System Requirements Specification

for

RTube NeMo Team

Version 3.4 approved

Prepared by Maegan Lucas, Maxwell Moolchan, Darian Hopkins, Adam Fitch, Taylor Sumlin

Embry-Riddle Aeronautical University

November 21, 2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Taylor Sumlin, Maegan Lucas	9/29/23	SRS First Draft	1.0
Adam Fitch, Maegan Lucas	10/30/23	Update and Revise First Draft	2.0
Maxwell Moolchan	10/30/23	Edited Sections: 1.2, 1.5, 2.1, 2.2, Minor Edits through entire document	2.1
Maegan Lucas	10/31/23	Edited Sections: 1.5, 2.1, Minor text edits through entire document	2.2
Maxwell Moolchan	10/31/23	Minor text edits, added comments for revision	2.3
Adam Fitch, Taylor Sumlin, Maegan Lucas	10/31/23	Edited and Revised through entire document	2.4
Maegan Lucas	10/31/23	Added models to Appendix B, Updating Table of Contents	2.5
Maegan Lucas	11/07/23	Added models to Appendix B	3.0
Taylor Sumlin	11/17/23	Updated multiple sections	3.1
Taylor Sumlin	11/20/23	Prep Final Version	3.2
Maegan Lucas	11/20/23	Final updates, Update diagrams in Appendix B	3.3
Taylor Sumlin	11/21/23	Minor Formatting Edits	3.4

1. Introduction

1.1 Purpose

The purpose of this document is to describe and communicate the system requirements of the NeMo Team. This team will be working on the development and integration of a database to store ATC communications data and be able to effectively display the data through a web application.

1.2 Document Conventions

First Level Headings: Font: Times New Roman, Font Size: 18
Second Level Headings: Font: Times New Roman, Font Size: 14
Content: Font: Times New Roman, Font Size: 12
Headings: Font: Times New Roman, Font Size: 10, Bold, Italics

1.3 Intended Audience and Reading Suggestions

This document is intended for stakeholders and ongoing developers and future members of the project. The SRS is sectioned off to relevant areas primarily discussing the software requirements and program specifications going into detail about user classes, cases and other interactions with the system.

1.4 Product Scope

This project capsulizes two teams to work on this ongoing project over the course of two semesters. The Kaldi team, which is in charge of developing and innovating upon the existing AI technologies to be able to convert ATC (Air Traffic Control) speech to text. While Kaldi technology is a long term goal, the purpose of this is to be able to utilize text data to be able to better understand communications for pilots, ATC Controllers & other relevant parties. The NeMo team focuses on integrating the existing AI model, NVIDIA NeMo, into a web application as well as developing a database to be able to store ATC communications and flight data. For Semester 1, NeMo is focused on transcribing ATC audios with NVIDIA NeMo speech recognition models, developing a database to store communications, and updating user interface.

1.5 References

Code Repository:
https://github.com/TheCreepOfWar/asr-webapp

Documentation Repository:
https://github.com/maeganlucas/CS490-ATC

Previous Team GitHub Repository:
https://github.com/Burnetb8/Senior-Capstone/tree/main/Documentation

Akbas, M. I. (2023). *System Design Document For <Project Name>.*

Team NeMo 2023 - 2024. (2023). *System Design Document.*
https://github.com/maeganlucas/CS490-ATC/blob/main/NeMo/Documents/System%20Design%20Document%20-%20NeMo%20-%20Final.docx.pdf

Team NeMo 2023 - 2024. (2023). System Test Plan.
https://github.com/maeganlucas/CS490-ATC/blob/main/NeMo/System%20Test%20Plan%20-%20NeMo.pdf

LeafletJS. (2023). *LeafletJS Reference.*
https://leafletjs.com/reference.html

Flask. (2023). *Flask User's Guide.*
https://flask.palletsprojects.com/en/3.0.x/

Opensky Network. (2023). *Opensky Network API Documentation.*
https://openskynetwork.GitHub.io/opensky-api/index.html

Sphinx. (2023). *Sphinx Documentation.*
https://www.sphinx-doc.org/en/master/

SQLAlchemy. (2023). *SQLAlchemy - The Database Toolkit for Python*
https://www.sqlalchemy.org/

GitHub. (2023). *NVIDIA NeMo GitHub.*
https://GitHub.com/NVIDIA/NeMo

NVIDIA. (2023). *NVIDIA NeMo tutorials.*
https://docs.nvidia.com/deeplearning/nemo/user-guide/docs/en/stable/starthere/tutorials.html

2. Overall Description

2.1 Product Perspective

The product is an improvement of an existing infrastructure of a web application that utilizes ASR models to convert ATC speech to text. While creating and training ASR models is not the direct scope of the NeMo team, ASR models created and trained by other teams will be integrated into the existing web application framework to improve upon the existing ASR model used in the previous iteration of the project as well as the user interface of the web application.

2.2 Product Functions

- Successfully track and display accurate flight data
- Provide recordings of ATC communications
- Transcribe ATC communication using ASR models into relatively accurate text files
- Store communication and flight data within a database

2.3 User Classes and Characteristics

2.3.1 Flight Instructor

The Flight Instructor will have the ability to see student flights in-air and be able to listen in and actively transcribe ATC communications so that the instructor can assist the pilot as needed.

2.3.2 Pilot

The pilot in most cases, is a student pilot that may be on their first solo flight. The pilot has the ability to communicate with ATC to assist in any issues. This allows pilots to see other flight transcriptions to better understand aerospace phraseology for future flights.

