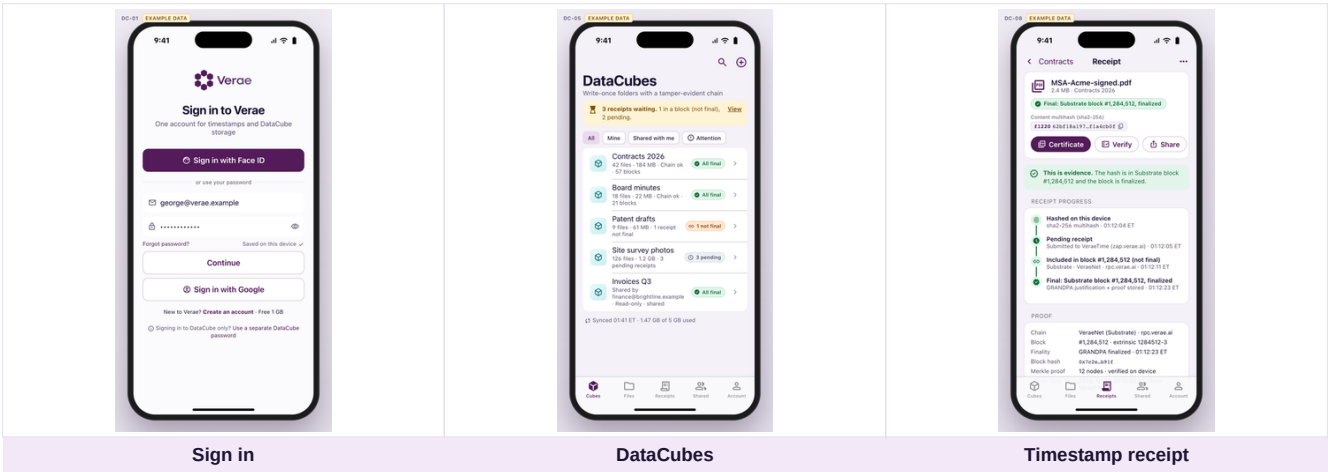


Verae

Timestamp registry · Digital DataCubes · Evidence you can keep

Executive introduction: services, products, platforms, pricing, administration, and APIs with parameter types and sample calls.



Local design frames only · customer.verae.ai · peergos.verae.ai
© Verae / FlightLookup Inc.

Contents

1. Executive summary
2. Core services: Timestamping & Digital DataCubes
3. Product portfolio
4. Platforms (mobile, desktop, web, API, MCP, Zapier)
5. DataCube: storage, sharing, timestamps, metadata, restamp & versions
6. Pricing (free, monthly, storage, stamp SKUs)
7. Customer, sales, admin, accounting & ops
8. API reference — parameters, types, sample calls
9. Glossary

1. Executive summary

Verae is a commercial platform for **digest-only timestamping** on the **Verae timestamp registry** and for **Digital DataCubes** — encrypted write-once folders on Peergos — so organizations can prove when content existed and retain tamper-evident history without uploading file bytes to the registry.

Customers use **customer.verae.ai**, long-lived xVerae - keys, and the same registry from phones, desktops, PrintIt, Zapier, MCP, or HTTP. DataCubes hold encrypted objects; each hash can carry a registry receipt and optional **restamp** chains with JSONL metadata and versioned diffs.

