froddy.

A control layer before each payout

One API call before each payout — the system returns a verdict: allow / hold-for-review / block.

PSP · CPA Networks · iGaming · Fintech

froddy.io

PROBLEM

Payout errors are not fraud. They are failures in your own automation

Most losses in automated payouts come not from external attacks but from duplicates, broken limits, and logic errors. Internal controls miss them because they are part of the same system.

Duplicate payouts

A logic error or race condition triggers the same payout multiple times. Caught later during reconciliation.

Broken limits

Limits are misconfigured or fail to trigger. Payouts keep going out until someone notices the problem.

Payout logic errors

Config changed — control broke. Payouts keep going out.

Common thread: controls lived inside the same system. When execution breaks — controls break with it.

DATA

Public estimates of losses from erroneous payouts (2024–2025)

GLOBAL · B2B AP

0.8–2.0%

in duplicate and erroneous payouts as a share of annual volume

Top performers — 0.8%, bottom — 2.0%. Geography: global survey. Source: APQC Open Standards Benchmarking, 2024

GLOBAL · BANKS

€8B

in bank losses from transaction failures

Execution and delivery failures are the costliest operational risk in banking. Source: ORX Global Op. Risk Report, 2024

EU · BUDGET

3.6% / €7.1B

in incorrect EU budget payments

Administrative errors, not fraud. Down from 5.6% (2023). Source: European Court of Auditors, Annual Report, 2024

RUSSIA · BANKS

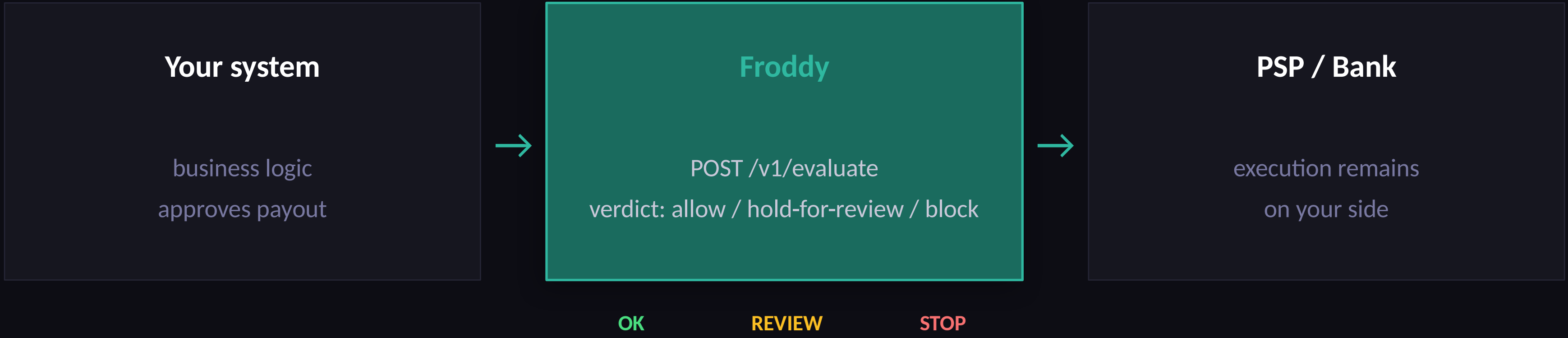
RUB 20B

in annual bank losses from payout errors

Only 47% is recovered. Duplicate charges and process errors, not fraud. Source: T1 Innotech, 2024

POSITION IN THE FLOW

One layer between your logic and money movement



— Froddy does not move money. It returns a verdict — what to do next is decided by your system.

— Shadow mode: Froddy evaluates and logs without intervening in payouts. Enforcement mode is enabled separately.

— Default behavior is pass-through. What happens when Froddy is unavailable is controlled on your side.

PRODUCT

What Froddy actually does

DUPLICATES

Duplicate check

A repeated request returns the same verdict without re-evaluation.

LIMITS

Limits

Limits on single payout amount, total daily payout volume, and transaction frequency over a period.

REVIEW

Hold for review

Hold-for-review means: send for manual review. Your system then acts according to your rules.

LOG

Decision log

Every decision is stored with the rule, data, outcome, and evaluation timestamp.

MARKET CONTEXT

Companies build this themselves — from fragments

Now — without Froddy

✗ Provider/bank payout infrastructure

✗ Rules — in-house or point solution

✗ Reconciliation — separate vendor or manual

✗ Logs — scattered, no unified decision log

✗ Rule changes — manual process

→

With Froddy

✓ Single rules layer on top of any execution

✓ Duplicate and repeat request detection

✓ Limits by amount, daily payout volume, and transaction frequency

✓ Decision log — all checks and decisions in one place

✓ Rules updated via API with version history

These controls are usually built in-house: separate logic, maintenance, and operational overhead. Froddy is the same class of controls as an external layer.

AUDIENCE

Who needs this

Payment providers / PSP

Automated merchant settlements, bulk payouts. Need control before money reaches the bank/provider.

CPA networks / partners

Performance-based partner payouts. Duplicates, unusual frequency, and linked accounts are the main risks.

iGaming

Player withdrawals and affiliate payouts. High frequency, unusual VIP payout volumes, bonus anomalies.

Fintech / lending / wallets

Automated payouts and transfers. Race conditions and request retries are common sources of duplicates.

Operations / risk / control

Teams that need a unified decision log before money moves — without building it in-house.

Product / engineering

Founders and product/engineering leads who need payout safeguards without a dedicated rules service.

HOW IT WORKS

One call — one verdict — one log

1 **Before each payout**

Your system sends POST /v1/evaluate with the recipient identifier (entity_id), amount, and event key (event_id).

2 **Froddy evaluates the request**

Checks run in sequence: payout amount, daily limit, transaction frequency, duplicates, and repeat requests.

3 **Verdict is returned synchronously**

allow → payout proceeds. hold-for-review → send for manual review in your system. block → reject or route for investigation.

4 **Decision is logged**

Every check is stored: what was evaluated, what the values were, what verdict was issued, and when.

```
POST /v1/evaluate

{
  "event_id":  "payout_42301",
  "entity_id": "partner_88",
  "amount":    15000.00,
  "currency":  "USD",
  "event_type": "payout"
}

// Response:
{
  "verdict": "hold-for-review",
  "reason":  "daily: $53k / $50k"
}
```

Shadow mode (default): all verdicts are advisory. Froddy evaluates but does not stop payouts. Enforcement mode is enabled separately.

VALUE

Why this is useful for the business

01

Fewer costly simple errors

Duplicate payouts and broken limits are not rare. Pre-payout control catches them before money leaves.

02

Less manual investigation

Selected payouts go for review. Everything else passes automatically.

03

Clearer payout rules

You can precisely reconstruct which rules drove any given decision.

04

Audit log

Always clear: why a payout passed or was stopped, and which rules did not trigger.

HOW TO GET STARTED

Three steps from demo to launch

1

Demo

Understand the scenario

Try the interactive demo at froddy.io — request / response / decision log in action, no integration required.

froddy.io/demo

→

2

Integration

Shadow mode

Create an account and get your project API key. POST /v1/evaluate before your payouts. Shadow mode — no automatic intervention in execution.

froddy.io/signup

→

3

Enforcement mode

When ready

Start with observation and no blocking. Once limits are configured, enforcement mode can be enabled separately.

Configured via the rules API

froddy.

See the product in action

Demo, documentation, and onboarding — at froddy.io

Open demo

Start for free

Email:	hello@froddy.io
Telegram:	@froddysup
Website:	froddy.io