2.4 Operating Environment

The operating environment of this project will be through a web application that will be hosted online using Flask. The host machine will need a NVIDIA GPU in order to utilize NVIDIA NeMo models. The host environment will then be able to push the web application online through a dedicated address, allowing users to access the application and speech-to-text capabilities without needing a NVIDIA GPU.

2.5 Design and Implementation Constraints

Design constraints consist primarily of hardware constraints. As the NVIDIA NeMo tool is only usable on NVIDIA GPUs, the only way to access the transcription models is to use a computer with a NVIDIA GPU. Additional constraints include the difficulty of working with the NeMo model due to the nature of the ASR web application. As the ASR application is primarily developed utilizing Python, it is difficult to connect NeMo to the application.

2.6 User Documentation

Currently user documentation is provided through text files that provide instructions and assistance for installing all project dependencies and instructions to install and run the program.

2.7 Assumptions and Dependencies

Current dependencies include the following:

- Python Version 9+, Pandas Version 2.0.1, and OpenPyxl version 3.1.2 for the backend portion & Python dependencies.
- Flask web application framework for the application itself.
- Flask SQLAlchemy for database functionalities.

- OpenskyAPI for the plane data which includes: icao24, callsign, origin country, time position, last contact, longitude, latitude, geospatial altitude, ground status, velocity, true track (heading), vertical rate, squawk, position source, and category of the aircraft.
- LeafletJS for the map framework.
- OpenStreetMap.org for the geographical map tileset & VFRMap.com for the aeronautical charts.
- NVIDIA NeMo framework is used to implement NVIDIA NeMo ASR models. In order to run NVIDIA NeMo and associated ASR models a NVIDIA GPU is required otherwise NVIDIA NeMo ASR models will be unable to run, disabling the ability to use the speech-to-text transcription feature. The server or machine that this software is deployed on will have a NVIDIA GPU but during development some machines may not have an appropriate GPU.
- NVIDIA NeMo dependencies include but are not limited to: braceexpand, editdistance, g2p_en, ipywidgets, jiwer, kaldi-python-io, kaldiio, librosa, marshmallow, matplotlib, packaging, pyannote.core, pyannote.metrics, pydub, ruamel.yaml, scipy, soundfile, sox, texterrors.
- Due to the vast amount of API dependencies, in the event one or more of them is down the application may not work partially or fully operate.

3. External Interface Requirements

3.1 User Interfaces

The user interface will be through a web application. The design is similar to existing flight tracking services such as FlightAware or FlightRadar24. This user interface allows for the user to interact with specific flights and be able to listen in on recorded ATC communications. The user is additionally presented with options to select individual flights and airports to see their respective data. Users are also presented with a choice of a geographic style map or an aeronautical style map.

3.2 Hardware Interfaces

The product is entirely based through a web application and there will be no hardware developed in the course of this project. There is, however, a hardware dependency where a NVIDIA GPU is required to transcribe communications.

3.3 Software Interfaces

The NeMo team will be developing a database to connect to the NeMo web application. This database will store data relating to ATC communications as well as general flight and airport data for future feature development. The NeMo web application also interacts with the OpenSky API to retrieve flight information. Near Aero is utilized to retrieve flight communication. The geographic map is sourced from Openstreetmap.org and the aeronautical maps are sourced from VFRMap.com.

3.4 Communications Interfaces

HTTP requests are used to call the OpenSky API for flight information.

4. System Features

4.1 Communication Database

4.1.1 Description and Priority

A database is to be developed to hold attributes of communications. This includes communication audio, transcript, time, date, location, and callsign. This is of High priority for the project. This has the highest benefit to the further development of the application as it allows for storing communications and important attributes that can be used for more features.

4.1.2 Stimulus/Response Sequences

When a user selects a flight path or airport, the database will query for the communications relevant to that flight path or airport.

4.1.3 Functional Requirements

Req. 4.1.3.1 The database shall store communication details in the form of callsign, time, date, longitude, latitude, audio, transcript, and a boolean value indicating whether the communication is coming from ATC.

Req. 4.1.3.2 The database shall be able to perform queries to recall communication details based on user input in the form of a flight path selection.

Req. 4.1.3.3 The database shall be able to perform queries to recall communication details based on user input in the form of an airport selection.

Req. 4.1.3.4 The user shall be able to click an airplane icon to view a popup with communication details corresponding with the selected flight path.

Req. 4.1.3.5 The user shall be able to click an airport icon to view a popup with communication details corresponding with the selected airport.

4.2 Transcription Using NVIDIA NeMo

4.2.1 Description and Priority

NVIDIA NeMo will be used to transcribe live ATC communications. The audio and transcript will then be saved into the communications database. Once the "Transcribe" button is selected, NeMo will be called to transcribe and stored the data to the database and will be displayed to the user. This is of High priority to the project.

4.2.2 Stimulus/Response Sequences

When a user selects the "Transcribe" button, NeMo will be called to transcribe the communication data. The audio and transcription will be saved into the database and the transcription will be displayed to the user.

4.2.3 Functional Requirements

Req. 4.2.3.1 The application shall transcribe communication data when the user selects the "Transcribe" button for a specific flight.

Req. 4.2.3.2 The application shall display transcribed audio to the user within 10 seconds of selecting the "Transcribe" button.

4.3 User Interface Update

4.3.1 Description and Priority

The user interface will be updated and will be polished as a whole. This will include adding a Foreflight-style chart option in addition to the geographical map, adding different icons for different airplane categories and types, and smoothing the movement of the airplane icons along their flight paths. This will also include the ability to have different configurations for varying use-cases, such as different flight schools. This is of Medium priority.

