

WEB DESIGN · DEVELOPMENT · PAYMENTS

Zyntrig Dispatch

A conversion website for an AI dispatch automation product, with a custom WordPress plugin that takes a \$1 card verification through Stripe.

CLIENT

Motus Labs LLC

LOCATION

The Woodlands, Texas,
USA

INDUSTRY

Logistics software

PLATFORM

WordPress + custom
plugin

LAUNCHED

July 2026

AT A GLANCE

A product site that has to sell, and a checkout that has to be right

Zyntriq Dispatch is an AI-powered dispatch automation product from Motus Labs LLC. The company needed a marketing site that explains the product, collects demo requests, and lets a new customer start by paying a one-time \$1 verification charge by card.

Motus Labs handed us two documents: a full assets package with every page, every line of copy and the locked brand, and a separate build packet for the payment module. Our job was to build both exactly to spec, connect them, and take the site live on the client's own domain without breaking the company email running on it.

12

marketing and
conversion pages

5

legal pages, linked site-
wide

1

custom payment plugin,
built from scratch

3

Stripe webhook events,
each signature-checked

3

screen widths tested on
every page

The challenge

- Copy, page order and brand were all locked by the client. Nothing could be invented or reworded.
- The packet was written for a Next.js stack. The site was going to WordPress, so the payment module had to be rebuilt as a plugin with the same behaviour.
- Real money on a live domain: every payment path had to work, and a fake webhook must never mark anything paid.
- The domain also carried live company mailboxes that could not go down at launch.

What we delivered

- The full marketing site in WordPress, built to the locked design system, with an animated, interactive homepage.
- Motus Payment, a custom WordPress plugin: Stripe Checkout, \$1 verification, verified webhooks, success and failure pages, admin settings and a payments log.
- Consent checkboxes before payment, five legal pages and a plans section the client edits from the admin.
- DNS cutover, HTTPS, real-card testing, a handoff document and a QA sheet.

Services

Front-end build

WordPress development

Custom plugin

Stripe Checkout

Webhooks

Legal and consent flow

DNS and launch

QA and handoff

THE BRIEF

Build it exactly as written

The client's packets were detailed down to button colours and form routing. That made the brief clear, and it also set one hard rule for the whole build.

Every word on the site comes from the client's documents. No added stats, no made-up section titles, no paraphrased claims.

Where content was still pending, such as final plan prices, testimonial names or live Stripe keys, we left a clearly marked placeholder in the code instead of filling the gap with our own words. The client could see exactly what was still open.

One change to the stack

The packet named Next.js, TypeScript, Tailwind and Vercel. We proposed WordPress and the client agreed. Everything else in the packet stayed locked, so the payment module had to behave exactly as specified, only on a different platform.

PACKET (NEXT.JS)	OUR BUILD (WORDPRESS)	RESULT
<code>/api/create-checkout-session</code>	Nonce-protected WordPress AJAX action	Same Stripe Checkout Session, works for guests
<code>/api/webhook</code>	REST route <code>/wp-json/motus-payment/v1/webhook</code>	Same three events, same signature check
React button component	Shortcode button, styled to the locked spec	Drop it on any page
Success and failed routes	Pages created by the plugin on activation	No manual setup
Stripe SDK	Direct calls with <code>wp_remote_post</code>	No Composer, no extra library to maintain

Stayed locked

Brand colours and type, page list and order, all copy, the Pay \$1 button spec, the "Powered by Stripe" badge, form routing and legal wording.

Checked with the client in writing

The stack change, where each pricing CTA should go, and that the plans are display only. The \$1 charge is the only thing Stripe handles.

DESIGN SYSTEM

A locked brand, applied the same way everywhere

The client's brand rules came fixed. We set them up once as shared styles so every page, every button and the payment plugin use the same values.

