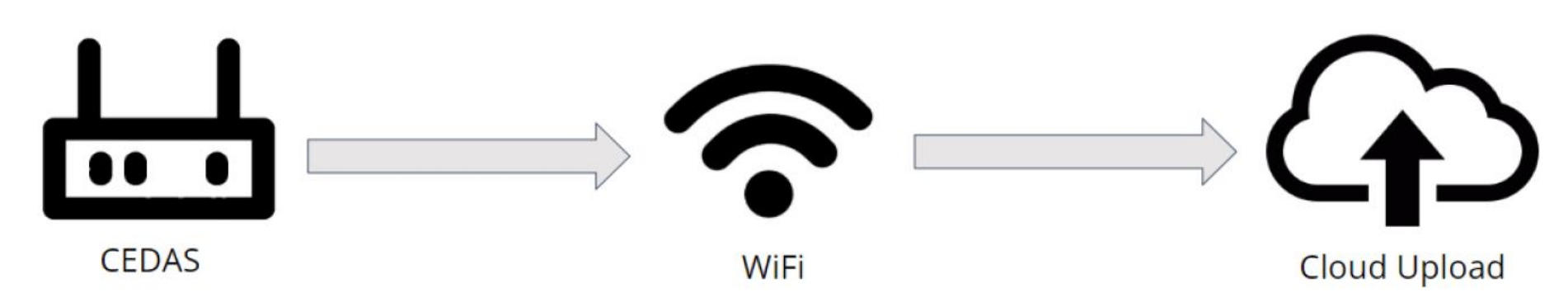
Flight Trackr: A Diagnostic System for a Diagnostic System

Team: Chloe Bates, Megan Mikami, Gennaro Napolitano, Ian Otto, and Dylan Shreiner --- **Mentor**: Scooter Nowak --- **Client**: Harlan Mitchell, Honeywell

Problem Statement

Mechanical delays routinely cause aircrafts to be grounded minutes before departure. This wastes time, money, and causes for an unpleasant flight experience. In order to prevent these unwanted delays and to ensure safe flights, engine maintenance statistics are received by engineers and mechanics via diagnostic reports.

Currently, our client, **Honeywell Aerospace**, has their engine operators download this report manually from the engine with a port connection and a USB cable. This process is tedious and only occurs once a month, leading to infrequent data collections and potential missed maintenance opportunities. To better this process and collect engine data more frequently, Honeywell has developed a product called the **Connected Engine Data Access System** (CEDAS). CEDAS is an embedded computer located in the engine that allows engines to autonomously upload its data wirelessly to a cloud.



Because CEDAS relies on WiFi connectivity, if the WiFi connection is spotty or nonexistent at a certain location, the diagnostics may not be sent. When this occurs and causes a data upload to defer from its schedule, it is difficult to determine an aircraft's location, whether it is inflight or grounded, or analyze potential engine problems.

Solution & Requirements

To solve Honeywell's problem, our team has produced the **Connected Engine Upload Status System (CEUSS)**. CUESS is a system consisting of a server backend and a frontend GUI application integrated to support its primary goals of providing engineers a way to understand logistics of failed uploads and providing aircraft operators with a visualization of locations that ensure upload success.

Server-side backend software
Responsible for the data import and execution of database correlations.

Front End GUI
Used as a graphical interface to display notifications, LON/LAT locations, and WiFi configurations.

Requirements:

- ❑ Predict when and where CEDAS uploads should occur
- ❑ Identify current aircraft locations
- ❑ Determine causes of upload failures
- ❑ Ensure upload success location

CEUSS will, in theory, help Honeywell keep track of the locations that airplanes land to give pilots and engineers an idea as to where the plane can get the best WiFi signal to upload critical maintenance logs to ensure safe flights across the world.

