⚠️ **Loyalty Program** — ZaarCash 💎, native to the platform (no separate incentive engine). Every rate is admin-configurable.

🛡️ Admin Guide — Managing the Loyalty Platform

For SwiftSoftware and ZaarHub administrators managing the Program backend.

⚡ Walkthrough — do this, in this order

From sign-in to a claim approved and the public pages matching your settings.

1. **Sign in.** Open `https://zaarhub.com/admin-panel.html` and sign in with your administrator credentials. This is the **only** admin console — older links such as `/admin-dashboard.html` or `/admin-login.html` now redirect straight back here.

2. **Read the numbers first.** Open the **Admin Dashboard** and look at the platform totals before touching anything — it tells you whether today is normal.

3. **Clear the claim queue.** Open the **Business Claim & Verification Flow**, check the claimant's evidence, then approve or reject. Approval is what gives a business its portal.

4. **Manage a tenant.** Use **Tenant Management** to inspect or adjust a directory/tenant.

5. **Watch the network.** Review **Fraud Monitoring** for suspicious earn and redeem patterns, and **Audit & Logging** for who changed what.

6. **Configure the public face.** Set the **Notifications Ticker**, **Editor's Picks**, **Verification Badges** and **Category Management** — these decide what every visitor sees.

7. **Own the loyalty numbers.** The earn rate, the redemption cap (10% of a bill), the minimum balance (100 ZaarCash) and free-item treatment are programme settings you control. Every user-facing guide quotes these numbers, so change them here and keep the guides in step.

8. **See the result.** Claims move, the ticker reflects real activity, and every public page matches the settings you just made.

✅ **You're done when:** you have cleared at least one claim, and the ticker, Editor’s Picks and loyalty settings all match what the public pages show.

🏪 1. Business Claim & Verification Flow

When a business owner claims a listing on ZaarHub, the system follows this flow:

Step 1 — Owner Submits Claim

- Owner goes to **/claim-business/** and searches for their business in the directory
- They must **find their business** in search results before proceeding — no duplicate listings can be created
- Step 2: they provide their name, email, phone, and **website URL** (required, validated format)

Step 2 — Auto-Approval (Instant)

The system checks two things for automatic approval:

- **Email-to-website match** — The domain in the owner's email (e.g. `@mybusiness.com`) is checked against the submitted website domain (e.g. `mybusiness.com`). If they match → **auto-approved**
- **Direct email match** — If the owner's email exactly matches the business's `email` field in the directory → **auto-approved**
- **Submitted website is saved** — If the business listing had no website before, the one the claimant submits is saved to the directory automatically

When auto-approved:

- A **visitor account** is created with `business_type = 'merchant'` and a random temp password
- A **business verification record** is created with `method = 'auto_domain'`
- The claimant sees their temp password on screen and can log in immediately at **/portal**
- Their JWT role is set to `company_admin` — they see the business dashboard

> ✅ **Auto-approval example:** Owner with email `john@bobsauto.com` claims a business and submits website `https://bobsauto.com`. The domain `bobsauto.com` matches → instant approval.

Step 3 — Manual Review (Admin Panel)

If auto-approval fails (email domain doesn't match the website), the claim goes to the admin panel for manual review:

1. Go to **https://zaarhub.com/admin/requests.html**
2. Unlock with the **admin key**
3. Click the **Claims** tab → review pending claims
4. Verify the claimant's identity by checking email, phone, or website

5. Click **Approve** or **Reject**

When manually approved via admin panel:

- The Node.js API creates a `visitor_accounts` row with `business_type = 'merchant'`
- A temp password is generated and sent via email (requires Mailgun configured)
- The approval syncs to CoreSwift CRM

> ⚠️ **No website = cannot claim:** If a claimant submits without a website URL, the frontend blocks the submission. If a business listing has no website on file and no website is submitted, the claim goes to manual review with a "no website" flag.

What Business Owners See After Approval

- Login at **https://zaarhub.com/portal** with their email and temp password
- **Business portal** at **/business-portal.html** with purchase PIN, offers management, scanner, guides
- They can change their password, manage their listing, create loyalty offers, and track performance

📊 2. Admin Dashboard

The admin panel is your central hub for monitoring and managing the entire ZaarHub Loyalty Program.

- **All tenants** — View every merchant, supplier, and participant across the platform
- **All transactions** — Monitor ZaarCash earnings, redemptions, and referral activity in real-time
- **Flagged activity** — Review transactions that triggered automated fraud detection rules
- **System health** — Check API status, queue lengths, and service availability

> 💡 **Tip:** Pin the dashboard to your browser. It auto-refreshes every 60 seconds so you can spot issues quickly.