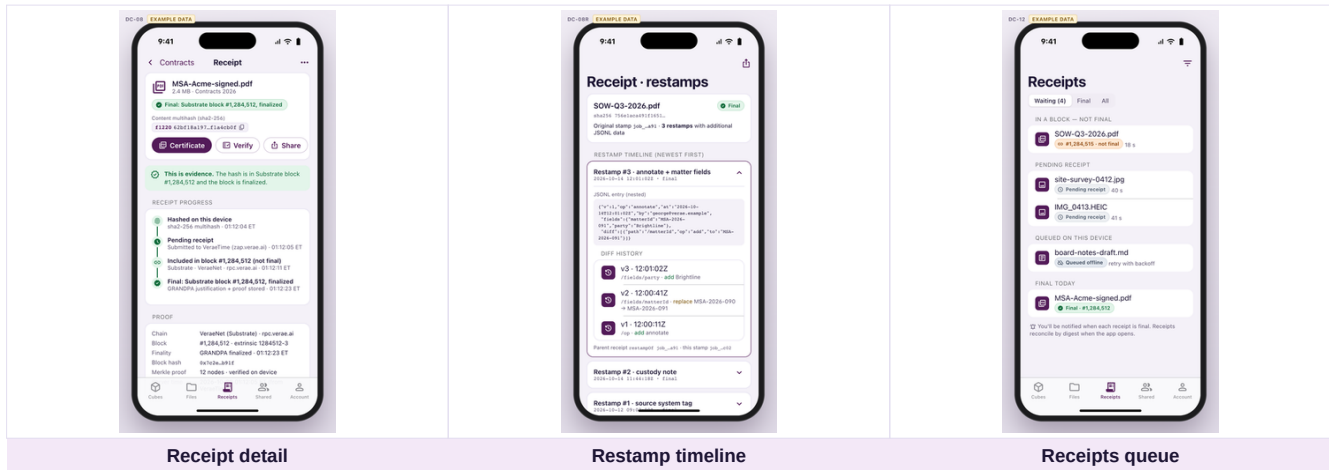
- Proof of existence without surrendering documents for stamping.
- One account for timestamps, storage metering, Stripe or Purchase Order.
- Same registry across iOS, Android, macOS, Windows, Debian, web, Zapier, MCP.
- Restamp/metadata tracking without rewriting the first certificate (default restamp:false).

2. Core services

2.1 Verae timestamping (timestamp registry)

A **Verae signature** means a **timestamp with a receipt**: SHA-256 of content (or of a metadata JSONL line) is posted to the commercial edge; the receipt moves pending → included/provisional → final. File bytes need not leave the device for the stamp.

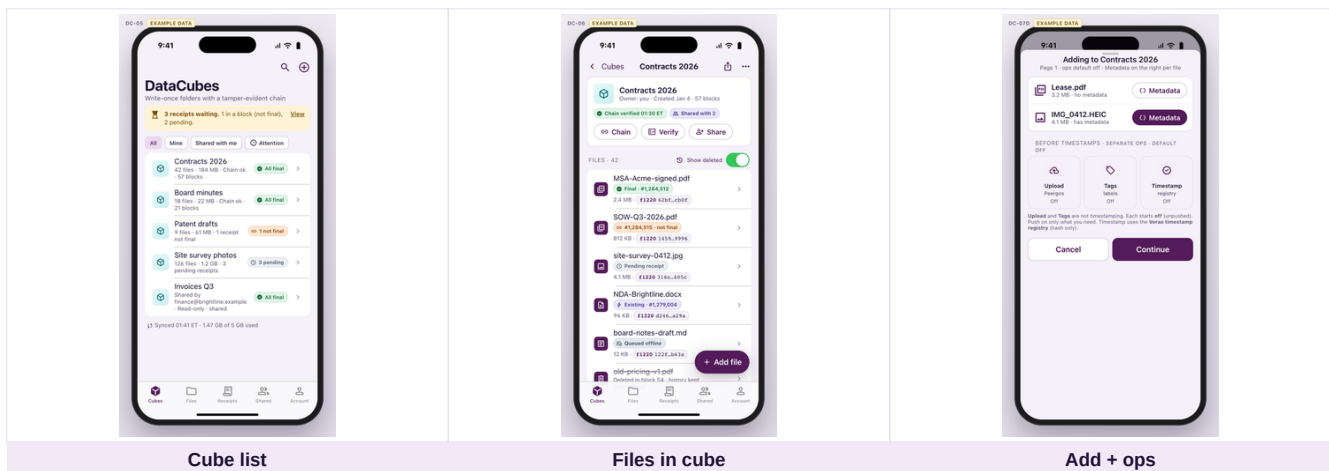
Restamping (`restamp: true`) adds a new receipt linked by `restampOf` for additional metadata, custody notes, or tag diffs without invalidating the original certificate.



