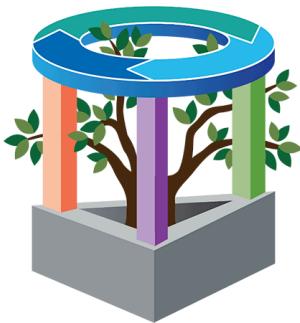


# Drone Based Mapping and Ground Based Robotic Precision Spray System for Field Crops

## Project Requirements and Specifications

AgAID



# AgAID

AI Institute for Transforming  
Workforce & Decision Support



Gil Rezin

Logan Sutton

[October 20, 2025]

<b>I. Introduction</b>	3
<b>II. System Overview</b>	3
<b>III. Architecture Design</b>	5
<b>III.1. Overview</b>	5
<b>III.2. Subsystem Decomposition</b>	6
<b><i>III.2.I Image Stitching Module</i></b>	6
<b><i>III.2.II Prescription Generation Module</i></b>	7
<b><i>III.2.III. Path Planning and Robot Communication Module</i></b>	9
<b><i>III.2.IV. Visualization and Control Module</i></b>	10
<b><i>III.2.V. Data Management and Reporting Module</i></b>	12
<b>IV. Data Design</b>	14
<b>V. User Interface Design</b>	15

# Solution Approach

## I. Introduction

The purpose of this document is to describe the solution approach and architectural design of the *Drone Based Mapping and Ground Based Robotic Precision Spray System for Field Crops (DBD)*. It provides a detailed explanation of the system's structure, key components, and the rationale behind major design decisions. This document serves as a guide for developers, project stakeholders, and faculty reviewers involved in the AgAID project, ensuring a shared understanding of the system's functionality and implementation strategy. It also provides a foundation for future revisions, testing, and deployment phases.

This version of the document represents an update to the initial *Project Requirements and Specifications* submission. While the earlier version focused primarily on defining functional and non-functional requirements, this revision expands upon those concepts by introducing a detailed architectural layout, subsystem decomposition, and design methodology. The updated content includes finalized module descriptions, clarified API interactions, and an expanded overview of data flow and user interface design.

The DBD project aims to create an integrated, offline-capable precision agriculture system that combines drone-based imaging, artificial intelligence, and autonomous ground robotics. Using aerial imagery collected by drones, the system automatically generates crop health maps, determines variable-rate pesticide prescriptions, and communicates optimal spray paths to a robotic ground sprayer. The overarching goal of this project is to reduce pesticide use and labor costs while improving efficiency, accuracy, and sustainability in field crop management, particularly in rural environments where internet connectivity is limited.

## II. System Overview

The DBD system is built as a fully offline, modular desktop application, ensuring full operability in rural areas without internet access. The system is designed modularly, utilizing various pieces of software in unison to provide a fluid and intuitive interface for the farmer end users. The modules are divided based on our pipelined flow of data, and also adhere to our system goals, which are listed below.

The primary system goals are to:

- Reduce pesticide and labor costs through automated, variable-rate spraying.
- Generate optimal application routes that avoid hazards such as steep hills.
- Operate autonomously without reliance on external cloud infrastructure.
- Maintain modularity and scalability for integration with other systems (e.g., John Deere navigation).

Each module operates independently and can be upgraded or replaced as technology advances. The five core modules are:

- 1. Image Stitching Module**

This backend module handles the import and validation of drone imagery and also handles orthophoto creation. The orthophotos are generated using a locally run image of NodeODM via Docker. This provides a free and easy-to-use image stitching functionality.

- 2. Prescription Generation Module**

This backend module produces AI-based dosage maps (field maps) from stitched imagery. These maps are produced using an R orthophoto analysis library called FIELDImageR - by calculating and then analyzing the per-pixel NDRE and NDVI indices with reinforcement learning, an individual plant's health can be determined and a pesticide amount is assigned.

- 3. Path Planning and Robot Communication Module**

This backend module produces a series of geographic waypoints for the robotic sprayer to follow and execute actions along. FarmNG provides the utilities to send these instruction files to the sprayer robot. It is not yet determined whether the sprayer robot has the functionality to automatically calculate a path given a field map, so the exact implementation is still not certain.

