Coding Challange - **Sushanth Thangamani**

Coding Exercise
The coding exercise is catered to solving some practical problem that we are currently addressing. The purpose of the coding challenge is to get to know how much hands on knowledge you have with respect to D3 and also how fast are you in learning something new for eg: React.js. **There will be a book that will be provided to you for your help**. The deadline to submit the result is **7th Nov 2016. You are free to submit it before that.**

You can feel free to ask as many questions as possible and we can address them via phone, video chat or also in person.
The coding challenge is divided into 2 parts

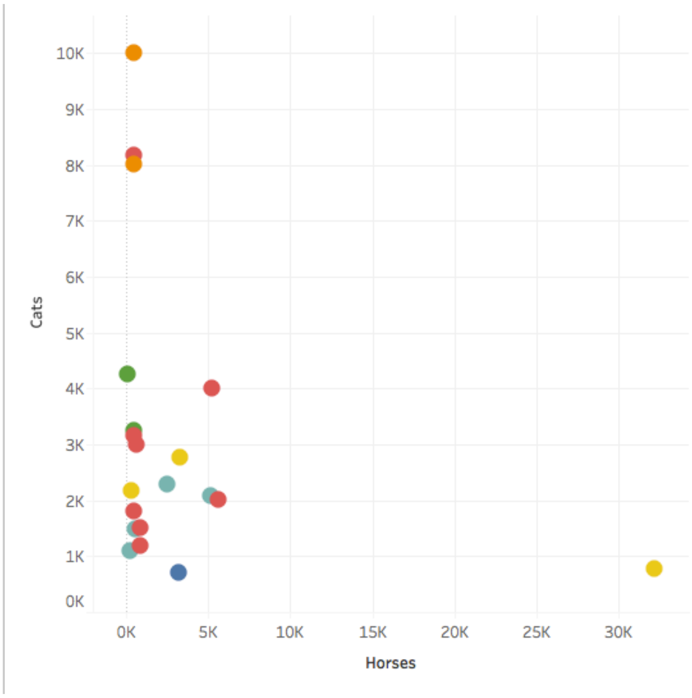
1. D3 - Mandatory
2. React JS - Bonus question.