👥 3. Tenant Management

As an admin, you have oversight of all tenant accounts. Your responsibilities include:

- **View accounts** — Access merchant, supplier, and participant profiles when needed for support
- **Manage configurations** — Adjust tenant settings when authorized (offer limits, account status)
- **Account recovery** — Help merchants and participants regain access to locked accounts
- **Account lifecycle** — Activate, suspend, or deactivate accounts according to Program rules

> ⚠️ **Important:** Never make configuration changes without documented authorization. All changes should be logged with the requesting party, reason, and approval.

🔍 4. Fraud Monitoring

Fraud detection is one of the most critical admin functions. The system flags suspicious transactions automatically, but you need to investigate.

What to look for:

- **Unusual patterns** — A single customer earning ZaarCash at many different businesses in a short period, or a business seeing unusual redemption spikes
- **PIN misuse** — Multiple verification failures with the same PIN, or verifications occurring at odd hours with no customers present
- **Self-referral loops** — Multiple accounts referring themselves using different emails or phone numbers
- **Inflated purchase amounts** — A business consistently reporting purchases much higher than their average ticket, especially when the customer is a frequent visitor
- **Collusion indicators** — The same customer and business transacting many times in a day with identical amounts



Review process: Open the flagged transaction → review the customer history and business history → check for matching IPs or account creation dates → make a determination: dismiss flag, warn the party, or escalate for suspension.

🚨 **Escalation:** If you confirm fraud, immediately suspend the involved accounts and document everything. Notify your supervisor and the security team within 24 hours.

🎧 5. Support & Escalation

Admins handle a variety of support requests from tenants. Common scenarios:

- **Account recovery** — Verify identity using email, phone, or security questions, then reset access
- **PIN resets** — Generate a new Purchase PIN for merchants who've lost access or suspect compromise
- **ZaarCash corrections** — Manually adjust ZaarCash balances when system errors occur. **Every manual adjustment must include an audit trail:** reason, authorization, timestamp
- **Dispute resolution** — Mediate between participants and merchants when a transaction is disputed. Review system logs to determine the correct outcome
- **Technical support** — Help tenants with portal access, scanner issues, or offer configuration

> 💡 **Tip:** For ZaarCash corrections, follow the policy: under 1,000 ZaarCash = admin-level approval; 1,000+ ZaarCash = supervisor authorization required.

📋 6. Audit & Logging

The ZaarHub Platform maintains a comprehensive audit trail. All admin actions are logged and subject to review.

- **All actions logged** — Every verification, ZaarCash adjustment, account modification, and offer change is recorded with the admin who performed it
- **Manual adjustments documented** — When you make a manual ZaarCash correction or account change, the system records the before/after state, reason, and your admin ID
- **Regular audit reviews** — Audit logs are reviewed on a recurring schedule to ensure compliance with policies
- **Retention** — Logs are retained for a minimum of 3 years per regulatory requirements

> ⚠️ **Important:** Assume everything you do as an admin is auditable and visible. Never perform actions outside official channels or for personal benefit.

🔒 7. Security Best Practices

Admin accounts have elevated privileges. Follow these practices to keep the platform secure:

- **MFA is required** — Multi-factor authentication must be enabled on your admin account. No exceptions
- **No personal account access** — Never use admin tools to access your own personal account or accounts of friends/family. If you need customer support, submit a ticket like anyone else
- **Report breaches immediately** — If you suspect any security incident (compromised credentials, data exposure, unauthorized access), report it to the security team immediately. Do not investigate on your own
- **Least privilege** — Only access the minimum data needed for your current task. Don't browse accounts or transactions out of curiosity
- **Secure your session** — Always log out when leaving your workstation. Don't share your admin session with anyone

> 🚨 **Non-compliance:** Violations of security policies are grounds for immediate revocation of admin access, termination of employment or contract, and possible legal action.

🏗️ 8. City Requests

Visitors can request cities that aren't listed yet:

- **City Requests tab** in sidebar shows all requests ranked by vote count
- **Pending** — gray badge, has "Mark Added" button when you launch that city
- **Added** — green badge, shows processed date
- Useful for prioritizing which new markets to open next

📊 9. Micro-Polls

Create one-question polls for your directory:

- **Polls tab** — create polls with question + up to 8 options + optional end date
- **Auto-close** polls on end date, or close manually anytime
- **Visitors vote once** per poll, see live percentage bars after voting
- Polls appear as a sidebar widget on directory pages

📋 10. Booking Requests (Merchant View)

Business owners manage service bookings from their dashboard:

- **Booking Requests tab** — shows incoming service requests sorted newest first
- Each request shows: visitor name, requested service, date/time, contact info, status badge
- **Confirm** or **Decline** buttons on pending requests
- Visitors also get a My Bookings page to track their request status

👥 11. Referrals & ZaarCash

Admins verify and manage referrals:

- **Referrals tab** — tabbed view: Pending | Verified | Paid
- **Pending** — shows referrer, referee, direction, and action buttons
- **Verify** button — marks referral as paid and grants ZaarCash to the referrer via the native loyalty engine
- **Reject** button — marks as rejected
- Summary cards: Total referrals, Pending count, Total ZaarCash awarded
- Amounts by direction: visitor→visitor: 50, business→business: 200 (2×), business→visitor: 50, visitor→business: 100

🏆 12. Native Loyalty & Clearinghouse Architecture

The ZaarHub Loyalty Program is **native to the Multi-Directory platform**. Loyalty capability is built into every directory — there is no separate IncentiveSwift engine. Each directory admin creates and configures their own loyalty programmes, and members are `visitor_accounts` of the directory. The native **clearinghouse** handles ZaarCash billing, reimbursement, and expiry.

The programme rules the guides state as fact (ZaarHub's current settings):

- **Currency:** ZaarCash 💎 — never "points" or "credits"
- **Earn:** 1 ZaarCash per $1.00 of earnable spend at any participating business
- **Value:** 100 ZaarCash = $1.00 (1 ZaarCash = 1¢)
- **Redemption cap:** at most 10% of a bill may be paid with ZaarCash
- **Minimum:** a member must bank at least 100 ZaarCash before redeeming
- **Free items:** free or fully discounted items earn nothing — only the paid subtotal earns
- **Network-wide:** earned and spent across every ZaarHub city; new cities join the same programme automatically
- **Opt-in:** only businesses that sign up participate — they show the ZaarCash 💎 badge
- **Native:** it is ZaarHub's own programme, not a third-party service

> ⭐ **Every one of those numbers is a per-programme setting, not a constant.** Edit them in the admin panel's ⭐ **Loyalty Programmes** card: the **earn rate** (ZaarCash per $1 spent), the **redemption cap %** (max % of a bill payable), the **minimum balance to redeem**, whether **free items earn nothing**, and the currency's **name and icon** — plus the **per-check-in**, **per-visit** and **per-redemption** rates. A programme may be scoped to one directory or a whole network (ZaarHub runs a single network-wide programme). Another directory could call its currency something else and run different numbers — same code, different settings.

🛡️ Your ZaarCash obligations — the administrator's view

The settings above are more than configuration: because the programme is customer-facing, **the numbers you choose are the promises customers see**. Publish your terms — earning rates, cap, minimum and expiry — and keep them accurate here, in the loyalty guide, and in the panel. Promotional and gift-card style rules vary by state and country; confirm what applies to you before launch, including whether unredeemed balances must be honoured or may expire.

Read the full ZaarCash programme guide for administrators →

> ⚠️ **DISCLAIMER — FOR ADMINISTRATORS**

> You are configuring a customer-facing rewards scheme, so the settings you choose are the promises customers see. Publish your terms (earning rates, cap, minimum, expiry) and keep them accurate in this guide and in the panel.
>
> Promotional and gift-card style rules vary by state and country — confirm what applies to you before launch, including whether unredeemed balances must be honoured or may expire.
>
> The platform stores and applies your settings; it does not decide them, does not reimburse redeemed value, and gives no warranty that a particular configuration is lawful or profitable in your jurisdiction.

Loyalty Programs (per-directory):

- `GET /api/v1/directories/:slug/loyalty/programs` — list programs
- `POST /api/v1/directories/:slug/loyalty/programs` — create a program (name, `points_per_checkin`, `max_checkins_per_day=1`, `points_expire_days=365`, currency name/icon/color, tier/milestone/streak/referral/birthday/social bonuses, is_active)
- `GET /api/v1/directories/:slug/loyalty/programs/:program_id` — program detail
- `PUT /api/v1/directories/:slug/loyalty/programs/:program_id` — update program
- `DELETE /api/v1/directories/:slug/loyalty/programs/:program_id` — delete program
- `POST /api/v1/directories/:slug/loyalty/programs/:program_id/enroll` — enroll a member (visitor)
- `POST /api/v1/directories/:slug/loyalty/programs/:program_id/checkin` — daily check-in (awards `points_per_checkin`; daily cap enforced)
- `GET /api/v1/directories/:slug/loyalty/members/:visitor_account_id` — member profile

Tiers (membership levels):