- 4. Visualization and Control Module**

This frontend module provides the user interface for visualization and user input. The interface is managed using React for a fast and simple locally hosted application. React provides all the necessary tools for providing visualization and image upload functionality, and is also simple to implement with AI tools. A simple user interface is ideal, as the user is intended to only have to upload drone imagery, and the entire system will autonomously complete its pipelined tasks without additional instruction.

- 5. Data Management and Reporting Module**

This backend module stores, logs, and reports field operations for compliance and review. We will be using a determined local file structure to accomplish this task, as we deemed using a locally run database was unnecessary and may cause slower operations with the transferring of images.

All backend modules are served from a common API layer, which distributes the necessary information to each responsible location. Additionally, the API is set up to serve the frontend React application via a series of robust endpoints. This provides a scalable and maintainable system that works entirely offline. The backend API layer is implemented using Python, specifically FastAPI.

All system computation will occur locally on a workstation meeting the following minimum requirements:

- Intel i7 CPU (or equivalent)
- NVIDIA RTX 3060 GPU (or equivalent)

These specifications ensure the system performs complex computations—such as stitching, machine learning inference, and path optimization—with acceptable time limits.

## III. Architecture Design

### III.1. Overview

The DBD system architecture is designed around modularity, local computation, and offline autonomy. Its structure follows a layered, service-oriented model in which all modules interact through a unified local API. This approach isolates responsibilities, simplifies maintenance, and ensures that upgrades to individual components, such as new AI models or imaging pipelines, can be integrated with minimal disruption to the broader system.

At the center of the architecture lies the FastAPI-based backend layer, which acts as a communication hub between processing modules and the user interface. Each backend service registers its own endpoints through this API, enabling seamless data flow between the front-end React application and the offline computational modules. This design provides a consistent, well-defined interface for data exchange while preserving the independence of each processing stage.

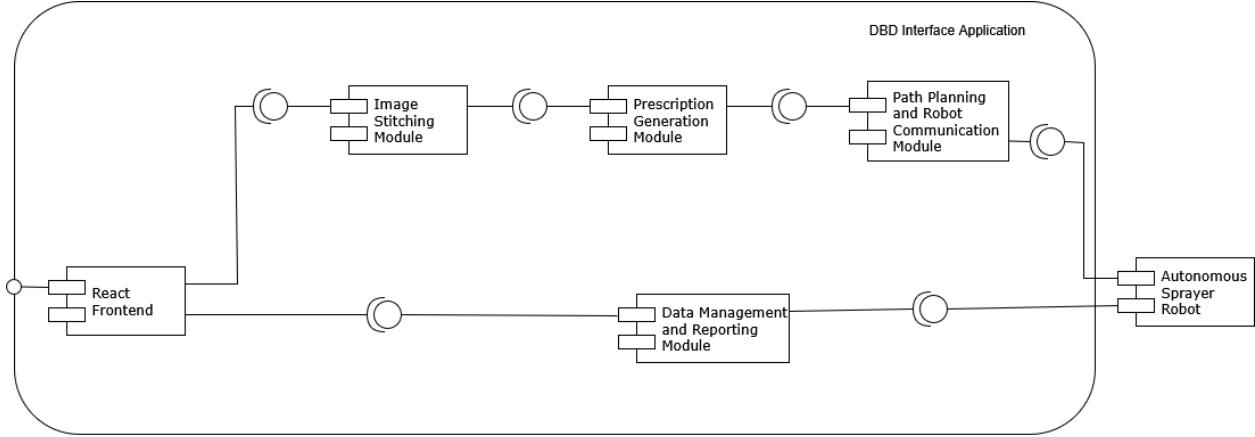
The decision to adopt an entirely local, offline architecture was driven by the system's target environment: rural agricultural areas with unreliable or nonexistent internet connectivity. By running all modules on a single workstation, the system ensures deterministic performance, data privacy, and full operability in the field without cloud dependencies. Additionally, this eliminates recurring costs associated with remote hosting or storage while maintaining high processing throughput through GPU-accelerated local computation.

