"Sky is the Limit: A Comprehensive Flight Data Visualization and Analysis

 by Group 40"

This report shows Group 40's final project, which focuses on creating and applying a complex platform for viewing and analysing flight data. The project's primary goal is to provide a comprehensive and user-friendly application that makes it easier to explore flight data using different filtering and visualisation methods. Individual flight records and collections of data points are stored and managed by the platform using DataPoint and DataList classes. An interactive bar chart, a date range search engine, a dynamic heatmap, and a geographic map visualisation are some of the platform's key features that give customers helpful insights into flight patterns, airport activity, flight lateness, and cancellations.

Introduction

Group 40's final project, "Sky is the Limit," offers a comprehensive platform for analysing and visualising flight data. The project uses the provided dataset containing individual flight records and manages and stores them using tailored Datapoints and DataList classes.

Main, Widget and Screen Classes.

The Widget Class is responsible for creating all widgets and buttons in the project. The Widget class is constructed to take every possible detail as a parameter. There are two types of buttons created in the project, a text and an image button. The type of button being drawn is cleverly controlled by the widget Type variable which can have 1 as a value for generating a text button and 2 as the value for generating an image button. Every widget is also assigned an ID to handle individual buttons.   
The Screen Class generates screens with a given ID, background colour and an array list of widgets linked to that screen.

The Widget class and Screen class are majorly responsible for making the project user inter

Datapoints and DataList classes.

The project’s cornerstones are the Datapoints and DataList classes. Individual flight records are stored as an instance of the Datapoints class, which has a string array that holds all the information attached to that record. All of the data points are stored using the DataList class, which at its heart has an ArrayList of Datapoints. DataList has a method called populate List, which takes the CSV data as a String array and adds it to the ArrayList. The DataList class also has all our queries, allowing us to get flights by date range, departing airport, and lateness. These methods all return a new DataList with the filtered data, allowing us to easily chain multiple queries on top of each other. The queries are all relatively simple, simply iterating through the ArrayList of Datapoints and adding all the ones that match the input to a new DataList to be returned.

Bar Charts

The Bar Chart class is a custom component that generates bar charts for various datasets. The height of each bar is calculated based on the number of bars to be drawn and the overall height of the chart. Furthermore, the width of the bars is decided by the proportion of each data point to the maximum value in the dataset and the width of the entire chart. A method for identifying the selected bar using the mouse location and bar coordinates is also included in the Bar Chart class.

However, while the BarChart class is designed to be static, the BarCharts class was created to allow for dynamic data loading and user interaction. The BarCharts class is in charge of displaying a dropdown menu and lateness text box, as well as managing user interactions with these elements and updating the data and BarChart as needed. If data is available, the draw method will generate the BarChart; otherwise, it will display a message asking the user to add airports so that data may be obtained. When data is added or removed from the airport dataset using the respective methods, the updateBarChart method is called to refresh the BarChart object. The list of tracked airports is stored in a simple array, which can be used to query the entire dataset iteratively.

Date Range

Date Range is a sophisticated search engine that enables users to effortlessly find flights within a specific start and end date range. The selection of the dates is seamlessly integrated into a user-friendly calendar icon that can be easily accessed by clicking on it. Once the user chooses the start date, the end date options are automatically restricted to dates after the selected start date, and the same is true if the user prefers to select the end date first.

When the user is ready to proceed, they click on the "search" button, and a visually stunning page of all the available flights will appear in the form of boarding passes, each with a QR code on the side. Upon scanning the code, the user will be directed to our GitHub repository. The boarding pass displays all relevant information, such as origin city, destination city, date, departure time, arrival time, airline, and arrival state abbreviation. They can efficiently scroll through the boarding passes using the mouse wheel.

By clicking anywhere on the boarding pass page, it will slide down smoothly, synchronizing with the new page sliding up. This new page showcases a heat map that visually depicts the density of flights for each day and hour of the selected date range.

Geographic Map

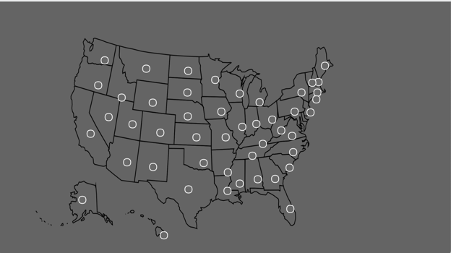
The USA\_MAP class is a visually pleasing interactive component that displays a Geographic map of the United States with clickable state buttons. The map has an easy-to-use interface that allows users to view flight data from multiple states, improving the way they fly.

When a state is selected from the map, the class displays a thorough list of flights. This information is presented tastefully, ensuring readability and easy navigation. The USA\_MAP class employs a set of unique widgets, including buttons for each state and a close button for returning to the map view, to manage the interface elements efficiently.

The class dynamically refreshes the flight list based on the selected state. When users interact with the map it makes them feel like they have real-time access to the data. Furthermore, the USA\_MAP class uses smooth animations and transitions to provide a visually appealing and gratifying user experience. For example, When a state is selected, the relevant button expands to span the screen, showing the flight data in a smooth transition. When the user clicks the cross widget, the USA\_MAP class activates a well-thought-out de-expanding functionality. This operation shrinks the extended state button to its original size, seamlessly returning the user to the map view. This feature provides a smooth and easy navigation experience, allowing users to explore different states effortlessly.

Further, The map also includes a scrolling functionality and expansion and de-expansion transitions. When provided with a list of flights from a specific state, users can easily browse the flight details using the mouse wheel. This scrolling functionality allows vast amounts of data to be displayed and guarantees that customers can swiftly navigate the available flight information without feeling overwhelmed.

Overall, the USA\_MAP class blends user-friendly navigation elements like de-expanding buttons and smooth scrolling with a visually appealing UI to deliver a pleasurable and valuable user experience.

Table

Description automatically generated

Graphical user interface

Description automatically generated with medium confidenceA picture containing text

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated