Group 3 Programming Project

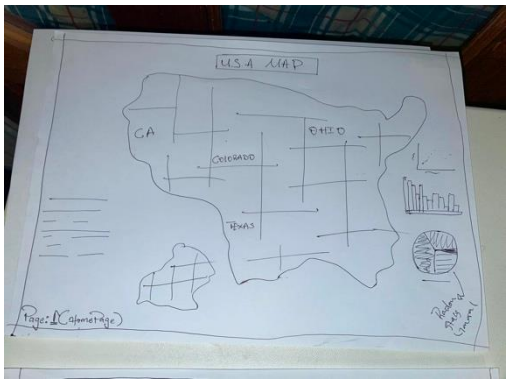
Outline of Design:

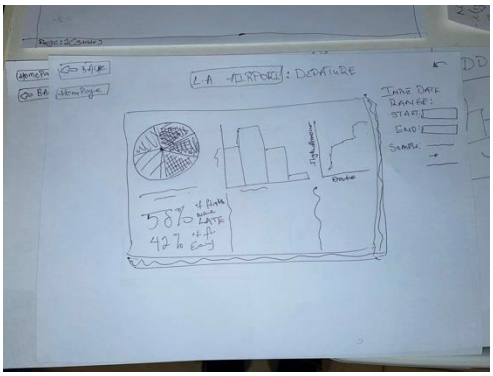
Our primary goal was to create a responsive, user-friendly program that offers insightful visualizations of the provided data. The home page, by default, displays quick insights with user interactions that reveal more in-depth information based on the selected state. Additional pages delve further into specific data streams, providing more targeted insights visually.

In essence, we aimed to simplify the complex flight data and generate concise, easily digestible insights, enabling users to quickly grasp the essence of the information presented. By maintaining a focus on simplicity and user interaction, we have developed an informative program that effectively communicates important data to users in a professional yet approachable manner.

Team Organization and Work Distribution:

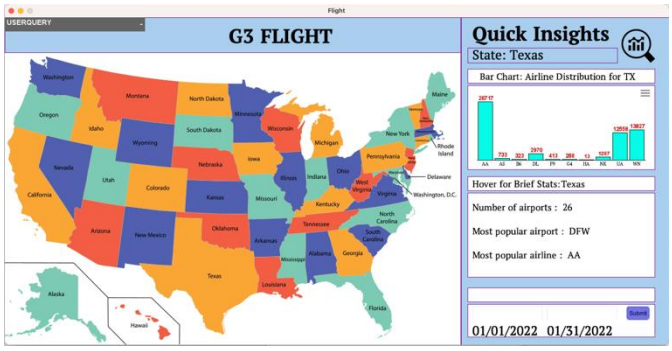
During our initial group meeting, we collaboratively discussed our vision for the project, which led to the first general project outline. We then assigned tasks based on each member's strengths and interests. We agreed to meet every Wednesday and maintained communication through a WhatsApp group chat and shared Google Docs for progress updates and idea exchange.

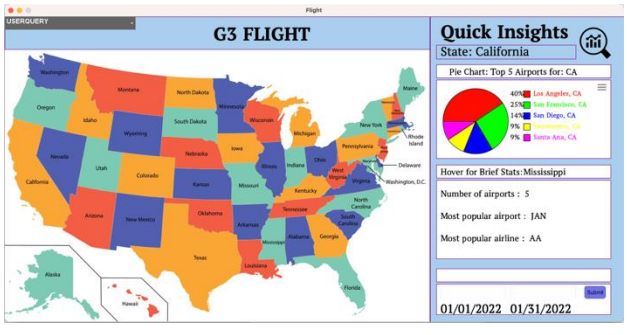




Implemented Features:

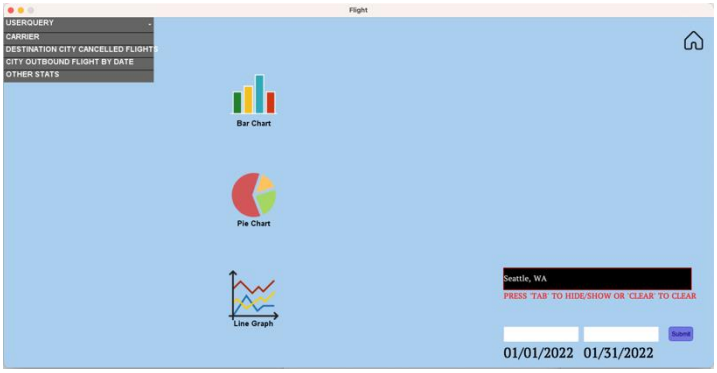
The home screen features an interactive map of the United States, making use of the mouseClicked function to enable users to refine the data displayed according to the selected state. We chose this design due to its user-friendly nature and aesthetic appeal. We further implemented hover over feature that displays brief and quick stats of the state that is being hovered over.

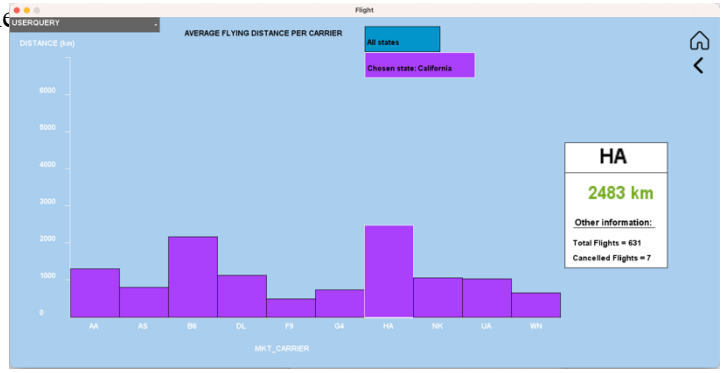




Additionally, our platform features a "Visualize Quick Stats" section tailored to the selected state. This section includes a bar chart representing the airline distribution within the chosen state and a pie chart displaying the percentage share of the top five airports. To ensure accurate data visualization, we made use of hash maps, hash sets, and bubble sort algorithm for these charts.

