PES University

Decision Log

NAME : Sandhya P

SRN : PES1UG22CS523

1. Project Overview

This project implements an AI-assisted drone operations coordination system for Skylark Drones. The system automates pilot and drone assignment, detects operational conflicts, and synchronizes all updates with Google Sheets in real time. A simple web dashboard allows users to interact with the system visually, while REST APIs support programmatic access.

The goal was to reduce manual coordination effort, minimize scheduling conflicts, and provide an intelligent decision-support tool for operations managers.

2. Key Assumptions

To make the system realistic yet implementable within project scope, the following assumptions were made:

1. **Google Sheets as Operational Database**
   Google Sheets is treated as the live source of truth for pilots, drones, and missions. This allows transparency and easy manual verification without building a separate database.
2. **Single Active Assignment per Resource**
   A pilot or drone can only be assigned to one mission at a time.
3. **Skills and Certifications Are Mandatory**
   A pilot must meet all required skills and certifications before assignment. Safety rules cannot be overridden.
4. **Location Represents Readiness**
   Pilot and drone locations are assumed to be their current operational base.
5. **Mission Priority Exists**
   Missions may have different urgency levels. Higher-priority missions influence reassignment logic.

3. System Architecture Decisions

Backend Framework

Spring Boot (Java) was selected for:

- Robust REST API support
- Dependency injection and modular design
- Production-ready deployment capability

Data Layer

The system uses the **Google Sheets API** instead of a database.

Trade-off:

- ✓ Easy integration and visibility

Layered Design

The backend follows a structured architecture:

- **Controllers** – Handle API requests
- **Services** – Business logic (assignment, conflicts)
- **Repositories** – Google Sheets read/write

This separation improves maintainability and testability.

4. Assignment Strategy

The assignment process follows two stages:

Stage 1 – Standard Assignment

The system searches for a pilot-drone pair where:

- Pilot status = Available
- Drone status = Available
- Skills match mission requirements
- Certifications are valid
- Both are in the mission location
- Drone is not under maintenance

If a valid pair is found, the system updates:

- Pilot → Status = Assigned, Current Assignment = Mission ID
- Drone → Status = Deployed, Current Assignment = Mission ID

These changes sync immediately to Google Sheets.

Stage 2 – Urgent Reassignment

If no standard match exists and the mission is high-priority:

The system attempts to:

- Reassign a pilot from a lower-priority mission
- Ensure the pilot still meets skill and certification requirements
- Pair with an available drone at the mission location

Urgency can override availability but **cannot override safety constraints**.

5. Conflict Detection

The system proactively detects:

Conflict Type	Description
Skill mismatch	Pilot lacks required skill
Certification mismatch	Missing regulatory certification
Location mismatch	Pilot/drone not in mission city
Drone maintenance	Drone unavailable due to maintenance
Double booking	Resource already assigned to another mission

Conflicts are returned in readable messages to help decision-making.

6. Google Sheets Synchronization

The system implements **2-way synchronization**:

Read Operations

- Pilot roster
- Drone fleet
- Missions

Write Operations

- Pilot status updates
- Pilot assignment updates
- Drone status updates
- Drone assignment updates

Only specific cells are modified to prevent accidental data loss.

7. Conversational and Visual Interface

Conversational Layer

A lightweight `/agent` endpoint interprets simple natural-language-like commands such as:

- “show pilots”
- “assign mission PRJ001”

This simulates an AI coordinator interface.

Visual Dashboard

A static HTML dashboard was added and hosted via Spring Boot. It provides:

- Buttons to view pilots, drones, and missions
- Mission assignment input
- AI agent query box

This improves usability for non-technical users.

8. Trade-offs Made

Decision	Benefit	Limitation
Google Sheets instead of DB	Simple, transparent	Not scalable for large fleets
Rule-based AI	Reliable and deterministic	Not true machine learning
REST + simple dashboard	Easy to test and deploy	Limited UI sophistication
No time-based scheduling engine	Simpler design	Cannot detect date overlaps deeply

9. What I Would Improve With More Time

1. Add time-based mission scheduling overlap detection
2. Introduce a real database (PostgreSQL) for scalability
3. Use LLM-powered natural language understanding
4. Add authentication and user roles
5. Build a richer frontend dashboard with maps and status indicators

10. Interpretation of "Urgent Reassignments"

Urgent reassignments were implemented as:

A controlled override mechanism allowing reassignment from lower-priority missions while still enforcing safety requirements such as skills and certifications.This balances responsiveness with operational safety.

11. Final Outcome

The final system successfully:

- Automates pilot and drone coordination
- Detects conflicts before assignment
- Handles urgent mission scenarios
- Synchronizes operational data in real time
- Provides both API and dashboard interfaces
- Runs as a live hosted cloud application

This demonstrates a practical AI-assisted operations coordination tool suitable for real-world logistics management.