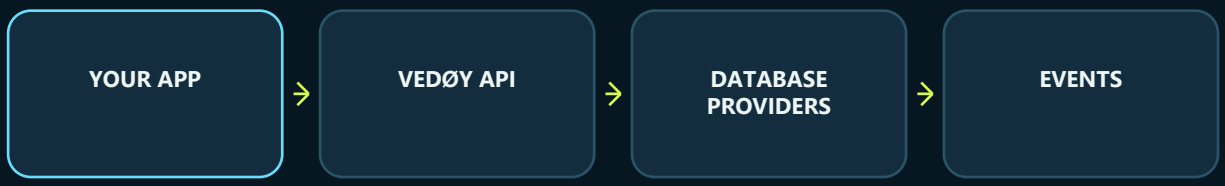
VEDØY DEVELOPER HANDBOOK

Build with Vedøy APIs

A practical guide to projects, API keys, phone numbers, messaging, webhooks, credits, agents and production integration.



Edition 1
Contract version 2026-10-09. Prepared for developers using the Vedøy Developer Portal. Live provider availability is stated separately from implemented code.

vedoy-developers.eiolf-leande-6585.chatgpt.site

CHAPTER 00

How to use this book

Start with the quickstart, then read the chapter for the capability you are integrating. Every code sample assumes server-side use of secrets.

- Use the account session only for project, usage and balance management.
- Use a project API key for application traffic.
- Reuse the same Idempotency-Key when retrying a provider operation after a timeout.
- Treat phone, message, agent and provider status as asynchronous state.
- Keep the OpenAPI file as the machine-readable source of truth for the current contract.

> Current live status
> The API health endpoint was reachable on 16 October 2026. It reported PostgreSQL connected to neondb and Telnyx not configured. Number search, purchase and SMS code exists, but the live carrier operations remain unavailable until production credentials and pricing are configured.

Contents

How to use this book 2

The platform model 5

The two authentication lanes . . . 5

Quickstart in ten minutes 6

Accounts, projects and keys 7

Create a project . . . 7

Create a project key . . . 7

The API contract 8

Base URL . . . 8

Requests, responses and errors 9

Safe retries and idempotency 10

Vedøy Credits and usage 11

Search and buy phone numbers 12

Send SMS 13

Receive trusted webhooks 14

Verify the signature before parsing business data . . . 14

Agent definitions and guarded runs 15

Connect a web or mobile app 16

Security before production 17

Operating the integration 18

Deployment checklist 18
Database note 18

Local development and deployment **19**
Server environment 19

Use the design system consistently **20**

Reference architecture **21**
Recommended services 21

End-to-end example **22**

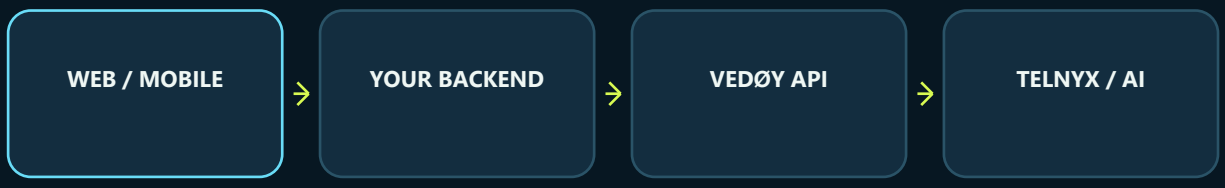
What is live and what comes next **23**
The next production milestone 23

Appendix **24**
Links 24
Minimal go-live gate 24

CHAPTER 01

The platform model

Vedøy Developers is the control plane. Your product is the client. The API sits between your trusted backend, project data and external providers.



The safest integration keeps the Vedøy project key in your backend. A browser or mobile app authenticates the user to your backend; your backend decides what that user may do and then calls Vedøy.

The two authentication lanes

METHOD	PATH	PURPOSE
USER	Supabase bearer token	Create and list projects; view usage and balances.
APP	vdy_live_... bearer token	Call project-scoped endpoints such as models, numbers, messages, webhooks and agents.