2.2 Digital DataCubes (Peergos)

A **DataCube** is end-to-end encrypted storage (Peergos cryptree) for files and directories, with capability sharing, an internal block chain (add / share / revoke / tombstone), and optional pairing of object hashes to Verae timestamp registry receipts.

When adding attachments, **Upload** (Peergos), **Tags**, and **Timestamp** are separate icon toggles (default **off**). Authors enable only what they need before commit.



3. Product portfolio

Product	Description	Users
DataCube app	Cubes, files, receipts, share, billing on phone/desktop	Customers
PrintIt	Print/share → hash → registry → certificate PDF	Office / legal
Zapier + HTTP automation	Stamp/verify/status from Zaps; same xVerae- metering	Ops
MCP timestamping	Agent tools; bearer xVerae- or JWT	Developers / AI
Public verify	Hash/file/QR check without login	Anyone
Customer portal	Signup, 2FA, keys, usage, Stripe & PO, invoices	Customers
Sales portal	Quotes, per-customer pricing, packages, segments	Sales
Admin / staff console	Customers, tiers, rates, support view, reports	Staff
Accounting export	Invoices, statements, ledger adjust, books export	Finance
humanIdentification (Sign)	Draw/type attest; digests only on registry	Signers
ITAD (optional)	Field logistics; ENABLE_ITAD build flag (off for share builds)	Field ops
Backend ops	Edge, middleware, archives, chain replica, health/phone-home	SRE

4. Platforms

Platform	Surfaces	Notes
iOS	DataCube app, PrintIt share, Sign	Face ID; DC-01...
Android	DataCube M3, print path	Biometrics; DC-A*
macOS	DataCube, PrintIt, MCP Timestamping, menu bar	DC-M*, MT-*, PT-*
Windows	DataCube, PrintIt / Sign printer	DC-W*, PT-*, SG-WN*
Debian/Linux	DataCube, Signed folder	DC-L*, SG-L*
Web	Portal, verify, admin, sales	customer.verae.ai
API	/v1 + /portal/api	xVerae- or session
MCP	verae-timestamping server	Bearer file / HTTP
Zapier	Zapier app + edge automation paths	Same meter as API

Local design frames (pre-generated)



5. DataCube capabilities

5.1 Secure storage

- Peergos end-to-end encryption; password-derived keys (optional separate DataCube password).
- Write-once with tombstones; history proves prior existence.
- Hosted GB on plan; BYO S3 optional.

5.2 Resource sharing

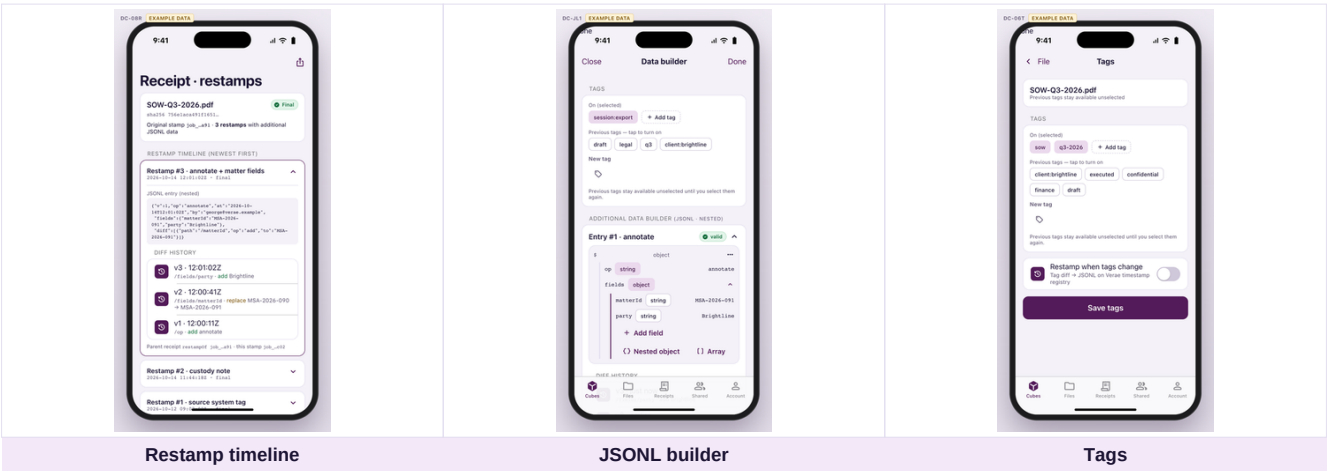
Scope	What	Controls
File	Single object	View/edit, expiry, revoke
Directory	Subtree	Capability share
DataCube	Whole cube	Invite, roles, links

