Payment Authorisation in EUDI Wallets: From specification to implementation (TS12 under the EU ARF)

TS12 (Technical Specification 12) defines how the EUDI Wallet performs Strong Customer Authentication (SCA) for electronic payments under eIDAS 2.0 and PSD2. This paper sets out what it takes to implement TS12 in production: the four transaction data types, the six payment flows they support, and the gaps we found and fed back to the specification. The work covers bank, PISP (Payment Initiation Service Provider) and merchant scenarios across both the EUDI Wallet and the European Business Wallet.

A paper by iGrant.io, Sweden, peer-reviewed by digital identity industry experts.

Published: July 2026

Editors: Lal Chandran, Lotta Lundin, George Padayatti

The editors thank the following who supported us in reviewing, correcting and ensuring the content quality:

- Nikolaos Triantafyllou (University of Aegean, Greece)
- Florin Coptil (Bosch, Germany)
- Bo Harald (Finland)
- Petros Kavassalis(University of Aegean, Greece)
- Benoit Krystlik (France)
- Cosmina Chis (Banca Transilvania, Romania)
- Leone Riello (Infocert, Italy)
- Viky Manaila (Intesi Group, Italy)
- Teemu Varpanen (ReceiptHero, Finland)

Table of Contents

The 2027 deadline is closer than most organisations realise

By 24 December 2027, banks and other in-scope payment service providers in the EU must accept the EUDI Wallet for payment authentication when the user requests it, as one of the accepted methods alongside their existing SCA. The obligation under eIDAS 2 (Regulation (EU) 2024/1183, Article 5f) applies to large and medium-sized relying parties in the listed sectors; micro and small enterprises are excluded. That is not a distant regulatory horizon. It is a hard technical and commercial deadline that touches every bank, every Payment Initiation Service Provider (PISP) and every merchant operating in the European market.

Most organisations are still at the architecture stage. They are mapping the protocol stack, evaluating wallet vendors, and working through what TS12 and the ARF actually require. Some have built proof-of-concept flows. Very few have deployed anything in production.

This paper draws on a complete implementation of the EUDI Wallet SCA experience across all four TS12 transaction data types: six payment flows covering instant, scheduled and recurring account payments; card payments; both mandate variants; and login, risk transaction authentication and account access consent. The work was carried out in production with ecosystem partners, including banks, PISPs and trust service providers.

This article explains how TS12 works, shows what we have built, describes what going to production has taught us, and sets out why it matters for everyone who needs payment workflows to work in the EUDI ecosystem.

What is the TS12 specification, and why does it matter?

TS12 (Technical Specification 12) is the EUDI Wallet standard that defines how Strong Customer Authentication (SCA) for electronic payments must be implemented in the European Digital Identity Wallet. The best starting point for explaining this further is through a working demonstration.

> A note on terminology: TS12 technically specifies the authentication step, the Strong Customer Authentication (SCA) performed by the user. “Payment authorisation” refers to the broader end-to-end flow in which that authentication is embedded. This paper uses “authentication” to refer specifically to the SCA user-verification step and “authorisation” to refer to the overall payment approval process.

> **VIDEO:** *EUDI Wallet in action: Prove ID, Apply discounts, Pay. Get a boarding pass and e-receipt in seconds. EWC production demo with Fast Ferries based on EWC specifications, precursor to TS12.* https://youtu.be/6GwCxyofSVE

The above video is a precursor to the production of TS12-compliant EUDI Wallet payment authorisation. A user buys a Fast Ferries ticket, verifies their identity and student status in the wallet, verifies the applicability of a requested discount by providing proof of student status attestation, adds billing and contact details via a loyalty card attestation, confirms payment on the EUDI Wallet consent screen, and receives both an e-receipt and a boarding pass, all of which are issued digitally and stored in the wallet. The entire transaction takes under two minutes.

TS12 defines four transaction data types in total:

1. `urn:eudi:sca:payment:1` for card and account payments
2. `urn:eudi:sca:emandate:1` for payment mandates
3. `urn:eudi:sca:login_risk_transaction:1` for login and sensitive account actions, and
4. `urn:eudi:sca:account_access:1` for account information access.

This article covers all four.

The **WYSIWYS** principle, What You See Is What You Sign, sits at the heart of TS12. It requires that the data displayed to the user on the consent screen is cryptographically identical to the data being signed. The lock icon visible in the demo is the ARF Visual Assurance of Dynamic Linking: the user's biometric confirmation is cryptographically bound to the specific transaction details on screen. **This is not a design choice. It is a normative requirement.**

This matters because, by 24 December 2027, in-scope EU banks must accept this flow at the user's request. TS12 is the technical means; the binding legal force comes from eIDAS 2, PSD2, and the forthcoming Payment Services Regulation (PSR). The organisations that understand and implement TS12 now will be ready. The rest will be catching up under a compliance deadline, risking penalties for violating the EUDI Regulation.