Important
Vedøy Phone is a separate customer application built on these APIs. It should not contain carrier credentials or a project secret in the shipped mobile bundle.

CHAPTER 02

Quickstart in ten minutes

Create a project in the portal, issue one API key, store it as a server secret, and make a harmless authenticated request.

- Sign in to the developer portal.
- Create a project.
- Create an API key and copy it immediately.
- Store it as VEDOY_API_KEY in your backend environment.
- Call GET /v1/models.

FIRST REQUEST

```
curl https://vedoy-dev-portal.vercel.app/api/v1/models \
  -H "Authorization: Bearer $VEDOY_API_KEY"
```

NODE.JS

```
const response = await fetch(
  "https://vedoy-dev-portal.vercel.app/api/v1/models",
  { headers: { Authorization: `Bearer ${process.env.VEDOY_API_KEY}` } }
);

if (!response.ok) throw new Error(`Vedøy API ${response.status}`);
const payload = await response.json();
```

> Key handling
> The full key is returned once. Store it in a secret manager. The database stores a SHA-256 hash and a short prefix, not the raw key.

CHAPTER 03

Accounts, projects and keys

A Vedøy account owns projects. A project owns API keys, credits, phone numbers, webhooks and agents.

Create a project

REQUEST

```
POST /api/v1/projects
Authorization: Bearer <SUPABASE_USER_JWT>
Content-Type: application/json

{ "name": "Acme Support", "slug": "acme-support" }
```

Create a project key

REQUEST

```
POST /api/v1/projects/{projectId}/api-keys
Authorization: Bearer <SUPABASE_USER_JWT>
Content-Type: application/json

{ "name": "Production backend" }
```

- Use one key per environment or service.
- Rotate keys by issuing a new key, deploying it, then revoking the old key.
- Never send a project key to a browser, iOS bundle or Android APK.
- The default project rate limit is 60 requests per minute unless changed administratively.

CHAPTER 04

The API contract

OpenAPI 3.1 describes paths, methods, security schemes and core request shapes. Generate clients from it, but still handle unknown fields and asynchronous provider outcomes.

Base URL

PRODUCTION

```
https://vedoy-dev-portal.vercel.app/api
```

METHOD	PATH	PURPOSE
GET	/v1/health	Service, database and provider readiness.
GET/POST	/v1/projects	List or create developer projects.
POST	/v1/projects/{id}/api-keys	Create a key shown once.
GET	/v1/models	List model routes.
GET	/v1/numbers/available	Search live carrier inventory and create quotes.
POST	/v1/numbers/purchase	Purchase a quoted number.
POST	/v1/messages	Send an outbound SMS.

METHOD	PATH	PURPOSE
GET/POST	/v1/webhooks/endpoints	Manage signed event destinations.
GET	/v1/webhooks/deliveries	Inspect delivery attempts.
POST	/v1/webhooks/deliveries/{id}/retry	Retry one delivery.
GET/POST	/v1/agents	Manage agent definitions.
GET/POST	/v1/agents/{id}/runs	List or request runs.
GET	/v1/usage	View recent account usage.
GET	/v1/credits/balance	View project balances.

CHAPTER 05

Requests, responses and errors

Every response is JSON and includes the API version header. Build clients around HTTP status, a stable error code and a human-readable message.

ERROR ENVELOPE

```
{
  "error": {
    "code": "api_key_required",
    "message": "Provide a Vedøy API key as a Bearer token."
  }
}
```

METHOD	PATH	PURPOSE
400	Bad request	Malformed JSON or missing idempotency key.
401	Unauthorized	Invalid user session or API key.
402	Payment required	Insufficient Vedøy Credits.
403	Forbidden	Resource not owned by this project.
409	Conflict	Operation pending or outcome requires reconciliation.
410	Gone	Quote expired.
422	Unprocessable	Input or provider rejection.
429	Rate limited	Wait for Retry-After.
503	Unavailable	Provider or feature not configured.

Log the request context
Record your own correlation ID, endpoint, status, Vedøy error code and provider operation ID. Never log bearer tokens, phone message contents or webhook secrets.

CHAPTER 06

Safe retries and idempotency

Provider actions can time out after the carrier accepted them. Idempotency prevents a network retry from buying the same number or sending the same message twice.

