2 Adapter includes a dedicated user interface, the Stripe advice UI, to manage this process:

1. Log into the SAP BTP cockpit (*https://cockpit.btp.cloud.sap*), navigate to your DP Test subaccount, and select **Services • Instances and Subscriptions** from the left navigation pane. Under **Subscriptions**, locate the SAP digital payments add-on demo subscription, click **…**, and select the **Go to Application** link as shown in Figure 4.42.

Figure 4.42: Add-On Subaccount: Go to Application Link

2. You'll be directed to the base application URL. The format depends on your SAP BTP region:
   - US10 Test:
     https://{subdomain}.test-digitalpayments-sap.cfapps.us10.hana.ondemand. com
   - EU10 Test:
     https://{subdomain}.demo-digitalpayments-sap.cfapps.eu10.hana.ondemand. com

   Here, *{subdomain}* is the subdomain you specified in Chapter 3 (e.g., *yourcompany-dp-test*). Figure 4.43 shows an example base application URL if your SAP BTP region is US10.

Figure 4.43: Add-On Application: Home Page

3. Append */stripeAdvice/index.html* to the base URL to access the Stripe advice UI, as shown in Figure 4.44.

Figure 4.44: Stripe Advice UI

The scheduler allows you to automate the retrieval of settlement data. Instead of manually downloading spreadsheets from Stripe, the adapter pulls the data and pushes it into the

digital payments add-on core, which then forwards it to consumer applications (e.g., SAP S/4HANA). When setting up a schedule, you must define the following parameters:

- **Recurrence Pattern**: Defines the frequency of the job, as follows:
  - **Daily**: Runs every day. Best for high-volume, B2C businesses.
  - **Weekly**: Runs on specific days (e.g., every Monday).
  - **Monthly**: Runs on specific dates (e.g., the 1st and 15th of the month).
- **Time Slot** (UTC): Specifies the exact time the job should run. It is critical to set this time for after Stripe has closed its daily settlement batch (usually midnight UTC) to ensure you capture the full day's data.
- **Starting Date**: The date from which the scheduler should become active.

To set a daily schedule, follow these steps:

1. Navigate to the **Advice Fetching Configuration** section (see Figure 4.45).

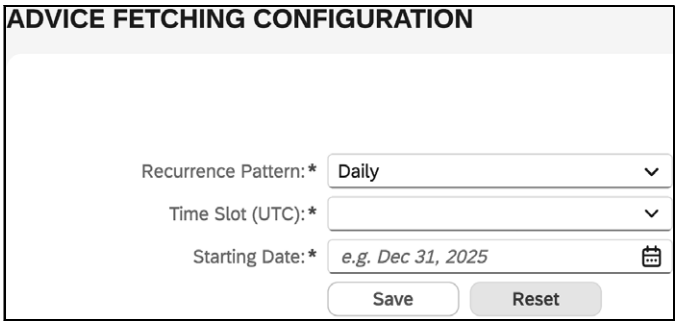

Figure 4.45: Advice Fetching Configuration UI

2. Set **Recurrence Pattern** to **Daily** and select a **Time Slot** (e.g., **02:00 - 03:59** UTC). This ensures the previous day's settlements are ready (see Figure 4.46).