Two products, one ecosystem view

Payment authorisation in the EUDI ecosystem requires two systems to work together: the organisation-side wallet that issues credentials and verifies transactions, and the individual wallet the user carries. Few vendors have built and integrated both sides in production. iGrant.io has done exactly that, operating them together end-to-end on live payment rails.

> **Note:** Throughout this article, "user" refers to the wallet holder who is acting as a Payment Service User (PSU). PSD2 distinguishes between the PSU (the entity using the payment service), the account holder (the legal owner of the account), the payer (who authorises the debit from the account) and the payee (the recipient, who may also initiate the transaction in merchant-initiated flows). These roles are not always filled by the same person, and SCA is bound to a natural person according to their role and power on the account.

The Organisation Wallet Suite is iGrant.io's European Business Wallet software offering for banks, PISPs and merchants. On the payment side, it operates as both issuer and verifier. As the issuer, it provisions a Payment Authenticator and a Qualified Electronic Attestation of Attributes (QEAA) directly into the user's EUDI Wallet via OpenID4VCI. The Strong Customer Authentication (SCA) attestation is bound to the Wallet Unit and to the associated payment instrument held by the PSP (i.e., the IBAN or card), rather than to the payer as a natural person. The payer identity is a separate concern, addressed through the PID (Person Identification Data) credential. As a verifier, it presents signed transaction data to the wallet via OpenID4VP, receives the dynamically linked biometric response, and fires webhook notifications to the bank backend in real time. No polling. No latency. No fragile handshakes.

The Organisation Wallet Suite also supports non-mobile holders: organisations, public bodies, merchants and any institution that needs to participate in the EUDI Wallet ecosystem without a smartphone-based wallet.

The Data Wallet is the individual user's side of the flow. It stores the Payment Authenticator issued by the bank, presents the transaction data for informed consent, binds the biometric confirmation cryptographically to the transaction details, and returns the signed authorisation.

Together, these two products deliver a complete, standards-aligned payment authorisation flow, from credential issuance at the bank to signed authorisation at the merchant, entirely within the EUDI Wallet ecosystem, using OpenID4VCI, OpenID4VP and OpenID4VC throughout. Because both sides implement these open standards, the loop is not closed to iGrant.io components: the Data Wallet can present to any standards-compliant verifier, and the Organisation Wallet Suite can issue to and verify against other conformant EUDI wallets. Cross-vendor interoperability is achieved through the EWC, WE BUILD pilots, and standards-driven programmes (e.g., ETSI, OpenID Foundation) and is part of our conformance roadmap.

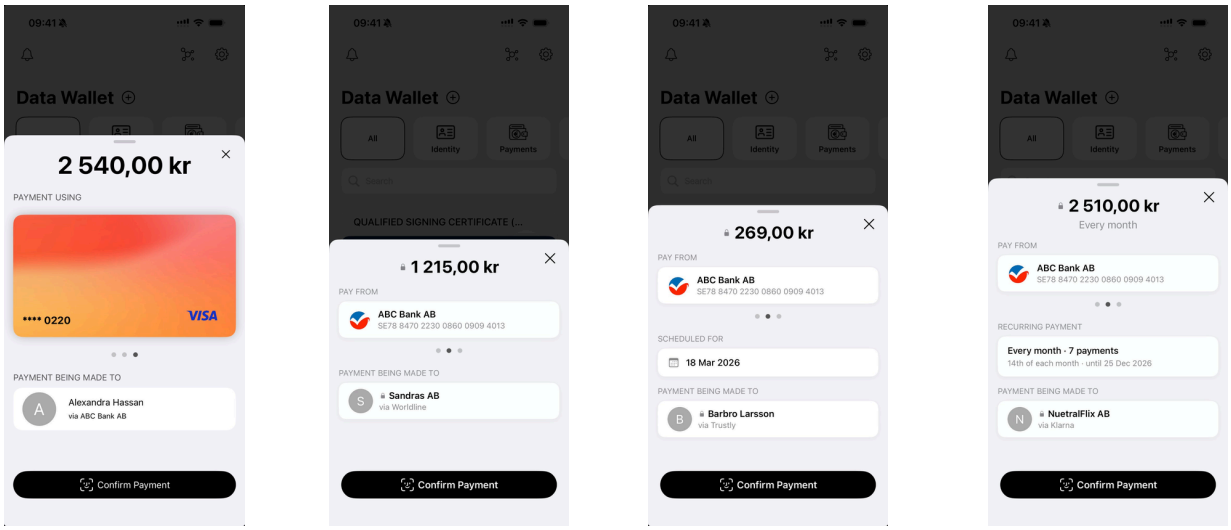
Both products were deployed and trialled on live payment rails (See the merchant-led payment YouTube Video) during the EWC Large-Scale Pilots and were further validated through the WE BUILD Consortium[1], both of which are EU-funded initiatives. Across both programmes, deployment partners included Visa, Worldline, Banca Transilvania and Fast Ferries (via UAegean).

