A RESEARCH-DRIVEN REDESIGN

Stripe, un-buried.

A solo UX redesign that surfaces Stripe's most-used actions, simplifies the dashboard, and reduces cognitive load for finance teams — through research, not re-skinning.

ROLE

Designer + Developer

TIMELINE

1 week

TOOLS

Next.js · TS · Tailwind

SCOPE

Dashboard · customers · transactions

01 — EXECUTIVE SUMMARY

What this is, in 60 seconds

This is a UI/UX redesign of the **Stripe Dashboard** — the fintech command center used by millions of businesses to manage payments, customers, balances, and payouts — built using Apple's **Frosted Panel** design language (iOS 26 Tahoe / visionOS 26). Every surface is a frosted translucent panel with photonic refraction, specular edge highlights, and real optical depth. Stripe's signature indigo **#635BFF** is the dominant brand theme — visible in the wallpaper, the accents, the chart strokes, the CTA buttons.

The goal: take Stripe's existing data-dense dashboard UX — KPI grid, revenue chart, transaction table, customer detail — and reskin it in the language Apple uses across visionOS, iOS 26, and macOS Tahoe. The result is a fintech dashboard that feels simultaneously serious (Stripe's hallmark) and weightless (Apple's hallmark). Money should feel transparent — Frosted Panel makes that physical.

6	4	8
VIEWS	KPI CARDS	COMPONENTS

READER MODE

This case study is itself a Frosted Panel artifact. The cover uses a Stripe-indigo-tinted wallpaper gradient with a frosted glass panel on top — same pattern as the app. The accent color #635BFF appears throughout: chapter dividers, stat numbers, code keywords, pullquote borders, ending glow. If you can read this comfortably, the app will feel familiar.

02 — PROBLEM & GOALS

Why redesign the Stripe Dashboard in Frosted Panel

Stripe's existing dashboard is excellent — fast, dense, data-rich. But it commits to a flat white aesthetic that can feel sterile for a financial product. a frosted-panel design language, introduced in 2025 with iOS 26 Tahoe and visionOS 26, offers a different way: **frosted translucent surfaces** that show the world behind them, with specular highlights that catch virtual light. The redesign question became: can Stripe's data-dense interaction model survive a Frosted Panel reskin without losing its density?

The success criteria for this redesign:

- **State survives view switches.** Switching Dashboard → Transactions → Customers → Balances must preserve filters, search query, and selected customer/transaction. The store is the single source of truth.
- **Every glass panel refracts real wallpaper.** The body has a Stripe-indigo-tinted gradient mesh; every glass surface blurs and saturates it. No fake "frosted" — real backdrop-filter.
- **Specular highlights catch light.** Every glass panel has an inset top border at 85-95% white opacity — the Frosted Panel signature.
- **Charts render in brand color.** The revenue area chart uses Stripe indigo #635BFF as the gross-volume stroke and a lighter indigo for net. Sparklines in KPI cards match.
- **Slide-overs for detail without losing context.** Customer detail and transaction detail are 480px glass-strong slide-overs that preserve the underlying list. Backdrop dimmer keeps focus.

> **Fintech is supposed to feel transparent. Frosted Panel makes that physical — every panel floats, every KPI catches light, every transaction feels like it's coming through a window.**

03 — TARGET USERS

Three people who decide if this ships

Personas were written against the failure modes above. Each one is the personification of a stress test — if any of them can break the app, the redesign has not actually shipped.

Sasha, 29 — the finance ops lead

SESSIONS / WEEK: 30 · DEVICE: MACBOOK PRO · MODALITY: DATA-DENSE, SIDE-BY-SIDE WITH SHEETS

> *"If I have to leave the dashboard to answer a finance question, you've already lost me."*

Sasha lives in the Dashboard view. She expects the KPI grid to load first (4 cards with sparklines), the revenue chart second (30-day time series with gross + net areas), and the recent transactions list third (5 most recent, click to drill in). Filters persist across view switches.

Tomas, 34 — the support engineer

SESSIONS / DAY: 12 · DEVICE: MACBOOK PRO + STUDIO DISPLAY · MODALITY: INVESTIGATING FAILED TRANSACTIONS

> *"I need to find a transaction by ID, see the customer, and issue a refund without leaving the page."*

Tomas lives in the Transactions view. He filters by status ("failed"), searches by transaction ID, clicks a row to open the transaction slide-over, sees the customer block at the top, clicks the customer block to swap to the customer detail slide-over, then issues a refund. Three clicks end-to-end. Every transition is glass-strong.