Colour

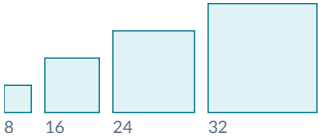
 Deep Navy #0A1A3F Primary. Header, footer, dark bands, button fill.	 Action Cyan #0E7C93 Links, interactive states, button hover.	 Accent Cyan #52C7DC Highlights on dark surfaces only.
---	---	---

Type

Montserrat
Bold for headings, SemiBold for buttons

Inter
Body text, forms and small print

Spacing



8px grid,
8px corner radius

The Pay \$1 button



Powered by Stripe



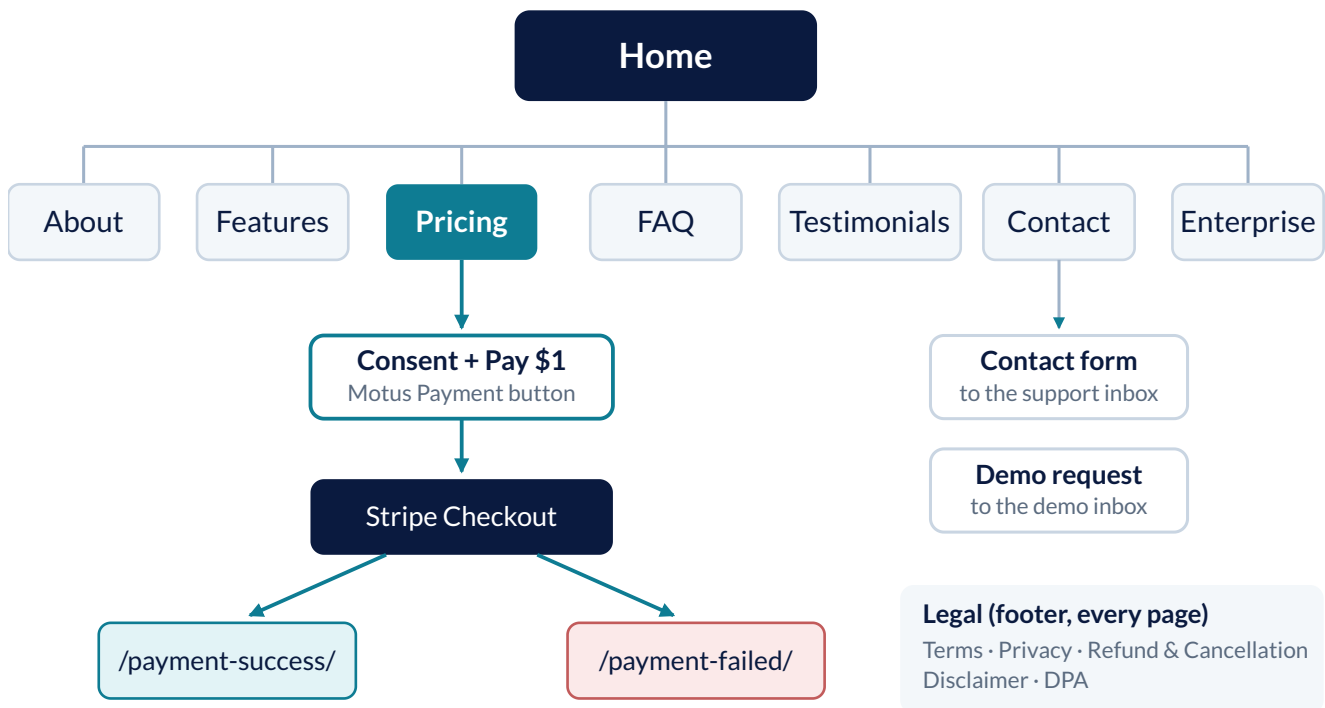
Hover state

The client fixed this button down to the pixel: navy fill, cyan hover, Montserrat SemiBold, 8px radius and the Stripe badge directly below. It is built into the plugin, so it looks the same on any page it is placed on.

SITE STRUCTURE

Every page points to one next step

The navigation follows the client's sitemap in order. The main call to action, Get Started, sends visitors to the pricing page, where the \$1 activation lives.



Navigation

Home, About, Features, Pricing, FAQ, Testimonials and Contact, in the order set by the client's sitemap. A "Get Started" button in the header goes straight to pricing.

Footer

Four columns: brand and email, Product, Company and Legal. The legal links were rolled out to every page so the terms are one click away from the checkout.

Forms that reach the right team

The contact form and the demo request form go to two separate company inboxes, so sales and support each get only their own leads. The address shown on the site stays the product's own support address.