4.3.2 Stimulus/Response Sequences

4.3.2.1 The user will be able to select between a geographical map and the aeronautical chart. Once selected, the application will display the user's choice.

4.3.2.2 The user will choose a configuration and the application will update accordingly.

4.3.3 Functional Requirements

Req. 4.3.3.1 The application shall display a choice of a Foreflight-style aeronautical chart or a geographical map.

Req. 4.3.3.2 The application shall display the user's choice map type.

Req. 4.3.3.3 The application shall display all icons properly over the selected map type.

Req. 4.3.3.4 The application shall display varying icons based on airplane category.

Req. 4.3.3.5 The application shall allow a user to choose their configuration

Req. 4.3.3.6 The application shall update to the chosen configuration within 10 seconds.

4.4 Communication Waypoints

4.4.1 Description and Priority

Waypoints, or "breadcrumbs," will be added to flight paths to depict where communication took place. They will update with the movement of the airplane icons. This is a Medium priority to the application.

4.4.2 Stimulus/Response Sequences

When a user selects an airplane icon, the flight path will be displayed with waypoints to depict where communication took place. A window of flight information including the ability to listen into specific radio frequencies will pop up, allowing the user to then transcribe ATC communications.

4.4.3 Functional Requirements

Req. 4.4.3.1 The application shall display a flight path when an airplane icon is selected by the user with visible waypoints to depict communication in their corresponding locations.

Req. 4.4.3.2 The visible waypoints shall update with airplane icon movement.

Req. 4.4.3.3 The side panel of flight information will pop up allowing the user to see relevant information.

4.5 Communication Identification

4.5.1 Description and Priority

Communications will be able to be tracked across frequencies using callsigns. A user will also be able to identify when someone is communicating over a specific frequency by the color of the icon. This is a medium priority.

4.5.2 Stimulus/Response Sequences

4.5.2.1 When communication is detected on a frequency, the icon for the frequency will change to indicate live use to the user.

4.5.2.2 The user will select an airplane icon and the system will track the communications of the corresponding callsign across frequencies to display along the flight path.

4.5.3 Functional Requirements

Req. 4.5.3.1 The icon for the frequency shall change colors to indicate communication detected on the frequency.

Req. 4.5.3.2 The application shall recognize communication of a certain call sign across different frequencies.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The final product needs to be able to run efficiently and without stutters as the program will be tracking live flight data from around the United States with further room for expansion to new regions in the future. To ensure this, the project must adhere to the following performance requirement(s):

Req. 5.1.1: The application shall not refresh plane icons in less than 5 seconds.

5.2 Safety Requirements

There are no safety requirements for this project.

5.3 Security Requirements

There are no security requirements for this project.

5.4 Software Quality Attributes

This project needs to be maintainable for future developers by having up-to-date documentation as well as by following good software practices such as comments and whitespace. This application should also be easy to use for various types of users.

5.5 Business Rules

All users will be able to use all functionalities of the NeMo web application.

6. Other Requirements

There are no other requirements for the NeMo team that are not already listed within this document.

Appendix A: Glossary

#	Term	Abbreviation	Description
1	Air Traffic Control	ATC	Refers to tower communication to or from aircraft.
2	Automatic Speech Recognition	ASR	Software that recognizes speech patterns.
3	Word Error Rate	WER	Refers to the error rate in ASR technology typically on a scale of 0 to 1 but can exceed 1 depending on sample comparisons.
4	Federal Aviation Administration	FAA	United States government agency that oversees and regulates civilian aviation in the United States.
5	NVIDIA NeMo		NVIDIA NeMo is a generative AI framework that allows for conversational AI to function and transcribe ATC communications within the web application.
6	Graphical Processing Unit	GPU	A hardware component used to handle complex algorithms.