Generate one stable key for one intended action. Reuse that exact key only when retrying the same action.

NODE.JS

```
const idempotencyKey = crypto.randomUUID();

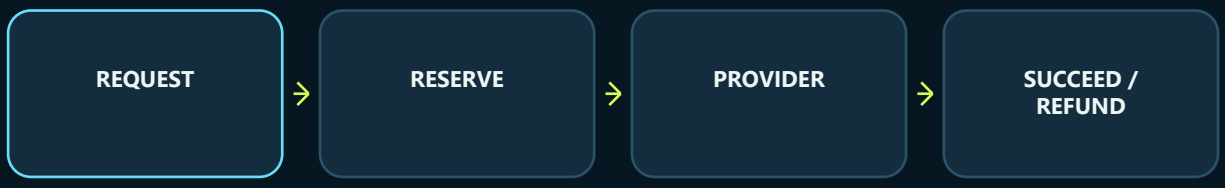
await fetch(`${VEDOY_API}/v1/messages`, {
  method: "POST",
  headers: {
    Authorization: `Bearer ${process.env.VEDOY_API_KEY}`,
    "Content-Type": "application/json",
    "Idempotency-Key": idempotencyKey
  },
  body: JSON.stringify(message)
});
```

- Retry 429 after the response's Retry-After delay.
- Retry a network timeout with the same idempotency key.
- Do not create a new key to force a second attempt while outcome is unknown.
- Treat 409 unknown as a reconciliation task, not an automatic retry.

CHAPTER 07

Vedøy Credits and usage

Credits form a project ledger. Usage and provider reservations create auditable entries instead of mutating an opaque balance.



- A new project receives a welcome ledger entry in the current schema.
- Balances are calculated from ledger entries.
- Provider cost is reserved before an external call.
- Deterministic failures can be refunded automatically.
- Unknown provider outcomes remain reserved until reconciliation.

BALANCE REQUEST

```
GET /api/v1/credits/balance
Authorization: Bearer <SUPABASE_USER_JWT>
```

USAGE REQUEST

```
GET /api/v1/usage
Authorization: Bearer <SUPABASE_USER_JWT>
```

Pricing display
Show estimated prices as estimates. A current quote, taxes, carrier fees, country rules and contract terms determine the final charge. Do not expose carrier wholesale pricing as the product identity.

CHAPTER 08

Search and buy phone numbers

Number search asks the carrier for current inventory and creates short-lived Vedøy quote IDs. Purchase consumes a quote, not an arbitrary phone number string.

SEARCH

```
GET /api/v1/numbers/available?country=NO&type=local&limit=10
Authorization: Bearer $VEDOY_API_KEY
```

PURCHASE

```
POST /api/v1/numbers/purchase
Authorization: Bearer $VEDOY_API_KEY
Idempotency-Key: number-order-018f...
Content-Type: application/json

{ "quoteId": "<quote-uuid>" }
```

- Country uses an ISO two-letter code such as NO.
- Allowed types include local, toll_free, mobile, national and shared_cost.
- A quote expires after ten minutes in the current implementation.
- Store purchase status and provider resource identifiers; provisioning may continue asynchronously.
- Country-specific identity, address and regulatory requirements still apply.

> Availability today
> The live production health response currently reports Telnyx not configured. Build the flow and error states, but do not present number purchase as live until health reports the provider configured and an end-to-end test succeeds.

CHAPTER 09

Send SMS

Messages may be sent only from a number owned by the same project. Both sender and recipient use E.164 format.

SEND MESSAGE

```
POST /api/v1/messages
Authorization: Bearer $VEDOY_API_KEY
Idempotency-Key: sms-018f...
Content-Type: application/json

{
  "from": "+4790000000",
  "to": "+4791111111",
  "text": "Your appointment is confirmed."
}
```

- The accepted response is HTTP 202, not proof of handset delivery.
- Use provider and Vedøy webhook events to update message state.
- Validate consent, opt-out rules and destination requirements before sending.
- Do not send passwords, permanent secrets or highly sensitive data over SMS.
- The current server disables SMS pricing until an explicit provider cost is configured.