The pipeline architecture reflects the natural data flow of the system, from raw imagery ingestion to output visualization, while remaining flexible enough to accommodate new components in the future (for instance, automated calibration or external database integration). The modular separation also enables independent development and testing of each component, supporting long-term scalability.

From a deployment standpoint, all computational modules are containerized where applicable (e.g., image processing within NodeODM's Docker instance) to ensure portability and reproducibility across different hardware setups. The front-end React interface is served locally through the same FastAPI framework, creating a cohesive, self-contained desktop environment that requires no external hosting or network services.

Figure 1 illustrates the overall system architecture, showing the interaction between the frontend interface, the centralized API layer, and the offline backend modules within the local computing environment.

**Figure 1:** Component Diagram



### **Workflow Summary:**

1. Drone captures raw imagery of the field.
2. Images are uploaded and stitched into a georeferenced field map.
3. The field map is analyzed to generate a dosage prescription map.
4. The system computes an optimized path for the robot sprayer.
5. The prescription and path are transmitted to the robot for execution.
6. Operational data are collected and analyzed to evaluate spraying effectiveness.

## **III.2. Subsystem Decomposition**

### **III.2.1 Image Stitching Module**

#### **Description:**

The Image Stitching module handles the ingestion, validation, and stitching of aerial drone imagery. The stitched output serves as the foundational data layer for subsequent modules. To achieve this task, we utilize NodeODM, a free and open-source image stitching software that can run locally and offline. NodeODM is used to generate the orthophoto and relevant data for the Prescription Generation Module.

#### **Concepts and Algorithms:**

The Image Stitching module handles several tasks. First, the module inputs several drone multispectral images that will all be combined to form a single image. Each image will be validated for correct file formats and similarities to ensure that every file can be merged into a single image representing the entire field. This process is done locally, as our program must be used in remote farms without access to high-speed internet. The module also provides a method for retrieving the processed data.

#### **Services Provided**

Service Name	Provided To	Description
Put Drone Imagery	Backend API	Receives raw drone images captured from farmer's field and prepares them for processing within the NodeODM environment.
Validate Imagery Input	Backend API	Ensures that uploaded drone imagery meets the required format, resolution, and metadata standards before processing.
Generate Orthophoto	Backend API	Uses the NodeODM engine to stitch individual drone images into a single high-resolution orthophoto (georeferenced map).
Get Processed Data	Backend API	Packages and returns processed outputs (orthophotos, DSMs, and metadata) to the backend API for downstream processing.

### Services Required

Service Name	Provided By	Description
Drone Imagery Upload	Farmer's Drone / Field Interface	Provides the raw imagery data required for processing.
Prescription Generation Module	Backend API	Consumes the orthophoto and derived outputs from NodeODM to produce AI-based dosage or field maps.

## III.2.II Prescription Generation Module

### Description:

This module analyzes the field map to generate a pesticide prescription that defines dosage levels for specific areas of the field. The prescription is generated utilizing machine learning techniques, specifically reinforcement learning, to identify proper spraying locations and optimal pesticide application amounts.

### Concepts and Algorithms:

The Prescription Generation module utilizes a machine-learning classification model to predict dosage of pesticide per subdivision of the field map. This is done through NDRE and NDVI scores, metrics that measure plant health from light reflectivity, analysis done via the R library FIELDimageR. Furthermore, deeper analysis can be done via reinforcement learning. By

analyzing the maps generated by FIELDImageR and historical spraying performance data, machine learning enables the optimal dosage of pesticide to be chosen per subdivision.

