

Based on the directive to proceed exclusively with a **Google Antigravity** build, the system architecture has been streamlined. We are removing the third-party app builder (Base44) and replacing it with a custom, agent-architected **Realco OS** (built in Next.js/React via Antigravity). In this architecture, **Antigravity is the factory**, **Databricks is the brain**, and the **custom web application is the body**.

Here is the revised **Realco OS System Blueprint**, organized by logical function.

## Part 1: The 7 Core Architectural Modules (System Logic)

These modules exist as the "Operating System" logic defined in your instructions.md and executed by Antigravity Agents. They govern how data is processed before it ever reaches a user interface.

### 1. Governance & Intelligence Control (The Authority)

- **Role:** The system's "Lead Agent."
- **Function:** Enforces the "No Hallucination" rule and Source Trust Hierarchy (e.g., Land Registry > CRM > Agent Notes). It manages Role-Based Access Control (RBAC) and logs every data mutation to an immutable Audit Log.
- **Key Action:** Blocks any data entry that violates the canonical schema (e.g., prevents creating a "4-bedroom studio").

### 2. Project & Content Discovery (The Harvester)

- **Role:** A read-only observer agent.
- **Function:** Continuously scans connected sources (Databricks, legacy Monday boards, PDF contracts) to discover new project launches or community updates. It registers "Intelligence Nodes" (e.g., "New Phase in Tilal Al Furjan") without altering the core database.

### 3. Canonical Data & Identity Resolution (The Truth Engine)

- **Role:** The deduplication engine.
- **Function:** Resolves multiple data points into single identities.
- **People:** Merges "User A" from WhatsApp and "Lead B" from Monday into one **Contact**.
- **Property:** Unifies "Villa 14" and "Unit 14-B" into one **Canonical Unit ID**.
- **Rule:** History is appended to the record; data is never overwritten.

### 4. Market & Area Intelligence (The Analyst)

- **Role:** The data scientist agent.
- **Function:** Aggregates unit-level data into community-level insights. It calculates "Transaction Velocity," "Price per SqFt Trends," and "Inventory Absorption Rates" by querying Databricks directly via the Model Context Protocol (MCP).

### 5. Seller & Buyer Intelligence (The Matchmaker)

- **Role:** The behavioral analyst.
- **Function:** Scores intent rather than just listing leads.
- **Sellers:** Ranked by "Pricing Gap" (Asking Price vs. Market Value).
- **Buyers:** Ranked by "Urgency Signals" (viewing frequency, budget realism).

### 6. Agent Copilot & Decision Support (The Strategist)

- **Role:** The proactive assistant.

- **Function:** Translates raw data into next-best-action recommendations. Instead of showing a graph, it generates a text card: "*Call Seller X; their property value just crossed their target threshold.*"

## 7. Deployment, Monitoring & Learning (The Optimizer)

- **Role:** The DevOps agent.
- **Function:** Monitors the health of the application and the accuracy of AI predictions. It tracks "Outcome data" (e.g., did the suggested price lead to a sale?) to refine future logic.

# Part 2: Functional Application Modules (The User Interface)

These are the visible modules of the web application that Antigravity will build (using Next.js/React). Users interact with these screens daily.

## A. Inventory Intelligence Module

- **Core Entity:** Unit.
- **UI Focus:** Map-first exploration.
- **Function:** Allows agents to filter properties by Zone, Area, and Project. Features "Truth Cards" for every unit, displaying specific specs, history, and status (Available/Sold/Reserved) directly from the canonical database.

## B. Network & CRM Module

- **Core Entity:** Contact.
- **UI Focus:** "Contact 360" Dashboard.
- **Function:** Manages relationships. A single contact profile displays their timeline across all roles (Buyer, Seller, Investor). Includes "Active Buyer" and "Active Seller" dynamic lists driven by recent activity.

## C. Deals & Finance Module

- **Core Entity:** Deal.
- **UI Focus:** Kanban Pipeline & Financial Tables.
- **Function:** Tracks the lifecycle of revenue. It manages deal flow (Offer → Contract → Transfer), calculates commissions, handles split-payments between agents, and tracks "Net Realco Revenue."

## D. Market Intelligence Module (DLD)

- **Core Entity:** Transaction.
- **UI Focus:** Interactive Charts & Heatmaps.
- **Function:** Visualizes the "Source of Truth" from Databricks. Displays live market trends, "Price vs. Volume" analysis, and community performance comparisons. This module is read-only to ensure market data integrity.

## E. Marketing & Roadshows Module

- **Core Entity:** Campaign.
- **UI Focus:** ROI Dashboard.
- **Function:** Links leads and deals back to their source (e.g., "Roadshow London 2025"). It tracks marketing spend effectiveness and manages asset distribution for international events.

## F. Admin & Connectivity Module

- **Core Entity:** SystemConfig.

- **UI Focus:** Control Panel.
- **Function:** The engine room for the Architect. Allows management of user roles, configuration of Databricks connections (MCP settings), and monitoring of system health/audit logs.

#### G. AI & WhatsApp CRUD Module

- **Core Entity:** Command.
- **UI Focus:** Chat Interface (WhatsApp).
- **Function:** A headless module that allows authorized agents to Add, Edit, or Search inventory using natural language voice notes or text on WhatsApp. It uses the "Governance Module" to validate requests before executing changes.

#### H. The "Pulse" Dashboard

- **Core Entity:** Signal.
- **UI Focus:** Daily Executive Briefing.
- **Function:** The homepage for every user. It aggregates critical alerts, daily tasks, and "Deferred Revenue Signals" into a clean, prioritized list to start the day.

### Part 3: The Data Flow (Antigravity Architecture)

1. **Ingestion:** Agents utilize **MCP (Model Context Protocol)** to read live data from Databricks and legacy exports from Monday/Pixxi without moving the data unnecessarily.
2. **Processing:** The **Observer Agent** (running on Google Cloud Run) processes incoming signals against the **Governance Rules**.
3. **Storage:** Operational data (CRM, Active Inventory) is stored in a **Postgres/Firebase** database managed by the app. Historical market data remains in **Databricks**, queried on-demand.
4. **Interface:** The **Next.js Frontend** renders the data, serving as the single pane of glass for all modules 1-4.