Payment Integration Design Document

Logan Health — loganhealth.com

Date: 03 February 2026

Version: 1.0 — Draft

Status: For Implementation via Claude Code

JotForm + Square Payments | Cloudflare Workers | Cloudflare KV

CONTENTS

SECTION 01

Overview

Logan Health operates a static website (loganhealth.com) where users complete a questionnaire to determine eligibility for GLP medication. Eligible users must pay before optionally booking a consultation.

This document describes the architecture for adding payment processing to the existing static site. The solution supports both one-off payments and recurring subscription plans using JotForm with Square payments, secured by Cloudflare Workers and Cloudflare KV for token-based booking access control.

Key Requirements

- Accept one-off payments for single medication purchases
- Accept recurring subscription payments for medication plans
- Verify payment server-side before allowing consultation booking
- Keep infrastructure completely separate from existing projects
- Stay within free tier limits for all services
- Produce a solution suitable for implementation via Claude Code

SECTION 02

User Flow

The complete user journey from questionnaire to consultation booking follows this sequence:

1. User visits loganhealth.com and completes the eligibility questionnaire
2. User qualifies for GLP medication
3. User is presented with a payment choice: one-off payment or monthly subscription plan
4. One-off path: user is directed to JotForm A (Square single charge)
5. Subscription path: user is directed to JotForm B (Square recurring billing)
6. On successful payment, JotForm sends a webhook to a Cloudflare Worker
7. The Worker verifies the payment with Square's API
8. The Worker generates a unique, single-use booking token and stores it in Cloudflare KV with a 72-hour TTL
9. The Worker sends the user an email with a tokenised booking link
10. User clicks the link; the static booking page calls the Worker to validate the token
11. If valid: the booking/scheduling UI loads. If invalid or expired: an error message is displayed
12. After booking, the token is marked as used (single-use enforcement)

Note: Both payment paths converge at step 6. From the Worker's perspective, the flow is identical regardless of payment type — the only difference is which JotForm triggered the webhook.

SECTION 03

Architecture Components

3.1 Static Website (Existing)

The existing static site at loganhealth.com hosts the questionnaire and all front-end pages. Two new pages or sections are required:

Payment Choice Page

Displayed after the user qualifies via the questionnaire. Contains two clear options (cards or buttons) — one for the one-off payment and one for the monthly subscription. Each option links to or embeds the corresponding JotForm.

Booking Page (/book)

A static HTML page that reads a `token` query parameter on load and calls the token validation Worker endpoint. If the token is valid, the booking/scheduling UI is rendered. If invalid or expired, a friendly error message is shown.

3.2 JotForm Configuration

Two separate JotForms are required. JotForm's Square integration is configured at the form level — either 'Sell Products' for one-off payments or 'Sell Subscriptions' for recurring. These cannot be mixed within a single form.

	JotForm A — One-Off	JotForm B — Subscription
Payment Type	Sell Products (single charge)	Sell Subscriptions (recurring)
Square Mode	Single transaction	Card on file, auto-billing
Collects	Patient name, email, payment	Patient name, email, payment
Billing Cycle	N/A	Configured in JotForm (e.g. monthly)
On Submission	Webhook to Cloudflare Worker	Webhook to Cloudflare Worker

Both forms must send a webhook POST to the same Cloudflare Worker endpoint on successful submission. The webhook payload must include the submitter's email and a field or metadata value indicating which payment type was used.

3.3 Cloudflare Workers

All serverless logic lives in a single Cloudflare Workers project with three routes. This runs on a separate Cloudflare account, completely isolated from any other infrastructure (including EveryVoyage / Hetzner).

Worker 1 — Webhook Receiver

```
POST /api/webhook
```

Triggered by JotForm on successful payment submission. Responsibilities:

- Receive and parse the JotForm webhook payload
- Extract customer email, payment details, and payment type
- Verify the payment was successful via Square's API (do not trust the webhook blindly)
- Generate a cryptographically random booking token (32-byte hex string using `crypto.getRandomValues()`)
- Store the token in Cloudflare KV with a 72-hour TTL
- Send a transactional email to the customer with the tokenised booking link
- Return 200 OK to JotForm

Worker 2 — Token Validator

```
GET /api/validate-token?token={token}
```

Called by the static booking page on load. Responsibilities:

- Look up the token in Cloudflare KV
- If found and not yet used: return `{ valid: true, email: '...', paymentType: '...' }`
- If not found, expired, or already used: return `{ valid: false }`

Worker 3 — Mark Token Used

```
POST /api/mark-used?token={token}
```

Called after a booking is successfully confirmed. Deletes the token from KV or updates its value to mark it as consumed. This enforces single-use tokens.

