Visualizing Bike Share data on the San Francisco Bay Area

Diogo Gomes (93025), Martinho Tavares (98262)

Information Visualization, 2023 (MSc in Engineering Physics/Informatics Engineering, University of Aveiro)

Abstract

This report provides a brief introduction to the San Francisco Bay Area bike share dataset and its potential users.
We present our visualization solution after defining the main questions that the user should be able to answer with the application developed.
A low-fidelity prototype evaluation and a heuristic evaluation were conducted by our colleagues, which helped refine our visualization.
Finally, we were able to develop a visualization tool that helps explore the dataset and answer the proposed questions. Three main pages are available in the application: a geographic view, station/city trip details, and global details.

Motivation and objectives

The objective of the project presented is to create a data visualization tool that gives information about bike rental services. We hope that by using this tool, the user will be able to comprehend the current state of the service they provide and how to improve it.

Users and the Questions

Characterization of the users and their context

The users this application has in mind are the administrative bodies of the San Francisco Bay Area bikes. They should be interested in understanding the main trends in the utilization of bikes and stations and how to improve it to meet their objectives.

Questions to Answer

When considering ways to improve the quality of the service, there are many questions to ponder. We have a list of questions that we hope the user can easily answer when using our application, as shown in the following example.

- For each type of client, which city will have the most traffic during a particular time interval?
- What is the incoming/outgoing traffic for a particular city/station and time interval?
- What was the ratio of incoming or outgoing traffic between two cities/stations during a particular time interval?
- Which bike was most frequently used during a particular period? How much?
- For a specific time interval, which client used the service the most?
- Was there any station that ran out of supply? If so, When?

Dataset

The dataset is accessible on Kaggle [1]. It contains a vast amount of information for 5 cities and 70 stations. Each file in the dataset is explained below.

- "station.csv": Contains information on each station. It contains its ID, name, coordinates, dock count, city and installation date;
- "status.csv": It has the status of each station for every second between 29/08/2013 and 01/09/2015. It contains the number of bikes and docks available for each station;
- "trips.csv": It contains information on every single trip. Start and end date and station, trip duration, bike id, and the type of client who used it;
- "weather.csv": It contains a large amount of weather information for various zip codes. It has max, min and mean values of temperature, dew point, humidity, sea level, miles of visibility and wind speed. It also has max gust speed, inches of precipitation and cloud cover.

The majority of the data in the dataset was put into use. The weather data was not included at all because it would make the application too cluttered and not provide much useful information when considering the complexity that it would bring.

As one can conclude by reading the dataset summary above, there is a lot of data. In fact, there is an astonishing 2 GB of data on "status.csv" only.
To guarantee a fast computation time on the client side (when filtering, for instance), some simplifying of the data was deemed necessary. Below are the files created to simplify the structure of the data:

- "metadata.json": This file contains miscellaneous data required by all visualizations. It has the name, coordinates, and city of each station. It also has the installation date and the number of docks. Additionally, it contains the minimum and maximum dates observed in the dataset, to populate a time interval filter;
- "status_small.csv": This file was created to reduce the amount of information in "status.csv". Instead of having the data aggregated in seconds, we aggregated it in days, effectively reducing its size to roughly 2 MB;

- "trip_small.csv": Here we aggregated the data by trips, and removed some attributes that weren't used in the visualization;
- "bike_usage.csv": Only the duration of each trip and the bike used, at the specified date, were extracted.

Visualization Solution

With the provided data, and the goals required by the users, we decided to develop 5 visualizations:

- a geographic map containing data regarding trips between cities and stations, as well as bike availability in each station. We thought this would be valuable since we can directly make use of geographical data in the dataset to associate it with other pertinent information such as trips and bike availability
- bar charts indicating information regarding trips from each city/station to other cities/stations
- bar charts to present overall trip information for each customer type
- a line chart showing the change in bike availability on a city/station over time
- a histogram presenting the distribution of usage of each individual bike (since each is uniquely identified)

We decided to distribute these visualizations in 3 different pages depending on context. For a geographic view, we used the map. For a detailed view on a specific city or station, we present the trip bar chart and the bike availability line chart. Finally, for a global view of the data we provide the client and trip bar charts, as well as the bike usage histogram.