Appendix B: Analysis Models

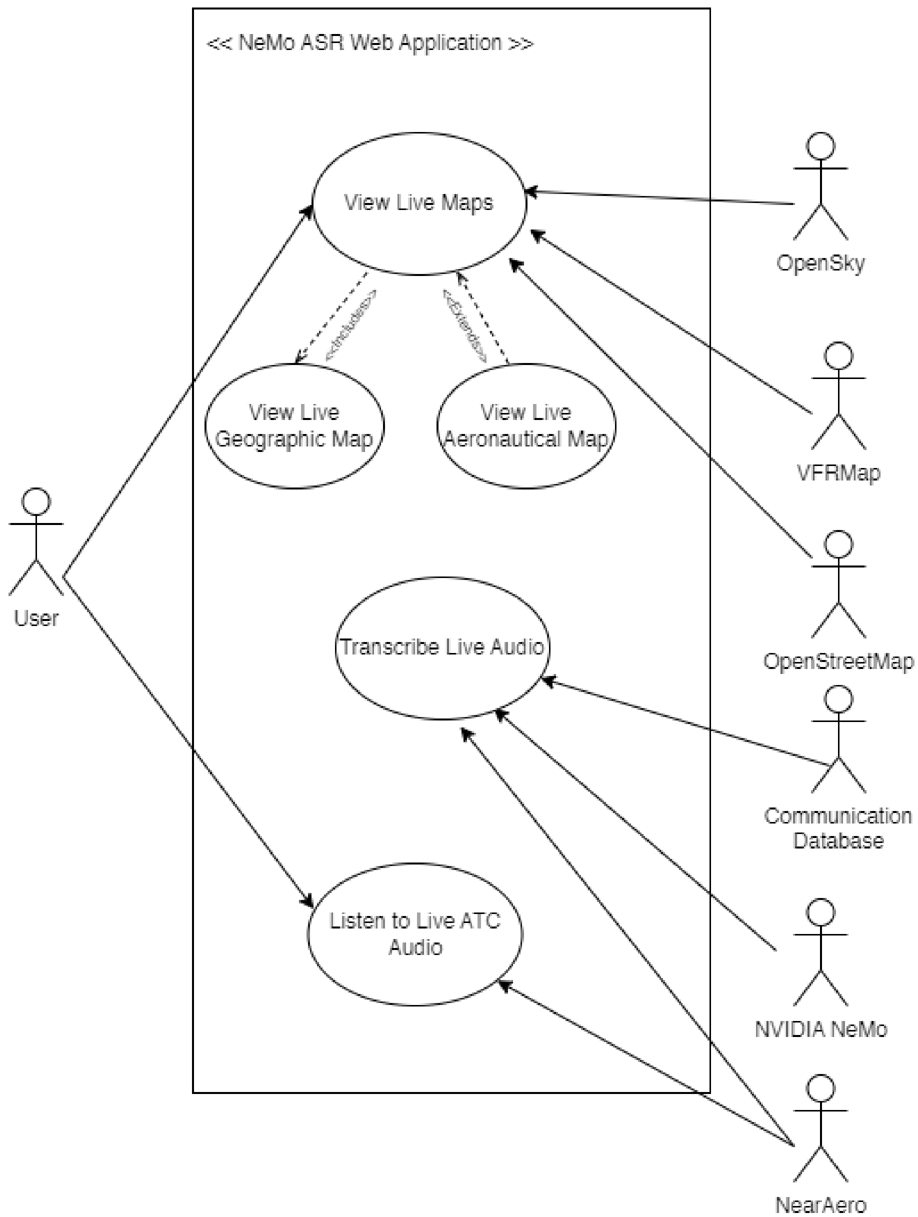

Figure 1: Use-Case Diagram for NeMo system.

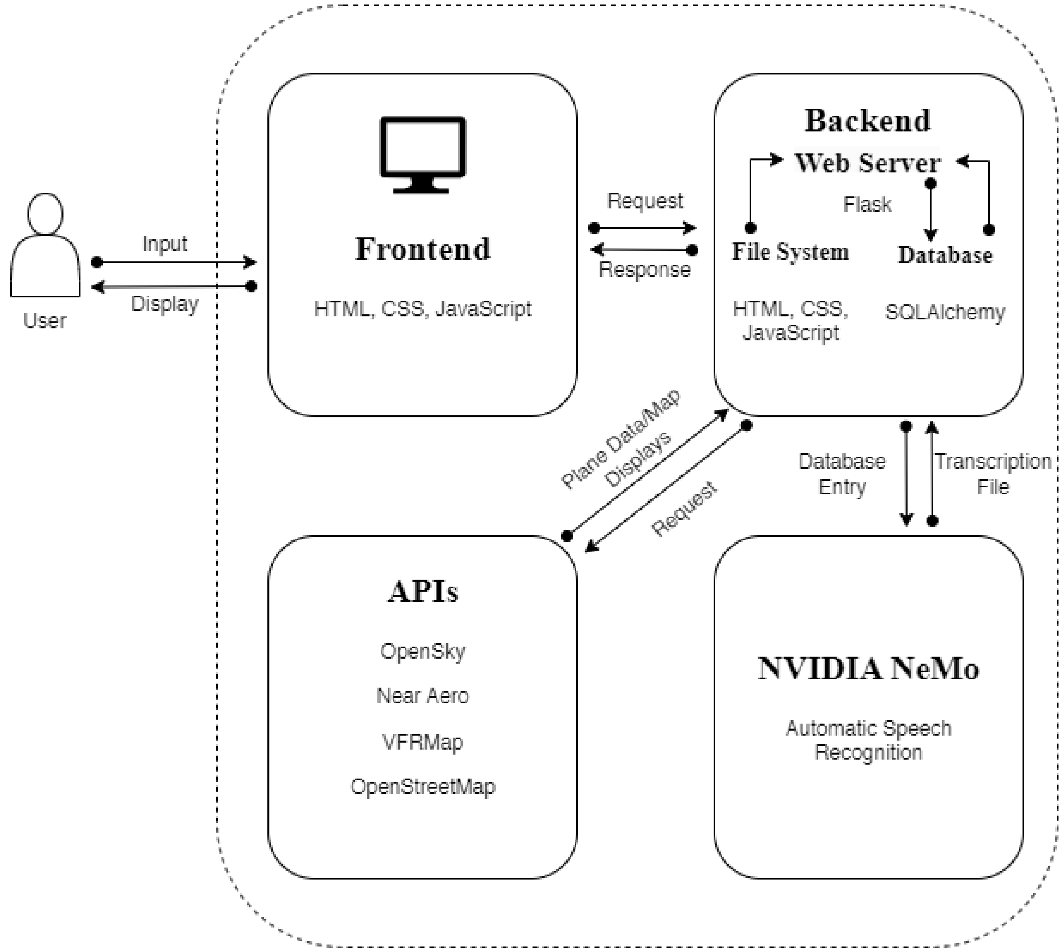

Figure 2: System Architecture Diagram for NeMo.