- `GET/POST /api/v1/directories/:slug/loyalty/programs/:program_id/tiers` — list / create a tier (name, `min_points`, color, perks, `multiplier`)
- `PUT/DELETE /api/v1/directories/:slug/loyalty/programs/:program_id/tiers/:tier_id` — edit a tier (name, `min_points`, color, perks, `multiplier`; omitted fields keep their value) / delete a tier

Rewards (catalog + earn + approval):

- `GET/POST /api/v1/directories/:slug/loyalty/programs/:program_id/rewards` — list / create a reward (`points_required`, `requires_approval`, `reward_tag`, sort_order)

- DELETE `/api/v1/directories/:slug/loyalty/programs/:program_id/rewards/:reward_id` — delete a reward
- POST `/api/v1/directories/:slug/loyalty/programs/:program_id/rewards/earn` — member redeems a reward (deducts ZaarCash, creates an earned record)
- GET `/api/v1/directories/:slug/loyalty/members/:member_id/rewards` — member's earned rewards
- POST `/api/v1/directories/:slug/loyalty/rewards/earned/:earned_id/approve` — admin approves a pending earned reward

Milestones (trigger-based bonuses):

- GET/POST `/api/v1/directories/:slug/loyalty/programs/:program_id/milestones` — list / create a milestone (trigger_type, trigger_value, bonus_points, once_per_member)

Clearinghouse (ZaarCash billing, `/networks/:slug/clear`):

- POST `/networks/:slug/clear/scan` — process a scan (check-in/purchase/redeem) through the clearinghouse
- GET `/networks/:slug/clear/treasury` — treasury summary
- GET `/networks/:slug/clear/ledgers` — business ledgers
- GET `/networks/:slug/clear/caps`, PUT `/networks/:slug/clear/caps` — redemption caps (max % of a bill payable with ZaarCash; 10% default)
- POST `/networks/:slug/clear/expire` — run ZaarCash expiry (365-day rolling)
- GET `/networks/:slug/clear/logs` — clearing logs

Scanner PWA (`/scanner`):

- Camera-based QR scanning using a decoder bundled with the app (250ms interval; works offline)
- Three scan types: Check-in (`points_per_checkin`), Purchase (earn rate × amount), Redeem (deduction with caps)
- Business auth via existing login/register → JWT stored in localStorage
- PWA: standalone mode, manifest, service worker for offline caching
- URLs: `/scanner`, `/scanner-manifest.json`, `/sw-scanner.js`

Business Profile Loyalty Section: Every business detail page (`/biz/:slug/:id`) fetches program & tier data and, if the visitor is logged in, displays their personal tier badge, ZaarCash balance, and scannable QR code. Non-enrolled visitors see a **Join** button that calls POST `.../loyalty/programs/:program_id/enroll`.

Per-Directory Feature Toggle: The `feature_config` JSONB column on directories controls whether loyalty features render. Set `{"loyalty_enabled": true}` to enable the loyalty section on business profiles for a given directory.

> 💡 **Tip:** All loyalty & clearinghouse endpoints require auth (JWT). Test with a valid token from `POST /auth/login` using the `Authorization: Bearer <token>` header. Bad tokens return 401. Loyalty program defaults: `max_checkins_per_day=1`, `points_per_checkin=10`, `points_expire_days=365`.

🔌 13. Integration CTA Buttons

Business owners configure CTA buttons on their listing from the **Integrations tab**:

- **Booking CTA** — toggles CoreSwift booking widget, selects CTA from pre-approved list
- **Campaign CTA** — toggles the MultiDirectory loyalty engine campaign widget, selects CTA
- **SMS CTA** — toggles FunnelSwift SMS funnel, selects CTA
- All CTAs use controlled dropdowns — no free text, keeps directory branding consistent
- Buttons only render on the business detail page when toggled on

🏅 13. Verification Badges & Trust Signals

Businesses in the directory can earn verification badges that display on their listings. These badges help visitors identify trusted businesses at a glance:

- **Verified Owner** — automatically awarded when a business owner completes the claim verification process and is confirmed as the legitimate owner
- **Top Rated** — awarded to businesses that maintain a high average rating (4.5+ stars) across a minimum number of reviews
- **Editor's Pick** — manually awarded by admins to businesses they want to feature (see Section 14 for Editor's Picks management)

Badge display: Badges appear as icons next to the business name on listing cards, search results, and the business detail page. They are controlled by the ZaarHub system and cannot be manually toggled by business owners (except Editor's Pick, which is admin-controlled).

> 💡 **Tip:** Use verification badges strategically. The "Verified Owner" badge is automatic once a claim is approved. Top Rated badges update automatically as new reviews come in. Editor's Pick is your manual curation tool for highlighting standout businesses.

