iFarmAssist: Technical Architecture & Tech Stack

This document provides a detailed breakdown of the technologies used in the iFarmAssist project. The system is designed as a modern, scalable, AI-powered application for precision agriculture.

1. Mobile Frontend (The Farmer's Interface)

Framework: React Native (via Expo)

We chose React Native to build a truly native mobile experience using JavaScript. Expo was used to accelerate development and testing.

Technology	Purpose
React Native	Cross-platform mobile UI framework.
Expo	Build tool and runtime for easy testing.
Axios	HTTP Client to communicate with the Backend API.
React Navigation	Handling screen transitions (Login -> Home).
Lucide React Native	Modern, lightweight icon set.

2. Backend API (The Brain)

Framework: FastAPI (Python)

FastAPI was selected for its high performance and native support for asynchronous programming, which is crucial for handling AI requests without blocking.

Technology	Purpose
Python 3.10+	Primary programming language.
FastAPI	High-performance web framework.
Uvicorn	ASGI Server to run the application.
Pydantic	Data validation and settings management.
Python-Dotenv	Managing environment variables securely.

3. Artificial Intelligence Engine

Core Feature: RAG (Retrieval Augmented Generation)

The system combines Generative AI with a custom Knowledge Base to give accurate, context-aware answers.

Technology	Purpose
Google Gemini 2.0	Main LLM for reasoning and text generation.
LangChain / Direct SDK	Orchestrating the AI workflow.

ChromaDB	Vector Database to store agricultural manuals.
Sentence-Transformers	Creating embeddings (all-MiniLM-L6-v2) for search.

4. Data Storage & External APIs

Technology	Description
SQLite (Dev)	Lightweight, file-based SQL database for local testing.
PostgreSQL (Prod)	Robust relational database for user data (Architecture ready).
OpenWeatherMap API	Real-time weather data fetching for context-awareness.

Generated automatically by iFarmAssist AI Agent.