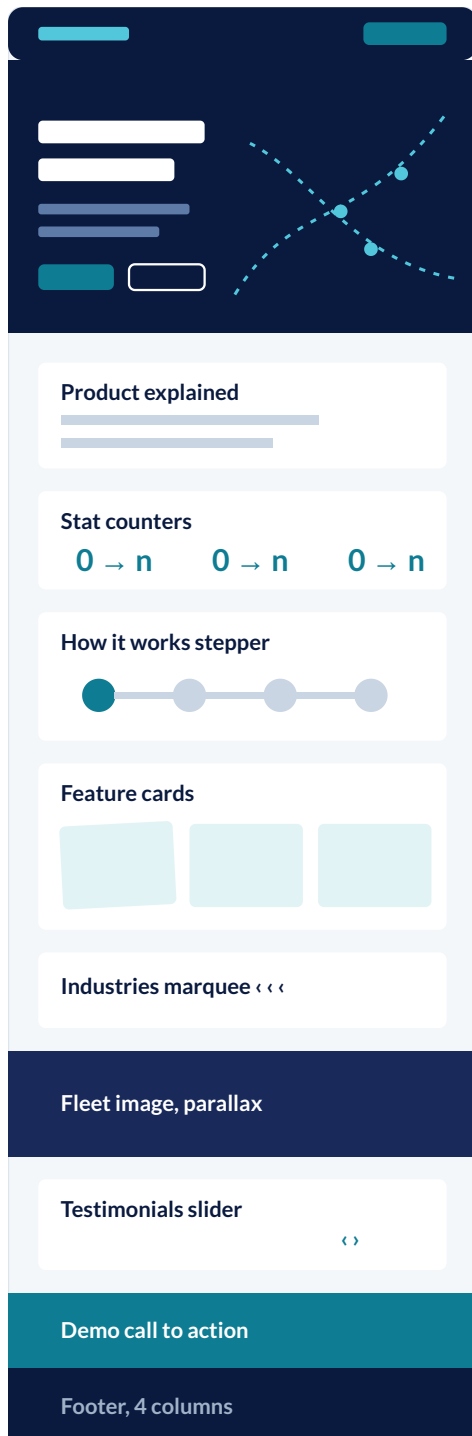
No Stripe on the forms

Request a Demo and Contact Sales are plain forms. The only thing that ever touches Stripe is the \$1 button on the pricing page.

HOMEPAGE

Showing a dispatch product in motion

Dispatch is about things moving from A to B, so the homepage moves too. The motion is there to explain the product, not to decorate it.



Animated dispatch routes

The hero carries an SVG map of routes with small pulse dots travelling along them, the way loads move across a dispatch board. It is drawn in code, so it stays sharp at any size and loads fast.

Stat counters

The numbers count up when they scroll into view, so they get read instead of skipped.

How it works, step by step

An autoplay stepper walks through the product one stage at a time. Visitors can also click any step to jump to it.

Feature cards, marquee and slider

Feature cards tilt slightly in 3D under the cursor and link through to the Features page. The industries the product serves scroll past in a marquee, and testimonials sit in a slider.

