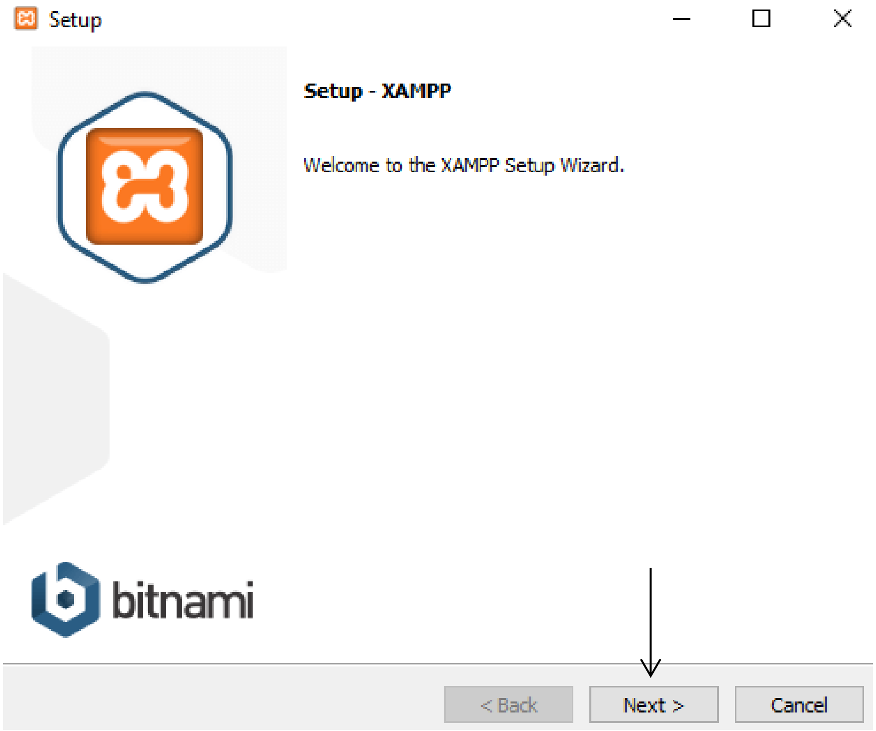
Practical - 1

AIM: Setup environment for all the tools

1. Install XAMPP/WAMPP server.

   i. Download the setup of Xampp Server from *https://www.apachefriends.org/download.html* and run it:

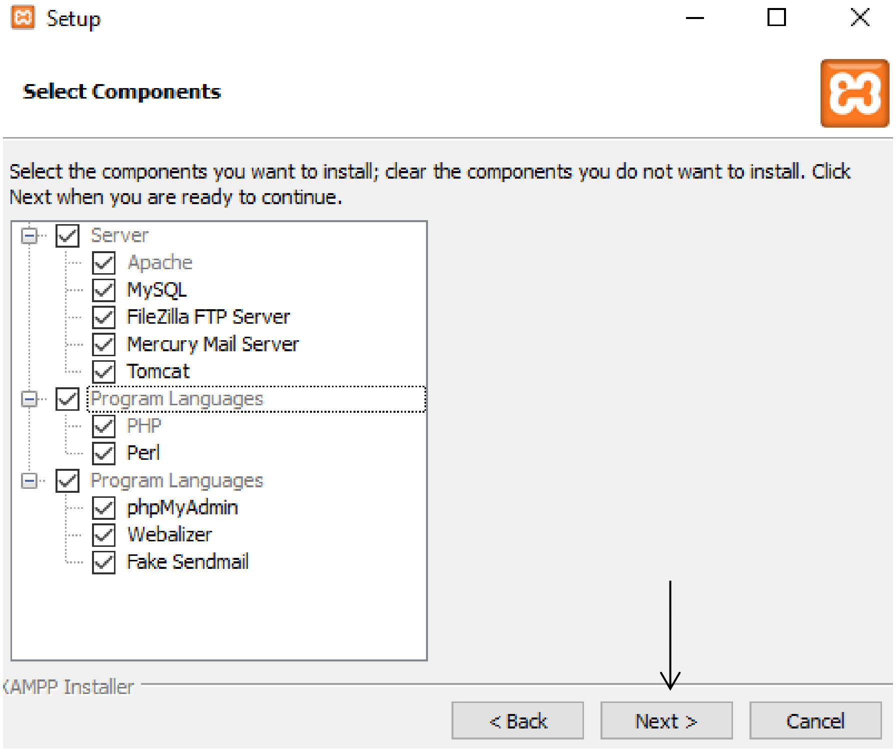


Setup

Setup - XAMPP

Welcome to the XAMPP Setup Wizard.

   ii. Select the components you want install :



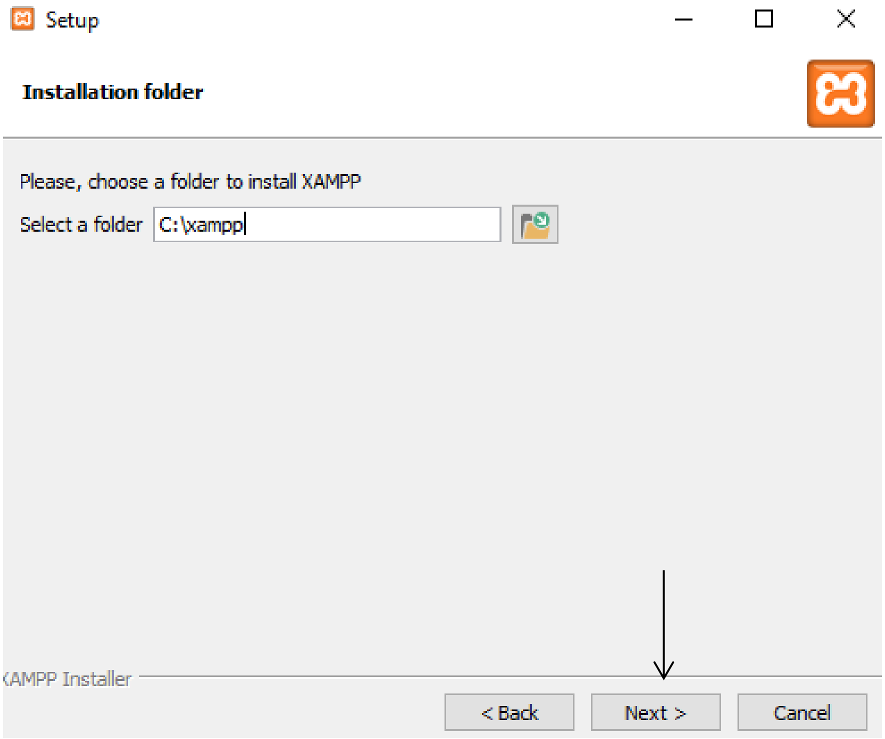
Setup

Select Components

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- ☑ Server
  - ☑ Apache
  - ☑ MySQL
  - ☑ FileZilla FTP Server
  - ☑ Mercury Mail Server
  - ☑ Tomcat
- ☑ Program Languages
  - ☑ PHP
  - ☑ Perl
- ☑ Program Languages
  - ☑ phpMyAdmin
  - ☑ Webalizer
  - ☑ Fake Sendmail

iii. Specify the installation location :