All visualizations share the same set of filters, which are persisted between pages:

- Within/Between cities filter: consider only data about trips made within/between cities, or consider all trips
- Customer/Subscriber filter: consider only data about trips made by customers/subscribers, or consider trips by any client
- Time interval: view data that falls within a specified time interval

Not all visualizations used the same data. Since the filters don't affect all data, not all visualizations are affected by the filters. To indicate this to the user, whenever a filter is hovered we highlight the visualizations that are affected by that filter.

Additionally, the time filter contains a playback feature. By clicking on its respective button, next to the time interval selector, the selected time interval will automatically advance forward by 1 month every 1.5 seconds, animating the visualizations in the background. We thought this would be useful for the end users, since it can provide a sliding window view of the aggregated data presented in the visualizations in an animated way.

Low fidelity prototype and user feedback

The low fidelity prototype, which we did in paper, is slightly different from the final version. It included the 3 main pages containing the individual visualizations. The user can navigate between pages by interacting with the hamburger menu on the top left.

The filters shared between all pages are presented at the bottom of all of them. The time interval was interactable with brushing, while the other two filters were represented by slider toggles with 3 distinct states: two states at either end indicating a specific value (for instance, for the within/between cities filter, the within cities and between cities states respectively), and a state in the middle which includes both values.

Figure 1 shows the first page. It contained the map of the San Francisco Bay Area, with the city borders drawn in color and the stations represented by pie charts. These pie charts represent the average portion of bikes and docks available at each station. We thought a pie chart would be a good fit for this visualization since only two variables are being encoded and both of them are part of a whole (the station's total number of docks, both available and unavailable).

The trips are also represented by vectors going in both directions between stations. The total number of trips made in that direction is encoded by the vector's width. We decided to use vector width as the representation for the number of trips since only relative comparisons would seem relevant in this case.

The user can select a city or a station and view only trips that started or ended at the selected city or station. Hovering over a city or station presents their label, but all city and station labels can be shown at once by clicking on the "Show labels" checkbox on the right menu, which would allow the user to better select a city or station by its name.

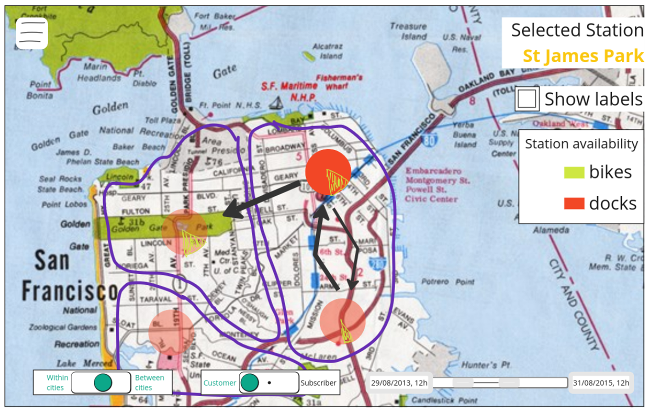


Figure 1: The prototype's first page, the map

Figure 2 shows the second page. It contained the trips bar chart, where the trips are only those that start in the city or station that has been selected. This selection is persisted between pages, so choosing a city/station in the first page will cause its details to be shown in the second page and vice-versa. In the second page, the user can also choose the city/station by clicking on the geographic marker next to the city/station name, which will open a popup with a map similar to the first page's allowing direct selection, but excluding extraneous information such as trip data and the pie charts.