⭐ 14. Editor's Picks Configuration

Editor's Picks are featured businesses hand-selected by admins to appear in a special "Editor's Picks" section on the directory and ZaarHub frontend. They're part of the ZaarHub curation system.

- **Add an Editor's Pick** — Go to the business listing admin panel, find the business, and click **Mark as Editor's Pick**. The business receives the Editor's Pick badge.
- **Display order** — Editor's Picks can be reordered via drag-and-drop in the admin panel. The order controls the display sequence on the frontend.
- **Remove** — Click the Editor's Pick badge or status toggle to remove curation. The badge disappears immediately from the public listing.
- **ZaarHub visibility** — Editor's Picks appear in the ZaarHub carousel alongside featured deals and spotlights, giving them cross-directory exposure.

> ⚠️ **Note:** Editor's Picks are purely a curation tool — they don't change the business's subscription tier or billing. Businesses don't pay for Editor's Pick status; it's earned through admin recognition of quality.

📋 15. B2B Pages Monitoring

Three server-rendered B2B pages are available for your directory network. As an admin, you should monitor their usage and content:

- **RFQ Marketplace** (`/rfq-marketplace`) — Businesses post Request-for-Quote listings. Review RFQ volume and supplier response rates periodically to gauge marketplace health.
- **Co-op Hub** (`/coop-hub`) — Cooperative purchasing and partnership hub. Monitor which directories are participating and flag any inappropriate partnership listings.
- **Lead Exchange** (`/lead-exchange`) — Business lead marketplace. Watch for lead quality and ensure all participants follow directory terms of service.

> 💡 **Tip:** All three B2B pages are server-side rendered for SEO and fast loading. They use the same design system as the rest of the platform. Check each page after major deployments to ensure rendering is working correctly.

🕐 16. Data Freshness & Enrichment

Business listings enriched from third-party sources (Google Places, Yelp, etc.) display a **Data Freshness Timestamp** to visitors. For example: "Data last updated: June 2025 from Google Places."

- **Automatic timestamps** — When a listing is enriched via the Data Company enrichment engine, the enrichment date and source are recorded and displayed on the public listing.
- **Stale data alerts** — Listings that haven't been refreshed in 6+ months are flagged in the admin panel. You can trigger a re-enrichment from the dashboard.
- **Manual refresh** — Use the **Enrich** button on individual business listings to pull fresh data from providers and update the timestamp.
- **Visitor confidence** — Freshness badges increase visitor trust. Listings with "Updated this month" tags see higher engagement than those with outdated data.

> 💡 **Tip:** Set up a recurring n8n workflow to auto-enrich listings older than 90 days. This keeps your directory data fresh without manual effort.

📡 17. Notifications Ticker

The **Notifications Ticker** is a scrolling bar that automatically displays recent platform activity on the directory frontend. Unlike admin-created announcements, this ticker populates automatically from real platform events — new businesses added, deals posted, reviews written, and claims verified.

- **Enable / Disable** — Go to **Settings** → **Notifications Ticker** and toggle the ticker on or off. When disabled, the ticker bar is completely hidden from all frontend pages.
- **Position** — Choose **top** (below the header navigation bar) or **bottom** (above the site footer). Default is top.
- **Scroll speed** — Set to Slow (60s cycle), Normal (30s), or Fast (15s). Slower speeds are more readable; faster speeds show more items.
- **Max items** — Limit how many entries appear in the ticker queue. Default is 5, maximum is 20. Older items rotate out as new events occur.
- **Event filters** — Toggle which event types appear: new businesses, new deals, new reviews, and verified claims. Turn off types you don't want displayed.
- **Customization** — Ticker colors, font size, and animation speed can be overridden via the **Theme** → **CSS Variables** panel. Variables: `--ticker-bg`, `--ticker-text`, `--ticker-accent`, `--ticker-font-size`, `--ticker-speed`.

> 💡 **Tip:** The Notifications Ticker and the Announcement Ticker (Section 30 of the main guide) serve different purposes. The Notifications Ticker is fully automatic — it shows system events with no admin input. Use the Announcement Ticker when you want to publish custom messages like holiday hours or event promotions.

📂 18. Category Management

Categories are organized with categories that contain subcategories (e.g., Fine Dining & Cuisine contains Italian, Chinese; Food & Drink contains Coffee Shop, Pizza). A business can belong to multiple subcategories (up to 3 by default, 5 with admin override).

> ⚠️ **Critical:** Categories must be created **before** searching for or adding businesses. The search engine filter dropdown is populated from existing categories. If no categories exist for a directory, the category filter will be disabled.