TS12 transaction types and scenarios

The TS12 payment schema and mandate schema together produce six distinct user-facing scenarios. Rather than templating a single layout and forcing every scenario into it, we designed each screen from first principles, starting with what the user needs to understand, not what the schema contains. The table below summarises the scenarios and the key design decisions required by each.

Scenario	Trigger	Key design decision
Card payment	Card credential present	Familiar carousel model; used as the design anchor for all account payment screens
Account payment: Instant	No execution date, any settlement rail	No date shown at all. Absence communicates immediacy better than any label.
Account payment: Scheduled	Future execution date present	SCHEDULED FOR block added; trigger is date presence, not the settlement rail flag
Account payment: Recurring	Recurrence object present	Frequency and derived charge day as a subtitle under the amount; frequency codes are mapped to plain language
Mandate: Single or deferred	eMandate schema, no recurrence	Merchant shown first above amount; charge window adapts to whatever date structure is present
Mandate: Recurring	eMandate schema with recurrence	SCHEDULE block shows frequency, count and validity; the merchant initiates every charge, made explicit

[1] https://www.webuildconsortium.eu/



Card payment *Instant account payment* *Scheduled payment* *Recurring payment*

Three decisions from this work are worth highlighting because they reveal something about the journey from specification to production implementation.

The scheduled payment trigger: The naive approach is to use the settlement rail flag to determine whether a payment is instant or scheduled. This is wrong. A payment can use standard settlement rails and still execute immediately if no future date is set. A better way is to trigger the SCHEDULED FOR block only when a future execution date is present. Using the wrong trigger produces the wrong screens for real users.

Frequency mapping: TS12 defines frequency as a code: MNTH, QUTR, TWMN, TOWK, and 10 others. None of these codes means anything to a consumer. Everything is mapped to plain language before it reaches the screen. QUTR becomes "Every quarter." TWMN becomes "Twice a month." TOWK becomes "Every two weeks."

The charge day question: The recurrence object defines frequency, start date, end date and payment count. The question of which day within the period a charge falls on is one that our implementation had to resolve in production. We fed our approach back to the TS12 editors, as described below.

Mandate screens: A fundamentally different mental model

A payment mandate is not a payment. Understanding this distinction is the most important thing about designing the mandate screens.

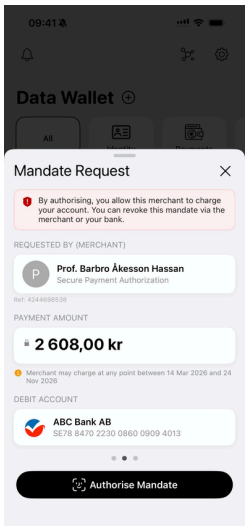
When a user confirms a payment, they are approving a single, defined transaction happening now or at a specified future time. When a user authorises a mandate, they grant a merchant the authority to initiate charges on their account at the merchant's discretion, within a defined validity window. The mental model is closer to signing a direct debit instruction than confirming a purchase.

This distinction shapes every element of the mandate screen. The screen title is "Mandate Request" rather than an imperative like "Authorise Payment Mandate." The merchant appears first, above the amount. The confirm button is labelled "Authorise Mandate." The consent language is explicit about what the authorisation means and what the user can do afterwards.

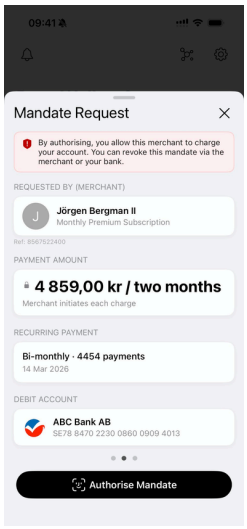
The charge window is the defining element of a single or deferred mandate. It communicates when the merchant may charge. The implementation must handle every data combination the schema can produce:

Schema state	What the user sees
Start date and end date are both present	"Merchant may charge at any point between 14 Mar 2026 and 24 Nov 2026"
No start date, end date present	"Merchant may charge from now until 24 Nov 2026"
Neither date present	"Merchant may charge at any time"
Start date present, no end date	"Merchant may charge from 14 Mar 2026 with no end date"

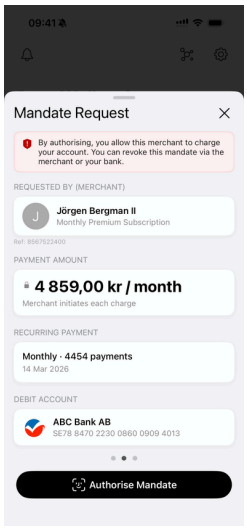
None of these cases shows a placeholder. None fall back to a generic message that is technically true but practically meaningless. Each renders exactly what the schema says.