Technologies Used

ADSBx Public Flight Database

Java Command Line

MongoDB Database

NGINX Back End

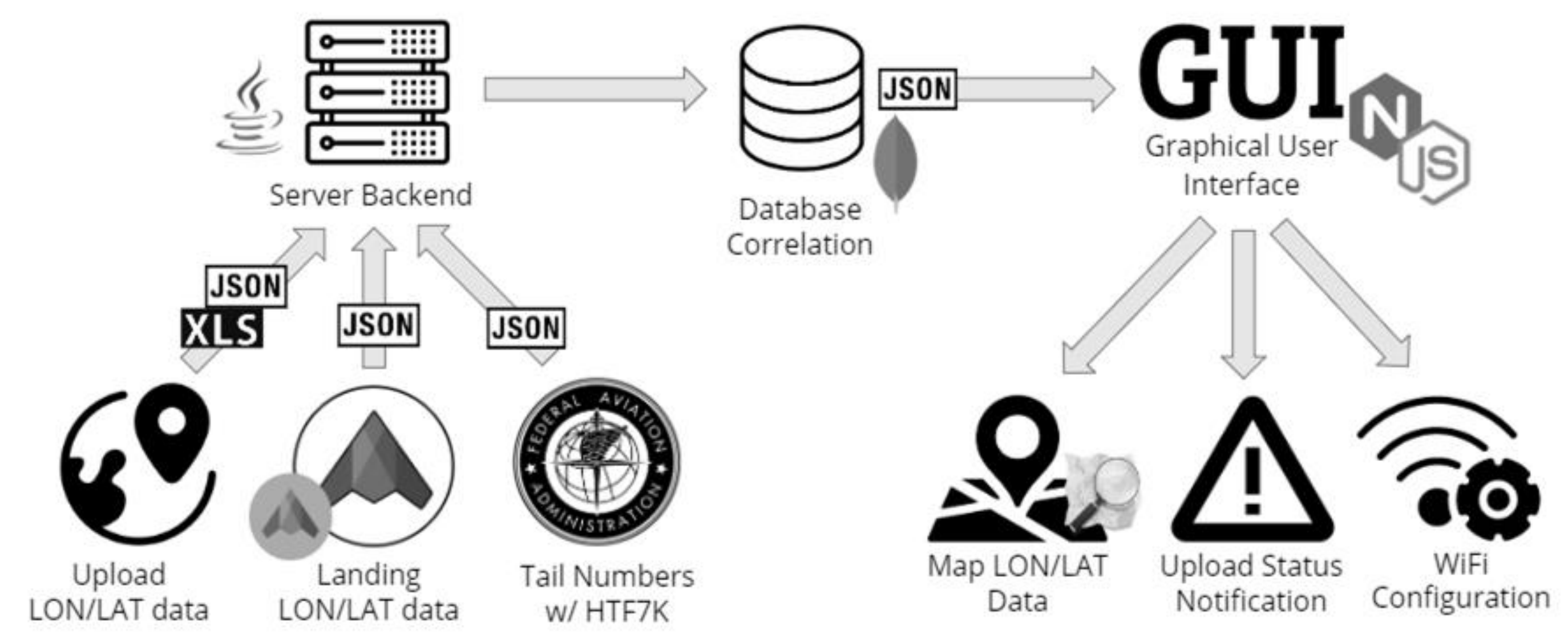
Node.js Front End

OpenStreetMap Graphical Illustration

Architectural Flow

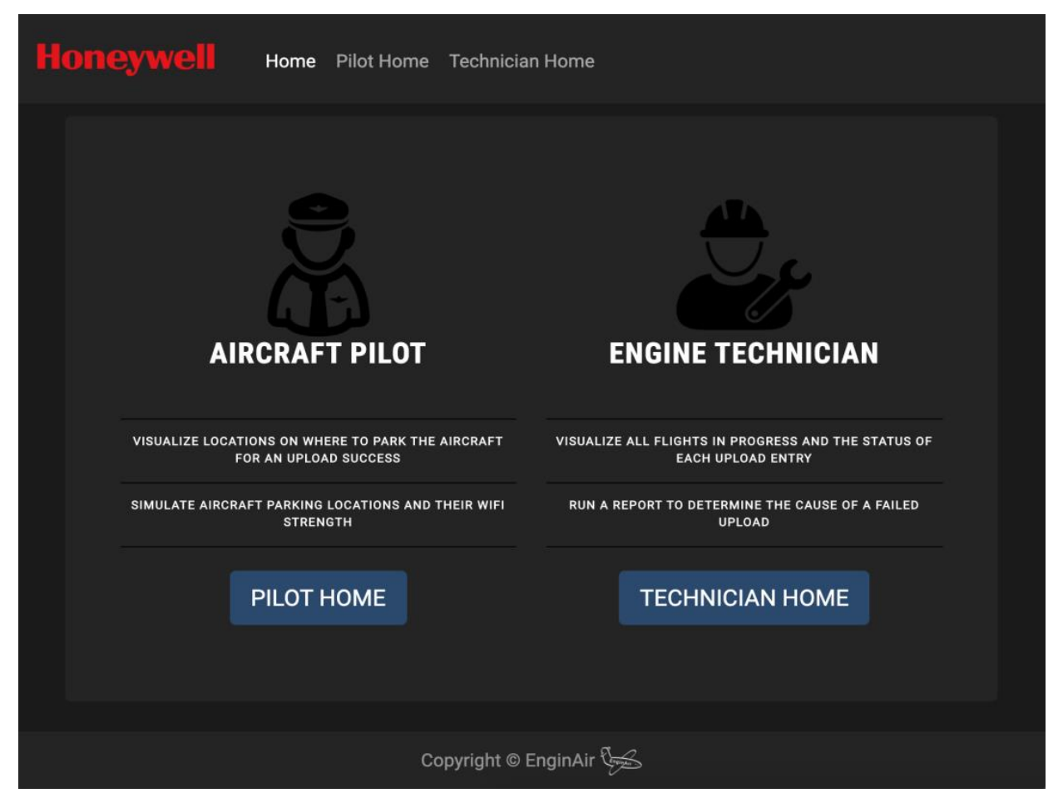
CUESS executes the following workflow:

- ➢ The backend component imports data from the following sources:
  - **CEDAS** engine upload LON/LAT coordinates are downloaded as an Excel file
  - **ADSB Exchange** provides landing LON/LAT coordinates from crowdsourced flight receivers as a JSON file
  - **Federal Aviation Administration's (FAA)** archived database provides aircraft tail numbers containing an HTF7K engine
- ➢ This data is translated into JSON files and deposited into **MongoDB**.
- ➢ The backend server then correlates the data in the database to determine flight upload statistics returned as a JSON file. The correlation can identify when and where missing uploads occur and can categorize the status of each upload.
- ➢ The frontend GUI renders the correlation result as **searchable maps** and **viewable databases**.



Key Features

Landing Page
When the user first loads the website, this page is displayed. The user, either an aircraft pilot or engine technician, can then select the link that obtains to their profession and access pages that are specific to their needs.