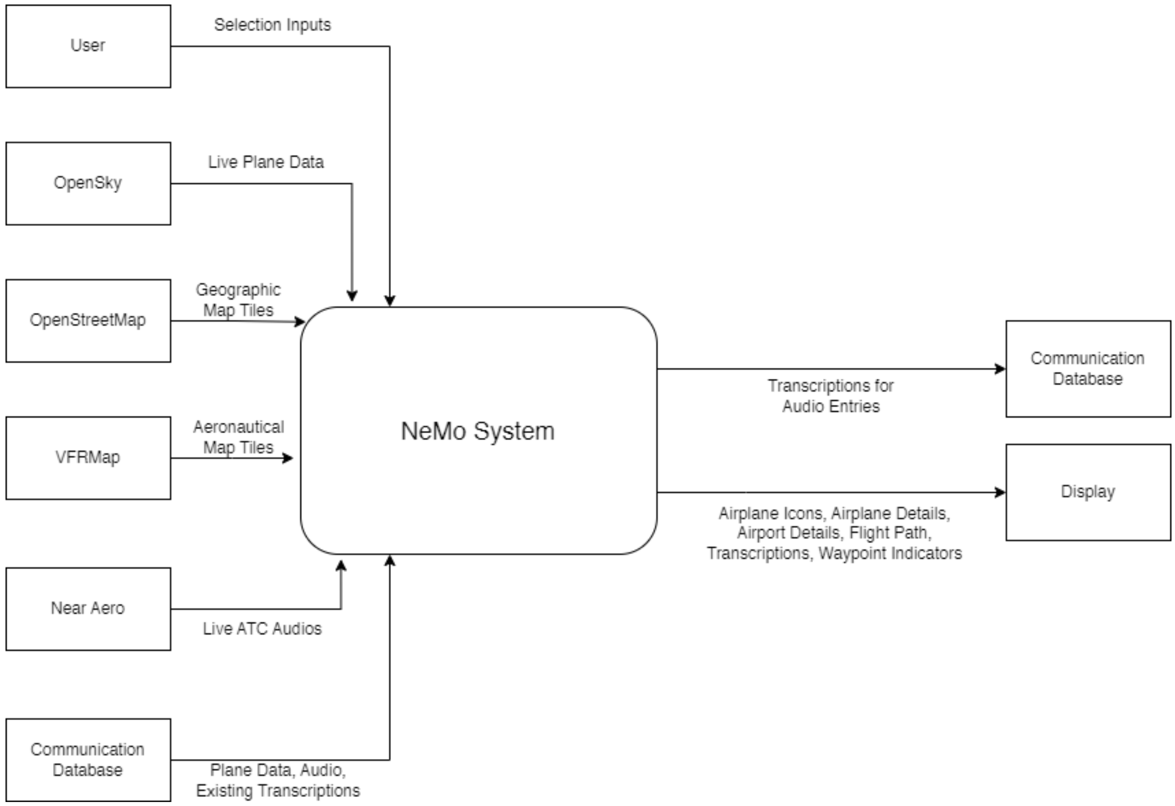

Figure 3: Context Diagram for NeMo system.

