IRYSS®

Iryss Tech Build Plan

Table of Contents

1. Internal AI Developer Cockpit

Centralized AI-powered development environment used by all internal engineering teams to accelerate build speed, standardize quality, and train proprietary AI models.

What It Is

A unified developer cockpit integrating the best-in-class AI tools for code generation, prompt workflows, documentation, and testing all embedded into a single interface. Every action is logged and used to **automatically train a proprietary Iryss AI model**, enabling the system to improve over time and become highly optimized for Iryss-specific platform builds.

What It Enables

- Instant productivity boosts from day one using top AI coding tools (e.g. GitHub Copilot, Cody, CodeWhisperer, GPT Assistants)
- Central dev hub with standardized prompts, reusable flows, and versioned knowledge
- **Self-training AI model** that learns from real Iryss workflows to generate high-accuracy, architecture-aware output over time
- Long-term reduction in dev time, bugs, and inconsistencies across services
- Full telemetry on prompt usage, dev pain points, and code refactors used for continuous system optimization

Headline Features

- **Integrated Prompt Interface** for best-in-class AI dev tools (multi-provider)
- **Code Output Logging** and prompt-response versioning
- **Custom Prompt Library** with reusable flows and Iryss-specific logic (e.g. order routing, OMS logic)
- **AI Feedback Loop Engine** feeds all activity back into the internal model trainer

- **Knowledge Memory + Context Indexing** for project docs, past prompts, error resolutions
- **Structured Data Export** for training the Iryss LLM across services
- **Role-Based Access** frontend, backend, devops, AI/ML teams
- Optional: Internal CLI or VSCode plugin

How It Works

All development is funneled through the cockpit using a standard prompt pipeline. Each prompt and code output is logged with metadata (context, user role, service layer, timestamp). This data is continuously used to fine-tune a **custom Iryss AI model** transforming real-world dev activity into a **self-training engineering assistant** tailored to your system.

Why It Matters

This cockpit gives Iryss a foundational advantage: the ability to build faster today while simultaneously training an elite internal AI. Over time, Iryss will run a **high-performance proprietary engineering model** that understands your microservices, APIs, schema, and naming conventions better than any generic tool ever could becoming a competitive edge in speed, precision, and maintainability.

Team & Build Scope

- **Initial Build Team:** 1 AI-focused full-stack engineer
- **Ongoing Training:** Passive (via use) + periodic LLM fine-tuning if needed
- **Timeframe:** ~3–4 weeks for full cockpit MVP with top tools integrated
- **Dependencies:** None this is built first and used to accelerate all 24 other projects

2. Web Scraping & Outreach Engine

What it is

A web scraping and task automation platform that powers Iryss's business development. It continuously scrapes websites, social platforms, and directories to extract leads, products, trends, and competitor data. It automates everything from data extraction to lead enrichment, CRM syncing, and outreach execution all designed to scale brand acquisition, product sourcing, and internal decision-making.

What it enables

- Automated lead generation for brands, influencers, and resellers
- Dynamic product scraping for trend discovery and design inspiration
- Real-time enrichment of contact data and segmentation by channel or region
- Automatic CRM sync and deduplication with lead tagging
- Segmented outreach queues with integrated tracking (email sent, opened, replied, qualified)
- Middleware integration for pushing scraped product catalogs into the Iryss system
- Dashboards for team-wide visibility on lead segments, pipeline status, and trend data

Core Features

- No-code scraping builder with XPath/selector config
- Rotating proxy pool to avoid detection and blocks
- Scraping modules for Shopify, Instagram, Faire, Ankorstore, etc.
- Lead enrichment: email, phone, category, product pricing, region
- CRM sync with tag logic and duplicate prevention
- Automated outreach queue with response logging and status changes

- Manual recheck UI for edge cases or flagged data
- Internal dashboards for viewing trends, segment insights, and sourcing leads
- Export tools for team-specific lists and actions

Build Details

- **Frontend:** React-based interface for scraper setup, lead browsing, outreach status
- **Backend:** Node.js or Python scraping jobs, Redis or BullMQ for queue management
- **Database:** PostgreSQL + Elasticsearch for structured lead storage and query
- **Hosting:** Docker/Kubernetes microservices with cloud autoscaling
- **Integrations:**
    - Iryss CRM or external CRM
    - Middleware (product ingestion sync)
    - SMTP or third-party email automation
    - Optional webhook support for advanced targeting

Team & Timeline

- **Team:** 1 backend (scraper, queue, enrichment), 1 frontend (UI, CRM logic)
- **Build Time:** ~6–8 weeks for MVP covering scraping, enrichment, CRM sync, and outreach

Long-Term Notes

- Powers scalable acquisition of supply (brands), demand (resellers), and promotion (influencers)
- Key pipeline input for marketing, trend analysis, and AI training
- Critical foundation for internal targeting, sourcing, and onboarding workflows

3: Universal Middleware Integration Layer

What It Is
The Universal Middleware acts as the connective tissue between all product sources (brands, resellers, scraping tools) and all destinations (Iryss Marketplace, Storefronts, Shopify, WooCommerce, etc.). It ingests, validates, transforms, and distributes product data across the ecosystem. It is the backbone of data fluidity.

What It Enables

- Seamless onboarding for any brand or product source
- Live product syncing across Iryss and third-party storefronts
- Standardized schema handling for cross-platform consistency
- Central control over enrichment, errors, fallback values, and transformations

Feature Scope

- **Product Ingestion Connectors**: CSV, API, Shopify, WooCommerce, Redicom, Magento, etc.
- **Schema Mapping Studio**: Drag-and-drop field alignment UI with AI fallback suggestions
- **Validation Rules Engine**: Detects missing fields, malformed values, illegal characters, etc.

- **Inline Fix Tool**: Allows users to patch issues inline before sync (e.g. fill price, resize image)
- **Default Field Filler**: Auto-fills common fields (e.g. default shipping country, tags)
- **Live Preview Tool**: Simulate how a product will look on Iryss before syncing
- **Data Versioning & Audit Logs**: View all past versions, track changes per item
- **Product Bundling Support**: Allow grouping of SKUs into shoppable sets
- **Translation Fallbacks**: Use multilingual fallback logic for missing fields (integrated with Localization microservice)
- **Rate Limiting & Queue Throttling**: Protect infrastructure and vendors during high-volume sync

Build Strategy

- **Backend**: Node.js/NestJS with job queue management (e.g. BullMQ)
- **Frontend**: React (admin UI and connector config dashboard)
- **Database**: PostgreSQL for structured product records, Redis for sync queues
- **File Management**: S3 or equivalent for image ingestion and caching
- **API**: REST/GraphQL exposed for sync triggers and store integrations
- **Security**: JWT/role-based access control per vendor/brand

Team Required

- 2 Backend Devs (connectors, queue handling, transformation logic)
- 1 Frontend Dev (UI for schema mapping, live preview, inline fix)
- 1 Full-stack or DevOps (infra, queue tuning, observability setup)

Estimated Time to Build: 6 weeks (MVP), extendable with new connectors

Future Considerations

- Add AI enrichment scoring and smart defaulting
- Add vendor self-service ingestion flows (SaaS UX layer)
- Auto-detect and merge duplicates across data sources
- Webhook triggers to notify third-party systems on sync success/failure

This is one of the most critical infrastructure layers in the Iryss ecosystem and must be engineered for scale, reliability, and flexibility from day one.

4. Customer Acquisition Engine

What it is
A full-stack acquisition and retargeting engine for driving paid growth across Iryss Marketplace and Storefronts. Combines product feed automation, pixel tracking, funnel logic, and AI-powered campaign execution into a unified microservice. This system ensures real-time targeting across Meta, TikTok, Google, and future DSP platforms, optimized for high ROAS and zero manual intervention.

What it enables

- End-to-end paid traffic growth across Marketplace and all Storefronts
- Dynamic product ads based on real user behavior and product feed sync
- Segmented retargeting flows and predictive scoring per SKU and user
- Live campaign syncing across Meta, Google, TikTok with dynamic creatives
- Attribution, budget rules, frequency control, and cross-platform suppression
- Future integration with DSPs like Criteo, AdRoll, and The Trade Desk

Core Features

- **Pixel & Conversions API Layer**
  Fires real-time events (ViewProduct, AddToCart, Purchase) across platforms using Meta Pixel, TikTok Pixel, Google GTM, and server-side CAPI.
- **Product Feed Engine**
  Auto-generates and syncs XML/CSV product feeds (Meta, Google, TikTok). Feeds update daily via cron and match exact product_id logic.
- **Dynamic Campaign Builder**
  Auto-creates DPA campaigns (Meta, Google Performance Max, TikTok Catalog Sales). Personalized ads trigger by user activity and feed matching.
- **Funnel Segmentation Rules**
  Audiences built dynamically based on session depth, recency, cart intent, and cross-channel behavior.
- **Budget & Frequency Control**
  Sets automated bidding rules (e.g. ROAS > 3.0 = scale; < 1.0 = pause). Prevents overexposure and suppresses fatigued users.
- **Multi-Channel Dispatch Layer (DSP-Ready)**
  Central event broker routes all user activity to Meta, Google, TikTok, Klaviyo, and future RTB/DSP endpoints via standardized schema.
- **Consent-Gated Pixel Logic**
  Ensures full GDPR compliance by gating tracking pixels behind user marketing consent status. Integrates with Cookie Manager microservice.
- **Performance Dashboard Connector**
  Syncs ROAS, spend, and conversion data back into Marketplace/Storefront dashboards. Enables live campaign metrics per brand/reseller.

Build Details

- **Frontend**: Admin config tool for feed setup, audience control, pixel monitoring, campaign status
- **Backend**: Node.js or Python microservice with Meta, Google, TikTok API integrations; feed generator; event broker
- **Infra**: Cloud-synced product feeds, containerized event dispatch system, Redis-based suppression logic

Team & Timeline

- **Team**: 1 full-stack engineer (pixel + backend), 1 frontend (admin + feed config), 1 part-time AI/ROAS ops analyst
- **Build Time**: 6–8 weeks to MVP with full Meta/Google/TikTok integration and automated product feed sync

Long-Term Notes

- RTB and DSP integrations will require only a webhook plugin due to broker-layer architecture
- All data is fed to the AI microservices and Data Pooling system for conversion scoring, ad prediction, and funnel optimization

5. Marketplace Orchestrator & Core Architecture (Deity-Based)

What it is
The foundational commerce infrastructure powering the entire Iryss ecosystem built using MACH architecture (Microservices, API-first, Cloud-native, Headless). Based on Deity's orchestration model, this orchestrator routes all commerce logic across B2C and B2B flows, handles multi-role access, and serves as the central rules engine, logic layer, and integration hub for the marketplace.

What it enables

- Unified backend for B2C marketplace, B2B resellers, and vendor operations
- Real-time routing of orders, roles, product updates, and marketplace events
- Multi-role access logic (admin, brand, reseller, influencer, customer)
- Plug-and-play microservices with fully decoupled system layers
- No-code workflow builder for non-technical marketplace ops
- Fast integration of new services, apps, and third-party tools

Core Features

- **Microservices-Based Commerce Logic**
  Each function (checkout, pricing, routing, roles, etc.) runs as an isolated service to ensure modularity, resilience, and scalability.
- **API Gateway + GraphQL Federation**
  All APIs are managed via Kong Gateway, and unified through a GraphQL federated layer to simplify external access and platform integration.
- **Multi-Role Access & Control**
  Role-based logic is handled at the architecture level (brand, vendor, reseller, customer, admin, influencer). Each user sees only relevant flows.
- **No-Code Orchestration Studio**
  Visual interface for configuring product rules, marketplace logic, workflows, and triggers without engineering help (based on Deity Studio).
- **Event-Driven Rule Engine**
  Real-time business logic (e.g. order routing, status updates, stock allocation) triggered via Kafka or RabbitMQ pub/sub model.
- **Real-Time Sync Layer**
  Ensures product and inventory data is synced live between vendor systems, storefronts, and the main marketplace through middleware pipes.
- **Marketplace Governance & Role Routing**
  Admins can pause sellers, manage tiers, and route product visibility based on compliance, stock, returns, or sales behavior.
- **Performance-Optimized Checkout System**
  Includes region-aware pricing, dynamic shipping options, and fulfillment routing based on OMS logic and buyer geography.

Build Details

- **Backend**: Node.js + NestJS architecture, GraphQL Federation, Kafka-based event bus, Kong API gateway
- **Frontend**: React + Next.js admin and routing UI for orchestration studio and core rule logic management
- **Integration**: Native support for CMS, OMS, Payments, Middleware, and AI modules via API orchestration layer

Team & Timeline

- **Team**: 2 backend (orchestration + APIs), 1 frontend (studio + rule builder), 1 systems architect
- **Build Time**: 10–12 weeks for core platform MVP with routing, roles, and no-code studio logic

Long-Term Notes

- All subsequent services (storefronts, TV, apps) plug into this architecture
- Enables Iryss to scale globally while enforcing centralized control and logic
- All future AI decision-making, pricing, or fraud logic integrates via this hub

6. OMS & Shipping Engine

What it is
A centralized order management and logistics orchestration system. This microservice handles the full lifecycle of order routing, fulfillment, return processing, shipping label generation, and multi-location inventory sync across all Iryss channels B2C, B2B, storefronts, and POS. It acts as the logistical brain behind every transaction, ensuring correct, efficient, and rule-compliant fulfillment.

What it enables

- Real-time order routing across vendors, 3PLs, and Iryss warehouses
- Location-aware shipping decisions (e.g. EU vs Egypt fulfillment)
- Dropship and wholesale support with rules-based logic
- Multi-warehouse stock allocation and sync
- Automated shipping label generation, courier selection, and return logic
- Carrier fallback routing and dynamic shipping options at checkout

Core Features

- **Unified Order Routing Engine**
  Routes each order to the correct fulfillment source based on stock availability, region, fulfillment mode (B2B vs B2C), and vendor settings.
- **Multi-Warehouse Inventory Sync**
  Tracks inventory across Iryss hubs, brand-owned locations, and 3PL partners with automated updates and low-stock alerts.
- **Dropship vs Wholesale Logic**
  Routes orders by type: dropship orders go to vendors; wholesale orders use warehouse logic and credit terms.
- **Shipping Label Automation**
  Generates dynamic labels using integrated APIs (Sendcloud, FedEx, Aramex). Handles both outbound and return shipments.
- **Return Management Ruleset**
  Determines correct return location (brand or Iryss) based on who fulfilled the original shipment. Supports return fees, return windows, and auto-restocking.
- **Shipping Zones & Carrier Rules**
  Enables dynamic carrier selection by region, speed, cost, and partner preference. Supports pickup point delivery and customer location logic.
- **Order Event Sync**
  Sends real-time status updates (fulfilled, delayed, returned) to dashboards, customer notifications, and the analytics engine.

Build Details

- **Backend**: Node.js + microservices with rule-based routing, stock logic, and shipping label generation
- **Integrations**:
  - Sendcloud (multi-carrier shipping automation)
  - FedEx, Aramex, DHL (direct shipping APIs)
  - Middleware (product/stock sync)
  - Payments (for shipping cost and refunds)

- **Database**: PostgreSQL + Redis for order events, routing states, and carrier preferences

Team & Timeline

- **Team**: 2 backend (routing logic + integrations), 1 systems designer for fulfillment logic, optional frontend if standalone UI needed
- **Build Time**: 6–8 weeks for MVP (routing + shipping integration + return flow)

Long-Term Notes

- This system is essential for fulfillment automation across Iryss and will feed into analytics, fraud signals, and AI delivery predictions.
- Built to support multi-region expansion from day one, including hybrid models (e.g., brand ships from Egypt; Iryss handles EU returns).
- Enables vendor compliance checks (e.g., delay penalties, shipping cutoff alerts) via order behavior tracking.

7. Payments & Accounting Microservice

What it is
A full-spectrum payments and financial operations engine for the Iryss ecosystem. This microservice manages all transactions, payouts, invoicing, subscriptions, commissions, affiliate earnings, and financial compliance across **Marketplace**, **Storefronts**, and **Iryss TV**. It acts as the central automation layer for all revenue movement and financial reporting.

What it enables

- Real-time checkout processing, vendor payments, and margin reconciliation
- Subscription billing for brands, resellers, and service providers (e.g. €75/month plans)

- Affiliate and influencer earnings calculation for Iryss TV and Storefronts
- Wholesale invoicing, credit tracking, and Net 30/60 payment workflows
- Automated VAT/tax logic across all EU markets and multiple currencies
- Brand co-op ad spend billing (shared performance campaigns)
- Unified financial reporting across all user types and platform roles

Core Features

- **Checkout & Transaction Routing**
    - Handles all B2C and B2B transactions across storefronts and marketplace
    - Routes payments to appropriate merchant (Iryss, vendor, or 3rd-party)
    - Waits for return window before triggering payouts where applicable
- **Vendor & Reseller Payouts**
    - Supports fixed commission, tiered margin, or hybrid payout models
    - Calculates and executes payouts by role (brand, reseller, influencer)
    - Connects to dashboard for downloadable monthly statements and performance history
- **Affiliate & Influencer Earnings**
    - Tracks commissions earned via Iryss TV or Storefront sharing
    - Automates payouts to influencers or brand partners with time-based reporting
    - Supports both fixed per-sale payouts and revenue-share logic
- **Invoice & Credit Management**
    - Auto-generates invoices for B2B wholesale orders (with VAT/EORI data)
    - Tracks outstanding balances and Net 30/60 term compliance
    - Credits store value automatically for unfulfilled or partially refunded items

- **Ad Co-Op & Performance Billing**
    - Calculates brand contributions for shared ad campaigns (Meta/TikTok co-branded)
    - Allocates spend based on clicks, views, or attributed conversions
    - Adds charges to end-of-month statement with full transparency
- **Tax & Currency Handling**
    - Multi-currency checkout and accounting
    - OSS VAT compliance for EU sales, including local tax breakdown per country
    - Export-ready transaction logs for accounting teams and auditors
- **Financial Automations**
    - Sends real-time webhook events to OMS, dashboards, or customer alerts
    - Triggers invoice emails, payout confirmations, overdue alerts
    - API-ready for accounting tools (Xero, QuickBooks, Zoho)

Build Details

- **Backend**: Node.js or Python microservice with currency and logic layers
- **Database**: PostgreSQL for transactions, invoices, payout history
- **External Integrations**:
    - Stripe or Adyen for checkout/payment handling
    - Klarna, PayPal for alternative payment options
    - VAT APIs (e.g., Avalara) for region-specific tax calculations
    - Sendcloud, FedEx for shipping cost reconciliation

Team & Timeline

- **Team**: 1 senior backend (payment logic), 1 integrations/devops, 1 part-time frontend for billing UI
- **Build Time**: ~6–8 weeks MVP with Stripe, payouts, invoice flow, and dashboard sync

Long-Term Notes

- This service becomes the backbone of Iryss's monetization strategy
- Supports future B2B credit risk scoring, dynamic pricing billing, and ad-based performance invoicing
- Must remain modular to plug into new revenue streams (subscriptions, SaaS models, influencer shops)

8. CMS Microservice (Content Management System)

What it is
A unified, headless content management microservice used to power structured and unstructured content across the entire Iryss platform including the Marketplace, Storefronts, and Iryss TV. It handles product stories, blog articles, campaign pages, rich media, and promotional content. Unlike Deity Studio (which handles drag-and-drop UI layout), the CMS focuses on managing content data, editorial workflows, and media assets delivered via API to all frontends.

What it enables

- Centralized content creation, editing, and publishing for all platform interfaces
- API-based content delivery to the Marketplace, Storefronts, Iryss TV, and Mobile Apps
- Flexible content models (blog posts, campaign pages, rich product stories)
- Global asset management (images, videos, metadata)
- Multilingual content fields and region-based publishing control

- Full editorial workflow: draft, review, approve, schedule
- SEO metadata control and integration with the Localization Microservice
- Easy reuse of content blocks across storefronts and campaigns
- Template-based content assembly with dynamic content injection (e.g., by product tag or collection)
- Embeddable shoppable widgets synced from PIM for rich content commerce
- Parameterized smart content blocks (e.g., "Spotlight by tag", "Top sellers per region")
- Personalization-aware delivery rules by user role, region, or behavior
- Creator-linked CMS structures (bios, campaign tagging, influencer stories)
- Scoped asset libraries and content tenancy per brand
- Localized variant previews to test layout with translated content
- AI-assisted content suggestions for SEO fields, intro blurbs, and alt text
- Real-time sync with product and campaign data from the PIM
- Clean decoupling of layout (Deity Studio) vs content (CMS API)

Core Features

- Modular content types (blogs, banners, landing pages, featured stories)
- Global asset manager with CDN integration and media tagging
- Role-based publishing workflows for brand teams and editorial teams
- Real-time preview of content including localization variants and per-role visibility
- Media grouping and tagging by campaign, product category, or creator
- Rich text editor with embedded media and internal links
- Scheduled publishing, version history, and rollback
- API delivery for frontend rendering via GraphQL/REST
- SEO meta fields and Open Graph data per language and region

- Parameterized block logic with automatic rendering based on tag, collection, or creator ID
- Shoppable widgets rendered from live PIM data (SKU, price, availability, region)
- Dynamic hide/show logic for out-of-stock or inactive products
- Personalization delivery logic (location, user segment, logged-in role)
- Full multilingual field support with fallback logic
- Scoped CMS folders and asset permissions by brand or influencer
- Creator model support: campaign participation, linked stories, media blocks
- Preview UI with real-time regional content simulation
- Optional AI prompt engine to assist in writing metadata, alt text, intros, and SEO summaries

How it works with Deity

- All layout and design is handled in Deity Studio, the low-code drag-and-drop builder used by Marketplace and Storefronts
- The CMS plugs into Deity Studio via content APIs feeding content blocks, text, imagery, and asset metadata into the drag-and-drop frontend
- For Storefronts, the same CMS provides shared access to brand blogs, influencer stories, and campaign content with editing permissions based on role
- Smart blocks and widgets from the CMS can be placed visually via Deity Studio while content is managed centrally

Build Details

- Frontend: React-based admin UI for editorial teams and brand operators
- Backend: Node.js or NestJS headless CMS engine (Strapi-style or custom)
- Database: PostgreSQL for structured content + S3/Cloudinary for assets

- Auth & Permissions: Role-based access for internal, brand, and creator users
- APIs: GraphQL/REST endpoints for content delivery and preview
- Integrations:
    - Localization Engine (Project 9)
    - Smart PIM (Project 23)
    - Personalization Engine (Project 14)
    - Deity Studio for layout composition

Team & Timeline

- Team:
    - 2 full-stack developers (CMS backend + admin UI)
    - 1 frontend developer (real-time preview, role-based UX)
    - 1 backend developer (PIM sync, personalization hooks)
    - 1 systems architect (multi-tenant permissions, integration layer)
    - 1 QA engineer (workflow, access control, publishing logic)
    - Optional: 1 AI engineer (content generation prompt engine)
- Timeline:
    - 8–10 weeks for full CMS MVP with personalization, PIM hooks, and preview
    - +2–3 weeks if AI editorial suggestion layer is added
    - Total build: 10–13 weeks for enterprise-grade system

Multi-Stage Rollout

- Stage 1 (Marketplace): CMS used to power homepage, stories, banners, campaigns, SEO, and product stories
- Stage 2 (Storefronts): Brands and influencers gain CMS access to manage blogs, lookbooks, announcements, and shoppable content

- Stage 3 (Iryss TV): CMS content supports show metadata, creator bios, product-linked promo stories, and embedded drops

Final Notes

- This CMS does not handle page layout that is done in Deity Studio
- It is fully decoupled and serves as a central, multi-platform content infrastructure
- All content can be localized via the Localization Engine (Project 9)
- Product references, tags, and SKU metadata are synced from the Smart PIM (Project 23)
- Built for long-term extensibility and high-scale performance across thousands of storefronts, creators, and campaigns
- Aligned with marketplace governance, brand permissions, and modular deployment

9. Localization & Multilingual Engine

What it is
A centralized localization microservice powering all language, regional, and currency logic across the Iryss ecosystem including the Marketplace, Storefronts, Iryss TV, the CMS, the Smart PIM, and Mobile Apps. It ensures a consistent, scalable localization framework for all Iryss content and commerce experiences without duplicating logic across services.

This engine manages translation flows, currency formatting, region-specific routing, fallback behavior, SEO localization, and media substitution. All frontend systems and backend modules plug into this layer to deliver fully localized experiences at scale.

What it enables

- Platform-wide localization across products, content, campaigns, and UI
- Auto-translation of structured data (products, metadata, CMS blocks) with human override interface
- Language-based routing (e.g. /fr/, /de/) and locale auto-detection via IP or browser headers
- Multi-currency pricing display with live or fixed exchange rate logic
- Per-market homepage, campaign, and content variation logic
- SEO-localized pages with canonical URL control and hreflang tag automation
- Real-time switching of language and currency via UI or geo-detection
- Dynamic fallback rendering for missing content (e.g. revert to English)
- RTL (right-to-left) layout support and font/script adaptation
- Scoped override access (e.g. local brand teams manage their own language)
- Shared translation memory to maintain consistency and accelerate rollout

Core Features

- Translation engine integration with memory and manual override UI
- CMS and Smart PIM localization sync via structured API contracts
- Locale detection system (IP-based, browser-language, user-preferred)
- Region-specific asset injection (e.g. banners, influencer media, video subtitles)
- Multi-currency pricing control (converted vs. fixed local pricing, rounding rules)
- Dynamic SEO metadata generator (localized titles, descriptions, hreflang tags)
- Canonical URL logic to ensure proper indexation by language
- Real-time market switcher (toggle, detection, fallback)

- Full RTL support for layout mirroring and Arabic script rendering
- Centralized fallback engine for content and price localization gaps
- Scoped permissions system for brand teams, editors, and translators

Build Details

- **Frontend**: Connected via React context and Next.js locale-aware routing
- **Backend**: Node.js microservice with translation store, pricing logic, and asset routing
- **Database**: PostgreSQL for locale overrides and settings, Redis for fallback cache
- **Translation Engine**:
    - **Primary**: DeepL API for structured and CMS content
    - **Fallback**: Google or Microsoft Translator via abstraction layer
    - Includes glossary and translation memory support, integrated into PIM and CMS override flows
- **Integrations**:
    - CMS Microservice (for localized content and campaigns)
    - Smart PIM (for product-level translations and attribute localization)
    - SEO Layer (canonical tags, hreflang)
    - Iryss TV (subtitles, language-tagged video metadata)
    - Storefront and Marketplace Frontends
    - Payment Gateway and Pricing Modules (for currency control)

Team & Timeline

- **Team**:
    - 1 frontend developer (routing, toggles, SEO implementation)
    - 1 backend developer (translation sync, fallback, pricing logic)

    - 1 content/localization manager (QA, override workflows, translation memory setup)
- **Timeline**:
    - 5–6 weeks for MVP supporting Marketplace
    - 3–4 weeks extension for Storefronts and Iryss TV localization rollout
    - Continuous integration with PIM and CMS translation interfaces

Long-Term Notes

- This engine is the **single source of truth** for all localization behavior in Iryss
- Enables consistent, region-accurate UX across hundreds of brands, languages, and surfaces
- Supports future expansion into any new geography without frontend duplication
- Directly improves search performance, campaign relevance, and conversion by delivering native-language experiences
- Structured to scale with large product catalogs, dynamic campaigns, and multilingual content velocity
- Translation engine is **abstracted**, allowing vendor replacement if required in future without major system refactor

10. Marketplace Dashboards System

What it is

The central UI system for all platform roles Super Admin, Admin, Brand, Vendor, Reseller, Influencer, and Customer. Each role accesses a tailored dashboard interface powered by API-driven microservices. These dashboards allow management of listings, inventory, orders, payouts, analytics, messages, and storefronts. The dashboards are deployed in three stages:

1. **Marketplace Dashboards** – for core commerce operations
2. **Storefront Dashboards** – for direct-to-consumer store control
3. **Iryss TV Dashboards** – for media and content asset management

Dashboards are built as a unified control layer across all Iryss products, directly connected to the **Product Information Management (PIM)** system, OMS, Payments, and AI layers.

