🏗️ System Structure & Component Map

This document explains the exact organization of the **Operations Copilot** project. Use this as a guide for your course presentation to explain where every feature lives and how they interact.

📂 1. The Core Infrastructure (`/core`)

The "brain" of the application. It handles the UI, User Management, and the primary AI Orchestration.

- **`crews/knowledge_crew/`**: Contains the **CrewAI** logic.
    - `agents.yaml`: Defines the personalities (Expert Technician, Safety Officer).
    - `tasks.yaml`: Defines what the agents must do (Retrieve info, summarize).
    - `crew.py`: The assembly line where agents and tasks are combined.
- **`flows/knowledge_retrieval_flow/`**: Manages the "State". It decides the step-by-step logic of how a question is answered.
- **`tools/`**: The bridge between AI and the real world.
    - `rag_tool.py`: Allows the AI to "read" your database documents.
    - `vision_llm.py`: Allows the AI to "see" uploaded images.
- **`templates/core/`**: The modern frontend.
    - `base.html`: The master layout (Sidebar, Theme, Fonts).
    - `dashboard.html`: The real-time operations center.
    - `knowledge_query.html`: The ChatGPT-like interface.

📂 2. Knowledge Base Module (`/knowledge_base`)

Handles the "Memory" of the system.

- **`models.py`**:
    - `Document`: Stores uploaded PDFs and DOCX files.
    - `KnowledgeQuery`: Logs every question asked and the AI's answer.
- **`services/rag_service.py`**: The logic that takes a user's question, finds the relevant text in a PDF, and sends it to the AI.
- **`views.py`**: API endpoints for uploading and searching documents.

📂 3. Field Operations Module (`/field_operations`)

Handles the "Vision" and "Analysis" of electrical equipment.

- **`models.py`**:
    - `FieldInspection`: Stores the equipment images and location data.

    - `InspectionAnalysis`: Stores the AI's results (Risk Level, Actions).
- **`services/image_analysis_service.py`**: The connection to the Vision LLM that identifies rust, leaks, or damage.
- **`services/agent_orchestration_service.py`**: Coordinates multiple agents to double-check high-risk findings.

📂 4. Audit & Integrity Module (`/audit`)

Ensures everything is tracked and transparent.

- **`models.py`**: Tracks system health and critical alerts.
- **`migrations/`**: Automatically generated files that build your database tables.

📂 5. Project Configuration (`/project`)

The "Nervous System" that connects all modules together.

- **`settings.py`**: Global configuration (Database, AI API Keys, Theme settings).
- **`urls.py`**: The routing map that directs users to the correct page or API.

🔗 How They Connect (The "Flow" for Your Presentation)

1. **Request**: A user asks a question in `core/templates/knowledge_query.html`.
2. **Logic**: `core/views/` triggers the `knowledge_retrieval_flow`.
3. **Search**: The Flow uses the `RAGTool` in `core/tools/` to talk to `knowledge_base/services/rag_service.py`.
4. **Database**: The Service looks up data in the `Document` model in `knowledge_base/models.py`.
5. **Response**: The AI synthesizes the answer and displays it back on the UI using **Marked.js** for pretty formatting.

🛠️ Essential Project Files

- **`manage.py`**: The command-center used to run migrations or start the server.
- **`requirements.txt`**: The list of all "libraries" (Django, CrewAI, LangChain) the project needs to run.
- **`.env`**: (Hidden) Stores your secret API keys safely.
- **`start_server.sh`**: A quick-launch script to boot the system.