ARCHITECTURE PRESENTATION

# Retail Insights Assistant

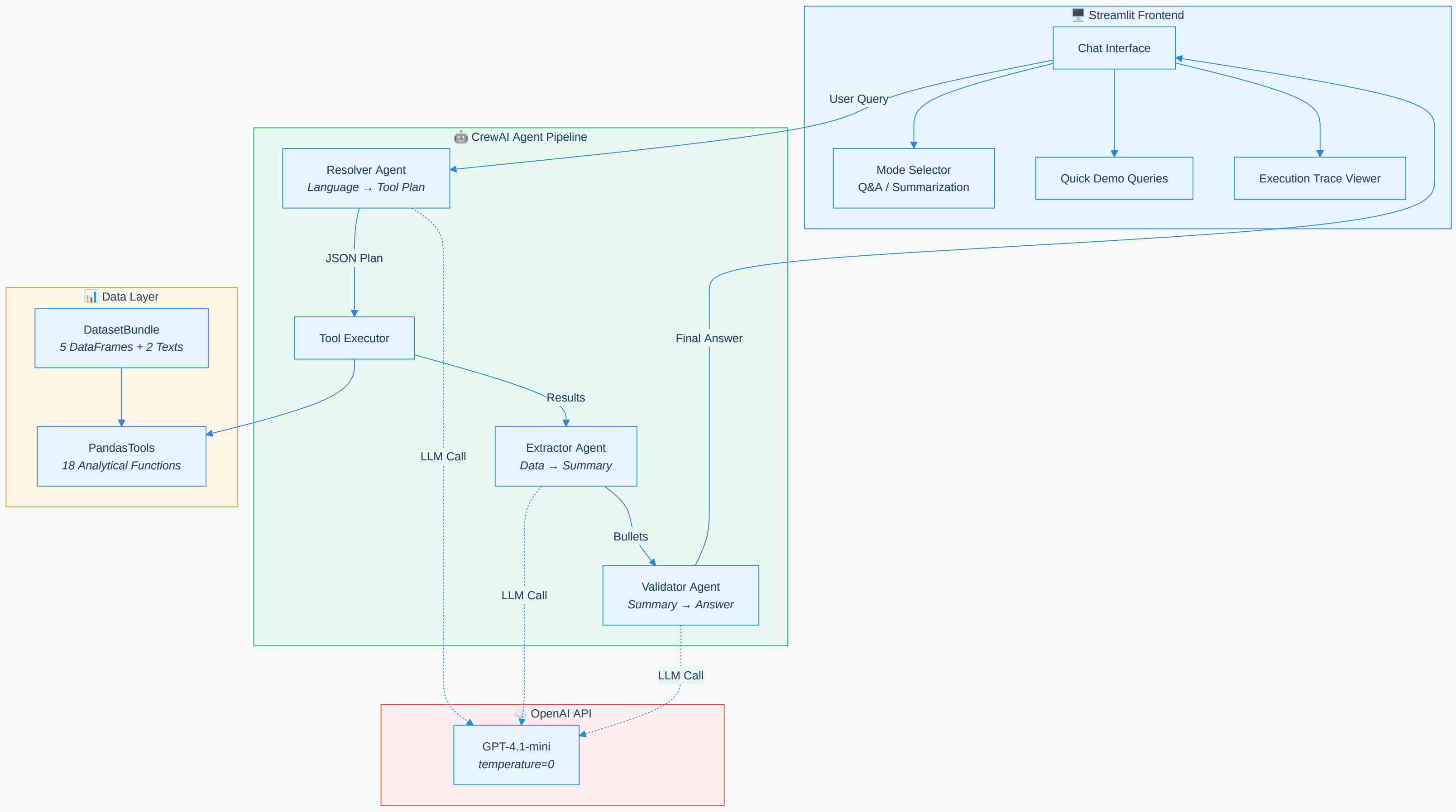
Multi-Agent AI Chatbot for E-Commerce Sales Analytics

Built with **Streamlit** · **CrewAI** · **OpenAI GPT-4.1-mini** · **Pandas**
February 2026

SLIDE 1

# System Architecture Overview

The Retail Insights Assistant is a **multi-agent AI system** that converts natural-language business questions into deterministic data operations on retail sales datasets, returning validated, business-friendly answers.