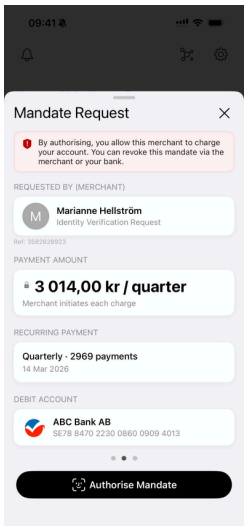
Start and end date



No start date

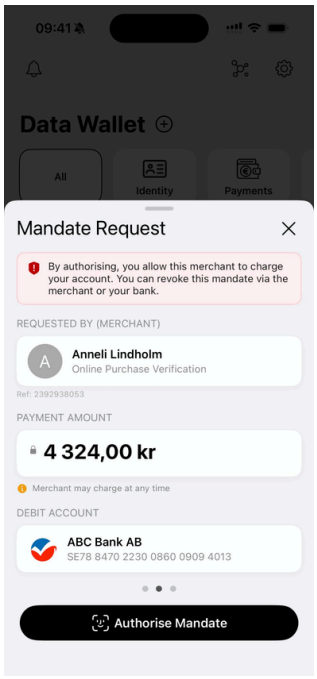


No dates

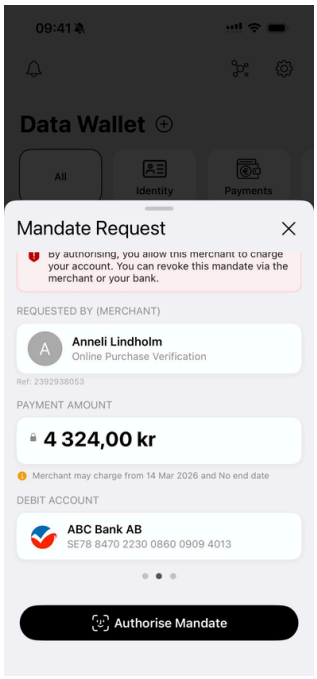


Start date only

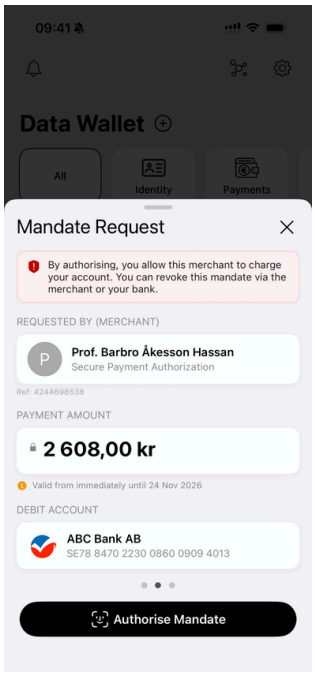
For recurring mandates, the SCHEDULE block replaces the charge window. The key distinction is that the merchant initiates every charge, unlike a standing order, where the account holder initiates. The screen makes this explicit, and the frequency covers all TS12 codes mapped to plain language.



Monthly



Quarterly



Bi-monthly

Account access, login, and risk transactions: the full TS12 scope

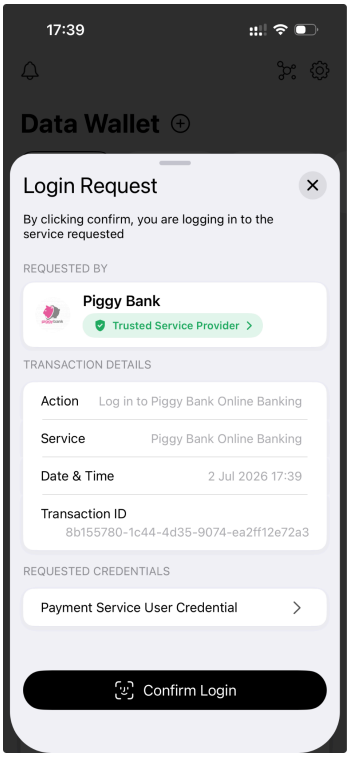
The six payment scenarios above are derived from two of the four transaction data types defined in TS12. The specification also defines urn:eudi:sca:login_risk_transaction:1 for login and sensitive account actions, and urn:eudi:sca:account_access:1 for account information access. iGrant.io has implemented all four.

TS12 transaction type	Covers	iGrant.io status
`urn:eudi:sca:payment:1`	Card and account payments (instant, scheduled, recurring)	Implemented
`urn:eudi:sca:emandate:1`	Payment mandates (single, deferred, recurring)	Implemented
`urn:eudi:sca:login_risk_transaction:1`	Login and sensitive account actions	Implemented
`urn:eudi:sca:account_access:1`	Account information access (AISP flows)	Implemented

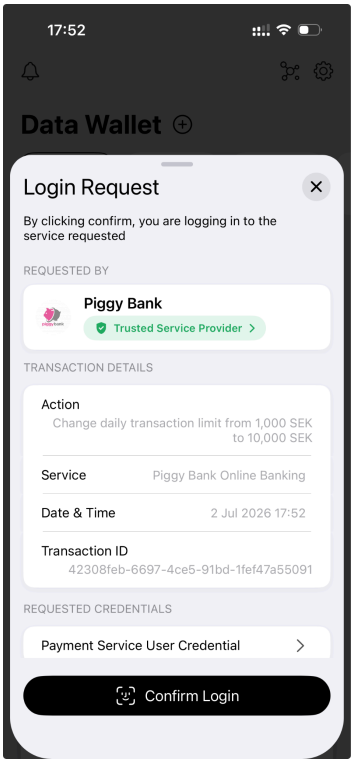
Login and risk transactions: The **`urn:eudi:sca:login_risk_transaction:1`** type covers any SCA event that is not a payment. This includes first-time login to online or mobile banking, step-up authentication for sensitive account actions, and risk-triggered re-authentication mid-session.

What makes this type particularly flexible is the free-text action field. The consent screen adapts to whatever the bank sends. Common examples include "Log in to Online Banking" for a standard login and "Change daily transaction limit from 1,000 EUR to 10,000 EUR" for a high-risk action requiring explicit user confirmation.

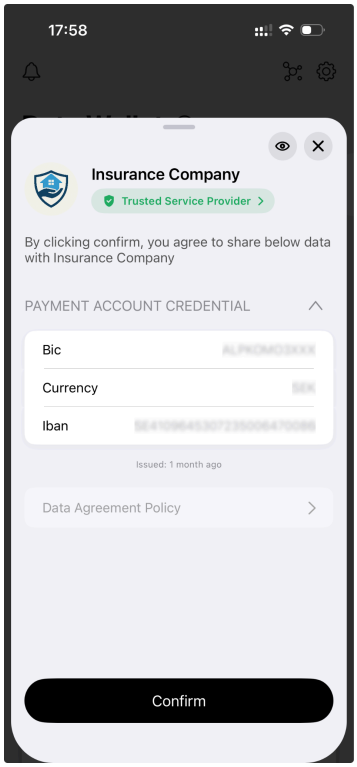
The WYSIWYS requirement applies here exactly as it does for payments. A user authenticating a limit change is signing that specific action, not a generic approval. The action field must be rendered exactly as received, without truncation or reformatting. It is user-facing legal content, not a debug string.



Bank Login



Risk Action



Account Access by 3rd party

During implementation validation, we found that combining login and risk actions into a single transaction type creates a consent-screen design challenge that cannot be resolved without parsing the free-text action field. We have raised this as a formal contribution to the TS12 editors via GitHub Issue #533 [13], proposing either two distinct transaction types or a deprecation path for the current combined type.

Account access: The `urn:eudi:sca:account_access:1` type covers consent for a third party to access account information, such as transaction history or account balances. This is the AISP (Account Information Service Provider) flow under PSD2, brought into the EUDI Wallet ecosystem through TS12. The consent model here builds on an established foundation: the Berlin Group's NextGenPSD2 framework [14] already defines a structured consent object for account access, with clear semantics for scope, duration and permissioned data categories. TS12 inherits this specificity, which is why account access consent is treated as a distinct case from payment or mandate authorisation.

The consent screen for account access must make clear what data is being shared, with whom, and for how long. The user is not authorising a charge or granting charging authority, but granting read access to their financial data to a regulated third party.

What production deployment taught us: Banca Transilvania and Fast Ferries

The most significant learning from building both sides of the EUDI Wallet payment flow does not come from the specification. It comes from deploying it in production with real partners on live payment infrastructure.

iGrant.io has deployed TS12 payment authorisation as a live-rails deployment within the EWC Large-Scale Pilots (and soon in WE BUILD) with Banca Transilvania, one of Romania's largest commercial banks. Separately, Fast Ferries operates the EWC precursor deployment on live rails, built on EWC specifications ahead of TS12. Both run on existing payment rails. These deployments are not running on test infrastructure or sandbox environments. They are processing real transactions with real users through the same payment rails that those organisations use today.

Going to production on existing rails, rather than on dedicated pilot infrastructure, exposes a category of problems that specification work and proof-of-concept builds do not address. Amount range edge cases, PISP brand resolution, date formatting across time zones, and implausible payment counts: these are not theoretical scenarios. They are things that happened, were resolved, and are now handled defensively in the codebase.

Building both the individual Data Wallet and the Organisation Wallet Suite, and operating them together in live deployments, also means that every integration point between the two systems is understood from both sides. When a bank issues a Payment Authenticator via OpenID4VCI, we know exactly how it arrives in the user's wallet, how it is presented on the consent screen, and what the webhook payload looks like when the signed authorisation is returned.

Contributing back to TS12: what full implementation adds to the specification

TS12 is doing something genuinely difficult. It defines a single, interoperable schema for payment consent across six distinct flow types, in a way that is legally compliant under PSD2, technically sound under the ARF, and legible to consumers across the EU. The EWC Large-Scale Pilots exist precisely to stress-test that work in real implementations, and the collaboration between iGrant.io and the ARF contributors has been productive

The TS12 recurrence object defines frequency, start date, end date and payment count. The question of which day within the period the charge will occur is left to implementations to determine. In practice, a user authorising a monthly recurring payment needs to know which day of the month the charge will fall. Our implementation resolves this by deriving the charge day from the day of the month on the start date and displaying it in plain language: "14th of each month," "14th of each quarter," "every two weeks from 14 March."

We have shared this approach as a formal contribution to the TS12 specification via GitHub Issue #531 [7], proposing two paths towards a normative resolution: a new execution-timing field in the recurrence object, or a normative statement that binds the start date day-of-period as the execution anchor. Two further contributions have been made based on production deployment experience: Issue #532 [12], on whether the IBAN should be displayed in full or partially masked on the consent screen, and Issue #533 [13], proposing the separation of login and risk action transactions into distinct types to enable appropriate treatment on the consent screen. All three issues are currently under review by the TS12 editors.

The depth of implementation behind every screen

Amount range validation: When a payment schema specifies a variable amount range, the minimum and maximum can arrive in either order, be equal, be zero, or be absent entirely.

Every case is handled: equal values are rendered as a fixed amount, inverted ranges are sorted before display, absent values fail gracefully, and zero or negative values are flagged. The amount field is dynamically linked: displaying it incorrectly is a WYSIWYS compliance failure, not just a UX bug.

Payment count suppression: When the recurrence payment count in the schema exceeds a threshold that is implausible for the date range, our implementation suppresses the count and displays 'Ongoing' instead. This prevents values such as '3,711 payments' from appearing on a user's consent screen. Ideally, the PISP would send only meaningful values in the payment count field. In practice, TS12 does not constrain the values a Relying Party may include, and production data from live integrations has shown implausible counts. The wallet must handle whatever arrives in the schema defensively, which is why the suppression logic exists at the wallet level rather than relying on upstream data quality.

Date formatting across all edge cases: A single payment date is formatted as "14 Apr 2026." A date with time becomes "14 Apr 2026, 09:00." Same-day start and end dates collapse to a single date with a singular payment count. Absent start dates produce "Valid from immediately." Absent end dates produce "No end date" rather than a blank field.

PISP brand display: The "via Worldline" or "via Trustly" line under the payee name indicates which regulated payment service is initiating the transaction on the bank's behalf. We display the brand name from the PISP object in the schema, never a raw credential label or system identifier.

Charge day derivation: The charge day is not explicitly defined in the schema. We derive it from the start date's day of the month, apply it to the frequency, and display it in natural language: "14th of each month," "14th of each quarter," "every two weeks from 14 Mar."

ARF compliance - Implemented against the current TS12 and ARF, tracking the forthcoming SCA rulebook

The ARF Visual Assurance of Dynamic Linking requirement, a lock icon adjacent to the amount and payee fields confirming cryptographic binding, is implemented on all payment authorisation screens. The lock icon sits directly before the amount on every screen.

> Note on governance: TS12 is the technical specification governing SCA implementation. Sitting above it is the forthcoming SCA Attestation Rulebook, referenced in ARF Topic AA, which will define the governance layer: the rules for attestation issuance, required attributes and display obligations that TS12 enables but does not itself specify. Several of the open questions noted in this section are expected to be resolved in the rulebook. Accordingly, the compliance position stated here is against the current TS12 and ARF, tracking the forthcoming rulebook.

We are also working collaboratively with the TS12 editors on two areas where normative guidance will benefit the whole ecosystem. The first is the reject-control question: whether a dismiss gesture satisfies the ARF requirement or whether a dedicated visible reject button is needed. The second is the precise scope of fields that require the dynamic-linking visual indicator.