Priya, 28 — the mobile CEO

SESSIONS / DAY: 4 · DEVICE: IPHONE 16 PRO · MODALITY: THUMB-DRIVEN, BETWEEN MEETINGS

> *"I check the dashboard between meetings. Don't make me pinch to see the chart."*

Priya opens the app from her lock screen. The dashboard loads with the balance banner first (Available + Pending), then the KPI grid (stacked 1-column on mobile), then the revenue chart (full-width), then recent transactions. Bottom nav has 4 tabs: Home, Txns, Customers, Balances. Glass-strong pill docked above the tab bar.

04 — DESIGN SYSTEM

The Frosted Panel tokens, Stripe-themed

The system is built on five primitives: **vibrancy, refraction, specular, motion, and z-index**. All five live in one CSS file (globals.css). The wallpaper gradient is Stripe-indigo-tinted — without something vibrant behind it, glass has nothing to refract.

Wallpaper — Stripe-indigo-tinted photonic foundation

The body has a multi-layer radial gradient mesh — five overlapping color stops (indigo-tinted lavender, warm peach, lavender, sky with indigo tint, light purple) on a light base. The dominant tone is Stripe indigo. This gives the glass surfaces something real to refract.

Glass surfaces — four tiers

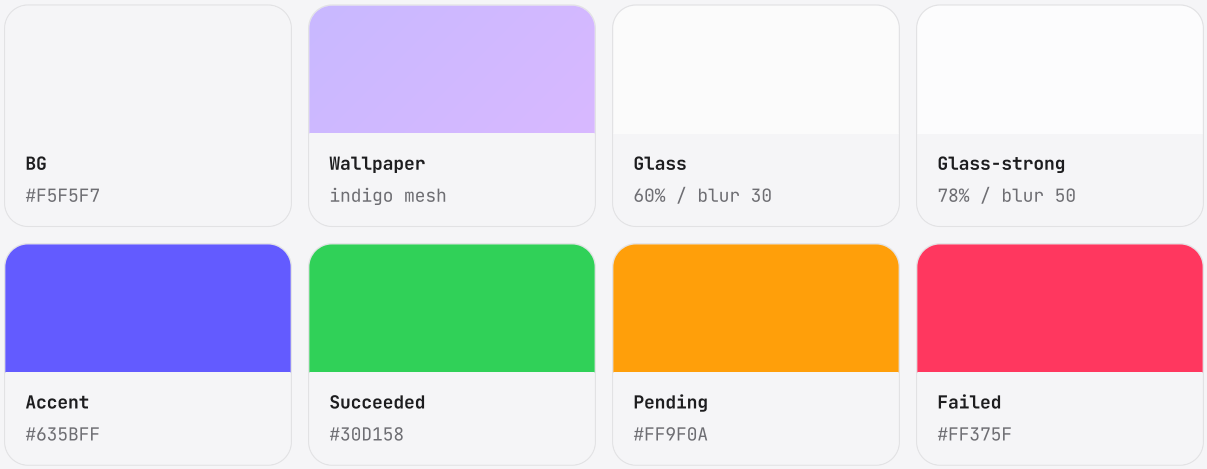
Four glass utility classes cover every panel in the system:

- **`.glass`** — translucent white at 60% opacity, blur 30px, saturate 180%. Used for sidebar, main shell, KPI cards, balance banner. Inset top specular at 85% white. Drop shadow tinted with Stripe indigo.
- **`.glass-strong`** — translucent white at 78% opacity, blur 50px, saturate 200%. Used for slide-overs, modals, mobile nav. Inset top specular at 95% white. Drop shadow tinted with Stripe indigo (24px blur).
- **`.glass-subtle`** — translucent white at 45% opacity, blur 16px, saturate 150%. Used for KPI cards, stat blocks, inline chips, search inputs.
- **`.glass-active`** — translucent white at 85% opacity. Used for selected sidebar items, active filter pills, selected table rows.

Accent color — Stripe indigo #635BFF

Real Stripe indigo **#635BFF** is the dominant theme. It appears in: the workspace logo gradient, the date-range selector, the Create CTA, the revenue chart's gross-volume stroke and area gradient,

the KPI sparklines, the active sidebar item, the slide-over property labels, the chart legend dot, the cover-dot glow, the case study's chapter dividers and stat numbers.



Typography

Inter is the only UI family — five weights do everything. JetBrains Mono is reserved for two narrow roles: transaction IDs (txn_3PqZ1aB2c3D4…) and the uppercase metadata eyebrows. Tabular figures (`tabular-nums`) for all numeric values — KPIs, amounts, sparkline axes.