CHAPTER 10

Receive trusted webhooks

Vedøy Notify sends event envelopes to public HTTPS endpoints. Each endpoint gets a signing secret returned once at creation.

CREATE ENDPOINT

```
POST /api/v1/webhooks/endpoints
Authorization: Bearer $VEDOY_API_KEY
Content-Type: application/json

{
  "url": "https://api.example.com/webhooks/vedoy",
  "events": ["message.updated", "number.updated"],
  "description": "Production event receiver"
}
```

EVENT ENVELOPE

```
{
  "id": "event-uuid",
  "type": "message.updated",
  "created_at": "2026-10-16T10:00:00Z",
  "data": { ... }
}
```

Verify the signature before parsing business data

NODE.JS PSEUDOCODE

```
const signed = `${timestamp}.${rawBody}`;
const expected = `v1=${createHmac('sha256', secret)
  .update(signed).digest('hex')}`;

if (!timingSafeEqual(Buffer.from(signature), Buffer.from(expected))) {
  return new Response('invalid signature', { status: 401 });
}
```

- Read the raw request body before JSON parsing.
- Reject stale timestamps according to your replay window.
- Deduplicate by event id.
- Return a 2xx quickly and process durable work asynchronously.
- Use the delivery log and manual retry endpoint for recovery.

CHAPTER 11

Agent definitions and guarded runs

The platform stores project-scoped agent configuration, instructions, model routing and a monthly budget guard.

CREATE AGENT

```
POST /api/v1/agents
Authorization: Bearer $VEDOY_API_KEY
Content-Type: application/json

{
  "name": "Support triage",
  "instructions": "Classify the request and suggest a queue.",
  "model": "vedoy/router",
  "monthlyBudgetMicrounits": 50000000,
  "status": "draft"
}
```

REQUEST RUN

```
POST /api/v1/agents/{agentId}/runs
Authorization: Bearer $VEDOY_API_KEY
Idempotency-Key: agent-run-018f...
Content-Type: application/json

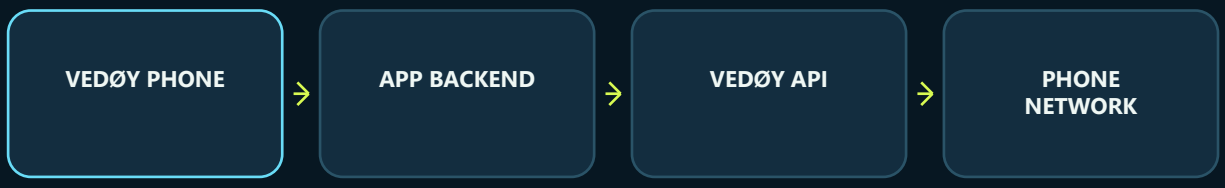
{ "ticketId": "T-1042", "text": "..." }
```

> Current boundary
> The contract records run requests as blocked with agent_runner_not_configured until a reviewed server-side runner and provider billing adapter are connected. Do not build a product promise around live runs yet.

CHAPTER 12

Connect a web or mobile app

A phone app is a public client. It should call your backend with a short-lived user session, while the backend calls Vedøy with the project key.



PUBLIC CLIENT

```
// Mobile app calls your own backend
await fetch('https://api.yourapp.no/messages', {
  method: 'POST',
  headers: {
    Authorization: `Bearer ${userSession}`,
    'Content-Type': 'application/json'
  },
  body: JSON.stringify({ to, text })
});
```

TRUSTED BACKEND

```
// Your backend enforces user/project permissions, then calls Vedøy
await fetch(`${VEDOY_API}/v1/messages`, {
  method: 'POST',
  headers: {
    Authorization: `Bearer ${process.env.VEDOY_API_KEY}`,
    'Content-Type': 'application/json',
    'Idempotency-Key': operationId
  },
  body: JSON.stringify({ from, to, text })
});
```

- Use the OS secure storage for user refresh tokens, not project keys.
- Bind each app user to permitted project resources server-side.
- Require explicit confirmation before billable actions.
- Show pending, sent, delivered and failed as separate states.
- Do not claim real voice calls: the current central contract documents numbers and SMS, not a voice call API.