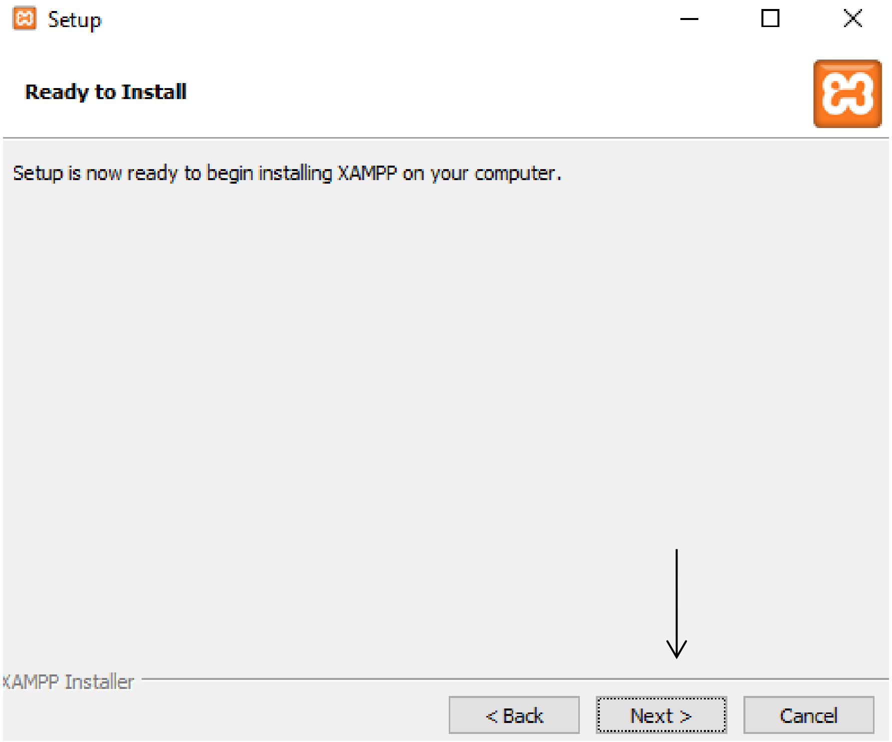
Setup

Installation folder

Please, choose a folder to install XAMPP

Select a folder C:\xampp

iv. Setup is now ready and will start the installation process :

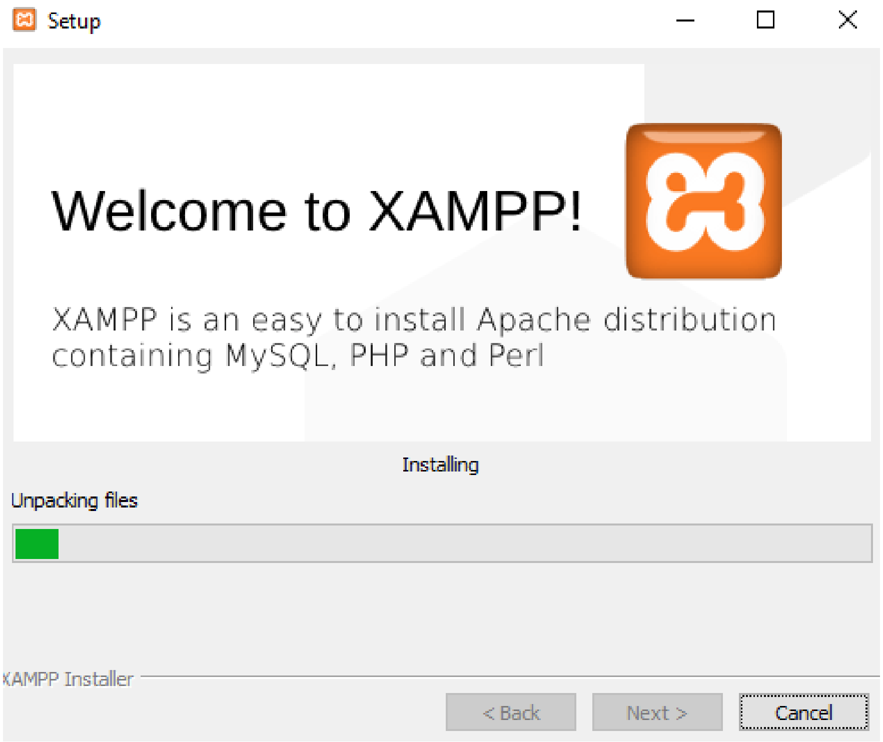


Setup

Ready to Install

Setup is now ready to begin installing XAMPP on your computer.

v. Wait until the installation finishes off :



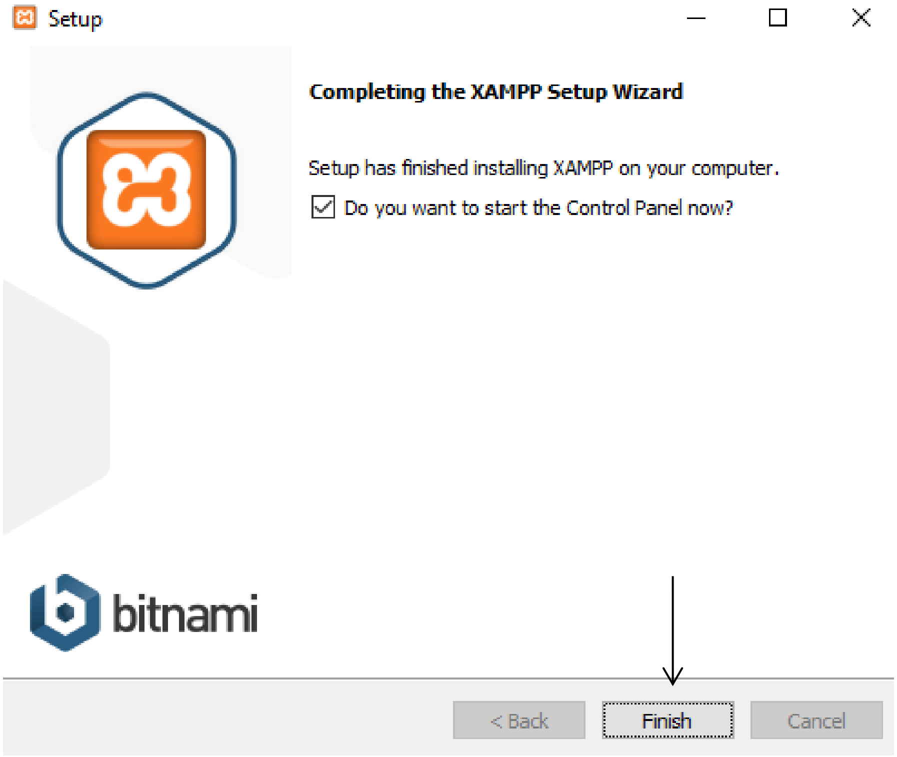
Welcome to XAMPP!

XAMPP is an easy to install Apache distribution containing MySQL, PHP and Perl

Installing

Unpacking files

vi. It is installed now and will launch the Control Panel :



Completing the XAMPP Setup Wizard

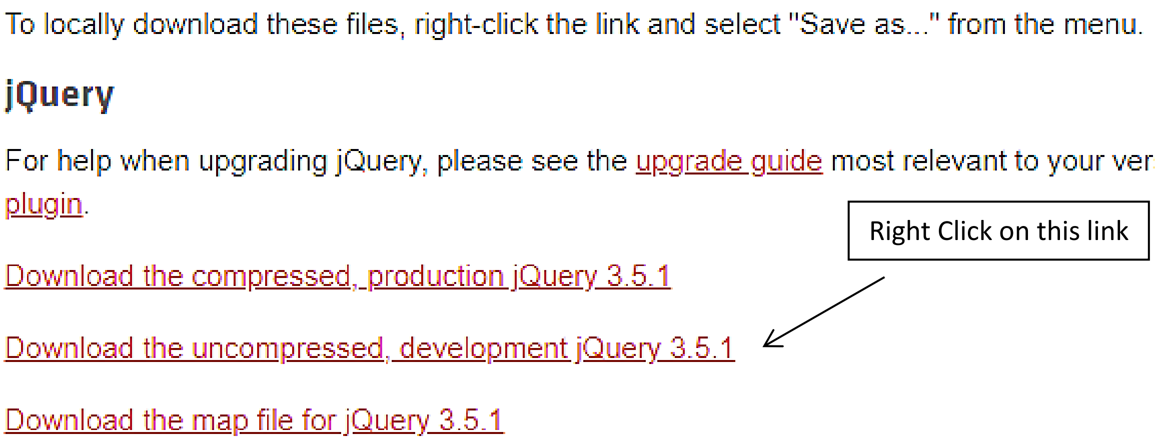
Setup has finished installing XAMPP on your computer.