5.3 Timestamps & metadata

- SHA-256 of file or JSONL line → Verae timestamp registry receipt.
- Tags: selected + previous unselected (tap to turn on); optional restamp on change.
- Add flow: Upload | Tags | Timestamp separate toggles (default off); Metadata button per file; Copy last metadata; JSONL builder with diffs.

5.4 Restamp & version tracking

New receipt linked via restampOf; JSONL additional data with nested objects and path-level diff history (add/replace/remove). Original certificate remains. Default first stamp is restamp:false.



6. Pricing

Credit-based metering (~1¢ list per credit before tier multiplier). Sales may set per-customer multipliers and packages. Pay with **Stripe** or **Purchase Order**.

Subscription tiers (defaults)

Tier	Monthly credits	Stamps/mo	Storage	Multiplier
Free	\$1 (100¢)	20	1 GB	1.0×
Pro	\$10 (1000¢)	250	50 GB	0.5×
Business	\$100 (10000¢)	2,500	500 GB	0.25×

Annual: volume = monthly×12; price ≈ monthly credits×12×0.8.

Stamp SKUs (list before multiplier)

Endpoint	Kind	List	Notes
timestamp	fixed	10¢	Plain; payload may quote 10/16/28
timestamp-re	fixed	5¢	Restamp path
timestamp-metadata-update	fixed	15¢	Metadata update
timestamp-verify / hash-lookup / receipt	free	0	Read paths

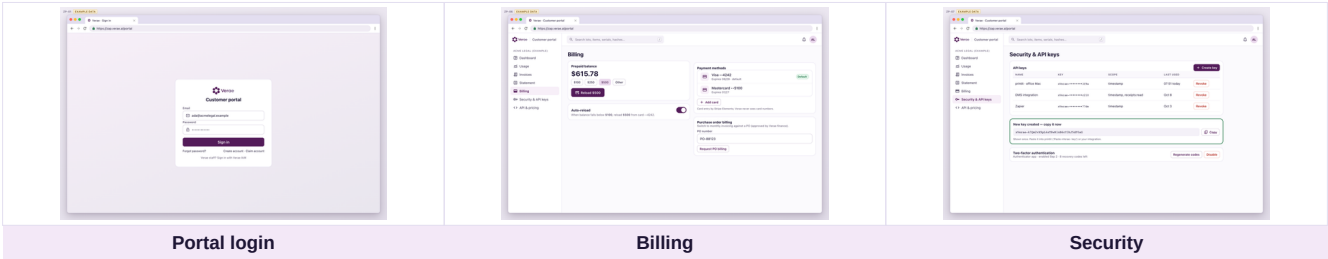
DataCube storage fees

Mode	Included	Overage / notes
Hosted	Per tier GB/mo	AWS S3-class pin + markup; compliance hosted higher
BYO S3	—	~1 credit/GB-month default + optional passthrough
Legacy per-call storage	—	10¢ base + per KB/MB (older Zapier SKU)

7. Customer, sales, admin, accounting & ops

Customer portal — customer.verae.ai/portal

- Signup/login/TOTP; password_hash storage (never plain).
- Profile; Stripe vs Purchase Order.
- xVerae- tokens (plaintext once; api_key_hash in DB).
- Usage, invoices, statements, reload / PO request.