CHAPTER 13

Security before production

Security depends on where trust lives. The central rule is simple: secrets stay on trusted servers, and every resource check is scoped to a project.

METHOD	PATH	PURPOSE
SECRET	Vedøy API key	Backend secret manager only.
SECRET	Webhook signing secret	Webhook receiver only.
SECRET	Carrier credentials	Vedøy provider adapter only.
PUBLIC	Supabase publishable key	Browser configuration; still enforce authorization.
TOKEN	User access token	Short-lived client session sent over HTTPS.

- Validate authorization for every project resource, not only at login.
- Enforce body size, field length and E.164 validation.
- Redact secrets and message bodies from logs.
- Rotate credentials and keep an audit trail.
- Set alerts for usage spikes, 401, 402, 409, 429 and 5xx responses.
- Protect administrative endpoints with explicit roles and re-authentication.
- Review data retention and regional storage requirements before onboarding customers.

CHAPTER 14

Operating the integration

Production readiness means more than receiving HTTP 200. Verify dependencies, migrations, provider settings, webhooks and one complete customer journey.

Deployment checklist

- The production health endpoint reports database connected.
- The database provider matches the intended production host.
- All migrations have run on that exact database.
- Telnyx reports configured before enabling number or SMS buttons.
- Webhook signing is configured and a signed delivery has been verified.
- Pricing inputs are configured for every enabled destination.
- An actual test number search, purchase and message has been reconciled.
- Rate-limit and insufficient-credit screens are tested.
- API keys can be rotated and revoked without downtime.
- Dashboards alert on provider unknown outcomes and failed deliveries.

Database note

The server uses portable PostgreSQL through the pg driver. The live health response on 16 October 2026 reported neondb. A migration copy may exist elsewhere, but that does not make it the active production database. Verify the health response after every cutover.

CHAPTER 15

Local development and deployment

Run the static portal and functions from one origin, apply migrations with a direct database connection, and keep provider credentials out of the repository.

LOCAL SETUP

```
cp .env.example .env.local
npm install
npm run db:migrate
npx vercel dev
npm run check
```

Server environment

METHOD	PATH	PURPOSE
DATABASE	DATABASE_URL	Pooled PostgreSQL connection.
DATABASE	DATABASE_URL_UNPOOLED	Direct connection for migrations.
AUTH	SUPABASE_URL	Supabase Auth project URL.
AUTH	SUPABASE_PUBLISHABLE_KEY	Publishable Auth key.
PHONE	TELNYX_API_KEY	Carrier API credential.
PHONE	TELNYX_PUBLIC_KEY	Verify carrier webhooks.
PHONE	TELNYX_MESSAGING_PROFILE_ID	Outbound messaging profile.
EVENTS	VEDOY_WEBHOOK_SIGNING_SECRET	Derives endpoint secrets; 32+ chars.

Never commit
Database URLs, service-role credentials, carrier API keys, webhook secrets and generated Vedøy API keys must remain server-side.

CHAPTER 16

Use the design system consistently

The portal also demonstrates reusable Vedøy UI elements for customer products: cards, actions, forms, code panels, statuses, responsive navigation and motion.

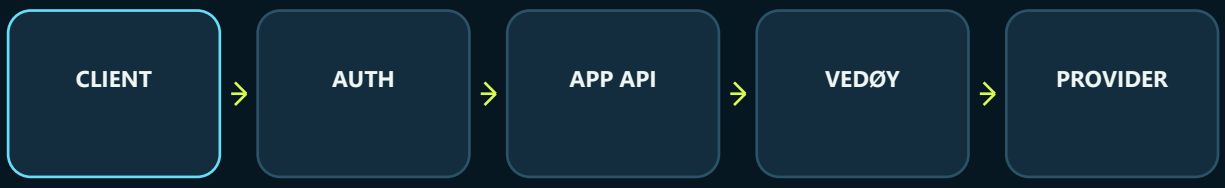
- Keep spacing and typography tokens consistent across products.
- Use one status language: success, pending, blocked, failed and unavailable.
- Make primary actions obvious and destructive actions deliberate.
- Preserve keyboard focus and reduced-motion behavior.
- Keep code examples copyable and pair them with visible simulated responses.
- Use responsive drawers instead of shrinking a desktop sidebar into an unusable strip.
- Use animation to explain state or hierarchy, never to hide controls.