🖥️ Streamlit Frontend

Chat Interface

User Query

Mode Selector
Q&A / Summarization

Quick Demo Queries

Execution Trace Viewer

🤖 CrewAI Agent Pipeline

Resolver Agent
*Language → Tool Plan*

JSON Plan

Tool Executor

Results

Extractor Agent
*Data → Summary*

Bullets

Validator Agent
*Summary → Answer*

Final Answer

LLM Call

LLM Call

LLM Call

📊 Data Layer

DatasetBundle
*5 DataFrames + 2 Texts*

PandasTools
*18 Analytical Functions*

☁️ OpenAI API

GPT-4.1-mini
*temperature=0*

> **Key Design Principle:** No dynamic code generation. All data operations use pre-built, deterministic Pandas functions — the LLM only decides *which* functions to call and with *what* parameters.

SLIDE 2

# End-to-End Data Flow

Syntax error in text

mermaid version 11.12.3

## Data Sources

| File | Type | Records | Description |
|---|---|---|---|
| `Amazon Sale Report.csv` | DataFrame | ~128K orders | Amazon India orders with status, amounts, geography, B2B flags |
| `International sale Report.csv` | DataFrame | ~37K transactions | International sales with customer, style, rate, gross amount |
| `Sale Report.csv` | DataFrame | ~9K SKUs | Inventory stock levels by SKU, category, size, color |
| `May-2022.csv` | DataFrame | Multi-channel | Pricing across Amazon, Flipkart, Myntra, Ajio, etc. |
| `P&L March 2021.csv` | DataFrame | Multi-channel | Historical pricing and transfer prices by category |
| `Expense IIGF.csv` | Text | — | Financial expense/income statement (comma-stripped text) |
| `Cloud Warehouse Compersion Chart.csv` | Text | — | Warehouse cost comparison: Shiprocket vs INCREFF |

## Cleaning Pipeline

### Column Normalization

Spaces → underscores, special characters removed, consistent casing

### Type Conversion

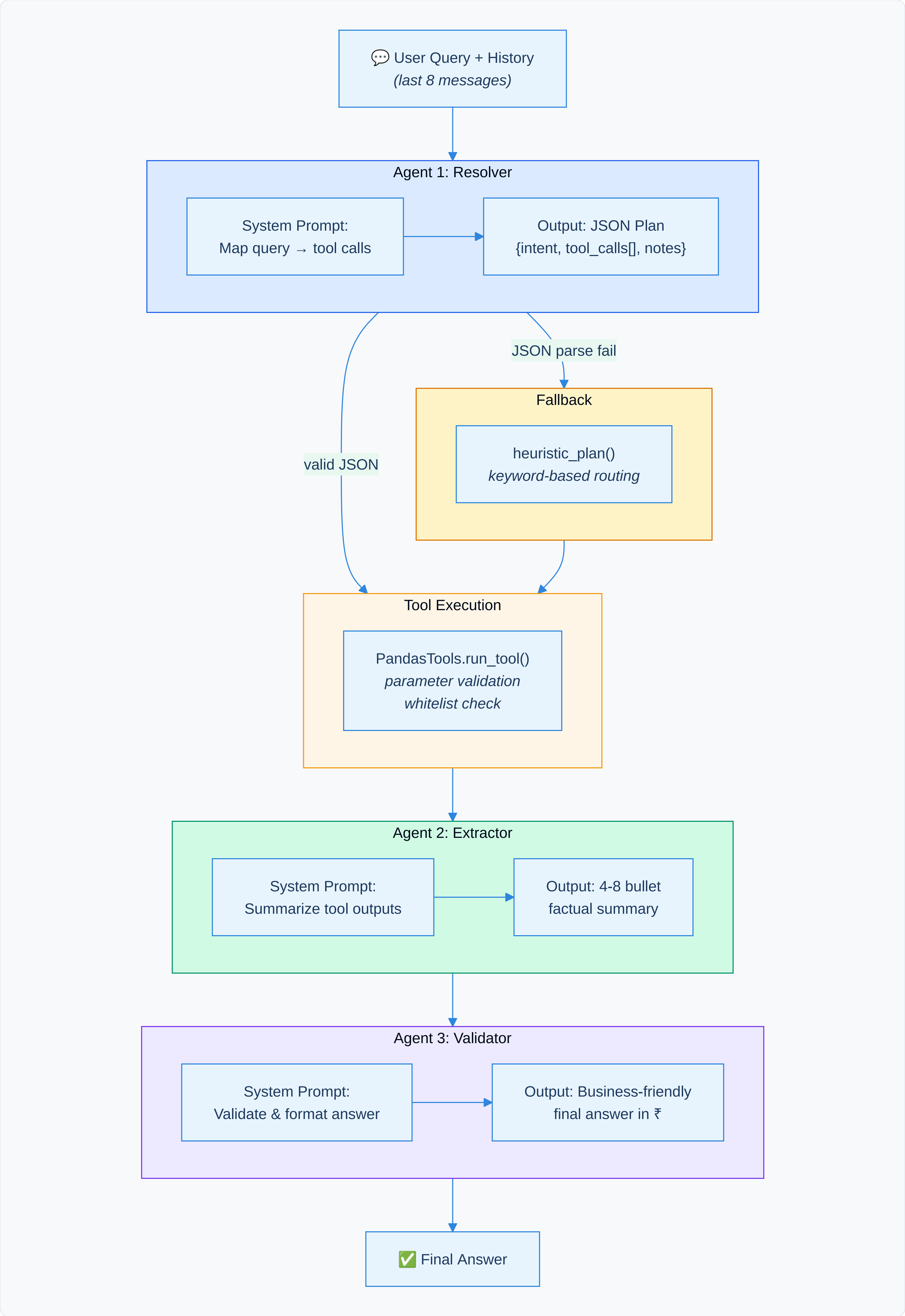
Numeric columns coerced, dates parsed per source format (Amazon vs International)