Step 1: Create Categories First
Before populating a directory with businesses, create your categories from the **Category Manager** (Admin Panel → Categories). Categories are organized as **categories → subcategories** (not "groups"): e.g., *Fine Dining* contains Italian, Chinese; *Food & Drink* contains Coffee Shop, Pizza. A business can belong to multiple subcategories (up to 3 by default, 5 with admin override).

Using the Populate Businesses tab
Go to **Admin Panel → Populate Businesses**. This is a **manual, admin-driven** flow (not an auto-scraper):

- **Google Places key** — the search uses your Google Places API key. The key is **swappable** (never hardcoded in source); it resolves in order: per-directory `api_config` → `provider_keys` table → `GOOGLE_PLACES_API_KEY` env var. To change it, update either the `provider_keys` row for `google_places` or the env var, then restart.
- **Search by city + category** — enter a city, pick a category/niche, and search. Results come live from Google Places.
- **Review + publish** — select which businesses to import. Duplicates are **hard-blocked by name + address** (a business cannot appear twice in the same directory).

Franchise / big-chain detection (auto-flag)
When you publish a business, the system **auto-flags likely franchises / big chains** using a known-chain name list (~120 national brands) plus Google Places `types` heuristics. You'll see a **manual override** before publishing: confirm *Local* (small/local business) or mark *Franchise*. The community directory is focused on **small, local businesses** — franchises/chains (and suppliers) are **excluded from the public front end** and only appear in the back office where relevant.

Step 2: Search for Businesses
Go to **Admin Panel → Search & Import**. The search engine includes a **Category Filter dropdown** that shows all existing categories. Select a category (or group) to narrow your search. The filter makes results highly targeted from the start — example: select "Coffee Shop" and the search returns coffee shops in the target city.

Step 3: Assign Categories — Individual or Bulk
After search results load, you have two ways to assign categories:

Individual assignment:

- Each search result row has a **category dropdown**. The admin selects one or more categories from the dropdown.
- Click **Save** on that row to update the business's categories immediately.

Bulk assignment:

- Select multiple businesses using the checkboxes in the search results table.
- Click the **Bulk Actions** button at the top of the table.
- Choose **"Assign Categories"** from the bulk actions menu.
- Select one or more target categories in the modal.
- Choose **Replace** (overwrites existing) or **Append** (adds to existing).
- Click **Apply** — all selected businesses update in one operation.

> 💡 **Tip:** Use bulk assign as an auto-categorization tool. Search by one category filter, assign those results, then search by another filter and assign again. Businesses that appear in both searches get both categories — giving them natural multi-category coverage.

Other Assignment Methods

- **CSV import** — Import templates include a category column. Map columns to existing categories during the import wizard.
- **Post-creation editing** — Edit any business from the admin panel and modify its categories via the multi-select. Changes take effect immediately.

Business Owner Self-Service Categories
After claiming their business, owners can manage categories through the Business Portal:

- **Removing a category** — Instant. Owner clicks "Remove" next to any assigned category. No admin approval needed.
- **Adding a category** — Requires admin approval. Owner browses categories and clicks "Request." Creates a pending request visible under **Pending Category Requests**.
- **Maximum categories** — Default cap is **3 per business**. Admins can override to 5 for verified/top-rated businesses.
- **Primary vs. secondary** — The first category is the **primary** and receives full search weight. Additional categories at 75% weight to prevent gaming.

Approving Category Requests
Go to **Pending Category Requests** in the admin sidebar. Each request shows business name, current categories, requested category, and optional rationale. One-click **Approve** or **Deny** with optional denial reason.

Category Cleanup & Merging

- **Rename** — Rename any category; all business associations preserved.
- **Delete** — Only empty categories (0 businesses) can be deleted.
- **Merge** — Select two categories and click "Merge Into." All businesses from source are re-assigned to target, source deleted.
- **Group reassignment** — Move a category between groups by editing its group name.

Current Category Structure (61 subcategories, 15 categories)

- ▶ 🍽️ **Fine Dining & Cuisine** (7 subcategories)
- ▶ 🥤 **Food & Drink** (7 subcategories)
- ▶ 🛍️ **Hyper-Local Retail & Specialty Food** (10 categories)
- ▶ 🔧 **Home & Outdoor** (10 categories)
- ▶ 🏥 **Health & Wellness** (7 categories)
- ▶ 💼 **Professional** (5 categories)
- ▶ 🎭 **Entertainment & Events** (4 categories)
- ▶ ✂️ **Personal Care** (5 categories)
- ▶ 🚗 **Automotive** (3 categories)
- ▶ 🐾 **Pet Services** (1 category)
- ▶ 📚 **Education & Family** (1 category)
- ▶ 📌 **Other** (1 category)

🔑 19. Google API keys — two different keys, two different jobs

Google does not give you one key. It gives you a key **per API**, and the mistake we see most often is using the wrong one — which is what breaks business discovery. There are two, and they are **not** interchangeable.