> 3D opening
> The 3D developer room introduces the platform before the portal. It is an orientation experience; the documentation and API console remain the task-focused workspace.

CHAPTER 17

Reference architecture

A production phone product needs clear boundaries between identity, application permissions, billable provider work and event processing.



The client authenticates a person. Your app API maps that person to a project and verifies the requested resource. Vedøy authorizes the project key, reserves credits and performs the provider operation. Signed events bring asynchronous results back to your system.

Recommended services

METHOD	PATH	PURPOSE
EDGE	Application API	User authorization, validation and request IDs.
WORKER	Event processor	Deduplication, durable retries and state transitions.
STORE	Product database	Users, project mappings, consent and business objects.
SECRET	Secret manager	Vedøy key and webhook endpoint secret.
OBSERVE	Logs and metrics	Latency, status, credits, delivery and reconciliation.

CHAPTER 18

End-to-end example

This sequence shows how a customer sends one message from Vedøy Phone without exposing a project secret.

- The customer signs in to Vedøy Phone.
- The app submits the recipient and text to its backend.
- The backend verifies the user may use the selected project-owned number.
- The backend generates an operation ID and calls Vedøy /v1/messages.
- Vedøy reserves credits and submits the message to the configured provider.
- The backend stores the 202 response as accepted.
- A signed webhook updates the final status.
- The UI shows delivered, failed or still pending.

SERVER ROUTE

```
// Simplified app-backend route
export async function sendMessage(req, res) {
  const user = await requireAppUser(req);
  const input = validateMessage(req.body);
  await requireNumberPermission(user, input.from);

  const operationId = req.headers['x-operation-id'] || crypto.randomUUID();
  const response = await fetch(`${VEDOY_API}/v1/messages`, {
    method: 'POST',
    headers: {
      Authorization: `Bearer ${process.env.VEDOY_API_KEY}`,
      'Content-Type': 'application/json',
      'Idempotency-Key': operationId
    },
    body: JSON.stringify(input)
  });

  const payload = await response.json();
  return res.status(response.status).json(payload);
}
```

CHAPTER 19

What is live and what comes next

The repository contains a substantial API control plane. Capability status must still be based on deployed configuration and completed end-to-end tests.

METHOD	PATH	PURPOSE
LIVE	Portal and docs	Published developer experience and contract.
LIVE	Health endpoint	Reachable; database connected at verification time.
BUILT	Projects and keys	Implemented with user and project scopes.
BUILT	Usage and credits	PostgreSQL ledger and usage model.
BUILT	Webhooks	Signed destinations, deliveries and manual retry.
BLOCKED	Telnyx operations	Production health reports not configured.
PREVIEW	Agent runs	Definitions stored; runner intentionally blocked.
MISSING	Central voice calling	No voice-call endpoint in current OpenAPI contract.

The next production milestone

Configure the carrier in the production environment, redeploy, verify health, run one controlled number purchase and SMS, verify signed status events, reconcile the ledger, and only then enable those actions for customers.

CHAPTER 20

Appendix

Keep these references beside your implementation and update generated clients when the contract version changes.

Links

- Developer portal: https://vedoy-developers.eiolf-leande-6585.chatgpt.site/
- Production API: https://vedoy-dev-portal.vercel.app/api
- OpenAPI contract: https://vedoy-dev-portal.vercel.app/openapi.yaml
- Health: https://vedoy-dev-portal.vercel.app/api/v1/health

Minimal go-live gate

- Provider configured in health.
- Correct production database active.
- Migrations complete.
- Secrets server-side.
- Real end-to-end action succeeds.
- Webhook signature verified.
- Unknown outcomes reconciled.
- User consent and regional requirements reviewed.

> Build carefully
> The contract gives you a clean surface. A reliable product comes from protecting the key, validating permission, handling asynchronous state and telling users the truth about what has completed.