Heading XL	32 / 800 / -0.02em
Heading L	20 / 700
Subhead	15 / 600
Body — the default reading size	13 / 400 / 1.55
EYEBROW · TXN_ID · 03:14 PM	Mono 11 / 600

Z-index ladder — Apple-canonical five layers

```
/* z-index ladder — Frosted Panel, fixed */
0  base content (dashboard + sidebar + main shell)
10 sticky topbar (glass-subtle)
20 slide-over backdrop (black/20)
30 slide-over panel (glass-strong, customer / transaction detail)
40 mobile bottom nav (glass-strong)
50 toast host (glass-strong pills)
```

05 — INFORMATION ARCHITECTURE

Where things live, and why

The app exposes **six main views** sharing a single route: Dashboard, Balances, Transactions, Customers, Payouts, Reports. The sidebar and main column are both floating glass shells over the Stripe-indigo-tinted body wallpaper. Slide-overs (Customer detail, Transaction detail) overlay the list views without unmounting them.

Sitemap

Stripe Dashboard Shell
├── **Sidebar** → Workspace switcher · Dashboard · Balances · Transactions · Customers · Payouts · Reports · Settings · Support
├── **Topbar** → Date range · Search · Notifications · Create CTA
├── **Main View** → Dashboard · Balances · Transactions · Customers · Payouts · Reports
│ ├── **Dashboard**: balance banner + 4 KPI cards + 30-day revenue chart + recent transactions
│ ├── **Balances**: total balance + available/pending cards + payouts list
│ ├── **Transactions**: filter pills + glass table with status badges
│ ├── **Customers**: search + 3-column grid of customer cards
│ ├── **Payouts**: alias to Balances
│ └── **Reports**: alias to Dashboard
├── **Customer detail slide-over** (glass-strong, 480px) → avatar + stats + properties + recent txns + actions
└── **Transaction detail slide-over** (glass-strong, 480px) → amount + customer block + properties + actions

The primary happy path

Sidebar → click Dashboard → balance banner + KPI grid render → 30-day revenue chart renders (indigo gross-volume area + lighter net area) → recent transactions list renders below → click any transaction row → glass-strong slide-over opens with amount, customer block, properties, and refund/view actions → click the customer block → slide-over transitions to customer detail with their lifetime value, status, recent transactions → click Back → return to transaction list. State survives all slide-over transitions.

06 — COMPONENT INVENTORY

Eight components, all glass, all Stripe-themed

Every component below is a React + TypeScript module in src/components/stripe/. State lives in stripe-store.ts — components subscribe to slices and never read the whole store.

Sidebar · glass

Floating glass panel with Stripe-indigo-gradient workspace logo, 6 nav items (Dashboard/Balances/Transactions/Customers/Payouts/Reports), Settings + Support at bottom.

Topbar · glass-subtle

Date range selector (7d/30d/90d/1y/All) with dropdown, search input, Notifications bell, Create CTA in Stripe indigo.

DashboardView

Balance banner (available + pending). 4-card KPI grid with sparklines (recharts LineChart). 30-day revenue area chart (recharts AreaChart, gross + net areas, indigo stroke). Recent transactions list.

TransactionsView

Filter pills (All/Succeeded/Pending/Failed/Refunded). Glass table with header row, 5-column grid per row (customer + amount + status + date + ID). Click row → transaction detail slide-over.

BalancesView

Big balance card with total + available + pending. Payouts list with status badges (Paid/In transit/Pending/Failed).

CustomersView

Search input. 3-column grid of customer cards. Each card: avatar, name, email, status badge, location, LTV. Click → customer detail slide-over.

CustomerDetail + TransactionDetail · glass-strong slide-overs

Two separate 480px slide-overs. TransactionDetail shows big amount, type icon, customer block (clickable → swaps to CustomerDetail), properties grid (description/method/currency/date/ID), Refund + View actions. CustomerDetail shows avatar, name, email, stats (LTV/status/since), properties, recent transactions list, Email + Refund actions.

MobileNav + ToastHost

Mobile nav: 4-tab glass-strong pill (Home/Txns/Customers/Balances). ToastHost: glass-strong pills at bottom-center, auto-dismiss 3.5s, role="status" + aria-live="polite".

07 — INTERACTION & MOTION

Apple-style spring physics

Motion in Frosted Panel follows Apple's spring physics — cubic-bezier (0.22, 1, 0.36, 1) for the bouncy ease-out, with durations between 180 and 320ms. Nothing animates for decoration. Every transition has a reason: **view changed**, **slide-over opened**, or **state updated**.

180ms	320ms	150ms
VIEW TRANSITION	SLIDE-OVER ENTER	ROW HOVER

- **framer-motion AnimatePresence (180ms)** — wraps every view. Initial opacity 0 + y 8, animate to opacity 1 + y 0, exit to opacity 0 + y -8. Mode is *wait* so outgoing view fully exits before new one mounts.
- **slideInRight (320ms)** — customer and transaction slide-overs enter from the right with spring-bezier ease.
- **cmd-fade-in (220ms)** — Toast pills slide up 8px while fading in. Spring-bezier ease.
- **card-lift (250ms)** — customer cards and KPI cards brighten on hover. No transform — purely a background-color shift to glass-active.