### Business Logic

Status flags added: `is_delivered` , `is_cancelled` for KPI computation

SLIDE 3

# Multi-Agent Architecture (3-Agent Pipeline)



### 🔍 Resolver Agent

### 📋 Extractor Agent

### ✅ Validator Agent

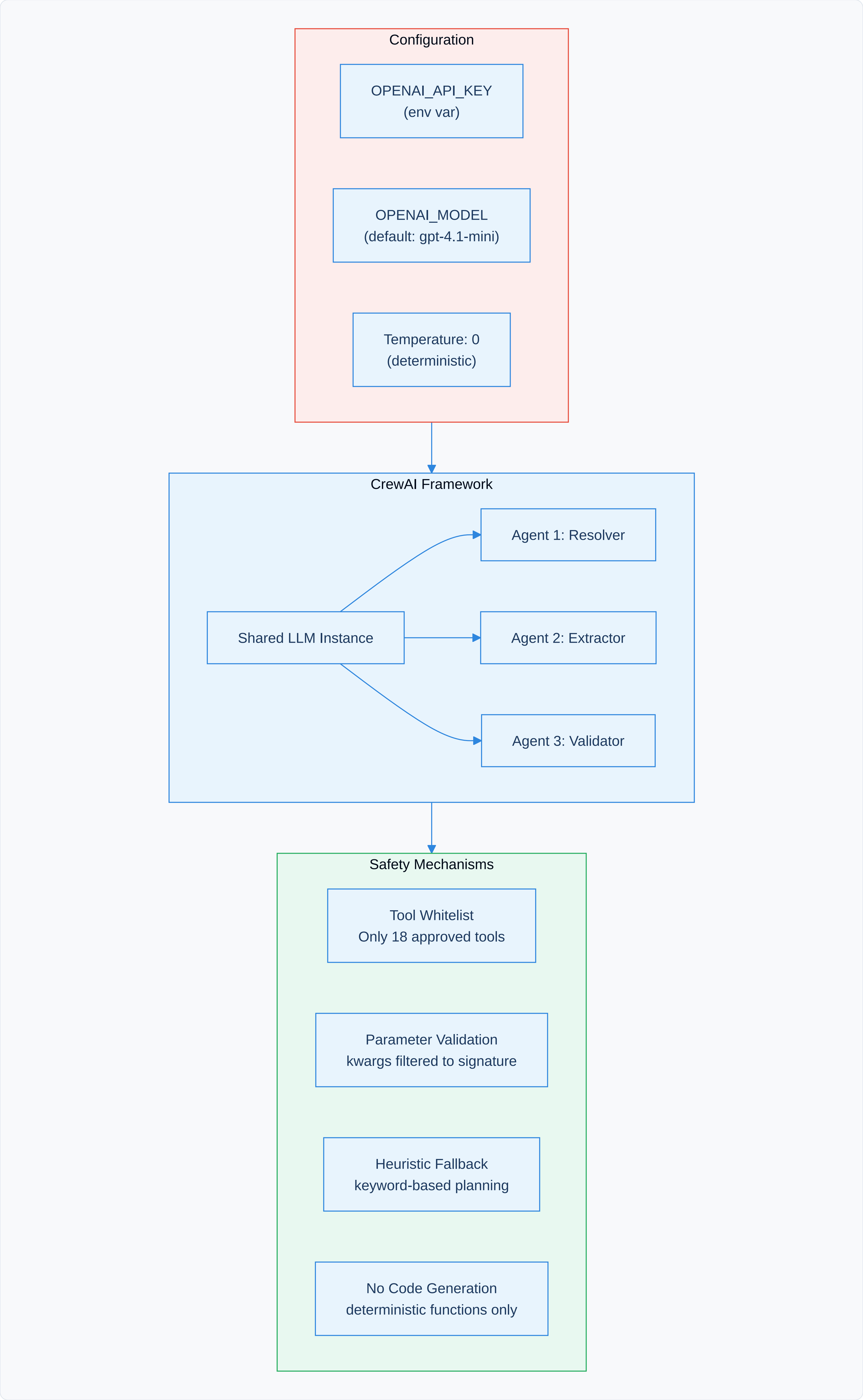
- Maps natural language to structured tool calls
- Outputs JSON with `intent` , `tool_calls[]` , `notes`
- Receives conversation history for context
- Falls back to heuristic planner on parse failure