> 🔒 **Places API (New) — the SERVER key.** This is the key that powers **business search, discovery and the data-enrichment cycle** inside the app. Those calls are made by the **server**, so in the Google Cloud console this key must be restricted by **server IP address** — **never** by website referrer. Google's console may list the API as **Places API** or **Places API (New)**: enable the Places API, and if both names appear, both.

> 🗺️ **Maps JavaScript API — the BROWSER key.** Only needed if a **map is actually drawn on a page in the visitor's browser**. Because that key travels to the browser and is visible in the page source, it **must** be restricted by **website / HTTP referrer** to your own domains (`*.zaarhub.com` plus any custom domains). It **cannot** do the server-side search job — a Maps JavaScript key sitting in the Places field is the single most common cause of "The provided API key is invalid".

> 🚫 **Do not use one "master" key with both APIs enabled.** It is convenient and it is the **least safe** option. A key that has to work from both a browser and a server cannot be tightened to either, so one leak exposes everything — and the browser copy is public by nature. Create **two** keys, each properly restricted, and never a combined one. (David deleted his combined key for exactly this reason — don't add one back.)

Where each key goes — exact screens and fields

- **Places API (New) — server key: Admin Panel → 🔑 Provider API Keys → the "Google Places" row → "Paste API key" → + Add key**. The same key can also be saved from **ZaarHub Admin → Populate tab → "Google Places API Key" → Save Key**. Both write the same stored key.
- **Maps JavaScript API — browser key:** the directory's **SEO settings → "Google Maps API Key"** field (it feeds map embeds). ZaarHub's public maps are drawn with Leaflet/OpenStreetMap and need *no* Google key at all — set this one only if a page actually renders a Google map, and keep it on referrer restriction when you do.

Set it once — not once per city

A key is **scoped**: platform-wide, to one **network**, or to one **directory**. A key saved at **network level is inherited by every city in that network**, so the ZaarHub Google Places key is pasted **once** and all cities use it — do not paste the same key into every city. A standalone directory that is not inside a network keeps its own key. Pick the scope in the **Scope** dropdown in the row (platform-wide / a

network / one directory) before pressing **+ Add key**; the 🔎 **Which key is in force for a directory** panel below it then shows, per provider, whether the winning key is the directory's **own**, **inherited from its network**, or the **platform** key — always masked.

Changing a key — you do not need the old value

- **Just paste the new one** into the same field and save. There is no "confirm the old key first" step.
- Keys are **stored encrypted** on the server, and the panel only ever shows a **masked preview — the first 4 and last 4 characters**. The full value is never sent back to the browser.
- An **empty field never means "no key"**. The row itself states whether a key is stored and **when it was last changed** — read that, not the blank input.

Never paste a key into a chat, a ticket, a document or a code file
Keys are entered **in the panel**, full stop. If a key has ever been shared anywhere — a chat, an email, a doc, a screenshot, in source code — treat it as leaked and **rotate it**: create a new key in Google Cloud, save it in the panel, then delete the old one in Google Cloud.

Wrong key type, or restricted wrong — the symptom and the fix

- **"The provided API key is invalid"** on a search or an enrichment run → the wrong *type* of key, most often a **Maps JavaScript** key pasted into the **Google Places** field, or a key that was deleted/regenerated in Google Cloud. Create a **Places API (New)** key and paste that instead.
- **"… is not authorized to use this service"** / **"API not enabled"** → the Places API is not enabled on that Google Cloud project, or the key's *API restrictions* exclude it. Enable the Places API and add it to the key's allowed APIs.
- **"Requests from referrer … are blocked"** on a server-side search → the key is restricted by **website referrer** instead of the server's IP. Switch the restriction to **IP addresses** and enter the server's IP.
- **"This IP, site or mobile application is not authorized…"** when a map loads in the browser → the browser key is IP-restricted, or your domain is missing from it. Switch it to **HTTP referrers** and add `*.zaarhub.com` plus each custom domain.
- **Search works but the map is blank** → the server key is fine and the **browser** key is missing or wrong. Set the directory's **Google Maps API Key** in SEO settings.

> 📘 **Full detail:** the API Keys Setup Guide — "Which Google key for which job", and the click-by-click Cloud-console steps under Google Places.

✅ 20. What only you can do — the setup checklist

Everything below needs something a machine can't supply: a key from a provider console, a business decision (which gateway, which CRM account), or a DNS record. **None of it is done yet**, and none of it can be done from code. Tick each line as it's finished — every line names the **exact panel screen**, so a VA can work straight down the list.