The trips bar chart is stacked, and shows the number of incoming and outgoing trips in different colored bars for each city/station in the dataset. The X axis can be sorted in 4 different ways: alphabetic (city/station name), total number of trips, number of incoming trips and number of outgoing trips. The sort can also be ascending or descending by clicking on the checkbox. Additionally, when a station is selected, we allow the bar chart to be scrollable horizontally. When we analyzed the dataset, we found that even though we had 5 cities, there were 70 stations. This would make the representation of all 70 stations at once on the X axis cumbersome, so we allowed the chart to overflow horizontally in this case.

The bike availability line chart has two lines: the average number of bikes available and the average number of docks available at the specified city/station. We found solid lines to be appropriate in this case because the values presented are real numbers rather than integers, since they are averages.

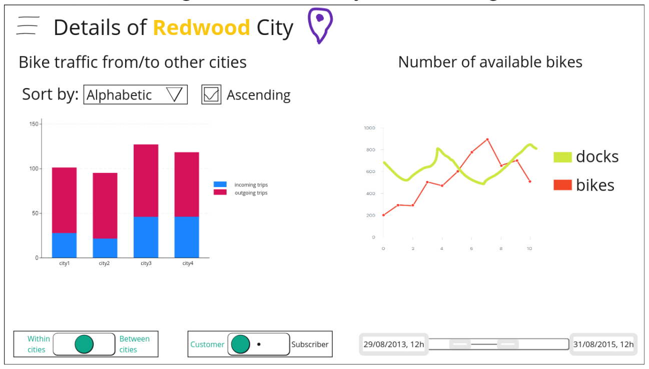


Figure 2: The prototype's second page

Figure 3 shows the third page, presenting global details. The trips bar chart is also present, but here all kinds of trips are considered, and not only those that start/end at a specific city/station. We can specify whether we want to view trip data in terms of cities or in terms of stations, similar to the second page.

Two bar charts are present, which show the total number of trips and average duration of trips made by each client type.

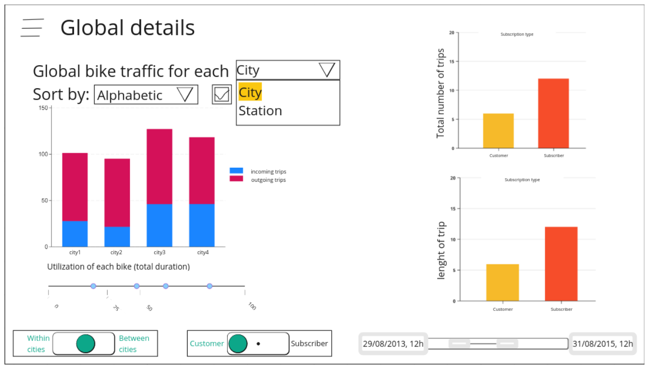


Figure 3: The prototype's third page

Finally, to allow the user to understand how each individual bike has been used, a 1D plot was meant to be used, which would allow the user to quickly perceive outliers requiring special attention. The 1D plot contained an axis representing the total amount of time that the bike has been used for, and each dot placed on this axis represented a single bike. Hovering over these dots would indicate the bike's ID.

The low-fidelity prototype was tested with volunteers. The questions to be answered were specific instances of the general questions presented in the Users and the Questions section:

- Which is the city with the most inner traffic in the Summer of August 2014, considering traffic from any kind of client?
- Which was the most used bike in 2015?
- Was there any station without available bikes in 2015?
- Which type of client used rental bikes the most in 2015?
- What is the exact incoming bike traffic between the topmost station of Mountain View city and station "San Antonio Shopping Center" in 2015?

