F&B AI Platform — Complete Documentation Index

> ***Comprehensive guide to all architecture diagrams, flows, and technical specifications*** *| Last updated: February 2026*

🎯 Quick Navigation

🚀 Start Here (Executives & PMs)

1. **Omnichannel WhatsApp-First (Visual Edition)** ⭐ **NEW** — Beautiful visuals, easy to scan
2. **Complete Architecture Diagrams** — Comprehensive visual overview of the entire system
3. **System Specification** — Executive summary, strategy, business value
4. **Autonomous Sales Agent** — Supplier-side value proposition

🏗️ Architecture (Engineers)

1. **Omnichannel WhatsApp-First (Technical)** — Detailed technical architecture
2. **Omnichannel WhatsApp-First (Visual Edition)** ⭐ **NEW** — Easy-to-read visual guide
3. **Complete Architecture Diagrams** — All layers, agents, integrations
4. **Deployment & Infrastructure** — Cloud setup, K8s, security, DR
5. **MedusaJS Architecture** — Backend framework & custom modules
6. **Agentic Architecture** — Multi-agent design patterns
7. **Data Model** — Database schema, Pydantic models, state machines

🔄 Implementation (Developers)

1. **Architecture & Flows** — Detailed workflows, ReAct patterns
2. **Detailed Flows** — Step-by-step: catalog upload, AI cart, GRN, invoice matching
3. **System Design Deep-Dive** — Journey mapping, algorithms, logic specs

---

📊 Architecture Layers

Layer 1: Presentation

- **Restaurant App** (Next.js + React) — Manager UI, dashboards, approvals
- **Supplier Portal** (Next.js + React) — Sales rep UI, performance dashboards
- **Admin Dashboard** (Next.js + React) — System configuration, monitoring
- **WhatsApp Integration** — Interactive buttons, flash deals, order notifications

Layer 2: API Gateway & Authentication

- REST API (Express.js)
- GraphQL API (optional)
- JWT + RBAC auth
- Rate limiting & request validation
- API versioning

Layer 3: Commerce Core (MedusaJS 2.0)

- **B2B Commerce Engine**
  - Products & pricing
  - Orders & fulfillment
  - Inventory management

    - Customer groups & spending limits
    - RFQ workflows
- **Custom Modules**
    - SKU normalization
    - AI suggested cart
    - GRN (Goods Received Notes)
    - Invoice matching (2-way / 3-way)
    - Waste tracking
- **Event Bus** (Redis/BullMQ)
    - order.created, order.updated
    - inventory.low_stock, inventory.adjusted
    - invoice.uploaded, invoice.matched
    - grn.completed, payment.due

Layer 4: AI Agent Orchestration (LangGraph)

Restaurant Side Agents

- **Planner Agent** — Decomposes tasks, routes to specialized agents
- **Catalog / Normalization Agent** — Parses pack sizes, normalizes names, queries vector DB
- **Sourcing Agent** — Compares suppliers, ranks by price/reliability
- **Purchasing Agent** — Drafts smart carts, calculates quantities, validates with Pydantic
- **Inventory Agent** — Monitors stock, depletes from POS, triggers reorders
- **Kitchen Copilot** — Generates prep plans from forecast & inventory
- **Compliance Agent** — Validates invoices, performs 3-way matching, generates e-invoices

Supplier Side Agents

- **Autonomous Sales Agent** — Instant quote generation, basket-aware negotiation, upsells
    - Quote-to-close in <3 seconds
    - Margin guardrails (configurable floor)
    - Menu-aware product recommendations
    - Distressed inventory liquidation

Layer 5: Tools & Functions

- **Pricing Tools**
    - `calculate_cost_per_kg()`
    - `apply_discount_authority()`
    - `validate_margin()`
- **Matching Tools**
    - `parse_pack()`
    - `normalize_name()`
    - `search_similar_skus()`
- **Compliance Tools**
    - `parse_invoice_ocr()`
    - `three_way_match()`
    - `generate_e_invoice()`

- **Notification Tools**
  - `send_whatsapp()`
  - `send_email()`
  - `send_sms()`

Layer 6: External Integrations

- **POS Systems**
  - Foodics API (OAuth 2.0)
  - Oracle Simphony STSG2
  - Generic CSV/Excel upload
- **Payment Gateway**
  - Telr (process payments, refunds)
  - 2Checkout (backup)
- **E-Invoicing**
  - Poppel Network (FTA compliance)
  - ZATCA integration
  - XML + PDF generation
- **Document Processing**
  - AWS Textract (OCR invoices, GRN photos)
  - Google Document AI (fallback)
- **Communication**
  - WhatsApp Business API (interactive messages)
  - SendGrid (email)
  - Twilio (SMS)

