

NO.	Practical Name	sign		CO
1	Setup Environment for All the Tools Install Wamp/XAMPP Server Install JQuery Library Install D3.js and Canvas.js Configure Google Chart API and Google Map API			1
2	Develop the following Program Using HTML5 CANVAS Develop the Different basic Graphical Shapes using HTML5 CANVAS Develop the Different Advanced Graphical Shapes using HTML5 CANVAS Develop the Different basic Graphical Shapes using HTML5 SVG Develop the Different Advanced Graphical Shapes using HTML5 SVG			2
3	Develop Following Program Using HTML5 and JavaScript Read the data .txt file and draw Data Table Read the data .csv file and draw Data Table Read the data XML file and draw Data Table Read JSON Data and draw Data Table			3
4	Develop Following Program Using HTML5 and JavaScript Develop the simple bar chart using HTML5 CANVAS Read the data .txt file and draw Simple Bar Chart Read the data .csv file and draw Column Bar Chart Read the data XML file and draw Simple Chart Read JSON Data and draw Simple Chart			4
5	Develop Following Program Using HTML5 and D3.js and Canvas.js Showing the data as a column chart (simple) Showing the data as a stacked column chart Showing the Data as a column chart for four age group			
6	Showing the data as a Line chart (single, fewer and multiple lines) Showing the data as a Pie Chart (single and multiple pie) Showing the data as a Bar Chart (Simple and multiple)			
7	Develop Following Program Using HTML5 and Google Charts API and Map API Using Google Charts API Basics draw charts like a Bar chart Using Google Charts API Basics draw charts like a Line chart Using Google Charts API Basics draw Pie Chart			
8	Develop Following Program Using HTML5 and Google Charts API and Map API Using Google Charts API Basics draw Donut Chart. Using Google Charts API Basics draw Candle Chart. Using Google Charts API Basics draw other types of Chart. Using Google API read JSON file and create Google Map.			
9	Case Study for Different Information Dashboard(Min 2)			
10	Build interconnected Dashboard Using Different Library for any one enterprise Solution			

## Practical – 1

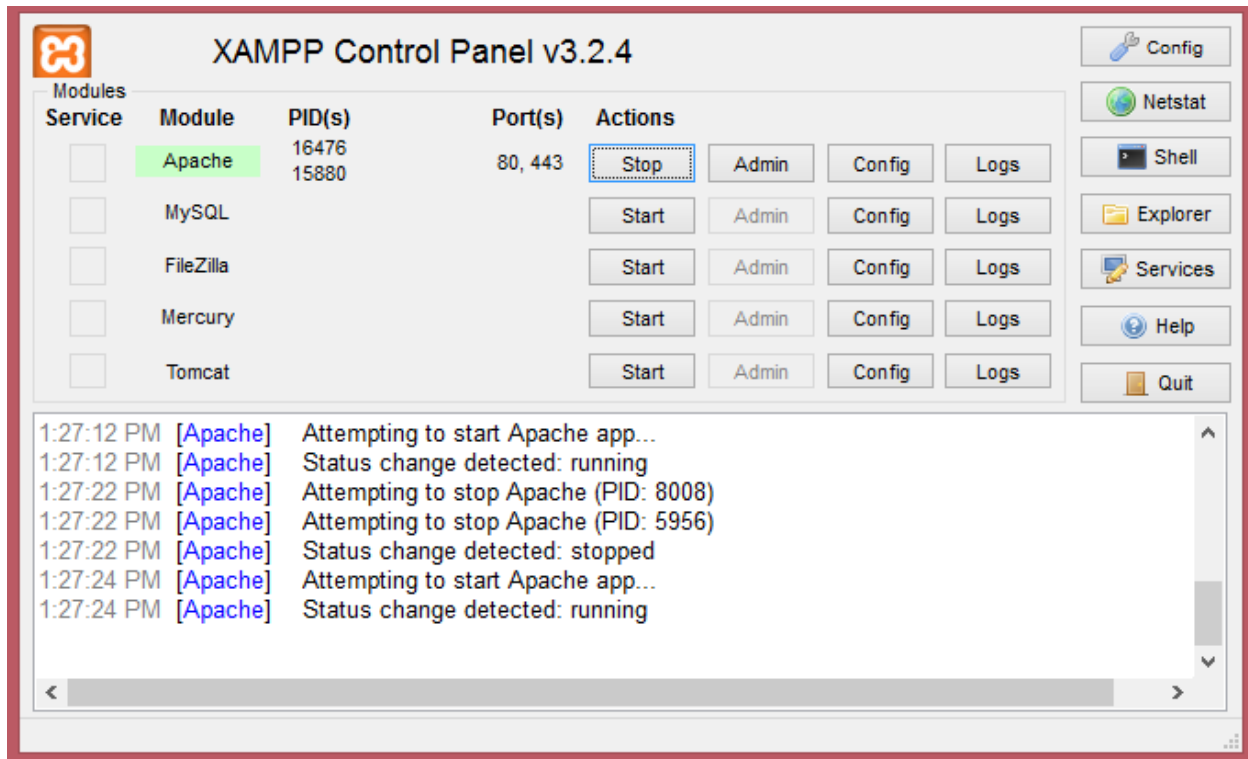
**AIM:** Setup Environment for All the Tools.

### Install Wamp/XAMPP Server:

What is XAMPP?

- XAMPP is the most popular PHP development environment
  - XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.
- 1) Open the XAMPP website. Go to <https://www.apachefriends.org/index.html> in your computer's web browser.
  - 2) Click XAMPP for Windows. It's a grey button near the bottom of the page.
  - 3) Double-click the downloaded file. This file should be named something like xampp-win32-7.2.4-0-VC15-installer, and you'll find it in the default downloads location (e.g., the "Downloads" folder or the desktop).
  - 4) Click yes when prompted. This will open the XAMPP setup window. You may have to click OK on a warning if you have User Account Control (UAC) activated on your computer.
  - 5) Click Next. It's at the bottom of the setup window.
  - 6) Select aspects of XAMPP to install. Review the list of XAMPP attributes on the left side of the window; if you see an attribute that you don't want to install as part of XAMPP, uncheck its box.
  - 7) Click Next. It's at the bottom of the window.
  - 8) Select an installation location. Click the folder-shaped icon to the right of the current installation destination, then click a folder on your computer.
  - 9) Click OK. Doing so confirms your selected folder as your XAMPP installation location.
  - 10) Click Next. You'll find it at the bottom of the page.
  - 11) Uncheck the "Learn more about Bitnami" box, then click next. The "Learn more about Bitnami" box is in the middle of the page.
  - 12) Begin installing XAMPP. Click next at the bottom of the window to do so. XAMPP will begin installing its files into the folder that you selected.
  - 13) Click Finish when prompted. It's at the bottom of the XAMPP window. Doing so will close the window and open the XAMPP Control Panel, which is where you'll access your servers.
  - 14) Select a language. Check the box next to the American flag for English
  - 15) Click Save. Doing so opens the main Control Panel page
  - 16) Start XAMPP from its installation point. If you need to open the XAMPP Control Panel in the future, you can do so by opening the folder in which you installed XAMPP, right-clicking the orange-and-white xampp-control icon, clicking Run as administrator, and clicking Yes when prompted. When you do this, you'll see red X marks to the left of each server type (e.g., "Apache"). Clicking one of these will prompt you to click Yes if you

want to install the server type's software on your computer. Counterintuitively, double-clicking the xampp\_start icon doesn't start XAMPP.



## Install JQuery Library:

What is jQuery?

- JQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: Write less, do more. JQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. JQuery is a JavaScript toolkit designed to simplify various tasks by writing less code. Here is the list of important core features supported by jQuery

- 1) DOM manipulation – the jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called Sizzle.
- 2) Event handling – The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- 3) AJAX Support – the jQuery helps you a lot to develop a responsive and feature rich site using AJAX technology.
- 4) Animations – the jQuery comes with plenty of built-in animation effects which you can use in your websites.
- 5) Lightweight – the jQuery is very lightweight library - about 19KB in size (Minified and gzipped).

- 6) Cross Browser Support - The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+
- 7) Latest Technology - the jQuery supports CSS3 selectors and basic XPath syntax.

How to use jQuery?

- There are two ways to use jQuery.
- Local Installation - You can download jQuery library on your local machine and include it in your HTML code.
- CDN Based Version - you can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

Now you can include jquery library in your HTML file as follows –

```
<html>
<head>
  <title>The jQuery Example</title>
  <script type = "text/javascript" src = "/jquery/jquery-2.1.3.min.js">
  </script>

  <script type = "text/javascript">
    $(document).ready(function() {
      document.write("Hello, World!");
    });
  </script>
</head>

<body>
  <h1>Hello</h1>
</body>
</html>
```

## Install D3.js and Canvas.js

What is D3.js?

- D3.js is a JavaScript library used to create interactive visualizations in the browser. The D3.js library allows us to manipulate elements of a webpage in the context of a data set. These elements can be HTML, SVG, or Canvas elements and can be introduced, removed, or edited according to the contents of the data set. It is a library for manipulating the DOM objects. D3.js can be a valuable aid in data exploration, it gives you control over your data's representation and lets you add interactivity.

Why Do We Need D3.js?

- D3.js is one of the premier framework when compare to other libraries. This is because it works on the web and its data visualizations are par excellence. Another reason it has worked so well is owing to its flexibility. Since it works seamlessly with the existing web technologies and can manipulate any part of the document object model, it is as flexible

as the Client Side Web Technology Stack (HTML, CSS, and SVG). It has a great community support and is easier to learn.

We need to include the D3.js library into your HTML webpage in order to use D3.js to create data visualization. We can do it in the following two ways –

1. Include the D3.js library from your project's folder.
2. Include D3.js library from CDN (Content Delivery Network).

### Download D3.js Library

D3.js is an open-source library and the source code of the library is freely available on the web at <https://d3js.org/> website. Visit the D3.js website and download the latest version of D3.js (d3.zip). As of now, the latest version is 4.6.0.

After the download is complete, unzip the file and look for d3.min.js. This is the minified version of the D3.js source code. Copy the d3.min.js file and paste it into your project's root folder or any other folder, where you want to keep all the library files. Include the d3.min.js file in your HTML page as shown below.

Example – Let us consider the following example.

```
<html>
<head>
  <script type = "text/javascript" src = "d3.v6.min.js"></script>
  <style>
    svg rect {
      fill: gray;
    }

    svg text {
      fill: yellow;
      font: 12px sans-serif;
      text-anchor: end;
    }
  </style>
</head>

<body>
  <script>
    var data = [10, 5, 12, 15];

    var width = 300
    scaleFactor = 20,
    barHeight = 30;
```

```

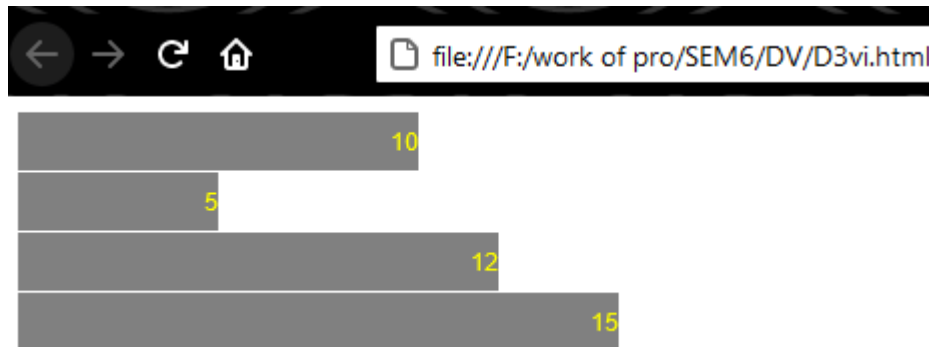
var graph = d3.select("body")
  .append("svg")
  .attr("width", width)
  .attr("height", barHeight * data.length);

var bar = graph.selectAll("g")
  .data(data)
  .enter()
  .append("g")
  .attr("transform", function(d, i) {
    return "translate(0," + i * barHeight + ")";
  });
bar.append("rect").attr("width", function(d) {
  return d * scaleFactor;
})

.attr("height", barHeight - 1);

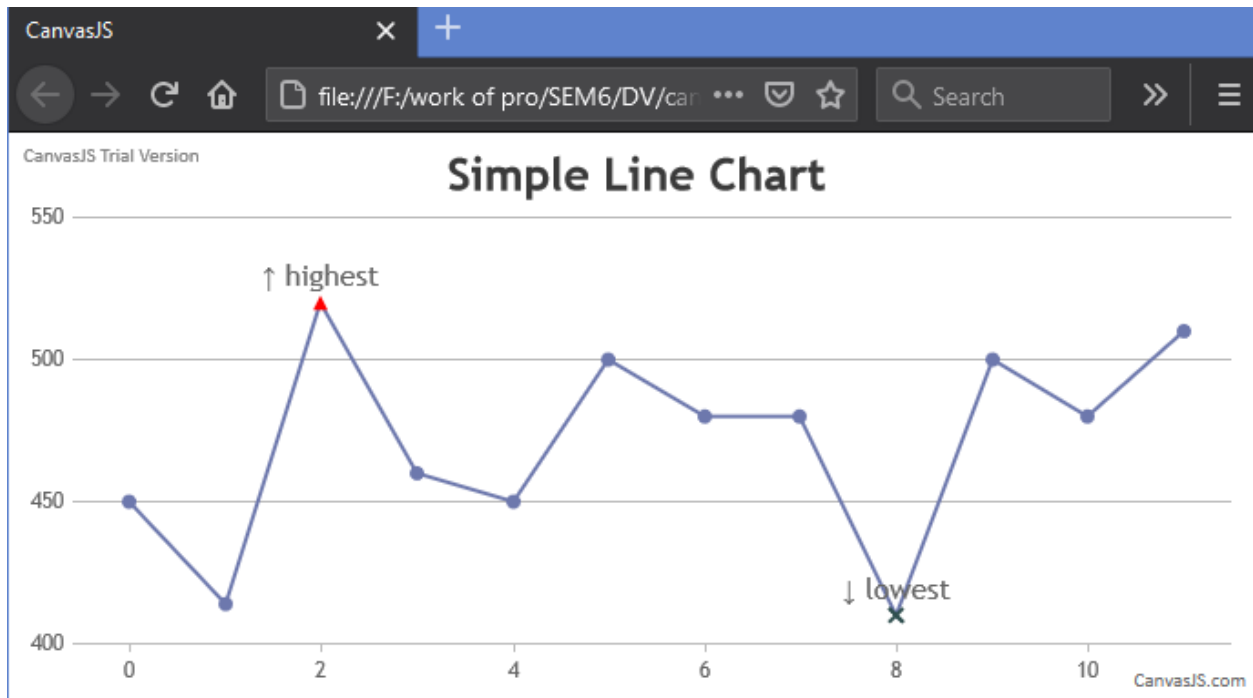
bar.append("text")
  .attr("x", function(d) { return (d*scaleFactor); })
  .attr("y", barHeight / 2)
  .attr("dy", ".35em")
  .text(function(d) { return d; });
</script>
</body>
</html>

```



We need to include the canvas.min.js library into your HTML webpage in order to use canvas.js to create data visualization. We can do it in the following two ways –

1. Include the canvas.min.js library from your project's folder.
2. Include canvas.js library from CDN (Content Delivery Network).



## Configure Google Chart API and Google Map API

- Google Charts is a pure JavaScript based charting library meant to enhance web applications by adding interactive charting capability. Google Charts provides wide variety of charts. For example, line charts, spline charts, area charts, bar charts, pie charts and so on.

There are two ways to use Google Charts.

1. Download – Download it locally from <https://developers.google.com/chart> and use it.
2. CDN access – you also have access to a CDN. The CDN will give you access around the world to regional data centers that in this case, Google Chart host <https://www.gstatic.com/charts>.

### Using Downloaded Google Chart

Include the googlecharts JavaScript file in the HTML page using following script –

```
<head>
  <script src = "/googlecharts/loader.js"></script>
</head>
```

### Using CDN

We are using the CDN versions of the Google Chart library throughout this tutorial.

Include the Google Chart JavaScript file in the HTML page using following script –

```
<head>
  <script src = "https://www.gstatic.com/charts/loader.js"></script>
```

</head>

**Step 1: Create HTML Page**

Create an HTML page with the Google Chart libraries.

Here container div is used to contain the chart drawn using Google Chart library. Here we are loading the latest version of corecharts API using google.charts.load method.

**Step 2: Create configurations**

Google Chart library uses very simple configurations using json syntax.

Here data represents the json data and options represents the configuration which Google Chart library uses to draw a chart withing container div using draw() method. Now we'll configure the various parameter to create the required json string.

**DataTable**

Configure the data to be displayed on the chart. DataTable is a special table structured collection which contains the data of the chart. Columns of data table represents the legends and rows represents the corresponding data. addColumn() method is used to add a column where first parameter represents the data type and second parameter represents the legend. addRows() method is used to add rows accordingly.

**Step 3: Draw the chart**

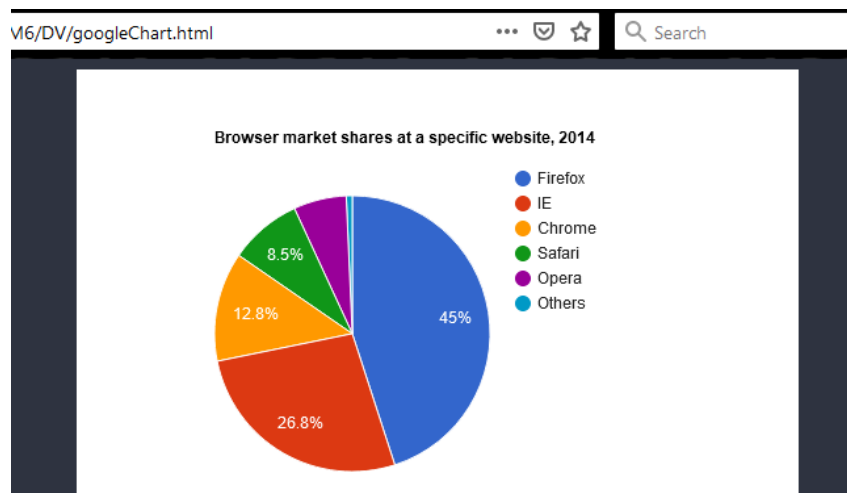
google.charts.setOnLoadCallback(drawChart);

```
<html>
  <head>
    <title>Google Charts Tutorial</title>
    <script type = "text/javascript" src = "loader.js">
    </script>
    <script type = "text/javascript">
      google.charts.load('current', {packages: ['corechart']});
    </script>
  </head>

  <body style="background: #2e3340;">
    <div id = "container" style = "width: 550px; height: 400px; margin: 0 auto">
    </div>
    <script language = "JavaScript">
      function drawChart() {
        // Define the chart to be drawn.
        var data = new google.visualization.DataTable();
        data.addColumn('string', 'Browser');
        data.addColumn('number', 'Percentage');
        data.addRows([
          ['Firefox', 45.0],
          ['IE', 26.8],
```



```
    ['Chrome', 12.8],  
    ['Safari', 8.5],  
    ['Opera', 6.2],  
    ['Others', 0.7]  
  );  
  
  // Set chart options  
  var options = {'title':'Browser market shares at a specific website, 2014', 'width':550,  
    'height':400};  
  
  // Instantiate and draw the chart.  
  var chart = new google.visualization.PieChart(document.getElementById ('container'));  
  chart.draw(data, options);  
}  
google.charts.setOnLoadCallback(drawChart);  
</script>  
</body>  
</html>
```



### What are Google Maps?

- Google Maps is a free web mapping service by Google that provides various types of geographical information. Using Google Maps, one can.
  - Search for places or get directions from one place to another.
  - View and navigate through horizontal and vertical panoramic street level images of various cities around the world.
  - Get specific information like traffic at a particular point.

Google Maps provides an API using which you can customize the maps and the information displayed on them. This chapter explains how to load a simple Google Map on your web page using HTML and JavaScript.

**Just include iframe:**

```
<iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d58882.21164645128!2d71.61154774849125!3d22.723103086866118!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x395940dfdc9f82af%3A0xa4b2578b8bf73d43!2sSurendranagar%2C%20Gujarat!5e0!3m2!1sen!2sin!4v1614767325087!5m2!1sen!2sin" width="600" height="450" style="border:0;"
allowfullscreen="" loading="lazy"></iframe>
```

**Or**

```
<!DOCTYPE html>
<html>
<head>
<title>google map API</title>

</script>
<style>
#map{
height: 300px; /* The height is 400 pixels */
width: 100%; /* The width is the width of the web page */
}
</style>
</head>

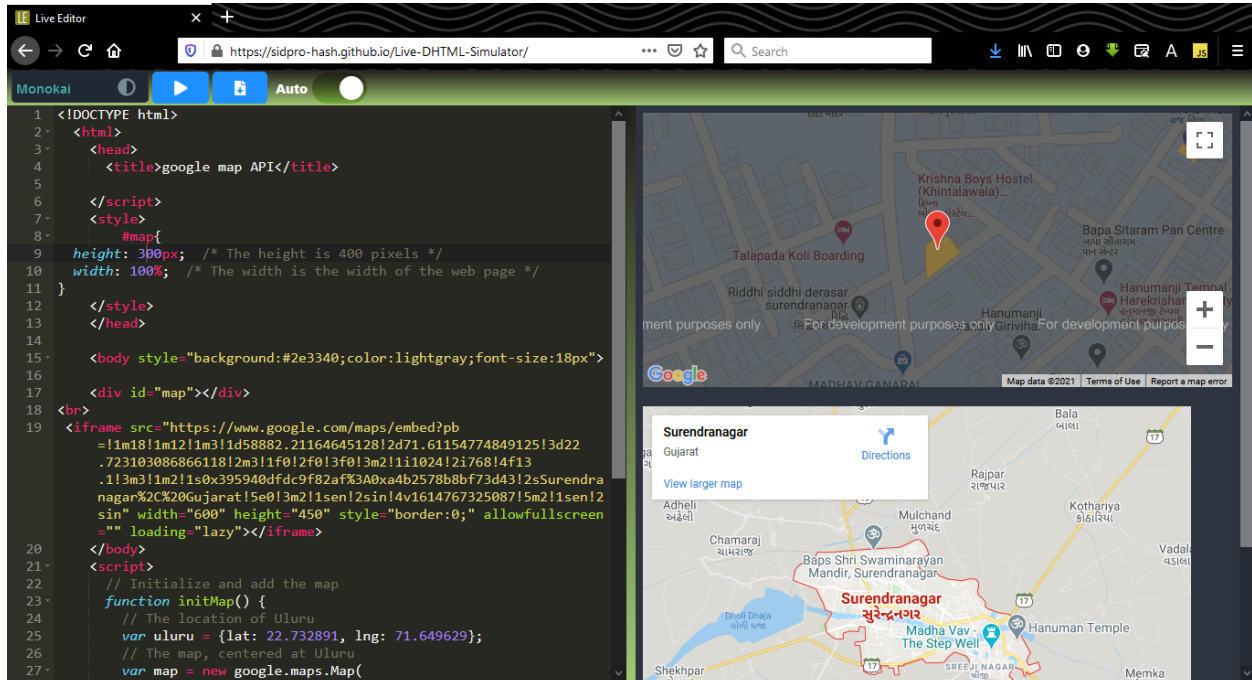
<body style="background:#2e3340;color:lightgray;font-size:18px">

<div id="map"></div>
</body>
<script>
// Initialize and add the map
function initMap() {
// The location of Uluru
var uluru = {lat: 22.732891, lng: 71.649629};
// The map, centered at Uluru
var map = new google.maps.Map(
document.getElementById('map'), {zoom: 18, center: uluru,rotateControl:
true,streetViewControl: true});
// The marker, positioned at Uluru
var marker = new google.maps.Marker({position: uluru, map: map});
function myFunction() {
var x = document.getElementById("navtop");
x.classList.toggle("responsive");
```

```

    }
    initMap();
  </script>
  <script async defer
    src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY
    &callback=initMap"></script>
  </html>

```



## Practical – 2

**AIM:** Develop the following Program Using HTML5 CANVAS

a) Develop the Different basic Graphical Shapes using HTML5 CANVAS

```
<!DOCTYPE html>
<html>
  <head>
    <title>Colleggeek</title>
  </head>

  <body style="background:#2e3340;color:lightgray;font-size:18px">

    <canvas id="canvas" style="background-color:"></canvas>
  </body>
  <script>
/**
 * Draws a rounded rectangle using the current state of the canvas.
 * If you omit the last three params, it will draw a rectangle
 * outline with a 5 pixel border radius
 * @param {CanvasRenderingContext2D} ctx
 * @param {Number} x The top left x coordinate
 * @param {Number} y The top left y coordinate
 * @param {Number} width The width of the rectangle
 * @param {Number} height The height of the rectangle
 * @param {Number} radius The corner radius. Defaults to 5;
 * @param {Boolean} fill Whether to fill the rectangle. Defaults to false.
 * @param {Boolean} stroke Whether to stroke the rectangle. Defaults to true.
 */
function roundRect(ctx, x, y, width, height, radius, fill, stroke) {
  if (typeof stroke == "undefined" ) {
    stroke = true;
  }
  if (typeof radius === "undefined") {
    radius = 5;
  }
  ctx.beginPath();
  ctx.moveTo(x + radius, y);
  ctx.lineTo(x + width - radius, y);
  ctx.quadraticCurveTo(x + width, y, x + width, y + radius);
  ctx.lineTo(x + width, y + height - radius);
  ctx.quadraticCurveTo(x + width, y + height, x + width - radius, y + height);
  ctx.lineTo(x + radius, y + height);
  ctx.quadraticCurveTo(x, y + height, x, y + height - radius);
```

```
ctx.lineTo(x, y + radius);
ctx.quadraticCurveTo(x, y, x + radius, y);
ctx.closePath();
if (stroke) {
  ctx.stroke();
}
if (fill) {
  ctx.fill();
}
}

var canvas = document.getElementById("canvas");
canvas.width = window.innerWidth;
canvas.height=window.innerHeight;
var ctx = canvas.getContext("2d");
var W2 = canvas.width/2;
var H4 = canvas.height/4 - 50;
var H3 = canvas.height/3 - 50;
//Head part
ctx.strokeStyle = "green";
ctx.lineWidth="2";
ctx.beginPath();
ctx.moveTo(W2,10);
ctx.lineTo(W2-30,70);
ctx.stroke();
ctx.moveTo(W2,10);
ctx.lineTo(W2+30,70);
ctx.stroke();
ctx.moveTo(W2-30,70);
ctx.quadraticCurveTo(W2,50,W2+30,70);
ctx.stroke();

ctx.strokeStyle="#FF0000";
ctx.lineWidth="2";
ctx.beginPath();
ctx.arc(W2,H4,40,0,2*Math.PI);
ctx.stroke();
ctx.beginPath();
ctx.arc(W2-15,H4-10,10,0,2*Math.PI);
ctx.stroke();
ctx.beginPath();
ctx.arc(W2+15,H4-10,10,0,2*Math.PI);
ctx.stroke();
ctx.beginPath();
ctx.moveTo(W2-15,H4+20);
ctx.quadraticCurveTo(W2, H3-20, W2+15, H4+20);
ctx.stroke();
```

```
//body
//context,x,y,width,height,radius,fill
roundRect(ctx,W2-75,H3,150,150,70);

//left hand
ctx.strokeStyle="#FF0000";
ctx.beginPath();
ctx.moveTo(W2-50,H3+10);
ctx.quadraticCurveTo(W2-140,H3+40,W2-220,H3+10);
ctx.stroke();

ctx.beginPath();
ctx.moveTo(W2-70,H3+40);
ctx.quadraticCurveTo(W2-140,H3+60,W2-220,H3+40);
ctx.stroke();

ctx.fillStyle="blue";
ctx.fillRect(W2-320,H3-20,100,100);
ctx.fillStyle="white";
ctx.fillRect(W2-310,H3-10,80,80);
ctx.stroke();
ctx.beginPath();
ctx.font = "16px Verdana";
    // Create gradient
    var gradient = ctx.createLinearGradient(0, 0, W2, 0);
    gradient.addColorStop("0", "magenta");
    gradient.addColorStop("0.5", "blue");
    gradient.addColorStop("1.0", "red");

    // Fill with gradient
    ctx.strokeStyle = gradient;
    ctx.strokeText("Exploring", W2-307, H3+10);
    ctx.strokeText("HTML", W2-295, H3+30);
    ctx.strokeText("Canvas", W2-300, H3+50);
    ctx.stroke();

//right hand
ctx.strokeStyle="#FF0000";
ctx.beginPath();
ctx.moveTo(W2+50,H3+10);
ctx.quadraticCurveTo(W2+140,H3+40,W2+220,H3+10);
ctx.stroke();

ctx.beginPath();
ctx.moveTo(W2+70,H3+40);
ctx.quadraticCurveTo(W2+140,H3+60,W2+220,H3+40);
```

```
ctx.stroke();

ctx.fillStyle="blue";
ctx.fillRect(W2+220,H3-20,100,100);
ctx.fillStyle="white";
ctx.fillRect(W2+230,H3-10,80,80);
ctx.stroke();

ctx.font = "16px Verdana";
ctx.strokeText("Exploring", W2+233, H3+10);
ctx.strokeText("HTML", W2+250, H3+30);
ctx.strokeText("Canvas", W2+240, H3+50);
ctx.stroke();
//left leg
ctx.strokeStyle="#FF0000";
ctx.beginPath();
ctx.moveTo(W2-60,H3+130);
ctx.quadraticCurveTo(W2-80,H3+180,W2-50,H3+220);
ctx.stroke();

ctx.beginPath();
ctx.moveTo(W2-50,H3+220);
ctx.quadraticCurveTo(W2-20,H3+260,W2-30,H3+330);
ctx.stroke();

//right leg
ctx.strokeStyle="#FF0000";
ctx.beginPath();
ctx.moveTo(W2+60,H3+130);
ctx.quadraticCurveTo(W2+30,H3+180,W2+50,H3+220);
ctx.stroke();

ctx.beginPath();
ctx.moveTo(W2+50,H3+220);
ctx.quadraticCurveTo(W2+70,H3+260,W2-20,H3+340);
ctx.stroke();

</script>
</html>
```



Fig – 2a

## b) Develop the Different Advanced Graphical Shapes using HTML5 CANVAS

```
<!DOCTYPE html>
<html>
  <head>
    <!-- live link: sidpro-hash.github.io/html-canvas -->
    <title>HTML5 Canvas - Sine wave</title>
    <style>
      body{
        background:#151515;
      }
      div{
        z-index:2;
        position:absolute;
      }
      input[type=range]{
        width:200px;
        height:15px;
        -webkit-appearance:none;
        background:#111;
        outline:none;
        margin:5px;
        position: relative;
        border-radius:15px;
        overflow:hidden;
```



```

        box-shadow:inset 0 0 5px rgba(0,0,0,1);
    }
    input[type=range]::-moz-range-thumb,input[type=range]::-webkit-slider-thumb{
        -webkit-appearance:none;
        width:15px;
        height:15px;
        border-radius:50%;
        background:#00fd0a;
        cursor:pointer;
        border:4px solid #333;
        box-shadow:-407px 0 0 400px #00fd0a;
    }
</style>
</head>
<body>
    <div>
        <input title="height" type="range" id="height" oninput="ranges()"
onchange="ranges()">
        <input title="wavelength" type="range" id="wavelength" oninput="ranges()"
onchange="ranges()">
        <input title="amplitude" type="range" id="amplitude" oninput="ranges()"
onchange="ranges()">
        <input title="frequency" type="range" id="freq" oninput="ranges()"
onchange="ranges()"><br>
        <input title="H" type="range" id="h" oninput="color()" onchange="color()">
        <input title="S" type="range" id="s" oninput="color()" onchange="color()">
        <input title="L" type="range" id="l" oninput="color()" onchange="color()">
    </div>
    <canvas style="background-color:white"></canvas>
</body>
<script>
    var inheight = document.getElementById('height');
    var inwavelen = document.getElementById('wavelength');
    var inamplitude = document.getElementById('amplitude');
    var infreq = document.getElementById('freq');
    var inh = document.getElementById('h');
    var ins = document.getElementById('s');
    var inl = document.getElementById('l');
    var canvas = document.querySelector('canvas');
    var c = canvas.getContext("2d");
    var y,wavelength,ampliuide,freq;
    var inc=0.01;//change to 0.01 to make //animation
    var H,W,h,s,l;
    // inc=parseFloat(freq);
    function animate(){
        requestAnimationFrame(animate);

```

```
c.fillStyle="rgba(0,0,0,0.01)";
c.fillRect(0,0,W,H); //animation
c.beginPath();
c.moveTo(0,y);
for(let i=0;i<canvas.width;++i){
    c.lineTo(i,y + Math.sin(i*wavelength+inc)*ampliude);
}
c.stroke();
c.strokeStyle = `hsl(${h},${s}%,${l}%)`;
inc+= parseFloat(freq); //animation
}
animate();
function ranges(){
    y=inheight.value/2;
    wavelength=inwavelen.value;
    ampliude=inamplitude.value;
    freq=infreq.value;
}
function color(){
    h = inh.value;
    s = ins.value;
    l = inl.value;
    c.strokeStyle = `hsl(${r},${g}%,${b}%)`;
}

canvas.width = window.innerWidth;
canvas.height = window.innerHeight;
H = canvas.height;
W = canvas.width;
//set min/max height of wave
inheight.min = 0;inheight.max = H;inheight.value=H;
//set min/max/step wavelength of wave
inwavelen.step=0.001;
inwavelen.min = -0.01;
inwavelen.max = 0.01;
inwavelen.value=0.01;
//set amplitude of wave
inamplitude.min=-300;
inamplitude.max=300;
inamplitude.value = 100;
//set frequency
infreq.step=0.01;
infreq.min=-0.1;
infreq.max=0.1;
infreq.value=0.01;
//set color values
inh.max=255;
```

```

    inh.min=0;
    inh.value=0;
    ins.max=100;
    ins.min=0;
    ins.value=50;
    inl.max=100;
    inl.min=0;
    inl.value=50;
    y=inheight.value/2;
    wavelength=inwavelen.value;
    amplitude=inamplitude.value;
    freq = infreq.value;
    inc = parseFloat(freq);
    h = inh.value;
    s = ins.value;
    l = inl.value;
    c.strokeStyle = `hsl(${s},${s}%,${l}%)`;
    window.onresize = () => {
        canvas.width = window.innerWidth;
        canvas.height = window.innerHeight;
        H = canvas.height;
        W = canvas.width;
    }
</script>
</html>

```

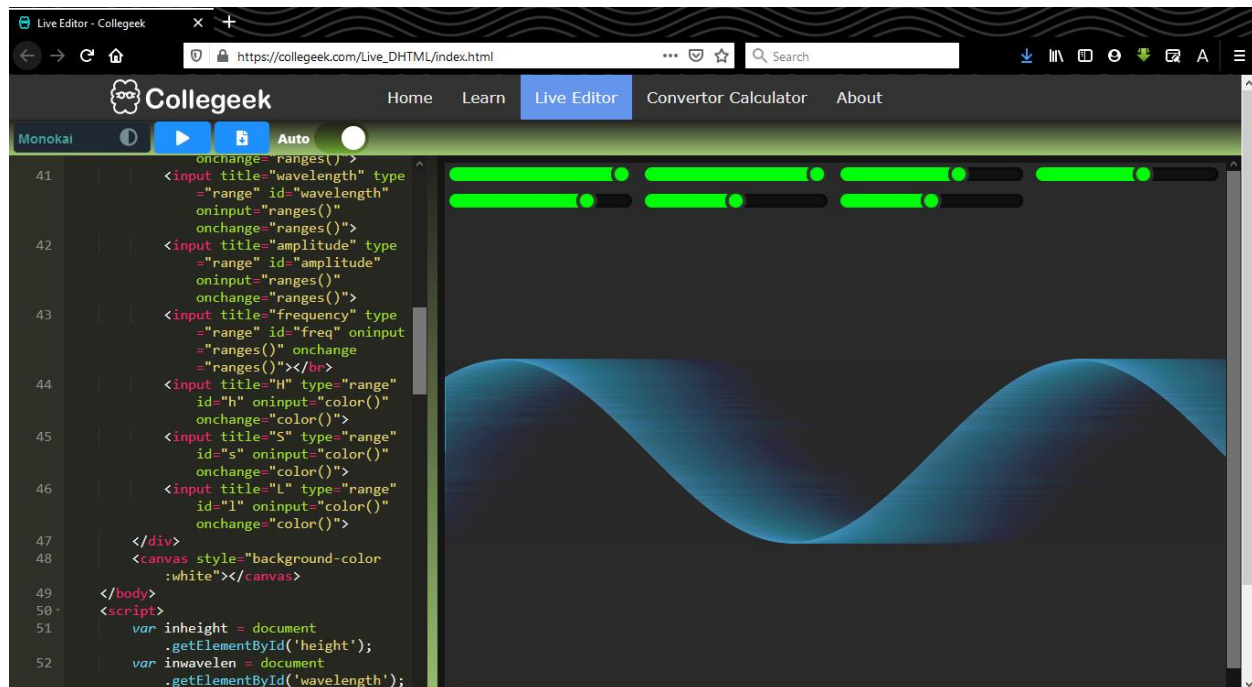


Fig – 2b

## c) Develop the Different basic Graphical Shapes using HTML5 SVG

```

<!DOCTYPE html>
<html>
  <head>
    <title>Collegeek</title>

    <style>
    </style>
  </head>

  <body style="background:#2e3340;color:lightgray;font-size:18px">
<svg height="150" width="500">
  <ellipse cx="240" cy="100" rx="220" ry="30" style="fill:purple" />
  <ellipse cx="220" cy="70" rx="190" ry="20" style="fill:lime" />
  <ellipse cx="210" cy="45" rx="170" ry="15" style="fill:yellow" />
</svg>
<svg height="210" width="500">
  <polygon points="100,10 40,198 190,78 10,78 160,198"
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;" />
</svg>
<div id="sine_wave">
  <svg width="1000" height="1000">
    <line x1="100" y1="0" x2="100" y2="200"
    style="stroke:red;stroke-width:1"/>
    <line x1="0" y1="100" x2="1000" y2="100"
    style="stroke:red;stroke-width:1"/>
  </svg>
</div>

</body>
<script>
var svg = document.getElementById('sine_wave').children[0];
var origin = { //origin of axes
  x: 100,
  y: 100
};
var amplitude = 40; // wave amplitude
var rarity = 3; // point spacing
var freq = 0.1; // angular frequency
var phase = 1; // phase angle

for (var i = -100; i < 1000; i++) {
  var line = document.createElementNS("http://www.w3.org/2000/svg", "line");

```

```

line.setAttribute('x1', (i - 1) * rarity + origin.x);
line.setAttribute('y1', Math.sin(freq*(i - 1 + phase)) * amplitude + origin.y);

line.setAttribute('x2', i * rarity + origin.x);
line.setAttribute('y2', Math.sin(freq*(i + phase)) * amplitude + origin.y);

line.setAttribute('style', "stroke:green;stroke-width:2");

svg.appendChild(line);
}
</script>
</html>

```

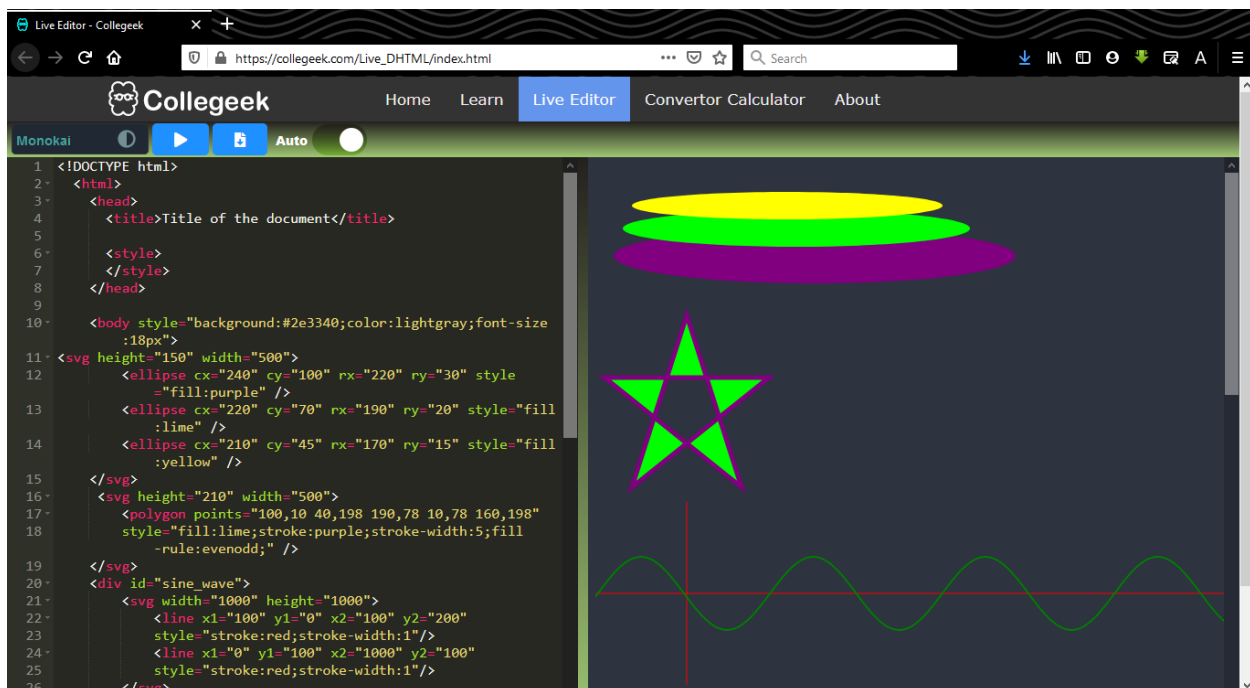


Fig – 2c

d) Develop the Different Advanced Graphical Shapes using HTML5 SVG

```

<!DOCTYPE html>
<html>
<head>
<title>Collegeeek</title>
</head>

<body style="background:#2e3340;color:lightgray;font-size:18px">

<svg height="600" width="700">

```

```

<defs>
  <filter id="f2" x="0" y="0" width="200%" height="200%">
    <feOffset result="offOut" in="SourceGraphic" dx="0" dy="20" />
    <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
    <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
  </filter>
</defs>

<path d="M 220 210 q 150 -390 300 0" filter="url(#f2)"
style="stroke:rgb(133, 67, 13);stroke-width:5;fill:none" />
<path d="M 470 100 q 50 -70 100 0"
style="stroke:yellow;stroke-width:5;fill:none" />
<line x1="490" y1="70" x2="480" y2="60"
style="stroke:yellow;stroke-width:2"/>
<line x1="477" y1="80" x2="465" y2="70"
style="stroke:yellow;stroke-width:2"/>
<line x1="520" y1="45" x2="520" y2="60"
style="stroke:yellow;stroke-width:2"/>
<line x1="550" y1="70" x2="560" y2="60"
style="stroke:yellow;stroke-width:2"/>
<line x1="565" y1="80" x2="577" y2="70"
style="stroke:yellow;stroke-width:2"/>
<path d="M 520 210 q 150 -400 300 0" filter="url(#f2)"
style="stroke:rgb(133, 67, 13);stroke-width:5;fill:none" />

<line x1="175" y1="70" x2="175" y2="150"
style="stroke:black;stroke-width:2"/>

<polygon points="175,70 250,100 175,130"
style="fill:lime;stroke:purple;stroke-width:1"/>

<line x1="100" y1="300" x2="175" y2="150"
style="stroke:rgb(255,0,0);stroke-width:2" />
<line x1="250" y1="300" x2="175" y2="150"
style="stroke:rgb(255,0,0);stroke-width:2" />

<rect width="150" height="150" x="100" y="300"
style="fill:black;fill-opacity:0.5;stroke:red"/>

<polyline points="250,450 270,450 270,465 290,465 290,480 310,480"
style="fill:#2e3340;stroke:blue;stroke-width:3"/>

<ellipse cx="250" cy="530" rx="250" ry="50"
style="fill:rgb(133, 67, 13)"/>
<ellipse cx="250" cy="530" rx="230" ry="40"
style="fill:cornflowerblue"/>

```

```

<rect width="700" height="600" fill="none" stroke="black" stroke-width="4"/>

<text x="400" y="25" fill="rgb(214,61,213)" transform="rotate(30 20,40)">મંદિર
તાડું વિશ્વ રૂપાળું, સુંદર સર્જનહારા રે;</text>

<text x="400" y="45" fill="rgb(34,61,213)" transform="rotate(30 20,40)">
પળ પળ તારાં દર્શન થાયે, દેખે દેખનહારા રે</text>

<text x="400" y="65" fill="rgb(214,61,213)" transform="rotate(30 20,40)">નહીં
પૂજારી નહીં કોઈ દેવા નહીં મંદિરને તાળાં રે;</text>

<text x="400" y="85" fill="rgb(34,61,213)" transform="rotate(30 20,40)">નીલ
ગગનમાં મહિમા ગાતા, ચાંદો સૂરજ તારા રે</text>

<text x="400" y="105" fill="rgb(214,61,213)" transform="rotate(30 20,40)">વર્ણન
કરતાં શોભા તારી, થાક્યા કવિગણ ધીરા રે;</text>

<text x="400" y="125" fill="rgb(34,61,213)" transform="rotate(30 20,40)">મંદિરમાં
તું કયાં છૂપાયો ? શોધે બાળ અધીરાં રે</text>

</svg>
</body>
</html>

```

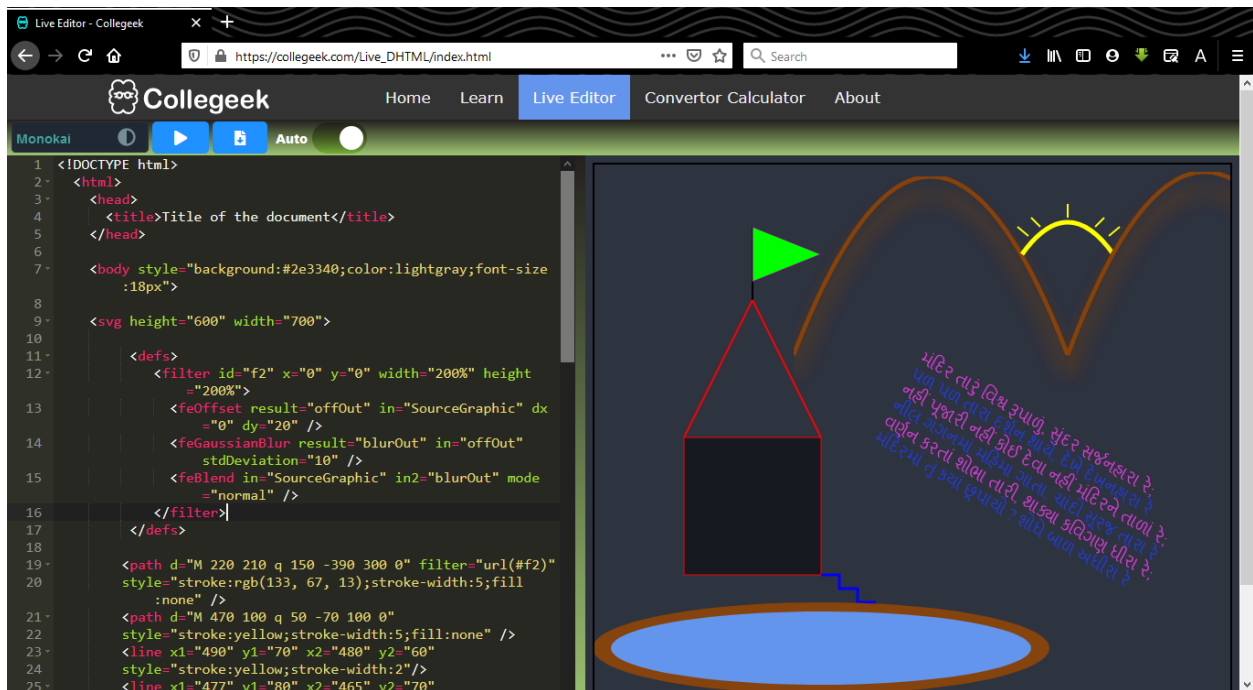


Fig – 2d

## Practical – 3

**AIM:** Develop Following Program Using HTML5 and JavaScript

a) Read the data .txt file and draw Data Table

```
<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read Text File to Data Table</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="https://code.jquery.com/jquery-3.5.1.min.js"
integrity="sha256-9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0="
crossorigin="anonymous"></script>
    <style>
      #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;
        width: 100%;
      }

      #mytable td, #mytable th {
        border: 1px solid #ddd;
        padding: 8px;
      }

      #mytable tr:nth-child(even){background-color: #f2f2f2;}


      #mytable tr:hover {background-color: #ddd;}

      #mytable th {
        padding-top: 12px;
        padding-bottom: 12px;
        text-align: left;
        background-color: cornflowerblue;
        color: white;
      }
    </style>
```



```
</head>
<body>
  <h1>Get data from text file</h1>
  <table id="mytable">
  </table>
</body>
<script>
var content = "";
$( document ).ready(function() {
  //just change the name of .txt file to load file
  $.get('Colors.txt', function(theData) {

    theData = theData.replace(/\r/g, "");
    theData = theData.replace(/\t/g, ' ');
    theData = theData.split('\n');
    totalRows = theData.length;
    theHead = theData[0].split(' ');
    content += "<tr>";
    theHead.forEach(TH);
    content += "</tr>";
    for(let i=1;i<totalRows;++i){
      theTD = theData[i].split(' ');
      content += "<tr>";
      theTD.forEach(TD);
      content += "</tr>";
    }
    $('#mytable').html(content);
  });
});
function TH(value){
  content += "<th>" + value + "</th>";
}
function TD(value){
  content += "<td>" + value + "</td>";
}
</script>
</html>
```



**Get data from text file**

Color	Value	Hex	RGB
Red	1	#f00000	rgb(255,0,0)
Orange	2	#ffa500	rgb(255,165,0)
Yellow	3	#ffff00	rgb(255,255,0)
Green	4	#008000	rgb(0,128,0)
Blue	5	#0000ff	rgb(0,0,255)
Purple	6	#800080	rgb(128,0,128)
Black	7	#000000	rgb(0,0,0)
White	8	#ffffff	rgb(255,255,255)

Fig – 3a

## b) Read the data .csv file and draw Data Table

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read CSV File to Data Table</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="jquery-3.5.1.min.js"></script>
    <script src="jquery.csv.js"></script>
    <style>
      #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;
        width: 100%;
      }

      #mytable td, #mytable th {
        border: 1px solid #ddd;
        padding: 8px;
      }

      #mytable tr:nth-child(even){ background-color: #f2f2f2;}

```

```

        #mytable tr:hover {background-color: #ddd;}

        #mytable th {
            padding-top: 12px;
            padding-bottom: 12px;
            text-align: left;
            background-color: cornflowerblue;
            color: white;
        }
    </style>
</head>
<body>
    <h1>Get data from CSV file</h1>
    <table id="mytable">
    </table>
</body>
<script>
var content = "";
$( document ).ready(function() {
    //just change the name of .csv file to load file
    $.get('stores.csv', function(theData) {
        console.log(theData);
        theData = theData.replace(/\n/g,"");
        console.log(theData);

        theData = theData.split(/\r?\n|\r/);
        console.log(theData);
        totalRows = theData.length;

        theHead = theData[0].split(',');
        content += "<tr>";
        theHead.forEach(TH);
        content += "</tr>";

        for(let i=1;i<totalRows;++i){
            theTD = theData[i].split(',');
            content += "<tr>";
            theTD.forEach(TD);
            content += "</tr>";
        }
        $('#mytable').html(content);

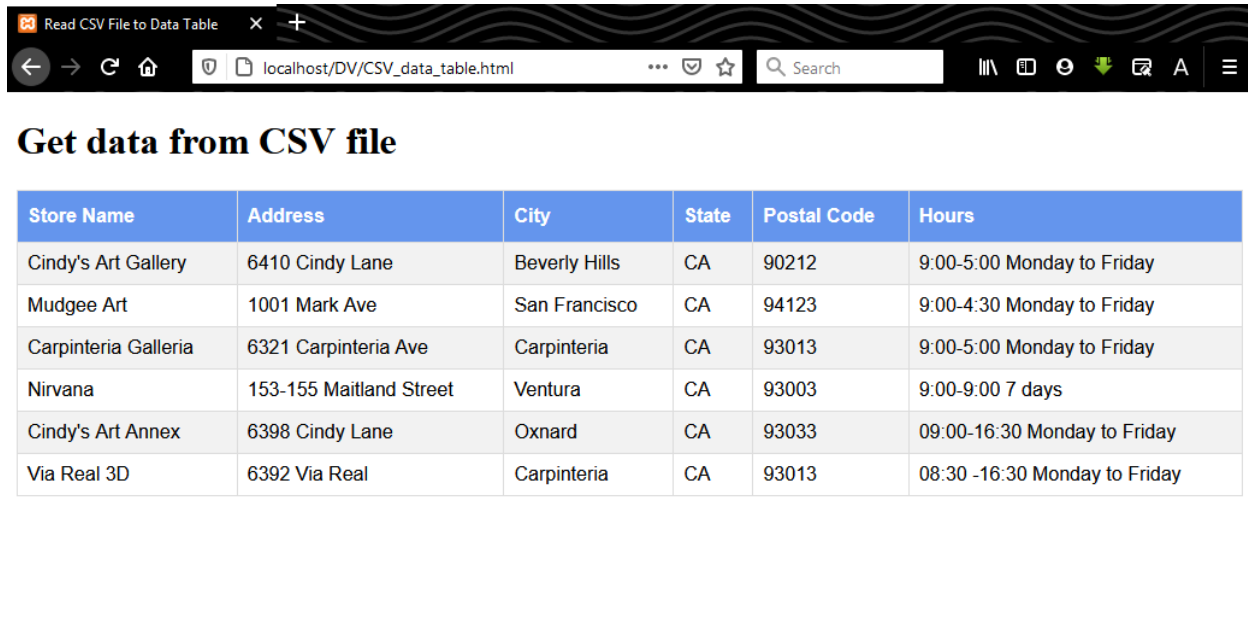
    });
});

```

```

function TH(value){
    content += "<th>" + value + "</th>";
}
function TD(value){
    content += "<td>" + value + "</td>";
}
</script>
</html>

```



**Get data from CSV file**

Store Name	Address	City	State	Postal Code	Hours
Cindy's Art Gallery	6410 Cindy Lane	Beverly Hills	CA	90212	9:00-5:00 Monday to Friday
Mudgee Art	1001 Mark Ave	San Francisco	CA	94123	9:00-4:30 Monday to Friday
Carpinteria Galleria	6321 Carpinteria Ave	Carpinteria	CA	93013	9:00-5:00 Monday to Friday
Nirvana	153-155 Maitland Street	Ventura	CA	93003	9:00-9:00 7 days
Cindy's Art Annex	6398 Cindy Lane	Oxnard	CA	93033	09:00-16:30 Monday to Friday
Via Real 3D	6392 Via Real	Carpinteria	CA	93013	08:30 -16:30 Monday to Friday

Fig3 – b

## c) Read the data XML file and draw Data Table

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read XML File to Data Table</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="jquery-3.5.1.min.js"></script>

```

```
<script src="jquery.csv.js"></script>
<style>
    #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;
        width: 100%;
    }

    #mytable td, #mytable th {
        border: 1px solid #ddd;
        padding: 8px;
    }

    #mytable tr:nth-child(even){ background-color: #f2f2f2;}

    #mytable tr:hover { background-color: #ddd;}

    #mytable th {
        padding-top: 12px;
        padding-bottom: 12px;
        text-align: left;
        background-color: cornflowerblue;
        color: white;
    }
</style>
</head>
<body>
    <h1>Get data from XML file</h1>
    <table id="mytable">
        <tr>
            <th>Territory</th>
            <th>Employees</th>
            <th>Sales</th>
            <th>Year</th>
        </tr>
    </table>
<script>
    $( document ).ready(function() {
        $.ajax({
            type: "GET",
            url: "regional_sales.xml",
            dataType: "xml",
            success: xmlParser
        });
    });
</script>
```

```

function xmlParser(xml) {
    $(xml).find('region').each(function () {
        var theTerritory = $(this).find('territory').text();
        var numEmployees = $(this).find('employees').text();
        var theAmount = $(this).find('amount').text();
        $('#mytable').append('<tr><td>' + theTerritory + '</td><td>' +
numEmployees + '</td><td>' + theAmount + '</td><td>' + '2013' + '</td></tr>');
    });
}
</script>
</html>

```



Fig3 – c

#### d) Read JSON Data and draw Data Table

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
    <head>
        <title>Read JSON File to Data Table</title>
        <meta charset="utf-8">
        <meta name="author" content="SidPro"/>
        <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
        <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>

```

```

        <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
        <script src="jquery-3.5.1.min.js"></script>
        <script src="jquery.csv.js"></script>
        <style>
            #mytable {
                font-family: Arial, Helvetica, sans-serif;
                border-collapse: collapse;
                width: 100%;
            }

            #mytable td, #mytable th {
                border: 1px solid #ddd;
                padding: 8px;
            }

            #mytable tr:nth-child(even){ background-color: #f2f2f2;}

            #mytable tr:hover { background-color: #ddd;}

            #mytable th {
                padding-top: 12px;
                padding-bottom: 12px;
                text-align: left;
                background-color: cornflowerblue;
                color: white;
            }
        </style>
    </head>
    <body>
        <h1>Get data from JSON file</h1>
        <table id="mytable">
        </table>
    </body>
    <script>
    var content = "";
    $(document).ready(function() {
        $.ajax({
            type: "Get",
            url: "regional_sales.json",
            dataType: "json",
            success: function(data) {
                var theData = data.sales.region;
                var theHead = Object.keys(theData[0]);

                content += "<tr>";

```

```

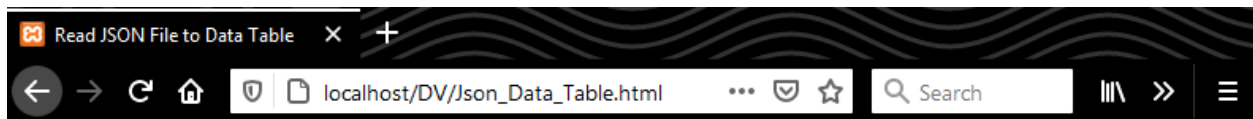
        theHead.forEach(TH);
        content += "</tr>";

        theData.forEach(function(v){
            v = Object.values(v);
            content += "<tr>";
            v.forEach(TD);
            content += "</tr>";
        });

        $('#mytable').html(content);
    },
    error: function(){
        alert("json not found");
    }
});

function TH(value){
    content += "<th>" + value.toUpperCase() + "</th>";
}
function TD(value){
    content += "<td>" + value + "</td>";
}
</script>
</html>

```



## Get data from JSON file

YEAR	TERRITORY	EMPLOYEES	AMOUNT
2013	Northeast	150	115,000
2013	Southeast	125	95,000
2013	Midwest	225	195,000
2013	West	325	265,000

Fig3 – d



## Practical – 4

**AIM:** Develop Following Program Using HTML5 and JavaScript

a) Develop the simple bar chart using HTML5 CANVAS

```
<!DOCTYPE html>
<html>
  <head>
    <title>Collegegeek</title>
  </head>
  <body>
    <canvas id="myCanvas" style="background: #2e3340;"></canvas>
    <legend for="myCanvas"></legend>
    <script type="text/javascript">
      var myCanvas = document.getElementById("myCanvas");
      myCanvas.width = window.innerWidth-30;
      myCanvas.height = window.innerHeight-100;

      var ctx = myCanvas.getContext("2d");

      function drawLine(ctx, startX, startY, endX, endY,color){
        ctx.save();
        ctx.strokeStyle = color;
        ctx.beginPath();
        ctx.moveTo(startX,startY);
        ctx.lineTo(endX,endY);
        ctx.stroke();
        ctx.restore();
      }

      function drawBar(ctx, upperLeftCornerX, upperLeftCornerY, width,
height,color){
        ctx.save();
        ctx.fillStyle=color;
        ctx.fillRect(upperLeftCornerX,upperLeftCornerY,width,height);
        ctx.restore();
      }

      var myVinyls = {
        "Face Cream": 2500,
        "Face Wash": 1500,
        "ToothPaste": 5200,
        "Bathing Soap": 9200,
        "Shampoo":1200,
```

```
        "moisturizer":10800
    };

    var Barchart = function(options){
        this.options = options;
        this.canvas = options.canvas;
        this.ctx = this.canvas.getContext("2d");
        this.colors = options.colors;

        this.draw = function(){
            var maxValue = 0;
            for (var categ in this.options.data){
                maxValue =
Math.max(maxValue,this.options.data[categ]);
            }
            var canvasActualHeight = this.canvas.height -
this.options.padding * 2;
            var canvasActualWidth = this.canvas.width -
this.options.padding * 2;

            //drawing the grid lines
            var gridValue = 0;
            while (gridValue <= maxValue){
                var gridY = canvasActualHeight * (1 -
gridValue/maxValue) + this.options.padding;
                drawLine(
                    this.ctx,
                    0,
                    gridY,
                    this.canvas.width,
                    gridY,
                    this.options.gridColor
                );

                //writing grid markers
                this.ctx.save();
                this.ctx.fillStyle = this.options.gridColor;
                this.ctx.textBaseline="bottom";
                this.ctx.font = "bold 10px Arial";
                this.ctx.fillText(gridValue, 10,gridY - 2);
                this.ctx.restore();

                gridValue+=this.options.gridScale;
            }

            //drawing the bars
```

```

var barIndex = 0;
var numberOfBars = Object.keys(this.options.data).length;
var barSize = (canvasActualWidth)/numberOfBars;

for (categ in this.options.data){
    var val = this.options.data[categ];
    var barHeight = Math.round( canvasActualHeight *
val/maxValue) ;

    drawBar(
        this.ctx,
        this.options.padding + barIndex * barSize,
        this.canvas.height - barHeight -
this.options.padding,

        barSize,
        barHeight,
        this.colors[barIndex%this.colors.length]
    );

    barIndex++;
}

//drawing series name
this.ctx.save();
this.ctx.textBaseline="bottom";
this.ctx.textAlign="center";
this.ctx.fillStyle = "#fff";
this.ctx.font = "bold 24px Arial";
this.ctx.fillText(this.options.seriesName,
this.canvas.width/2,this.canvas.height);
this.ctx.restore();

//draw legend
barIndex = 0;
var legend =
document.querySelector("legend[for='myCanvas']");
var ul = document.createElement("ul");
legend.append(ul);
for (categ in this.options.data){
    var li = document.createElement("li");
    li.style.listStyle = "none";
    li.style.borderLeft = "20px solid
"+this.colors[barIndex%this.colors.length];
    li.style.padding = "5px";
    li.textContent = categ;
    ul.append(li);
    barIndex++;
}

```

```

    }
  }
}
var myBarchart = new Barchart(
{
    canvas:myCanvas,
    seriesName:"Sales Data of January",
    padding:40,
    gridScale:500,
    gridColor:"#eeeeee",
    data:myVinyls,
    colors:["#a55ca5","#67b6c7",
"#bccd7a","#eb9743","#32a852","#a432a8"]
}
);
myBarchart.draw();
</script>
</body>

</html>

```

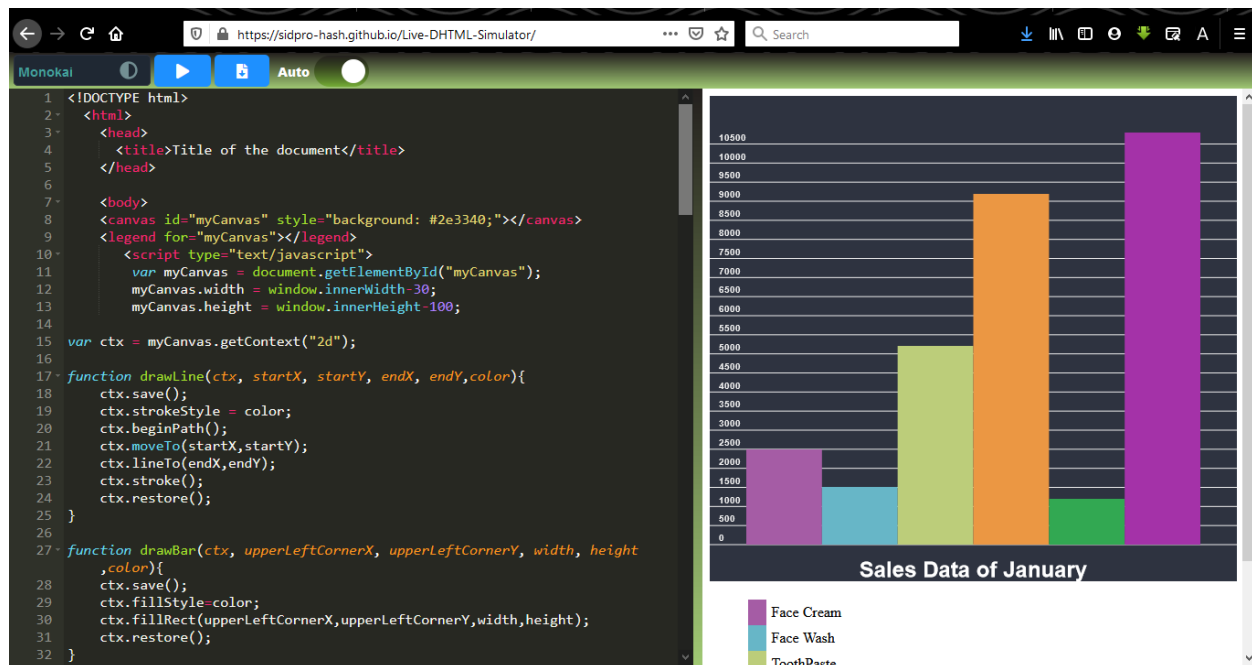


Fig4 – a

## b) Read the data .txt file and draw Simple Bar Chart

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read Text File to Draw Simple Bar Chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="jquery-3.5.1.min.js"></script>
    <style>
      #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;
        width: 100%;
      }

      #mytable td, #mytable th {
        border: 1px solid #ddd;
        padding: 8px;
      }

      #mytable tr:nth-child(even){ background-color: #f2f2f2;}

      #mytable tr:hover {background-color: #ddd;}

      #mytable th {
        padding-top: 12px;
        padding-bottom: 12px;
        text-align: left;
        background-color: cornflowerblue;
        color: white;
      }
      legend{
        float:right;
      }
    </style>
  </head>

```

```

<body>
  <h1>Get data from text file and draw Simple Bar Chart</h1>
  <label style="font-size:18px" for="month"><b>Choose a Month:
</b></label>
  <select id="month" onchange="draw()">
    <option value="1">January</option>
    <option value="2" selected>February</option>
    <option value="3">March</option>
    <option value="4">April</option>
    <option value="5">May</option>
    <option value="6">June</option>
    <option value="7">July</option>
    <option value="8">August</option>
    <option value="9">September</option>
    <option value="10">October</option>
    <option value="11">November</option>
    <option value="12">December</option>
  </select><br><br>

  <canvas id="myCanvas" style="background: #2e3340;"></canvas>
<legend for="myCanvas"></legend>
</body>
<script>
var myCanvas = document.getElementById("myCanvas");
myCanvas.width = window.innerWidth-200;
myCanvas.height = window.innerHeight-120;

var ctx = myCanvas.getContext("2d");

function drawLine(ctx, startX, startY, endX, endY,color){
  ctx.save();
  ctx.strokeStyle = color;
  ctx.beginPath();
  ctx.moveTo(startX,startY);
  ctx.lineTo(endX,endY);
  ctx.stroke();
  ctx.restore();
}

function drawBar(ctx, upperLeftCornerX, upperLeftCornerY, width,
height,color){
  ctx.save();
  ctx.fillStyle=color;

```

```
        ctx.fillRect(upperLeftCornerX,upperLeftCornerY,width,height);
        ctx.restore();
    }

    var Barchart = function(options){
        this.options = options;
        this.canvas = options.canvas;
        this.ctx = this.canvas.getContext("2d");
        this.colors = options.colors;

        this.draw = function(){
            var maxValue = 0;
            for (var categ in this.options.data){
                maxValue =
Math.max(maxValue,this.options.data[categ]);
            }
            var canvasActualHeight = this.canvas.height -
this.options.padding * 2;
            var canvasActualWidth = this.canvas.width -
this.options.padding * 2;

            //drawing the grid lines
            var gridValue = 0;
            while (gridValue <= maxValue){
                var gridY = canvasActualHeight * (1 -
gridValue/maxValue) + this.options.padding;
                drawLine(
                    this.ctx,
                    0,
                    gridY,
                    this.canvas.width,
                    gridY,
                    this.options.gridColor
                );

                //writing grid markers
                this.ctx.save();
                this.ctx.fillStyle = this.options.gridColor;
                this.ctx.textBaseline="bottom";
                this.ctx.font = "bold 10px Arial";
                this.ctx.fillText(gridValue, 10,gridY - 2);
                this.ctx.restore();
            }
        }
    }
```

```

        gridValue+=this.options.gridScale;
    }

    //drawing the bars
    var barIndex = 0;
    var numberOfBars = Object.keys(this.options.data).length;
    var barSize = (canvasActualWidth)/numberOfBars;

    for (categ in this.options.data){
        var val = this.options.data[categ];
        var barHeight = Math.round( canvasActualHeight *
val/maxValue) ;

        drawBar(
            this.ctx,
            this.options.padding + barIndex * barSize,
            this.canvas.height - barHeight -
this.options.padding,

            barSize,
            barHeight,
            this.colors[barIndex%this.colors.length]
        );

        barIndex++;
    }

    //drawing series name
    this.ctx.save();
    this.ctx.textBaseline="bottom";
    this.ctx.textAlign="center";
    this.ctx.fillStyle = "#fff";
    this.ctx.font = "bold 24px Arial";
    this.ctx.fillText(this.options.seriesName,
this.canvas.width/2,this.canvas.height);
    this.ctx.restore();

    //draw legend
    barIndex = 0;
    var legend =
document.querySelector("legend[for='myCanvas']");
    var ul = document.createElement("ul");
    legend.append(ul);
    for (categ in this.options.data){

```



```

        var li = document.createElement("li");
        li.style.listStyle = "none";
        li.style.borderLeft = "20px solid
"+this.colors[barIndex%this.colors.length];
        li.style.padding = "5px";
        li.textContent = categ;
        ul.append(li);
        barIndex++;
    }
}
//just change month number
var month = parseInt(document.getElementById('month').value);
var myVinyls = { };
var data;
var months = document.getElementById('month');

$( document ).ready(function() {
    //just change the name of .txt file to load file
    $.get('data.txt', function(theData) {
        theData = theData.replace(/\r/g, "");
        theData = theData.replace(/\t/g, ' ');
        theData = theData.split('\n');
        data = theData;
        totalRows = theData.length;
        theHead = theData[0].split(' ');
        theRow = theData[month].split(' ');
        for(let i=1;i<theHead.length-2;++i)
            myVinyls[theHead[i]] = theRow[i];

        var myBarchart = new Barchart({
            canvas:myCanvas,
            seriesName:"Sales Data of
"+months.options[months.selectedIndex].text,
            padding:40,
            gridScale:500,
            gridColor:"#eeeeee",
            data:myVinyls,
            colors:["#a55ca5","#67b6c7",
"#bccd7a","#eb9743","#32a852","#a432a8"]
        });

        myBarchart.draw();
    });

```

```
        });
    });
    function draw(){
        theData = data;
        month = parseInt(document.getElementById('month').value);
        theHead = theData[0].split(' ');
        theRow = theData[month].split(' ');
        for(let i=1;i<theHead.length-2;++i)
            myVinyls[theHead[i]] = theRow[i];

        var legend = document.querySelector("legend[for='myCanvas']");
        legend.remove();

        var x = document.createElement("LEGEND");
        x.setAttribute("for","myCanvas");
        document.body.appendChild(x);

        ctx.clearRect(0,0,myCanvas.width,myCanvas.height);
        var myBarchart = new Barchart({
            canvas:myCanvas,
            seriesName:"Sales Data of
"+months.options[months.selectedIndex].text,
            padding:40,
            gridScale:500,
            gridColor:"#eeeeee",
            data:myVinyls,
            colors:["#a55ca5","#67b6c7",
"#bccd7a","#eb9743","#32a852","#a432a8"]
        });

        myBarchart.draw();
    }
</script>
</html>
```

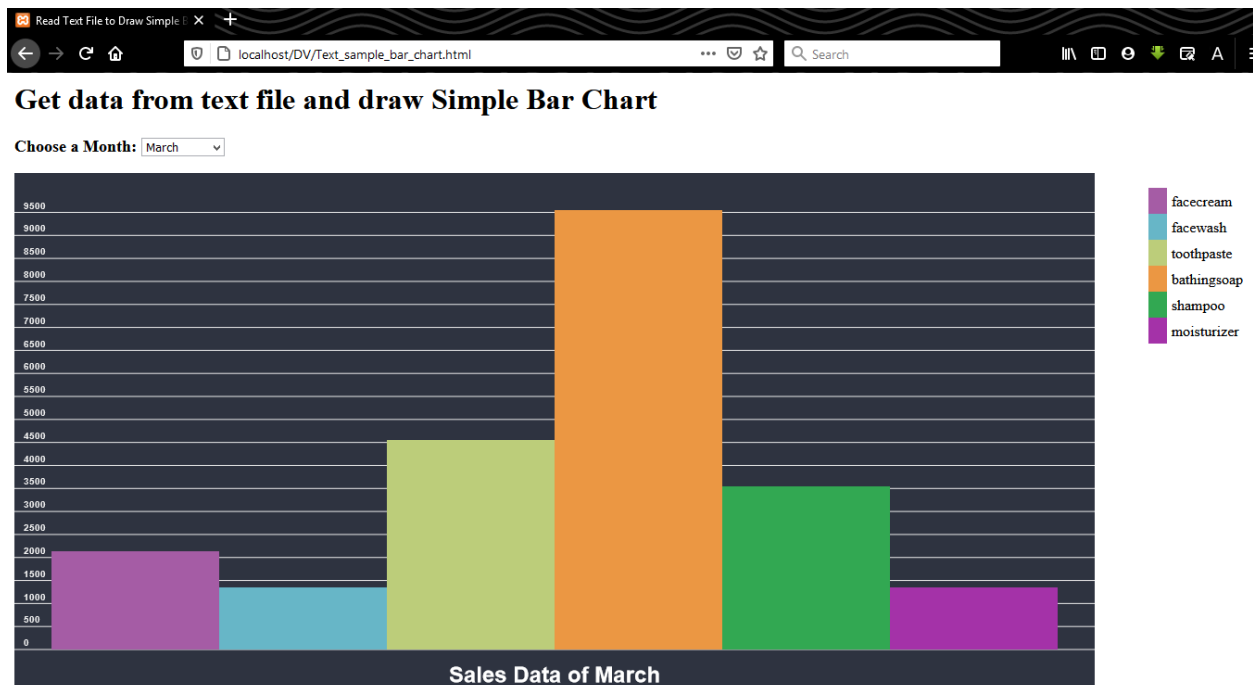


Fig4 – b

## c) Read the data .csv file and draw Column Bar Chart

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read CSV File to Draw Simple Column Bar Chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="jquery-3.5.1.min.js"></script>
    <style>
      #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;
        width: 100%;
      }

```

```

        #mytable td, #mytable th {
            border: 1px solid #ddd;
            padding: 8px;
        }

        #mytable tr:nth-child(even){background-color: #f2f2f2;}

        #mytable tr:hover {background-color: #ddd;}

        #mytable th {
            padding-top: 12px;
            padding-bottom: 12px;
            text-align: left;
            background-color: cornflowerblue;
            color: white;
        }
        legend{
            float:right;
        }
    </style>
</head>
<body>
    <h1>Get data from CSV file and draw Column Simple Bar Chart</h1>
    <label style="font-size:18px" for="month"><b>Choose a Month:
</b></label>
    <select id="month" onchange="draw()">
        <option value="1">January</option>
        <option value="2" selected>February</option>
        <option value="3">March</option>
        <option value="4">April</option>
        <option value="5">May</option>
        <option value="6">June</option>
        <option value="7">July</option>
        <option value="8">August</option>
        <option value="9">September</option>
        <option value="10">October</option>
        <option value="11">November</option>
        <option value="12">December</option>
    </select><br><br>

    <canvas id="myCanvas" style="background: #86A09E;"></canvas>
    <legend for="myCanvas"></legend>
</body>

```

```
<script>
var myCanvas = document.getElementById("myCanvas");
myCanvas.width = window.innerWidth-200;
myCanvas.height = window.innerHeight-120;

var ctx = myCanvas.getContext("2d");

function drawLine(ctx, startX, startY, endX, endY,color){
    ctx.save();
    ctx.strokeStyle = color;
    ctx.beginPath();
    ctx.moveTo(startX,startY);
    ctx.lineTo(endX,endY);
    ctx.stroke();
    ctx.restore();
}

function drawBar(ctx, upperLeftCornerX, upperLeftCornerY, width,
height,color){
    ctx.save();
    ctx.fillStyle=color;
    ctx.fillRect(upperLeftCornerX,upperLeftCornerY,width,height);
    ctx.restore();
}

var Barchart = function(options){
    this.options = options;
    this.canvas = options.canvas;
    this.ctx = this.canvas.getContext("2d");
    this.colors = options.colors;

    this.draw = function(){
        var maxVal = 0;
        for (var categ in this.options.data){
            maxVal =
Math.max(maxVal,this.options.data[categ]);
        }
        var canvasActualHeight = this.canvas.height -
this.options.padding * 2;
        var canvasActualWidth = this.canvas.width -
this.options.padding * 2;
```

```
//drawing the grid lines
var gridValue = 0;
while (gridValue <= maxValue){
    var gridY = canvasActualHeight * (1 -
gridValue/maxValue) + this.options.padding;
    drawLine(
        this.ctx,
        0,
        gridY,
        this.canvas.width,
        gridY,
        this.options.gridColor
    );

    //writing grid markers
    this.ctx.save();
    this.ctx.fillStyle = this.options.gridColor;
    this.ctx.textBaseline="bottom";
    this.ctx.font = "bold 10px Arial";
    this.ctx.fillText(gridValue, 10,gridY - 2);
    this.ctx.restore();

    gridValue+=this.options.gridScale;
}

//drawing the bars
var barIndex = 0;
var numberOfBars = Object.keys(this.options.data).length;
var barSize = (canvasActualWidth)/numberOfBars;

for (categ in this.options.data){
    var val = this.options.data[categ];
    var barHeight = Math.round( canvasActualHeight *
val/maxValue) ;

    drawBar(
        this.ctx,
        this.options.padding + barIndex * barSize,
        this.canvas.height - barHeight -
this.options.padding,
        barSize,
        barHeight,
        this.colors[barIndex%this.colors.length]
    );
}
```

```

        barIndex++;
    }

    //drawing series name
    this.ctx.save();
    this.ctx.textBaseline="bottom";
    this.ctx.textAlign="center";
    this.ctx.fillStyle = "#fff";
    this.ctx.font = "bold 24px Arial";
    this.ctx.fillText(this.options.seriesName,
this.canvas.width/2,this.canvas.height);
    this.ctx.restore();

    //draw legend
    barIndex = 0;
    var legend =
document.querySelector("legend[for='myCanvas']");
    var ul = document.createElement("ul");
    legend.append(ul);
    for (categ in this.options.data){
        var li = document.createElement("li");
        li.style.listStyle = "none";
        li.style.borderLeft = "20px solid
"+this.colors[barIndex%this.colors.length];
        li.style.padding = "5px";
        li.textContent = categ;
        ul.append(li);
        barIndex++;
    }
    }

    //just change month number
    var month = parseInt(document.getElementById('month').value);
    var myVinyls = { };
    var data;
    var months = document.getElementById('month');

    $( document ).ready(function() {
        //just change the name of .csv file to load file
        $.get('data.csv', function(theData) {
            theData = theData.replace("/g,");
            theData = theData.split(/\r?\n\r/);

```

```

        console.log(theData);
        totalRows = theData.length;
        data = theData;
        totalRows = theData.length;
        theHead = theData[0].split(',');
        theRow = theData[month].split(',');
        for(let i=1;i<theHead.length-2;++i)
            myVinyls[theHead[i]] = theRow[i];

        var myBarchart = new Barchart({
            canvas:myCanvas,
            seriesName:"Sales Data of
"+months.options[months.selectedIndex].text,
            padding:40,
            gridScale:500,
            gridColor:"#eeeeee",
            data:myVinyls,
            colors:["#2A9D8F","#E9C46A",
"#F4A261","#30BCED","#D9BA41","#7D001A"]
        });

        myBarchart.draw();
    });
    function draw(){
        theData = data;
        month = parseInt(document.getElementById('month').value);
        theHead = theData[0].split(',');
        theRow = theData[month].split(',');
        for(let i=1;i<theHead.length-2;++i)
            myVinyls[theHead[i]] = theRow[i];

        var legend = document.querySelector("legend[for='myCanvas']");
        legend.remove();

        var x = document.createElement("LEGEND");
        x.setAttribute("for","myCanvas");
        document.body.appendChild(x);

        ctx.clearRect(0,0,myCanvas.width,myCanvas.height);
        var myBarchart = new Barchart({
            canvas:myCanvas,

```



```

        seriesName:"Sales Data of
"+months.options[months.selectedIndex].text,
        padding:40,
        gridScale:500,
        gridColor:"#eeeeee",
        data:myVinyls,
        colors:["#2A9D8F","#E9C46A",
"#F4A261","#30BCED","#D9BA41","#7D001A"]
    });

    myBarchart.draw();
}
</script>
</html>

```

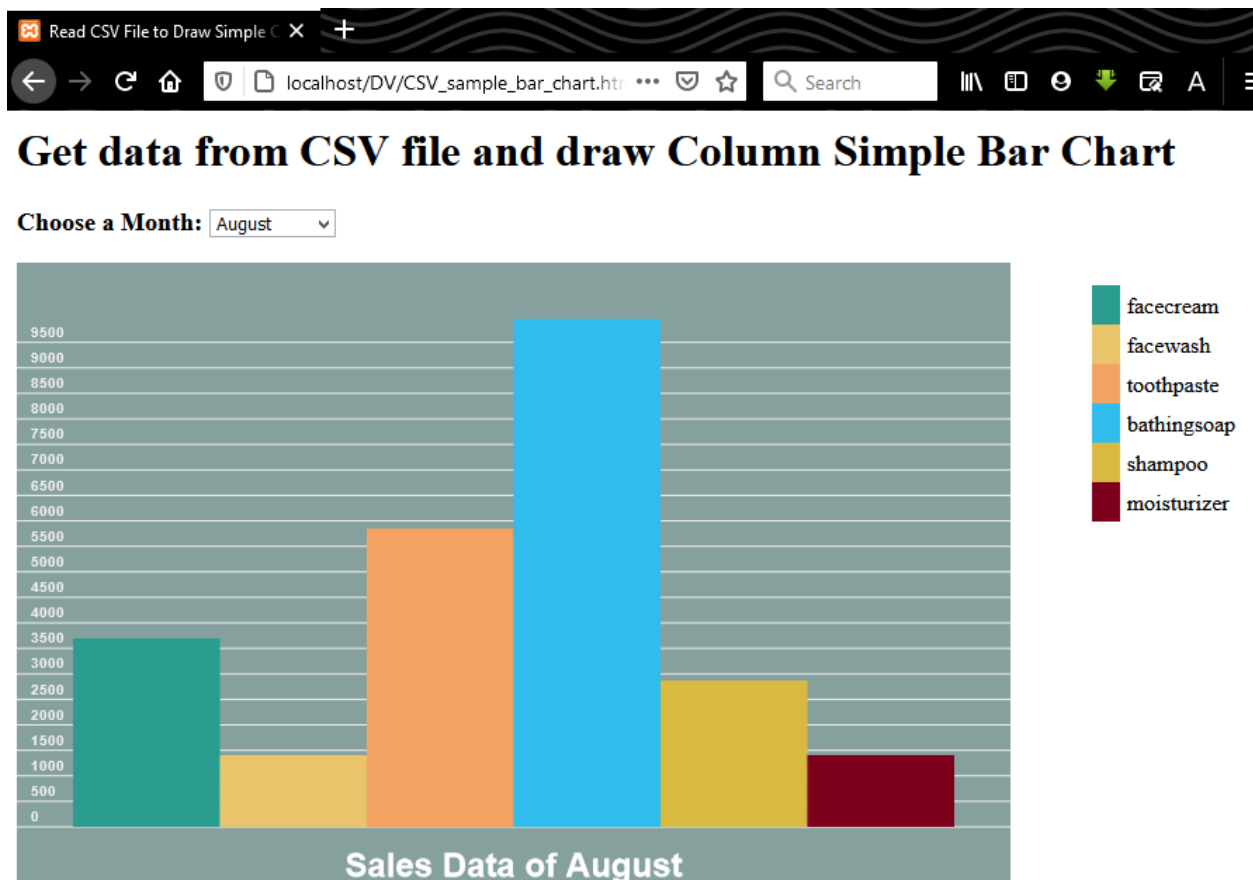


Fig4 – c

## d) Read the data XML file and draw Simple Chart

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read XML File to Draw Simple Bar Chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="jquery-3.5.1.min.js"></script>
    <style>
      #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;
        width: 100%;
      }
      body{
        background:cornflowerblue;
      }
      #mytable td, #mytable th {
        border: 1px solid #ddd;
        padding: 8px;
      }

      #mytable tr:nth-child(even){ background-color: #f2f2f2;}

      #mytable tr:hover {background-color: #ddd;}

      #mytable th {
        padding-top: 12px;
        padding-bottom: 12px;
        text-align: left;
        background-color: cornflowerblue;
        color: white;
      }
      legend{
        float:right;
      }

```

```

        </style>
    </head>
    <body>
        <h1>Get data from XML file and draw Simple Bar Chart</h1>
        <canvas id="myCanvas" style="background: #2e3340;"></canvas>
    <legend for="myCanvas"></legend>
    </body>
    <script>
var myCanvas = document.getElementById("myCanvas");
myCanvas.width = window.innerWidth-200;
myCanvas.height = window.innerHeight-120;

        var ctx = myCanvas.getContext("2d");

        function drawLine(ctx, startX, startY, endX, endY,color){
            ctx.save();
            ctx.strokeStyle = color;
            ctx.beginPath();
            ctx.moveTo(startX,startY);
            ctx.lineTo(endX,endY);
            ctx.stroke();
            ctx.restore();
        }

        function drawBar(ctx, upperLeftCornerX, upperLeftCornerY, width,
height,color){
            ctx.save();
            ctx.fillStyle=color;
            ctx.fillRect(upperLeftCornerX,upperLeftCornerY,width,height);
            ctx.restore();
        }

        var Barchart = function(options){
            this.options = options;
            this.canvas = options.canvas;
            this.ctx = this.canvas.getContext("2d");
            this.colors = options.colors;

            this.draw = function(){
                var maxValue = 0;
                for (var categ in this.options.data){

```

```
        maxValue =
Math.max(maxValue,this.options.data[categ]);
    }
    var canvasActualHeight = this.canvas.height -
this.options.padding * 2;
    var canvasActualWidth = this.canvas.width -
this.options.padding * 2;

    //drawing the grid lines
    var gridValue = 0;
    while (gridValue <= maxValue){
        var gridY = canvasActualHeight * (1 -
gridValue/maxValue) + this.options.padding;
        drawLine(
            this.ctx,
            0,
            gridY,
            this.canvas.width,
            gridY,
            this.options.gridColor
        );

        //writing grid markers
        this.ctx.save();
        this.ctx.fillStyle = this.options.gridColor;
        this.ctx.textBaseline="bottom";
        this.ctx.font = "bold 10px Arial";
        this.ctx.fillText(gridValue, 10,gridY - 2);
        this.ctx.restore();

        gridValue+=this.options.gridScale;
    }

    //drawing the bars
    var barIndex = 0;
    var numberOfBars = Object.keys(this.options.data).length;
    var barSize = (canvasActualWidth)/numberOfBars;

    for (categ in this.options.data){
        var val = this.options.data[categ];
        var barHeight = Math.round( canvasActualHeight *
val/maxValue) ;

        drawBar(
```

```

        this.ctx,
        this.options.padding + barIndex * barSize,
        this.canvas.height - barHeight -

this.options.padding,

        barSize,
        barHeight,
        this.colors[barIndex%this.colors.length]

    );

    barIndex++;
}

//drawing series name
this.ctx.save();
this.ctx.textBaseline="bottom";
this.ctx.textAlign="center";
this.ctx.fillStyle = "#fff";
this.ctx.font = "bold 24px Arial";
this.ctx.fillText(this.options.seriesName,
this.canvas.width/2,this.canvas.height);
this.ctx.restore();

//draw legend
barIndex = 0;
var legend =
document.querySelector("legend[for='myCanvas']");
var ul = document.createElement("ul");
legend.append(ul);
for (categ in this.options.data){
    var li = document.createElement("li");
    li.style.listStyle = "none";
    li.style.borderLeft = "20px solid
"+this.colors[barIndex%this.colors.length];
    li.style.padding = "5px";
    li.textContent = categ;
    ul.append(li);
    barIndex++;
}
}
var myVinyls = { };
var data;
$( document ).ready(function() {

```

```
$.ajax({
  type: "GET",
  url: "regional_sales.xml",
  dataType: "xml",
  success: xmlParser
});
});

function xmlParser(xml) {
  $(xml).find('region').each(function () {
    var theTerritory = $(this).find('territory').text();
    var numEmployees = $(this).find('employees').text();
    var theAmount = $(this).find('amount').text();
    theAmount = theAmount.replace(/,/g,"");
    myVinyls[theTerritory] = parseInt(theAmount);
    console.log(parseInt(theAmount));
    console.log(theTerritory);
  });

  var myBarchart = new Barchart({
    canvas:myCanvas,
    seriesName:"Regional Sales Data",
    padding:50,
    gridScale:50000,
    gridColor:"#eeeeee",
    data:myVinyls,
    colors:["#2A9D8F","#E9C46A",
"#F4A261","#30BCED","#D9BA41","#7D001A"]
  });

  myBarchart.draw();
}
</script>
</html>
```

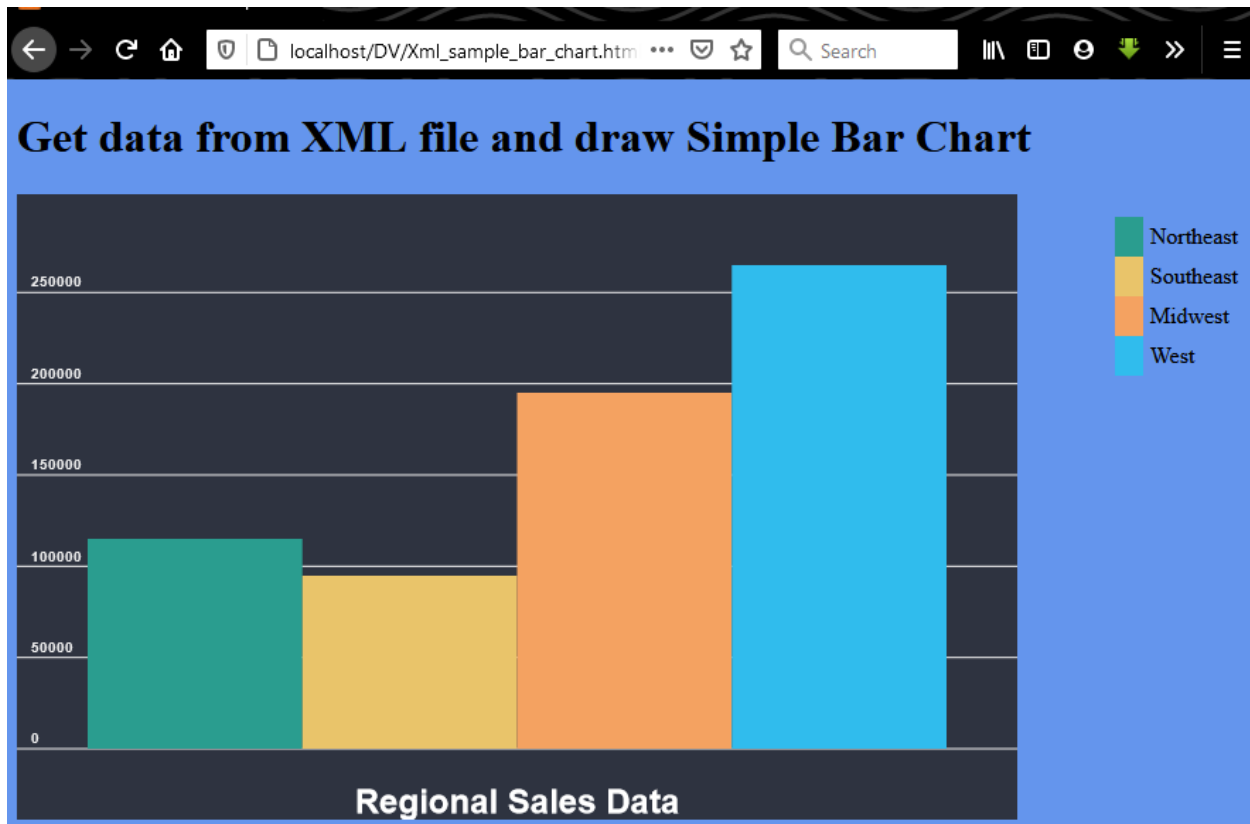


Fig4 – d

## e) Read JSON Data and draw Simple Chart

```

<!DOCTYPE html>
<html lang="en-IN" dir="ltr">
  <head>
    <title>Read JSON File to Draw Simple Bar Chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Collegeek is online learning platform
where we consistently strive to offer the best of education"/>
    <meta name="keywords" content="Collegeek,learning
programing,programming tutorial"/>
    <script src="jquery-3.5.1.min.js"></script>
    <style>
      #mytable {
        font-family: Arial, Helvetica, sans-serif;
        border-collapse: collapse;

```

```
        width: 100%;
    }
    body{
        background:#e4fde1;
    }
    #mytable td, #mytable th {
        border: 1px solid #ddd;
        padding: 8px;
    }

    #mytable tr:nth-child(even){ background-color: #f2f2f2;}

    #mytable tr:hover {background-color: #ddd;}

    #mytable th {
        padding-top: 12px;
        padding-bottom: 12px;
        text-align: left;
        background-color: cornflowerblue;
        color: white;
    }
    legend{
        float:right;
    }
</style>
</head>
<body>
    <h1>Get data from JSON file and draw Simple Bar Chart</h1>
    <canvas id="myCanvas" style="background: #2e3340;"></canvas>
<legend for="myCanvas"></legend>
</body>
<script>
var myCanvas = document.getElementById("myCanvas");
myCanvas.width = window.innerWidth-200;
myCanvas.height = window.innerHeight-120;

    var ctx = myCanvas.getContext("2d");

    function drawLine(ctx, startX, startY, endX, endY,color){
        ctx.save();
        ctx.strokeStyle = color;
        ctx.beginPath();
        ctx.moveTo(startX,startY);
```



```
        ctx.lineTo(endX,endY);
        ctx.stroke();
        ctx.restore();
    }

    function drawBar(ctx, upperLeftCornerX, upperLeftCornerY, width,
height,color){
        ctx.save();
        ctx.fillStyle=color;
        ctx.fillRect(upperLeftCornerX,upperLeftCornerY,width,height);
        ctx.restore();
    }

    var Barchart = function(options){
        this.options = options;
        this.canvas = options.canvas;
        this.ctx = this.canvas.getContext("2d");
        this.colors = options.colors;

        this.draw = function(){
            var maxValue = 0;
            for (var categ in this.options.data){
                maxValue =
Math.max(maxValue,this.options.data[categ]);
            }
            var canvasActualHeight = this.canvas.height -
this.options.padding * 2;
            var canvasActualWidth = this.canvas.width -
this.options.padding * 2;

            //drawing the grid lines
            var gridValue = 0;
            while (gridValue <= maxValue){
                var gridY = canvasActualHeight * (1 -
gridValue/maxValue) + this.options.padding;
                drawLine(
                    this.ctx,
                    0,
                    gridY,
                    this.canvas.width,
                    gridY,
                    this.options.gridColor
                );
            }
        }
    }
```

```
);

//writing grid markers
this.ctx.save();
this.ctx.fillStyle = this.options.gridColor;
this.ctx.textBaseline="bottom";
this.ctx.font = "bold 10px Arial";
this.ctx.fillText(gridValue, 10,gridY - 2);
this.ctx.restore();

gridValue+=this.options.gridScale;
}

//drawing the bars
var barIndex = 0;
var numberOfBars = Object.keys(this.options.data).length;
var barSize = (canvasActualWidth)/numberOfBars;

for (categ in this.options.data){
    var val = this.options.data[categ];
    var barHeight = Math.round( canvasActualHeight *
val/maxValue) ;

    drawBar(
        this.ctx,
        this.options.padding + barIndex * barSize,
        this.canvas.height - barHeight -
        this.options.padding,
        barSize,
        barHeight,
        this.colors[barIndex%this.colors.length]
    );

    barIndex++;
}

//drawing series name
this.ctx.save();
this.ctx.textBaseline="bottom";
this.ctx.textAlign="center";
this.ctx.fillStyle = "#fff";
this.ctx.font = "bold 24px Arial";
this.ctx.fillText(this.options.seriesName,
this.canvas.width/2,this.canvas.height);
```

```

        this.ctx.restore();

        //draw legend
        barIndex = 0;
        var legend =
document.querySelector("legend[for='myCanvas']");
        var ul = document.createElement("ul");
        legend.append(ul);
        for (categ in this.options.data){
            var li = document.createElement("li");
            li.style.listStyle = "none";
            li.style.borderLeft = "20px solid
"+this.colors[barIndex%this.colors.length];
            li.style.padding = "5px";
            li.textContent = categ;
            ul.append(li);
            barIndex++;
        }
    }
}
var myVinyls = {};
var data;
$( document ).ready(function() {
    $.ajax({
        type: "GET",
        url: "regional_sales.json",
        dataType: "json",
        success: jsonParser
    });
});

function jsonParser(json) {
    var theData = json.sales.region;
    theData.forEach(function(v){
        myVinyls[v.territory]=parseInt(v.employees);
    });

    var myBarchart = new Barchart({
        canvas:myCanvas,
        seriesName:"Regional Employees Data",
        padding:50,
        gridScale:25,
        gridColor:"#eeeeee",

```

```
data:myVinyls,  
  colors:["#ED553B","#20639B",  
"#3CAEA3","#F6D55C","#173F5F","#7D001A"]  
});  
  
myBarchart.draw();  
}  
</script>  
</html>
```

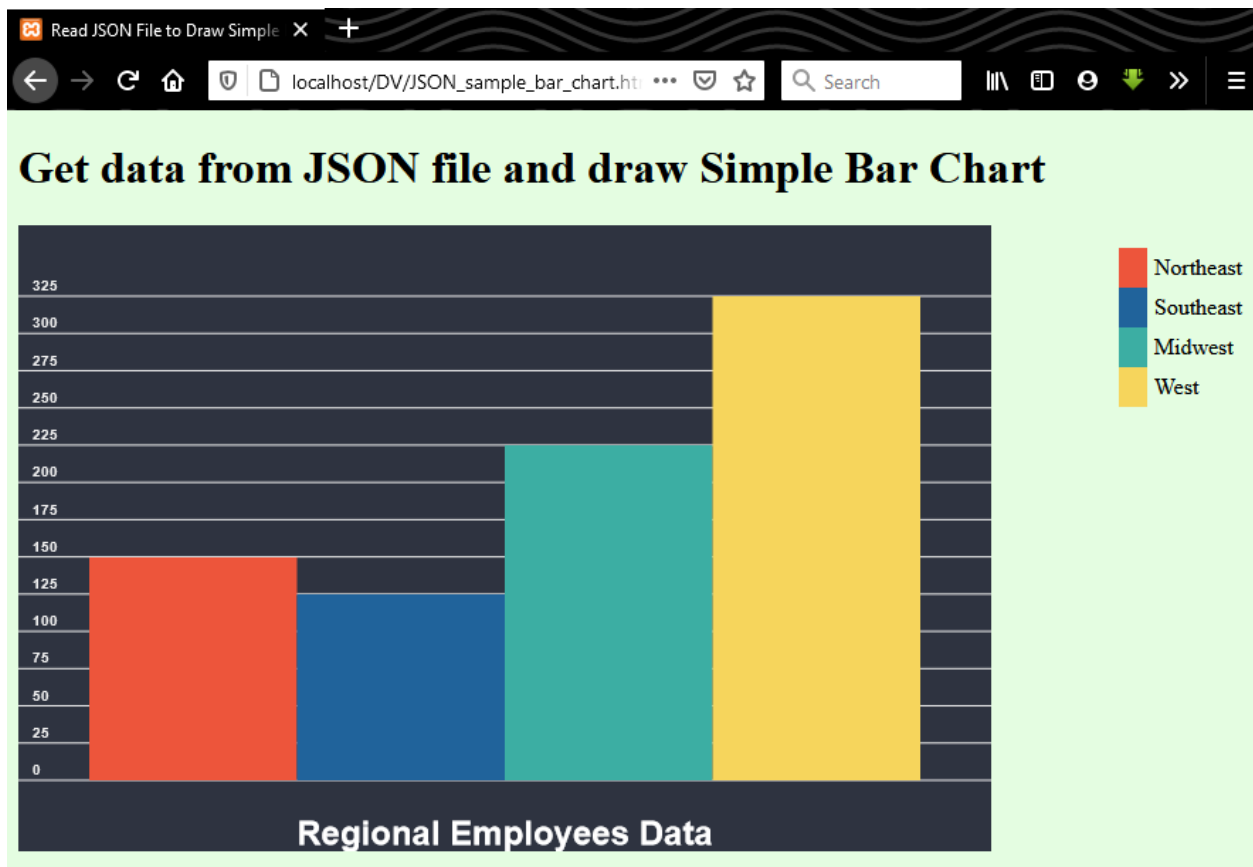


Fig4 – e

## Practical – 5

**AIM:** Develop Following Program Using HTML5 and D3.js and Canvas.js

a) Showing the data as a column chart (simple)

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Showing the Data as a column chart using D3.js</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1.0"/>
  <meta name="description" content="Collegeek is online learning platform where
we consistently strive to offer the best of education"/>
  <meta name="keywords" content="Collegeek,learning programing,programming
tutorial"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="d3.v5.min.js"></script>
  <style>
    body {
      font-family: 'Open Sans', sans-serif;
    }

    div#layout {
      text-align: center;
    }

    div#container {
      width: 1000px;
      height: 600px;
      margin: auto;
      background-color: #2F4A6D;
    }

    svg {
      width: 100%;
      height: 100%;
    }

    .bar {
      fill: #80cbc4;
    }

    text {
      font-size: 12px;
```

```
    fill: #fff;
  }

  path {
    stroke: gray;
  }

  line {
    stroke: gray;
  }

  line#limit {
    stroke: #FED966;
    stroke-width: 3;
    stroke-dasharray: 3 6;
  }

  .grid path {
    stroke-width: 0;
  }

  .grid .tick line {
    stroke: #9FAAAE;
    stroke-opacity: 0.3;
  }

  text.divergence {
    font-size: 14px;
    fill: #2F4A6D;
  }

  text.value {
    font-size: 14px;
  }

  text.title {
    font-size: 22px;
    font-weight: 600;
  }

  text.label {
    font-size: 14px;
    font-weight: 400;
  }

  text.source {
```

```

        font-size: 10px;
    }
</style>
</head>
<body>
    <h2>Showing the Data as a column chart using D3.js </h2>
    <div id='layout'>
        <div id='container'>
            <svg />
        </div>
    </div>
</body>
<script>

    var month = 2;
    var data = [];
    var data;

    $( document ).ready(function() {
        $.get('data.csv', function(theData) {
            theData = theData.replace(/"/g,"");
            theData = theData.split(/\r?\n\r/);

            totalRows = theData.length;

            totalRows = theData.length;
            theHead = theData[0].split(',');
            theRow = theData[month].split(',');
            for(let i=1;i<theHead.length-2;++i){
                myVinyls = { };
                myVinyls["saleType"] = theHead[i];
                myVinyls["saleAmount"] = parseInt(theRow[i]);
                data.push(myVinyls);
            }
            console.log(data);

            //https://blog.risingstack.com/d3-js-tutorial-bar-charts-with-
javascript/

            const margin = 80; // margin value which gives a little extra padding to
the chart

            const width = 1000 - 2 * margin;
            const height = 600 - 2 * margin;

            const svg = d3.select('svg');
            const svgContainer = d3.select('#container');
            // Padding can be applied with a <g> element translated by the
desired value.

```

```

    const chart = svg.append('g').attr('transform', `translate(${margin},
    ${margin})`);

    /*
    It converts a continuous input domain into a continuous
    output range.
    Notice the range and domain method. The first one takes
    the length that
    should be divided between the limits of the domain values.
    */
    // the SVG coordinate system starts from the top left corner that's
    why
    // the range takes the height as the first parameter and not zero.
    const yScale = d3.scaleLinear().range([height, 0]).domain([0,
    6000]);

    // axis on the left is as simple as adding another group and calling
    d3's
    // axisLeft method with the scaling function as a parameter
    chart.append('g').call(d3.axisLeft(yScale));

    const xScale = d3.scaleBand()
      .range([0, width])
      .domain(data.map((d) => d.saleType))
      .padding(0.2);

    chart.append('g')
      .attr('transform', `translate(0, ${height})`)
      .call(d3.axisBottom(xScale));

    const makeYLines = () => d3.axisLeft().scale(yScale)
    chart.append('g').call(d3.axisLeft(yScale));

    // vertical grid lines
    // chart.append('g')
    //   .attr('class', 'grid')
    //   .attr('transform', `translate(0, ${height})`)
    //   .call(makeXLines()
    //     .tickSize(-height, 0, 0)
    //     .tickFormat("")
    //   )

    chart.append('g')
      .attr('class', 'grid')
      .call(makeYLines()
        .tickSize(-width, 0, 0)

```



```

        .tickFormat("")
    );

    /*
        selectAll elements on the chart which returns with an empty result
        set. Then, data function tells how many
        elements the DOM should be updated with based on the array
        length. enter identifies elements that are missing if
        the data input is longer than the selection. This returns a new
        selection representing the elements that need to
        be added. Usually, this is followed by an append which adds
        elements to the DOM.
    */

```

```

const barGroups = chart.selectAll()
    .data(data)
    .enter()
    .append('g');

barGroups
    .append('rect')
    .attr('class', 'bar')
    .attr('x', (g) => xScale(g.saleType))
    .attr('y', (g) => yScale(g.saleAmount))
    .attr('height', (g) => height - yScale(g.saleAmount))
    .attr('width', xScale.bandwidth())
    .on('mouseenter', function (actual, i) {
        console.log(actual.saleAmount);
        d3.selectAll('.value')
            .attr('opacity', 0)

        d3.select(this)
            .transition()
            .duration(300)
            .attr('opacity', 0.6)
            .attr('x', (a) => xScale(a.saleType) - 5)
            .attr('width', xScale.bandwidth() + 10)

        const y = yScale(actual.saleAmount)

        line = chart.append('line')
            .attr('id', 'limit')
            .attr('x1', 0)
            .attr('y1', y)

```

```

        .attr('x2', width)
        .attr('y2', y)

    barGroups.append('text')
        .attr('class', 'divergence')
        .attr('x', (a) => xScale(a.saleType) +
xScale.bandwidth() / 2)
        .attr('y', (a) => yScale(a.saleAmount) + 30)
        .attr('fill', 'white')
        .attr('text-anchor', 'middle')
        .text((a, idx) => {
            console.log(a.saleAmount);
            console.log(actual.saleAmount);
            const divergence = (a.saleAmount -
actual.saleAmount).toFixed(1)

            let text = "
            if (divergence > 0) text += '+'
            text += `${divergence}`

            return idx !== i ? text : "";
        })
    })
    .on('mouseleave', function () {
        d3.selectAll('.value')
            .attr('opacity', 1)

        d3.select(this)
            .transition()
            .duration(300)
            .attr('opacity', 1)
            .attr('x', (a) => xScale(a.saleType))
            .attr('width', xScale.bandwidth())

        chart.selectAll('#limit').remove()
        chart.selectAll('.divergence').remove()
    })
    /*chart.selectAll()
        .data(data.length)
        .enter()
        .append('rect')
        .attr('x', (s) => xScale(s.saleType))
        .attr('y', (s) => yScale(s.saleAmount))
        .attr('height', (s) => height - yScale(s.saleAmount))
        .attr('width', xScale.bandwidth())

```

```

        .attr('x', (actual, index, data) =>
xScale(actual.saleAmount));    */

    barGroups
    .append('text')
    .attr('class', 'value')
    .attr('x', (a) => xScale(a.saleType) + xScale.bandwidth() / 2)
    .attr('y', (a) => yScale(a.saleAmount) + 30)
    .attr('text-anchor', 'middle')
    .text((a) => `${a.saleAmount}`)

    svg
    .append('text')
    .attr('class', 'label')
    .attr('x', -(height / 2) - margin)
    .attr('y', margin / 2.4)
    .attr('transform', 'rotate(-90)')
    .attr('text-anchor', 'middle')
    .text('sales Unit')

    svg.append('text')
    .attr('class', 'label')
    .attr('x', width / 2 + margin)
    .attr('y', height + margin * 1.7)
    .attr('text-anchor', 'middle')
    .text('Products sales')

    svg.append('text')
    .attr('class', 'title')
    .attr('x', width / 2 + margin)
    .attr('y', 40)
    .attr('text-anchor', 'middle')
    .text('Sales Data of February Month')

    svg.append('text')
    .attr('class', 'source')
    .attr('x', width - margin / 2)
    .attr('y', height + margin * 1.7)
    .attr('text-anchor', 'start')
    .text('Source: PDS sales.csv, 2020')

    });
  });
</script>
</html>

```

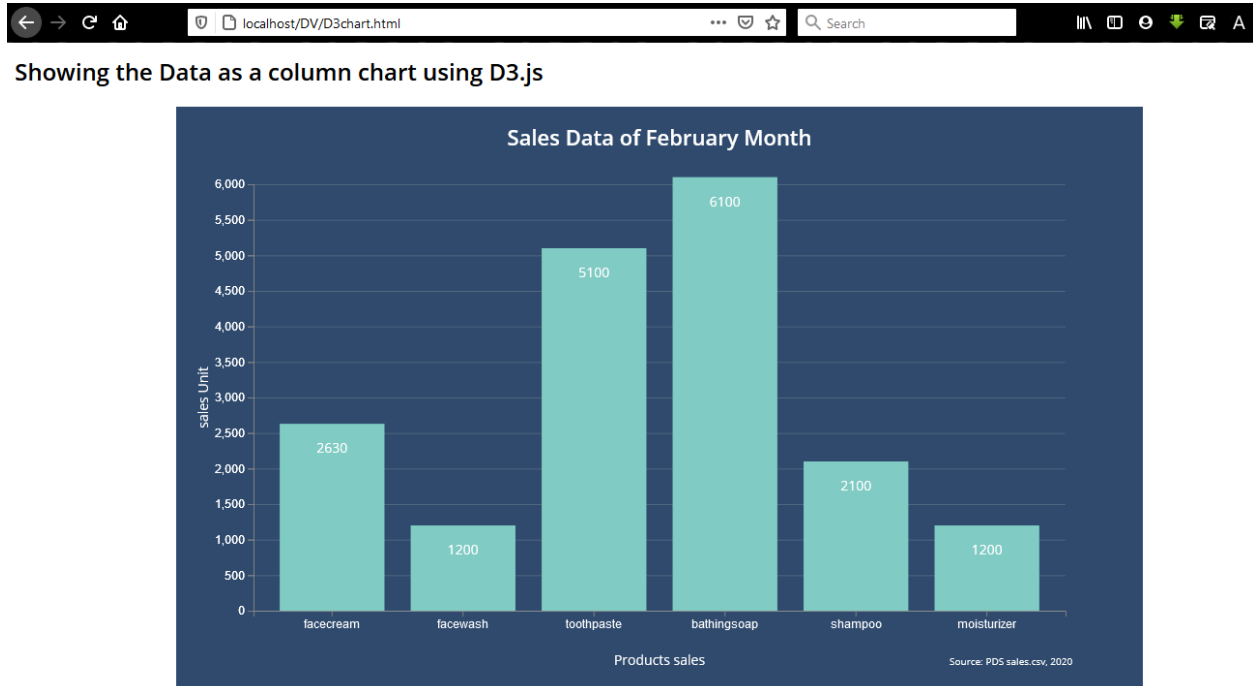


Fig5.a – Sales Data using D3

## b) Showing the data as a stacked column chart

```

<!DOCTYPE HTML>
<html>
<head>
  <title>Showing the Data as a column chart for four age group</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1.0"/>
  <meta name="description" content="Collegeek is online learning platform where
we consistently strive to offer the best of education"/>
  <meta name="keywords" content="Collegeek,learning programing,programming
tutorial"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="canvasjs.min.js"></script>
</head>
<body>
  <h2>Showing the Data as a column chart for died and survived person</h2>
  <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
  <script type="text/javascript">

// Millennials = ages of 18 to 34
// Gen X = ages of 35 to 50

```

```

// Baby Boomers = ages of 51 to 69
// Silent generation = ages of 70 to 87
var age1=0,age2=0,age3=0,age4=0,j=-1;
var age1died=0,age2died=0,age3died=0,age4died=0;
window.onload = function () {

    //just change the name of .csv file to load file
    $.get('titanic.csv', function(theData) {

        theData = theData.replace(/"/g,"");
        theData = theData.split(/\r?\n\r/);
        totalRows = theData.length;

        for(let i=1;i<totalRows;++i){
            theTD = theData[i].split(',');

            value = parseInt(theTD[4]);
            died = (theTD[2]=="died")?true:false;
            if(value>=18 && value<=34){
                if(died)age1died+=1;
                else age1+=1;
            }
            else if(value>=35 && value<=50){
                if(died)age2died+=1;
                else age2+=1;
            }
            else if(value>=51 && value<=69){
                if(died)age3died+=1;
                else age3+=1;
            }
            else if(value>=70 && value<=87){
                if(died)age4died+=1;
                else age4+=1;
            }

        }

        var chart = new CanvasJS.Chart("chartContainer", {
            title:{
                text: "Stacked Column chart for Four Age Group in
Titanic"
            },
            axisX:{
                title: "Four Age group",
            },

```

```

        axisY:{
            interval: 50
        },
        legend: {
            fontSize: 20
        },
        theme: "dark2", // "light1", "light2", "dark1", "dark2"
        data: [
            {
                // Change type to "stackedArea",
                "stackedColumn", "column", "doughnut", "line", "splineArea" etc.
                type: "stackedColumn",
                showInLegend: true,
                legendText: "died",
                dataPoints: [
                    { label: "Millennials died", y:
age1died },
                    { label: "Gen X died", y: age2died },
                    { label: "Baby Boomers died", y:
age3died },
                    { label: "Silent generation died", y:
age4died }
                ]
            },{
                // Change type to "stackedArea",
                "stackedColumn", "column", "doughnut", "line", "splineArea" etc.
                type: "stackedColumn",
                showInLegend: true,
                legendText: "survived",
                dataPoints: [
                    { label: "Millennials", y: age1 },
                    { label: "Gen X", y: age2 },
                    { label: "Baby Boomers", y: age3 },
                    { label: "Silent generation", y: age4
                ]
            }
        ]
    });
    chart.render();
});

```

```
}  
</script>  
</html>
```

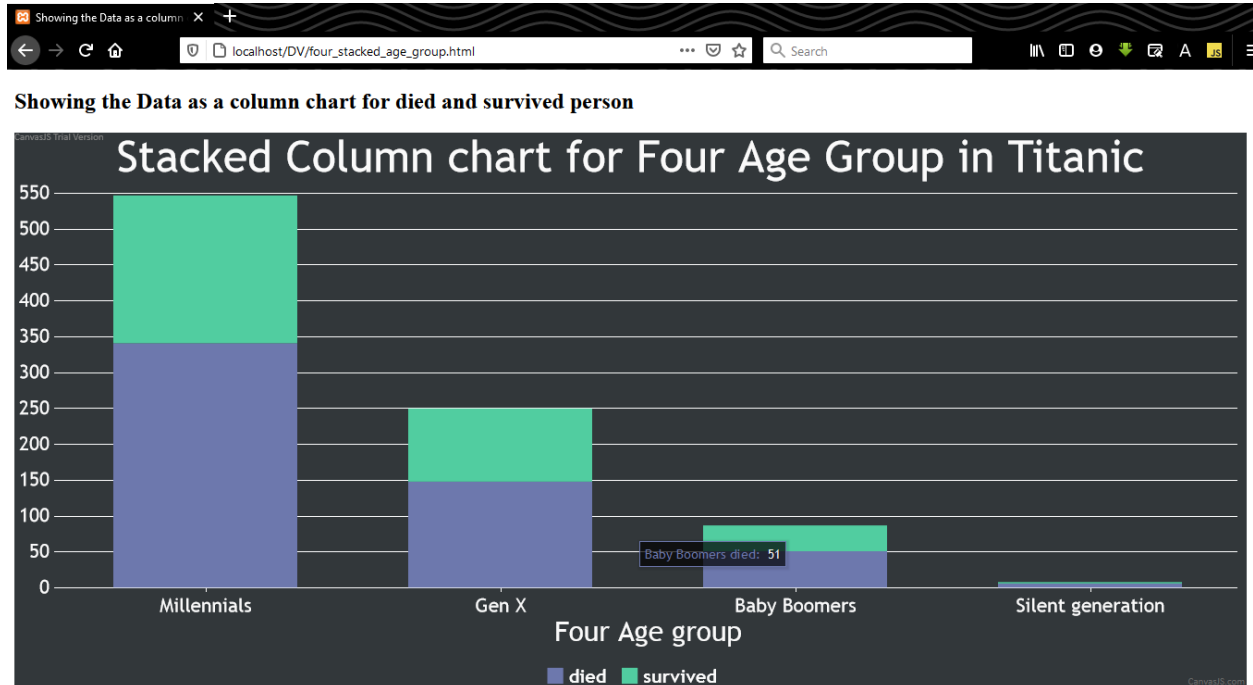


Fig5.b – Stacked column chart of **died and survived** in titanic

## c) Showing the Data as a column chart for four age group

```

<!DOCTYPE HTML>
<html>
<head>
  <title>Showing the Data as a column chart for four age group</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-scale=1.0"/>
  <meta name="description" content="Collegeek is online learning platform where
we consistently strive to offer the best of education"/>
  <meta name="keywords" content="Collegeek,learning programing,programming
tutorial"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="canvasjs.min.js"></script>
</head>
<body>
  <h2>Showing the Data as a column chart for four age group</h2>
  <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
  <script type="text/javascript">

```

```

// Millennials = ages of 18 to 34
// Gen X = ages of 35 to 50
// Baby Boomers = ages of 51 to 69
// Silent generation = ages of 70 to 87

```

```

var age1=0,age2=0,age3=0,age4=0;
window.onload = function () {

  //just change the name of .csv file to load file
  $.get('titanic.csv', function(theData) {
    theData = theData.replace("/g,");
    theData = theData.split(/\r?\n\r/);
    totalRows = theData.length;
    for(let i=1;i<totalRows;++i){
      theTD = theData[i].split(',');
      value = parseInt(theTD[4]);
      if(value>=18 && value<=34)age1+=1;
      else if(value>=35 && value<=50)age2+=1;
      else if(value>=51 && value<=69)age3+=1;
      else if(value>=70 && value<=87)age4+=1;
    }
    var chart = new CanvasJS.Chart("chartContainer", {
      title:{

```



```

    text: "Column chart for Four Age Group in Titanic"
  },
  theme: "dark1", // "light1", "light2", "dark1", "dark2"
  data: [
    {
      // Change type to "column", "doughnut", "line",
      // "splineArea", etc.
      type: "column",
      dataPoints: [
        { label: "Millennials", y: age1 },
        { label: "Gen X", y: age2 },
        { label: "Baby Boomers", y: age3 },
        { label: "Silent generation", y: age4 }
      ]
    }
  ]
});
chart.render();
});
}
</script>
</html>

```

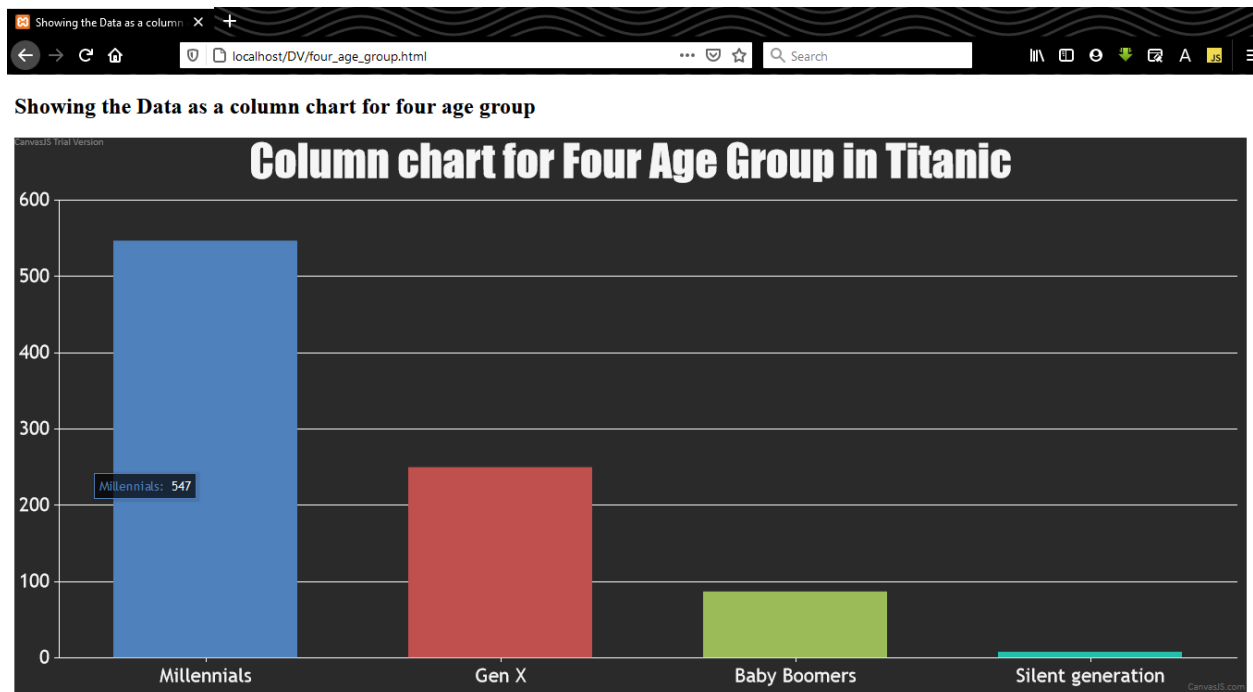


Fig5.c – Column chart for four age group

## Practical – 6

**AIM:** Develop Following Program Using HTML5 and D3.js and Canvas.js

1. Showing the data as a Line chart (single, fewer and multiple lines)

### Single line chart

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Single Line chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Showing the data as a Single Line
chart"/>
  <meta name="keywords" content="Single,Line,Chart,Single Line
Chart,Line"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="canvasjs.min.js"></script>
</head>
<body>
  <h2>Showing the data as a Single Line chart</h2>
  <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
<script type="text/javascript">

window.onload = function () {
  /*CanvasJS.addColorSet("colors",
  [//colorSet Array
    "#2F4F4F",
    "#008080",
    "#2E8B57",
    "#3CB371",
    "#90EE90"
  ]);*/
  var chart = new CanvasJS.Chart("chartContainer", {
    //colorSet: "colors",
    title:{
      text: "Showing the sales data of
Facewash as a Single Line chart",
      fontSize: 30,
```

```
    },
    axisX:{
        title: "Month",
    },
    axisY:{
        title:"Sales Unit",
    },
    theme: "dark1", // "light1", "light2",
    "dark1", "dark2"

    data: [
        {
            // Change type to "column",
            "doughnut", "line", "splineArea", etc.
            type: "line",
            showInLegend: true,
            legendText: "Facewash",
            dataPoints: [
                { label: "Jan", y: 1200 },
                { label: "Feb", y: 2100 },
                { label: "Mar", y: 3550 },
                { label: "Apr", y: 1870 },
                { label: "May", y: 1560 },
                { label: "Jun", y: 1890 },
                { label: "Jul", y: 1780 },
                { label: "Aug", y: 2860 },
                { label: "Sep", y: 2100 },
                { label: "Oct", y: 2300 },
                { label: "Nov", y: 2400 },
                { label: "Dec", y: 1800 }
            ]
        }
    ]
});
chart.render();
}
</script>
</html>
```



### Showing the data as a Single Line chart

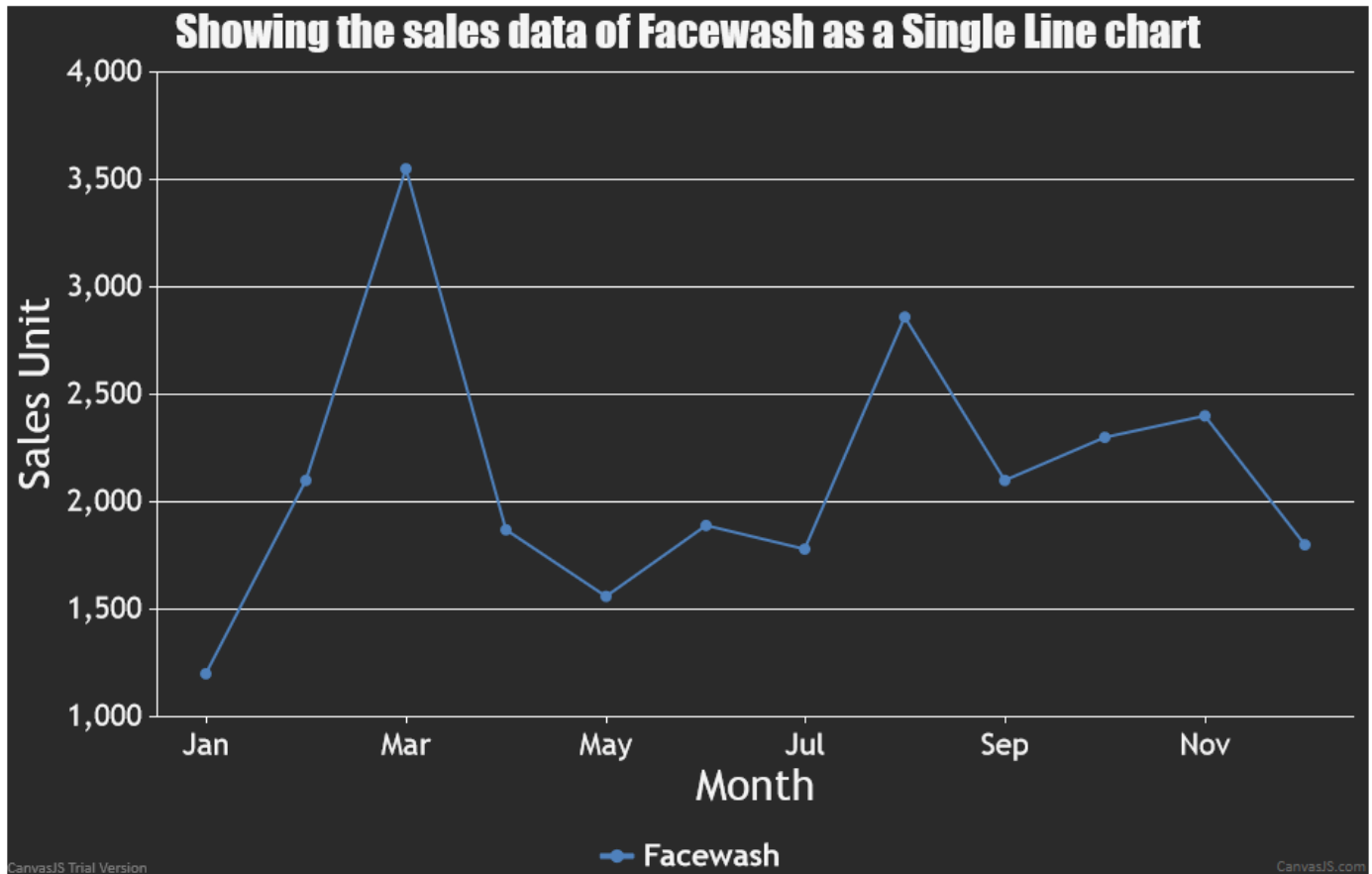


Fig6a1 – single line chart

### Fewer and Multiple line chart

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Multi Line chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Showing the sales data of Products
as a Multi Line chart"/>
  <meta name="keywords" content="Multi,Line,Chart,Multi Line
Chart,Line"/>
</head>
</html>
```

```

<script src="jquery-3.5.1.min.js"></script>
<script src="canvasjs.min.js"></script>
</head>
<body>
  <h2>Showing the sales data of Products as a Multi Line chart</h2>
  <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
<script type="text/javascript">

window.onload = function () {
  /*CanvasJS.addColorSet("colors",
    //colorSet Array
    "#2F4F4F",
    "#008080",
    "#2E8B57",
    "#3CB371",
    "#90EE90"
  );*/

  var chart = new CanvasJS.Chart("chartContainer", {
    //colorSet: "colors",
    title:{
      text: "Showing the sales data of
Products as a Multi Line chart",
      fontSize: 30,
    },
    axisX:{
      title: "Month",
    },
    axisY:{
      title:"Sales Unit",
    },
    theme: "dark1", // "light1", "light2",
"dark1", "dark2"

    data: [
      {
        // Change type to "column",
"doughnut", "line", "splineArea", etc.

        type: "line",
        showInLegend: true,
        legendText: "Facewash",
        dataPoints: [
          { label: "Jan", y: 1200 },
          { label: "Feb", y: 2100 },
          { label: "Mar", y: 3550 },
          { label: "Apr", y: 1870 },

```

```

        { label: "May", y: 1560 },
        { label: "Jun", y: 1890 },
        { label: "Jul", y: 1780 },
        { label: "Aug", y: 2860 },
        { label: "Sep", y: 2100 },
        { label: "Oct", y: 2300 },
        { label: "Nov", y: 2400 },
        { label: "Dec", y: 1800 }
      ]
    }, {
      // Change type to "column",
      "doughnut", "line", "splineArea", etc.

      type: "line",
      showInLegend: true,
      legendText: "Toothpaste",
      dataPoints: [
        { label: "Jan", y: 1500 },
        { label: "Feb", y: 1200 },
        { label: "Mar", y: 1340 },
        { label: "Apr", y: 1130 },
        { label: "May", y: 1740 },
        { label: "Jun", y: 1555 },
        { label: "Jul", y: 1120 },
        { label: "Aug", y: 1400 },
        { label: "Sep", y: 1780 },
        { label: "Oct", y: 1890 },
        { label: "Nov", y: 2100 },
        { label: "Dec", y: 1760 }
      ]
    },
    {
      // Change type to "column",
      "doughnut", "line", "splineArea", etc.

      type: "line",
      showInLegend: true,
      legendText: "Facecream",
      dataPoints: [
        { label: "Jan", y: 2500 },
        { label: "Feb", y: 2630 },
        { label: "Mar", y: 2140 },
        { label: "Apr", y: 3400 },
        { label: "May", y: 3600 },
        { label: "Jun", y: 2760 },
        { label: "Jul", y: 2980 },
        { label: "Aug", y: 3700 },

```

```

        { label: "Sep", y: 3540 },
        { label: "Oct", y: 1990 },
        { label: "Nov", y: 2340 },
        { label: "Dec", y: 2900 }
      ]
    }
  ]
});
chart.render();
}
</script>
</html>

```



Showing the sales data of Products as a Multi Line chart

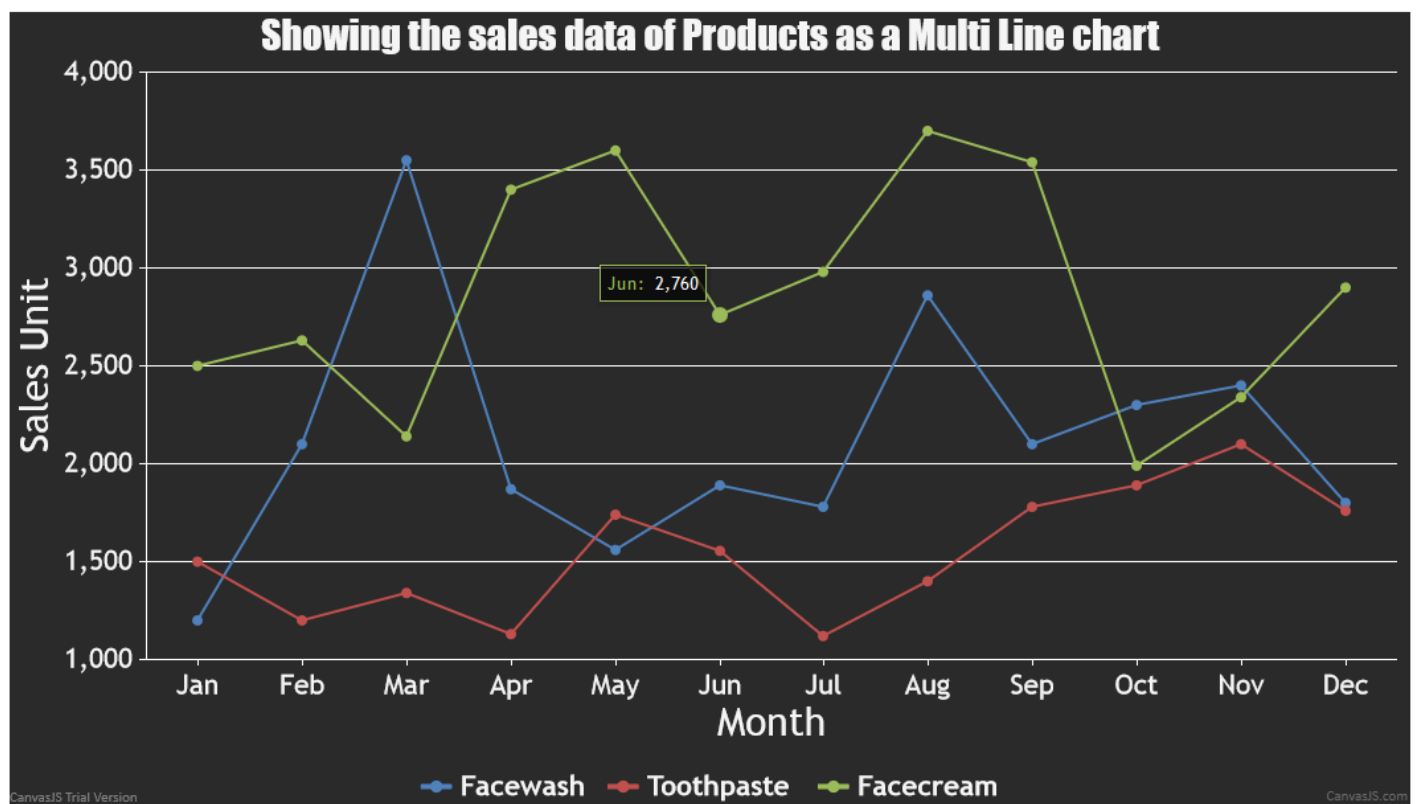


Fig6a2 – multiple line chart

## 2. Showing the data as a Pie Chart (single and multiple pie)

### Single Pie chart

```

<!DOCTYPE HTML>
<html>
<head>
    <title>Single Pie chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Showing the data as a Single Pie
chart"/>
    <meta name="keywords" content="Single,Pie,Chart,Single Pie
Chart,Line"/>
    <script src="jquery-3.5.1.min.js"></script>
    <script src="canvasjs.min.js"></script>
</head>
<body>
    <h2>Showing the data as a Single Pie chart</h2>
    <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
<script type="text/javascript">
var facecream=0,facewash=0,toothpaste=0
,bathingsoap=0,shampoo=0,moisturizer=0;
window.onload = function () {
    $.get('data.txt', function(theData) {
        theData = theData.replace(/\r/g, '');
        theData = theData.replace(/\t/g, ' ');
        theData = theData.split('\n');
        totalRows = theData.length;
        data = theData;
        for(let i=1;i<totalRows;++i){
            theData = data[i].split(' ');
            facecream+=parseInt(theData[1]);
            facewash+=parseInt(theData[2]);
            toothpaste+=parseInt(theData[3]);
            bathingsoap+=parseInt(theData[4]);
            shampoo+=parseInt(theData[5]);
            moisturizer+=parseInt(theData[6]);
        }
    });
    console.log(facecream,facewash,toothpaste,bathingsoap,shampoo,moisturizer);
}

```



```

        total =
facecream+facewash+toothpaste+bathingsoap+shampoo+moisturizer;
        //console.log(total);
        facecream = facecream*100/total;
        facewash = facewash*100/total;
        toothpaste = toothpaste*100/total;
        bathingsoap = bathingsoap*100/total;
        shampoo = shampoo*100/total;
        moisturizer = moisturizer*100/total;

//console.log(facecream,facewash,toothpaste,bathingsoap,shampoo,moisturizer)
;

        CanvasJS.addColorSet("colors",
[//colorSet Array
"#2F4F4F",
"#008080",
"#2E8B57",
"#3CB371",
"#90EE90",
"#5d9e9e"
]);

        var chart = new CanvasJS.Chart("chartContainer", {
            colorSet: "colors",
            theme: "light2",
            exportEnabled: false,
            animationEnabled: true,
            title: {
                text: "Yearly sales of Products"
            },
            legend:{
                cursor: "pointer"
            },
            subtitles: [{
                text: "Single Pie chart",
                fontSize: 16
            }],
            data: [
                {
                    type: "pie",
                    showInLegend: true,
                    indexLabelFontSize: 18,
                    radius: 180,
                    indexLabel: "{name} - {y}",
                    yValueFormatString: "###0.0\\\"%\\\"\"",
                    dataPoints: [

```

```

        { y: facecream, name: "Facecream", exploded: true },
        { y: facewash, name: "Facewash" },
        { y: toothpaste, name: "Toothpaste" },
        { y: bathingsoap, name: "Bathingsoap" },
        { y: shampoo, name: "Shampoo" },
        { y: moisturizer, name: "Moisturizer" }
      ]
    }
  ]
});
chart.render();
});
}
</script>
</html>

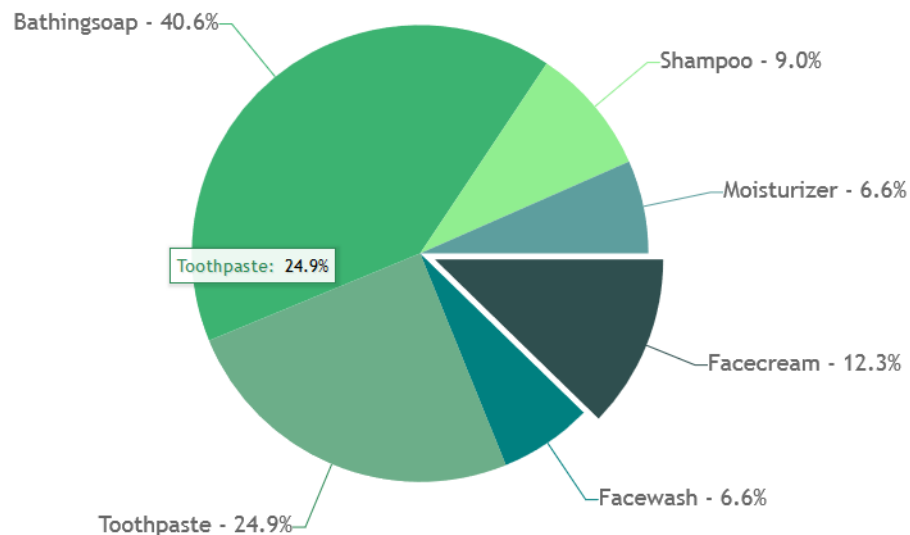
```



Showing the data as a Single Pie chart

## Yearly sales of Products

Single Pie chart



CanvasJS Trial Version ● Facecream ● Facewash ● Toothpaste ● Bathingsoap ● Shampoo ● Moisturizer CanvasJS.com

Fig6b1 – Single Pie chart

## Multiple Pie chart

```

<!DOCTYPE HTML>
<html>
<head>
    <title>Multiple Pie chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Showing the data as a Multiple Pie
chart"/>
    <meta name="keywords" content="Multiple,Pie,Chart,Multiple Pie
Chart,Pie"/>
    <script src="jquery-3.5.1.min.js"></script>
    <script src="canvasjs.min.js"></script>
</head>
<body>
    <h2 style="text-align:center">Showing the data as a Multiple Pie
chart</h2>
    <div id="chartContainer1" style="height:600px;width:45%;display: inline-
block;"></div>
    <div id="chartContainer2" style="height:600px;width:45%;display: inline-
block;"></div>
</body>
<script type="text/javascript">

window.onload = function () {
var toothpaste=[],bathingsoap=[];
$.get('data.txt', function(theData) {
    theData = theData.replace(/\r/g, '');
    theData = theData.replace(/\t/g, ' ');
    theData = theData.split('\n');
    totalRows = theData.length;
    data = theData;
    for(let i=1;i<totalRows;++i){
        theData = data[i].split(' ');

        toothpaste[i-1]=parseInt(theData[3])*100/69910;
        bathingsoap[i-1]=parseInt(theData[4])*100/114010;

    }

    CanvasJS.addColorSet("colors",
[//colorSet Array
"#264e70", "#95adbe", "#574f7d", "#503a65", "#3c2a4d", "#f9b4ab",

```

```

        "#fdebd3", "#e0f0ea", "#679186", "#2E8B57", "#3CB371", "#5d9e9e"
    });

    var chart = new CanvasJS.Chart("chartContainer1", {
        colorSet: "colors",
        theme: "light2",
        exportEnabled: false,
        animationEnabled: true,
        title: {
            text: "Monthly sales of Toothpaste",
            fontSize: 20
        },
        legend: {
            cursor: "pointer"
        },
        subtitles: [{
            text: "Pie chart",
            fontSize: 16
        }],
        data: [
            {
                type: "pie",
                showInLegend: false,
                indexLabelFontSize: 18,
                radius: 180,
                indexLabel: "{name} - {y}",
                yValueFormatString:
                    "###0.0\\ \"%\\\"",

                dataPoints: [
                    { y: toothpaste[0], name: "January", exploded: true },
                    { y: toothpaste[1], name: "February" },
                    { y: toothpaste[2], name: "March" },
                    { y: toothpaste[3], name: "April" },
                    { y: toothpaste[4], name: "May" },
                    { y: toothpaste[5], name: "June" },
                    { y: toothpaste[6], name: "July" },
                    { y: toothpaste[7], name: "August" },
                    { y: toothpaste[8], name: "September" },
                    { y: toothpaste[9], name: "October" },
                    { y: toothpaste[10], name: "November" },
                    { y: toothpaste[11], name: "December" },
                ]
            }
        ]
    });

```

```

        chart.render();
        CanvasJS.addColorSet("colors",
        [//colorSet Array
        "#122c91", "#2a6fdb", "#48d6d2", "#81e9e6", "#fefcbf", "#361d32",

        "#543c52", "#f55951", "#edd2cb", "#f1e8e6", "#2F4F4F", "#008080"
        ]);

        var chart = new CanvasJS.Chart("chartContainer2", {
            colorSet: "colors",
            theme: "light2",
            exportEnabled: false,
            animationEnabled: true,
            title: {
                text: "Monthly sales of Bathing
soap",

                fontSize: 20
            },
            legend: {
                cursor: "pointer"
            },
            subtitles: [{
                text: "Pie chart",
                fontSize: 16
            }],
            data: [
                {
                    type: "pie",
                    showInLegend: false,
                    indexLabelFontSize: 18,
                    radius: 180,
                    indexLabel: "{name} - {y}",
                    yValueFormatString:
                    "###0.0\"%\"",

                    dataPoints: [
                        { y: bathingsoap[0], name: "January", exploded: true },
                        { y: bathingsoap[1], name: "February" },
                        { y: bathingsoap[2], name: "March" },
                        { y: bathingsoap[3], name: "April" },
                        { y: bathingsoap[4], name: "May" },
                        { y: bathingsoap[5], name: "June" },
                        { y: bathingsoap[6], name: "July" },
                        { y: bathingsoap[7], name: "August" },
                        { y: bathingsoap[8], name: "September" },
                        { y: bathingsoap[9], name: "October" },

```

```

        { y: bathingsoap[10], name: "November" },
        { y: bathingsoap[11], name: "December" },
      ]

    }

  ]

});

chart.render();

});

}
</script>
</html>

```



### Showing the data as a Multiple Pie chart

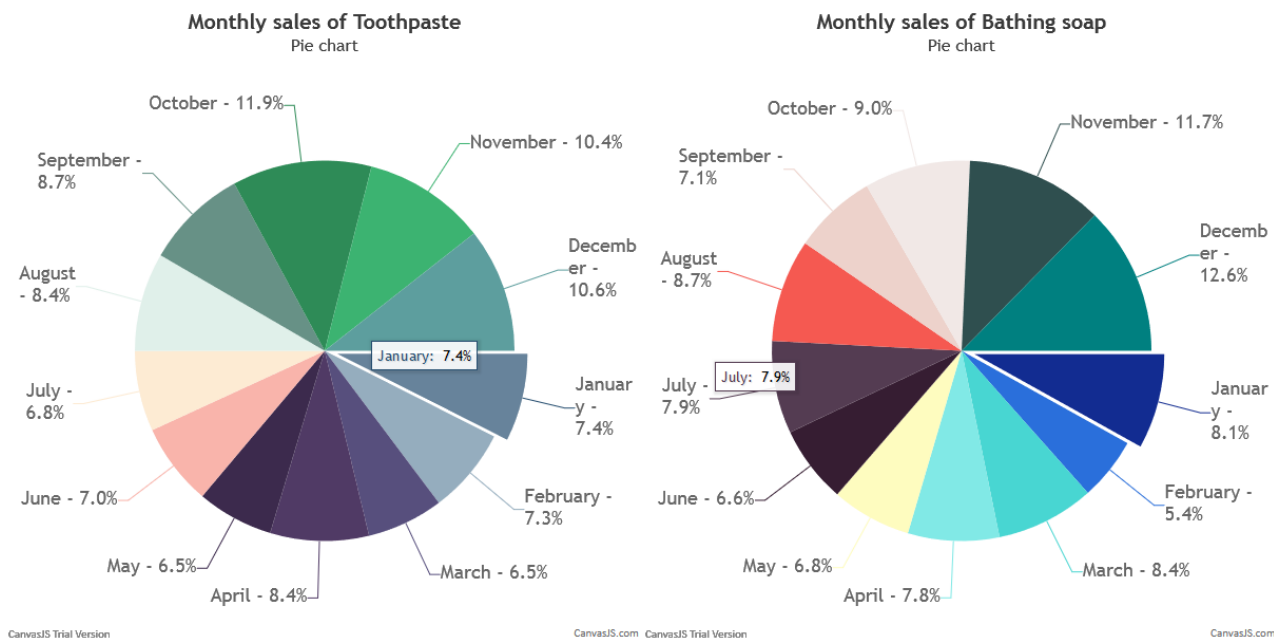


Fig6b2 – Multiple Pie chart

### 3. Showing the data as a Bar Chart (Simple and multiple)

#### Simple bar chart

```

<!DOCTYPE HTML>
<html>
<head>
  <title>Showing the data as a Simple Bar Chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Showing the sales data of
Facecream as a Simple Bar chart"/>
  <meta name="keywords" content="Single,Bar,Chart,Single Bar
Chart,Line"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="canvasjs.min.js"></script>
</head>
<body>
  <h2>Showing the data as a Simple Bar Chart</h2>
  <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
<script type="text/javascript">
window.onload = function () {
  var chart = new CanvasJS.Chart("chartContainer", {
    title:{
      text: "the sales data of Facecream"
    },
    axisX:{
      title: "Month",
    },
    axisY:{
      title:"Sales Unit",
    },
    theme: "dark1", // "light1", "light2", "dark1", "dark2"
    data: [
      {
        // Change type to "column", "doughnut", "line", "splineArea", etc.
        type: "column",
        dataPoints: [
          { label: "Jan", y: 2500 },
          { label: "Feb", y: 2630 },
          { label: "Mar", y: 2140 },
          { label: "Apr", y: 3400 },
        ]
      }
    ]
  });
  chart.render();
}

```

```

        { label: "May", y: 3600 },
        { label: "Jun", y: 2760 },
        { label: "Jul", y: 2980 },
        { label: "Aug", y: 3700 },
        { label: "Sep", y: 3540 },
        { label: "Oct", y: 1990 },
        { label: "Nov", y: 2340 },
        { label: "Dec", y: 2900 }
      ]
    }
  });
  chart.render();
}
</script>
</html>

```



### Showing the data as a Simple Bar Chart

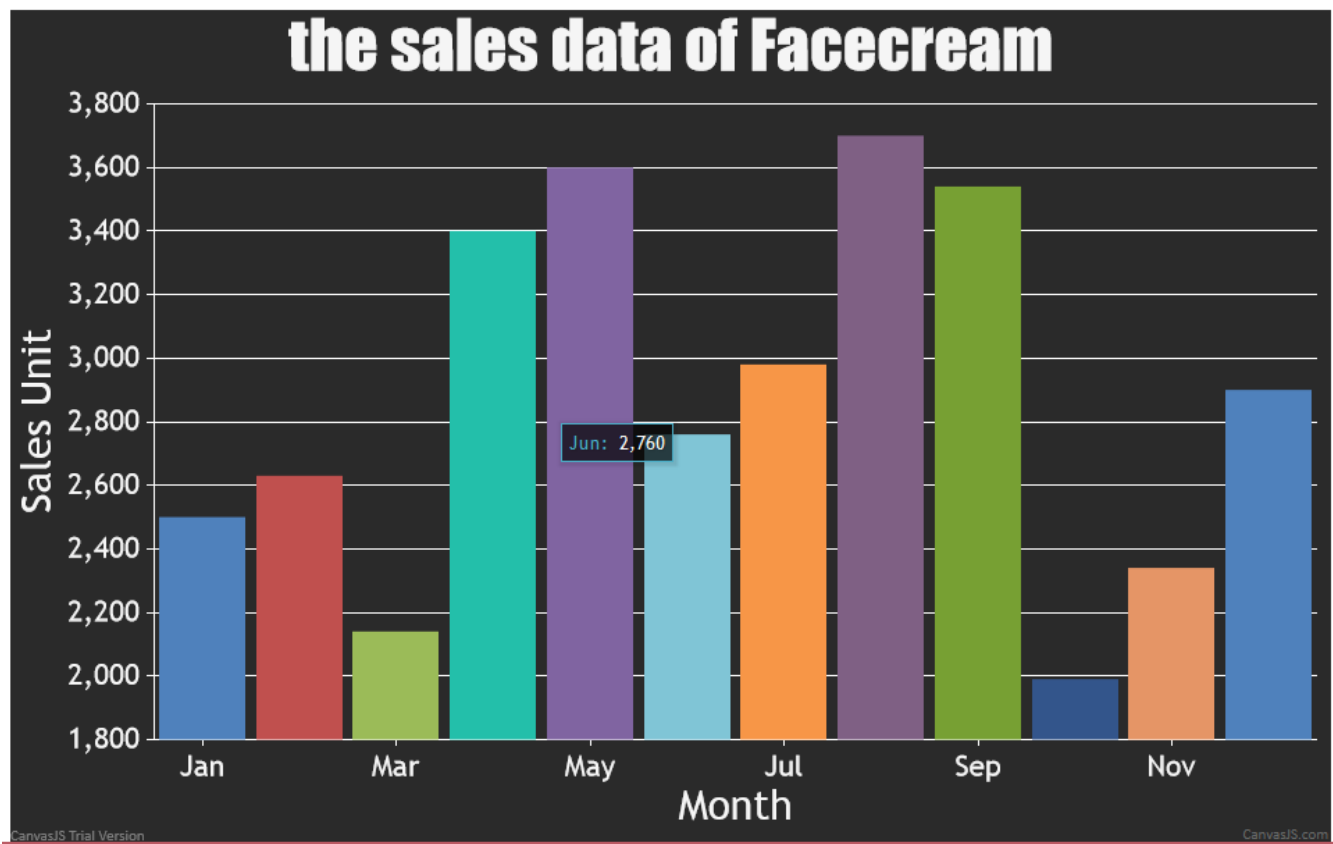


Fig6c1 – Simple bar chart



## Multiple Bar chart

```

<!DOCTYPE HTML>
<html>
<head>
    <title>Multi Bar Chart</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Showing the sales data of
Facecream as a Simple Bar chart"/>
    <meta name="keywords" content="Multiple,Bar,Chart,Multiple Bar
Chart,Line"/>
    <script src="jquery-3.5.1.min.js"></script>
    <script src="canvasjs.min.js"></script>
</head>
<body>
    <h2>Showing the data as a Multiple Bar Chart</h2>
    <div id="chartContainer" style="height: 600px; width: 100%;"></div>
</body>
<script type="text/javascript">
window.onload = function () {
    var chart = new CanvasJS.Chart("chartContainer", {
        animationEnabled: true,
        title:{
            text: "Salse of Facewash vs
Facecream"
        },
        theme: "dark1", // "light1", "light2",
"dark1", "dark2"

        axisY: {
            title: "Facewash Salse (Unit/Month)",
            titleFontColor: "#6d78ad",
            lineColor: "#6d78ad",
            labelFontColor: "#6d78ad",
            tickColor: "#6d78ad"
        },
        axisY2: {
            title: "Facecream Salse (Unit/Month)",
            titleFontColor: "#51cda0",
            lineColor: "#51cda0",
            labelFontColor: "#51cda0",
            tickColor: "#51cda0"
        },
    },

```

```

    tooltip: {
      shared: true
    },
    legend: {
      cursor: "pointer",
      itemclick: toggleDataSeries
    },
    data: [{
      type: "column",
      color: "#6d78ad",
      name: "Facewash Salse (Unit/Month)",
      legendText: "Facewash",
      showInLegend: true,
      dataPoints: [
        { label: "Jan", y: 1200 },
        { label: "Feb", y: 2100 },
        { label: "Mar", y: 3550 },
        { label: "Apr", y: 1870 },
        { label: "May", y: 1560 },
        { label: "Jun", y: 1890 },
        { label: "Jul", y: 1780 },
        { label: "Aug", y: 2860 },
        { label: "Sep", y: 2100 },
        { label: "Oct", y: 2300 },
        { label: "Nov", y: 2400 },
        { label: "Dec", y: 1800 }
      ]
    },
    {
      type: "column",
      color: "#51cda0",
      name: "Facecream Salse (Unit/Month)",
      legendText: "Facecream",
      axisYType: "secondary",
      showInLegend: true,
      dataPoints: [
        { label: "Jan", y: 2500 },
        { label: "Feb", y: 2630 },
        { label: "Mar", y: 2140 },
        { label: "Apr", y: 3400 },
        { label: "May", y: 3600 },
        { label: "Jun", y: 2760 },
        { label: "Jul", y: 2980 },
        { label: "Aug", y: 3700 },
        { label: "Sep", y: 3540 },

```

```

        { label: "Oct", y: 1990 },
        { label: "Nov", y: 2340 },
        { label: "Dec", y: 2900 }
      ]

    });

    chart.render();

    function toggleDataSeries(e) {
      if (typeof(e.dataSeries.visible) === "undefined" || e.dataSeries.visible) {
        e.dataSeries.visible = false;
      } else {e.dataSeries.visible = true;}
      chart.render();
    }
  }
</script>
</html>

```



### Showing the data as a Multiple Bar Chart

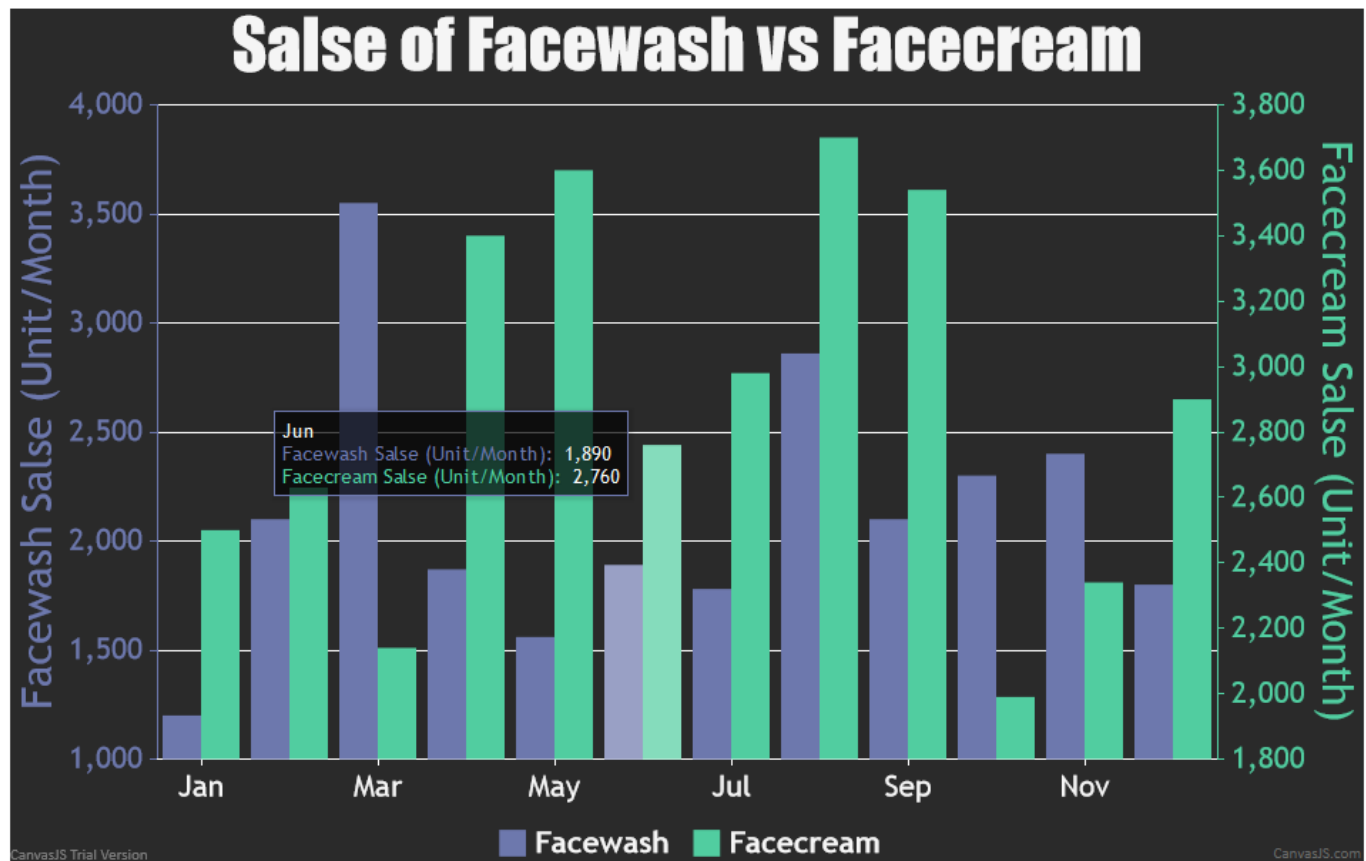


Fig6c2 – Multiple Bar chart

## Practical – 7

**AIM:** Develop Following Program Using HTML5 and Google Chats API and Map API.

### 1. Using Google Charts API Basics draw charts like a Bar chart

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Google API Bar chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Using Google Charts API Basics
draw charts like a Bar chart"/>
  <meta name="keywords" content="Google,API,Chart,Bar Chart,Bar"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20bar%20chart.html -->
<body>
  <h2>Using Google Charts API Basics draw charts like a Bar chart</h2>
  <div id="chart" style="width: 900px; height: 500px"></div>
</body>
  <script type="text/javascript">
google.charts.load('current', {'packages':['bar']});
google.charts.setOnLoadCallback(drawChart);

function drawChart() {
  var data = google.visualization.arrayToDataTable([
    ['Month', 'Toothpaste', 'Bathingsoap', 'Shampoo'],
      ['Jan', 5200, 9200, 1200],
    ['Feb', 5100, 6100, 2100],
    ['Mar', 4550, 9550, 3550],
    ['Apr', 5870, 8870, 1870],
      ['May', 4560, 7760, 1560],
      ['Jun', 4890, 7490, 1890],
      ['Jul', 4780, 8980, 1780],
      ['Aug', 5860, 9960, 2860],
      ['Sep', 6100, 8100, 2100],
      ['Oct', 8300, 10300, 2300],
```

```

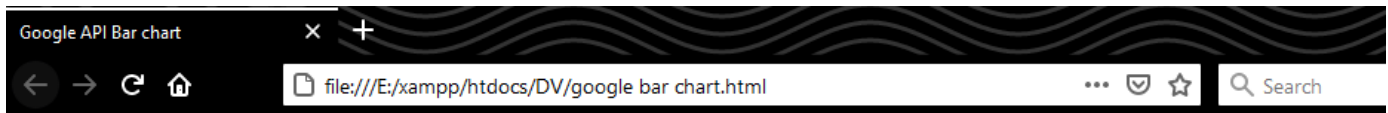
        ['Nov', 7300, 13300, 2400],
        ['Dec', 7400, 14400, 1800],
    ]);

    var options = {
        chart: {
            title: 'Company Sales',
            subtitle: 'Toothpaste ,Bathingsoap and Shampoo Monthly',
        },
        chartArea:{
            backgroundColor:'#2a2a2a'
        },
        backgroundColor: '#2a2a2a',
        colors: ['#9bbb58','#4f81bc','#c0504e'],
        titleTextStyle:{color: 'white',bold: true,fontSize:
20,italic: false},
        hAxis:{
            title:'Months',
            titleTextStyle: {color: 'white',bold: false,italic:
false},
            textStyle:{color: 'white',bold: false,italic: false},
            baselineColor:'#51cda0',
            gridlines:{color: 'green',minSpacing: 20}
        },
        vAxis:{
            title: 'Sales Unit',
            titleTextStyle: {color: 'white',bold: false,italic:
false},
            baselineColor:'white',
            gridlines:{color: 'gray'},
            textStyle:{color: 'white',bold: false,italic: false}
        },
        animation:{duration:1000,startup:true,easing:'linear'},
        tooltip:{textStyle: {color: '#871b47'}, showColorCode:
true},
        legend: { position: 'bottom',textStyle: {color: 'white', fontSize:
16} }
    };

    var chart = new google.charts.Bar(document.getElementById('chart'));

    chart.draw(data, google.charts.Bar.convertOptions(options));
}
</script>
</html>

```



## Using Google Charts API Basics draw charts like a Bar chart

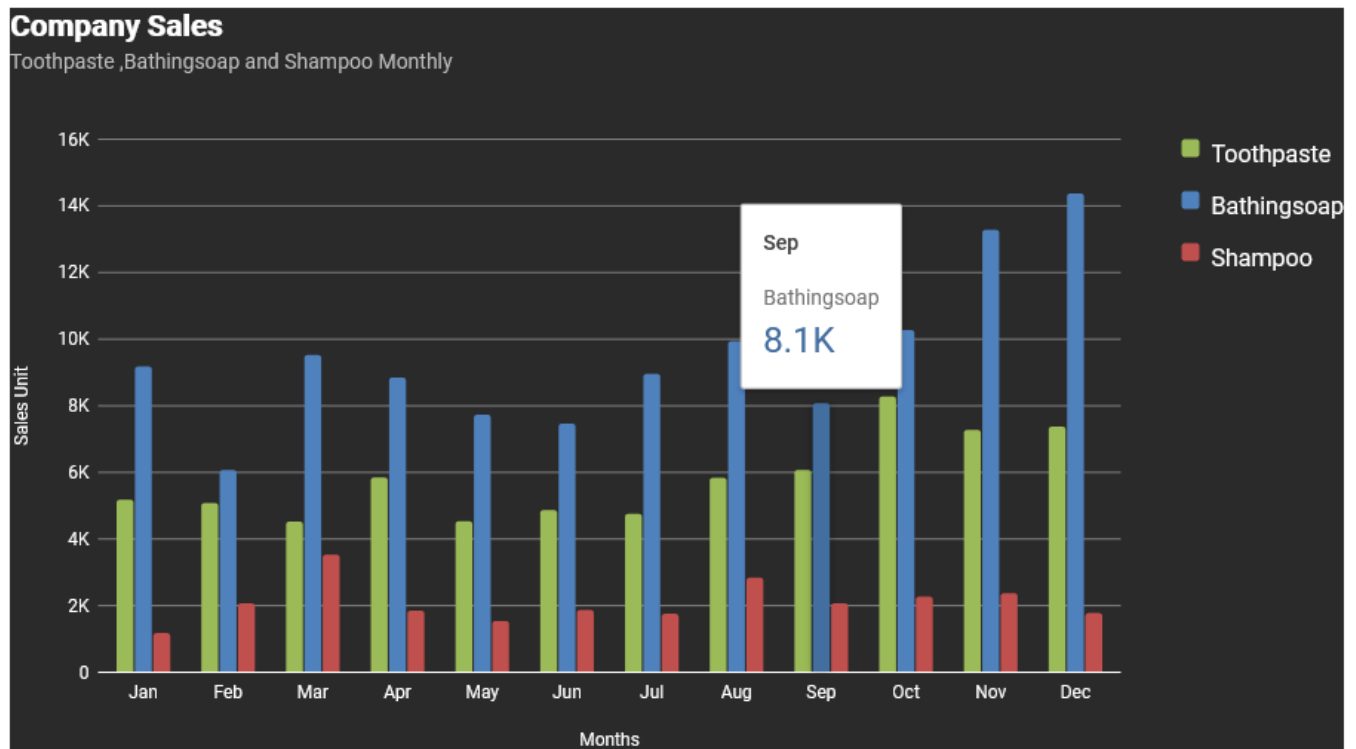


Fig7.a – Bar chart

## 2. Using Google Charts API Basics draw charts like a Line chart

```

<!DOCTYPE HTML>
<html>
<head>
  <title>Google API Line chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Using Google Charts API Basics
draw charts like a Line chart"/>
  <meta name="keywords" content="Google,API,Chart,Line Chart,Line"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20line%20chart.html -->
<body>
  <h2>Using Google Charts API Basics draw charts like a Line
chart</h2>
  <div id="chart" style="width: 900px; height: 500px"></div>
</body>

<script type="text/javascript">
google.charts.load('current', {'packages':['corechart']});
google.charts.setOnLoadCallback(drawChart);

function drawChart() {
  var data = google.visualization.arrayToDataTable([
    ['Month', 'Shampoo', 'Moisturizer'],
    ['Jan', 1500 , 2110],
    ['Feb', 1200 , 1833],
    ['Mar', 1340 , 2247],
    ['Apr', 1130 , 2227],
    ['May', 1740 , 2096],
    ['Jun', 1555 , 2014],
    ['Jul', 1120 , 2955],
    ['Aug', 1400 , 3614],
    ['Sep', 1780 , 2340],
    ['Oct', 1890 , 2667],
    ['Nov', 2100 , 4128],
    ['Dec', 1760 , 3002],
  ]);

  var options = {

```

```

        title: 'Monthly Sales Data of Shampoo and Moisturizer',
        curveType: 'none',
        backgroundColor: '#2a2a2a',
        colors: ['#51cda0', '#6d78ad'],
        titleTextStyle:{color: 'white',bold: true,fontSize:
20,italic: false},
        hAxis:{
            title:'Months',
            titleTextStyle: {color: 'white',bold: false,italic:
false},
            textStyle:{color: 'white',bold: false,italic: false},
            baselineColor:'#51cda0',
            gridlines:{color: 'green',minSpacing: 20}
        },
        vAxis:{
            title: 'Sales Unit',
            titleTextStyle: {color: 'white',bold: false,italic:
false},
            baselineColor:'white',
            gridlines:{color: 'gray'},
            textStyle:{color: 'white',bold: false,italic: false}
        },
        animation:{duration:1000,startup:true,easing:'linear'},
        tooltip:{textStyle: {color: '#871b47'}, showColorCode:
true},
        legend: { position: 'bottom',textStyle: {color: 'white', fontSize:
16} }
    };

    var chart = new
google.visualization.LineChart(document.getElementById('chart'));

    chart.draw(data, options);
}

</script>
</html>

```





### Using Google Charts API Basics draw charts like a Line chart

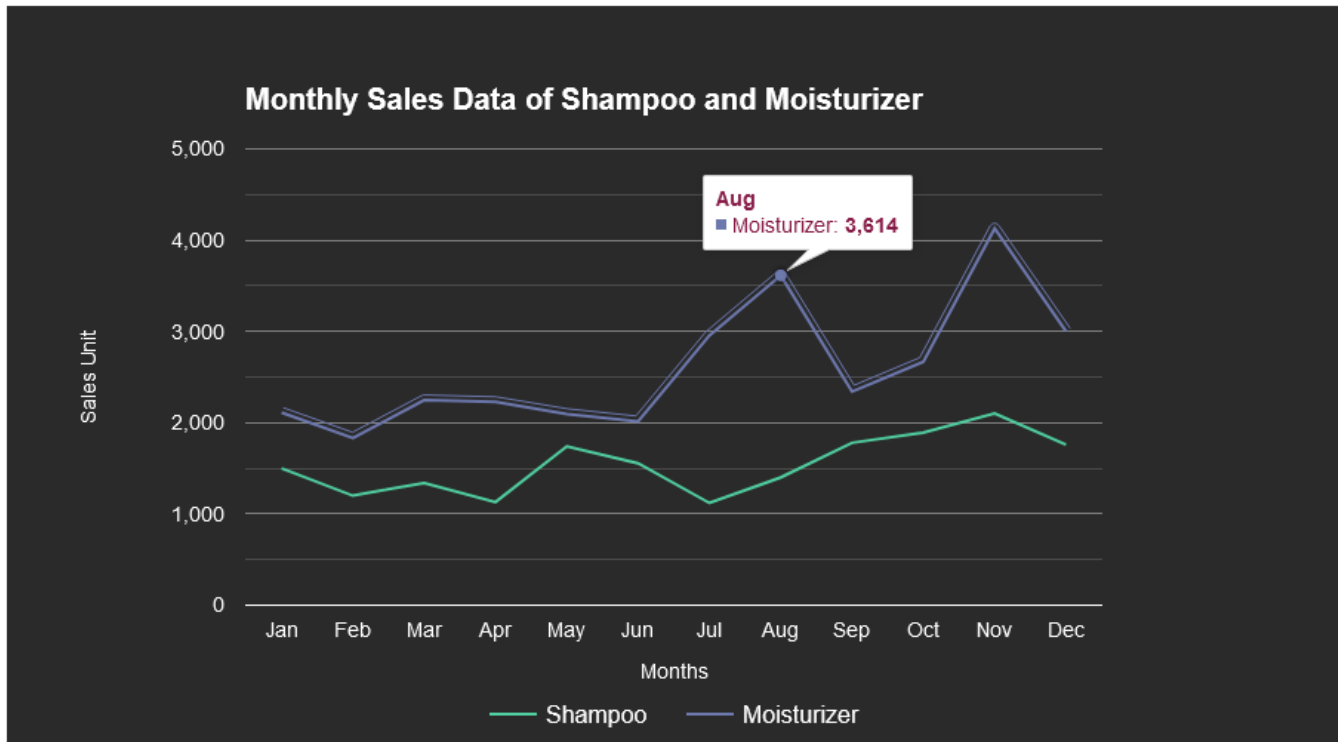


Fig7.b – Line Chart

### 3. Using Google Charts API Basics draw Pie Chart.

```

<!DOCTYPE HTML>
<html>
<head>
  <title>Google API Pie chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Using Google Charts API Basics
draw charts like a Pie chart"/>
  <meta name="keywords" content="Google,API,Chart,Pie Chart,Pie"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20pie%20chart.html -->
<body>
  <h2>Using Google Charts API Basics draw charts like a Pie chart</h2>
  <div id="chart" style="width: 900px; height: 500px"></div>
</body>
  <script type="text/javascript">
    google.charts.load("current", {packages:["corechart"]});
    google.charts.setOnLoadCallback(drawChart);
    function drawChart() {
      var data = google.visualization.arrayToDataTable([
        ['Products', 'Sales'],
        ['Bathingsoap', 114010],
        ['Shampoo', 25410],
        ['Moisturizer', 18515],
        ['Facecream', 34480],
        ['Facewash', 18515],
        ['Toothpaste', 69910]
      ]);
      var options = {
        title: 'Yearly sales of Products',
        titleTextStyle:{color: 'white',bold: true,fontSize:
20,italic: false},
        is3D: true,
        tooltip:{textStyle: {color: '#871b47'}, showColorCode:
true},
        backgroundColor: '#2a2a2a',
        legend: { position: 'right',textStyle: {color: 'white',
fontSize: 16} }

```

```
};  
  
var chart = new  
google.visualization.PieChart(document.getElementById('chart'));  
chart.draw(data, options);  
}  
</script>  
</html>
```