## Services Provided

Service Name	Provided To	Description
Generate Vegetation Indices	Backend API	Uses the R library FIELDImageR to calculate NDVI and NDRE scores from orthophotos, providing quantitative measures of plant health across the field.
Analyze Field Subdivisions	Backend API	Segments the processed orthophoto into smaller field subdivisions to enable localized analysis and dosage prediction.
Predict Optimal Dosage	Backend API	Utilizes a trained machine learning classification model to predict the optimal pesticide dosage for each field subdivision based on vegetation indices and historical data.
Reinforcement Learning Optimization	Backend API	Applies reinforcement learning techniques to refine dosage recommendations over time, using performance feedback from previous spraying outcomes.
Generate Prescription Map	Backend API	Produces a spatially-referenced prescription (dosage) map that encodes the optimal pesticide levels per subdivision, ready for integration with autonomous spraying systems or robotic applicators.
Export Prescription Data	Path Planning & Robot Communication Module	Formats and sends the generated prescription map and dosage data for use in path planning and automated field execution.

## Services Required

Service Name	Provided By	Description
Orthophoto and Vegetation Imagery	NodeODM Module	Provides high-resolution orthophotos and related imagery data used for vegetation analysis.

Historical Spraying Data	Farm Database / API	Supplies prior spraying results and field performance metrics used to train and improve the machine learning dosage prediction model.
Reinforcement Learning Feedback	Robot Communication Module	Returns real-world performance data from implemented prescriptions, enabling continuous learning and dosage optimization.

### III.2.III. Path Planning and Robot Communication Module

#### Description:

This module calculates an optimal route for the ground-based sprayer robot based on the field prescription and terrain constraints, and communicates the resulting path and prescription data to the robot for execution. The results from the Image Stitching module (an orthophoto) are fed in as input. The orthophoto provides geo data that our path planning software can utilize to create a waypoint script for the sprayer robot.

#### Concepts and Algorithms:

The main objective is to calculate the optimal route from an input field Shapefile. Pathfinding is utilized to minimize overlap and travel distance, while maximizing field coverage. This is primarily done through field heading selection, as field coverage is mostly done through straight lines, with the exception of obstacle avoidance, such as telephone poles and difficult terrain. This module communicates directly with the robot's controller to execute the actions generated by this module.

#### Services Provided

Service Name	Provided To	Description
Import Orthophoto Data	Backend API	Receives georeferenced orthophotos and associated field shapefiles from the Image Stitching (NodeODM) module for spatial analysis and route generation.
Generate Field Coverage Path	Backend API	Computes an optimal traversal path for the sprayer robot based on the field prescription map, terrain constraints, and field geometry, ensuring complete coverage with minimal overlap.
Obstacle Detection and Avoidance	Backend API	Identifies obstacles such as poles, uneven terrain, or restricted zones using map data and adjusts the robot's route accordingly to ensure safe navigation.

Generate Waypoint Script	Robot Sprayer	Converts the optimized path into a series of GPS waypoints and commands interpretable by the robot's control system for automated execution.
Communicate Prescription and Route Data	Robot Sprayer	Sends the final prescription and waypoint data to the robot's onboard controller, initiating automated field spraying operations.
Monitor Execution Feedback	Backend API	Receives updates from the robot regarding task completion, movement accuracy, and spray progress, enabling operational monitoring and future optimization.

### Services Required

Service Name	Provided By	Description
Orthophoto and Field Map	Image Stitching (NodeODM) Module	Provides the high-resolution orthophoto and shapefile data necessary for spatial route calculation.
Prescription Map	Prescription Generation Module	Supplies the dosage map that determines where and how much pesticide to apply along the generated path.
Terrain and Obstacle Data	Farm Database / Onboard Sensors	Provides information about terrain elevations, obstacles, or restricted zones that must be considered during route planning.
Robot Control Interface	Sprayer Robot System	Receives the generated waypoint scripts and prescription data for autonomous execution and returns operational feedback to the module.

### III.2.IV. Visualization and Control Module

#### Description:

This module acts as the primary user interface for interacting with the DBD system. It provides visualization of stitched maps, prescriptions, and planned spraying paths. Additionally, this module provides the interface for uploading drone imagery and provides slider options for adjusting various parameters correlated to the desired field map. Farmers can easily view their uploads and get real-time status updates based on the corresponding step in the pipeline that is being completed.