Part 1: D3 - **Data will be provided in the mail.**

---

1. Create a simple scatter plot with the following encodings
   x-axis: horses
   y-axis: cats
   color: gender

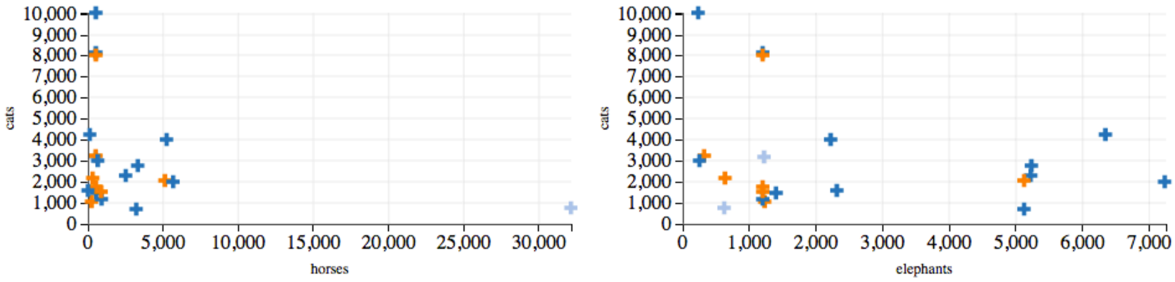
Graph attached here



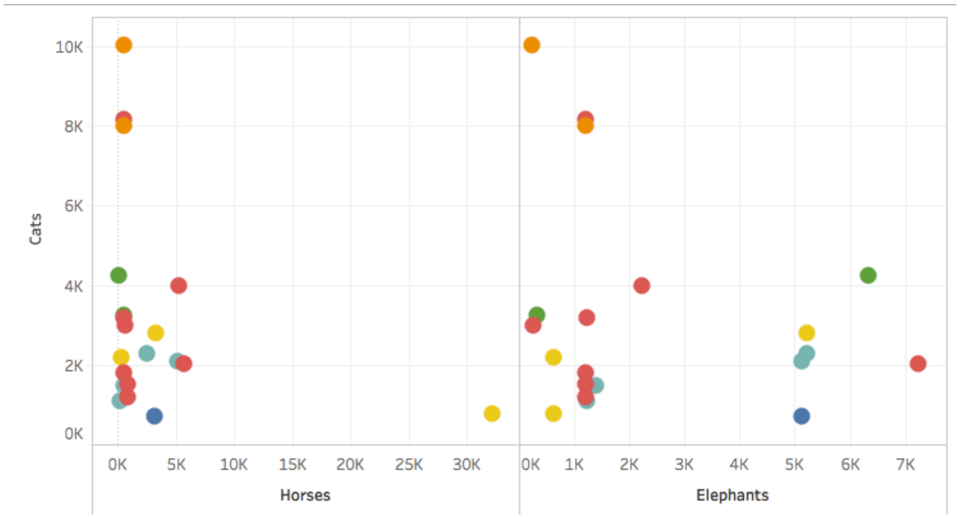
2. Create 2 scatter plot(Side by Side)
   x-axis: horses, elephants
   y-axis: cats

color: gender

Something like this.



3. Since both the scatter plots share the same y-axis let them share a common axis.
Note: Don't hide the axis of the other chart but try to implement some logic because of which both the charts
have the same yAxis.
Graph attached here.