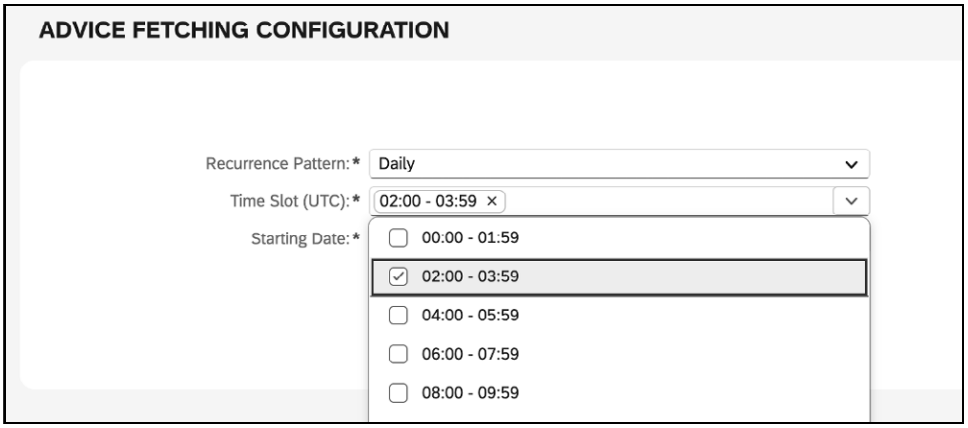

Figure 4.46: Advice Fetching Configuration: Daily Fetch Pattern Example

3. Set the **Starting Date** to today (e.g., December 4), as shown in Figure 4.47.

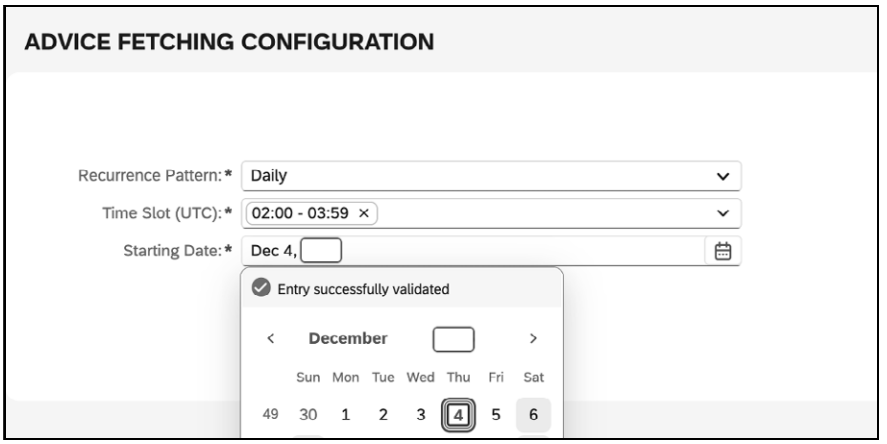

Figure 4.47: Fetch Payment Advice: Choosing Current Date Example

4. Click **Save** (see Figure 4.48).

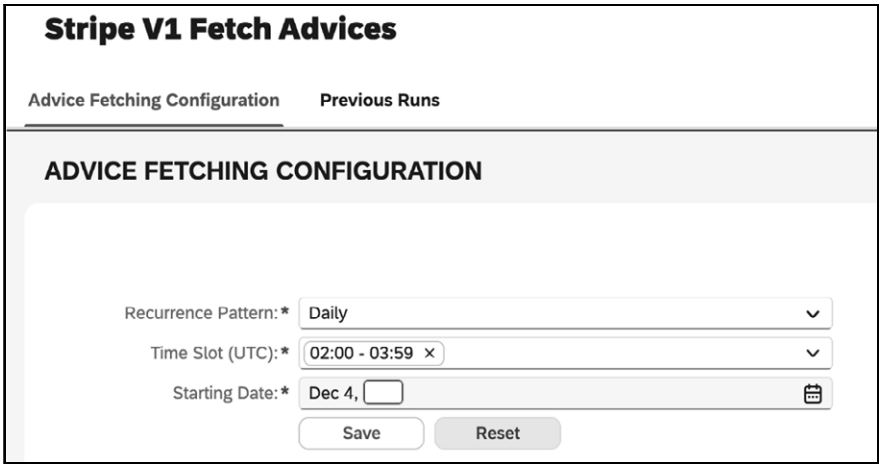

Figure 4.48: Advice Fetching Configuration: Daily Schedule Example

For testing purposes or period-end closing, you may need to fetch advice data immediately without waiting for the scheduler. Click the **Run Immediately** button in the top-right corner of the advice UI (see Figure 4.49).

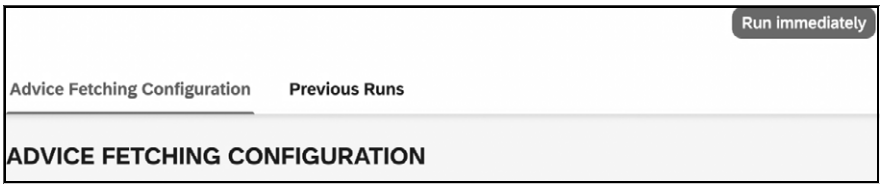

Figure 4.49: Stripe Advice: Ad Hoc Run

The system will trigger a background job to fetch the data. You can view the status in the **Previous Runs** table on the main dashboard. A green status icon indicates the advice was successfully fetched and handed off to the digital payments add-on core (see Figure 4.50).