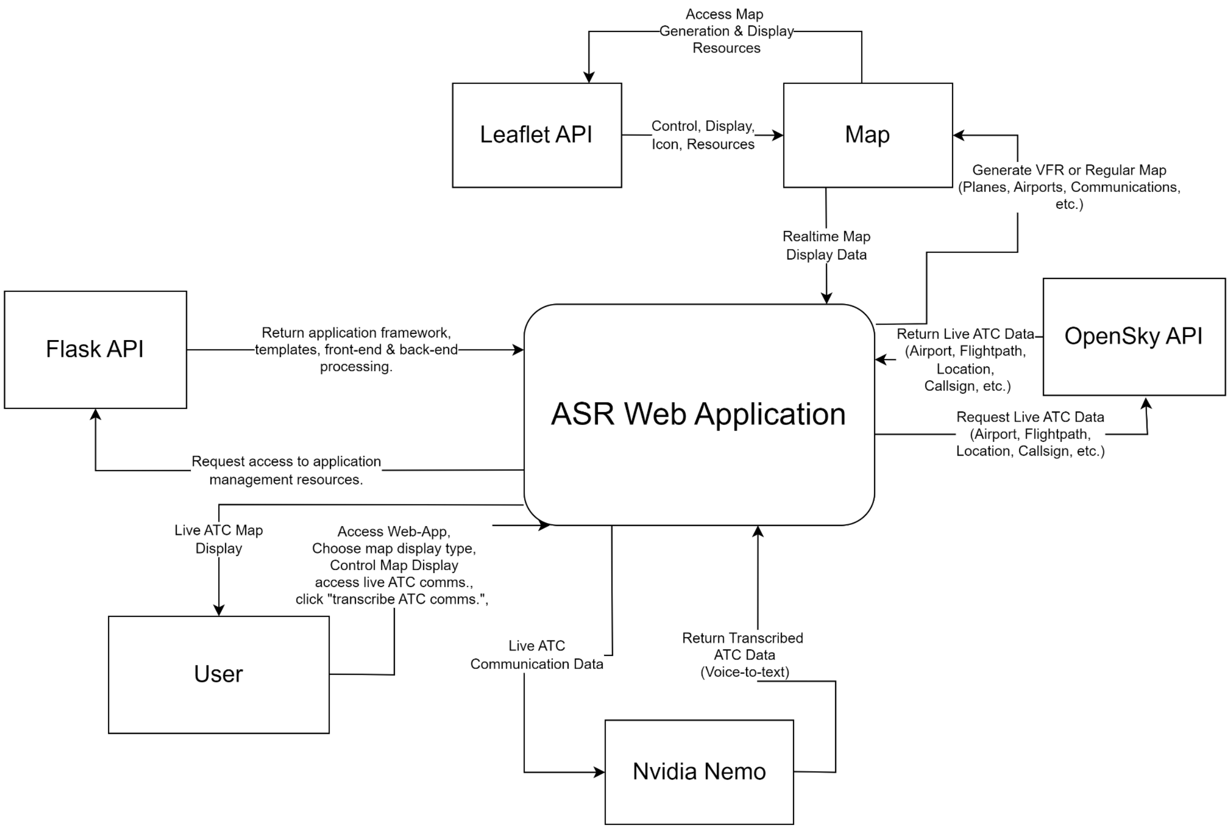

Figure 4: Level 0 Data Flow Diagram

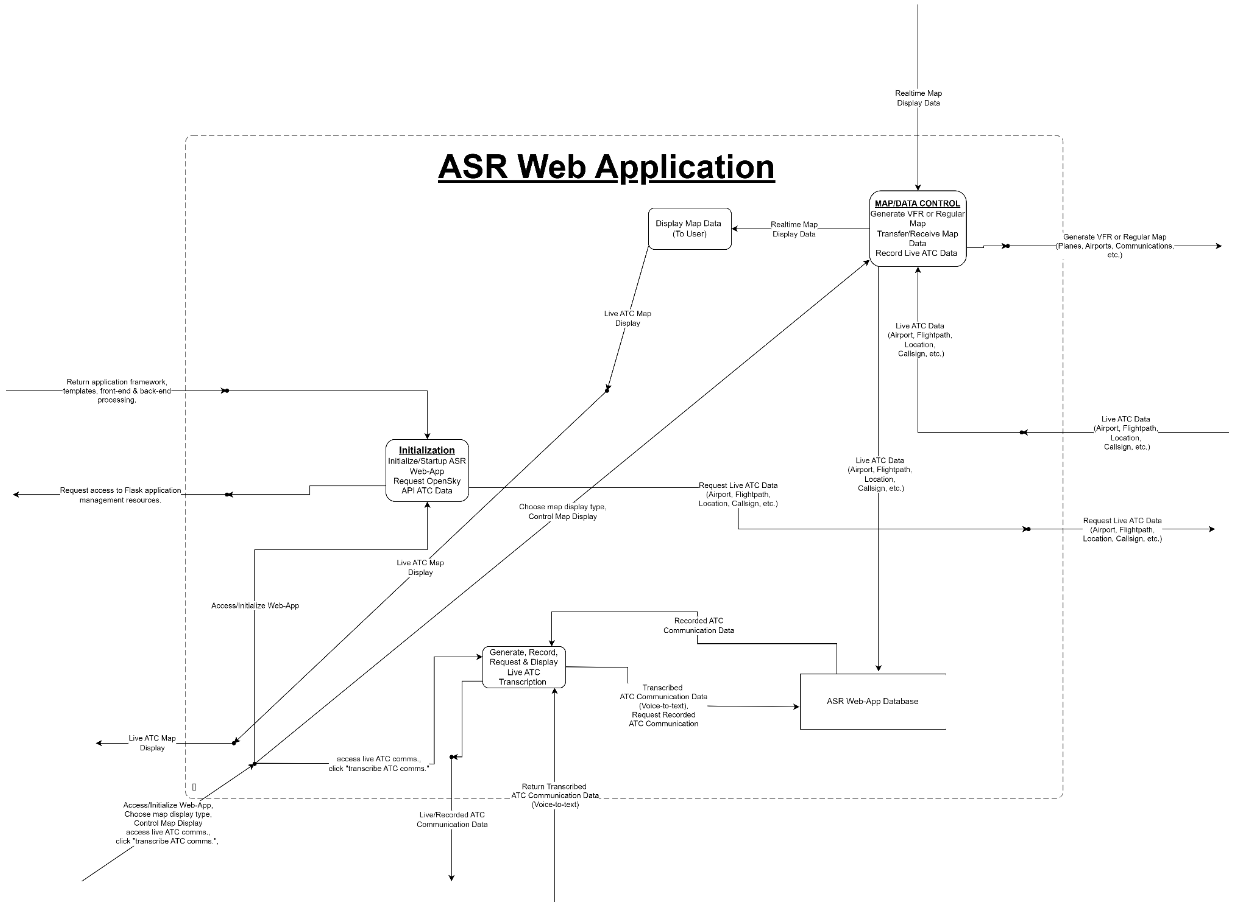

Figure 5: Level One Data Flow Diagram

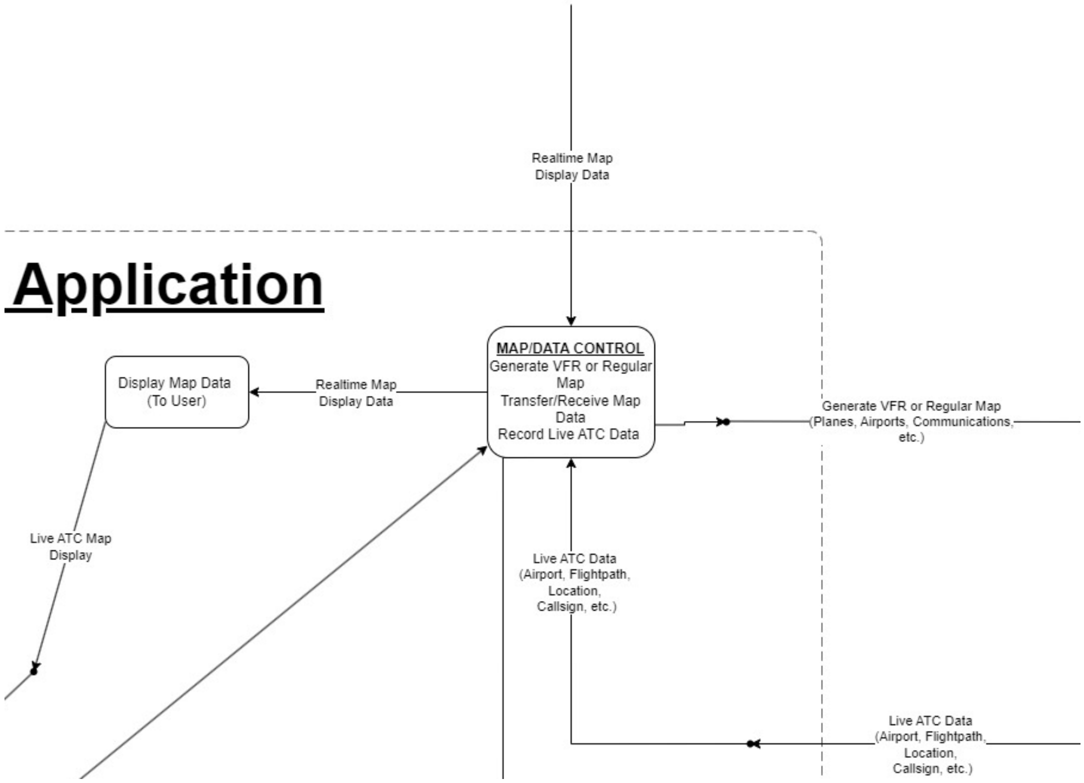