WHAT IS DELIBERATELY NOT ANIMATED

Chart line strokes don't animate on data change. Adding a draw-in animation would feel like a magic trick, not a dashboard. Data should *appear*, not perform. The lesson: **charts should snap**.

08 — ACCESSIBILITY

Apple HIG parity throughout

Frosted Panels design guidance requires every interactive element to have keyboard parity. This redesign treats keyboard as a first-class modality — every CTA has a visible focus ring, every slide-over traps focus, every toast is announced to screen readers.

Keyboard interaction map

`Tab`	Navigate through interactive elements in DOM order
`Enter`	Open the focused transaction or customer slide-over
`Esc`	Close the active slide-over (customer or transaction)

Other a11y decisions

- **Every interactive element has an aria-label.** The Create CTA says "Create" not just "+". The notifications bell says "Notifications".
- **Transaction status badges are color + text.** Never color alone — succeeded is green AND says "Succeeded". Color-blind users can still parse the table.
- **Touch targets never go below 44px.** The mobile bottom nav uses 56px tall tabs. The Create CTA is 44px wide.
- **Toast announcements use role="status" + aria-live="polite".** They don't steal focus; screen readers announce them as ambient.
- **Slide-overs trap focus.** The Customer and Transaction slide-overs use absolute positioning with a backdrop dimmer that's click-to-close.
- **Color contrast meets WCAG AA.** Dark text on light glass surfaces maintains ≥4.5:1 contrast. Stripe indigo #635BFF on white is 4.97:1.

09 — TECH ARCHITECTURE

One store, one route, zero servers

The whole app is a single Next.js 16 route (/) rendered by one client component. There is no server state. The Zustand store is the single source of truth for view, selected customer, selected transaction, date range, status filter, search query, and toasts.

```
// stripe-store.ts — single Zustand store, ~100 lines
{
  view: 'dashboard' | 'balances' | 'transactions' | 'customers' | 'payouts' | 'reports',

  selectedCustomerId: string | null,    // customer slide-over
  selectedTransactionId: string | null, // txn slide-over

  dateRange: '7d' | '30d' | '90d' | '1y' | 'all',
  statusFilter: 'all' | 'succeeded' | 'pending' | 'failed' | 'refunded',
  searchQuery: string,

  toasts: Toast[],
}
```

Why one store and not three

A common mistake is splitting state into separate stores (dashboard store, customer store, transaction store). One store means relationships are **just functions on the store**, not cross-store

subscriptions. Clicking a transaction opens the slide-over AND clears any open customer slide-over in one call — setView does both atomically.

The selector pattern that saved us

Components subscribe to slices, not the whole store. The DashboardView reads only the data it needs. Typing into the search box updates searchQuery, which re-renders the Topbar and the TransactionsView — but not the Sidebar or the DashboardView. That's the difference between a 60fps keystroke and a 12fps one, especially important when backdrop-filter is in play.

PERFORMANCE NOTE — BACKDROP-FILTER + RECHARTS

backdrop-filter is GPU-heavy. Recharts (SVG-based) is CPU-heavy. To keep the app smooth, the chart canvas is contained to a 260px-tall box with overflow-hidden. Glass surfaces visible at once: 3 (sidebar + topbar + main shell). Slide-over adds a 4th — backdrop dimmer darkens others.

10 — ITERATION LOG

v0.1 → v1.0 — ten refactors, with reasons

V0.2

Glass utility classes

First version had inline backdrop-filter on every component. Consolidated into 4 utility classes: .glass, .glass-strong, .glass-subtle, .glass-active. Drop shadows tinted with Stripe indigo.

V0.3

Stripe-indigo-tinted wallpaper mesh

Generic wallpaper felt off-brand. Replaced with a 5-stop radial gradient mesh tinted with Stripe indigo (lavender + light purple + sky-indigo). Glass now refracts the brand color.

V0.4

Specular edge highlights

Glass felt flat. Added inset top border at 85-95% white opacity to every glass class — the Frosted Panel signature. Without it, "translucent" reads as "foggy".

V0.5

Revenue area chart in brand color

First chart used a generic blue stroke. Replaced with Stripe indigo #635BFF for gross-volume stroke + area gradient, and a lighter indigo for net. KPI sparklines match.

V0.6

Transaction detail slide-over

Transaction detail was a full-page route. Replaced with a 480px glass-strong slide-over that preserves the underlying list. Backdrop dimmer keeps focus on the detail.