After we tested the prototype with volunteers, we got some pertinent feedback:

- For the map visualization, it wasn't clear whether the areas or the pie charts were cities/stations. We believe this is also due to the poor interactivity of the low-fidelity prototype. In the final application, we intended for this to be explicit when the user hovered over the area or pie chart.
- For the stacked bar charts, provide a tooltip when the user hovers the bars showing the absolute value. This is important, since the stacked bar chart doesn't easily tell the absolute values that they represent for bars that are on top of others. Another suggestion that was proposed to us was allowing the user to specify whether they want to see the bar chart stacked or not. In the implementation of the functional prototype, we found out that this was in fact a necessity;
- For the bike usage visualization, for which we proposed a 1D plot, we were suggested using a different chart. A box plot was recommended which would allow a better

perception on the distribution of bike usage than a cluster of points. If details were needed, the box plot could be interacted with by clicking a specific quartile, showing a 1D plot of points in that quartile. This was suggested with Shneiderman's information seeking mantra in mind, allowing an overview over bike usage and then permitting filtering over a quartile, after which bike IDs can be queried by the user on hover. We left this idea for consideration, but it indicated us that the 1D visualization needed some rethinking;

- The map visualization's pie chart legend could be more descriptive rather than ambiguous;
- It was suggested to us that we include the bike availability line chart in the third page as well, which houses global dataset details. We thought this would be unnecessary and uninformative, since we are considering averages of bike and dock availability over the entire Bay Area, which would result in an overall unchanging chart (if bike availability goes down in a station, then another station would obligatorily have bike availability going up since bikes shouldn't disappear in a station-to-station trip);
- Provide a tooltip for each filter containing a description of what they do. One of the testers found it difficult to understand what some of them do, especially the within/between cities filter.

We ended up incorporating most of the suggestions. The only ones that weren't included were the box plot for the bike usage visualization and the bike availability line chart in the global details page.

Functional prototype

The functional prototype was implemented in HTML, CSS and JavaScript, and the D3.js library was used [2]. For styling and some navigational functionality for the web application, Bootstrap was used [3]. The code for this project is present on GitHub [4].

The project is organized in folders: "css" contains the CSS code for styling the web application, "data" contains the preprocessed dataset, "scripts" contains data preprocessing scripts and "pages" contains the HTML and JavaScript source code. "index.html" is the starter file, which simply redirects to "pages/intro.html".

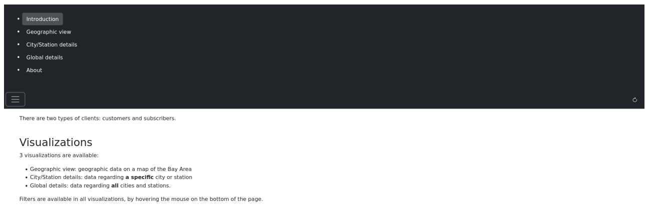
Each page has its specific HTML file in the "pages" folder. The "pages/charts" folder contains D3 code that is reused between pages (for instance, the trips bar charts that are used on the second and third pages). The "base" folder simply contains HTML for the navbar and the filters, which are used across pages, and the "base.js" script is JavaScript code that is necessary in all pages. For specific page functionality, a script block is included in its respective HTML file.

The D3 code is organized similarly between the script blocks of all HTML pages. Each visualization has a specific function "vis<Name>", which is responsible for initializing the chart and returning a function that accepts a set of data and draws it on the initialized chart. This way, these returned functions can be called later when filters are applied without recreating the chart again.

The data is loaded at the end of the script block. After loading, "draw<DataType>" functions are called which initialize all visualizations requiring the specific "DataType" (for instance, trip data). The draw functions also listen for "data-update" events which are triggered when data is filtered. When the "data-update" event is received, the methods returned by the "vis<Name>" functions are called, updating the visualizations.

This centralized design allows the data to be filtered only once at the "draw<DataType>" functions and then be distributed to each visualization draw function.

Regarding the web application itself, figure 4 shows the overall design of the application. It is navigable using the hamburger menu as in the first prototype. An "introduction" and "about" pages are also included. "Introduction" presents the dataset and the visualizations that are offered in the application, while the "about" page presents references and author information. Additionally, on the top right of the page, a button to reset filters to the default values was included, which wasn't considered in the first prototype.



There are two types of clients: customers and subscribers.

Visualizations

3 visualizations are available:

- Geographic view: geographic data on a map of the Bay Area
- City/Station details: data regarding **a specific** city or station
- Global details: data regarding **all** cities and stations.