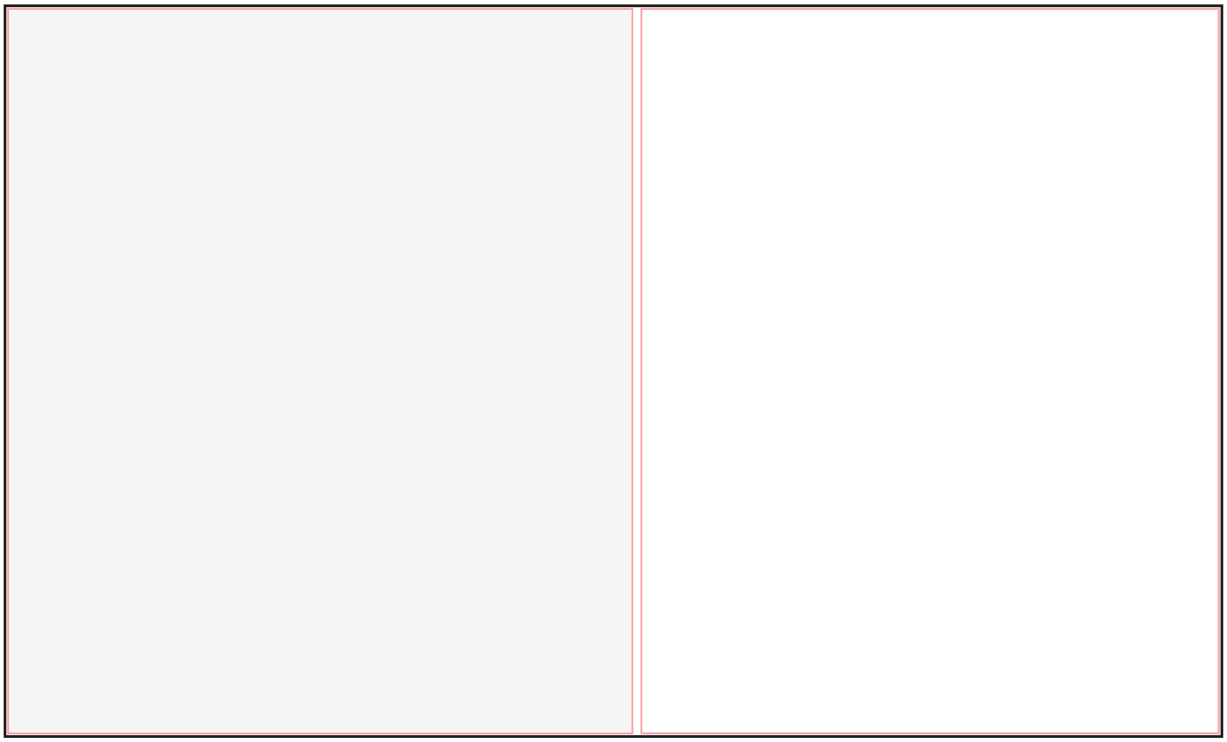


BONUS EXERCISE

1) **Create a React JS application with a parent container component and two child components:**

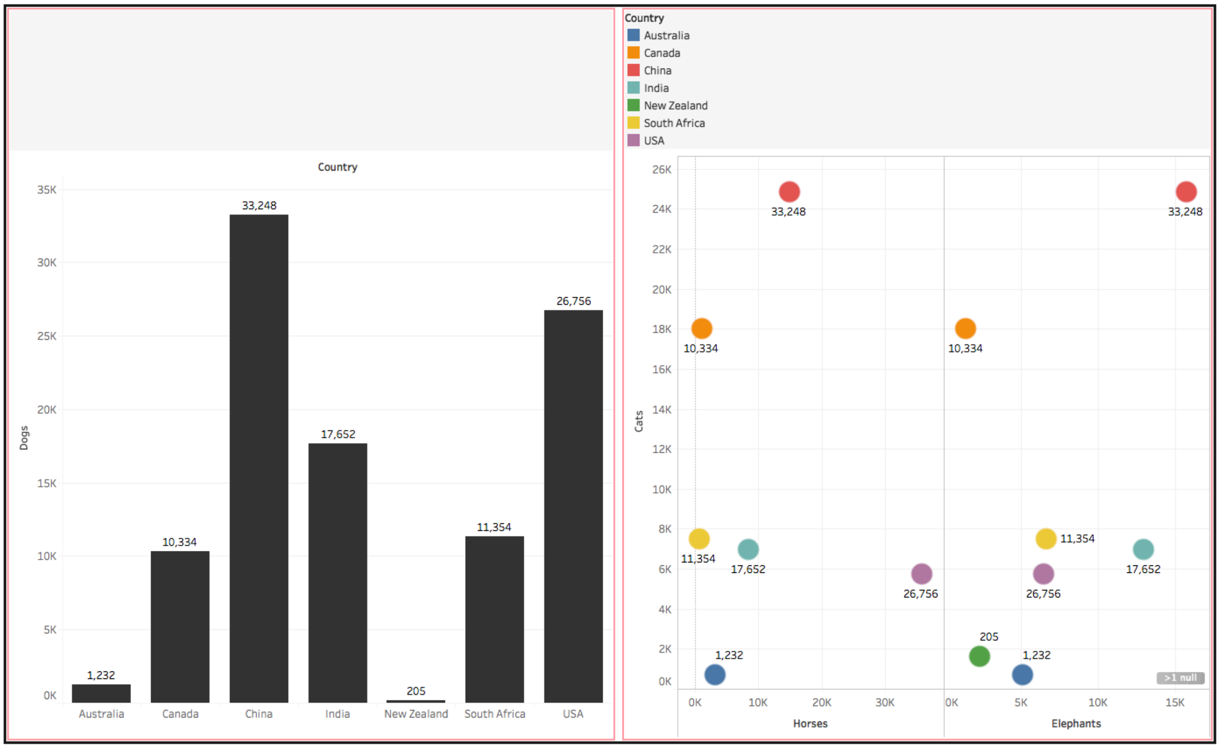
The following image is an example of a parent container with black borders, and two child divs (React components) in pink borders.

React JS requires a server side application to run, so as to be able to compile JSX code in React. You could use any minimal server side language (Java/NodeJS/Python Flask) to support your front end application.