What it enables

- Role-based access to operational tools with permission matrix and tiered visibility
- Full management of listings, inventory, pricing, and variants (via PIM microservice)
- Real-time order, payout, stock, and traffic visibility
- Embedded microservice widgets (e.g. messaging, shipping, analytics, AI tools)
- Unified user experience across Marketplace, Storefronts, and Iryss TV
- Cross-channel control for multi-surface vendors (e.g., Marketplace + Storefront)
- Scalable support for thousands of concurrent vendors, brands, and influencers
- Mobile-optimized access for brands and teams on the move

Core Features

- Modular, widget-based dashboard layout with microservice plugin support
- Full PIM integration: listing creation, product enrichment, and bulk updates
- Super Admin impersonation and role-switching for platform support teams
- Global search, advanced filtering, and audit log trails
- Tiered internal user roles within each vendor/reseller account (e.g. staff vs owner)
- Central message center with system alerts, chat, and push notifications
- Bulk management: orders, payouts, listings, and media
- Custom branding options: logos, dark/light modes (Phase 2)
- Dynamic dashboard modules per surface (Marketplace, Storefront, TV)

New Enhancements

- **Workflow Automation Builder (Phase 2):**
  No-code flow builder to automate common logic (e.g. flag low stock, notify on sales spike)
- **Support Panel with Team Access:**
  Allow brands/resellers to add sub-users with limited permissions (inventory manager, marketer)
- **Theme Customization Layer (Phase 3):**
  Enable visual dashboard customization to match storefront brand identity

Integrations

- **PIM Microservice:** Full control over listings, variants, enrichment, sync across surfaces
- **OMS Microservice:** Order and returns management
- **Payments Microservice:** Payouts, invoice history, billing logic
- **Analytics Platform:** Sales performance, campaign data, traffic sources

- **Messaging Microservice:** Buyer, vendor, and internal communication threads
- **Localization Engine:** Regional display logic, language toggles, multi-market support
- **Consent Manager & Notification Layer:** UI compliance and messaging controls
- **AI Application Layer:** Personalized suggestions, auto-tagging, behavior scoring

Build Details

- **Frontend:** React (Next.js), modular layout system with context-driven rendering
- **Backend:** Node.js/NestJS, consuming GraphQL APIs from all microservices
- **Design System:** Shared UI component library (e.g., Tailwind, ShadCN/UI)
- **Authentication & Permissions:** JWT/GraphQL-based access with role hierarchy
- **Mobile Optimization:** Fully responsive from Phase 1

Team & Timeline

- **Phase 1 – Marketplace Dashboards:**
  2 Frontend Developers
  1 Backend Engineer
  1 UI/UX Designer
  Timeline: 6–8 weeks
- **Phase 2 – Storefront Dashboards:**
  Extend existing team
  Timeline: 5–6 weeks
- **Phase 3 – Iryss TV Dashboards:**
  Extend again for media and content management
  Timeline: 5–6 weeks

Long-Term Notes

- Dashboards act as the **main gateway for all vendor and internal operations**
- Maintained as a **standalone front-end platform** interacting with Iryss microservices
- Will expand to include **future user roles, partner interfaces, and vertical-specific views**
- Fully modular, enabling future features (AI overlays, loyalty controls, merchandising UI) to be embedded without rework

11. Data Pooling & Warehouse Infrastructure

What it is

The central data backbone of the Iryss ecosystem. This microservice aggregates, structures, and continuously updates all platform data including products, users, orders, campaigns, and engagement signals into a unified warehouse that powers AI, analytics, personalization, and automation.

What it enables

- Real-time and batch ingestion of data from all marketplace, storefront, and TV interactions
- Clean, queryable structure for downstream AI and analytics services
- Continuous training loops for AI microservices (e.g., personalization, fraud scoring, outreach targeting)
- Data segmentation and metric availability across dashboards, scoring engines, campaign tools, and internal tools
- Real-time syncing into tracking, CRM, recommendation engines, and retargeting systems
- Automated historical data snapshotting for trend analysis and forecasting

Core Features

- Centralized event ingestion pipelines (Kafka / event bus) from all services
- Schema-normalized warehousing of product, order, user, and media data
- Preprocessing engine for data cleaning, enrichment, and transformation
- Role-based querying layer for internal tools and dashboards
- Historical archiving and snapshot logic for time-series analysis
- AI-ready dataset formatting for model training and model serving
- Secure and scalable infrastructure with access control and audit trails

Build Details

- **Infrastructure**: Snowflake, BigQuery, or Redshift for warehousing; Kafka or AWS EventBridge for pipeline logic
- **ETL Layer**: dbt or custom scripts for transforming raw events into structured tables
- **Hosting**: Managed cloud infrastructure with autoscaling and redundancy
- **Data Security**: Role-based access, encryption, and audit logging built in

Integrations

- AI microservices (recommendation, scoring, automation)
- Marketplace dashboards, campaign manager, product feed system
- Tracking and attribution engines
- Third-party analytics or BI tools (e.g., Looker, Metabase)
- Internal CRM or segmentation tools

Team & Timeline

- **Team**: 2 data engineers, 1 backend engineer, 1 part-time data analyst
- **Build Time**: ~6–8 weeks for v1 infrastructure and pipelines, followed by continuous enrichment

Long-Term Notes

- This infrastructure is the **core of Iryss's self-learning flywheel**: it continuously ingests platform-wide data, feeds AI microservices, collects output and feedback, and retrains models over time to drive compounding optimization
- Powers every downstream service that requires structured, up-to-date data: personalization, automation, scoring, outreach, ads, analytics, and fraud signals
- Built to scale as a **central brain** for all current and future microservices designed for adaptability and modular evolution
- Enables AI-driven commerce by turning all platform activity into structured intelligence loops, allowing the system to autonomously optimize store operations, catalog merchandising, customer journeys, and revenue performance over time

12: Analytics & Insights Platform

What it is

The Analytics & Insights Platform is a centralized, queryable microservice that delivers real-time and historical metrics across all parts of the Iryss ecosystem. It powers every business intelligence function and is the backbone of performance visibility, optimization feedback, and data-driven decision-making across roles (Admin, Brand, Reseller, Influencer, Customer).

What it enables

- Unified analytics layer used by dashboards, storefronts, campaign manager, and admin panels
- Custom KPIs, trend reports, and benchmarks per brand, channel, campaign, or cohort
- AI training data loops (e.g. conversion rates by segment, price elasticity models)
- A/B testing, channel attribution, and storefront-level performance visibility

- Real-time feedback on ad ROAS, reseller behavior, customer funnel drop-off
- Self-service analytics widgets embedded across dashboards
- Central audit trail of platform activity, behavior patterns, and role-specific metrics

Core Features

- Queryable metrics API for any frontend or third-party tool
- Segmented cohort analysis (by vendor, role, traffic source, region, etc.)
- Sales funnel tracking (session to checkout, by storefront or campaign)
- Campaign analytics (clickthroughs, ROAS, conversions, attribution windows)
- Storefront analytics (customer behavior, top SKUs, bounce rates)
- Trend detection across SKUs, search, and sales velocity
- Time-series dashboards for seasonality and benchmark monitoring
- Predictive overlays for inventory, pricing, and influencer ROI
- A/B test manager with statistical significance engine

Build Details

- **Backend:** Node.js or Python with analytics frameworks (e.g. TimescaleDB, ClickHouse, or BigQuery)
- **Storage:** Connects directly to Data Pooling & Warehouse Infrastructure (Project 11)
- **API Layer:** GraphQL or REST interface to serve metrics to dashboards, apps, and services
- **Security:** Role-based access to metrics (e.g. vendor can only see their own data)
- **Compliance:** Consent-aware and cookie-filtered to match GDPR and Consent Manager (Project 17)

Integrations

- Feeds into: Dashboards System (Project 10), Storefronts (Project 13), Campaigns/Ads (Project 4), AI Layer (Project 14)
- Draws data from: Data Pooling & Warehouse (Project 11), Orders, Payouts, Traffic, Events
- Optional: Integration with Looker, Metabase, or custom BI UIs if needed

Team & Timeline

- Team: 1 Data Engineer, 1 Backend Engineer, 1 Frontend Engineer (for dashboard widgets)
- Timeline: ~6 weeks for MVP (API, core metrics, live feeds); +2 weeks for A/B testing + cohort manager

Long-Term Value

- Enables real-time monitoring, business optimization, and AI feedback loops
- Reduces support requests through vendor self-service insights
- Forms the intelligence layer behind revenue-generating decisions (e.g. ad spend, inventory allocation, discounting strategy)
- Critical to transforming Iryss into a self-optimizing commerce platform with continuously improving performance

13. Storefronts Infrastructure (Shopify Alternative)

What it is

The Iryss Storefronts system is a fully modular, no-code website infrastructure that allows any approved brand, reseller, or influencer to launch their own direct-to-consumer web store within minutes. These storefronts are built as a connected extension of the Iryss commerce ecosystem sharing inventory, orders, marketing assets, AI features, and analytics with the main marketplace and platform microservices. Designed to match or exceed Shopify in ease of use, this infrastructure delivers high-level automation, personalization, and intelligent product and campaign management with minimal effort from the user.