- Reads raw tool outputs (records, summaries)
- Produces 4–8 factual bullet points
- Handles post-filtering for specific queries
- Formats currency amounts in ₹

- Cross-checks extraction against raw results
- Produces business-friendly final answer
- Mentions partial evidence if data is incomplete
- Mode-aware: Q&A vs Summarization

SLIDE 4

# LLM Integration Strategy



### Why This Approach?

### 🛡️ Security

No `exec()` or `eval()` — LLM cannot execute arbitrary code. All operations go through pre-defined, tested Pandas functions.

### 🎯 Reliability

Temperature=0 ensures consistent outputs. Heuristic fallback guarantees a response even when the LLM returns malformed JSON.

### 💰 Cost Efficiency

GPT-4.1-mini keeps per-query cost low (~$0.001–0.003). Three focused prompts avoid wasted tokens from a single monolithic prompt.

## Prompt Architecture

| Prompt | Input Context | Output Format |
|---|---|---|
| Resolver System Prompt | Query + history (8 msgs) + tools list | JSON: `{intent, tool_calls[], notes}` |
| Extractor System Prompt | Query + tool results JSON | 4–8 bullet points (₹ currency) |
| Validator System Prompt | Mode + query + summary + raw results | Business-friendly final answer |

SLIDE 5

# Analytical Tool Catalog (18 Tools)

Each tool is a deterministic Pandas function returning a `ToolResult` with `tool_name`, `summary`, and `records[]` (capped at 20 rows).

### Amazon Sales (8 tools)

- `total_sales_amazon` — aggregate with date filter
- `units_sold_by_category` — unit counts per category
- `category_sales_rank` — top N categories by ₹
- `state_sales_rank` — top N states by ₹
- `city_sales_rank` — top N cities by ₹
- `order_status_breakdown` — status distribution
- `cancellation_rate` — % cancelled
- `b2b_vs_b2c_summary` — B2B vs B2C split

### International Sales (3 tools)

- `international_total_sales` — with month filter
- `top_customers_international` — top N by revenue
- `top_styles_international` — top N styles

### Inventory (3 tools)

- `total_stock_by_category` — stock per category
- `low_stock_items` — below threshold (default 5)
- `stock_by_color` — stock per color

### Pricing (2 tools)

- `channel_price_comparison_may2022` — cross-channel Δ
- `price_snapshot_march2021` — historical snapshot