Parallax fleet band

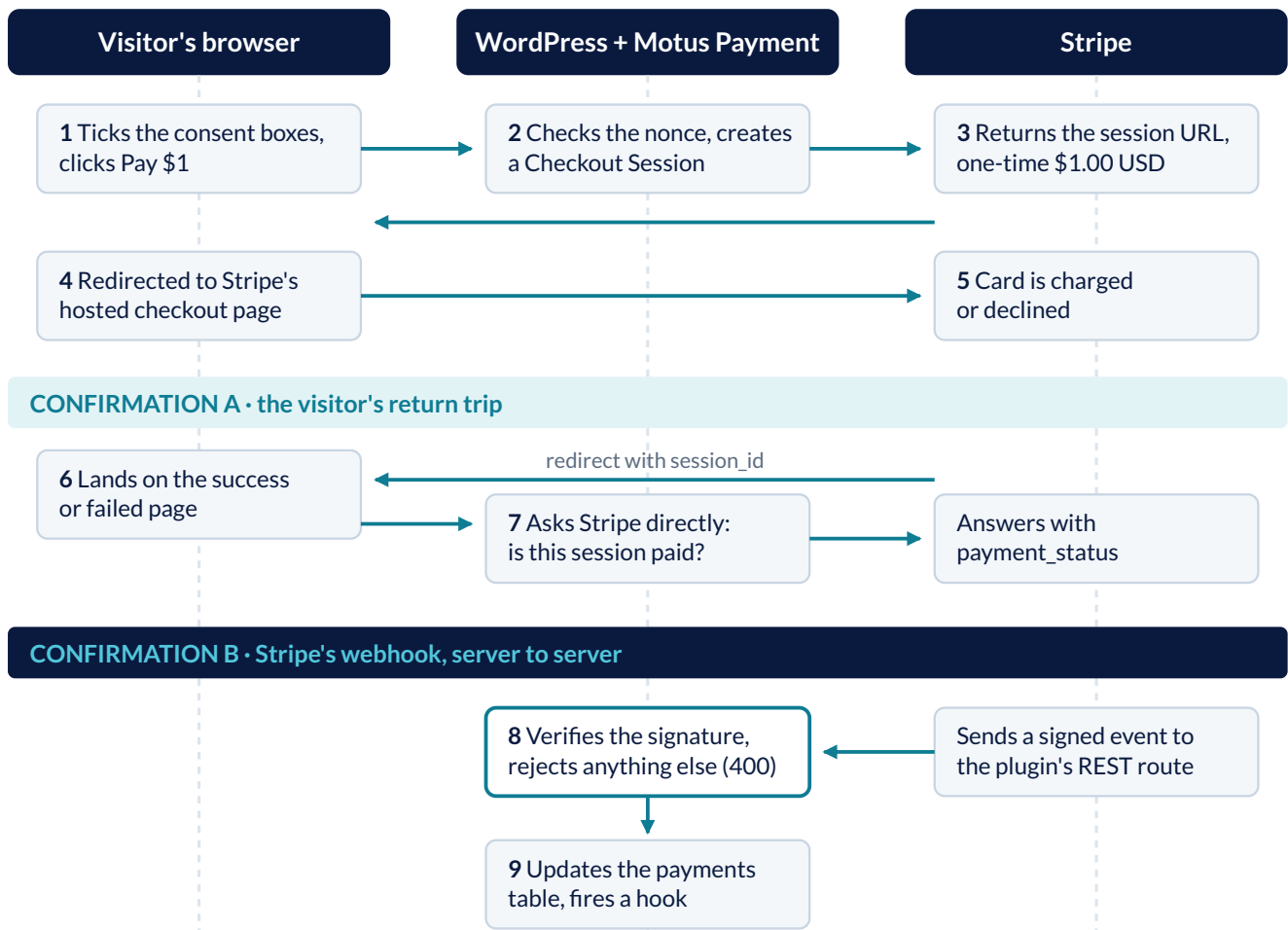
A full-width fleet image moves at a slower speed than the page, which breaks up the long scroll before the demo call to action.

All of this runs on one shared stylesheet and one script used across every page, written in lightweight jQuery. The client's logo files were placed in their set roles: the wordmark in the nav and footer, the emblem in the hero, and the full lockup as a watermark in the call-to-action band.

MOTUS PAYMENT PLUGIN

How a \$1 verification actually works

Motus Payment is a WordPress plugin we wrote from scratch for this site. It runs one job and runs it carefully: a one-time \$1 card charge through Stripe Checkout, confirmed two separate ways.



Why two confirmations?

Webhooks can arrive late. If the visitor gets back before Stripe's event does, the success page checks with Stripe itself, so the customer still sees "payment confirmed" and the record is marked paid. When the webhook lands, it agrees.

The three events handled

```
checkout.session.completed
payment_intent.succeeded
payment_intent.payment_failed
```

INSIDE THE PLUGIN

Small plugin, no shortcuts

It handles real money on a live site, so every part of it was written with failure in mind: forged requests, late webhooks, wrong keys and test data leaking into live mode.

Security

- Every webhook is checked against Stripe's HMAC signature, with a 5-minute window to block replayed events. Anything unverified gets a 400 and changes nothing.
- The checkout request is nonce-protected, so it can't be triggered from another site.
- Live mode runs on a restricted Stripe key, scoped only to what checkout needs, not a full secret key.

Reliability

- Success page double-checks payment with Stripe, so a slow webhook never shows a paying customer an error.
- Separate test and live settings: secret key, price ID and webhook secret per mode, switched with one toggle.
- Charge runs in Stripe's one-time payment mode, matching the \$1 product, not a subscription.

Zero setup on install

On activation the plugin creates its own `wp_motus_payments` table plus the `/payment-success/` and `/payment-failed/` pages, with the right shortcodes already inside. The settings screen shows the exact webhook URL to paste into Stripe.

Light on the site

No Stripe SDK and no Composer. The plugin talks to Stripe with WordPress's own HTTP functions. Its CSS and JS only load on pages that use its shortcodes, and a version constant busts the cache on every update.