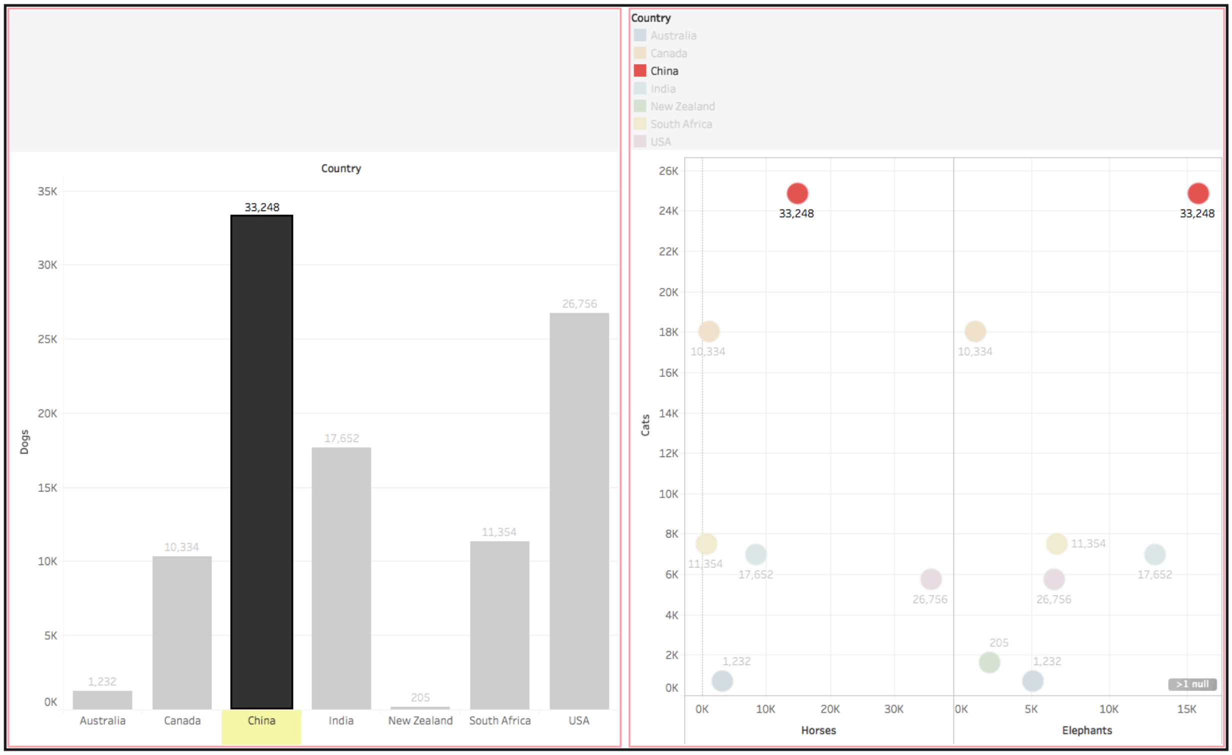
2) **Render the scatter plot you had built in one of the child components. Also, render a bar chart of "Dogs by Country" in the other child component.**