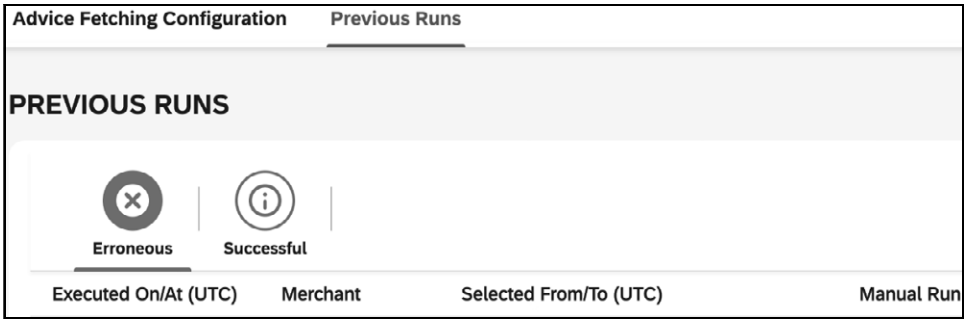

Figure 4.50: Stripe Advice Schedule Viewer UI

4.6 Testing PSP Integration

Stripe provides a comprehensive set of test card numbers that simulate various payment scenarios without processing real transactions or charging actual cards. These test cards work exclusively in test mode and enable validation of both successful payments and various decline scenarios.

Implementation teams typically have three broad categories of test scenarios: one for success or happy path payment card tests, one for error or decline scenarios; and one for certain regions that enforce additional authentication steps, like 3D Secure.

Table 4.17 lists standard test cards that can be used for testing success scenarios.

Card Number	Brand	CVC	Expiry	3D Secure Required	Test Scenario
4242 4242 4242 4242	Visa	Any three digits	Any future date	No	Successful payment
5555 5555 5555 4444	Mastercard	Any three digits	Any future date	No	Successful payment
3782 822463 10005	American Express	Any four digits	Any future date	No	Successful payment
6011 1111 1111 1117	Discover	Any three digits	Any future date	No	Successful payment
3056 9300 0902 0004	Diners Club	Any three digits	Any future date	No	Successful payment

Table 4.17: Stripe PSP: Standard Test Cards

Card Number	Brand	CVC	Expiry	3D Secure Required	Test Scenario
3566 0020 2036 0505	JCB	Any three digits	Any future date	No	Successful payment
6200 0000 0000 0005	UnionPay	Any three digits	Any future date	No	Successful payment

Table 4.17: Stripe PSP: Standard Test Cards (Cont.)

Table 4.18 lists test cards that can be used for negative test cases, so that implementation teams can test how the application behaves when card processing errors out.

Test Card Number	Decline Reason	API Error Code	Test Scenario
4000 0000 0000 0002	Generic decline	`card_declined`	Generic decline (no specific reason)
4000 0000 0000 9995	Insufficient funds	`insufficient_funds`	Customer account has insufficient balance
4000 0000 0000 9987	Lost card	`lost_card`	Card reported lost by cardholder
4000 0000 0000 9979	Stolen card	`stolen_card`	Card reported stolen by cardholder
4000 0000 0000 0069	Expired card	`expired_card`	Card expiration date has passed
4000 0000 0000 0127	Incorrect CVC	`incorrect_cvc`	CVC verification failed
4000 0000 0000 0119	Processing error	`processing_error`	Generic PSP processing error
4000 0000 0000 3220	3D Secure required	`authentication_required`	3DS authentication must be completed

Table 4.18: Stripe PSP: Error and Decline Test Cards

With the enforcement of SCA in Europe and other regions, testing the 3D Secure flow is important. Table 4.19 lists test cards that simulate 3D Secure–enabled card processing steps.