Sales portal

Per-customer pricing, packages, segments, rate overrides (staff).

Admin / staff

Area	Functions
Customers	Accounts, tiers, balances, support view (no key echo)
Pricing	Rate card, tiers, individual overrides
Invoices/exports	Generate, email, accounting export
Reports / IAM	Usage/revenue; staff TOTP & roles

Accounting

Invoices, credit ledger, statements, ledger adjust, finance exports.

Phone-home, health & backend maintenance

- Edge healthcheck; webhook/email on fail; Prometheus textfiles.
- Middleware/portcullis; seal HMAC; job-complete core NATS only (no JetStream).
- Worms/trees; chain replica; cold object storage.
- core-front, status page, object-lock export; verae-ops runbooks.

8. API reference — parameters, types, sample calls

Host <https://customer.verae.ai> unless noted. Passwords → password_hash. API keys → api_tokens / api_key_hash.

POST /portal/api/signup

Auth: none

Parameter	Type	Req
name	string	yes
email	string	yes
password	string	yes
acceptTos	boolean	yes

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/signup \
-H 'content-type: application/json' \
-d '{"name":"Ada","email":"ada@example.com","password":"CorrectHorse1!","acceptTos":true}'
# 201 { "token":"...", "apiKey":"xVerae-...", "customer":{"..."} }
```

POST /portal/api/login

Auth: none

Parameter	Type	Req
email	string	yes
password	string	yes
totpCode	string?	no

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/login \
-H 'content-type: application/json' \
-d '{"email":"ada@example.com","password":"CorrectHorse1!"}'
```

GET /portal/api/profile

Auth: Bearer session

Parameter	Type	Req
Authorization	header	yes

Sample call

```
curl -sS https://customer.verae.ai/portal/api/profile -H "Authorization: Bearer $TOKEN"
```

POST /portal/api/profile

Auth: Bearer session

Parameter	Type	Req
billingType	stripe purchase_order	no
companyName	string?	no
phone	E.164?	no

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/profile \
-H "Authorization: Bearer $TOKEN" -H 'content-type: application/json' \
-d '{"billingType":"stripe","companyName":"Brightline"}'
```

POST /portal/api/2fa/setup

Auth: Bearer session

Parameter	Type	Req
(none)	—	—

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/2fa/setup \
-H "Authorization: Bearer $TOKEN" -H 'content-type: application/json' -d '{}'
```

POST /portal/api/2fa/enable

Auth: Bearer session

Parameter	Type	Req
code	string(6)	yes

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/2fa/enable \
-H "Authorization: Bearer $TOKEN" -H 'content-type: application/json' -d '{"code":"482193"}'
```

POST /portal/api/tokens

Auth: Bearer session

Parameter	Type	Req
name	string?	no

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/tokens \
-H "Authorization: Bearer $TOKEN" -H 'content-type: application/json' -d '{"name":"iphone"}'
# plaintext xVerae- once; server stores hash
```

GET /portal/api/usage

Auth: Bearer or x-api-key

Parameter	Type	Req
peergosUsername	string?	no

Sample call

```
curl -sS https://customer.verae.ai/portal/api/usage -H "Authorization: Bearer $TOKEN"
```

GET /portal/api/invoices

Auth: Bearer session

Parameter	Type	Req
(none)	—	—

Sample call

```
curl -sS https://customer.verae.ai/portal/api/invoices -H "Authorization: Bearer $TOKEN"
```

POST /portal/api/billing/reload

Auth: Bearer session

Parameter	Type	Req
amountCents	number	yes

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/billing/reload \
-H "Authorization: Bearer $TOKEN" -H 'content-type: application/json' -d '{"amountCents":5000}'
```

POST /portal/api/billing/po-request

Auth: Bearer session

Parameter	Type	Req
poNumber	string	yes
terms	string?	no

Sample call