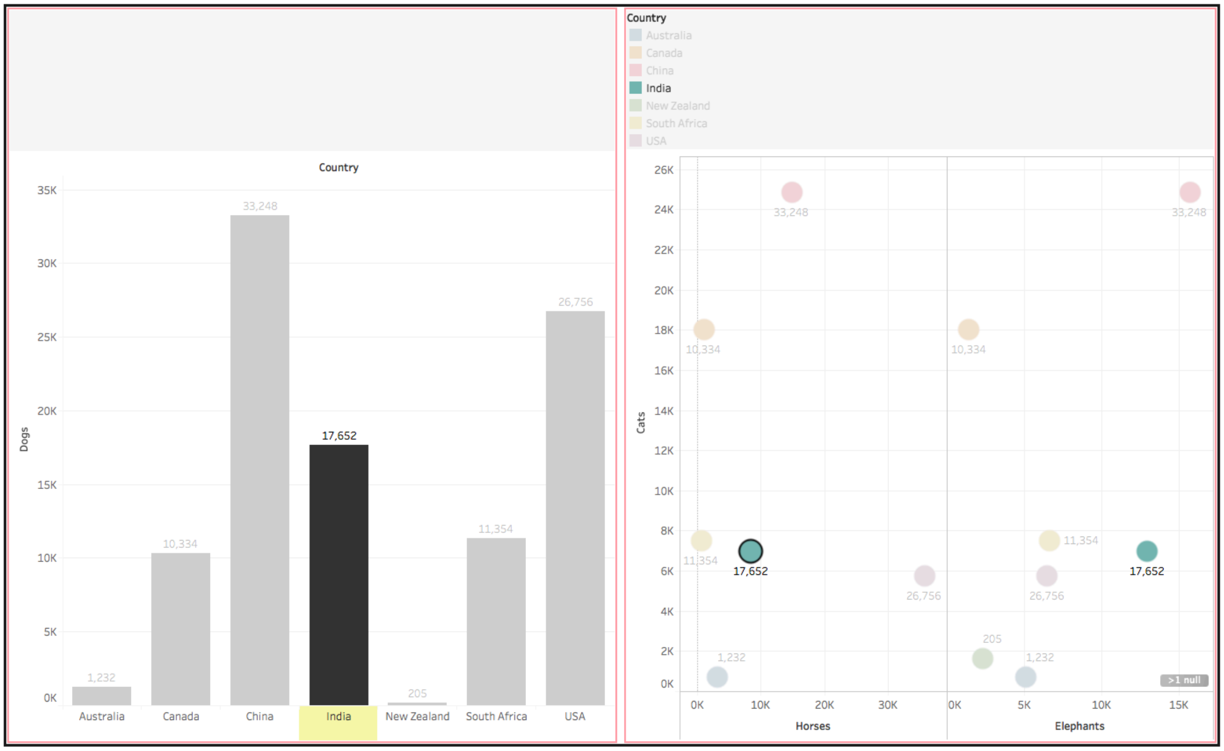
Please refer the image below to see how your implementation could look like.



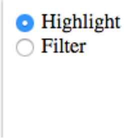
3) **Add both-sided interactivity — clicking on a bar highlights the associated dots on the scatter plot, clicking on a dot highlights the associated bar in the bar chart.**
In the following two images, you can see:
i) clicking on the bar (country: China) highlights all dots of China and makes all other dots transparent.
ii) clicking on a dot (country: India) highlights the bar of India.

It's highly advisable to handle the interactivity on React JS.

