# AeroXplorer Deliverables

Client Meeting

May 6th, 2025

Business Instructional Facility
University of Illinois Urbana-Champaign

CUBE CONSULTING

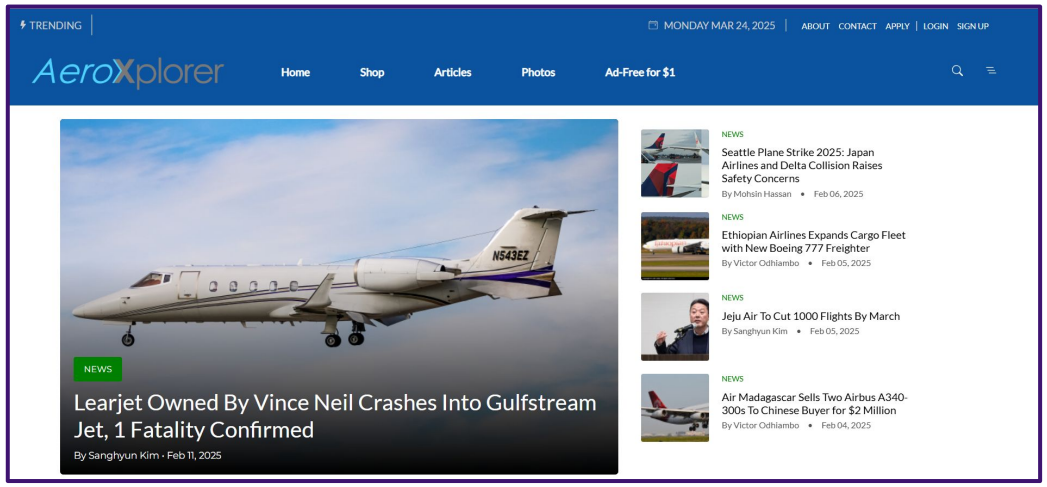
Champaign-Urbana Business & Engineering Consulting

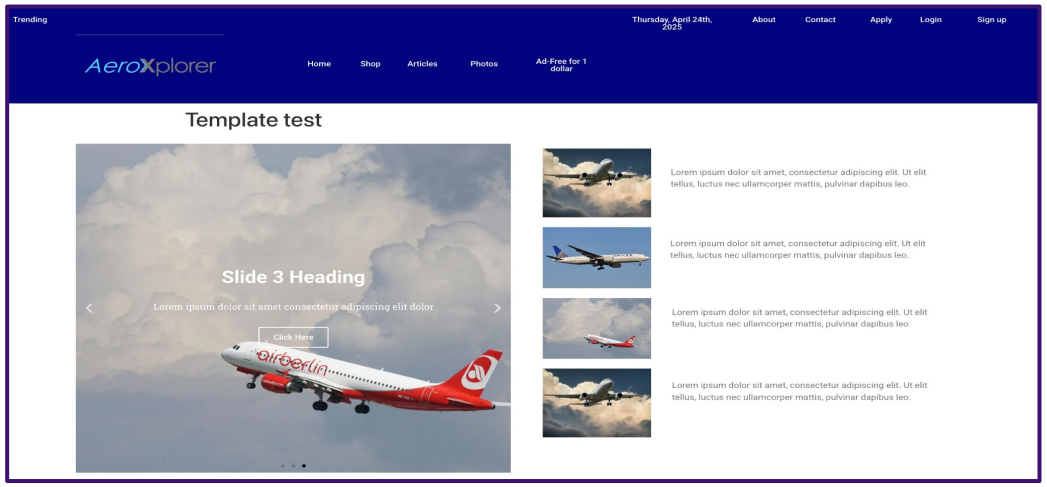
Junior Enterprise United States of America

ILLINOIS

# Current Wordpress Progress

CUBE CONSULTING




## Completed Components

- Replicated the static structure of the homepage
- Carousel slider integrated for featured articles
- “More News” section implemented with dynamic layout
- Added placeholders for social media and interesting reads

## Next Steps

- Implement article database integration (e.g. WordPress posts)
- Add user interaction elements (likes, shares, comments)
- Finalize responsive design for mobile/tablet views

Strong progress has been made replicating key visual elements of the site and focus is now shifting to implementing dynamic features.

CUBE Consulting

# AI Integration Overview & Image QA Filtering

# AI Integration Overview

AWS Rekognition

- Airlines
- Flight State
- Time of Day
- Weather

Tag Logic

| | Filename | Airline_1 | Airline_2 | FlightState | TimeOfDay |
|---|---|---|---|---|---|
| 2 | imagesCUBE/united_morning_takeoff.jpg | United Airlines | | Takeoff | Morning |
| 3 | imagesCUBE/southwest_dusk_landing.jpg | Southwest | | Landing | Dusk |
| 4 | imagesCUBE/jetblue_dawn_taxiing.jpg | JetBlue | | Taxiing | Dawn |
| 5 | imagesCUBE/alaska_day_cruising.jpg | Alaska Airlines | | Cruising | Day |
| 6 | imagesCUBE/lufthansa_night_approach.jpg | Lufthansa | | Approach | Night |
| 7 | imagesCUBE/emirates_afternoon_parking.jpg | Emirates | | Parking | Afternoon |
| 8 | imagesCUBE/british_airways_evening_boarding.jpg | British Airways | | Boarding | Evening |
| 9 | imagesCUBE/delta_united_day_maintenance.jpg | Delta | United Airlines | Maintenance | Day |
| 10 | imagesCUBE/american_qatar_night_refueling.jpg | American Airlines | Qatar Airways | Refueling | Night |

JSON Dictionary

mapped_tags_output.csv

CLI Search

**JSON Results**

AI integration connects specialized components in a workflow where raw data is processed to produce standardized outputs that can be queried and analyzed.