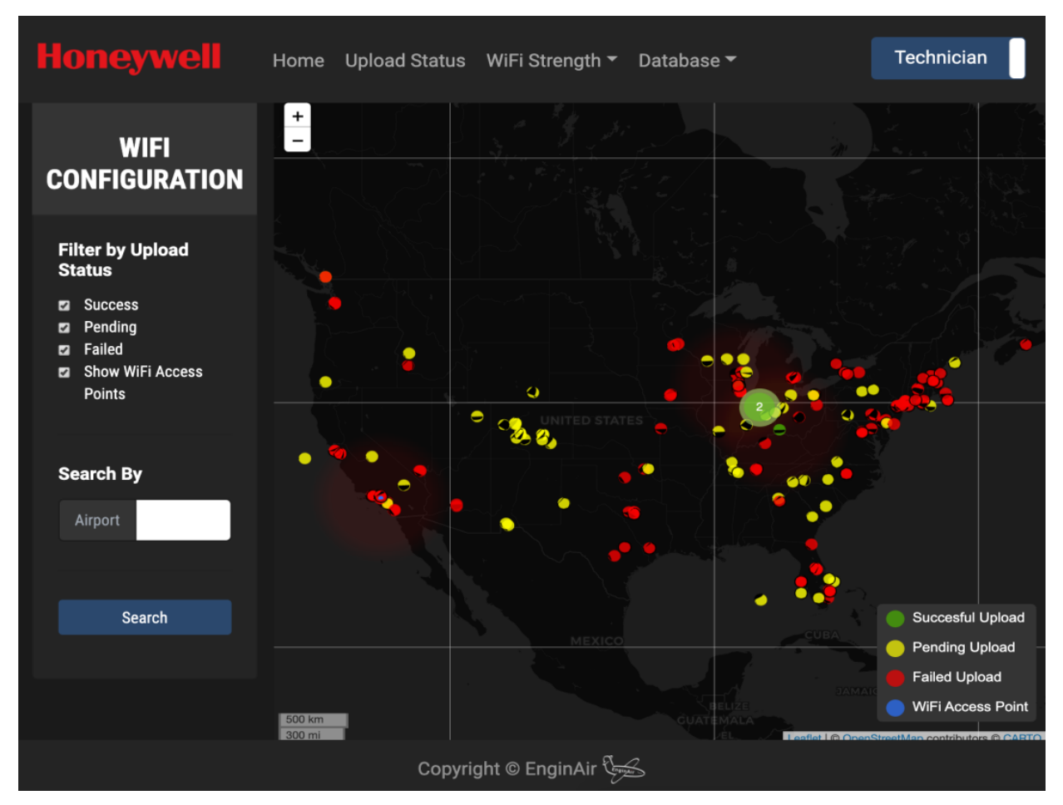
VISUALIZE LOCATIONS ON WHERE TO PARK THE AIRCRAFT FOR AN UPLOAD SUCCESS

SIMULATE AIRCRAFT PARKING LOCATIONS AND THEIR WIFI STRENGTH

VISUALIZE ALL FLIGHTS IN PROGRESS AND THE STATUS OF EACH UPLOAD ENTRY

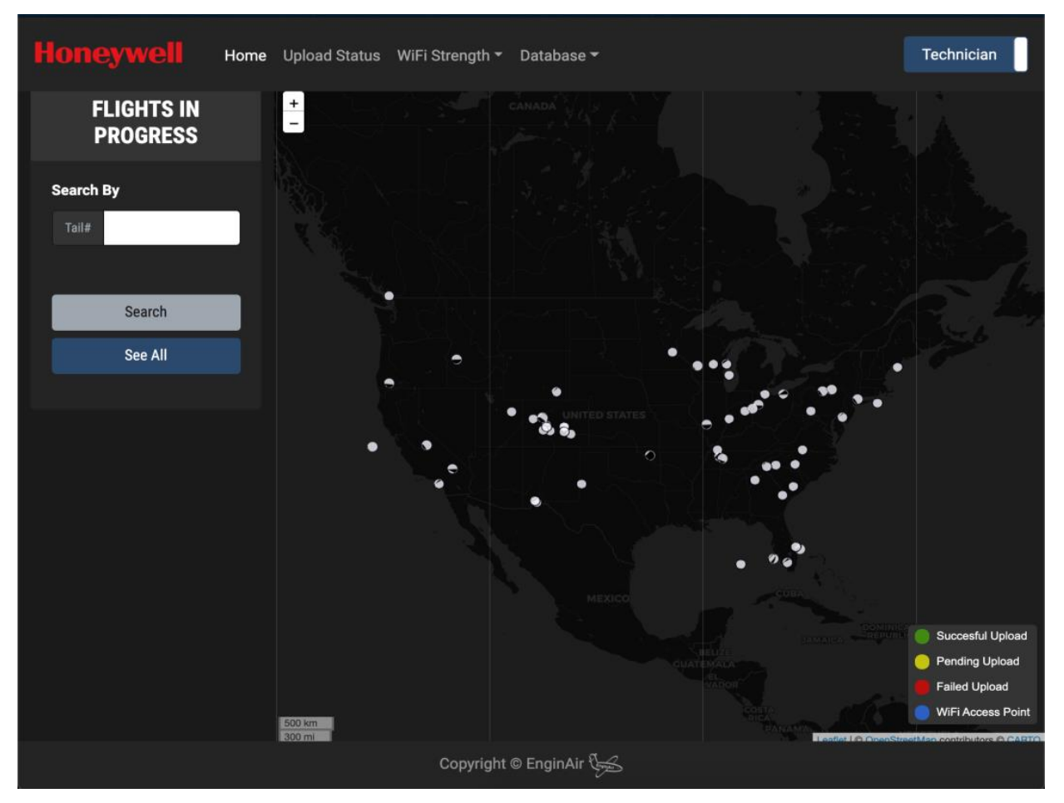
RUN A REPORT TO DETERMINE THE CAUSE OF A FAILED UPLOAD