☑ Do you want to start the Control Panel now?

2. Install jQuery library.

There are three ways to have jQuery all setup:

1) Directly downloading the uncompressed version of jQuery and storing it locally.
   - Head over to *https://jquery.com/download/* and right click on the below shown link and choose "Save as.." option from the pop-up.



To locally download these files, right-click the link and select "Save as..." from the menu.

jQuery

For help when upgrading jQuery, please see the upgrade guide most relevant to your version. plugin.

Download the compressed, production jQuery 3.5.1

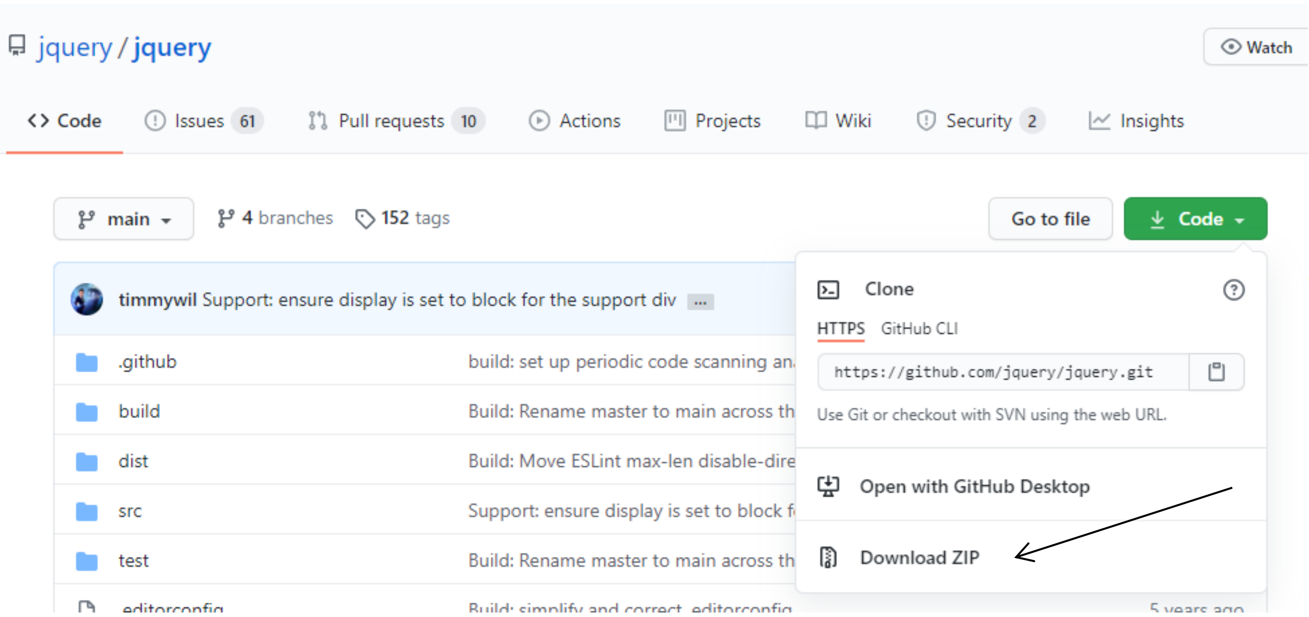
Download the uncompressed, development jQuery 3.5.1

Download the map file for jQuery 3.5.1

Right Click on this link

   - Store it locally and always give path to that location when you want to use jQuery in your projects.

2) Pulling the jQuery source code from GitHub to the local machine.
   - You can either head over to GitHub at *https://github.com/jquery/jquery*, and download the project from there as shown below.



timmywil Support: ensure display is set to block for the support div ...

https://github.com/jquery/jquery.git

Use Git or checkout with SVN using the web URL.

.github	build: set up periodic code scanning an
build	Build: Rename master to main across th
dist	Build: Move ESLint max-len disable-dire
src	Support: ensure display is set to block f
test	Build: Rename master to main across th
.editorconfig	Build: simplify and correct .editorconfig 5 years ago

   - Or can you run the following git clone command in your command prompt:

```
git clone git://github.com/jquery/jquery.git
```

3) Only using CDNs.

   - CDNs can offer a performance benefit by hosting jQuery on servers spread across the globe.

   - Include one of the following CDNs in a script tag in your file where you need the usage of jQuery.

   - jQuery CDN: `<script src = " https://code.jquery.com/jquery-3.5.1.js "> </script>`

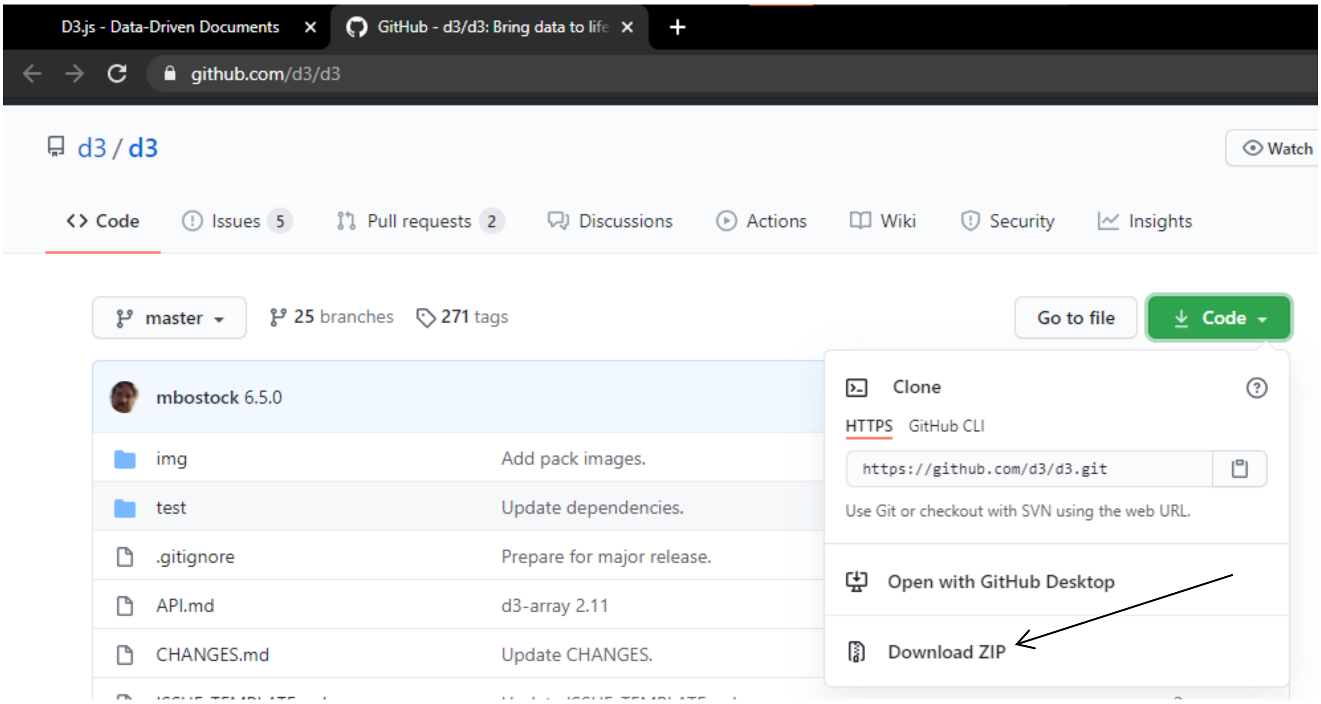
   - Google CDN: `<script src = "https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js "> </script>`

   - Microsoft CDN: `<script src = "https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.5.1.js"> </script>`

3. Install D3.js and Canvas.js.

   ○ D3.js:

   - Download the source code from GitHub at *https://github.com/d3/d3*, as zip and store it locally; or, use run the following command in your command prompt:
     `git clone git://github.com/d3/d3.git`



https://github.com/d3/d3.git

Use Git or checkout with SVN using the web URL.