Filters are available in all visualizations, by hovering the mouse on the bottom of the page.

Figure 4: Introduction page and navigation bar

Figure 5 shows the filters box present in all visualization pages. The filters are as presented in the first prototype, with small info icons right next to each one providing brief descriptions of what the filter does on hover.

Additionally, the time interval filter contains a play button on the right which permits the playback feature mentioned previously. By clicking on it, the playback is activated and the time interval is automatically advanced over time, until the button is clicked again or the maximum date is reached.

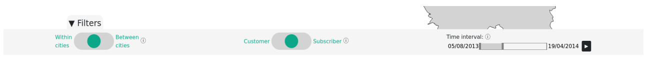


Figure 5: Functional prototype filters

Figure 6 presents the geographic view, which is the first page of the prototype. It shows a map of the San Francisco Bay Area, which is drawn using GeoJSON data obtained externally. For the city borders, San Francisco city already had borders included in this GeoJSON, but we needed to draw the borders of the other dataset's cities by hand since we could not find border data online.

The visualization mostly follows what has been presented in the first prototype. Zooming and panning can be performed to get a better view of the stations in each city, and the user can also hover their mouse over cities and stations to know their name. Hovering over stations also shows the absolute value of each slice of the respective pie chart.

Some changes were made since the first prototype. Firstly, the trip vectors now only encode the number of trips rather than the number of trips and the direction of those trips. Since the map already gets crowded with vectors when direction is not represented, we decided to simply encode the number of trips by the vector length to avoid cluttering the visualization. Additionally, we don't allow the option to present all city and station labels at once to allow visual selection by city or station name. Due to implementation issues, mainly in how the labels were going to be positioned, we decided to allow selection by name on the X axis of the trips bar charts of the second page, which will be detailed later.

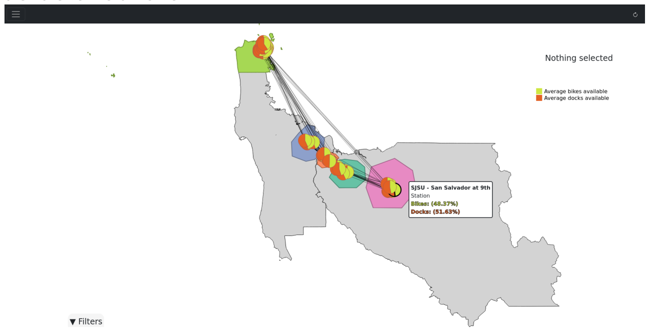


Figure 6: First page, geographic view

Figure 7 shows the second page, which contains details regarding a specific city or station. As was projected for the first prototype, the city/station selection is persisted between the second and first pages. Still, a city or station can also be selected in this page using the map by clicking on the pin button at the end of the page's title, opening a popup with a small and simplified view of the first page's map. Most of the other features detailed in the first prototype were implemented here.

On the left is the trips bar chart for the selected city/station. The bars can be hovered and a tooltip is shown containing the absolute value that they represent. The X axis is interactable, and clicking on an unselected city/station selects it. This should aid the user in selecting a city or station by name, since the X axis can also be sorted alphabetically, and it also allows the user to inspect other cities or stations directly while comparing them in the bar chart.

Also, a checkbox that allows toggling between the stacked and unstacked versions of the bar chart was added, as suggested in the first prototype's feedback. Something worth mentioning is the fact that the trips data when comparing cities is very unbalanced, with San Francisco city having a disproportionately larger number of trips compared to the other cities. This led us to implement the Y axis on a logarithmic scale, which makes data for other cities much more perceptible. On the other hand, this meant that a stacked bar chart no longer made sense when comparing cities. As such, the checkbox is disabled in this case. An informative tooltip is next to this option in order to explain to the user why the checkbox is disabled when comparing cities.