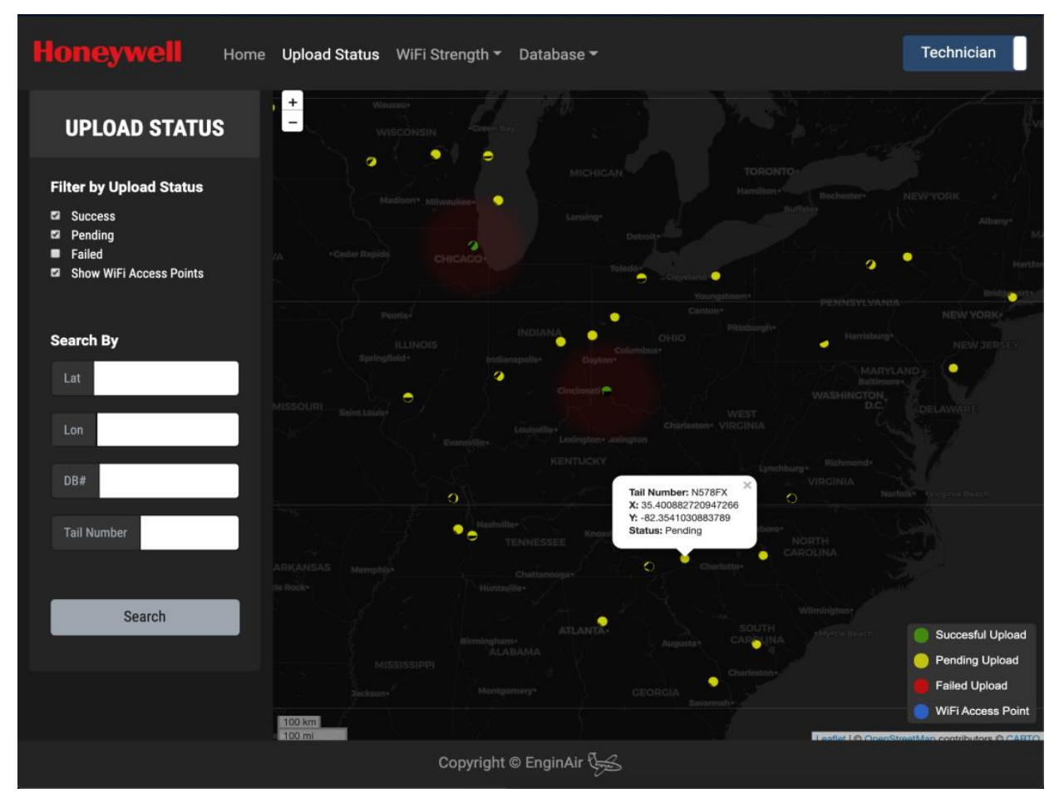
Copyright © EnginAir



WiFi Configuration
A visual representation of all flights and their upload statuses. This allows pilots to gather information on where previous flights landed, in order to guarantee a successful upload.

Flights In Progress
It is important for the engine technicians to track where all aircrafts with the HTF7K engine are. This increases efficiency in maintenance scheduling by allowing the technicians to plan for what what possible aircrafts will need maintenance work.