What it enables

- **Unified commerce control**: Brands can manage their Iryss Marketplace listings and their own storefront from a single dashboard.
- **Lightning-fast setup**: No-code builder and default templates make it possible to launch a store in minutes.
- **Full automation and personalization**: Storefronts self-optimize over time using the Iryss AI engine (e.g. smart pricing, product highlights, content blocks).
- **Content and media sync**: CMS and Iryss TV integrations allow rich visual merchandising, video banners, and influencer reels on storefronts.
- **Channel-wide data sharing**: All storefronts feed into and learn from the centralized data pool, training the AI engine and enhancing recommendation accuracy.
- **SaaS-ready model**: Storefronts can be monetized via subscription, made public-facing for non-marketplace brands, or offered as white-label solutions.

Core Features

- Drag-and-drop no-code storefront builder (layout blocks, product sections, hero banners)
- Shared inventory sync with Marketplace via Middleware

- Integrated checkout, shipping, returns, and taxes via OMS & Payments microservices
- CMS-managed content: homepage, collection pages, product templates, campaigns
- Smart merchandising (AI-driven cross-sell, trending blocks, price drop badges)
- Built-in localization: multi-language, currency switcher, market-specific inventory
- Integrated analytics: visitor data, funnel tracking, conversion rates
- Iryss TV video embedding: live episodes, reels, brand promos
- Chat/messaging: customer interaction tools pulled from Messaging microservice
- Consent & cookie banners embedded automatically
- Theme customization and custom domains (via SaaS mode)
- Optional storefront subscription tiers and billing (future)

Build Considerations

- **Three-phase rollout**:
    1. **MVP Launch**: Using existing CMS and Deity Studio logic to build the first connected storefronts with shared product sync.
    2. **UX Expansion**: Customization tools, analytics view, customer messaging, AI merchandising.
    3. **SaaS Mode**: White-label access, public storefront onboarding, tiered subscriptions, and standalone operations.
- **Connected Microservices**:
    - **CMS Microservice**: Controls layout, content, and media blocks.
    - **Localization Engine**: Language, currency, and regional logic.
    - **Payments Microservice**: Unified checkout and invoicing.
    - **AI Application Layer**: Merchandising, pricing, scoring, recommendations.

- **Analytics Platform**: Dashboards and conversion analysis.
- **Iryss TV Distribution API**: Embedded video and media tools.
- **Consent, Notification, Messaging Microservices**: Compliance and user interaction.

Team & Timeline

- **Team Required**:
  - 1 Frontend Engineer (UI builder, React/Next.js)
  - 1 Backend Engineer (sync logic, domain setup, SaaS handling)
  - 1 UX/UI Designer (Shopify-grade templates and builder UX)
  - 0.5 DevOps (containerization, DNS, scaling)
- **Estimated Timeline**:
  - **MVP Storefronts**: 6–8 weeks
  - **Full UX + AI Expansion**: 6 weeks
  - **SaaS Launch**: 4–6 weeks (if activated)

14. AI Application Layer (Personalization & Automation)

What it is

The core intelligence engine of the Iryss platform this microservice centralizes and powers all AI functionality across Marketplace, Storefronts, and Iryss TV. It transforms raw behavioral, transactional, product, and marketing data into actionable automation. It enables personalization, decision-making, predictive logic, and continuous optimization. This is the foundation of Iryss's self-learning, AI-native commerce OS.

What it enables

- Personalization of all user touchpoints: homepage, search, banners, content, and notifications
- Smart automation of backend operations: tagging, pricing, onboarding, curation, scoring, fraud flags
- Continuous self-learning from every user, order, return, product update, and campaign
- Data activation across all channels and microservices through API calls or event-driven triggers
- Intelligent storefront configuration and merchandising without human input

Core Features

- **AI Recommendations Engine**: Learns from views, adds to cart, and purchases to optimize product feeds
- **Dynamic Pricing Engine**: Adjusts prices based on demand, competitor scraping, sell-through, and margin targets
- **Smart Product Tagging**: Classifies product attributes from imagery, text, and known data powering search and SEO
- **AI Description & Copy Generator**: Auto-generates localized product text, banners, and content blocks
- **Auto-Merchandising Engine**: Curates categories, storefront layouts, and promotional placements using real-time data
- **Reseller Fit Engine**: Recommends ideal products for each reseller based on their niche, audience, and performance
- **Buyer Scoring & Segmentation**: Categorizes users by behavior, value, fraud risk, or likely future purchase intent
- **Reseller/Influencer Scoring**: Identifies top performers, underperformers, and content/brand fit using data from multiple services
- **Onboarding Optimization Bot**: Uses real-time behavior to guide brands and resellers step-by-step through setup

- **Fraud Signal Engine**: Flags orders based on return abuse, payment behavior, device fingerprinting, and anomaly detection
- **Campaign Feedback Loop**: Measures performance of creative assets, influencer content, and ad segments to improve future delivery
- **Self-Learning Feedback Hooks**: Embedded across microservices to update scoring, tagging, personalization, and risk detection in real-time
- **Optional Future Features**:
    - Smart size/fit recommender
    - Product design suggestion engine
    - Forecasting for production quantities

AI Deployment Model

- **Training Layer**: Central model training based on structured data from the Data Pooling & Warehouse system
- **Inference Layer**: Deployed AI models accessible to any microservice via API or event triggers
- **Segmentation Logic**: All outputs tailored by platform (Marketplace, Storefront, Iryss TV) and user role (admin, brand, customer, etc.)
- **Continuous Optimization**: Every interaction improves model accuracy, forming a true self-improving commerce OS

Build Details

- **Data Ingested From**: Product updates, user sessions, orders, returns, campaign logs, reviews, influencer media
- **Data Output To**: CMS, Dashboards, Middleware, TV content sorting, Storefront auto-layout, etc.
- **Integration Points**: Touches nearly all Iryss microservices each one is AI-infused and continuously feeds back into the loop

Team & Timeline

- **Team**:
    - 1 Lead AI Engineer (architecture, model selection, feature strategy)
    - 1 Data Engineer (data flow pipelines, labeling, preprocessing)
    - 1 Backend Engineer (service APIs, event triggers, inference layer)
    - 0.5 Frontend Engineer (UI embedding: personalization, pricing, scores)
- **Timeline**:
    - MVP (Recommendations, Tagging, Pricing, Onboarding): 8 weeks
    - Phase 2 (Scoring Engines, Auto-Merchandising, Risk Layer): 6–8 weeks
    - Ongoing: Continuous training, refinement, and expansion of use cases

Final Notes

- This is the **central optimization engine** of Iryss. Every service is either feeding it or benefiting from it.
- This project creates long-term defensibility by enabling **AI-native commerce** not just AI-added.
- The value compounds with every brand, product, user, and reseller added.
- Fully modular: future AI tools (design engine, personalization-as-a-service, custom GPTs) plug in here.

15. Event & Notification Microservice

What it is

A real-time trigger and messaging engine that powers all alerts, nudges, and status changes across the Iryss ecosystem. This microservice listens for platform events (e.g. order placed, product out of stock, return approved, payment completed), then broadcasts actions via email, SMS, in-app UI, or webhook. It supports both transactional and behavioral logic and integrates with AI-driven nudges and automations.