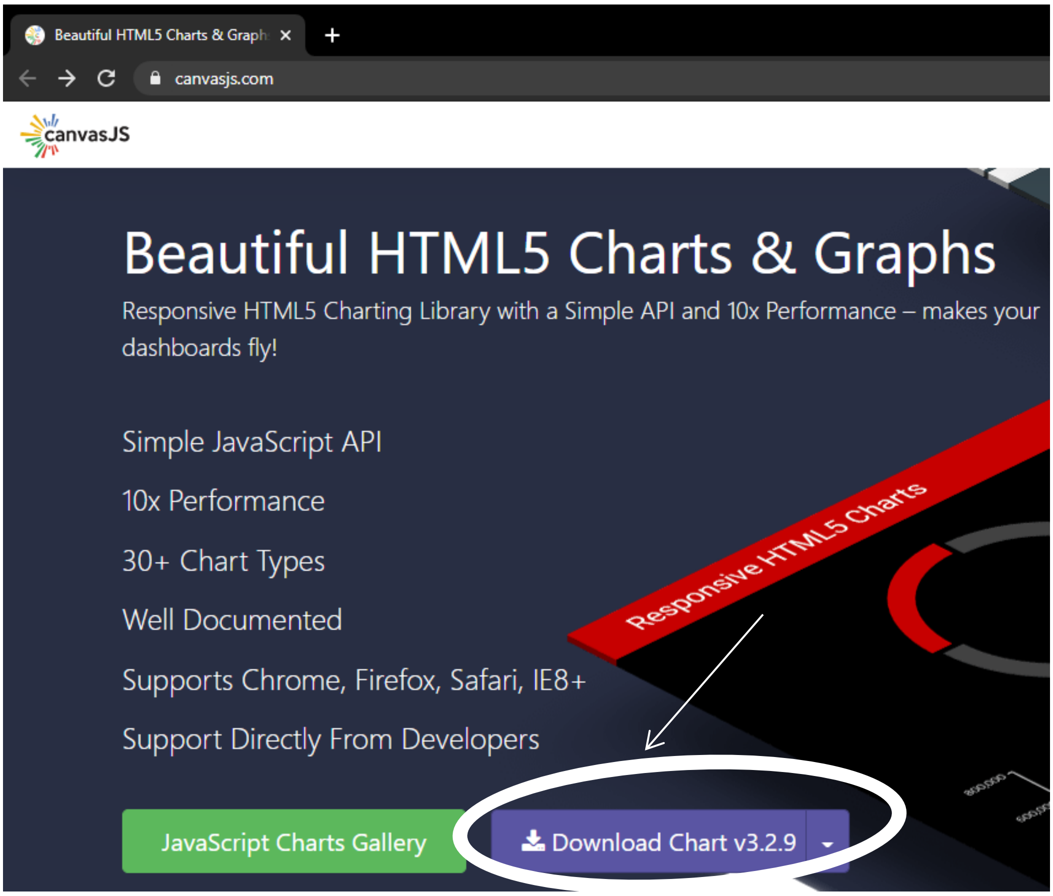
mbostock 6.5.0

img	Add pack images.
test	Update dependencies.
.gitignore	Prepare for major release.
API.md	d3-array 2.11
CHANGES.md	Update CHANGES.

   - Or directly use the CDN in the script tag in your file as:
     `<script src="https://d3js.org/d3.v6.min.js"></script>`

- Canvas.js:

  - Download the source code from *https://canvasjs.com/* as zip and store it locally.



Beautiful HTML5 Charts & Graphs

Responsive HTML5 Charting Library with a Simple API and 10x Performance – makes your dashboards fly!

Simple JavaScript API

10x Performance

30+ Chart Types

Well Documented

Supports Chrome, Firefox, Safari, IE8+

Support Directly From Developers

  - Or directly use the CDN in the script tag in your file as:

```
<script src =
"https://cdnjs.cloudflare.com/ajax/libs/canvasjs/1.7.0/canva
sjs.js"> </script>
```

4. Configure Google Chart API and Google Maps API.

- Google Chart API:

  - To use Google Chart API in your file, you can include following script tag in your head or body section:

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

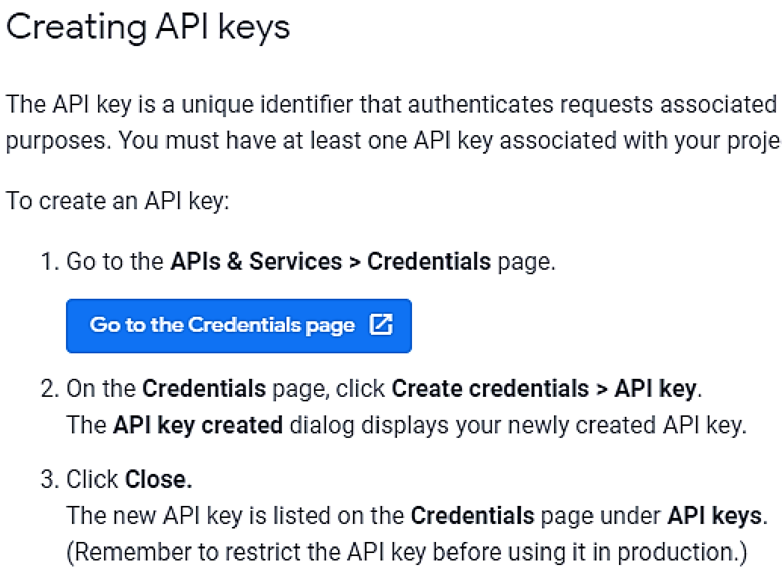
  - This loads the loader itself, which is needed to be done only one time.

  - After loading the loader, the *google.charts.load* function can be called one or more times to load packages for particular chart types.

```
<script>
   google.charts.load('current', {packages: ['corechart'],
callback: drawChartFunction});
</script>
```

- Parameters of the *google.charts.load* function:
    - Version: It is the first argument that specifies the version number; it can typically be of one of three values: 'current', 'upcoming', and some specific version number.
    - Load Settings: This second parameter is an object for specifying settings. The following properties are supported for settings:
        - Packages: An array of zero or more packages. Each package that is loaded will have the code required to support a set of functionalities, typically a type of chart. Ex. 'corechart', 'mapchart', etc.
        - Language: The code for the language or locale that should be to customize text that might be part of the chart.
        - Callback: A function that will be called once the packages have been loaded. Alternatively, you can specify this function by calling *google.charts.setOnLoadCallback*.
        - mapsApiKey: This setting lets you specify a key that you may use with Geochart and Map Chart.

- Google Maps API:
    - To use Map Chart, that uses Maps JavaScript API, you will require mapsApiKey first.
    - To get your own key head over to https://developers.google.com/maps/documentation/javascript/get-api-key and follow the steps shown there.



Creating API keys

The API key is a unique identifier that authenticates requests associated with your project for usage and billing purposes. You must have at least one API key associated with your project.

To create an API key:

1. Go to the **APIs & Services > Credentials** page.

2. On the **Credentials** page, click **Create credentials > API key**.
   The **API key created** dialog displays your newly created API key.
3. Click **Close.**
   The new API key is listed on the **Credentials** page under **API keys**.
   (Remember to restrict the API key before using it in production.)