A third open question concerns the display treatment of the IBAN on the consent screen. TS12 and the ARF do not currently specify whether the IBAN should be shown in full or partially masked. Full display supports WYSIWYS by ensuring the user sees exactly what is being

signed. Masking reduces the exposure of sensitive financial data to shoulder surfing and screen capture. We have raised this as a formal question to the TS12 editors via GitHub Issue [12].

Beyond the specification-level questions above, production implementation surfaces broader open topics at the boundary between TS12 and the payments ecosystem. The most significant concern is transaction risk. Under PSD2, the bank (ASPSP) bears responsibility for the payment, including fraud detection and transaction risk analysis. The Organisation Wallet Suite is a secure technical enabler in this chain; it does not assume the PSP's SCA responsibility, which remains with the regulated payment service provider even where operational elements are outsourced. The fraud-signal work underway in the Large-Scale Pilots is directly relevant here: wallet-based SCA can provide richer contextual signals to the bank's risk engine than traditional browser-based authentication, making this not solely a legal compliance matter but also an operational improvement to the payment security chain. Concretely, the wallet can supply signals such as wallet-unit attestation status, device integrity and the freshness of the biometric verification, each cryptographically bound to the transaction, giving the risk engine higher-quality inputs than a browser session can offer. Separately, PSD2's SCA exemption framework for low-value payments (transactions under 30 EUR, subject to cumulative limits) remains unaddressed by TS12. While wallet-based SCA may ultimately make this exemption unnecessary, if the user experience is seamless enough that requiring SCA on every transaction imposes no friction, the use case exists today, and implementations must decide how to handle it at the ASPSP level. Our approach and recommendation: the ASPSP applies its exemption logic before invoking wallet SCA, so the wallet flow is triggered only when SCA is actually required, and cumulative-limit tracking remains with the ASPSP, where the account data lives.

What this means for banks, PISPs, merchants and public sector bodies

For banks and ASPSPs: The Organisation Wallet Suite provides a production-ready issuer and verifier for the TS12 payment flow, proven in live deployment with Banca Transilvania. Real-time webhook notifications mean your backend receives authorisation events immediately, with no polling and no latency. Existing payment rails are fully reused: no replacement of current infrastructure is required.

For PISPs: The complete payment schemas, namely, instant, scheduled, recurring, single mandate and recurring mandate, are handled. Your brand appears correctly in the "via [brand]" line on every payment screen. WYSIWYS compliance is implemented, not aspirational. The Fast Ferries deployment demonstrates the flow operating on live payment infrastructure with real merchant transaction volumes.

For merchants: Mandate screens are built and tested across all schema variants, with the consent language legally accurate under PSD2 Article 80(4). The charge window adapts to whatever date structure your backend provides.

For public sector bodies: The Organisation Wallet Suite supports non-mobile holders, meaning organisations that need to issue, verify or hold credentials without a smartphone-based wallet. Banks operating automated pipelines, public sector bodies issuing credentials at scale, and merchants with non-phone-based point-of-sale environments can all participate fully in the EUDI ecosystem.

Getting started

The iGrant.io developer environment is publicly available. Everything you need to evaluate the integration is available at docs.igrant.io, including a full API reference, an integration guide, a demo video, and an end-to-end developer playground where you can try the complete TS12 payment flow without writing a single line of production code.

To connect, you need your payment backend or core banking API and your authentic source, the system you use today to perform SCA with real customers. The Organisation Wallet Suite connects via REST API and webhooks. Standard integration takes weeks, not months. Deployment options cover cloud and SaaS hosted by iGrant.io, private cloud, and on-premise.

The 2027 deadline is approaching. The consent screen is not a detail. It is where trust is created or destroyed, where compliance is demonstrated or fails, and where users decide whether they believe in the EUDI Wallet.

iGrant.io has built the full stack on both sides of the flow in production, with real partners, on existing payment rails. We are ready to deploy it with you.

Book a demo at https://www.igrant.io/#book-a-meeting **or contact us at** support@igrant.io**.**

Frequently asked questions

What is TS12? TS12 (Technical Specification 12) is the EUDI Wallet standard defining how Strong Customer Authentication for electronic payments must be implemented using the European Digital Identity Wallet. It specifies the credential schema, the OpenID4VP presentation flow, and the WYSIWYS requirements for the payment consent screen.

What is WYSIWYS? What You See Is What You Sign. This is the requirement that the data displayed to the user on the consent screen is cryptographically identical to the data being signed. Any discrepancy between displayed and signed data is a compliance failure under the ARF Dynamic Linking requirements.