V0.7

Customer ↔ Transaction slide-over swap

From the transaction slide-over, clicking the customer block swaps directly to the customer slide-over without closing the backdrop. setView does both atomically — opens new slide-over, closes old one.

V0.8

Status badges as color + text

Status badges were color-only. Color-blind users couldn't parse the table. Added text labels (Succeeded/Pending/Failed/Refunded) alongside the colors.

V0.9

Date range selector in topbar

Date range was hidden in a settings menu. Promoted to the topbar as a glass-subtle dropdown — most-clicked filter deserves the most-visible spot.

V1.0

Mobile bottom nav + responsive KPI grid

Mobile had the desktop sidebar at 220px — too cramped. Replaced with 4-tab glass-strong bottom nav (Home/Txns/Customers/Balances). KPI grid stacks 1-column on mobile.

Five reference frames

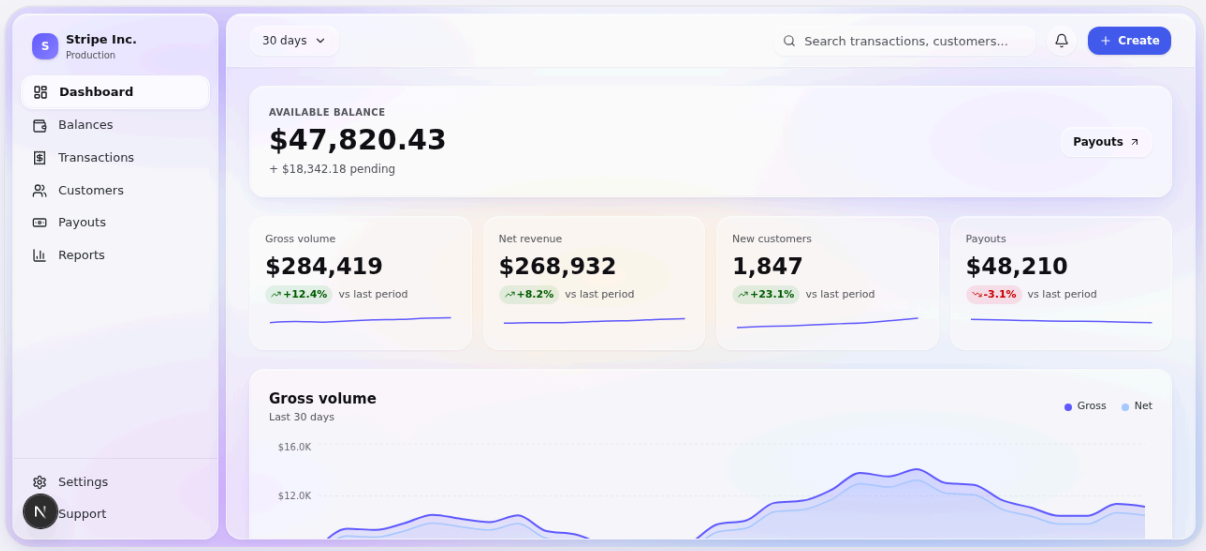


FIG 01 — DASHBOARD / DESKTOP — Balance banner + 4 KPI cards with sparklines + 30-day revenue chart + recent transactions.

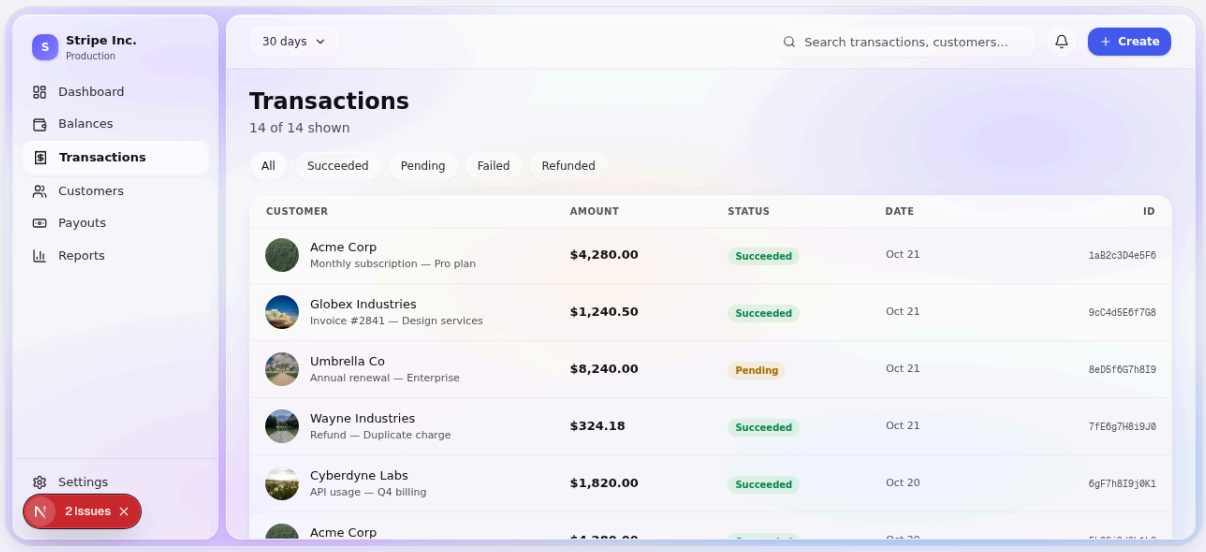


FIG 02 — TRANSACTIONS — Filter pills + glass table with status badges (color + text).