3.4 Cloudflare KV Store

A single KV namespace (e.g. `BOOKING_TOKENS`) stores all active booking tokens.

Key format:

The token string itself, e.g. `a8f3e2b1c9d4e5f6...`

Value format (JSON):

```
{ "email": "patient@example.com", "paymentType": "one-off | subscription",
  "paymentRef": "sq_txn_abc123", "createdAt": "2026-02-03T14:30:00Z", "used": false
}
```

TTL: 72 hours (259,200 seconds)

Set at write time. KV handles expiry automatically — no background cleanup jobs are needed. Expired keys simply disappear.

3.5 Email Delivery

The webhook receiver Worker sends a transactional email containing the booking link. Recommended providers with free tiers:

Provider	Free Tier	Integration
Resend	100 emails/day	Simple REST API, easy from Workers
SendGrid	100 emails/day	REST API, well-documented
Mailgun	100 emails/day (trial)	REST API

The email should include: a greeting using the patient's name, confirmation that payment was received, the tokenised booking link, and a note that the link expires in 72 hours.

SECTION 04

Security Considerations

4.1 Token Security

- Tokens must be cryptographically random — use `crypto.getRandomValues()` in the Worker
- Tokens are single-use — once a booking is made, the token is invalidated
- Tokens auto-expire after 72 hours via KV TTL
- The booking page URL is not linked from anywhere on the public site
- The booking page must include a `noindex` meta tag
- The booking page path must be excluded from `sitemap.xml`

4.2 Webhook Verification

- Verify JotForm webhook authenticity (JotForm sends a signature header that can be validated)
- Cross-verify the payment with Square's API — do not rely solely on the webhook payload
- Rate limit the webhook endpoint to prevent abuse

4.3 Token Validation Endpoint

- Rate limit the validation endpoint to prevent brute-force token guessing
- Return minimal information in error responses (just `valid: false`, no details about why)
- Add CORS headers to restrict calls to the loganhealth.com origin only

SECTION 05

Free Tier Capacity Analysis

All infrastructure runs within Cloudflare's free tier. The effective ceiling is determined by the most constrained resource.

Resource	Free Limit	Per Transaction	Max Txns/Day
Worker Requests	100,000 / day	~3–4	~25,000
KV Reads	100,000 / day	~1–2	~50,000
KV Writes (bottleneck)	1,000 / day	~1–2	~500
Email (Resend)	100 / day	1	100

Effective ceiling: approximately 100 paid bookings per day, limited by the email provider's free tier. For a clinic doing 10–20 consultations per day, this is significantly more than required. If volume grows,

the email provider is the first component to upgrade.

> **Note:** KV writes can be optimised to 1 per transaction by relying on TTL for expiry rather than explicitly marking tokens as used. This increases the KV ceiling to ~1,000/day, but the email limit remains the binding constraint.

SECTION 06

Project Structure

The following project structure is recommended for the Cloudflare Workers implementation:

`loganhealth-payments/`	
`  wrangler.toml`	Cloudflare Workers configuration
`  src/`	
`    index.js`	Worker entry point and route handler
`    webhook.js`	POST /api/webhook — JotForm webhook receiver
`    validate.js`	GET /api/validate-token — token validation
`    mark-used.js`	POST /api/mark-used — mark token consumed
`    email.js`	Email sending utility (Resend / SendGrid)
`    square.js`	Square API payment verification utility
`  static-site-changes/`	
`    payment-choice.html`	New page: payment option selection
`    book.html`	New page: booking page with token validation
`  README.md`	

SECTION 07

Configuration and Secrets

The following must be configured in the Cloudflare Workers environment:

KV Namespace Binding

- `BOOKING_TOKENS` — bound to the KV namespace for token storage

Environment Variables / Secrets

Variable	Purpose
SQUARE_ACCESS_TOKEN	Square API access token for payment verification
SQUARE_ENVIRONMENT	‘production’ or ‘sandbox’
EMAIL_API_KEY	API key for the chosen email provider (e.g. Resend)