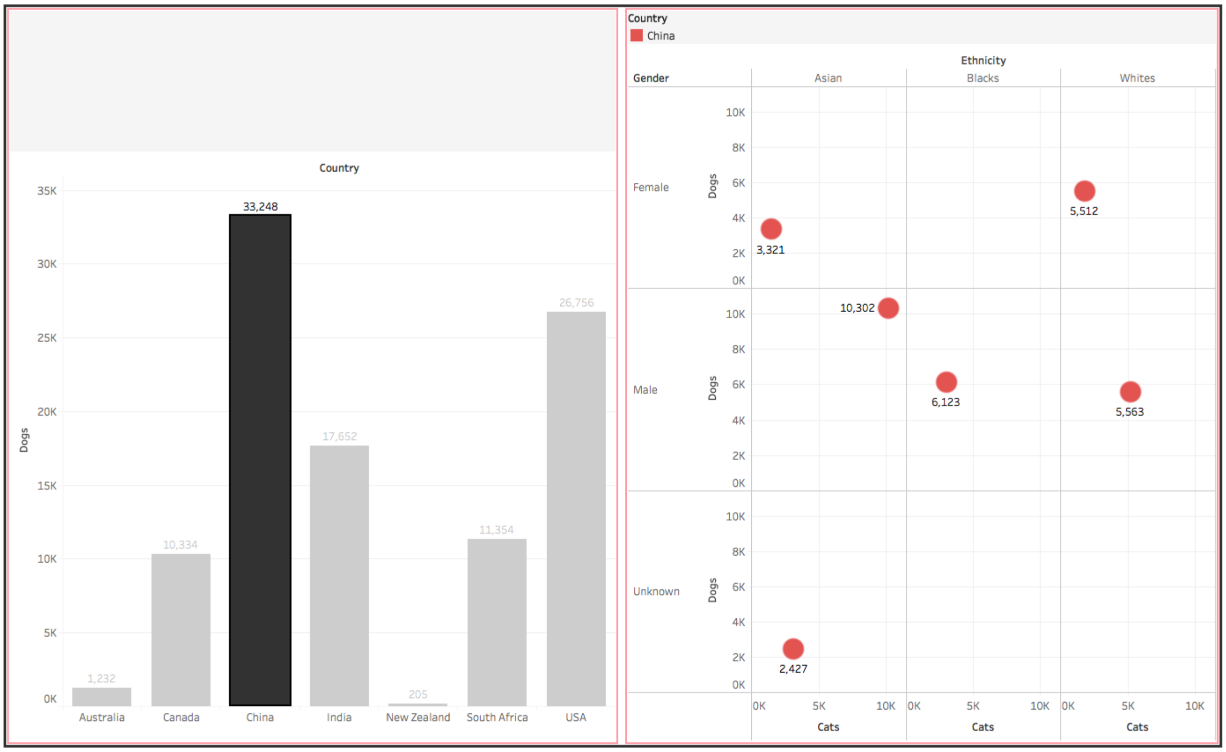




4) **Give the user a filter to select the kind of interactivity — Highlight or Filter**
This will be a third child (React component) to the parent container.

