Scented Mirage Order + Tracking System

Category: Web forms and client-facing pages

Problem: Manual order logging, invoicing, and payment tracking slowed daily operations.

What I Built: Built a website with Google Sheets backend handling product catalog, order capture, invoice emails, and tracking timeline.

Tools Used: HTML, CSS, JavaScript, Google Sheets, Google Apps Script

---

Output: Customer order form, invoice email, payment tracking system

---

Scented Mirage Order + Tracking System

Website + Google Sheets + Apps Script + Email

Quick backstory

The owner started with nothing beyond the product idea. They needed a simple catalog and a way to accept orders fast, without paying for a full ecommerce platform or managing everything in chat. So I built a lightweight front end website and a Google Sheets based back end that handles orders, invoicing, payment references, and customer tracking.

The Problem

A small shop usually ends up doing this manually:

- Answer product questions one by one
- Copy customer details from messages into a sheet
- Compute totals and shipping by hand
- Create and send invoices manually
- Track GCash references through screenshots
- Reply to “Where’s my order?” messages all day
- Maintain product lists by editing the website code

It works, but it is slow and easy to mess up, especially once orders start coming in daily.

The System I Built

Instead of using a full ecommerce platform, I built a simple system using tools that are easy to maintain.

Step 1: Customer facing website pages

Built a branded website from scratch with:

- Product discovery page with filters and a shopping bag
- Order form with auto totals and shipping
- Payment reference submission page
- Thank you page
- Track and trace timeline page
- Contact and FAQs page with a contact form that sends straight to email

Step 2: Product catalog powered by Google Sheets
To avoid hardcoding products in the website:

- Product details live in a “Fragrance DB” sheet
- Filter options live in a “Filters” sheet
- Shipping fee lookup lives in a “J&T SF” sheet
- Apps Script publishes these as JSON endpoints so updates are as simple as editing a row in Sheets

Step 3: Order capture and logging
When a customer confirms an order:

- The website generates an order number and order date
- The order is submitted to an Apps Script web app
- The script logs the full order and customer details into the “Order Form&Client Info” sheet automatically

Step 4: Admin order tracker for invoicing
To make admin work clean and repeatable:

- New orders are imported into a “To Invoice” sheet
- Duplicate order numbers are automatically skipped
- This becomes the working sheet for invoicing and fulfillment updates

Step 5: One click invoice email
When the admin ticks “Send Invoice”:

- A branded HTML invoice email is sent to the customer automatically
- Includes item breakdown, totals, and a “Proceed to Payment” button
- The button brings the customer to the payment reference page with the order number already included in the link

Step 6: Payment reference capture
After paying via GCash, the customer:

- Submits their GCash reference number

- Apps Script updates the matching order row in the “To Invoice” sheet
- Customer is redirected to a thank you page

Step 7: Customer self serve tracking timeline
Customers can track without messaging the seller:

- They enter their order number on the Track and Trace page
- The page calls an Apps Script endpoint
- The endpoint reads the admin sheet fields and returns a timeline of statuses (payment verification, packed, handed to courier, out for delivery with tracking number, delivered)

The Tools Used

- HTML, CSS, JavaScript
- jQuery + Select2 (filter UI)
- Google Sheets (database and tracker)
- Google Apps Script Web Apps (APIs for orders, catalog, shipping fees, tracking)
- Gmail sending via Apps Script (invoice email)
- EmailJS (contact form email delivery)
- GitHub Pages (static hosting)

The Measurable Result
Reasonable estimates for a small shop doing 10 orders per day:

Before automation:

- 15 to 30 minutes per order for logging details, computing totals, preparing invoice, tracking payment proof
- 2.5 to 5 hours daily admin time

After automation:

- Order logging becomes automatic
- Invoice sending becomes a checkbox action

- Payment reference becomes a structured field instead of chat screenshots
- Tracking questions reduce because customers can self check

Expected outcome:

- 30 to 60 minutes daily admin time for review and exceptions
- About 70 to 85 percent reduction in repetitive admin effort

Why This Matters Beyond This Store
This same build style can be reused for:

- Client intake forms that feed a database
- Lead capture with automatic email replies
- Request tracking portals (service requests, bookings, applications)
- Admin dashboards for approvals and follow ups
- Automated reminders (payments, renewals, missed steps)
- Simple weekly or monthly reports and dashboards