Admin, for the client

A "Motus Payment" menu in WordPress with two screens:

- **Settings:** test or live mode, keys per mode, price ID, webhook secret, button label and customer portal link.
- **Payments log:** email, amount, status and the Stripe session and payment IDs for every attempt, so any charge can be traced in seconds.

Built to grow

The packet planned onboarding logic for later. The plugin is ready for it without being edited:

HOOK / SHORTCODE	DOES
<code>mtsp_payment_completed</code>	Fires after a verified payment
<code>mtsp_payment_failed</code>	Fires after a failed payment
<code>[motus_payment_success]</code>	Success page content
<code>[motus_payment_failed]</code>	Failed page with retry
<code>[motus_portal]</code>	Portal button, placeholder until set

CONSENT, LEGAL AND PLANS

Clear terms before anyone pays

A \$1 charge is small, but it is still a card payment, and it leads into a paid service later. The client wanted the customer to agree to the terms first, in so many words.

Activate your account

☐ Consent checkbox one (client wording)

☐ Consent checkbox two (client wording)

Pay \$1

Powered by Stripe

Diagram of the gate. The button only works once both boxes are ticked.

The consent gate

Two checkboxes sit above the Pay \$1 button, using the exact wording from the client's legal document. Until both are ticked, checkout won't start. The agreement happens on the site, before the visitor ever reaches Stripe.

Five legal pages

- Terms of Service
- Privacy Policy
- Refund and Cancellation Policy
- Disclaimer
- Data Processing Agreement, also linked from the Enterprise page

All five were built word for word from the client's document and linked from the footer on every page.

Plans the client can edit

The pricing plans are display cards with their own calls to action. Prices and limits were still being finalised, so every figure is editable from the WordPress admin. No code change needed when the numbers change.



Plans are display only. The packet is clear that Motus Payment is not a subscription engine, so no recurring billing was built behind them.

LAUNCH AND QA

Going live without breaking email

The client's domain already ran the company's live mailboxes. A careless DNS change at launch would have cut off their email on the same day the site went up.

DNS cutover

- Before touching anything, we checked the live MX, SPF and A records and confirmed mail was healthy.
- Only the A records for the root domain and www were changed. Nameservers stayed where they were, so the mail records were never at risk.
- Checked again after the switch: site live, email still flowing.
- No DMARC record existed. We didn't add one on launch day, but flagged it in the handoff as the next step to harden email.

HTTPS

- SSL certificates for both the root and www were confirmed valid and set to auto-renew.
- HTTPS was forced from the hosting panel instead of hand-editing server config on a live payments site.
- Mixed-content warnings from old http:// asset links were cleaned up, so every page loads fully secure.

Payment paths tested

SCENARIO	WHAT WE CHECKED	RESULT
Successful payment	Real card, \$1 charged, success page confirms, record marked paid	Pass
Declined card	Visitor lands on the failed page with a retry button, nothing marked paid	Pass
Refund	Refunded charge reflected correctly, no broken state on the site	Pass
Late webhook	Success page still confirms by asking Stripe directly	Pass
Unsigned or forged webhook	Rejected with a 400, database untouched	Pass
Consent boxes unticked	Checkout does not start	Pass

Every page, three widths

All pages checked end to end, including the legal pages, at 375px, 768px and 1440px, with screenshots of each kept for the record.

Handed over properly

A branded handoff document covering setup, Stripe settings and DNS, plus a QA sheet listing every check and its result.

THE OUTCOME

What Motus Labs got

- A product site that matches their brand packet to the letter, with no invented copy.
- A homepage that shows what dispatch automation looks like instead of only describing it.
- A clear route from any page to a paid, verified customer.
- A payment plugin they own, with test and live modes, a full payments log and hooks for onboarding later.
- Consent and legal pages in place before the first real charge.
- A launch with no email downtime, and documents their next developer can pick up from.

STACK

WordPress · custom PHP plugin · WordPress REST API · Stripe Checkout and webhooks · jQuery · SVG animation · Montserrat and Inter

Building something that takes payments?

We build WordPress, Shopify, Laravel and Next.js sites for clients across Europe, Australia and the US, including custom plugins for checkout, enrolment and membership.

EMAIL	sales@rishfotech.com
PHONE	+91 98747 00540
BOOK A CALL	calendly.com/rishav-rishfotech/30min
WEB	rishfotech.com