Variable	Purpose
EMAIL_FROM	Sender email address (e.g. noreply@loganhealth.com)
JOTFORM_API_KEY	For webhook signature verification
BOOKING_TOKEN_TTL	Token expiry in seconds (default: 259200 = 72 hours)
SITE_URL	Base URL for booking links (e.g. https://loganhealth.com)

SECTION 08

JotForm Setup Checklist

Complete the following steps to configure both JotForms:

1. Create a Square merchant account and connect it to JotForm
2. Create JotForm A (one-off): add patient name and email fields, configure Square as ‘Sell Products’, set the one-off price
3. Create JotForm B (subscription): add patient name and email fields, configure Square as ‘Sell Subscriptions’, set the recurring price and billing frequency (e.g. monthly)
4. On both forms, configure a webhook integration pointing to https://{worker-domain}/api/webhook
5. Add a hidden field or form identifier to each form so the webhook payload distinguishes which form was submitted (e.g. paymentType = ‘one-off’ or ‘subscription’)
6. Test both forms in Square sandbox mode before going live
7. Embed or link both forms from the payment choice page on the static site

SECTION 09

Static Site Changes

Payment Choice Page

After the user qualifies via the questionnaire, they are shown a page with two clear options. Each option should display the price, a brief description of what is included, and a call-to-action button that either opens the corresponding JotForm in an iframe/modal or redirects to the hosted JotForm URL.

Booking Page (/book)

A static HTML page with the following behaviour:

- On page load, read the `token` query parameter from the URL
- Call `GET https://{worker-domain}/api/validate-token?token={token}`
- If valid: display the booking/scheduling interface (e.g. embedded Calendly, Square Appointments, or a custom scheduler)
- If invalid or expired: display a friendly error message — ‘This booking link has expired or has already been used. Please contact us if you need assistance.’
- After a booking is confirmed, call `POST https://{worker-domain}/api/mark-used?token={token}` to invalidate the token

SECTION 10

Testing Plan

Sandbox Testing

Configure Square in sandbox mode. Submit test payments through both JotForms. Verify webhooks fire correctly and tokens are created in KV with the expected TTL and metadata.

Token Flow — Happy Path

Confirm the emailed booking link works, loads the booking page, and validates successfully. Verify the booking UI appears and the user can complete a booking.

Token Expiry

Create a token with a short TTL (e.g. 60 seconds) and confirm it is rejected after expiry.

Single-Use Enforcement

Use a valid token to book, then attempt to use the same token again. Confirm it is rejected on the second attempt.

Direct URL Access

Navigate to /book without a token, with an empty token, and with a random invalid token. Confirm all three show the error state, not the booking UI.

Email Delivery

Verify emails arrive promptly with correct booking links. Test with multiple email providers (Gmail, Outlook, etc.).

Production Cutover

Switch Square to production mode, update all API keys and secrets, and test with a real low-value transaction end-to-end.

SECTION 11

Future Enhancements (Out of Scope)

The following are explicitly out of scope for the initial implementation but noted for potential future work:

- Admin dashboard to view payment and booking status
- Automated reminder emails for unused booking tokens approaching expiry
- Direct Square Subscriptions API integration (bypassing JotForm) for more granular control
- Payment receipt / confirmation page on the static site (currently handled by JotForm’s thank-you page)
- Cancellation and refund self-service flow

- Multiple consultation types with different pricing

SECTION 12

Decision Log

Key architectural decisions and their rationale:

Decision	Rationale
Two JotForms instead of one	JotForm configures payment type (one-off vs subscription) at form level — cannot mix both payment types within a single form
Cloudflare Workers + KV	Free tier comfortably handles 100+ bookings/day, zero infrastructure to manage, completely isolated from other projects
Token-based booking access	Server-verified proof of payment before booking — prevents unauthorised access to consultation scheduling
72-hour token expiry	Gives patients reasonable time to book without tokens lingering indefinitely
Single-use tokens	Prevents link sharing — one payment grants one booking opportunity
Separate Cloudflare account	Complete isolation from EveryVoyage infrastructure and Hetzner server
JotForm handles Square integration	Avoids building Square Web Payments SDK integration manually; PCI compliance handled entirely by JotForm and Square
Resend for email delivery	Generous free tier (100/day), simple REST API callable from Cloudflare Workers, reliable deliverability