The bike availability line chart is to the right. The first prototype included both the bike and dock availability lines, but since one is the inverse of the other we decided to use only one of them (the bike availability line). Additionally, in order to show the dock capacity of each station, we included a red horizontal line showing that value. The line chart's individual dots can also be hovered in order to get a tooltip indicating the specific X and Y values at that point (date and average bikes available, respectively).

The user can grab and move the legend, since it can obstruct the chart's data. It's worth noting that, when a city is selected, the legend changes to modify the meaning of the red horizontal line. When a station is selected, it means the station's maximum capacity, i.e. the number of docks. A city, however, doesn't have a number of docks associated. We could sum up the capacity of all stations each city contains, but this results in a very large number that flattens the bike availability line. As such, for the cities, we consider only the maximum bike availability value for the red line, and change the legend to indicate that.

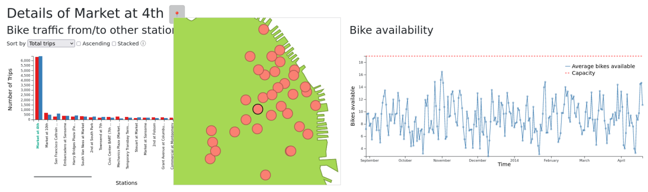
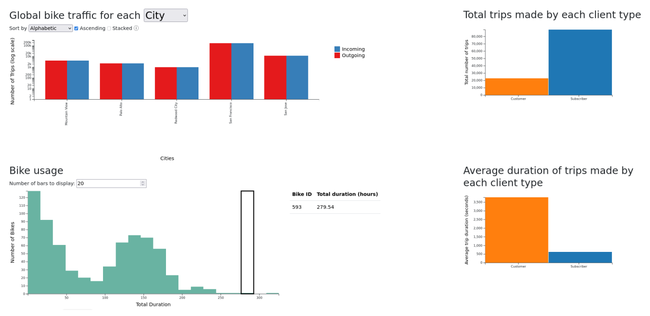


Figure 7: Second page, city/station details

Figure 8 shows the last page, containing general data regarding all cities and stations. A trips bar chart is present similar to the second page's, with the added dropdown that allows choosing whether we want to compare cities or stations. The client type bar charts, showing the total number of trips and average trip length of each client type, are also present on the right.

On the bottom left, we have a histogram showing the bike usage distribution. In the first prototype, we wanted to use a 1D plot to represent the bike usage, but during implementation of the functional prototype, we were suggested to use a histogram instead, which would allow a much better view into the distribution of the data. Since the histogram representation is heavily dependent on the data, we also allow the user to specify the number of bins in a number input above the histogram. Also, in order to know which bikes belong to which bins, which is crucial for the use case of knowing over- and under-used bikes, we allow selection of each drawn bin. Selecting a bin populates a table on the right, listing the bikes present in the bin, and showing their ID and total usage in hours. The bin selection box spans the entirety of the histogram's Y axis, to allow selection of bins that contain an extremely low number of bikes, which is imperative.



Bike ID	Total duration (hours)
593	279.54

Figure 8: Third page, global details

Implementation challenges

One of the main problems when implementing these visualizations was in figuring a modular design for the D3 code. Data was supposed to be filtered only once on each filter update, and then that piece of data had to be distributed to the other visualizations to update their state. Even then, the modularity and centralized design as implemented brought awkward behavior, such as the client type bar charts being affected by the client type filter, which doesn't make sense. Choosing a single client type discards the other client from the bar chart, showing a bar chart with a single bar. However, we found that, in trying to implement these exceptions, it would be complicated to coordinate the data between visualizations, because while the client type bar chart can't use a version of the data that is filtered by the client types, the other visualizations require that, which means that multiple copies of the data have to be present at once. Due to the large complications that fixing this would bring compared to the time available for large refactoring, we decided to leave this issue as is.

Additionally, we found difficulty in representing all cities in the map. We could find a GeoJSON file containing the San Francisco city's borders, but not for the rest. This required us to roughly draw the borders of the cities by hand.