- You must include an API key with every Maps JavaScript API request. In the following example, replace YOUR_API_KEY with your API key.

```
<script async defer src =
"https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&callb
ack=initMap" type = "text/javascript"> </script>
```

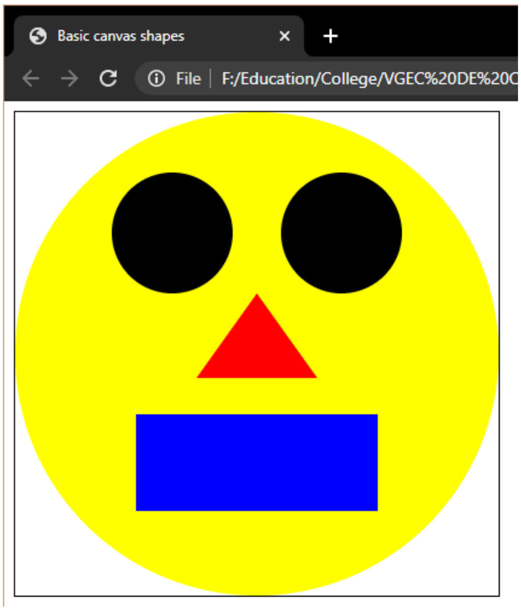
Practical - 2

AIM: Develop the following Programs Using HTML5 CANVAS and SVG TAG

1. Develop the Different basic Graphical Shapes using HTM5 CANVAS.
   - Program:

```
<html>
<head><title>Basic canvas shapes</title></head>
<body>
<canvas id="basic" width="400" height="400" style="border:1px solid #000;"/>
<script>
var c = document.getElementById("basic");
var ctx = c.getContext("2d");

//face
ctx.beginPath();
ctx.arc(200, 200, 200, 0, 2 * Math.PI);
ctx.fillStyle = "yellow";
ctx.fill();

//eyes
ctx.fillStyle = "black";
ctx.beginPath();
ctx.arc(130, 100, 50, 0, 2 * Math.PI);
ctx.fill();

ctx.beginPath();
ctx.arc(270, 100, 50, 0, 2 * Math.PI);
ctx.fill();

//nose
ctx.beginPath();
ctx.moveTo(200, 150);
ctx.lineTo(200, 220);
ctx.lineTo(250, 220);
ctx.fillStyle = "red";
ctx.fill();
ctx.closePath();

ctx.beginPath();
ctx.moveTo(200, 150);
ctx.lineTo(200, 220);
ctx.lineTo(150, 220);
ctx.fillStyle = "red";
ctx.fill();
ctx.closePath();

//mouth
ctx.beginPath();
ctx.fillStyle = "blue";
ctx.fillRect(100, 250, 200, 80);
</script>
</body>
</html>
```

- Output:

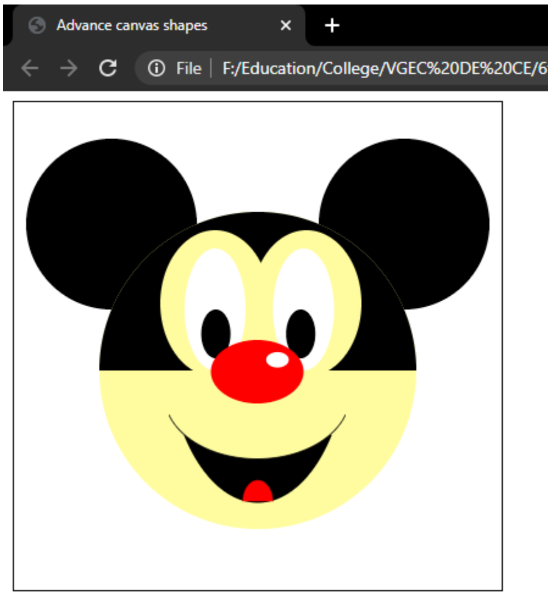


2. Develop the Different advance Graphical Shapes using HTM5 CANVAS.

   - Program:

```
<html>
<head><title>Advance canvas shapes</title></head>
<body>
<canvas id="basic" width="400" height="400" style="border:1px solid #000;"/>
<script>
var c = document.getElementById("basic");
var ctx = c.getContext("2d");

//ears
ctx.beginPath();
ctx.arc(80, 100, 70, 0, 2 * Math.PI);
ctx.fillStyle = "black";
ctx.fill();

ctx.beginPath();
ctx.arc(320, 100, 70, 0, 2 * Math.PI);
ctx.fillStyle = "black";
ctx.fill();

//face
ctx.beginPath();
ctx.arc(200, 220, 130, 0, 2 * Math.PI);
ctx.fillStyle = "#FFFCA0";
ctx.fill();

      //half circle
ctx.beginPath();
ctx.arc(200, 220, 130, Math.PI,0);
ctx.fillStyle = "black";
ctx.fill();
ctx.save();
        //eye gap
ctx.scale(0.75,1);
ctx.beginPath();
ctx.arc(220, 165, 60, 0, Math.PI*2, false);
ctx.fillStyle = "#FFFCA0";
ctx.fill();
ctx.restore();

ctx.save();
ctx.scale(0.75,1);
ctx.beginPath();
ctx.arc(320, 165, 60, 0, Math.PI*2, false);
ctx.fillStyle = "#FFFCA0";
ctx.fill();
ctx.restore();

ctx.save();
//eyes
ctx.scale(0.50,1);
ctx.beginPath();
ctx.arc(330, 170, 50, 0, Math.PI*2, false);
ctx.fillStyle = "white";
ctx.fill();
ctx.restore();

ctx.save();
ctx.scale(0.50,1);
ctx.beginPath();
ctx.arc(475, 170, 50, 0, Math.PI*2, false);
ctx.fillStyle = "white";
ctx.fill();
ctx.restore();

ctx.save();
ctx.scale(0.60,1);
ctx.beginPath();
ctx.arc(276, 190, 20, 0, Math.PI*2, false);
ctx.fillStyle = "black";
ctx.fill();
ctx.restore();

ctx.save();
ctx.scale(0.60,1);
ctx.beginPath();
ctx.arc(392, 190, 20, 0, Math.PI*2, false);
ctx.fillStyle = "black";
ctx.fill();
ctx.restore();

ctx.save();
//nose
ctx.scale(0.95,0.65);
ctx.beginPath();
ctx.arc(210, 340, 40, 0, Math.PI*2, false);
ctx.fillStyle = "red";
ctx.fill();

ctx.scale(0.65,0.65);
ctx.beginPath();
ctx.arc(350, 500, 15, 0, Math.PI*2, false);
ctx.fillStyle = "white";
ctx.fill();

ctx.restore();

ctx.save();
//mouth
ctx.scale(0.50,0.9);
ctx.beginPath();
ctx.arc(400, 215, 150, 0.2 * Math.PI, 0.8 * Math.PI, false);
ctx.fillStyle = "black";
ctx.fill();

ctx.restore();

ctx.save();
ctx.scale(0.95,0.65);
ctx.beginPath();
ctx.arc(210, 370, 80, 0.1 * Math.PI, 0.9 * Math.PI, false);
ctx.fillStyle = "#FFFCA0";
ctx.fill();

ctx.restore();

ctx.save();
ctx.scale(0.95,0.65);
ctx.beginPath();
ctx.arc(210, 370, 80, 0.1 * Math.PI, 0.9 * Math.PI, false);
ctx.stroke();

ctx.restore();

    //tongue
ctx.save();
ctx.beginPath();
ctx.scale(0.25,0.35);
ctx.arc(800, 935, 50, Math.PI,0);
ctx.fillStyle = "red";
ctx.fill();
ctx.stroke();
ctx.restore();

</script>
</body>
</html>
```

- Output:

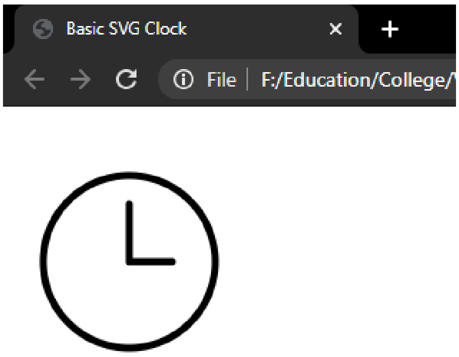


3. Develop the Different basic Graphical Shapes using HTM5 SVG.

- Program:

```
<html>
<head><title>Basic SVG Clock</title></head>
<body>
<svg width="300" height="300" fill="none" stroke="black" stroke-width="5" stroke-linecap="round" stroke-linejoin="round">
  <circle cx="80" cy="100" r="60"/>
  <path d="M80 60V100 H110" />
</svg>
</body>
</html>
```

- Output:

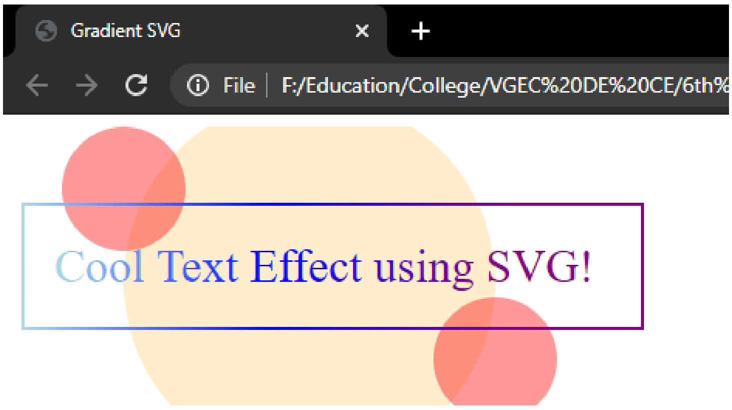


4. Develop the Different advance Graphical Shapes using HTM5 SVG.

   - Program:

```
<html>
<head>
<title>Gradient SVG</title>
</head>
<body>
<svg width="500" height="180" viewBox="0 0 500 180">
  <defs>
    <linearGradient id="txt-grad">
      <stop offset="0%" stop-color="lightblue" />
      <stop offset="40%" stop-color="blue" />
      <stop offset="80%" stop-color="purple" />
    </linearGradient>
  </defs>
  <circle cx="190" cy="100" r="120" style="fill:orange;fill-opacity:0.2"/>
  <circle cx="70" cy="40" r="40" style="fill:red;fill-opacity:0.4"/>
  <circle cx="310" cy="150" r="40" style="fill:red;fill-opacity:0.4"/>
  <rect x="5" y="50" width="400" height="80" style="stroke:url(#txt-grad);stroke-
width:2;fill:transparent"/>
  <text x="25" y="100" style = "font-size: 30;fill: url(#txt-grad)">Cool Text Effect using
SVG!</text>
</svg>
</body>
</html>
```

   - Output:



Cool Text Effect using SVG!

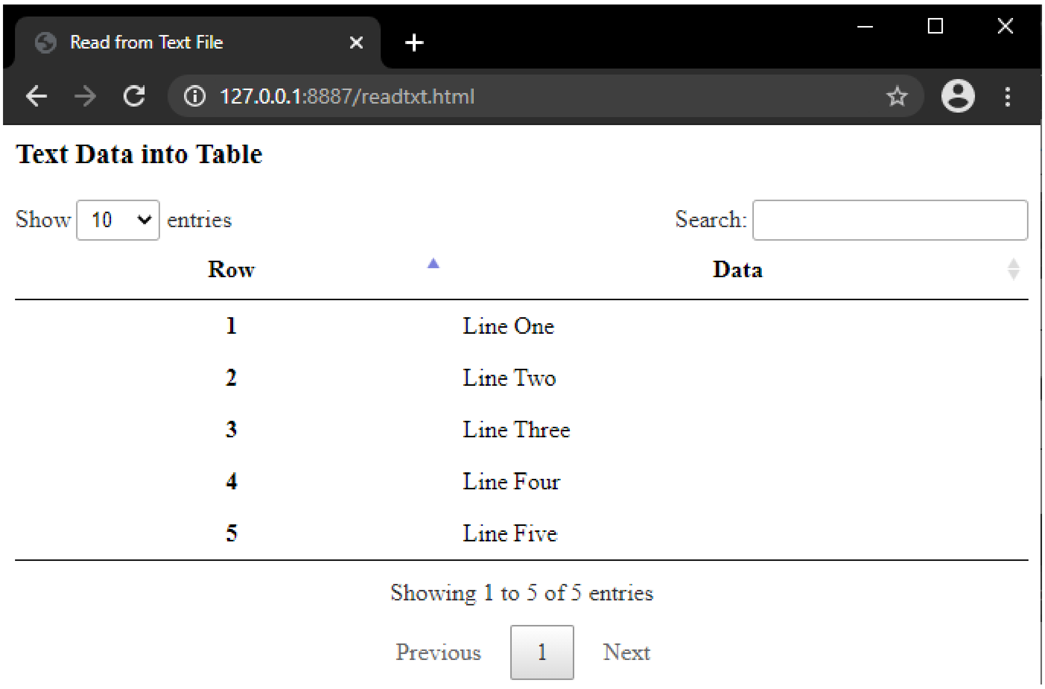
Practical - 3

AIM: Develop the following Programs Using HTML5 and JavaScript

1. Read the data .txt file and draw Data Table.
   - Program:

```
<html>
<head>
<title>Read from Text File</title>
<script src="https://code.jquery.com/jquery-3.5.1.js"></script>
<script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
<link rel="stylesheet" type="text/css"
href="https://cdn.datatables.net/1.10.23/css/jquery.dataTables.min.css">
</head>
<body>
<h3>Text Data into Table</h3>
<table/>
<script>
    var tableContent = '';
    tableContent += '<thead>';
    tableContent += '<tr>';
    tableContent += '<th scope="col"> Row </th>';
    tableContent += '<th scope="col"> Data </th>';
    tableContent += '</tr>';
    tableContent += '</thead>';

    $.get('http://127.0.0.1:8887/data.txt', function(content) {

    tableContent += '<tbody>';

    var lines = content.split('\n');

    $.each(lines, function(key,line){
      tableContent += '<tr>';
      tableContent += '<th scope="row">' + (key+1) + '</th>';
      tableContent += '<td>' + line + '</td>';
      tableContent += '</tr>';
    });

    tableContent += '</tbody>';
    $('table').html(tableContent);

    }).done(function(){
       $('table').DataTable();
    });
</script>
</body>
</html>
```

- Output:

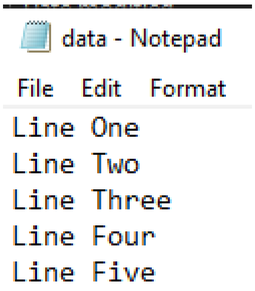


Text Data into Table

Show 10 entries Search:

Row	Data
1	Line One
2	Line Two
3	Line Three
4	Line Four
5	Line Five

Showing 1 to 5 of 5 entries



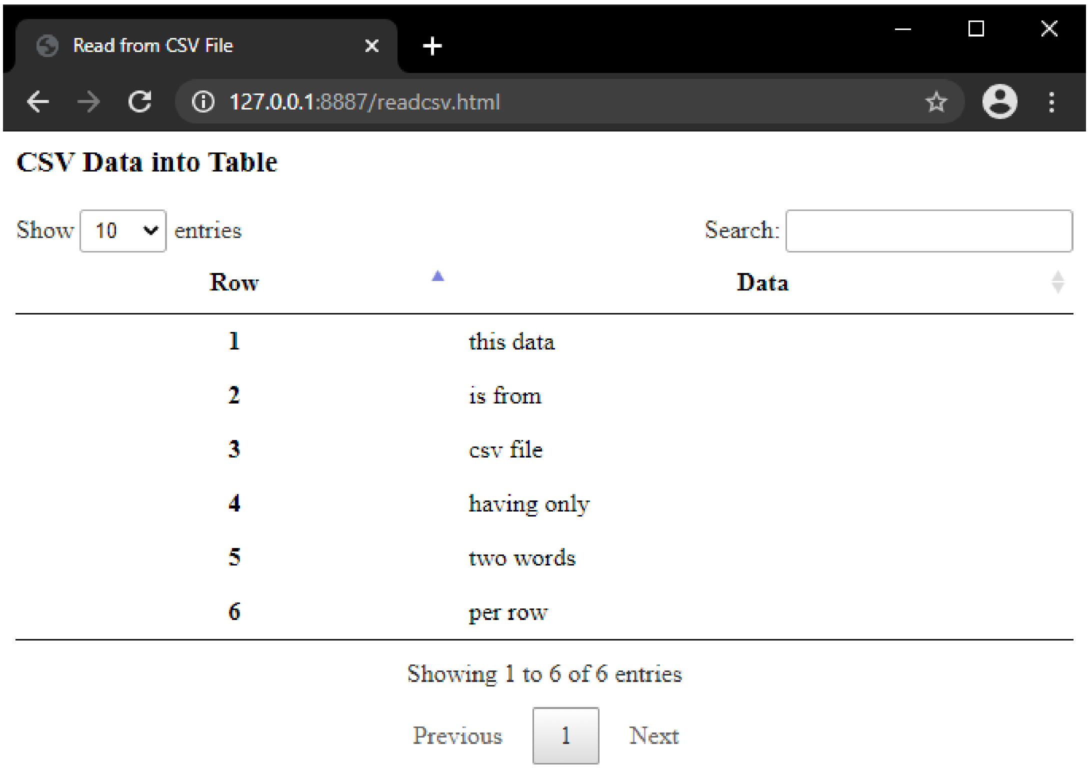
```
Line One
Line Two
Line Three
Line Four
Line Five
```

2. Read the data .csv file and draw Data Table.
   - Program:

```
<html>
<head>
<title>Read from CSV File</title>
<script src="https://code.jquery.com/jquery-3.5.1.js"></script>
<script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
<link rel="stylesheet" type="text/css"
href="https://cdn.datatables.net/1.10.23/css/jquery.dataTables.min.css">
</head>
<body>
<h3>CSV Data into Table</h3>
<table/>
<script>
    var tableContent = '';

    $.get('http://127.0.0.1:8887/dc.csv', function(content) {
    var lines = content.split(',');

    tableContent += '<thead>';
    tableContent += '<tr>';
```

```
        tableContent += '<th scope="col">' + lines[0] + '</th>';
        tableContent += '<th scope="col">' + lines[1] + '</th>';
        tableContent += '</tr>';
        tableContent += '</thead>';

        tableContent += '<tbody>';
        $.each(lines, function(index,line){
            if(index<2) return true;
            tableContent += '<tr>';
            tableContent += '<th scope="row">' + (index-1) + '</th>';
            tableContent += '<td>' + line + '</td>';
            tableContent += '</tr>';
         });

        tableContent += '</tbody>';
        $('table').html(tableContent);

        }).done(function(){
            $('table').DataTable();
        });
</script>
</body>
</html>
```

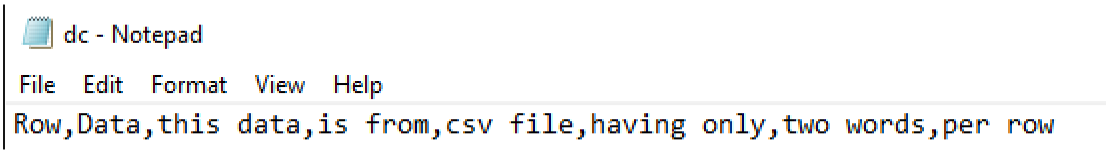
- Output:



CSV Data into Table

Row	Data
1	this data
2	is from
3	csv file
4	having only
5	two words
6	per row

Showing 1 to 6 of 6 entries

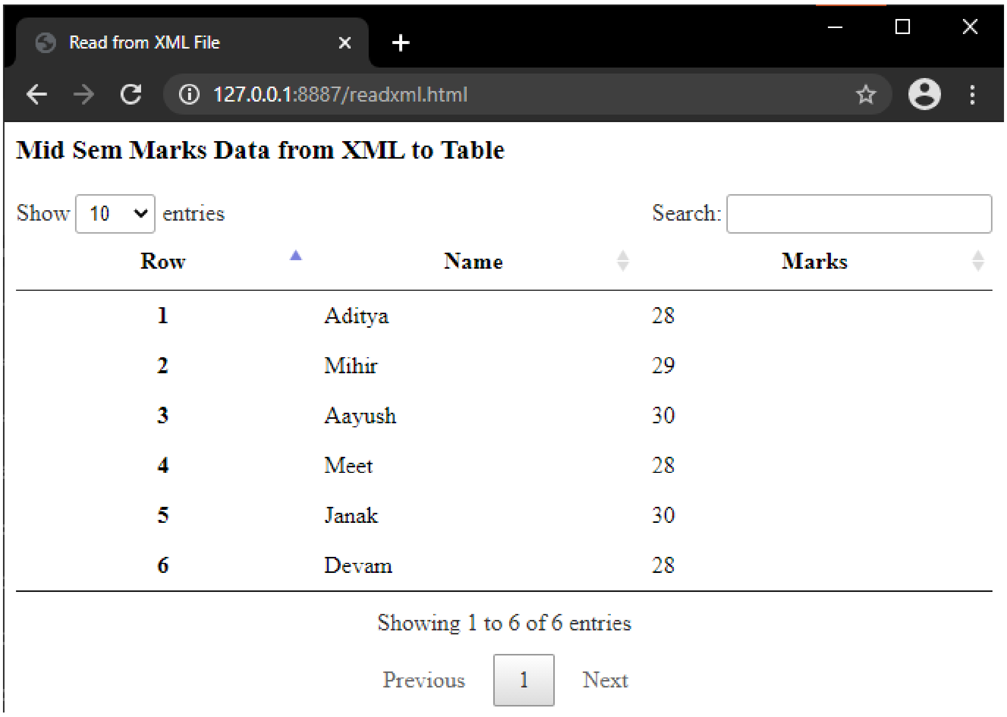


```
Row,Data,this data,is from,csv file,having only,two words,per row
```

3. Read the data XML file and draw Data Table.
   - Program:

```
<html>
<head>
<title>Read from XML File</title>
<script src="https://code.jquery.com/jquery-3.5.1.js"></script>
<script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
<link rel="stylesheet" type="text/css"
href="https://cdn.datatables.net/1.10.23/css/jquery.dataTables.min.css">
</head>
<body>
<h3>Mid Sem Marks Data from XML to Table</h3>
<table/>
<script>
    var tableContent = '';
    tableContent += '<thead>';
    tableContent += '<tr>';
    tableContent += '<th scope="col">Row</th>';
    tableContent += '<th scope="col">Name</th>';
    tableContent += '<th scope="col">Marks</th>';
    tableContent += '</tr>';
    tableContent += '</thead>';

    $.get('http://127.0.0.1:8887/dx.xml', function(content) {
    tableContent += '<tbody>';

    var $Students = $(content).find('student');
     $Students.each(function(index){
        tableContent += '<tr>';
        tableContent += '<th scope="row">' + (index+1) + '</th>';
        tableContent += '<td>' + $(this).find('name').text() + '</td>';
        tableContent += '<td>' + $(this).find('marks').text() + '</td>';
        tableContent += '</tr>';
     });

    tableContent += '</tbody>';
    $('table').html(tableContent);

    }).done(function(){
        $('table').DataTable();
    });
</script>
</body>
</html>
```