What it enables

- Seamless communication across user roles (vendors, brands, resellers, shoppers)
- Real-time operational updates (order events, payment updates, delivery alerts)
- Platform-wide messaging automation (abandoned carts, review requests, payout notices)
- Smart customer nudges using AI-scored actions (timing, tone, content)
- Reseller alerts for new uploads, restocks, or marketing materials
- Influencer notifications for performance updates or campaign tasks
- Developer and admin logs for infrastructure or workflow errors

Core Features

- Unified event bus (Kafka or RabbitMQ) to capture platform-wide events
- Custom rules engine to define triggers, conditions, and actions
- Template builder for dynamic emails, SMS, and push messages
- Multi-channel delivery with fallback logic
- Time-based scheduling (e.g. delay send, retry logic, drip campaigns)
- User-level preference center (opt-in, opt-out, frequency control)
- Admin panel for monitoring, logs, and delivery performance

- Localization and language auto-matching for message templates
- Webhook triggers for external system alerts (ERP, CRM, 3PL, etc.)
- Notification hooks for internal UIs (storefront dashboard, marketplace, mobile apps)

Build Details

- **Backend**: Node.js or Python microservice, powered by Redis queues or Kafka streams
- **Frontend**: Admin UI in React for rules, templates, and performance logs
- **DB**: PostgreSQL or MongoDB for template storage, logs, and delivery analytics
- **External Services**:
    - Email (e.g. SendGrid, Mailgun)
    - SMS (e.g. Twilio, Vonage)
    - Push (e.g. Firebase, OneSignal)

Team & Timeline

- **Team**: 1 backend, 1 frontend
- **Build Time**: ~4 weeks for core event engine, channel integration, and admin panel
- **Post-Launch**: Add advanced logic (e.g. multivariate testing, send time optimization), webhook APIs, message score tuning

Long-Term Use

- Powers every Iryss notification or alert from marketplace to TV to storefronts
- Forms the automation layer for future AI-driven personalization (send the right message at the right moment)
- Can eventually support promotional campaigns, loyalty nudges, and AI-generated comms

16. Consent, Cookie & GDPR Manager

What it is

A centralized microservice responsible for capturing, storing, and enforcing user consent across all Iryss properties Marketplace, Storefronts, and Iryss TV in compliance with EU GDPR, ePrivacy Directive, and global privacy standards. It governs all cookies, trackers, and personalized services based on real-time consent state and provides granular control for both end users and platform admins.

What it enables

- Full legal compliance with GDPR, ePrivacy, CCPA, and other data protection frameworks
- Dynamic cookie banners and consent modals localized by region
- Real-time enforcement of opt-in/opt-out for marketing, tracking, and personalization
- Persistent consent storage linked to user session, device, and account
- Consent-based activation of AI services, retargeting pixels, and data sharing
- Audit trail and proof of consent for regulatory review
- Configurable admin dashboard to define consent categories, integrations, and policies

Core Features

- Real-time consent state engine API queries return per-session status (granted, denied, pending)
- Geo-based display rules for banners (EU, UK, US, etc.)
- Cookie category management (Essential, Performance, Marketing, Personalization)
- Integration with third-party tools: Google Tag Manager, Meta Pixel, TikTok Pixel, Segment

- Multilingual consent banners with auto-detection of user language
- Backend service to activate/deactivate scripts and AI functions based on user choices
- Consent log database with timestamps, versions, and status updates
- Admin interface to manage banner design, consent text, and integration scripts
- Syncs across devices for logged-in users (browser, app)
- Soft opt-in tracking and grace period support for email and retargeting
- Links with Preference Center and Account Settings for user review and changes

Build Details

- **Frontend**: Consent modal + banner in React, localized and dynamically loaded
- **Backend**: Node.js microservice with Redis or PostgreSQL for session storage and API responses
- **Storage**: Encrypted database for consent logs, status, and user/device mapping
- **Middleware**: Connected to Tag Manager, server-side script manager, and event queue
- **Legal Base Management**: Predefined logic for GDPR Article 6 (Consent, Legitimate Interest, Contract)

Team & Timeline

- **Team**: 1 frontend (modal/UI/logic), 1 backend (API/state engine)
- **Build Time**: ~4 weeks for core MVP with frontend UI, backend state sync, and GTM integration
- **Post-Launch**: Add consent versioning, region-based customization, and regulatory dashboard for audit exports

Long-Term Use

- Acts as the privacy compliance backbone across all future Iryss surfaces (web, mobile, TV)
- Enables conditional loading of future AI features, personalization, or paid ad tools
- Protects Iryss from regulatory risk while allowing intelligent tracking where legally permitted

17. Messaging & Chat Microservice

What it is
A secure, real-time messaging engine enabling structured communication between users across the Iryss platform. Designed to support marketplace interactions (buyers, sellers, resellers), B2B conversations (brands ↔ resellers), and storefront-native messaging (customer ↔ brand). The system supports both synchronous and asynchronous messaging with full audit, role permissions, and notification triggers.

What it enables

- In-platform conversations without third-party tools
- Role-based access (e.g. vendors can't chat directly with customers unless a purchase is made)
- Threaded communication between any two valid parties: reseller↔brand, buyer↔vendor, Iryss↔seller
- Escalation logic for disputes or flagged content
- Internal communication (e.g. Iryss team ↔ vendors/resellers)

Core Features

- **Threaded chat by role and transaction**
  e.g. "Order #98765 – Buyer ↔ Brand" or "Reseller ↔ Brand – Product Inquiry"
- **Smart message routing**
  Dynamic creation of threads based on user action (purchase, wishlist inquiry, etc.)
- **Real-time delivery**
  WebSocket or server-sent events with fallback to polling for low-resource environments
- **Mobile & Desktop compatibility**
  Integrated natively into Iryss Marketplace, Storefronts, and Iryss TV mobile apps
- **Multi-language message templating**
  Smart pre-filled templates based on user region and intent (e.g., return, sizing query)
- **Notifications system integration**
  Triggers email/push/SMS alerts for unread messages or critical updates
- **Moderation tools**
  Spam filtering, block/report functionality, Iryss moderator access
- **AI assistance hooks** (optional future use)
  Ability to embed generative AI for support auto-replies or FAQ surfacing
- **Escalation workflows**
  Message flagging leads to Iryss team intervention or automatic dispute escalation

How it connects to the platform

- Embedded in all user dashboards (Marketplace, Storefront, Admin)
- Works across buyer, brand, reseller, influencer, and internal roles
- Integrated into the notification microservice and dashboards system
- Shared logic across web and mobile (all three apps)

Build Details

- **Frontend:** React chat component (widget-style) with role-based rendering
- **Backend:** Node.js service with socket or event bus (e.g., Redis pub/sub, WebSocket)
- **Database:** MongoDB (message storage), Redis (presence), Elasticsearch (search in threads)
- **Security:** End-to-end encrypted storage, scoped visibility by user and thread
- **Scalability:** Horizontally scalable with stateless architecture and message queue for delivery

Team & Timeline

- **Team**: 1 frontend, 1 backend, 1 infra/devops
- **Build Time**: ~5–6 weeks MVP (core messaging, presence, role threads, notifications)
- **Post-launch**: Add auto-translation, AI reply support, storefront-native chat UI variant

18. Iryss TV Infrastructure

What it is

A dedicated content infrastructure microservice powering the full backend and logic behind Iryss TV the video-first media and entertainment layer of the Iryss ecosystem. It handles creator uploads, branded video content, tagging, encoding, and syndication into storefronts and marketplace pages. This microservice enables Iryss to deliver a Netflix-meets-QVC experience with shoppable, influencer-driven content embedded across all channels.

What it enables

- Branded product storytelling through video content
- Creator-driven reels, episodes, or live segments attached to product listings
- Upload, preview, tag, and manage video across Marketplace, Storefronts, and Iryss TV
- Shoppable video overlays linked to synced product SKUs
- Full syndication into storefronts and external reseller stores
- SEO-friendly and embed-ready video pages with metadata
- Moderation, scheduling, and performance analytics
- Future AI-driven tagging and video content recommendations
- Integration with campaign and product lifecycle workflows for smarter content use

Core Features

- Video Upload Portal for brands, influencers, and internal creators
- Content Management UI with version control and approval workflow
- Tagging System for product, category, style, creator, season, and mood
- Encoding & Format Optimization (HLS/MP4, mobile-first streaming)
- Live/Pre-Recorded Support for drops, events, or launches
- Embed SDK & Iframe Logic to power syndication across all storefronts
- Metadata Sync to keep video content and product listings unified
- Creator Attribution (track views, sales impact, usage)
- Scheduling, Drafting, and Expiry Logic
- Content Flagging & Moderation Tools
- Performance Analytics Hooks (views, CTR, conversion)

Build Details

- **Frontend:** React-based uploader and CMS interface
- **Backend:** Node.js microservice with REST/GraphQL endpoints
- **Media Layer:** Cloud storage + CDN (e.g., Cloudflare Stream, Mux, or AWS MediaConvert)
- **Database:** PostgreSQL for metadata and tagging; Elastic for fast search
- **Security:** Role-gated access for upload and editing; CDN token auth
- **Infrastructure:** Dockerized, horizontally scalable (content storage and CDN-driven)

Integrations

- CMS Microservice (pulls associated product content for linking)
- Storefronts & Marketplace (embed player + autoplay hooks)
- Content Distribution API (resellers pull videos and assets)
- Analytics & Data Pooling (track watch behavior and feed AI training)
- Consent/Cookie Layer (to ensure GDPR-safe video playback)

Team & Timeline

- **Team:**
    - 1 Frontend Developer (upload UI, tagging interface, playback logic)
    - 1 Backend Developer (storage, metadata, API endpoints)
    - 1 Media Engineer (encoding, streaming config, CDN setup)
    - 0.5 Product/Data Manager (tagging taxonomy, scheduling, QA)
- **Timeline:**
    - MVP (pre-recorded upload, playback, tagging, CMS integration): 6–8 weeks
    - Phase 2 (live video, creator dashboards, performance analytics): +5–6 weeks

Long-Term Considerations

- AI-Assisted Tagging of videos based on visuals and speech
- Creator dashboards for performance tracking and revenue attribution
- Add interactive video overlays and live chat integration
- Syndicate videos to external platforms (YouTube, TikTok, affiliate portals)
- Expand into branded video series and creator monetization tools
- Deeper linking with Campaign Manager, AI Layer, and Analytics Systems

19. Mobile App – Iryss Marketplace

What it is
A native mobile app for the Iryss B2C marketplace, delivering a seamless shopping experience to consumers on iOS and Android. Designed to replicate and enhance the core web functionality discovery, browsing, checkout while integrating native features like push notifications, biometric login, and real-time updates. Built with scalability, performance, and entertainment-first commerce in mind.