### **Concepts and Algorithms:**

This module consists of everything the user interacts with. The UI utilizes component-based architecture with TypeScript for type safety, implementing a state management pattern through React hooks (useState, useEffect, useCallback) to handle complex application state across multiple views including upload, processing, gallery, field maps, and pesticide prescriptions. The file upload system incorporates advanced drag-and-drop functionality using the react-dropzone library, implementing file validation algorithms that check file types (JPEG, PNG, TIFF), size limits (50MB), and generate unique identifiers for each upload session. The real-time processing monitoring employs a sophisticated polling algorithm that automatically queries the backend every 3 seconds to track task status updates, implementing state synchronization between frontend and backend systems through localStorage persistence and custom event dispatching. The application features responsive design algorithms using Tailwind CSS with breakpoint-based layouts, progressive enhancement patterns for offline functionality, and error handling mechanisms with retry logic and user feedback systems.

### **Services Provided**

<b>Service Name</b>	<b>Provided To</b>	<b>Description</b>
Upload Drone Imagery	Image Upload Module	Provides the user interface for farmers to upload raw drone imagery, including drag-and-drop functionality, file validation (type, size, and format), and unique upload session handling.
Visualize Orthophotos and Field Maps	End Users (Farmers)	Displays high-resolution stitched maps generated by NodeODM, overlaid with prescription data and planned spraying paths for intuitive field visualization.
Adjust Field Map Parameters	Backend API	Enables users to modify adjustable parameters (e.g., vegetation thresholds or dosage sensitivity) through interactive sliders and UI controls, sending updated parameters to backend modules for reprocessing.
Monitor Processing Pipeline	End Users (Farmers)	Provides real-time status updates of each stage in the processing pipeline (image stitching, prescription generation, path planning), using a polling algorithm that queries backend services every 3 seconds.
Display Prescription and Path Plans	End Users (Farmers)	Renders the AI-generated prescription maps and optimized spraying routes, allowing users to review dosage distributions and coverage before execution.
Manage Application State	Internal Frontend System	Implements advanced state management using React hooks to synchronize data across multiple views (upload, processing, gallery, map, prescriptions) while maintaining local persistence.

Provide Responsive and Reliable UI	End Users (Farmers)	Ensures seamless and adaptive user experiences across devices through responsive design (Tailwind CSS), offline support, and robust error handling with retry and user feedback systems.
------------------------------------	---------------------	--

### Services Required

Service Name	Provided By	Description
Orthophoto and Processed Imagery	NodeODM Module	Supplies the stitched and georeferenced map data displayed within the visualization interface.
Prescription Map Data	Prescription Generation Module	Provides dosage maps and plant health metrics to visualize AI-generated treatment recommendations.
Path Planning Output	Path Planning and Communication Module	Supplies the optimized waypoint routes and spraying paths to be rendered on the user interface.
Processing Status Data	Backend API	Returns real-time task updates for synchronization with the frontend's status monitoring and visualization system.

### III.2.V. Data Management and Reporting Module

#### Description:

This module manages the persistence of operational data, system logs, and generated reports. It ensures reproducibility, accountability, and long-term analysis capabilities. The storage of this data will be handled using a systematic local file structure for ease-of-access and fast operations.

#### Concepts and Algorithms:

The Data Management and Reporting Module handles the data generated by the program through the local file system. Categories of file are sorted by folder within the filesystem, such as logs, reports, and generated prescription maps. When logs are generated by the robot, they are sent wirelessly to this module for storage within the logs folder. Logs contain timestamps, dosage, and GPS information for debugging purposes. When the robot finishes its spraying, a report is generated by this module regarding spray performance and compliance. When requested by the Visualization and Control module, these files become visible to the user.

## Services Provided

Service Name	Provided To	Description
Store Operational Data	Backend System	Manages the structured storage of operational data, including orthophotos, prescription maps, and path plans, within an organized local file directory for quick access and reproducibility.
Log Robot Activity	Path Planning & Robot Communication Module	Receives and stores logs transmitted wirelessly from the robot, including timestamped dosage, GPS coordinates, and performance metrics, enabling detailed traceability and debugging.
Generate Spray Reports	Visualization & Control Module	Automatically compiles comprehensive post-operation reports detailing spray coverage, dosage accuracy, and compliance information once the robot completes its spraying tasks.
Manage File System Organization	Internal Backend System	Maintains a systematic folder hierarchy (e.g., /logs, /reports, /maps) to ensure consistent file organization, fast I/O operations, and simplified long-term maintenance.
Provide Report Access	Visualization & Control Module	Supplies user-facing modules with access to stored reports, logs, and historical data upon request, allowing users to view and download past operation summaries.