Figure 6: An expanded view of the top-right corner of the Data Flow Diagram depicted in Figure 5.

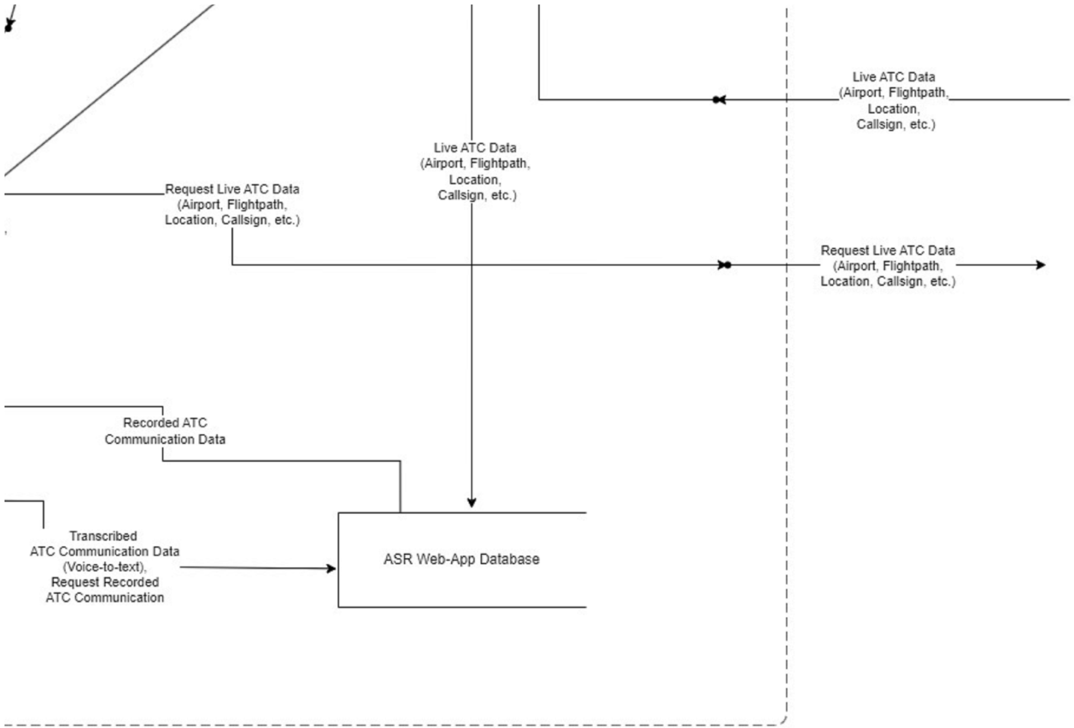

Figure 7: An expanded view of the bottom-right corner of the Data Flow Diagram depicted in Figure 5.