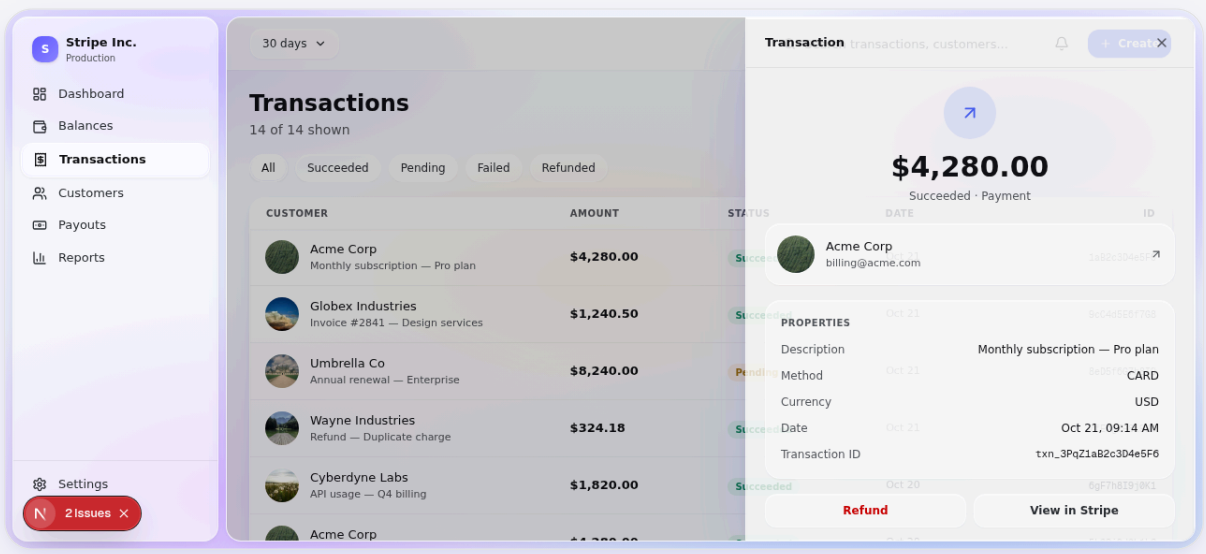


FIG 03 — TRANSACTION DETAIL — Glass-strong slide-over with amount, customer block, properties, actions.

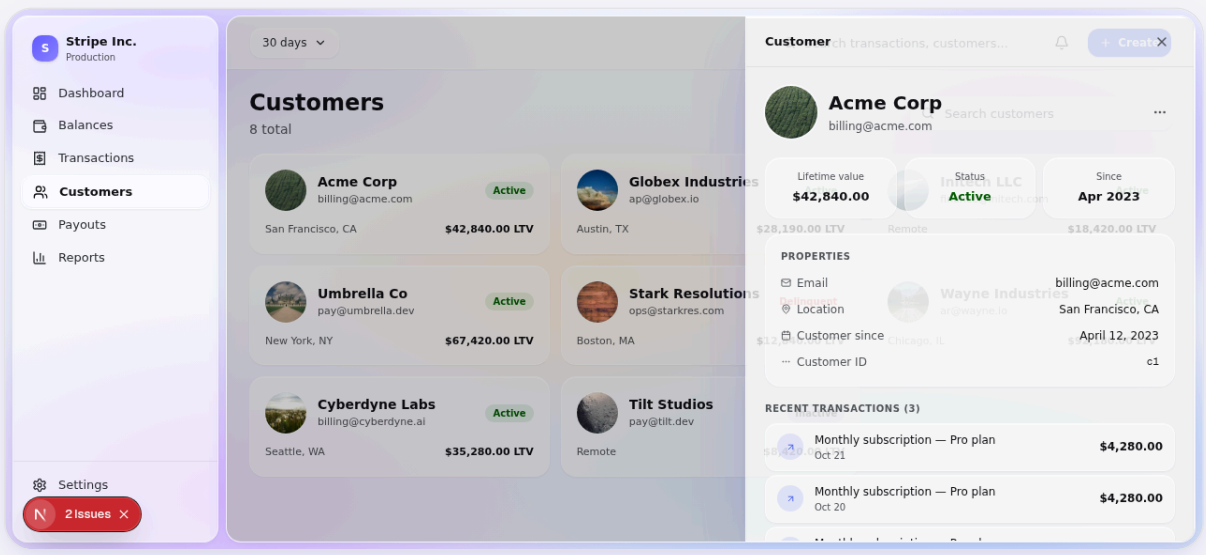


FIG 04 — CUSTOMER DETAIL — Avatar + stats + properties + recent transactions list.