## Services Required

Service Name	Provided By	Description
Operation Logs	Path Planning & Robot Communication Module	Supplies real-time or post-operation logs detailing robot activity, GPS traces, and dosage data for storage and later analysis.
Prescription Maps and Results	Prescription Generation Module	Provides final dosage maps and related analytical outputs for recordkeeping and inclusion in generated reports.
User Report Requests	Visualization & Control Module	Triggers the retrieval and display of reports, logs, and stored data for user viewing and download within the UI.

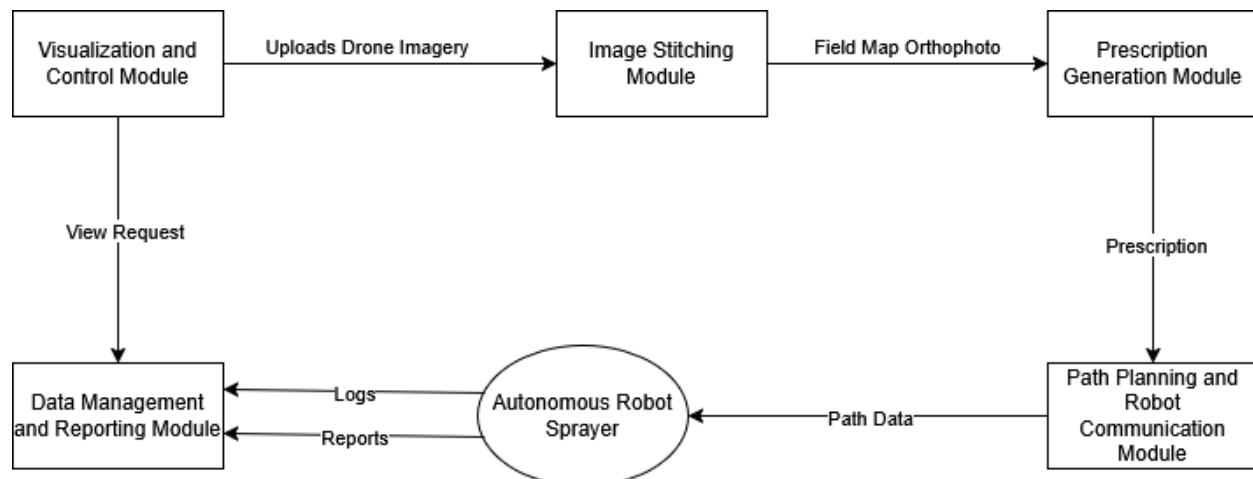
## IV. Data Design

The DBD system handles multiple data types throughout its workflow, including:

Data Type	Description	Stored In	Example Use
Drone Imagery	Raw aerial photos captured from the field	Local file system, .tif files	Input to stitching process
Field Map Orthophoto	Stitched, georeferenced image of the field	Local or cached storage	Input to prescription generation
Prescription	Dosage map per field region	Local Shapefile	Input to path planning
Path Data	Coordinates for optimized traversal	Local data store	Uploaded to robot
Logs	Application and spraying data	Local log files	Compliance and debugging
Reports	Post-spray summaries	Local file system	User and regulatory review

Figure 2 illustrates the conceptual data flow between the modules.