### Using Google Charts API Basics draw charts like a Pie chart

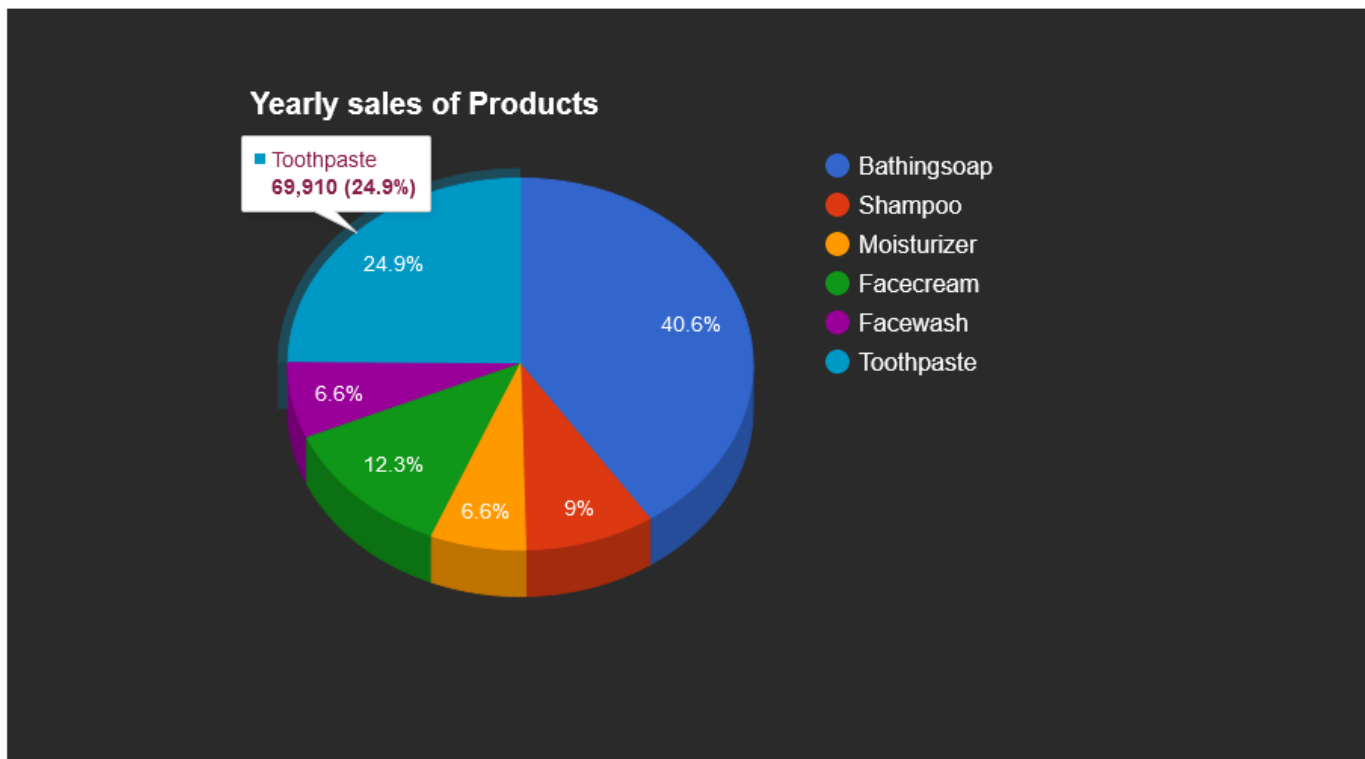


Fig7.c – Pie Chart

## Practical – 8

**AIM:** Develop Following Program Using HTML5 and Google Charts API and Map API.

### 1. Using Google Charts API Basics draw Donut Chart.

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Google API Donut chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Using Google Charts API Basics
draw charts like a Donut chart"/>
  <meta name="keywords" content="Google,API,Chart,Donut Chart,Donut"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20donut%20chart.html -->
<body>
  <h2>Using Google Charts API Basics draw charts like a Donut
chart</h2>
  <div id="chart" style="width: 900px; height: 500px"></div>
</body>
<script type="text/javascript">
  google.charts.load("current", {packages:["corechart"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var data = google.visualization.arrayToDataTable([
      ['Products', 'Sales'],
      ['Bathingsoap', 114010],
      ['Shampoo', 25410],
      ['Moisturizer', 18515],
      ['Facecream', 34480],
      ['Facewash', 18515],
      ['Toothpaste', 69910]
    ]);
    var options = {
      title: 'Yearly sales of Products',
```

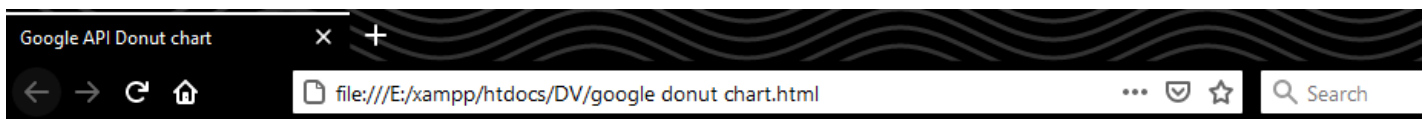
```

        titleTextStyle:{color: 'white',bold: true,fontSize:
20,italic: false},

//colors:['#102b2c','#503491','#3c3f17','#551e3a','#2b240f','#724628'],
        pieHole: 0.5,
        tooltip:{textStyle: {color: '#871b47'}, showColorCode:
true},

        backgroundColor: '#2a2a2a',
        legend: { position: 'right',textStyle: {color: 'white',
fontSize: 16} }
    };
    var chart = new
google.visualization.PieChart(document.getElementById('chart'));
    chart.draw(data, options);
}
</script>
</html>

```



### Using Google Charts API Basics draw charts like a Donut chart

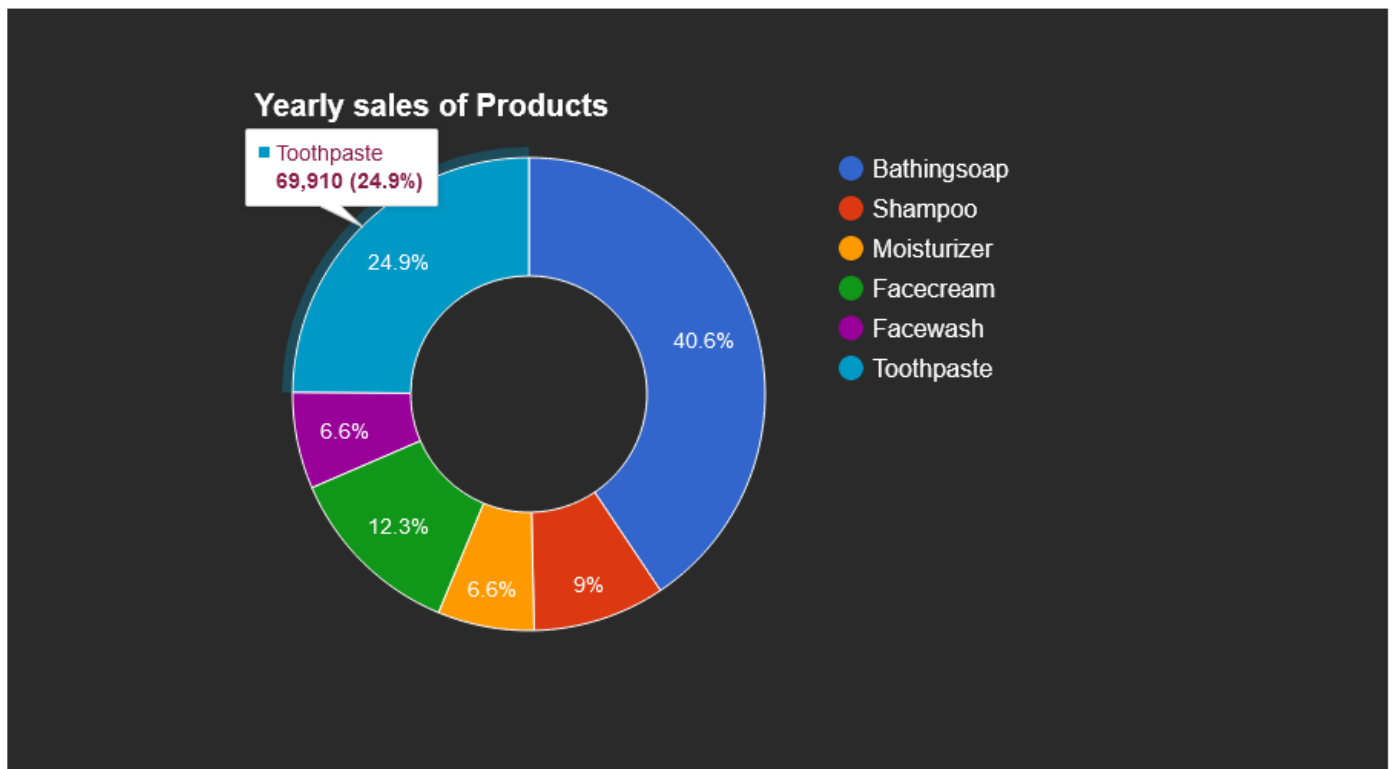


Fig8.a – Donut chart

## 2. Using Google Charts API Basics draw Candle Chart.

```

<!DOCTYPE HTML>
<html>
<head>
  <title>Google API CandleStick chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Using Google Charts API Basics
draw charts like a CandleStick chart"/>
  <meta name="keywords" content="Google,API,Chart,CandleStick
Chart,CandleStick"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20candle%20chart.html -->
<body>
  <h2>Using Google Charts API Basics draw charts like a CandleStick
chart</h2>
  <div id="chart" style="width: 900px; height: 500px"></div>
</body>
  <script type="text/javascript">
    google.charts.load("current", {packages:["corechart"]});
    google.charts.setOnLoadCallback(drawChart);
    function drawChart() {
      var data = google.visualization.arrayToDataTable([
        //(NSE Indices - Nifty 500)
        //data format low,open,close,high if open<close then
candle will be filled;else candle will be hollow
        ['Just Dial', 784, 820, 870, 885],
        ['Tata Chemical', 590, 614, 732, 743],
        ['Canara Bank', 680, 840, 706, 860],
        ['Torrent Power', 560, 600, 710, 743],
        ['IRCTC', 520, 560, 712, 743],
        ['Airtel', 630, 720, 670, 743],
        // https://www.investopedia.com/trading/candlestick-charting-what-
is-it/
        // Treat first row as data as well.
      ], true);

      var options = {
        title:'(NSE Indices - Nifty 500)',

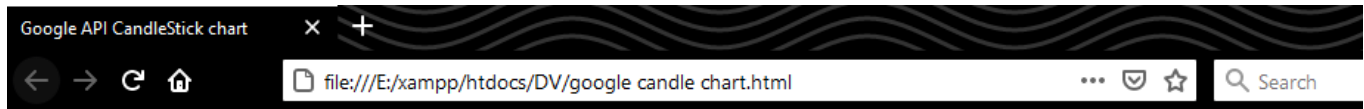
```

```
titleTextStyle:{color: 'white',bold: true,fontSize: 20,italic:
false},
colors:['#6d78ad'],
backgroundColor: '#2a2a2a',
animation:{ duration:1000,easing:'linear',startup:true},
tooltip:{textStyle: {color: '#871b47'}, showColorCode: true},
hAxis:{
  title:'Company Name',
  titleTextStyle: {color: 'white',bold: false,italic: false},
  textStyle:{color: 'white',bold: false,italic: false},
  baselineColor:'#51cda0',
  gridlines:{color: 'green',minSpacing: 20}
},
vAxis:{
  title: 'Stock stat',
  titleTextStyle: {color: 'white',bold: false,italic: false},
  baselineColor:'white',
  gridlines:{color: 'gray'},
  textStyle:{color: 'white',bold: false,italic: false}
},
candlestick:{hollowIsRising:true,

fallingColor:{fill:'white',stroke:'#6d78ad',strokeWidth:2},

risingColor:{fill:'#6d78ad',stroke:'#6d78ad',strokeWidth:2}},
  legend: 'none'
};

var chart = new
google.visualization.CandlestickChart(document.getElementById('chart'));
chart.draw(data, options);
}
</script>
</html>
```



### Using Google Charts API Basics draw charts like a CandleStick chart

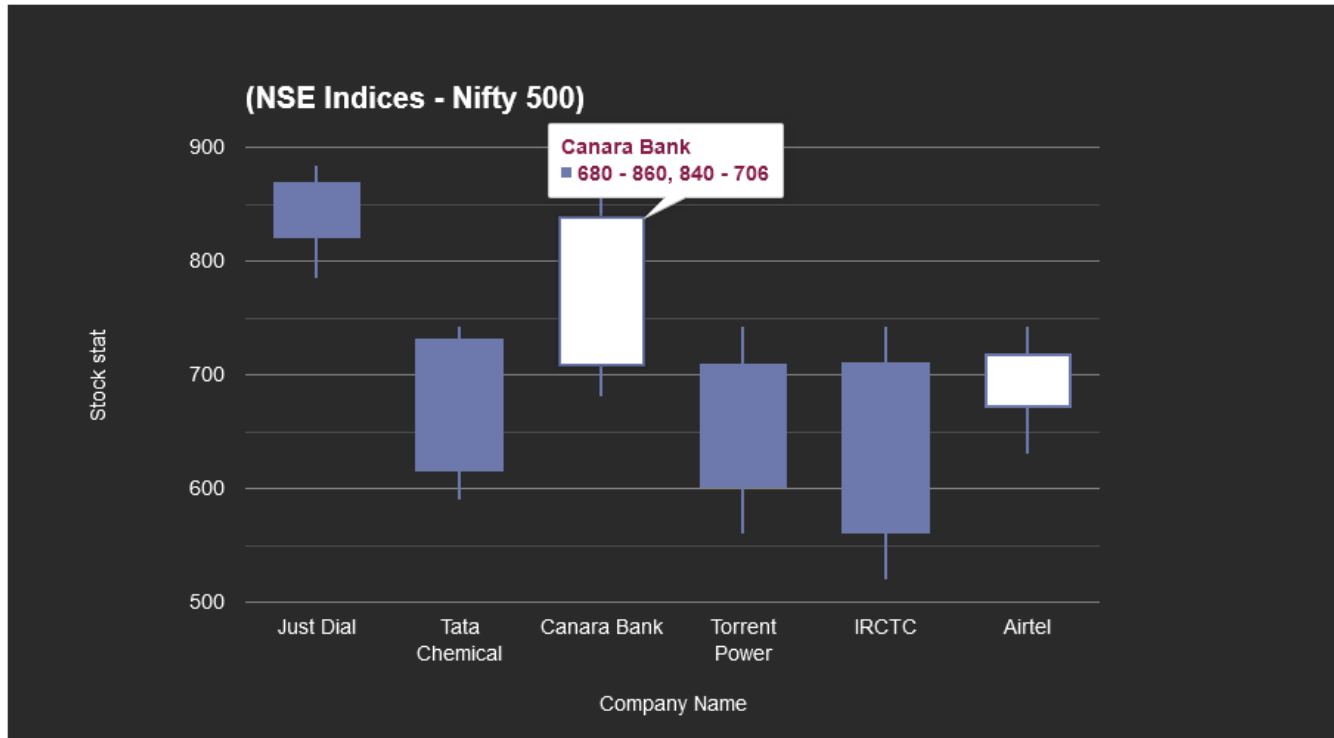


Fig8.b – Candlestick chart



### 3. Using Google Charts API Basics draw other types of Chart.

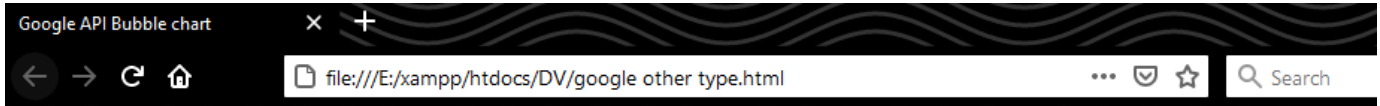
```

<!DOCTYPE HTML>
<html>
<head>
  <title>Google API Bubble chart</title>
  <meta charset="utf-8">
  <meta name="author" content="SidPro"/>
  <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
  <meta name="description" content="Using Google Charts API Basics
draw chart(other type) bubble chart"/>
  <meta name="keywords" content="Google,API,Chart,Bubble
Chart,Bubble"/>
  <script src="jquery-3.5.1.min.js"></script>
  <script src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20other%20type.html -->
<body>
  <h2>Using Google Charts API Basics draw chart(other type) bubble
chart</h2>
  <div id="chart" style="width: 900px; height: 500px"></div>
</body>
<script type="text/javascript">
  google.charts.load('current', {'packages':['corechart']});
  google.charts.setOnLoadCallback(drawSeriesChart);
  function drawSeriesChart() {
    var data = google.visualization.arrayToDataTable([
      ['ID', 'Life Expectancy', 'Fertility Rate', 'Region',
'Population'],
      ['CAN', 80.66, 1.67, 'North America',
33739900],
      ['DEU', 79.84, 1.36, 'Europe',
81902307],
      ['DNK', 78.6, 1.84, 'Europe',
5523095],
      ['EGY', 72.73, 2.78, 'Middle East',
79716203],
      ['GBR', 80.05, 2, 'Europe',
61801570],
      ['IRN', 72.49, 1.7, 'Middle East',
73137148],
      ['IRQ', 68.09, 4.77, 'Middle East',
31090763],
    ]
  )
  
```

```

        ['ISR',      81.55,      2.96,      'Middle East',
7485600],
        ['RUS',      68.6,      1.54,      'Europe',
141850000],
        ['USA',      78.09,      2.05,      'North America',
307007000]
    });
    var options = {
        title: 'Correlation between life expectancy, fertility
rate ' +
            'and population of some world countries (2010)',
        curveType: 'none',
        backgroundColor: '#2a2a2a',
        colors: ['#51cda0','#6d78ad'],
        titleTextStyle:{color: 'white',bold: true,fontSize:
20,italic: false},
        hAxis:{
            title:'Life Expectancy',
            titleTextStyle: {color: 'white',bold: false,italic:
false},
            textStyle:{color: 'white',bold: false,italic: false},
            baselineColor:'#51cda0'
        },
        vAxis:{
            title: 'Fertility Rate',
            titleTextStyle: {color: 'white',bold: false,italic:
false},
            baselineColor:'white',
            gridlines:{color: 'gray'},
            textStyle:{color: 'white',bold: false,italic: false}
        },
        animation:{duration:1000,startup:true,easing:'linear'},
        tooltip:{textStyle: {color: '#871b47'}, showColorCode:
true},
        bubble: {textStyle: {color: 'white', fontSize: 11}},
        legend: {textStyle: {color: 'white', fontSize: 16} }
    };
    var chart = new
google.visualization.BubbleChart(document.getElementById('chart'));
    chart.draw(data, options);
}
</script>
</html>

```



### Using Google Charts API Basics draw chart(other type) bubble chart

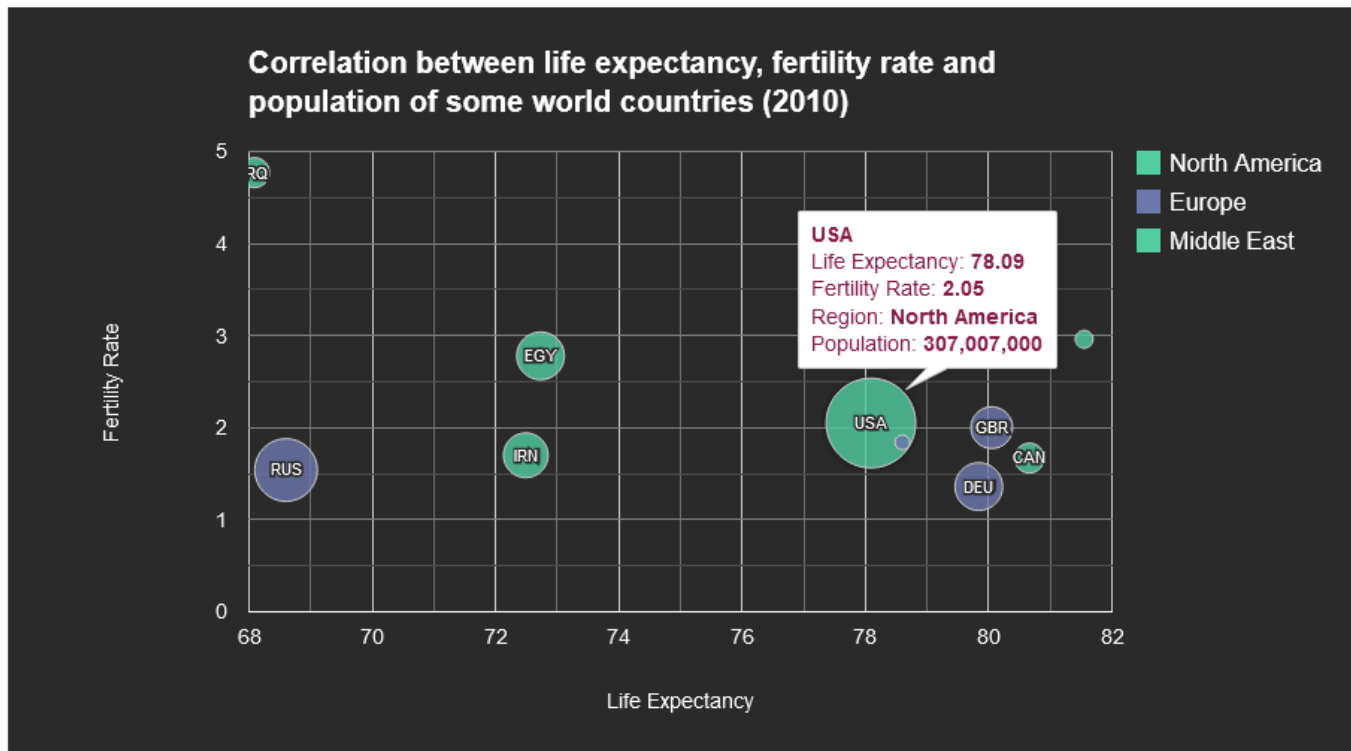


Fig8.c – Bubble chart

#### 4. Using Google API read JSON file and create Google Map.

```

<!DOCTYPE HTML>
<html>
<head>
    <title>Google API JSON to Google Map</title>
    <meta charset="utf-8">
    <meta name="author" content="SidPro"/>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0"/>
    <meta name="description" content="Using Google API read JSON file
and create Google Map."/>
    <meta name="keywords" content="Google,API,Geo Chart,Covid19 Geo
Chart,Covid19"/>
    <script src="jquery-3.5.1.min.js"></script>
    <script type="text/javascript" src="loader.js"></script>
</head>
<!-- Live link: https://sidpro-hash.github.io/HTML-
Canvas/Data%20visulization%20JavaScript/google%20map%20json.html -->
<body>
    <h2>Read JSON file and create Google Map of Total COVID-19 cases in
country [Date:01/04/2021]</h2>
    <div id="chart" style="width: 900px; height: 500px"></div>
</body>
<script>
    // Visualization API with the 'corechart' package.
    google.charts.load('current', {
        'packages':['geochart'],
        // Note: you will need to get a mapsApiKey for your project.
        // See:
        https://developers.google.com/chart/interactive/docs/basic_load_libs#load-
settings
        'mapsApiKey': 'AIzaSyD-9tSrke72PouQMnMX-a7eZSW0jkFMBWY'
    });

    google.charts.setOnLoadCallback(drawChart);

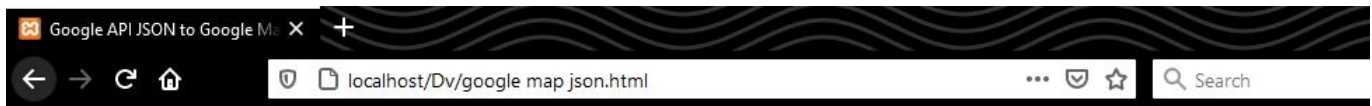
    function drawChart() {
        $.ajax({
            url: "Covid-19.json",
            dataType: "json",
            type: "GET",
            contentType: "application/json; charset=utf-8",
            success: function (data) {
                var arr = [['Country', 'Total COVID-19 Cases']];

```

```
$.each(data, function (k, v) {
    arr.push([v.city, v.n]);
});

var options = {
    colorAxis: {colors:
['#9bbb58', '#4f81bc', '#c0504e']},
    backgroundColor: '#81d4fa',
    defaultColor: '#f5f5f5',
    tooltip:{textStyle: {color:
'#871b47'}, showColorCode: true}
    };