What it enables

- On-the-go access to the full Iryss marketplace for end customers
- Native speed and experience for better conversion and engagement
- Push-based customer reactivation and campaign performance boosts
- Deep integration with Iryss TV for shoppable content
- Full multilingual and multi-currency support for global usage

Core Features

- Marketplace browsing (category filters, search, personalized feeds)
- Product pages (media-rich, shoppable video, reviews, and specs)
- Add to cart, wishlist, and saved items sync

- Native checkout (Apple Pay, Google Pay, Klarna, stored cards, etc.)
- Account area (orders, returns, addresses, payment methods)
- Push notifications (order updates, promotions, abandoned cart)
- Integration with Iryss TV (shoppable video episodes and reels)
- Deep link handling for ads, email campaigns, and social sharing
- Offline state handling, biometric login, fast loading UI

Tech Stack & Build

- **Frontend**: React Native or Flutter for cross-platform speed and single codebase
- **Backend Integration**: Connects via GraphQL to the core Deity-based backend
- **State Management**: Redux or equivalent for user/session/data control
- **Notifications**: Firebase Cloud Messaging (Android), Apple Push Notification Service (APNs)
- **Authentication**: OAuth2, biometric login (FaceID, fingerprint)

Team & Timeline

- **Team**:
    - 1 Mobile Lead Developer
    - 1 UI/UX Designer (shared)
    - 1 Backend/API Integrator (shared)
- **Timeline**:
    - MVP in 8–10 weeks
    - Feature expansion and testing in parallel with Marketplace launch

Long-Term Scope

- In-app live shopping events powered by Iryss TV
- Cart recovery journeys with predictive messaging
- In-app loyalty and referral modules
- AI personalization and real-time ranking of products
- App Store optimization for global reach

20. Mobile App – Storefronts

What it is
A native iOS and Android mobile application that powers the individual storefronts created by brands, influencers, and resellers using the Iryss Storefronts Infrastructure (Shopify Alternative). Each storefront app is white-labeled and synced in real time with the central Iryss ecosystem, enabling end customers to browse, shop, and engage directly with their favorite creators and labels via a native experience.

What it enables

- Standalone mobile commerce for each brand/influencer storefront
- Seamless sync with Marketplace listings and inventory
- Real-time app publishing with modular layouts and branded styling
- Native features like push notifications, biometric login, and mobile wallet checkout
- Access to Iryss TV media, lookbooks, and product storytelling tools

Core Features

- Branded storefront UI with custom logo, color palette, and layout
- Product listing pages with filter/sort tools and visual merchandising

- Product detail pages with support for Iryss TV videos, multiple images, swatches
- Cart, wishlist, customer login, order tracking, and return initiation
- Checkout (Apple Pay, Google Pay, Klarna, and standard card options)
- Real-time inventory sync with Iryss middleware and marketplace backend
- Push notifications for promotions, restocks, new arrivals
- Shared customer profiles and order history across storefront + marketplace
- Optional “Powered by Iryss” footer and affiliate attribution

Tech Stack & Build

- **Frontend**: React Native or Flutter (configurable theming and layout builder)
- **Backend Integration**: GraphQL API and Webhooks from Iryss middleware
- **Content Delivery**: CMS-connected via microservice for banners, pages, and video
- **Notifications**: Firebase Cloud Messaging, APNs
- **Custom Branding**: Dynamically configured during storefront onboarding

Team & Timeline

- **Team**:
    - 1 Mobile Developer
    - 1 UI/UX Designer (shared across Storefront + TV)
    - 1 Backend/API Integrator (shared)
- **Timeline**:
    - MVP for storefront browsing and checkout: 6–8 weeks
    - Full template system + push logic: 3–4 weeks post-MVP

Long-Term Scope

- In-app curation tools for brand teams to manage products, content, and collections
- Integration with Iryss TV API to embed promotional videos
- Loyalty, referrals, and customer review collection
- Live chat and CRM extensions
- App Store deployment tools with auto-update capability

21. Mobile App – Iryss TV

What it is

A native mobile app (iOS & Android) for Iryss TV, the video-first, entertainment-led content platform designed to merge commerce with storytelling. This app allows users to watch episodes, reels, and product videos from brands, creators, and influencers and shop directly from them. It's built to support live shopping, bingeable series, product curation, and interactive media experiences.

What it enables

- A TikTok-style, swipe-based discovery experience for fashion and beauty products
- High-engagement video commerce combining storytelling and shopping
- Live drops, influencer takeovers, and episodic brand features
- Direct product purchasing via embedded links
- Social-style interaction (likes, comments, shares) with analytics
- Seamless sync with Iryss product catalog and storefronts
- Real-time audience data for improving content and conversion

Core Features

- Swipe-based video feed for reels, mini-episodes, and live content
- Tap-to-shop overlays on all video content
- Creator and brand profile pages with follow options
- Live shopping mode with countdown timers and inventory updates
- Shop-the-look feature for multi-item video tagging
- Personalized video feed powered by AI viewing behavior
- Integrated push notifications for live content and product drops
- Link-outs to marketplace or storefronts for extended browsing
- Deep CMS and Iryss TV backend integration for easy video management

Tech Stack & Build

- **Frontend**: React Native or Flutter with video-optimized UX
- **Backend Integration**: Pulls data and video assets from Iryss TV Infrastructure
- **Video Delivery**: Uses CDN with HLS/MP4 for fast global playback
- **Checkout Integration**: Deep links into marketplace or storefront flows
- **Recommendation Engine**: Tied to AI Application Layer for feed personalization

Team & Timeline

- **Team**:
    - 1 Mobile Developer (video and UI-heavy)
    - 1 Backend Integrator (Iryss TV sync, product feed connection)
    - 1 UI/UX Designer (shared with Storefronts TV content team)
- **Timeline**:
    - MVP (video feed, tap-to-shop, CMS integration): 6 weeks
    - Full launch (live mode, personalization, content sync): +4–5 weeks

Long-Term Scope

- Creator affiliate monetization with dashboard tracking
- Shoppable series and cross-series promotions
- Integration into influencer storefronts
- UGC support and creator submissions
- In-app social community layer (comments, reactions, shares)

22. POS & In-Person Sales System

What it is
The Iryss POS (Point-of-Sale) System is a mobile-first, hardware-agnostic sales tool that enables brands to manage in-person retail transactions in physical stores, pop-ups, and events. Fully integrated with the Iryss backend, it offers real-time sync with online inventory, unified customer profiles, and seamless order tracking—bridging offline and online commerce into a single ecosystem.

What it enables

- Unified order management across physical and digital channels
- Live inventory syncing with storefronts and marketplace
- Centralized customer purchase history across all sales points
- Branded checkout experiences for boutiques, showrooms, and pop-ups
- Instant generation of digital receipts, returns, and in-store credits
- Multi-device and location sales enablement under one vendor account

Core Features

- Touch-based checkout interface (tablet/mobile optimized)
- Product barcode/QR scanning and search
- Multi-payment method support (cash, card, mobile wallet, Klarna, etc.)

- Receipt printing or email/SMS issuance
- Discount code and promotion application
- Customer lookup (email, phone) and order history view
- Returns, refunds, and exchanges directly from POS
- Inventory deduction in real-time across marketplace and storefronts
- Sales reporting dashboard by device, staff, and location
- Staff PINs and permission levels for multi-person use

Build Details

- **Frontend**: React Native tablet/mobile app (with optional web fallback)
- **Backend**: Connects to Marketplace OMS, Payments Microservice, Analytics Platform
- **Hardware Compatibility**: Built to support card readers, receipt printers, barcode scanners via Bluetooth or local network (e.g. Square, Stripe Terminal)
- **Inventory Source**: Pulls from the Middleware Layer and Storefronts Infra
- **Analytics Integration**: Tied to Iryss Analytics Platform for unified reporting
- **Payments**: Handled through existing Payments Microservice (supports POS methods)

Team & Timeline

- Team: 1 mobile dev (React Native), 1 backend integrator, optional QA/devops
- Timeline: ~5–6 weeks for MVP with tablet interface, checkout, and sync
- Optional extensions post-launch: Hardware pairing UI, advanced reporting, multi-device sync

Long-Term Considerations

- Enables full omnichannel commerce for SME brands
- Can support physical retail rollouts or branded pop-ups across Europe and MENA
- Loyalty points or in-store CRM features can be layered using AI Application Layer
- Compatible with both Iryss Marketplace and Storefront infrastructures
- White-label potential for Iryss enterprise clients or partners

23 – Iryss Smart PIM (Product Information Management System)

What it is

The Iryss Smart PIM is the centralized product data control layer powering all structured product information across the Iryss ecosystem including the Marketplace, Storefronts, B2B reseller portal, Iryss TV, and internal tools. It governs product schema, lifecycle states, enrichment, versioning, localization, campaign mapping, and omnichannel export. It replaces fragmented product management with a single, AI-enhanced, multi-tenant infrastructure purpose-built for marketplace scalability.

What it enables

- Acts as the single source of truth for all SKUs, variants, and metadata
- Powers storefront, marketplace, and Iryss TV listings from one unified product record
- Automates product enrichment using AI tagging and extraction logic
- Supports multi-region product logic (currencies, sizes, languages)
- Streamlines onboarding and editorial workflows at scale
- Enables rich media-linking, influencer attribution, and campaign mapping

- Provides clean, exportable feeds to all connected systems

Core Features

Master Product Repository

- Category-based flexible schema (e.g. fashion, beauty, home)
- Variant logic (e.g. size, color, material)
- Role-based permissions for brands, staff, and resellers
- Lifecycle management: Draft → QA → Published → Expired
- Product completeness scoring and visual dashboards
- Field-level change logs, versioning, rollback, and diff viewer
- Bulk edit engine and mass classification tool
- Merging and deduplication engine for brand duplicates or white-label products

AI Onboarding Tool

- Scrapes brand product pages or documents and maps to internal schema
- Extracts images, descriptions, SKUs, tags, prices, and dimensions
- Auto-classifies into internal taxonomy
- Previews onboarding result before approval
- Enrichment pipeline triggers automatically post-ingestion

AI-Powered Enrichment

- Text and image-based attribute extraction (e.g. neckline, fabric, pattern)
- Vision AI detects silhouette, garment type, or packaging
- Confidence scoring and per-category calibration
- Conflict detection (e.g. “cotton” + “leather”) auto-suppressed

- Enrichment fallback logic using brand defaults or taxonomy presets
- Manual override UI with annotation, history, and review logs
- Reinforcement learning loop from override feedback

Localized Data Layer

- Field-level multilingual support (titles, descriptions, tags)
- Region-specific availability and price visibility
- Local taxonomy alignment per market (e.g. “abaya” vs. “maxi dress”)
- Size mapping matrices for EU/UK/US/MENA formats
- Translation integration via DeepL or equivalent service

Channel-Aware Sync Engine

- Tailored output to Marketplace, Storefronts, B2B, and Iryss TV
- Visibility toggles and variant-level channel routing
- Region-specific sync logic for currency, size, and stock
- Draft, QA, and publish states mapped per channel

Feed Generator

- Headless output to:
    - Orchestrator (Project 5)
    - OMS (Project 6)
    - CMS (Project 8)
    - Localization Engine (Project 9)
    - Search Index (Project 12)
    - Storefront Builder (Project 13)
    - Iryss TV platform (Project 18)
- Webhooks on status change, field update, or publication event

- Sync logs with failure handling and retry logic

Media & Campaign Mapping

- SKU-level asset linking: images, video, model specs
- Campaign tagging, capsule collection grouping, and visual merchandising metadata
- Creator/influencer mapping for Iryss TV and Storefronts
- Attribution data fed to CMS and Iryss TV for stories, bios, and show links

Bulk Upload & Import Layer

- Spreadsheet import with schema validation and auto-fixes
- Per-brand interface with preview, QA, and error suggestions
- Auto-triggers AI enrichment after successful ingest
- Field mapping interface with smart recommendations

How it connects across the Iryss platform

- **Marketplace Dashboards (Project 10):** Brands manage PIM content via their dashboard interface; dashboards display completeness, warnings, and sync history
- **Storefront Infrastructure (Project 13):** All product content for storefronts flows from PIM; visibility toggles and localized content apply per storefront
- **Iryss TV (Project 18):** Video drops, capsule launches, and creator content link to products via PIM metadata
- **CMS (Project 8):** Campaign pages and product stories draw from PIM content for consistency
- **Localization Engine (Project 9):** Pulls only product-level localized fields; UI/local content handled separately
- **Search & Filters (Project 12):** Enriched metadata feeds directly into Elasticsearch or equivalent

System Dependencies

Consumes From:

- Brand Onboarding Engine (scraped product data, spreadsheets)
- AI Application Layer (tagging, image analysis, retraining pipeline)
- CMS content imports (for enrichment and reference)
- Manual inputs and enrichment feedback from QA teams

Pushes To:

- Marketplace Orchestrator (Project 5)
- OMS & Shipping Engine (Project 6)
- CMS Microservice (Project 8)
- Localization Engine (Project 9)
- Search & Filtering Engine (Project 12)
- Storefront Infrastructure (Project 13)
- Iryss TV Infrastructure (Project 18)

Build Details

- **Backend:** Node.js (or NestJS) microservice architecture with modular pipeline for import, enrichment, sync
- **Frontend (admin):** React interface for internal teams and brand users
- **Storage:** PostgreSQL for structured data, object store (e.g. S3) for media
- **AI Integration:** Python microservice layer calling internal/external models for attribute extraction
- **APIs:** REST and GraphQL endpoints for full read/write + webhook publishing
- **Search:** Optional Elasticsearch index integration for filters and discovery

Team & Timeline

- **Team:**
    - 2 backend engineers (schema + sync logic)
    - 1 AI/ML engineer (enrichment + onboarding)
    - 1 frontend engineer (UI for QA/override/import)
    - 1 product architect (taxonomy + flow design)
- **Timeline:**
    - MVP: 10–12 weeks (AI onboarding + enrichment + basic sync)
    - Full rollout: +6–8 weeks (localization, campaign mapping, feed logic)

Long-Term Value

- Centralizes and automates product governance for all Iryss surfaces
- Ensures quality, localization, and structure without manual redundancy
- Future-proofs the platform to support thousands of brands and real-time product changes
- Enables AI-powered workflows that improve with usage, not degrade

Iryss Tech Build Plan – Full Development Timeline & Team Structure

Total Duration: 36 months

Strategy: 3 microservice projects in parallel at all times

Post-Launch: Every project retains at least one dedicated developer for maintenance, upgrades, and roadmap features

Philosophy: Rapid parallel delivery with embedded cross-functional teams to ensure each system scales and improves autonomously

Core Development Team Requirements (Build & Post-Launch)

Role	Quantity (Peak Build)	Post-Launch (Maintenance)
Frontend Devs (React/Next.js)	7	4
Backend Devs (Node.js/NestJS)	8	5
AI/ML Engineers	3	2
Full-Stack Developers	4	2
Data Engineers	2	2
DevOps + Site Reliability	2	2
QA Engineers	2	2
UI/UX Designers	2	1
Project Managers	3	1

Notes:

- Frontend devs cover admin panels, dashboards, mobile apps, and all marketplace/storefront UIs
- Backend devs build and maintain microservices and data flows across the entire architecture
- Full-stack roles fill gaps, especially on early sprints and interservice integration
- DevOps ensures high-availability deployments, rollback pipelines, observability, and scaling
- QA engineers support automation and system-level regression coverage
- AI engineers work closely with data and backend teams on smart pricing, personalization, fraud, etc.
- Designers and PMs rotate across project groups to ensure consistency and roadmap alignment

Development Timeline – Month-by-Month Execution Plan

Phase	Months	Projects in Parallel	Notes
Phase 1	1–6	Projects 1, 2, 3	Core system setup: dev cockpit, scraping engine, middleware infrastructure
Phase 2	7–12	Projects 4, 5, 6	Customer acquisition, advertising, marketplace orchestration
Phase 3	13–18	Projects 7, 8, 9	Payments & accounting engine, CMS, multilingual engine
Phase 4	19–24	Projects 10, 11, 12	Marketplace dashboards, data pooling, analytics engine
Phase 5	25–30	Projects 13, 14, 15	Storefront system, AI app layer, notifications
Phase 6	31–36	Projects 16, 17, 18	Consent/GDPR, real-time chat, Iryss TV
Post-36	—	Projects 19–23 (apps, POS, fraud)	Lighter builds handled by mixed taskforces

Guiding Logic:

- Build order is determined by logical dependency and commercial importance (e.g., AI Cockpit → Scraping → Middleware is foundational; Storefronts come after CMS is live)
- Each microservice is treated as a standalone project with cross-functional ownership
- Apps and POS are built after core platform maturity to avoid premature duplication of features

Strategic Execution Notes

- **Each project team owns delivery, documentation, post-launch expansion, and bug handling** for their service. This avoids backlogs and orphaned features across builds.
- **Devs rotate to new builds** once a project is stable, while one stays permanently assigned for continuous improvement and support.
- **QA is embedded** starting in Month 3 and scales as needed for complex projects (Storefronts, Orchestrator, AI Engine).

- **Design is centralized early** to maintain visual consistency across dashboards, apps, storefronts, and CMS layouts.
- **AI and data roles work continuously**, as training, optimization, and data feedback loops span the entire system lifecycle.
- **Parallelization is enforced strictly**: Always 3 microservices in concurrent development unless special integration timing is needed (e.g., POS tied to Payments).