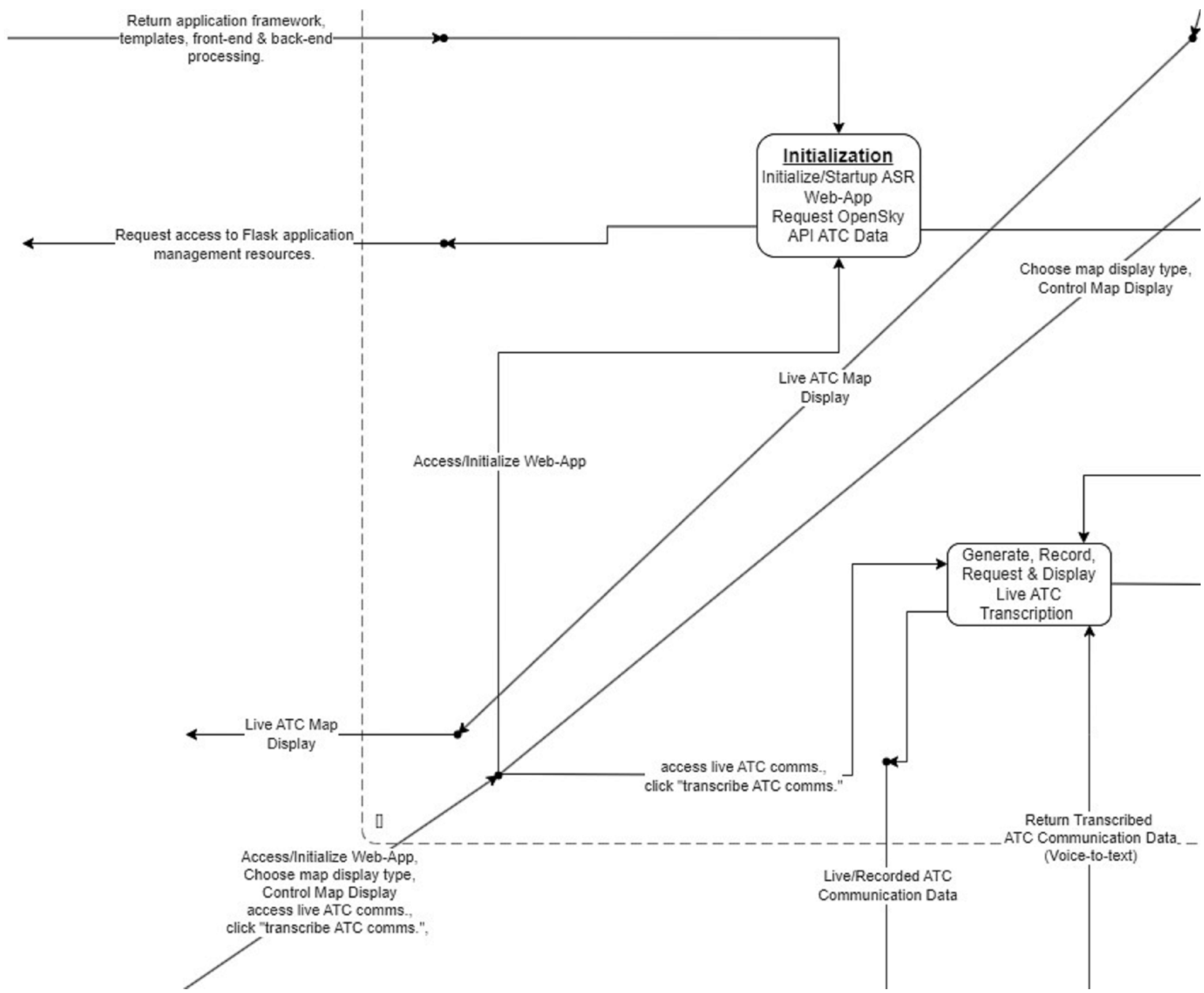

Figure 8: An expanded view of the bottom-left corner of the Data Flow Diagram depicted in Figure 5.

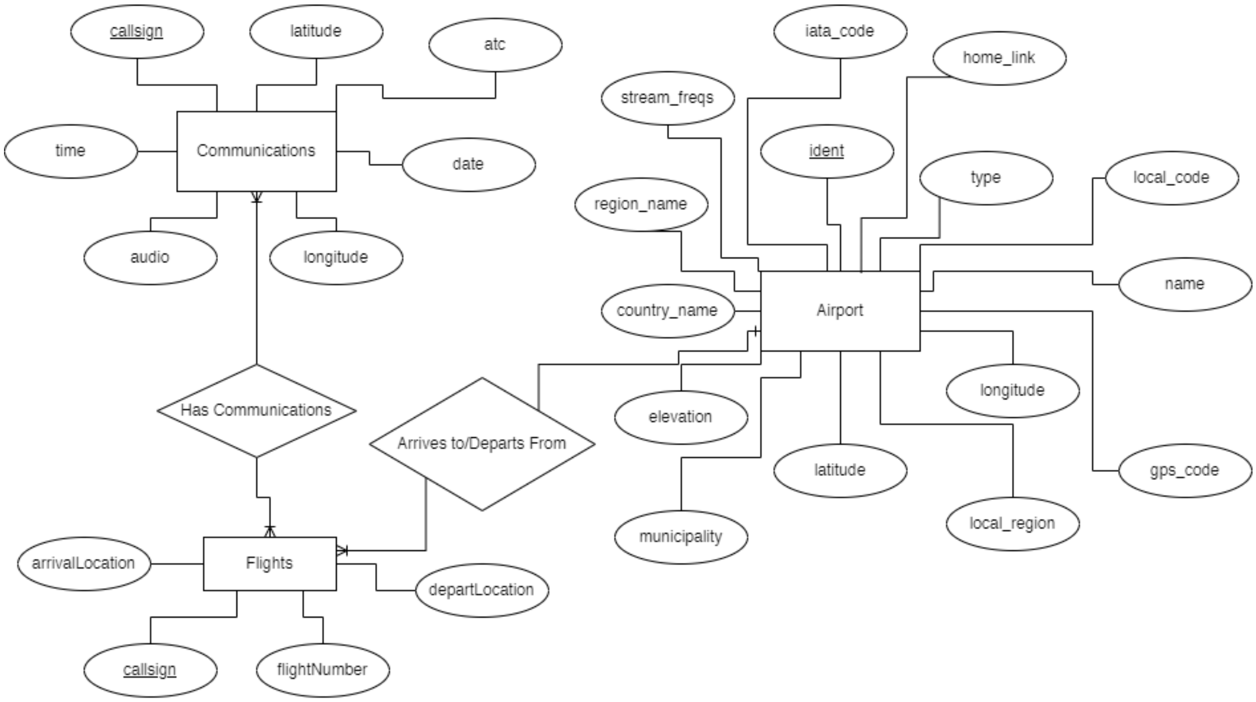

Figure 9: Entity-Relationship Model for Communications database