Layer 7: Data & Storage

- **PostgreSQL** (Primary DB)
  - Order & transaction data
  - Supplier catalogs
  - Normalized SKUs
  - Invoices, GRNs, audit logs
  - Replicated to 2+ regions for HA
- **Weaviate** (Vector DB)
  - SKU embeddings (OpenAI Ada-002)
  - Semantic search for product matching
  - Equivalency groups
- **Redis** (Cache & Queue)
  - Session storage
  - Cart caching
  - Job queue (BullMQ)
  - Event streaming

- **AWS S3** (Object Storage)
  - Supplier catalogs (CSV, PDF)
  - Invoice PDFs & images
  - GRN delivery photos
  - System documents

🔄 Core Workflows

Workflow 1: Restaurant Low Stock → AI Cart → Approval → PO

```
Low Stock Triggered
    ↓
Inventory Agent: Fetch current levels, par, lead time
    ↓
Catalog Agent: Normalize SKU names, search equivalents
    ↓
Sourcing Agent: Compare 3 suppliers, rank by price
    ↓
Purchasing Agent: Draft cart with quantities & reasoning
    ↓
Pydantic Validation: Ensure data integrity
    ↓
Human Approval: Manager reviews (can edit or reject)
    ↓
PO Created: Send to supplier via Poppel/WhatsApp
    ↓
GRN Scheduled: Reminder set for delivery
```

See detailed flows in: **Detailed Flows**

Workflow 2: Supplier Autonomous Sales Agent

```
Trigger Event:
  - Quote request from chef
  - Predictive order signal
  - Flash deal opportunity
  - POS depletion alert
    ↓
Agent Perception: Intent classification, sentiment, menu parsing
    ↓
Decision Engine: Price authority check, margin calculation
    ↓
Offer Generation:
  - Quote: <3 second response, within guardrails
  - Predictive draft: Smart substitutes, cheaper alternatives
  - Flash deal: Menu-matched, scarcity-driven
    ↓
WhatsApp Interactive: Action buttons, 1-tap acceptance
    ↓
Auto-Confirm PO: Reserve stock, generate E-Invoice
```

```
    ↓
Upsell Check: Bundle recommendations (protect margin)
    ↓
Payment Link: Instant e-invoice via Poppel + Telr
```

See detailed flows in: **Autonomous Sales Agent**

Workflow 3: Invoice Reconciliation (2-Way / 3-Way Match)

```
Invoice Uploaded (PDF)
    ↓
OCR Extraction (AWS Textract / Google Document AI)
    ↓
LLM Parsing: Extract line items, quantities, prices
    ↓
2-Way Match: PO Qty = Invoice Qty?
    ├─ ✅ YES → Approve payment
    └─ ❌ NO → Check GRN
    ↓
3-Way Match: PO vs GRN vs Invoice
    ├─ Short delivery? → Flag shortage
    ├─ Quality issue? → Photo evidence review
    ├─ Overcharge? → Flag price variance
    └─ Match resolved → Approve payment
    ↓
Dispute (if needed): Contact supplier, request credit memo
    ↓
Audit Log: All steps recorded, immutable
```

See detailed flows in: **Architecture & Flows**

🛠️ Key Technologies

Component	Technology	Why?
Backend	Node.js + TypeScript	Type-safe, async-first, REST + GraphQL
Commerce	MedusaJS 2.0	Open-source B2B, extensible, event-driven
AI Orchestration	LangGraph	Stateful agents, human-in-the-loop, tool calling
Frontend	Next.js 14+	SSR, App Router, TypeScript, mobile-ready
Database	PostgreSQL 15+	ACID, JSON support, replication
Vector DB	Weaviate	Semantic search, clustering, managed cloud
Cache/Queue	Redis + BullMQ	Real-time, job processing, event streaming
LLM	OpenAI GPT-4	Reasoning, attribute extraction, negotiation

Embeddings	OpenAI text-embedding-ada-002	1536-dim semantic vectors
OCR	AWS Textract	Invoice & document extraction
Container	Docker + ECS/Fargate	Serverless containers, auto-scaling
Orchestration	Kubernetes (optional)	For agent mesh scaling
Monitoring	Datadog	APM, logs, alerts, dashboards
E-Invoicing	Poppel Network	FTA compliance, ZATCA integration

🌐 External API Integrations

POS Systems

- **Foodics** — OAuth 2.0, fetch orders/inventory, webhooks
- **Oracle Simphony** — STSG2 REST API, recipe-based depletion
- **Generic** — CSV upload, scheduled ETL