What is SCA in the context of the EUDI Wallet? Strong Customer Authentication in the EUDI Wallet context means using the wallet's biometric binding and dynamic linking capability to authenticate payment transactions, replacing legacy SCA methods with a standards-based, wallet-native flow under PSD2 Article 97 and the Commission Delegated Regulation (EU) 2018/389.

When does the EUDI Wallet payment mandate take effect? By 24 December 2027, in-scope EU banks and payment service providers must accept the EUDI Wallet for payment authentication at the user's request, as one of the accepted methods alongside existing SCA. The obligation covers large and medium-sized relying parties in the listed sectors; micro and small enterprises are excluded. This is a hard deadline under eIDAS 2, Regulation (EU) 2024/1183, Article 5f.

What is the difference between a payment and a mandate? A payment authorisation covers a single transaction occurring now or at a defined future time. A mandate grants a merchant the authority to initiate charges within a validity window, at the merchant's discretion. It is closer to a direct debit instruction than a payment confirmation.

Does EUDI Wallet payment authorisation require replacing existing payment rails? No. iGrant.io's implementation is designed to work on top of existing payment infrastructure. The deployments with Banca Transilvania and Fast Ferries both operate on their existing payment rails.

What are PSD3 and the PSR, and how do they relate to TS12? PSD3 is the forthcoming revision to the EU Payment Services Directive (PSD). While TS12 implements SCA requirements under the current PSD2 framework, the EUDI Wallet infrastructure being built now is designed to carry forward into the PSD3 regulatory environment. Equally important is the Payment Services Regulation (PSR), which accompanies PSD3 and is directly applicable across all EU Member States without requiring national transposition. For SCA harmonisation, the PSR is arguably more structurally significant than PSD3, as it sets uniform rules that banks and PISPs must follow throughout the EU.

Can organisations participate without a mobile wallet? Yes. iGrant.io's Organisation Wallet Suite supports non-mobile holders: banks, public-sector bodies, merchants, and any institution that needs to issue, verify, or hold credentials outside a smartphone-based wallet.

References

[1] European Digital Identity Wallet Project, 2024. TS12 Electronic Payments SCA Implementation with Wallet. Available at: https://github.com/eu-digital-identity-wallet/eudi-doc-standards-and-technical-specifications

[2] European Digital Identity Wallet Project, 2024. ARF Topic AA: Support of Electronic Payments Customer Authentication (SCA) with the Wallet. Available at the European Digital Identity Github Repository

[3] European Parliament and Council, 2015. Directive (EU) 2015/2366 on Payment Services (PSD2), Article 80(4): Revocation of payment orders. Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32015L2366

[4] European Parliament and Council, 2015, Directive (EU) 2015/2366 on Payment Services (PSD2), Article 97: Strong Customer Authentication requirements. Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32015L2366

[5] European Commission, 2018. Commission Delegated Regulation (EU) 2018/389: SCA and common and secure open standards of communication. Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32018R0389

[6] European Banking Authority, 2019. EBA Q&A 2019/4496: Revocation of future-dated Payment Initiation Services payments. Available at: https://www.eba.europa.eu/single-rule-book-qa/qna/view/publicId/2019_4496

[7] iGrant.io, 2025. GitHub Issue #531: Recurrence object missing execution timing field, Available at the European Digital Identity Github Repository

[8] EWC Consortium, 2024. EWC RFC-008: Payment Data Confirmation. Available at: https://github.com/EWC-consortium/eudi-wallet-rfcs

[9] iGrant.io, 2024. EWC production demo: EUDI Wallet in action: Prove ID, Apply discounts, Pay. Available at: https://www.youtube.com/watch?v=6GwCxyofSVE

[10] iGrant.io, 2024. Organisation Wallet Suite. Available at: https://www.igrant.io/organisationwallet-for-eudi-wallet.html

[11] iGrant.io, 2024. Data Wallet. Available at: https://www.igrant.io/datawallet-for-eudi-wallet.html

[12] iGrant.io, 2026. GitHub Issue #532: SCA Attestation display: should IBAN be shown in full or masked on the consent screen? Available at: EU-ARF Github Repository

[13] iGrant.io, 2026. GitHub Issue #533: login_risk_transaction: proposal to add an optional transaction_category field to distinguish login from risk actions. Available at the European Digital Identity GitHub Repository

[14] The Berlin Group (2025) NextGenPSD2 Access to Account Interoperability Framework. Available at: https://www.berlin-group.org/psd2-access-to-bank-accounts (Accessed: 2 July 2026)

[15] European Parliament and Council, 2024. Regulation (EU) 2024/1183 amending Regulation (EU) No 910/2014 as regards the establishment of the European Digital Identity Framework (eIDAS 2), Article 5f: reliance-party obligation to accept the EUDI Wallet by 24 December 2027. Available at: https://eur-lex.europa.eu/eli/reg/2024/1183/oj