**Figure 2: Data Flow Diagram for DBD System Components**

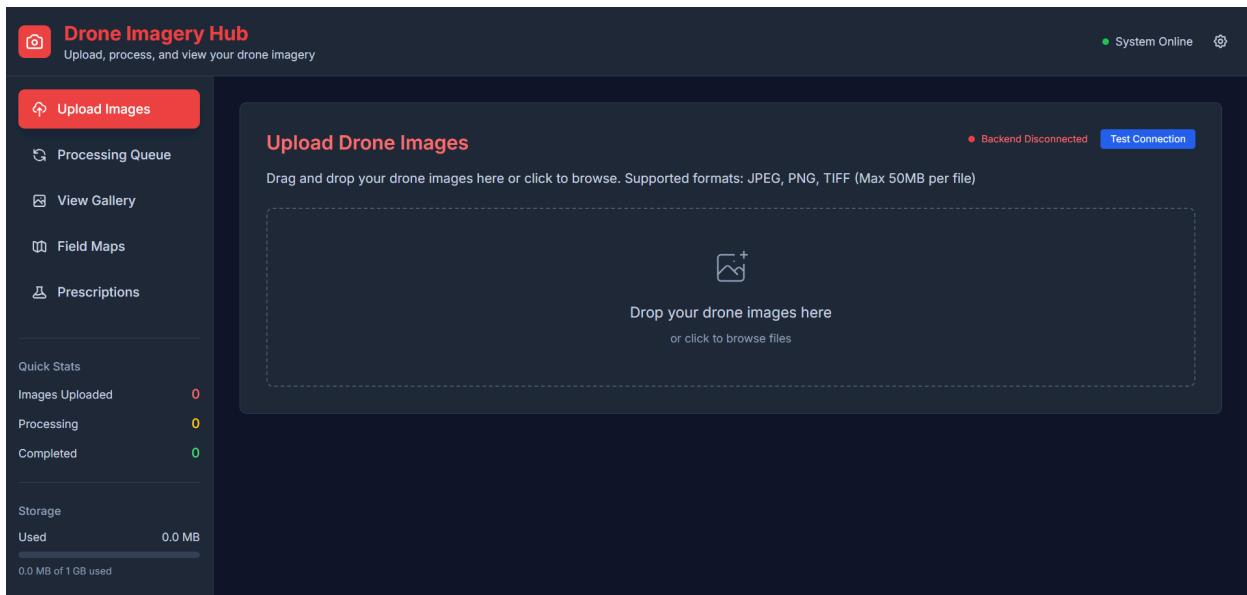


## V. User Interface Design

The user interface provides an accessible control panel that integrates all stages of the spraying workflow: imagery upload, map generation, path preview, and operation reporting. The design emphasizes simplicity for non-technical users while maintaining transparency in system behavior.

We designed the Drone Imagery Hub frontend with a sophisticated dark theme aesthetic that prioritizes both functionality and visual appeal for agricultural professionals. The interface employs a red and black color scheme (#dc2626 primary red on #0f172a dark backgrounds) to create a modern, professional appearance that conveys the technical nature of drone imagery processing while maintaining excellent readability and contrast. The application utilizes Inter font family throughout for its clean, modern typography that ensures optimal legibility across all components and screen sizes. The component-based architecture implements a sidebar navigation system with five main sections (Upload Images, Processing Queue, View Gallery, Field Maps, and Pesticide Prescriptions) that provides intuitive workflow progression from data input to final agricultural outputs.

**Figure 3: Upload Drone Images UI Page (WIP)**



The responsive design employs Tailwind CSS utility classes with breakpoint-based layouts that adapt seamlessly from mobile devices (if a mobile app is wanted in the future) to large desktop displays, ensuring consistent user experience across all platforms. The upload interface features an advanced drag-and-drop zone with visual feedback states, progress tracking bars, and file queue management that provides real-time status updates and error handling. The processing view implements real-time polling algorithms with status indicators and progress visualization that keep users informed of backend processing states. The gallery and field maps sections utilize grid and list view toggles with thumbnail previews and metadata displays that

help users quickly identify and select relevant agricultural data. The pesticide prescriptions module incorporates filtering and sorting algorithms with robot compatibility indicators and cost estimation displays that support decision-making workflows. All interface elements feature smooth transitions (200ms duration), hover effects, and glass morphism styling with subtle borders and rounded corners that create a cohesive, modern aesthetic while maintaining the professional, technical character essential for agricultural drone operations.