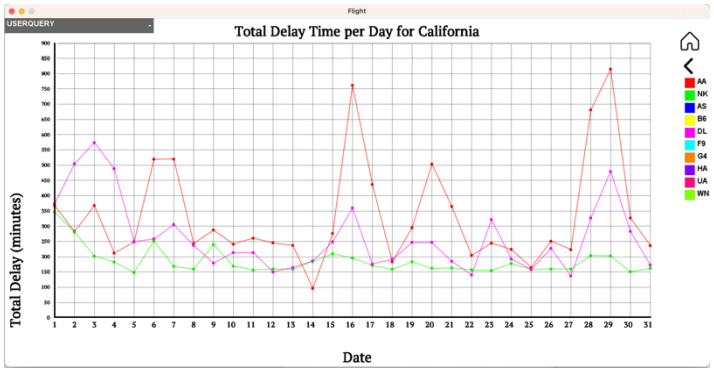
On the home page, users will find a date range viewer, which indicates the current date range for the displayed data (defaulted to a full month/full data set). Text boxes are provided above the

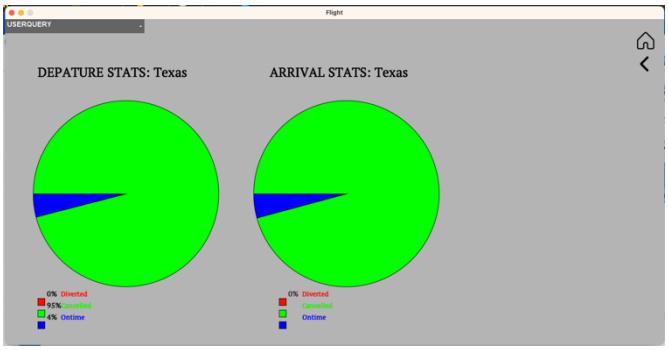




On the Query screen (page 2), users are presented with three buttons that provide access to three distinct data visualization tools: a histogram, pie charts, and a line graph. The date range editor is also present on the Query screen to display the current date range and permit the user to specify the date range if needed.

A key method of accessing the data was via our pup up text search bar which is refined by a drop-down box. Here the user has access to the search box with the ‘tab’ button on the keyboard. The user can then access data instantly by selecting either cancelled flights by destination city or total flights by carrier.





As depicted in the accompanying screenshots, we have integrated a histogram that displays the average flying distance of carriers based on the state selected from the home page, or alternatively, users can choose to view data for all states. By hovering over each bar in the histogram, users can access additional information about the specific airline on the display screen. Furthermore, we included a pie chart that illustrates departure and arrival data for the selected state.

Lastly, a line graph presents the total delay time for airlines per chosen state, with the 31days of the month featured on the x-axis. This delay time value represents the punctuality of airlines from the respective state, where lower values indicate timelier arrivals. Coloured

checkboxes adjacent to the graph correspond to individual airlines, and by selecting these boxes, users can view the line graph tailored to the checked airline.

Problems Encountered:
During the flight data parsing process, we encountered challenges with various data points from the CSV file. Inconsistencies in the file structure required us to make adjustments, such as using error handling techniques like try and catch methods for proper date conversion (e.g., 1/1/2022 vs. 01/01/2022). Handling dates stored in the mm/dd/yyyy hh:mm format was a notable example.

We also needed to implement defensive coding strategies to address occasional issues arising from empty cells or null data points. This approach helped us prevent errors like indexOutOfBoundsError and NullPointerException, ensuring a more robust and reliable program.

// Add more examples or issues encountered once remembered.

Individual contributions:
Eniola:

- Wrote the code to parse the data
- The Query Line graph
- The homepage layout
- Added Icons, images and few other design elements
- Home page Pie chart
- Implemented the Search Bar
- Added the textbox functionality and date range filter
- Sorted the data after parsing using algorithms like bubble sort and Hash Sets
- Added a new pie chart constructor for main page to take in more data
- Implemented a dropdown menu

Eoin:

- Implemented the screen class
- Implemented the widget class
- Created the dynamic bar chart that was displayed on home screen
- Implemented a button that allowed user to switch between the bar chart and pie chart
- The brief stats feature method for the home page
- Implemented the histogram class
- Made the histogram on the second page so that it could change between displaying data for all states and data for the chosen state.
- Added feature to histogram that displayed info of the carrier hovered over by the user

Kate:

- Created pie chart based on flight arrival times for the second page.
- Changed the colours and text to make the project more visually appealing.
- Worked on the original state class which involved downloading state images, altering them to fit the screen and creating different screens for each state, we ended up abandoning this idea as it caused delays and made our program slow.

Daire:

- Implemented the hover method for screen 1 displaying data based on mouse location
- Formed the States class implemented with the mouseClicked which impacts all data based on user input
- Assigned the appropriate way to display the data per query, based on knowledge gained from STU11002