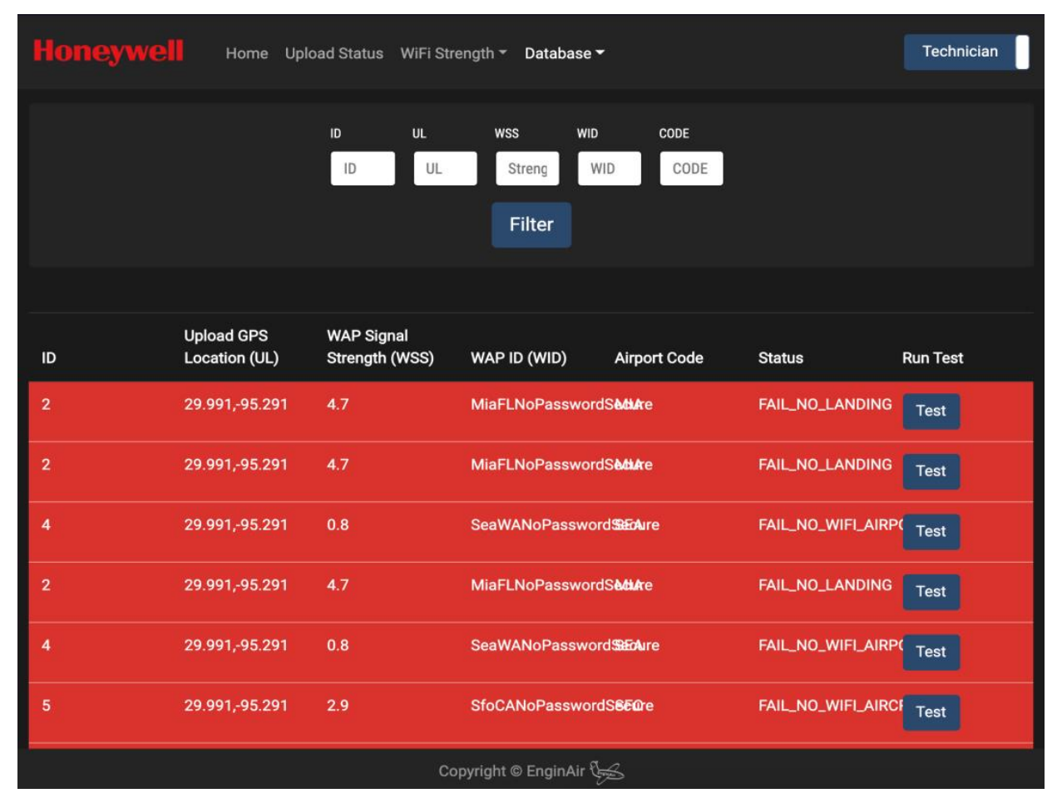




Tail Number: N578FX
X: 35.400882720947266
Y: -82.3541030883789
Status: Pending

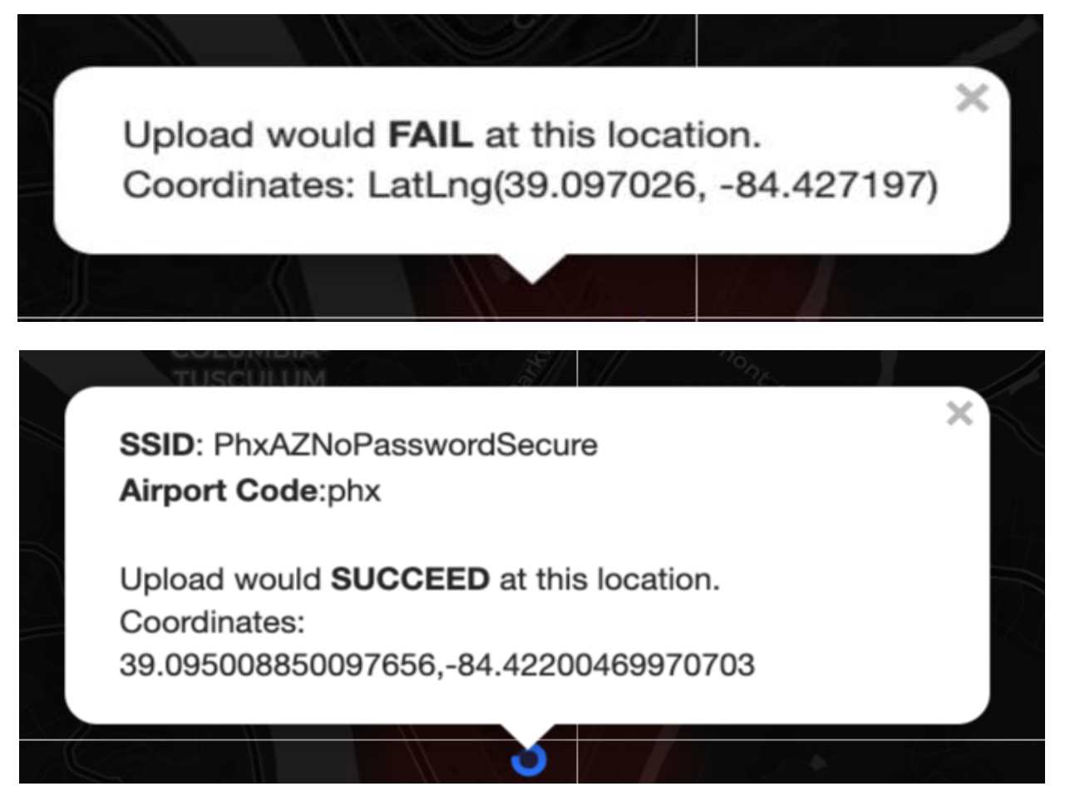
Upload Status Page
To visualize where aircrafts have potential problems uploading CEDAS data, technicians can view all aircraft upload statuses based on status color. This screenshot shows only pending statuses.

Database Diagnostic Testing
This table displays all failed upload status entries and allows the technician to run a test to determine the cause of failure. Also, they can single out types of failures by status, location, etc.



ID	Upload GPS Location (UL)	WAP Signal Strength (WSS)	WAP ID (WID)	Airport Code	Status	Run Test
2	29.991,-95.291	4.7	MiaFLNoPasswordSecure	[illegible]	FAIL_NO_LANDING	Test
2	29.991,-95.291	4.7	MiaFLNoPasswordSecure	[illegible]	FAIL_NO_LANDING	Test
4	29.991,-95.291	0.8	SeaWANoPasswordSecure	[illegible]	FAIL_NO_WIFI_AIRP	Test
2	29.991,-95.291	4.7	MiaFLNoPasswordSecure	[illegible]	FAIL_NO_LANDING	Test
4	29.991,-95.291	0.8	SeaWANoPasswordSecure	[illegible]	FAIL_NO_WIFI_AIRP	Test
5	29.991,-95.291	2.9	SfoCANoPasswordSecure	[illegible]	FAIL_NO_WIFI_AIRC	Test

Copyright © EnginAir



Upload would **FAIL** at this location.
Coordinates: LatLng(39.097026, -84.427197)

SSID: PhxAZNoPasswordSecure
Airport Code:phx

Upload would **SUCCEED** at this location.
Coordinates:
39.095008850097656,-84.42200469970703

WiFi Simulation Page
Pilots and technicians can determine which locations are within range of the hangar's WiFi. Upon clicking the map, a prediction as to whether the upload will succeed, or fail at the selected coordinates, are displayed.

Challenges

GoogleMaps API costs money
Originally, we planned on using the GoogleMaps API because of its wide range of its mapping features. Once we realized it was not free, we changed to OpenStreetMaps API, which provides a similar service and API backend.

Node.js Scalability
By default, Node.js only operates one one thread, meaning only one thread can service all the clients. To avert this problem, we used a process manager for Node.js, PM2, to allow us to scale to more instances of our web app.

MongoDB Indexing Speed
A potential future issue is MongoDB's indexing and how it pertains to speed. MongoDB's indexing is complex and difficult to determine whether we have an effective index. We don't see this as a problem yet, but we do believe it should be monitored as the project progresses.

Testing

Unit Testing
Focuses on the backend data imports and correlations. JUnit is used to ensure that data provided by the JSON and Excel files is properly added to the MongoDB database and that the information is pulled out, correlated, and added back to the database correctly. For the data imports, 8 tests were created and 5 were created for the correlation. CodeCov was also used to ensure 100 percent code coverage in our tests.

Integration Testing
Focuses on the interactions between the server-side app and the web-app. Using a simulated web browser, Postman testing utility is used to help verify that our web API calls correctly access the server backend database and return accurate results.

Usability Testing
Focuses on the user's ability to utilize CEUSS. Google LightHouse is used to automate the usability testing procedures in determining performance, accessibility, best practices, and search engine optimization. Google LightHouse also produces reports of failing audits on each webpage and provides reference pages on how to fix them.

Unit Testing — Integration Testing — Usability Testing

Future Work

Authentication
CEUSS version 1.0 does not have authentication when logging into technician or pilot pages. Version 2.0 would include this feature to make the application more secure.

CEDAS Data
For security purposes, our client was unable to give us the CEDAS upload data, so we created a "fake" data set that resembled a CEDAS file. Version 2.0 would incorporate actual engine data to enable CEUSS to be used as software rather than an idea prototype.

ADSBx Data
Because we were not able to access all the ADSBx data, we were only able to use the free archived information from a single day. Version 2.0 would have access to all ADSBx data in real-time which would ensure more accurate correlations and aircraft location mapping.

Integrate the application
Currently CUESS, is a stand-alone application. Version 2.0 would be integrated into Honeywell's current production environment, possibly in the form of a widget.