**Flight Information System Documentation**

# Introduction

The Flight Information System is a Java application that stores, sorts, and displays flight details. It consists of three main classes: Flight, FlightTest, and Form1. This documentation provides an overview of the project's objectives, implementation details, and limitations.

# Objectives

The primary objectives of the Flight Information System are as follows:

1. Implement a Flight class to represent airline flights with various attributes.
2. Create Flight objects using user input and the Scanner class.
3. Sort Flight objects based on their flight numbers.
4. Display sorted flight information in a graphical user interface (GUI).

# Implementation Details

## 3.1 Flight Details

### 3.1.1 Attributes

* airline\_name: Name of the airline operating the flight.
* flight\_number: Unique identifier for the flight.
* flight\_origin: Departure city or airport.
* destination\_cities: Destination city or cities.
* airFare: Airfare for the flight.
* departureTime: Departure time in HH:mm format.
* arrivalTime: Arrival time in HH:mm format.
* totalMinutes: Total travel time in minutes.
* flightDuration: Total travel time formatted as hours (with two decimal places).

### 3.1.2 Methods

* calculateTotalTravelTime(): Calculates and returns the total travel time in hours.
* toString(): Provides a formatted string representation of the flight details.

### 3.1.3 Getters and Setters

* Various getter methods to access private attributes.
* setFlightDuration(double flightDur): Setter method for the flightDuration attribute.

## 3.2 FlightTest Class

### 3.2.1 User Input

* Uses the Scanner class to read user input for creating Flight objects.
* Prompts the user for flight details, creates Flight objects, and calculates travel time.

### 3.2.2 Sorting

* Sorts Flight objects based on flight numbers using the Arrays.sort() method and lambda expressions.

## 3.3 Form1 Class (GUI)

### 3.3.1 GUI Components

* JTable: Displays flight information in a tabular format.
* JButton (Display Results): Triggers the display of sorted flight information.
* JScrollPane: Provides a scrollable view for the JTable.

### 3.3.2 GUI Customization

* Table headers and cell contents are formatted for better readability.
* Cell contents are center-aligned for a consistent look.

# Limitations

* The application does not handle input validation extensively, assuming valid user inputs.
* Sorting is implemented only for flight numbers; additional sorting options could enhance functionality.
* The GUI lacks advanced features like search, filtering, or exporting to other formats.
* Limited error handling in case of unexpected user inputs or file read/write failures.

# Assumptions

* Users are expected to provide accurate and valid flight details during input.
* The application assumes a single day for flight duration calculation; flights spanning multiple days are not supported.

# Conclusion

The Flight Information System demonstrates fundamental Java concepts, including classes, objects, user input, sorting, and GUI programming. While functional, the system can be further enhanced with additional features and robust error handling to create a more comprehensive flight management tool.

Note: This documentation serves as a comprehensive overview of the Flight Information System. It provides insight into the project's objectives, implementation, limitations, and assumptions. For detailed code and UML diagrams, please refer to the submitted project files.