# zeroG Airlines Schedule Data Processor

|  |  |
| --- | --- |
| Title | zeroG Airlines Schedule Data Processor |
| Submission date | 16-11-2023 |
| Prepared By | Shweta Shivamogga Dattatri |
| Prepared For | zeroG |
| Version | V.1 |

# Table of contents

[zeroG Airlines Schedule Data Processor 1](#_Toc28660)

[Table of contents 2](#_Toc7934)

[Introduction: 3](#_Toc29169)

[Key Features: 3](#_Toc4431)

[Architecture Overview: 4](#_Toc325)

[Benefits: 5](#_Toc17191)

[Use Cases: 5](#_Toc5216)

# Introduction:

The zeroG Airlines Schedule Data Processor is a data engineering solution tailored to efficiently manage and process schedule data for zeroG Airlines. The project is designed to provide a comprehensive solution for retrieving and consolidating aviation-related data. The system comprises two main functionalities: lookup and merge. These operations leverage a structured dataset consisting of flight schedules, fleet information, and airport details to deliver insightful results.This project enhances accessibility, simplifies data retrieval, and enables comprehensive analysis.

# Key Features:

1. Azure Blob Storage Integration:

* Secure connection to Azure Blob Storage for retrieving schedule data files.

2. Two Modes of Operation:

* On-Demand Flight Insights:

Dive into the specifics of individual flights effortlessly with the lookup mode. Whether it's departure times, gate assignments, or passenger counts, this mode provides instant access to the details that matter.

* Holistic Data Synthesis:

In the realm of merge mode, witness the magic of consolidation. The library expertly weaves together the threads of information from the three storage files, presenting a unified and organized CSV file. This output becomes your canvas for comprehensive analysis, insightful reporting, or seamless integration into other data workflows.

# Architecture Overview:

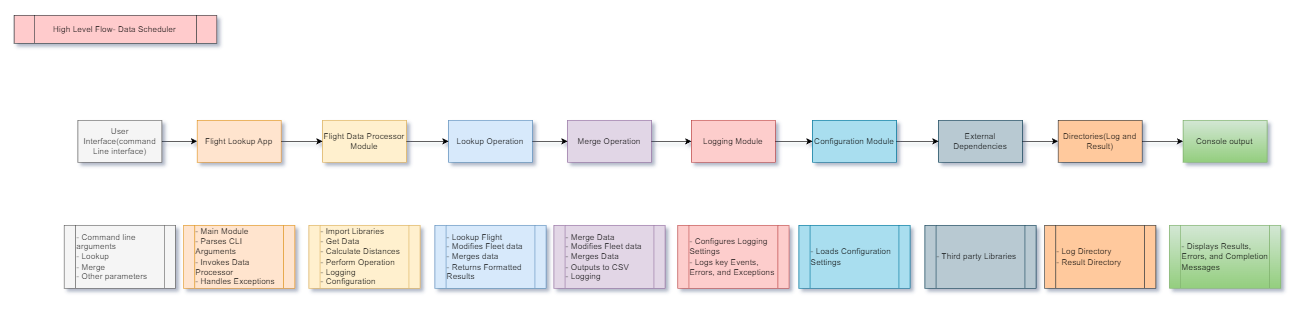


Diagram Link: <Architecture Flow diagram.drawio>

Data Flow:

* The data flow starts with user input from the Command Line Interface.
* User requests are processed by the Flight Lookup App, which directs the flow to the appropriate operation (lookup or merge) in the Flight Data Processor.
* Data is processed through various stages, including calculations, data retrieval, and integration.
* Processed results are displayed in the console and stored in log and result directories.

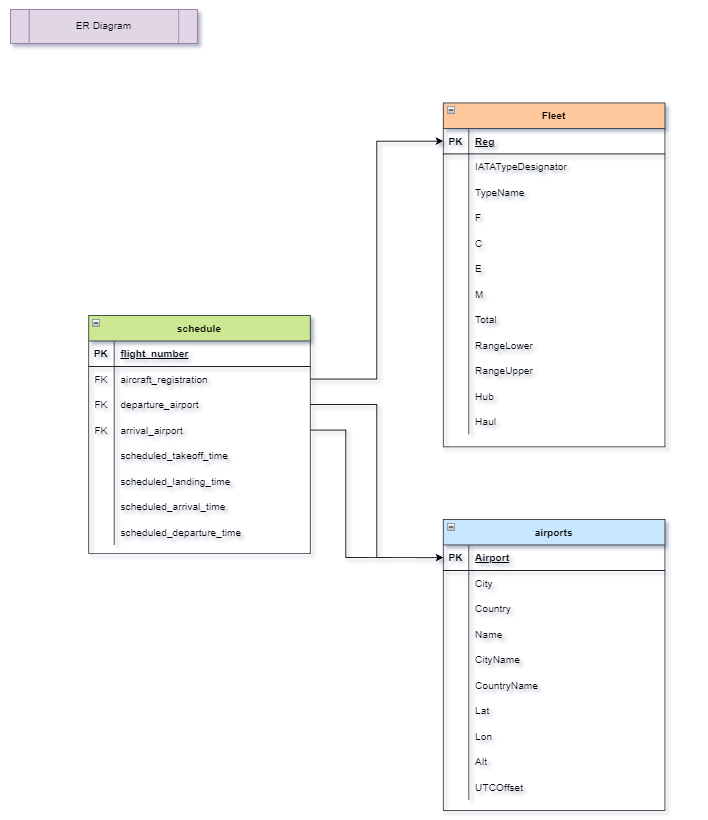


Diagram Link: *<ER Diagram.pdf>*

In a relational model, data is organized into tables, and relationships between tables are established using keys (such as primary and foreign keys).

* Schedule Table: Contains information about flights, and it has foreign keys that establish relationships with the Fleet and Airports tables.
* Fleet Table: Contains information about aircraft, and it has a primary key (Reg) that can be referenced as a foreign key in the Schedule table.
* Airports Table: Contains information about airports, and it has a primary key (Airport) that can be referenced as a foreign key in the Schedule table.

Connections:

* The aircraft\_registration column connects the Schedule table to the Fleet table.
* The Reg column connects the Fleet table back to the Schedule table.
* The departure\_airport and arrival\_airport columns in the Schedule table connect to the Airport column in the Airports table.

# Benefits:

1. Streamlined Data Access:

* Enables quick retrieval of specific flight details (`lookup` mode).
* Facilitates easy analysis of entire schedule data (`merge` mode).

2. Improved Efficiency:

* Efficient file handling and data processing for optimal performance.

3. Secure Data Interactions:

* Secure integration with Azure Blob Storage.
* Input validation to enhance security.

# Use Cases:

1. On-Demand Flight Details:

* Use `lookup` mode to quickly access details for specific flights.

2. Comprehensive Analysis:

* Leverage `merge` mode for in-depth analysis of the entire schedule.

Test Case Documentation: <Testing documentation.docx>