# Airline Detection


```
Text Detected: 'SPIRIT', 'N708NN' → Airline_1: Spirit
```

```
Text Detected: 'AA', 'American' → Airline_2: American Airlines
```

## Detection Goals:

- Multiple Airlines
- Tail Number
- Airline Brand
- Airplane Brand
- Airplane Make

The goal is to make a detection model for anything a user can search for.

# Flight State Labels

CUBE CONSULTING

## Flight State Label Mapping Rules

We classify images into flight states based on the presence of specific labels detected by Amazon Rekognition:

- **Taxiing:**
  If any of the following labels are present: `Runway`, `Airport`, `Tarmac`.
- **In-Flight:**
  If any of the following labels are present: `Flying`, `Sky`, `Airplane`.
- **Grounded:**
  If any of the following labels are present: `Gate`, `Terminal`, `Hangar`, `Parking`, `Jet Bridge`.

If none of the above labels are matched, classify the image as **Unknown**.

```
rekognition_outputs = {
    'test_image1.jpg': ['Runway', 'Airplane', 'Tarmac'],
    'test_image2.jpg': ['Flying', 'Sky'],
    'test_image3.jpg': ['Building', 'Vehicle'],
    'test_image4.jpg': ['Airport', 'Runway'],
    'test_image5.jpg': ['Sky', 'Cloud'],
    'test_image6.jpg': ['Flying', 'Bird'],
    'test_image7.jpg': ['Runway'],
    'test_image8.jpg': ['Airplane', 'Sky', 'Cloud'],
    'test_image9.jpg': ['Tarmac', 'Runway'],
    'test_image10.jpg': ['Flying', 'Airplane'],
    'test_image11.jpg': ['Gate', 'Airplane'],
    'test_image12.jpg': ['Terminal', 'Parking'],
    'test_image13.jpg': ['Hangar', 'Jet Bridge'],
}
```

| | Image Name | Flight State |
|---|---|---|
| 2 | test_image1.jpg | Taxiing |
| 3 | test_image2.jpg | In-Flight |
| 4 | test_image3.jpg | Unknown |
| 5 | test_image4.jpg | Taxiing |
| 6 | test_image5.jpg | In-Flight |
| 7 | test_image6.jpg | In-Flight |
| 8 | test_image7.jpg | Taxiing |
| 9 | test_image8.jpg | In-Flight |
| 10 | test_image9.jpg | Taxiing |
| 11 | test_image10.jpg | In-Flight |
| 12 | test_image11.jpg | In-Flight |
| 13 | test_image12.jpg | Grounded |
| 14 | test_image13.jpg | Grounded |

**Flight state rules in md file**

**13 test images**

**Test output**

The flight state labelling logic has three states: taxiing, in-flight, and grounded, which can be edited as per client's preferences.

# Final Tagging Script Functions

CUBE CONSULTING

**3 Key Responsibilities**

- Extracts Labels + Text
  Uses Rekognition APIs
  Parses raw image data
- Applies Tagging Rules:
  Maps airline, state, time
  and uses pre-set logic files
- Generates formatted,
  Tagged CSV & powers
  search queries

```
import csv; from label_logic import map_tags; from airline_ocr import detect_airlines
with open("rekognition_tags_output.csv") as i, open("mapped_tags_output.csv", "w", newline='') as o:
    r, w = csv.DictReader(i), csv.DictWriter(o, fieldnames=["Filename", "Airline_1", "Airline_2", "FlightState", "TimeOfDay", "Weather"])
    w.writeheader(); [w.writerow({"Filename": row["Filename"], "Airline_1": a1, "Airline_2": a2, **tags})
     for row in r for a1, a2 in [detect_airlines(row["Detected Text"])]
     for tags in [map_tags(row["Detected Labels"])]]
```

Converts Rekognition output into searchable tags with CLI support.

# Real-Time CLI Sample Images



Sample inputs we ran to test CLI application.

# Real-Time CLI Search Output (JSON)

CUBE CONSULTING

Primary airline (from OCR) [Applies to File 1 & 2]

Flight state (based on labels) [Applies to File 1 & 2]

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS


```
[
  {
    "Filename": "imagesCUBE/airplane1.jpg",
    "Airline_1": "United Airlines",
    "Airline_2": null,
    "FlightState": "In-Flight",
    "TimeOfDay": "Day",
    "Weather": "Clear"
  },
  {
    "Filename": "imagesCUBE/airplane2.jpeg",
    "Airline_1": "Air New Zealand",
    "Airline_2": null,
    "FlightState": "Takeoff",
    "TimeOfDay": "Day",
    "Weather": "Clear"
  }
]
```


Time tag (sunlight detection) [Applies to File 1 & 2]

Weather condition (from Rekognition) [Applies to File 1 & 2]

User inputs a tag query like "United, Day" or "Spirit, Night" → CLI returns matching images with structured metadata.