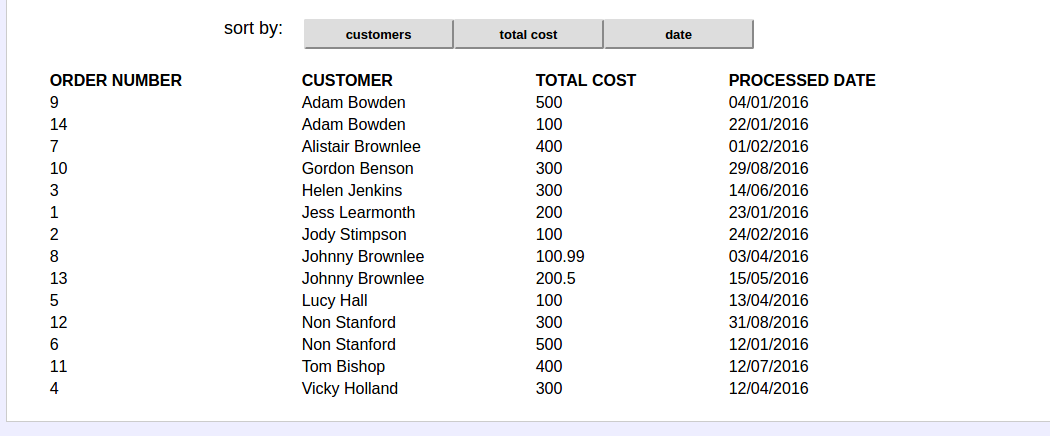
**Tinder Foundation App Report**  
Christian Walker-Spiers

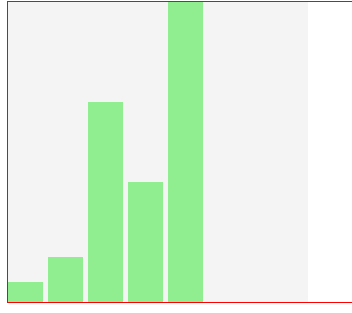
**Dependencies**  
- jquery-1.12.2  
- d3.v4  
  
**Production Tools**  
- Sublime Text 3  
- Chrome  
  
**Folder Structure**  
 Tinder-app

index.html  
 main.js  
 style.css  
 Report.docx

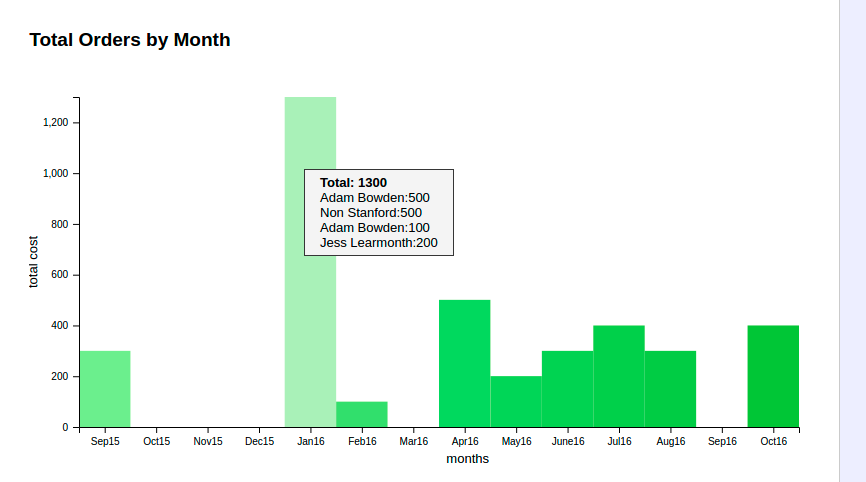
**Process**  
  
**Objectives 1,2,& 4 (render and order data)**  
  
**1) Loop through the given JSON data**  
- console logged the JSON data so I knew the code was looping through the objects correctly  
 - had to make a minor adjustment to the JSON as there were missing commas after total cost  
  
(screen 1 ) - then created a loop for creating the table content and rendered that into html on page load  
  
  
**2) Sort data**  
then I googled - 'javascript order array of objects by key' as first search and found best solutions on stack overflow  
- reminded myself of the sort function and tested its use by sorting the array by name alphabetically  
  
**3) Make a master sort function**  
- made an encapsulating function where the key can be passed as a variable  
- created a separate function for converting the date string to the d/m/y format so it could be converted to a JS date.

  
*(screenshot 2) - the bare bones of html layout*

**4) Refactoring and optimisation**   
- decided to hard code the table head thead into the html as opposed to creating it within the js loop for obvious performance reasons  
- modified the functions so that they were all providing return values (this way you can easily link functions together. This is a cleaner method than just having functions modifying global variables   
 - the original JSON data doesn't get changed this way  
  
  
**Objective 5 (turn data into a chart)**  
  
**1) Research and planning**  
- originally decided to hard code the table using varying length divs   
- researched some more and found that D3 seemed to be one of the most popular charting libraries  
- read the basic documentation then followed a youtube series about creating SVGs.  
- then followed on to look at how to create a basic bar chart  
  
**2) creating svg shapes with D3.js**

  
*(screenshot 3) creating basic svg rectangles using D3.js*  
  
- followed a tutorial for inserting rectangular static shapes to a svg canvas  
- then learnt how to position a series of rectangles along the width of the canvas (the bars of the chart)  
- each rectangle had a fixed width   
  
**3) making variable scales for the y and x axis and bar widths**  
  
- var xScale = d3.scaleBand() // ordinal means you can have specified intervals  
 .domain(d3.range(0,myData.length)) // the  
 .range([0,width]) // defines the x area for the bands to be created  
var yScale = d3.scaleLinear()  
 .domain([0,d3.max(myData)]) // this is the range of the values 0- max of data  
 .range([0,height]) // the height of the original svg  
(encountered some errors as some of the information I was reading was using outdated syntax)  
  
**4) Colouring the bar charts with a gradient**  
- applied a linear scale using two colours as ranges.  
- applied the colour style passing the increment as a variable for each rectangle  
  
**5) Developing the tooltip**  
- used mouseover events to trigger the display of a tooltip to provide information about each individual bar  
  
**6) create function for calculating all of the information needed for the bar chart**  
- firstly had to create an array of objects for all of months between the earliest and latest date  
- for each of these months the total cost was calculated and each individual customer purchase was stored in an object ‘customers’

**7) updated tooltip to provide information on each customer purchase**



*(final graph) - showing tooltip on mousehover*

**Final Thoughts**  
  
The project on a whole was relatively straightforward and not too many problems were encountered in terms of understanding the logic needed for the application. There was already familiarity with dealing and presenting JSON data so the core tasks were completed within a short time. Knowledge from recent Node.js learning was beneficial when thinking about the architecture of the code. More specifically, cleaner functions with return values or callback functions (operating in the same vein as Node’s callback pattern) alongside an awareness of trying not to pollute the global namespace.  
  
The main difficulty was with learning the D3.js library. However no more so than when learning anything new as it just involved the usual hurdles of finding good reading resources and practical examples of the technology. It was great to be exposed to this powerful resource and there is hope to use such technology again in the future.  
  
If I could do it again...  
  
 1) I would change the app so that whenever the JSON data is updated the table and graph changes dynamically in accordance to the new data. I was beginning to research about MVC design patterns but felt it may have been overkill for the project requirements.

2) I would spend a little longer with D3.js trying to understand it the fundamentals in a way where I would be comfortable building a chart from scratch (without tutorial aid). This would better help solidify my understanding of the technology. I really enjoyed working with it.  
  
3) I would experiment with some different designs for the page content and include some of Tinder Foundation branding just purely for aesthetics.

4) would fix the tooltip hover issue (when you hover over the tooltip the browser removes it because you are no longer hovering over the bar)  
  
  
*Christian Walker-Spiers*