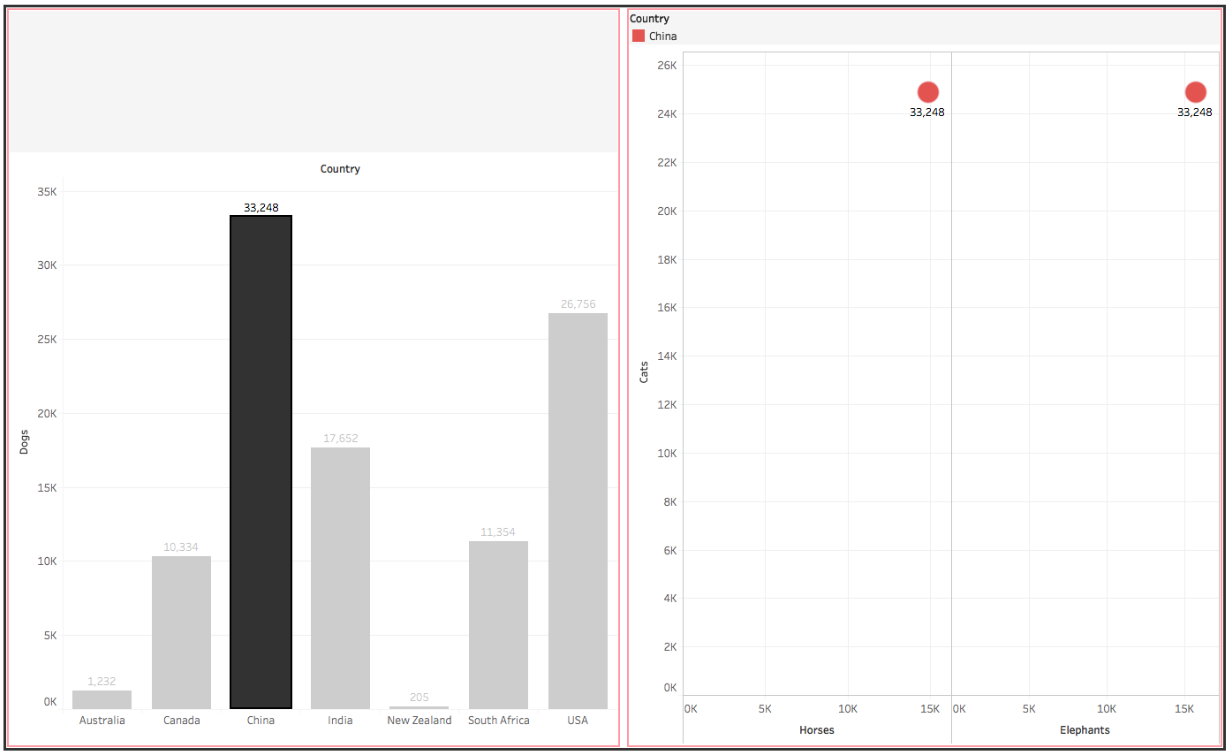


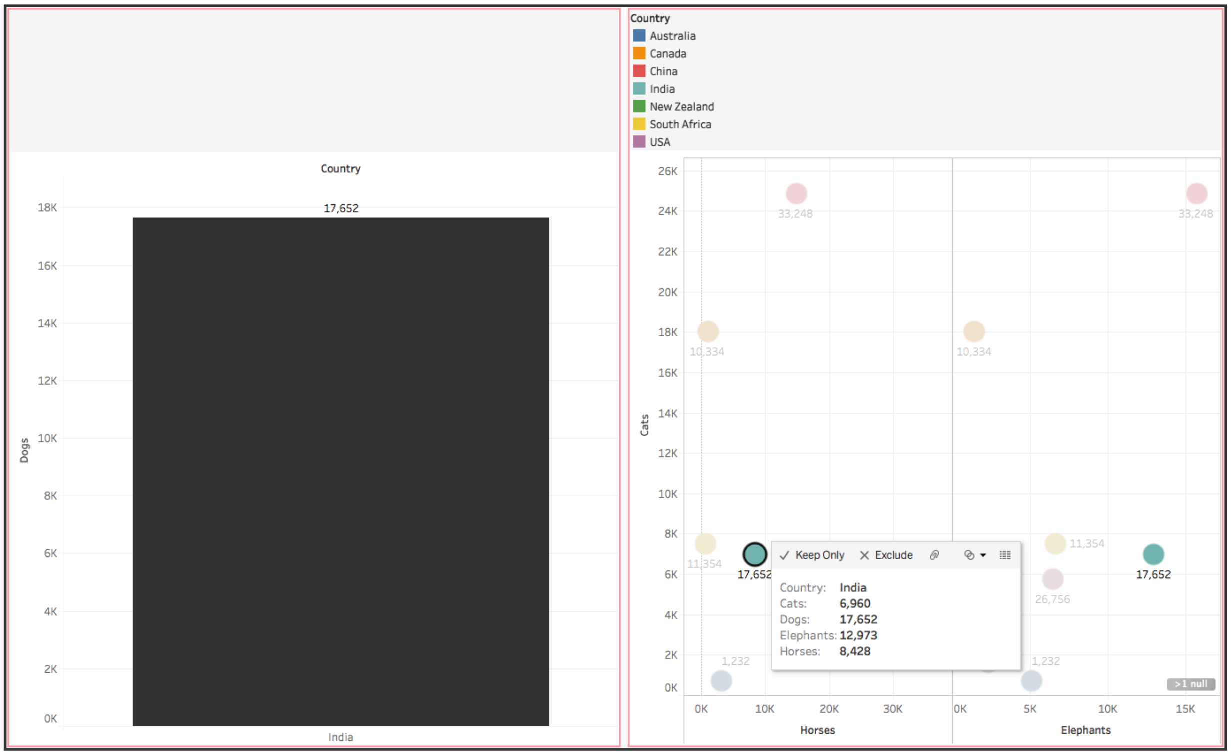
● Highlight
○ Filter



5) **When “Filter” is selected by the user, the chart should demonstrate a different kind of interactivity — clicking on a bar filter the data of the scatter plot, clicking on a dot in the scatter plot filters the data in the bar chart.**
In the following two images, you can see:
i) clicking on the bar (country: China) filters data in the scatter plot. to only dots of China and resizes the axis scale.
ii) clicking on a dot (country: India) filters data in the bar chart to only one bar of India.