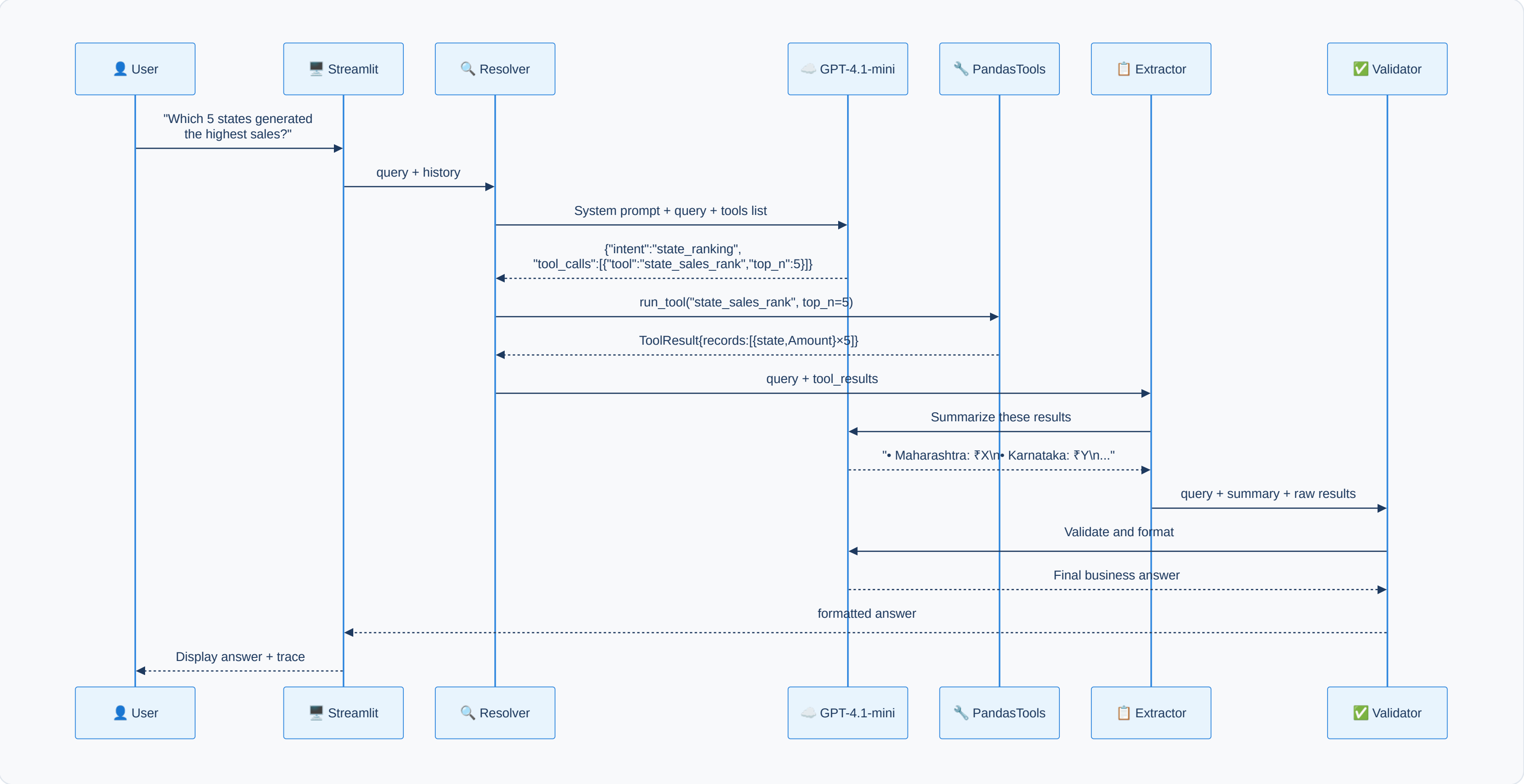
### Text Retrieval (2 tools)

- `expense_statement_text` — Expense IIGF (10K chars)
- `warehouse_comparison_text` — Warehouse chart (10K chars)

SLIDE 6

# Example Query-Response Pipeline

**User asks:** *"Which 5 states generated the highest sales?"*



👤 User
🖥️ Streamlit
🔍 Resolver
☁️ GPT-4.1-mini
🔧 PandasTools
📋 Extractor
✅ Validator

"Which 5 states generated the highest sales?"