The panel lives at `https://zaarhub.com/admin-panel.html`. Each screen below is a section on that page, in the order the links appear at the top.

1 ☐ **Rotate the two Google keys.**
Why: the keys were shared in a chat/message, so treat them as **seen** and replace them.
How: Google Cloud console → **Credentials** → create a new key → restrict it (**Places API (New)** by **server IP address**; **Maps JavaScript** by **website / HTTP referrer**) → paste it into the matching labelled field below → then **delete the old key** in Google Cloud. Which key is which is section 19 above.
Where: the server key goes in **Admin Panel** → 🔑 **Provider API Keys** → **the "Google Places" row** → **"Paste API key"** → **+ Add key**; the browser key goes in the directory's **SEO settings** → **"Google Maps API Key"**.
Rule: keys are entered *in the panel* — never in a chat, ticket, document or code file. The panel only ever shows a **masked preview (first 4 + last 4 characters)**, and an **empty box never means "no key"**; the row itself says when a key is stored and when it last changed.

2 ☐ **Pick a payment provider and enter its key.**
Why: no gateway is set, so checkout refuses and payouts stay pending.
Where: Admin Panel → 💳 **Payment Gateways** — connect Stripe and/or PayPal. A gateway **cannot be switched on without its webhook verification value**, and the receiver refuses any event it can't verify, so enter that too.

3 ☐ **Enter email credentials, per directory.**
Why: with none set, **statement emails and reminders are skipped** — the panel reports the service as "not configured".
Where: Admin Panel → ✉️ **Email / SMTP** — choose the directory, fill **From name**, **From email**, **Reply-to** and the transport credentials, then press **Send test email** and confirm it arrives.

4 ☐ **Turn ZaarCash earning ON.**

Why: the **ZaarCash-per-redemption rate is still 0**, so customers earn **nothing** when they redeem.

Where: Admin Panel → ⭐ **Loyalty Programmes** — set the units per **redemption** (and per **check-in** and **visit** if you use them). A rate of **0 switches that source off**, so a 0 here is a deliberate "no earning", not a default.

5 ☐ **Set the clearinghouse settlement settings.**

Where: Admin Panel → 🏦 **Clearinghouse Settlement** — set **Settlement enabled**, **Cycle day (1–28)**, the issue/redeem rates, minimum payout and **ZaarCash expiry (days)**, then **Save settlement settings**.

Note: with no payment provider configured (item 2) a settlement run cannot pay out — it stays **pending**, by design.

6 ☐ **Author the onboarding questionnaire(s), then test the flow.**

Where: Admin Panel → ✍️ **Onboarding Questionnaires** — write the questions for each audience (**customer, supplier, business**), then **Publish** it: a **draft never reaches the public**. Finally walk each onboarding flow end to end yourself.

What it does: completing a questionnaire credits the network currency (**100 units = US$1**) through the native loyalty programme and pushes the answers to CoreSwift; if the CRM is unreachable the answers and the currency still land, and the reason is recorded on the response.

7 ☐ **Decide the canonical CoreSwift account for the ZaarHub network, and connect it.** ⏳ **PENDING — not yet a screen in the panel**

Decision (do this now): from the candidates in the audit, choose the **one** CoreSwift account the ZaarHub network should own its leads in.

Connect (wait for the screen): the server already exposes the connection endpoints (`/api/v1/integrations/coreswift/connection`, **…**`/connection/targets`, **…**`/connection/test`) but there is **no admin screen for them yet** — the "Integration settings →" button on the panel leads to a section that does not exist. Until that screen ships, the connection **cannot be made from the panel**. This is written down here as **pending** rather than sending you to a screen that is not there; an endpoint with no UI control is a defect to close, and this line is the reminder.

8 ☐ **Point DNS for whatever domain mapping is chosen.**
Where: Admin Panel → 🌐 Domains, Subdomains & Subfolders — choose **a subdomain of the network** (e.g. `palm-bay.zaarhub.com`), **its own custom domain**, or **a subfolder of a host** (e.g. `zaarhub.com/palm-bay`), then **+ Add mapping**.
Why it needs you: the code side can be done from the panel, but a mapping only goes live once **DNS resolves**. After creating the record, press **Check live** — a host whose DNS still resolves elsewhere is reported **not live**, never as working.

✅ **How to use this list:** items **1–6** are panel-only work a VA can complete with a provider login. Item **7** needs your decision first (and a screen that doesn't exist yet). Item **8** needs the domain's DNS access. Nothing below item 1 is optional — until each is done, the matching feature is silent, refusing, or earning zero.