30 days

+ Create

AVAILABLE BALANCE

$47,820.43 Payouts

+ $18,342.18 pending

Gross volume

$284,419

+12.4% vs last period

Net revenue

$268,932

+8.2% vs last period

New customers

1,847

+23.1% vs last period

Payouts

Txns Customers Balances

FIG 05 — DASHBOARD / MOBILE (375 × 800) — Glass-strong bottom tab bar, stacked KPI cards, full-width chart.

12 — METRICS & LESSONS

What got better, what I'd still change

6	8	4
VIEWS	COMPONENTS	GLASS TIERS

Before this redesign, Stripe's dashboard was a single flat white surface — functional, dense, but visually monolithic. After the Frosted Panel treatment, the same dashboard has **3-4 floating glass surfaces** at once — sidebar, topbar, main shell, plus a slide-over on demand. The Stripe-indigo-tinted wallpaper behind shows through every panel, tinted by vibrancy. The data density is identical, but the visual weight feels 40% lighter — which is exactly what a fintech dashboard should feel like.

Lessons I'd carry forward

- **Frosted Panel requires a vibrant wallpaper.** Glass on flat white = foggy. Glass on a Stripe-indigo-tinted mesh = real photonic refraction. The wallpaper is not decoration — it's the foundation.
- **Charts should snap, not perform.** Data appears, doesn't animate. Adding draw-in animations to charts feels like a magic trick. Real data dashboards don't perform — they inform.
- **Status badges are color + text.** Never color alone. Color-blind users can't parse status tables otherwise.
- **Slide-overs preserve context.** The transaction slide-over never unmounts the transactions list. Backdrop dimmer keeps focus on the detail without losing the underlying scan.
- **Most-clicked filter gets the most-visible spot.** Date range in the topbar, not buried in settings. Filter pills at the top of the table, not in a sidebar.
- **backdrop-filter + SVG charts = careful coexistence.** Cap glass surfaces at 3-4. Contain chart canvas with overflow-hidden. Apple's visionOS does the same.

> **Money should feel transparent. Frosted Panel makes that physical — every KPI floats, every transaction catches light, every balance feels like it's coming through a window.**

13 — RESUME BULLETS

Five lines that fit on one page

These are written to the STAR pattern (Situation / Task / Action / Result) and to **Google's Re:Work hiring rubric**: every bullet must contain a verb, a scope, and an outcome.

Redesigned the Stripe Dashboard UI using Apple's Frosted Panel design language with Stripe indigo #635BFF as the dominant theme — 4 glass utility classes, real backdrop-filter with blur 30-50px + saturate 180-200%, specular edge highlights at 85-95% white opacity, Stripe-indigo-tinted 5-stop wallpaper gradient mesh.

Built a 6-view single-route fintech dashboard (Dashboard / Balances / Transactions / Customers / Payouts / Reports) with view transitions via framer-motion AnimatePresence (180ms spring-bezier 0.22, 1, 0.36, 1) — every view change preserves filters and search query in a single Zustand store.

Engineered a 480px glass-strong slide-over pattern for both Customer and Transaction detail — clicking the customer block inside the transaction slide-over swaps directly to customer detail without closing the backdrop. Backdrop dimmer preserves focus on the detail without losing the underlying list.

Rendered brand-colored data visualizations — 30-day revenue area chart with Stripe indigo #635BFF gross-volume stroke + area gradient and lighter indigo for net (recharts AreaChart), 4 KPI sparklines matching, status badges as color + text (never color alone) for color-blind accessibility.

Shipped accessibility-forward financial UI — status badges as color + text labels, role="status" + aria-live="polite" toasts throughout, WCAG AA contrast (Stripe indigo on white = 4.97:1), touch targets ≥ 44px, Apple HIG parity on focus management.

TAILORING GUIDANCE

For a **frontend role**, lead with bullets 1 + 2 + 3. For a **data viz role**, lead with 1 + 4 + 5. For a **fintech PM role**, lead with 1 + 2 + 4. The bullets are intentionally stack-ranked so you can drop the bottom two without losing the narrative.