```
curl -sS -X POST https://customer.verae.ai/portal/api/billing/po-request \
-H "Authorization: Bearer $TOKEN" -H 'content-type: application/json' \
-d '{"poNumber":"PO-2026-091","terms":"Net-30"}'
```

POST /v1/timestamp

Auth: Bearer xVerae- / x-api-key

Parameter	Type	Req
sha256	string(64 hex)	yes
restamp	boolean?	no
restampOf	string?	no

Sample call

```
curl -sS -X POST https://customer.verae.ai/v1/timestamp \  
-H "Authorization: Bearer $XVERAE" -H 'content-type: application/json' \  
-d '{"sha256":"e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855"}'  
  
# Restamp JSONL line digest:  
curl -sS -X POST https://customer.verae.ai/v1/timestamp \  
-H "Authorization: Bearer $XVERAE" -H 'content-type: application/json' \  
-d '{"sha256":"<line_sha256>","restamp":true,"restampOf":"<jobId>"}'
```

GET /v1/timestamp/status

Auth: xVerae- or policy

Parameter	Type	Req
sha256	query string(64)	yes

Sample call

```
curl -sS 'https://customer.verae.ai/v1/timestamp/status?sha256=e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855' \  
-H "Authorization: Bearer $XVERAE"
```

POST /sign/v1/requests

Auth: xVerae- owner

Parameter	Type	Req
documentSha256	string(64)	yes
documentName	string	yes
signers	array<{email,name?}>	yes
requireOtp	boolean?	no

Sample call

```
curl -sS -X POST https://sign.verae.ai/sign/v1/requests \  
-H "Authorization: Bearer $XVERAE" -H 'content-type: application/json' \  
-d '{"documentSha256":"<64hex>","documentName":"msa.pdf","signers":[{"email":"a@b.c"}]}'
```

POST /sign/v1/public/:token/sign

Auth: public + OTP

Parameter	Type	Req
email	string	yes
otp	string?	no
signatureSha256	string(64)	yes
kind	draw type upload	yes
consent	boolean	yes

Sample call

```
curl -sS -X POST https://sign.verae.ai/sign/v1/public/$TOKEN/sign \  
-H 'content-type: application/json' \  
-d '{"email":"a@b.c","otp":"123456","signatureSha256":"<64hex>","kind":"draw","consent":true}'
```

Peergos / DataCube (app bridge)

Local Rust session: login, createCube, upload, listChain, share. Optional POST /v1/timestamp after hash. Add-flow flags uploadPeergos, addTags, timestampRegistry (bool, default false).

```
ops = {upload:false, tags:false, timestamp:false} # defaults OFF  
for file in batch:  
  meta = file.meta or copy_last_metadata()  
  if ops.tags: apply_tags(file, on_tags | previous_selected)  
  if ops.upload: peergos.upload(cube, file.bytes)  
  d = sha256(file.bytes)  
  if ops.timestamp: POST /v1/timestamp {"sha256": d}  
  for line in meta.jsonl: POST /v1/timestamp {"sha256": sha256(line), "restamp": true, "restampOf": jobId}
```

9. Glossary

Term	Meaning
Verae signature / timestamp	Digest on Verae timestamp registry + receipt/finality
humanIdentification	Person draw/type attest (Sign product)
DataCube	Peergos encrypted write-once folder + chain
Restamp	New receipt linked to prior job; JSONL/metadata diffs
xVerae-	Customer API token; server stores hash
password_hash	Portal password storage
Stripe / PO	Card prepaid vs Purchase Order invoicing
ENABLE_ITAD	Build flag; share builds use false

All product screenshots in this PDF are pre-generated local files (Desktop/Verae-Design-Screens-Local or samples-clean/thumbs). No oversized source conversion.

© Verae / FlightLookup Inc. · [customer.verae.ai](#) · [peergos.verae.ai](#)