query + history

System prompt + query + tools list

{"intent":"state_ranking", "tool_calls":[{"tool":"state_sales_rank","top_n":5}]}

run_tool("state_sales_rank", top_n=5)

ToolResult{records:[{state,Amount}×5]}

query + tool_results

Summarize these results

"• Maharashtra: ₹X\n• Karnataka: ₹Y\n..."

query + summary + raw results

Validate and format

Final business answer

formatted answer

Display answer + trace

## Step-by-Step Breakdown

| Step | Component | Action | Output |
|---|---|---|---|
| 1 | Resolver | Parse intent, select tool | `state_sales_rank(top_n=5)` |
| 2 | Tool Executor | Validate tool name, filter kwargs, run Pandas | 5 records: state + Amount |
| 3 | Extractor | Summarize records into bullets | 4–8 factual bullets with ₹ values |
| 4 | Validator | Cross-validate, format for business user | Natural language answer |

SLIDE 7

# Technology Stack & Dependencies

### Frontend

Streamlit

- Chat UI with message history
- Mode selector (Q&A / Summarization)
- Quick demo queries sidebar
- Execution trace expander

### AI / LLM

CrewAI OpenAI

- 3-agent sequential pipeline
- GPT-4.1-mini (configurable)
- Structured prompts per agent

### Data Processing

Pandas Python 3.10+

- In-memory DataFrame processing
- 18 deterministic analytical tools
- Data cleaning & normalization

### Configuration

python-dotenv

- `.env` for API keys
- Configurable model selection
- Cached initialization via `@st.cache_resource`

SLIDE 8

# Demo Evidence & Q&A Examples

**Video Demo:** A full screen recording of the Streamlit application is included as `Screencast for retail insights assistant.mp4` in the project root, covering all 12 demo queries below.

## Sample Q&A Interactions

| # | Question | Tool(s) Invoked | Answer Type |
|---|---|---|---|
| 1 | How many blouse were sold? | `units_sold_by_category` | Single number |
| 2 | Which category sold the most units? | `category_sales_rank` | Ranked list |
| 3 | Which 5 states generated the highest sales? | `state_sales_rank` | Top-5 ranking |
| 4 | What is the cancellation rate? | `cancellation_rate` | Percentage KPI |
| 5 | Compare B2B and B2C sales. | `b2b_vs_b2c_summary` | Comparison table |
| 6 | Who are the top 5 customers by revenue? | `top_customers_international` | Ranked list |
| 7 | Total international sales in Jun-21? | `international_total_sales` | Aggregate with filter |
| 8 | Which items are low in stock (below 5)? | `low_stock_items` | Filtered list |
| 9 | Stock by category? | `total_stock_by_category` | Grouped aggregation |
| 10 | Price difference: Amazon vs Flipkart May 2022? | `channel_price_comparison_may2022` | Channel comparison |
| 11 | Summarize the Expense IIGF statement. | `expense_statement_text` | Text summary |
| 12 | Warehouse file on fill-rate penalty? | `warehouse_comparison_text` | Text retrieval |

## Interaction Modes

### Q&A Mode

Full 3-agent pipeline: Resolver → Tool Execution → Extractor → Validator. Handles analytical, ranking, comparison, and text-retrieval queries.

### Summarization Mode

One-click summary using top-category, top-state, cancellation rate, and inventory metrics. No LLM call needed — purely deterministic.

SLIDE 9

# Cost & Performance Considerations

## Per-Query Cost Estimate (GPT-4.1-mini)

| Agent | Avg Input Tokens | Avg Output Tokens | Est. Cost |
|---|---|---|---|
| Resolver | ~800 | ~150 | ~$0.0004 |
| Extractor | ~1,200 | ~200 | ~$0.0006 |
| Validator | ~1,500 | ~250 | ~$0.0008 |
| **Total per query** | ~3,500 | ~600 | **~$0.002** |

## Performance Characteristics

### Latency

- Data loading: ~2–3s (one-time, cached)
- Tool execution: <100ms (in-memory Pandas)
- LLM calls: ~1–3s per agent (3 calls total)
- **Total: ~4–8s per query**

### Reliability

- Heuristic fallback covers ~90% of demo queries
- Tool whitelist prevents invalid operations
- Parameter filtering avoids runtime errors
- Graceful error handling with informative messages