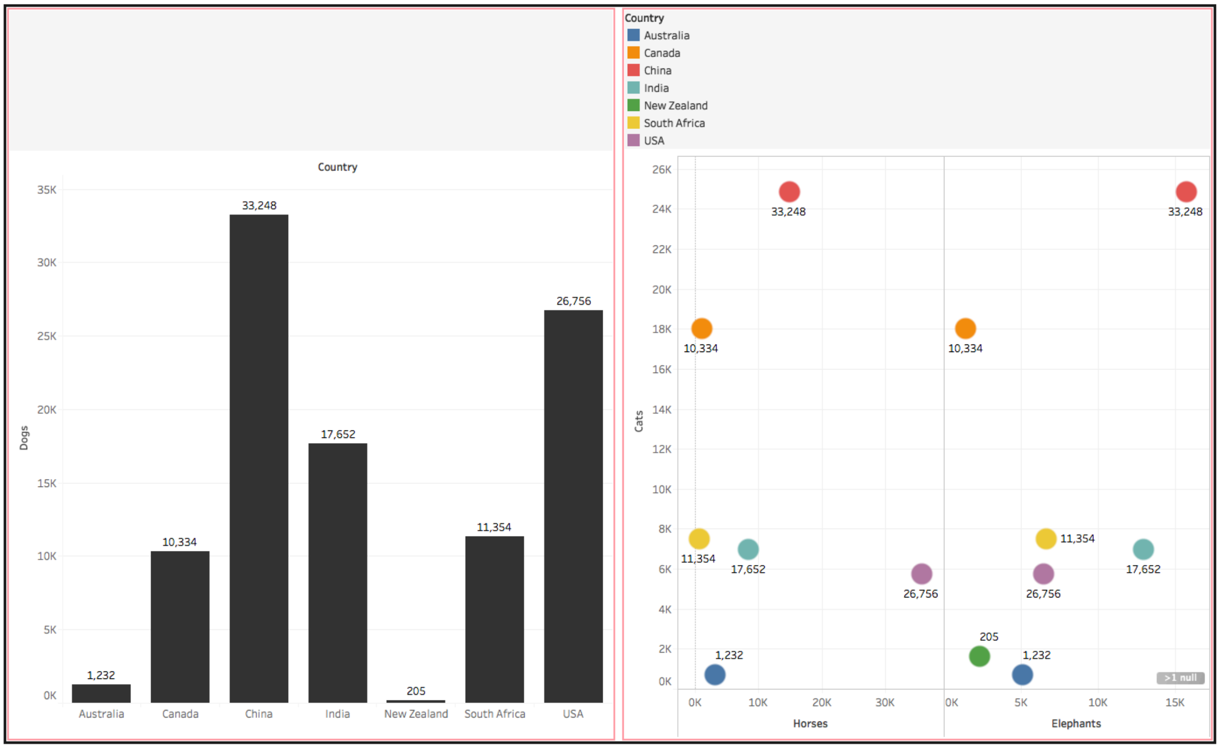
It’s highly advisable to handle the interactivity on React JS.





Country: India
Cats: 6,960
Dogs: 17,652
Elephants: 12,973
Horses: 8,428

6) **Unfiltering/Unhighlighting the dots or bar brings us back to the old charts with all data.** Please refer image below.


Country
Dogs
35K
30K
25K
20K
15K
10K
5K
0K
1,232
10,334
33,248
17,652
205
11,354
26,756
Australia
Canada
China
India
New Zealand
South Africa
USA
Country
Australia
Canada
China
India
New Zealand
South Africa
USA
Cats
26K
24K
22K
20K
18K
16K
14K
12K
10K
8K
6K
4K
2K
0K
33,248
33,248
10,334
10,334
11,354
17,652
26,756
11,354
26,756
17,652
205
1,232
1,232
>1 null
0K
10K
20K
30K
Horses
0K
5K
10K
15K
Elephants