Finally, the "show labels" feature that was supposed to be present in the map and would show the labels of all cities and stations at the same time presented implementation issues regarding proper positioning of the labels within the SVG, depending on the pie charts for the stations but also on the polygons for the cities. Due to this, we decided to opt for another feature that would help the user achieve the same goal, which is the selection on the X axis of the second page's trips bar chart.

Evaluation and changes in the prototype

The heuristic evaluation had Nielson's 10 general usability heuristics in mind: visibility, match, control, consistency, prevention, recognition, flexibility, minimalism, recover and help [5].

- Visibility of System Status

It was pointed out that the pie charts that represented the cities on the geographic view lacked information on the percentage of bike availability (severity 2). We then added the percentages to the tooltip.
It was also pointed out that the geographic view visualization lacked a title (severity 2). We ignored this problem in favor of fixing other more pertinent issues, especially since the current page title is already shown when the hamburger menu is opened and in the web browser's tab.
Another issue was that the histogram lacked labels on the X- and Y-axis (severity 4). This suggestion was implemented.

- Match Between System and the Real World

Before the evaluation, instead of "Number of bars to display" above the histogram, it was "Number of bins". Our colleagues advised us to change it, as it could not be clear what the number of bins was for a non-technical user, as "bins" may be technical jargon (severity 2).

- User Control and Freedom

They also noticed that our visualizations lacked a reset button, to clean the tweaks made by the user (severity 2). The button was added after the evaluation to the top right of all pages, on the navigation bar.

- Aesthetic and Minimalistic Design

Our colleagues thought that the charts were too close to the border, which resulted in a not very attractive design (severity 1). We added padding to solve this problem.
Additionally, the second page had too few and small graphs which don't occupy the vertical empty space efficiently (severity 1). We were suggested to use relative dimensions that depend on the size of the page, but we discarded that since it's resolution dependent and had more urgent improvements to make. This way, chart dimensions are also homogenized between the second and third pages.

- Help and Documentation

Here it was pointed out that the home page lacked some explanation on how to start using the application (severity 1). We agreed and added a little instruction to click on the hamburger button in the top left corner.
Also, the graphs didn't have descriptions of what information they were showing, and some of them had unhelpful or wrong labels, which confused the user. For instance, the "Number of bins" input that was mentioned previously (severity 3). We ended up fixing the incorrect labels.

Conclusion and Future Work

With this project, we created an application that helps explore and better understand the San Francisco Bay Area bike share dataset.
After analyzing the dataset and defining a target user, we proceed to create the primary problems and questions that this visualization should be capable of answering.
Then we proceeded to do some preprocessing of the data, as the raw data contained too much information, which would make a smooth experience impossible when using the visualization.

We created a web visualization in HTML using D3 once the data was ready, with three main pages: geographical view, city/station details, and global details.
Finally, a heuristic evaluation was conducted by a group of colleagues, which helped get feedback and refine our work.

There are still some issues that need to be addressed in the future due to time constraints. Some charts should not be modified by certain filters, as it makes little sense. The problem is a result of the code's modularity and design. The data filtering process also needs improvement, as it takes some time to update the charts when the filters are tweaked. Finally, the line chart for bike availability could be smoothed, as when large time intervals are considered, the chart becomes difficult to read due to the data changing rapidly.

References

[1] SF Bay Area Bike Share. (2023, November 27). Retrieved from https://www.kaggle.com/datasets/benhamner/sf-bay-area-bike-share/data

[2] D3 by Observable | The JavaScript library for bespoke data visualization. (2023, November 08). Retrieved from https://d3js.org

[3] Mark Otto, Jacob Thornton, Bootstrap contributors. (2023, September 14). Bootstrap. Retrieved from https://getbootstrap.com

[4] vi-project. (2023, November 27). Retrieved from https://github.com/martinhoT/vi-project

[5] 10 Usability Heuristics for User Interface Design. (2020, November 15). Retrieved from https://www.nngroup.com/articles/ten-usability-heuristics