Payment & Invoicing

- **Telr** — Payment gateway, refunds, settlement
- **Poppel Network** — E-Invoice generation, FTA compliance, ZATCA
- **SendGrid** — Email delivery
- **Twilio** — SMS & WhatsApp

Document Processing

- **AWS Textract** — Invoice OCR, expense extraction
- **Google Document AI** — Invoice parser, fallback

🔐 Security & Compliance

Data Protection

- **Encryption at Rest**: S3, RDS, Redis (all encrypted)
- **Encryption in Transit**: TLS 1.3 for all APIs
- **Key Management**: AWS KMS for master keys, annual rotation

Compliance

- **UAE E-Invoicing**: FTA/ZATCA XML + PDF generation via Poppel
- **GDPR**: If EU data, full compliance mode
- **PCI-DSS**: Payment handling via Telr (no card storage)
- **Audit Logs**: Every agent action logged, immutable, 3-year retention

Access Control

- **JWT + RBAC**: Role-based access control (Admin, Manager, Storekeeper, etc.)
- **Multi-Factor Auth**: Optional via Okta/Auth0
- **WAF**: AWS Web Application Firewall + CloudFlare

📈 Performance Metrics

System Health

- **API Response Time**: <200ms (p95)
- **Database Query Latency**: <50ms (p95)
- **Event Queue Lag**: <1s
- **Service Uptime**: 99.9% SLA

Business Metrics

- **AI Cart Generation**: <30 seconds
- **Autonomous Sales Response**: <3 seconds
- **Invoice Matching Accuracy**: >99%
- **Order Processing Time**: <5 minutes (end-to-end)

Agent Performance

- **Agent Execution Latency**: 1-5s (depending on complexity)
- **Tool Call Success Rate**: >99%
- **Approval Wait Time**: Avg 15 min (manager action)

📚 Document Structure

```
docs/
├── complete-architecture-visual.md     ⭐ START HERE
│   └── System layers, agents, integrations, flows
├── deployment-infrastructure.md        ⭐ FOR DEVOPS
│   └── Cloud setup, K8s, security, disaster recovery
├── system-specification.md             📋 EXECUTIVE SUMMARY
│   └── Strategy, business value, platform vision
├── medusajs-architecture.md            🏗️ BACKEND
│   └── MedusaJS 2.0, custom modules, LangGraph
├── agentic-architecture.md             🤖 AI AGENTS
│   └── Multi-agent design, ReAct patterns, tools
├── autonomous_sales_agent.md           🎯 SALES AGENT
│   └── Instant-close, negotiation, upsell logic
├── architecture-and-flows.md           🔄 WORKFLOWS
│   └── System flows, approval workflows, data flows
├── data-model.md                       💾 DATA
│   └── ER diagram, Pydantic schemas, state machines
├── detailed-flows.md                   📝 IMPLEMENTATION
│   └── Step-by-step catalog, cart, GRN, invoice flows
├── system-design-deep-dive.md          🔍 DETAILED ANALYSIS
│   └── Journey mapping, algorithms, technical logic
└── architecture-diagrams.md            📊 SIMPLE DIAGRAMS
    └── MVP flow, normalization, SKU matching (legacy)
```

🚀 Deployment Checklist

- ☐ **Infrastructure**: AWS account, VPC, RDS, S3, ECS setup

- [ ] **Secrets**: Environment variables, API keys in AWS Secrets Manager
- [ ] **Database**: PostgreSQL primary + replicas, backups configured
- [ ] **Cache**: Redis cluster, BullMQ setup
- [ ] **Vector DB**: Weaviate cloud instance, embeddings pipeline
- [ ] **External APIs**: Poppel, Telr, Foodics API keys configured
- [ ] **Monitoring**: Datadog agent deployed, dashboards created
- [ ] **Security**: WAF, SSL certificates, KMS keys, audit logging
- [ ] **CI/CD**: GitHub Actions, Docker builds, ECR registry
- [ ] **Testing**: Unit, integration, E2E test suites passing
- [ ] **Documentation**: Runbooks, API docs, operator guides

💬 Support & Questions

For questions about:

- **Architecture**: See `complete-architecture-visual.md`
- **Deployment**: See `deployment-infrastructure.md`
- **API Design**: See `medusajs-architecture.md`
- **AI Agents**: See `agentic-architecture.md` + `autonomous_sales_agent.md`
- **Data Models**: See `data-model.md`
- **Workflows**: See `detailed-flows.md` + `architecture-and-flows.md`

Last Updated: February 2026
Platform Version: 2.0
Status: Production-Ready Architecture