Test Card Number(s)	3D Secure Behavior	Test Scenario
4000 0027 6000 3184	3D Secure required; authentication succeeds	Successful SCA completion
4000 0082 6000 3178	3D Secure required; authentication fails	Customer abandons or fails authentication
4000 0025 0000 3155	3D Secure supported but not required	Frictionless flow (no challenge)
4242 4242 4242 4242	3D Secure not required	Standard authorization (non-EEA merchant)

Table 4.19: 3D Secure Test Cards

Table 4.20 lists common issues implementation teams face, potential causes, and resolution mechanisms.

Issue	Cause	Resolution
No PSP found error	The PSP determination rules in the digital payments add-on are not matching the data sent from the consumer application.	Check the Payment Service Provider Determination app. Ensure there is a catchall row at the bottom with empty conditions pointing to your default Stripe alias.
Transaction is pending indefinitely	Webhooks are failing or not firing.	Check the Stripe dashboard under **Developers • Webhooks**. If webhooks are failing (non-200 response), verify that the digital payments add-on tenant is online. If no webhooks are generating, ensure the event types are subscribed.
Level 3 data not appearing in Stripe	The upstream SAP system (e.g., SAP S/4HANA) is not sending line-item details, or the **Send Level 2/3 Data** box is unchecked in the adapter configuration.	Verify the checkbox in the Stripe configuration UI. If enabled, verify that the XML/JSON payload from SAP S/4HANA contains item-level details, not just header totals.

Table 4.20: Troubleshooting Guide

Issue	Cause	Resolution
Advice job fails to run	The **Run Immediately** job option was triggered for a time window in which no transactions occurred, or the API token for the advice service has expired.	Ensure transactions exist in Stripe for the requested time window. If the error persists, refresh your API credentials in the Stripe configuration UI.

Table 4.20: Troubleshooting Guide (Cont.)

4.7 Summary

This chapter provided comprehensive guidance for integrating PSPs with the SAP digital payments add-on, using Stripe as the primary implementation example. You learned about the PSP integration architecture (adapter framework, API patterns, certified PSP ecosystem); created and configured PSP accounts with API credentials in both test and production environments; activated and configured the Stripe V2 adapter with merchant onboarding and routing rules; enabled new payment methods across cards, wallets, and external payment types with multicurrency support; configured webhook endpoints for asynchronous event handling and payment advice scheduling; and validated the complete integration through comprehensive test scenarios using Stripe's test cards.

Your PSP integration environment is now operational, with adapters activated in both test and production SAP BTP subaccounts, merchant accounts onboarded with verified connectivity to Stripe, merchant aliases mapped for routing configuration, PSP determination rules directing transactions to appropriate merchants based on company code and currency, payment methods enabled in the Stripe dashboard and configured through the payment page setup UI, webhook endpoints registered with confirmed delivery, and the payment advice scheduler configured for automated reconciliation.

The adapter framework architecture ensures PSP independence; each adapter operates as an isolated microservice with its own API protocols and webhook endpoints. This isolation enables version updates without affecting the digital payments add-on core service, allows adding new PSPs through configuration rather than code changes, and ensures that failures in one PSP don't impact others.

The subsequent chapters build on this PSP foundation: Chapter 5 covers the SAP ERP implementation requiring custom ABAP development for payment authorization during order entry, capture during billing, and refund handling through credit memos with sample code and batch job configuration. Chapter 6 the addresses the SAP S/4HANA implementation, leveraging standard SAP Fiori applications, out-of-the-box order-to-cash integration, and built-in analytics for payment monitoring.

Saravana Kumar Kuppusamy

Digital Payments with SAP

Credit Card Processing, PSP Integration, and PCI Compliance

- Deploy the SAP digital payments add-on to process credit card transactions
- Configure the add-on to protect your SAP S/4HANA or SAP ERP system
- Integrate with payment service providers and stay PCI compliant

www.sap-press.com/6294

The Author

Saravana Kumar Kuppusamy has 25 years of experience with SAP implementation and global rollout projects, as well as 10 years of experience managing payment platforms. He has led payments modernization initiatives, such as migrating from legacy payment processing solutions to the SAP digital payments add-on.

ISBN 978-1-4932-2836-2 • 449 pages • 06/2026

E-book: $84.99 • **Print book:** $89.95 • **Bundle:** $99.99

Rheinwerk Publishing