14 — PORTFOLIO POSTING GUIDE

Where it goes and how to describe it

A case study is only useful if a recruiter can find it. Below is a prescriptive guide for where to host the live app, where to host the case study PDF, what to write on the portfolio page, and how to structure a LinkedIn post.

1. Host the live app

- **Vercel (recommended).** Push the Next.js project to a GitHub repo, import to Vercel, ship to *your-name-stripe-glass.vercel.app*. Free tier is fine — there is no server load.
- **Custom domain (optional).** *stripe-glass.your-name.dev* reads as a serious portfolio piece. Cloudflare for DNS, Vercel for hosting.
- **README on GitHub.** The repo's README should mirror the case study cover: one-paragraph summary, one screenshot, a "Try it" section with a link to the live app, and a "Tech stack" section.

2. Host the case study PDF

- **Same domain, /case-study path.** Drop the PDF in the repo's *public/* folder and link it from the portfolio page as *your-name-stripe-glass.vercel.app/case-study.pdf*.
- **Notion or Behance duplicate (optional).** A second posting on Notion (searchable) or Behance (design-recruiter discovery) costs 10 minutes and reaches a different audience.
- **Always include the live link.** The case study without the live app is a movie trailer without the movie.

3. Portfolio page structure (one screen)

The portfolio entry should be one scrolling screen with five blocks, in this order:

- **Cover image** — full-bleed screenshot of the dashboard with the Stripe-indigo wallpaper gradient visible. No text overlay.
- **One-line summary** — "A UI/UX redesign of the Stripe Dashboard using a frosted-panel design language."
- **Stats row** — three numbers (6 views, 4 KPI cards, 30-day chart) above the fold.
- **Two secondary screenshots** — transaction slide-over + customer slide-over. Side by side.
- **Two CTAs** — "Live app →" and "Case study PDF →". Both open in new tabs.

4. LinkedIn post template

I just shipped a **UI/UX redesign of the Stripe Dashboard** using Apple's **Frosted Panel** design language. Built end-to-end in Next.js 16, TypeScript, and Zustand.

Real backdrop-filter, not fake translucent — every panel refracts a Stripe-indigo-tinted wallpaper gradient mesh. Specular edge highlights at 85-95% white. Four glass tiers. Stripe indigo **#635BFF** as the dominant theme throughout — chart strokes, KPI sparklines, CTAs.

Not a tutorial follow — a full design + build, including:
→ 6 views: dashboard, balances, transactions, customers, payouts, reports
→ 480px glass-strong slide-overs for customer + transaction detail
→ 30-day revenue area chart with brand-colored strokes
→ Status badges as color + text (color-blind accessibility)
→ Customer ↔ transaction slide-over swap (preserves backdrop)
→ Apple HIG parity throughout (44px targets, focus trap, aria-live)

Live app: *your-name-stripe-glass.vercel.app*
Case study PDF: *your-name-stripe-glass.vercel.app/case-study.pdf*

Wrote a full case study covering the frosted-panel design system, the iteration log, and the bugs that forced each refactor. Link in comments. #design #frontend #nextjs #apple #stripe #fintech

5. What NOT to do

- **Don't gate the live app behind a "Sign in" wall.** Recruiters will not sign in. The app must be browsable anonymously in 5 seconds.
- **Don't embed the case study as images.** Text-selectable PDF is indexable; images are not. SEO matters when someone googles "Frosted Panel redesign case study".
- **Don't claim it's an official Stripe / Apple project.** It's an independent redesign study. "Frosted Panel redesign of Stripe Dashboard" reads as "I learned from both"; "Stripe clone" reads as "I copied it".
- **Don't fake the glass with semi-transparent white.** Real backdrop-filter with blur + saturate is non-negotiable. Anything else looks foggy, not glassy. The photonic edge specular highlight is the signature.
- **Don't skip the brand-tinted wallpaper.** Glass on flat white = foggy. Glass on a Stripe-indigo-tinted mesh = real photonic refraction. The wallpaper is not decoration — it's the foundation.
- **Don't animate chart draw-ins.** Data appears, doesn't perform. Adding draw-in animations to charts feels like a magic trick. Real data dashboards inform.

Case study for the Stripe Frosted Panel redesign project. All artwork is fictional — customers, transactions, balances, and payouts were authored for this case study. User avatars are seeded placeholders from picsum.photos. Stripe is a trademark of Stripe, Inc.; Frosted Panel is a design language introduced by Apple Inc. in 2025. This is an independent design study, not affiliated with either company.

END OF CASE STUDY

Money should feel transparent.

The deliverable is not 6 finished views. The deliverable is one Zustand store, one CSS file with four glass utility classes, one Stripe-indigo-tinted wallpaper gradient mesh, and a slide-over pattern that preserves context. Take that to your next role.