var Data = google.visualization.arrayToDataTable(arr);
var chart = new
google.visualization.GeoChart(document.getElementById('chart'));
chart.draw(Data, options);
},
error: function (XMLHttpRequest, textStatus, errorThrown) {
    alert('File Error or Cross-origin police Error');
}
});
}
</script>
</html>
```



**read JSON file and create Google Map of Total COVID-19 cases in country [Date:01/04/2021]**

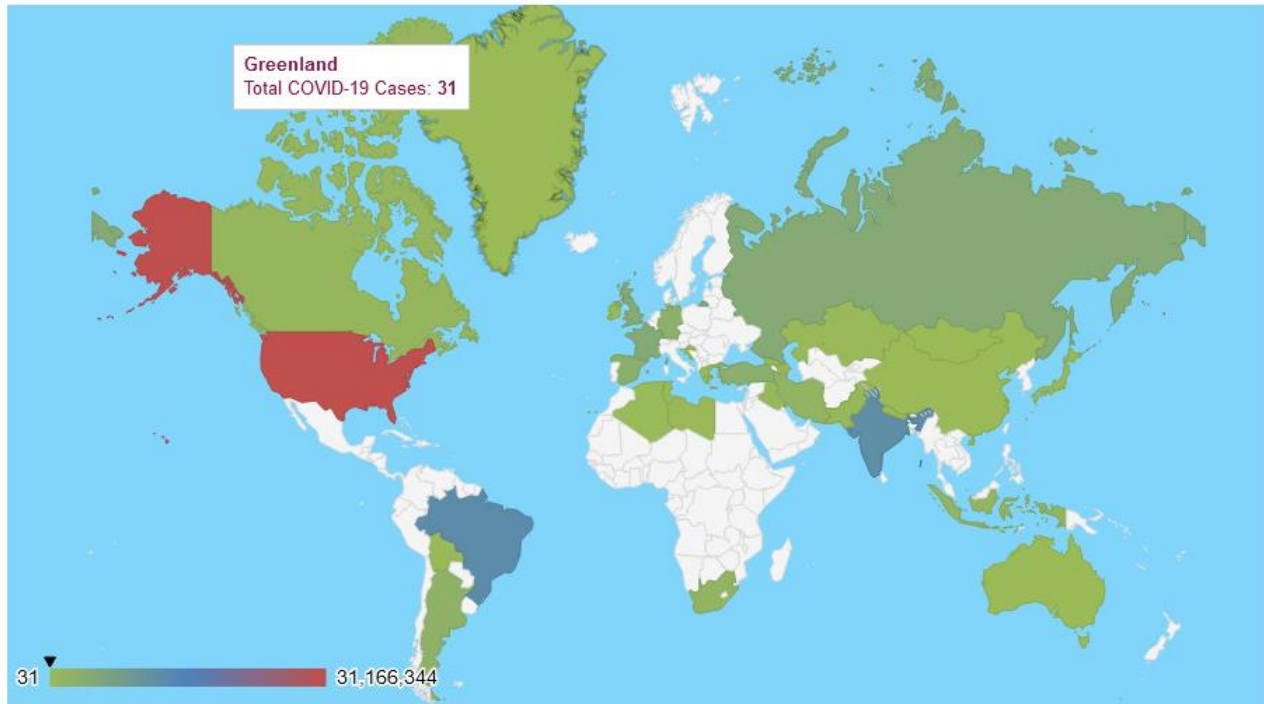


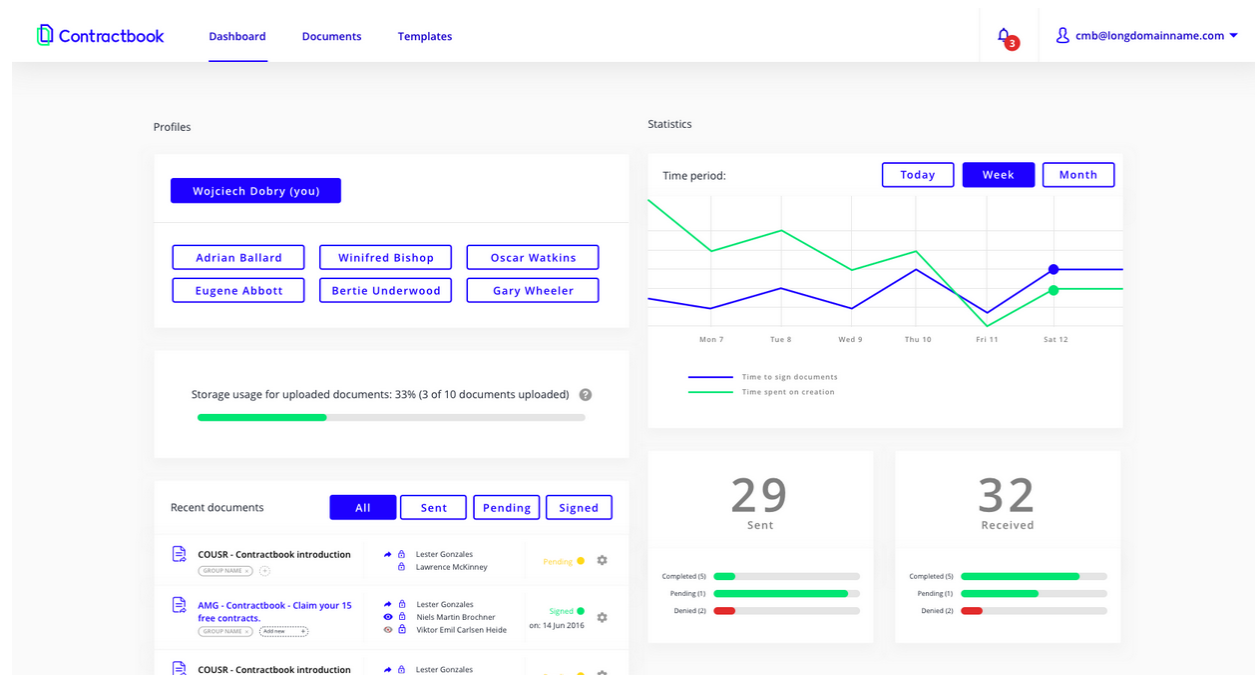
Fig8.d – Covid-19 cases

## Practical – 9

### AIM: Case Study for Different Information Dashboard.

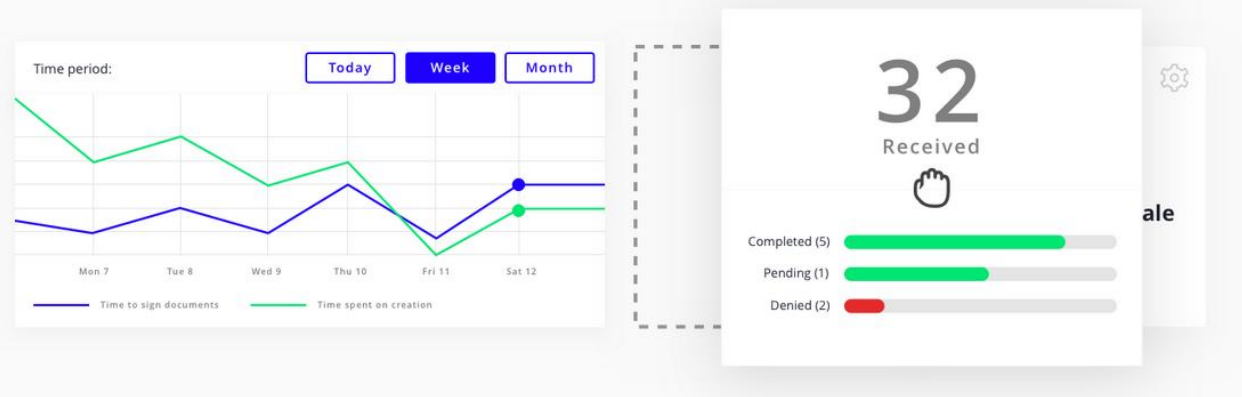
Dashboards are a unique and powerful way to present data-based intelligence using data visualization techniques that display relevant, actionable data as well as track stats and key performance indicators (KPIs). Dashboards should present this data in a quick, easy-to-scan format with the most relevant information understandable at a glance.

A dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance.



Great dashboards are clear, intuitive, and customizable.

- They communicate information quickly.
- They display information clearly and efficiently.
- They show trends and changes in data over time
- They are easily customizable.
- The most important widgets and data components are effectively presented in a limited space.



Great dashboards provide everything one click away.

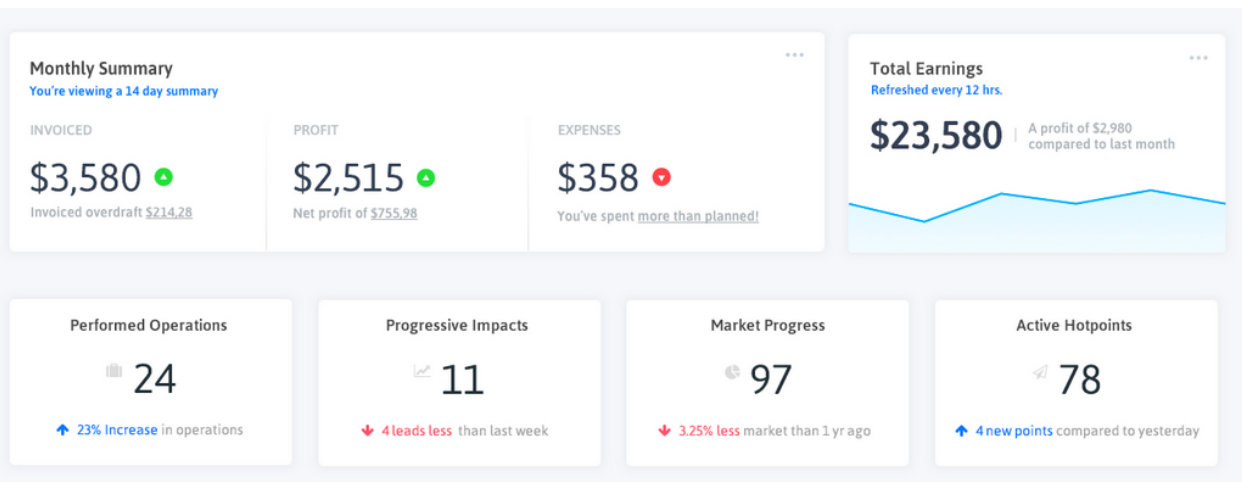
- All essential information is immediately accessible.
- Data is prioritized.
- Information is displayed clearly in a visual hierarchy on one screen.
- The design provides a coherent overview that includes sparse, clear initial data with additional opportunities to drill down for more.
- Elements (chart, table, form) are displayed in a minimized view with the ability to bring up more details in a modal window or go to a page with more detail.
- The design improves usability with filters allowing users to customize how data is displayed and filters content using labels, categories, and KPIs.

## Reduced Complexity Provides Clarity

In a world overwhelmed with data, providing clear information is one of the most difficult things to accomplish. Presenting only the most relevant data on dashboards is essential—the more information we display, the harder it is for users to find what they need.

Effective dashboard design decisions should be guided by:

- the project goals
- the nature of the data
- the needs of users





When reading a visualization (or any other kind of communication), your reader has a limited amount of brainpower to dedicate to the problem. Some of this brainpower will be dedicated to decoding the visualization; any brainpower that is left may then be used to understand the message (if the reader hasn't yet given up in frustration).

The core objective of a dashboard is to make complex information accessible and easy to digest. Therefore, the interface presenting the data should be clean and straightforward in order to minimize users' cognitive load and time spent searching.

The information architecture should present the essential data first while allowing access to supporting or secondary metrics. A progressive drill-down system should be designed that starts with a general overview and then goes into more detail—it facilitates data prioritization, and creates clarity.

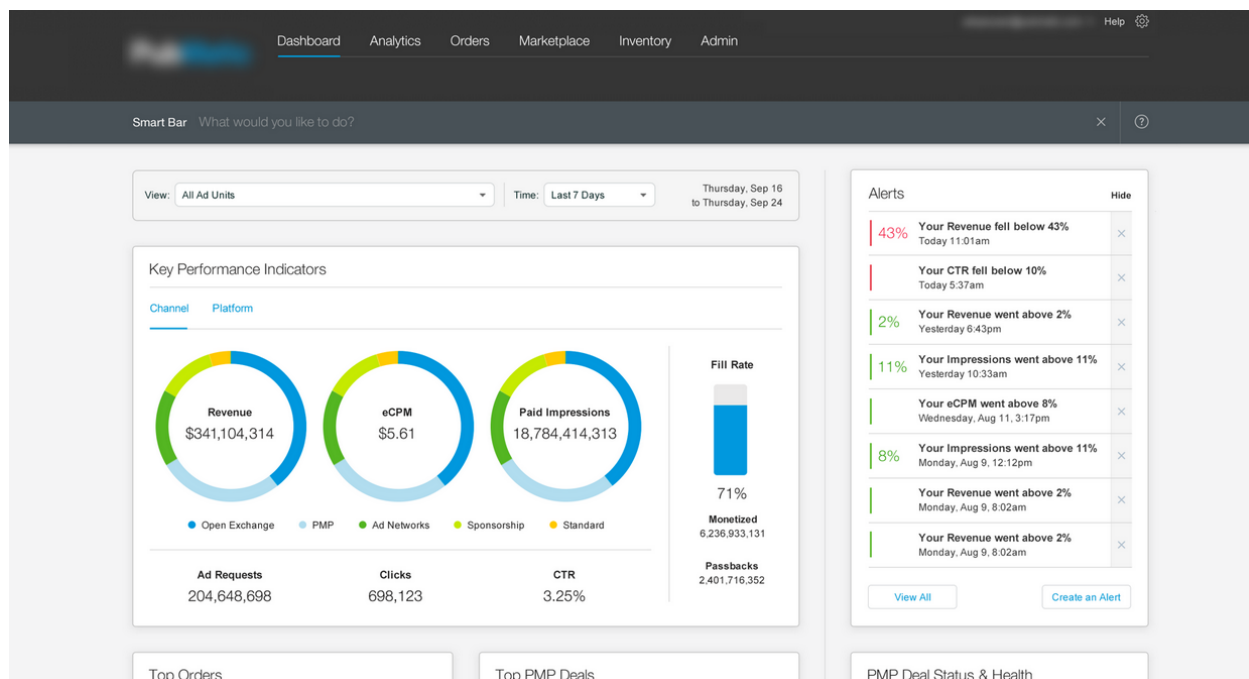
## Determining Dashboard Goals and Displaying Appropriate Data

When designing dashboards, successful dashboard designers start with a well-defined set of goals focusing on the problem to solve and the key, actionable insights people need to take away from the data.

Good design goals promote efficient and precise execution. Employing the **S.M.A.R.T framework** for goal setting puts the focus on **specific, measurable, actionable, realistic, and time-based objectives**.

A few key questions to ask when determining dashboard design goals:

- How many steps do users need to take to achieve a specific goal?
- Is the interface intuitive enough for the user to reach their goal on their own?
- What information does the user need to successfully achieve their goal?



Goal-centric design focuses on solutions to real problems and is the foundation for all great dashboard design. Start with a clear understanding of business objectives, consider user goals, and then convey the key information that needs to be communicated.

## Context in Dashboard Design

One of the biggest challenges of dashboard design is serving multiple personas. Once each user role is defined, it becomes critical to understand where their needs overlap and where they diverge.

Effective communication is the underlying principle of every successful dashboard design. Foreseeing potential scenarios in which users may find themselves will contribute to a better understanding of the user's circumstances.

Always keep users context in mind when designing-identify their technical knowledge, their familiarity with the system overall, their goals, and so on.



Be sure to ask the following questions when trying to determine user behavior and context:

- Does the design consider the direction the visitor is used to reading in?
- Does interaction with the dashboard require technical knowledge?
- Will users manage to accomplish most of the actions in just a few clicks?
- Does the design align with user context by creating drill-down menus; does it use suggestive iconography and colour palettes?

The color palette used in a dashboard's design should also be considered as a context. Many business-to-business SaaS product dashboards are designed in a dark-themed UI because they are used for several hours straight.

Dark-themed UIs can help reduce eye strain and support visual clarity within the interface.



## Progressive Disclosure in Dashboard Design

Progressive disclosure is a technique used to maintain a user's attention by reducing clutter. Creating a system of progressive disclosure assists in creating a user-centric environment, which helps prioritize user attention, avoid mistakes, and save time. It also allows users to focus on the key features that matter to them and not be forced to go through all of the features—including the ones they don't need or are not interested in.

Progressive disclosure is a dashboard design best practice that will also reduce error rates considerably; it will improve efficiency and help users improve their understanding of dashboards when a system is based on feature prioritization.

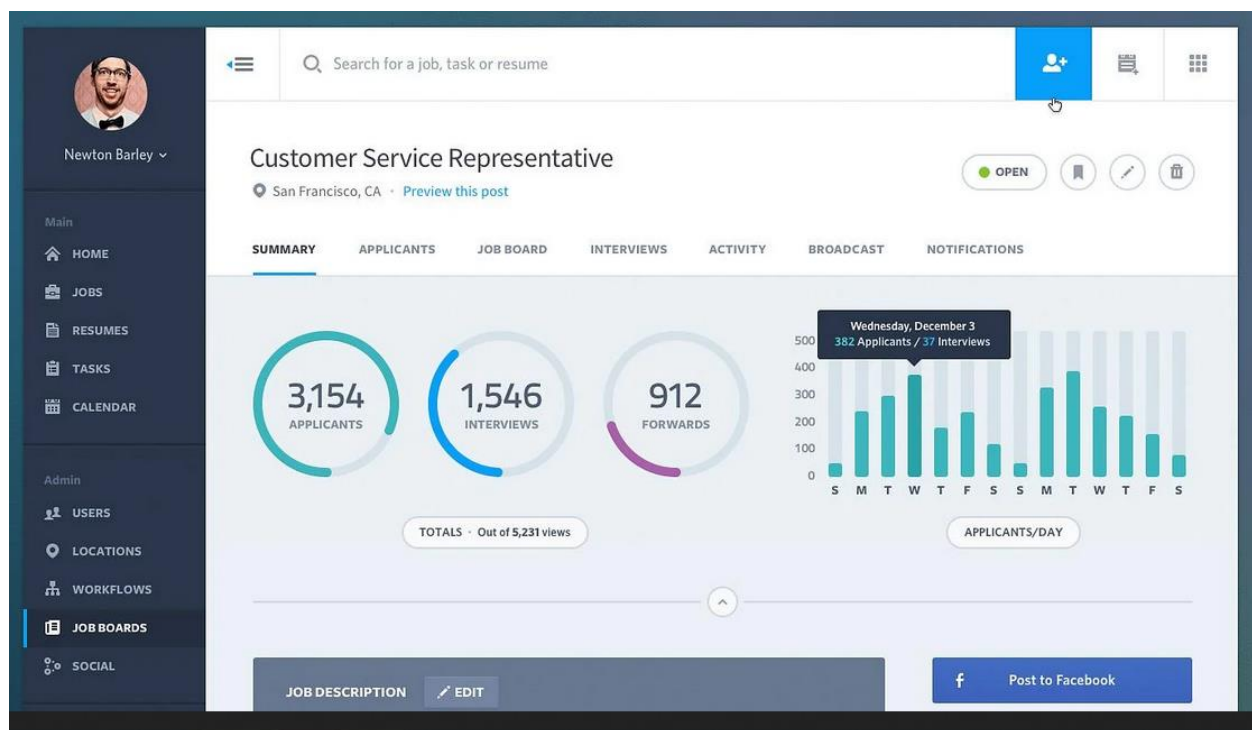
Advantages of using progressive disclosure in dashboards:

- Reducing user uncertainty and anxiety (by displaying signs of progress, the users are assured that everything is working as intended)
- Providing users with something to look at (partial display of data) rather than make them wait

- Providing users with a clear expectation of the upcoming steps, and creating an understanding of the way information is presented hierarchically

Potential issues with progressive disclosures and loading of data includes:

- Using indicators and disclosures in an inappropriate manner. Short loading times and useless steps will create distraction and work against usability principles.
- A lag in the retrieval of data without a clear indicator of progress can lead to user uncertainty and higher bounce rates.
- Using static progress indicators is a solution with little meaning, and does not offer enough information about progress, which can also lead to user uncertainty and higher bounce rates.



## Better Dashboard Design with User Research

User research helps create an environment in which users are presented with data that is relevant, clear, and concise. This helps them think about the content and the data they're looking for, rather than how to use and access it.

Some dashboards have to work or be effortlessly customizable to users of different roles looking at the same basic dashboard. User research is important because it helps determine the user's goals, mental models, environmental context, and pain points. These are factors that greatly influence the final dashboard design

## Summary

Dashboards are a powerful way to communicate data and other information, especially with a user-centered, goal-centric design that follows dashboard design best practices and proper data visualization. Although every dashboard is different and has its own goals, requirements, and limitations, following these fundamental principles will help in creating outstanding designs regardless of the specifics:

- First and foremost, empathize with your user types and understand their goals.
- Convey a clear story to users by making use of suggestive visuals, labels, progressive disclosure techniques, and animation.
- Make the complex things easy by applying user research techniques.
- Reveal data and information at the appropriate time, in a drill-down system.
- Use data visualization to express information in a meaningful way.

## Practical – 10

**AIM:** Build interconnected Dashboard Using Different Library for any one enterprise Solution.

**Team Members:**

180170107030 – Gabu Siddharth

180170107033 – Gohil Chandrarajsinh

180170107046 – Kava Chirag

180170107048 – Kosrekar Adarsh

**Live link:** <https://sidpro-hash.github.io/Dashboard/>

The GTU RESULT VISUALIZATION DASHBOARD is made using HTML, CSS, JAVASCRIPT, JQUERY and CANVAS.JS.

### GTU RESULT VISUALIZATION DASHBOARD

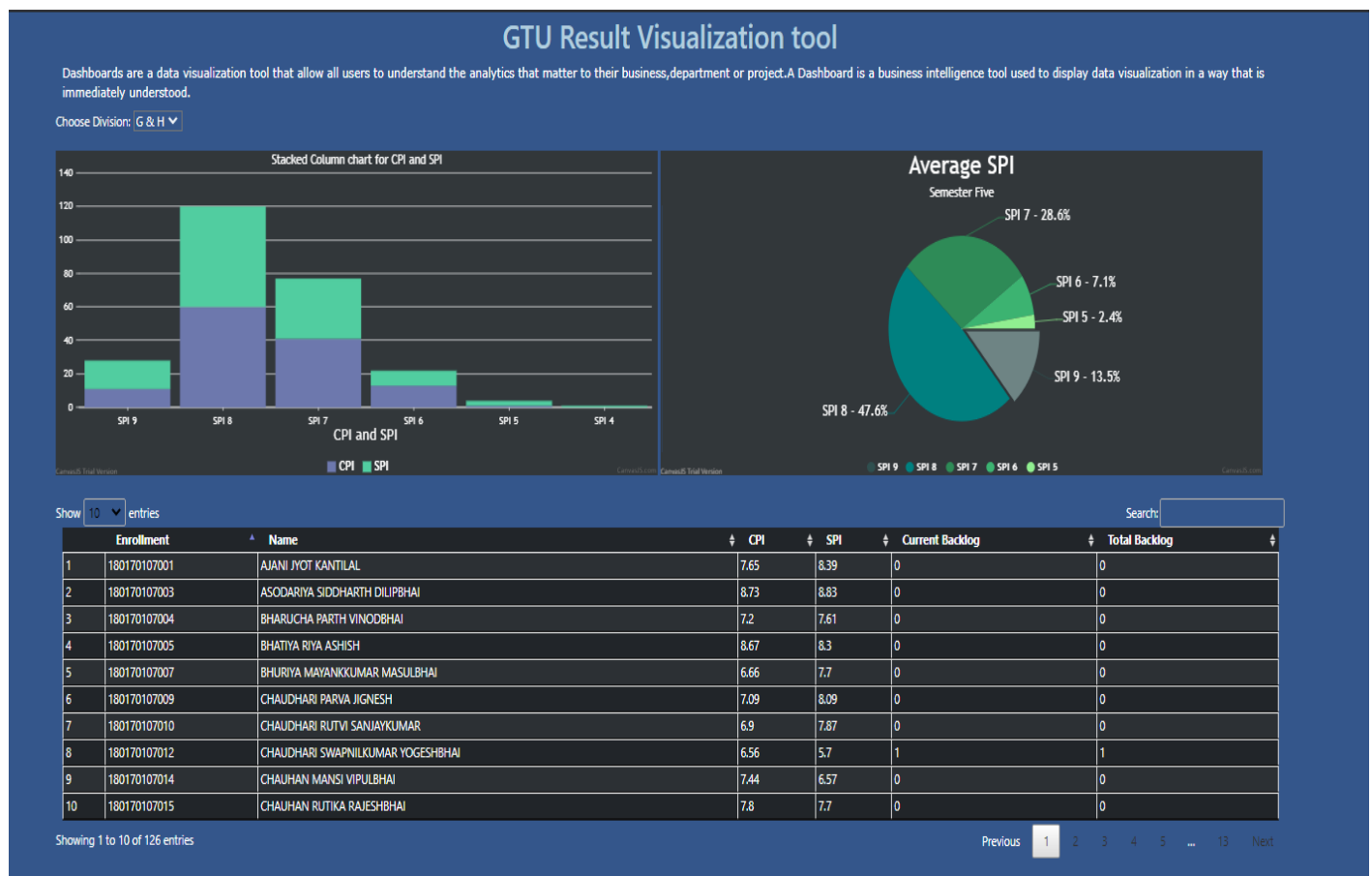


Fig 10.1

Above figure shows two different visualization of result of students of G and H division in

stacked column chart and pie chart.

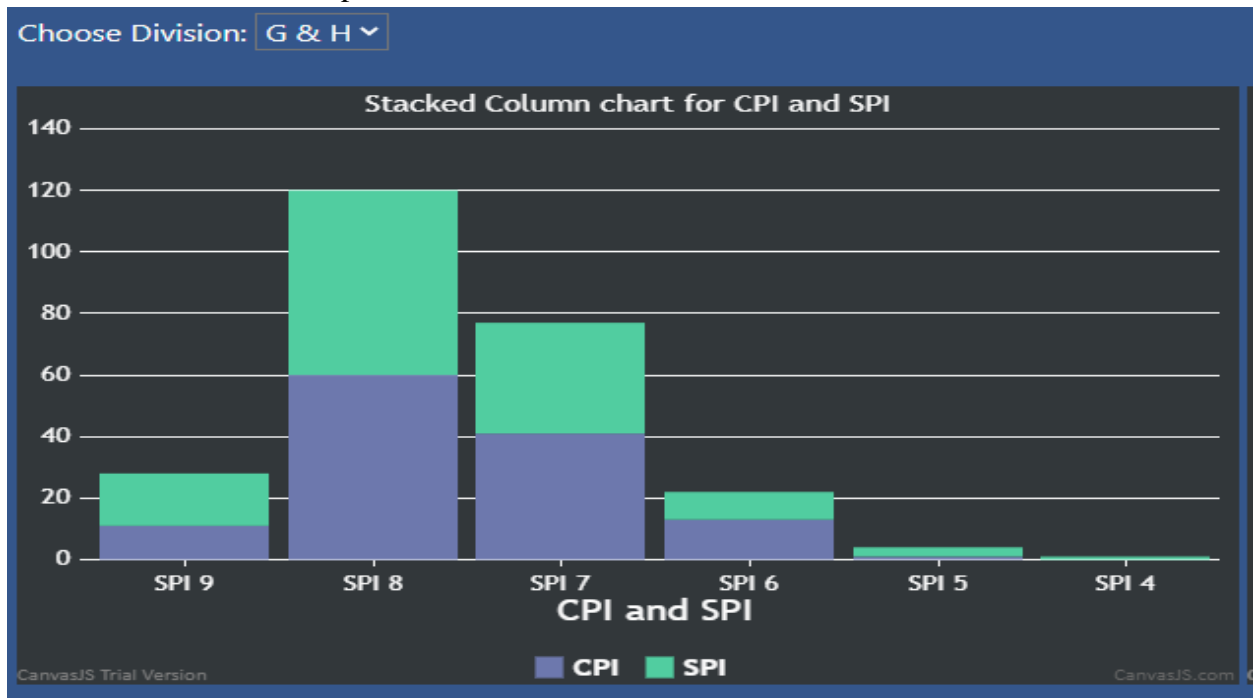


Fig 10.2

Above figure shows number of students having CPI and SPI in a specific range using stacked column chart.

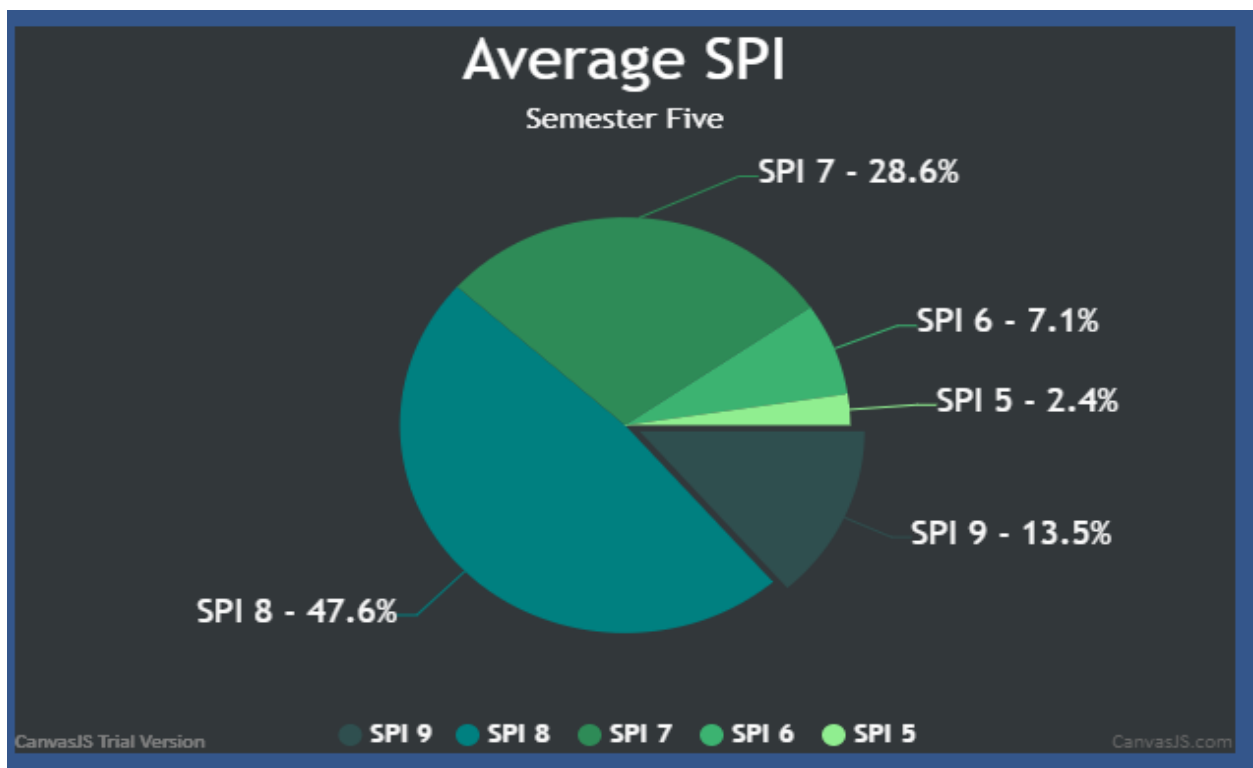


Fig 10.3

Above figure shows the percentage of students scoring specific SPI in semester five.

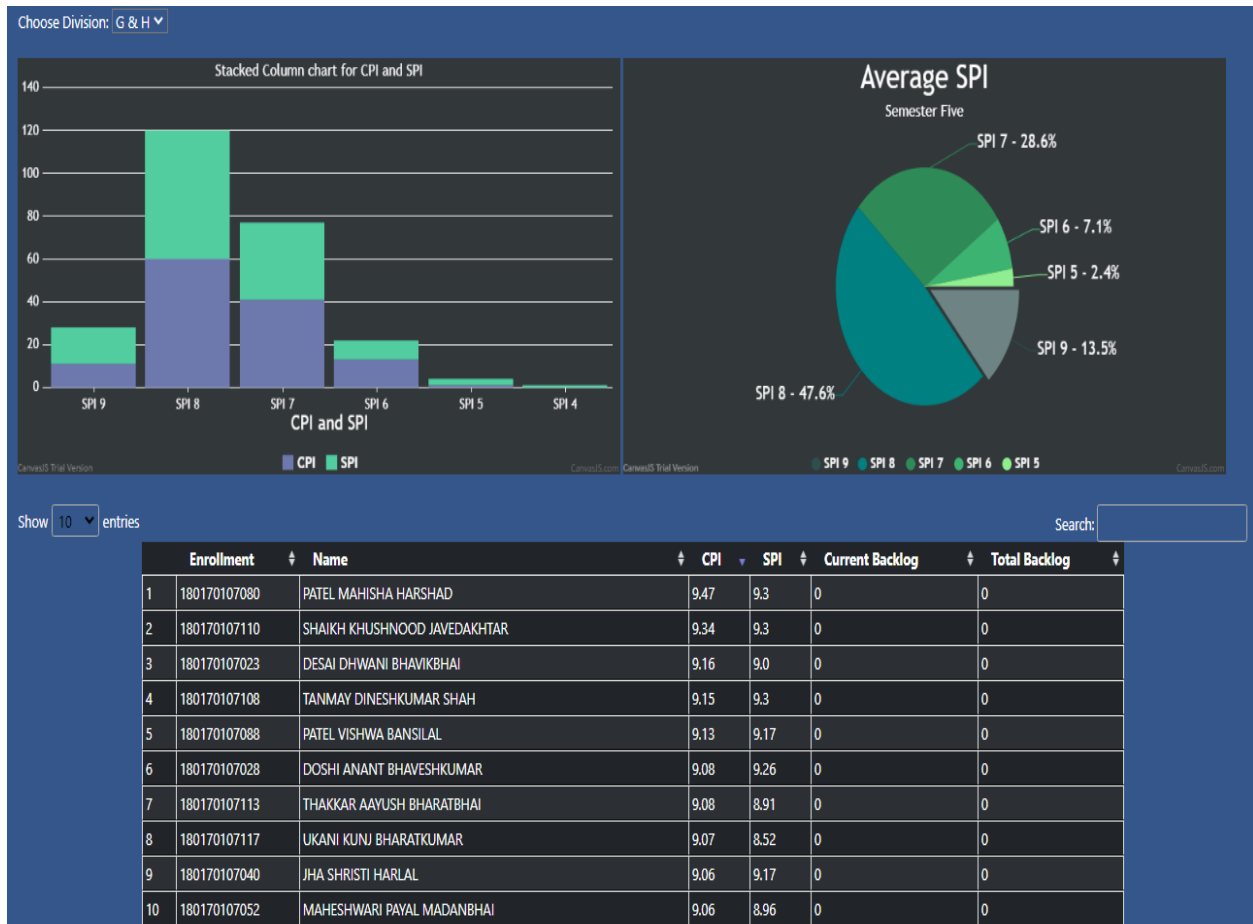


Fig 10.4

Above figure shows top 10 students of G and H division according to CPI.

Show 10 ▾ entries Search:

	Enrollment	Name	CPI	SPI	Current Backlog	Total Backlog
1	180170107077	PATEL KARM BHARATBHAI	8.91	9.52	0	0
2	180170107080	PATEL MAHISHA HARSHAD	9.47	9.3	0	0
3	180170107110	SHAIKH KHUSHNOOD JAVEDAKHTAR	9.34	9.3	0	0
4	180170107108	TANMAY DINESHKUMAR SHAH	9.15	9.3	0	0
5	180170107028	DOSHI ANANT BHAVESHKUMAR	9.08	9.26	0	0
6	180170107116	TRIVEDI TIRTHKUMAR NEHALKUMAR	8.91	9.22	0	0
7	180170107092	PESHEN VIRESH AVTAR	8.36	9.22	0	0
8	180170107088	PATEL VISHWA BANSILAL	9.13	9.17	0	0
9	180170107040	JHA SHRISTI HARLAL	9.06	9.17	0	0
10	180170107090	PATEL YASH NARESHBHAI	8.56	9.13	0	0

Showing 1 to 10 of 126 entries Previous 1 2 3 4 5 ... 13 Next

Fig 10.5

Above figure shows top 10 students of G and H division according to SPI of semester 5.



Show

10

entries

Search:

SIDDHARTH

	Enrollment	Name	CPI	SPI	Current Backlog	Total Backlog
1	180170107003	ASODARIYA SIDDHARTH DILIPBHAI	8.73	8.83	0	0
2	180170107030	GABU SIDDHARTH MERAMBHAI	7.96	8.96	0	0

Showing 1 to 2 of 2 entries (filtered from 126 total entries)

Previous

1

Next

Fig 10.6

Above figure shows the name of person according to Search result.