- Output:

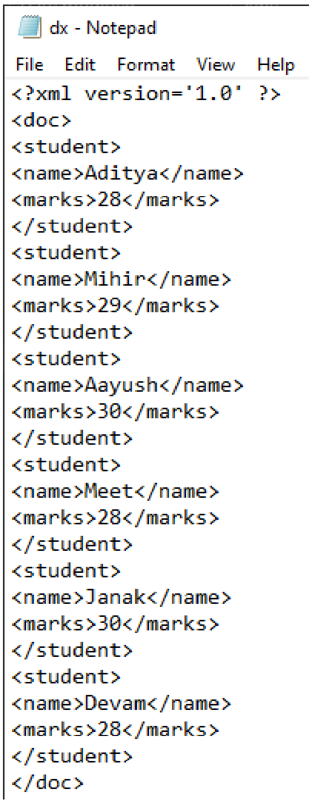


Mid Sem Marks Data from XML to Table

Show 10 entries Search:

Row	Name	Marks
1	Aditya	28
2	Mihir	29
3	Aayush	30
4	Meet	28
5	Janak	30
6	Devam	28

Showing 1 to 6 of 6 entries

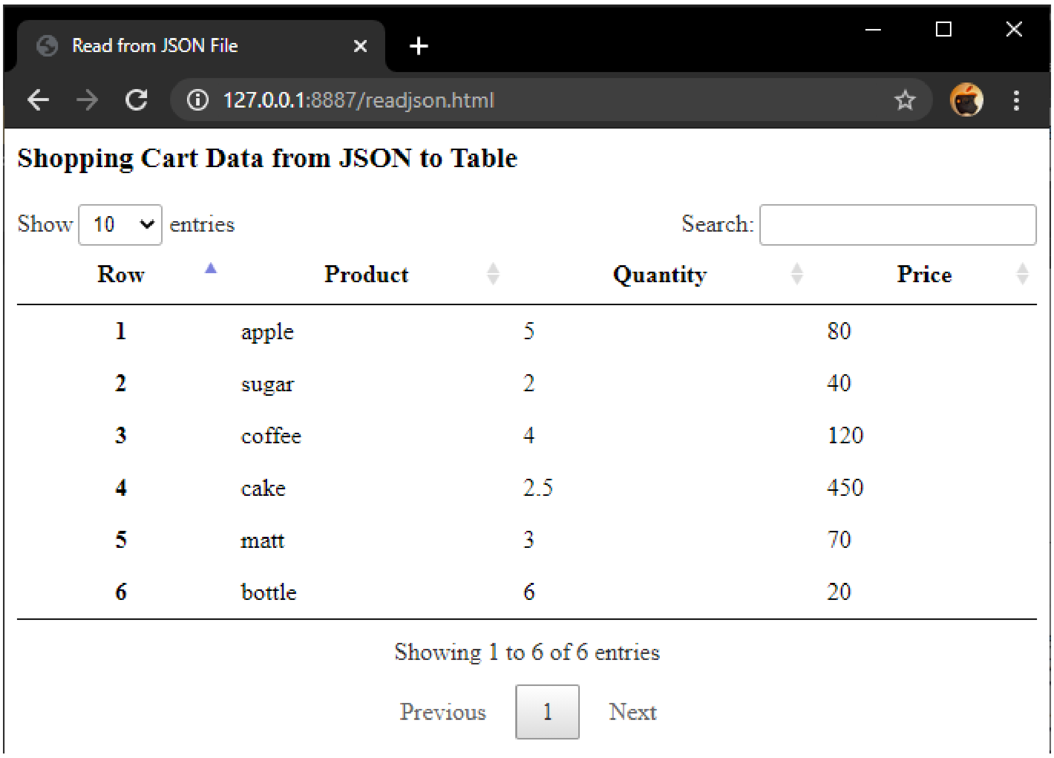


```
<?xml version='1.0' ?>
<doc>
<student>
<name>Aditya</name>
<marks>28</marks>
</student>
<student>
<name>Mihir</name>
<marks>29</marks>
</student>
<student>
<name>Aayush</name>
<marks>30</marks>
</student>
<student>
<name>Meet</name>
<marks>28</marks>
</student>
<student>
<name>Janak</name>
<marks>30</marks>
</student>
<student>
<name>Devam</name>
<marks>28</marks>
</student>
</doc>
```

4. Read JSON Data and draw Data Table.
   - Program:

```
<html>
<head>
<title>Read from JSON File</title>
<script src="https://code.jquery.com/jquery-3.5.1.js"></script>
<script src="https://cdn.datatables.net/1.10.23/js/jquery.dataTables.min.js"></script>
<link rel="stylesheet" type="text/css"
href="https://cdn.datatables.net/1.10.23/css/jquery.dataTables.min.css">
</head>
<body>
<h3>Shopping Cart Data from JSON to Table</h3>
<table/>
<script>
    var tableContent = '';
    tableContent += '<thead>';
    tableContent += '<tr>';
    tableContent += '<th scope="col">Row</th>';
    tableContent += '<th scope="col">Product</th>';
    tableContent += '<th scope="col">Quantity</th>';
    tableContent += '<th scope="col">Price</th>';
    tableContent += '</tr>';
    tableContent += '</thead>';

    $.getJSON('http://127.0.0.1:8887/dj.json', function(content) {

    tableContent += '<tbody>';
    $.each(content,function(key,value){
        tableContent += '<tr>';
        tableContent += '<th scope="row">' + (key+1) + '</th>';
        tableContent += '<td>' + value.product + '</td>';
        tableContent += '<td>' + value.quantity + '</td>';
        tableContent += '<td>' + value.price + '</td>';
        tableContent += '</tr>';
     });

    tableContent += '</tbody>';
    $('table').html(tableContent);

    }).done(function(){
        $('table').DataTable();
    });
</script>
</body>
</html>
```

- Output:



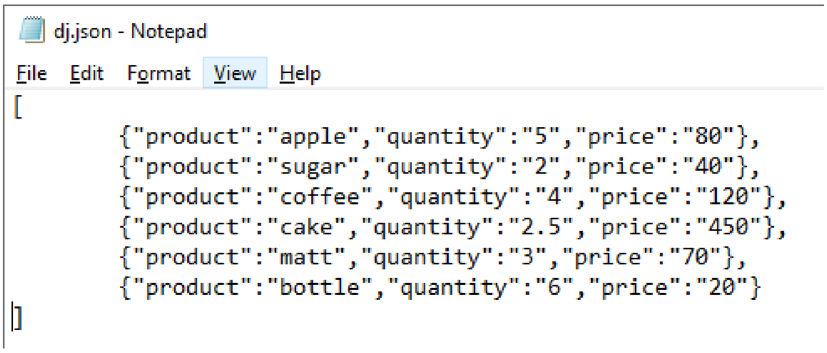
Shopping Cart Data from JSON to Table

Show 10 entries Search:

Row	Product	Quantity	Price
1	apple	5	80
2	sugar	2	40
3	coffee	4	120
4	cake	2.5	450
5	matt	3	70
6	bottle	6	20

Showing 1 to 6 of 6 entries

Previous 1 Next



```
[
	{"product":"apple","quantity":"5","price":"80"},
	{"product":"sugar","quantity":"2","price":"40"},
	{"product":"coffee","quantity":"4","price":"120"},
	{"product":"cake","quantity":"2.5","price":"450"},
	{"product":"matt","quantity":"3","price":"70"},
	{"product":"bottle","quantity":"6","price":"20"}
]
```

Practical - 4

AIM: Develop the following Programs Using HTML5 and JavaScript

1. Develop the simple bar chart using HTM5 CANVAS.
    - Program:

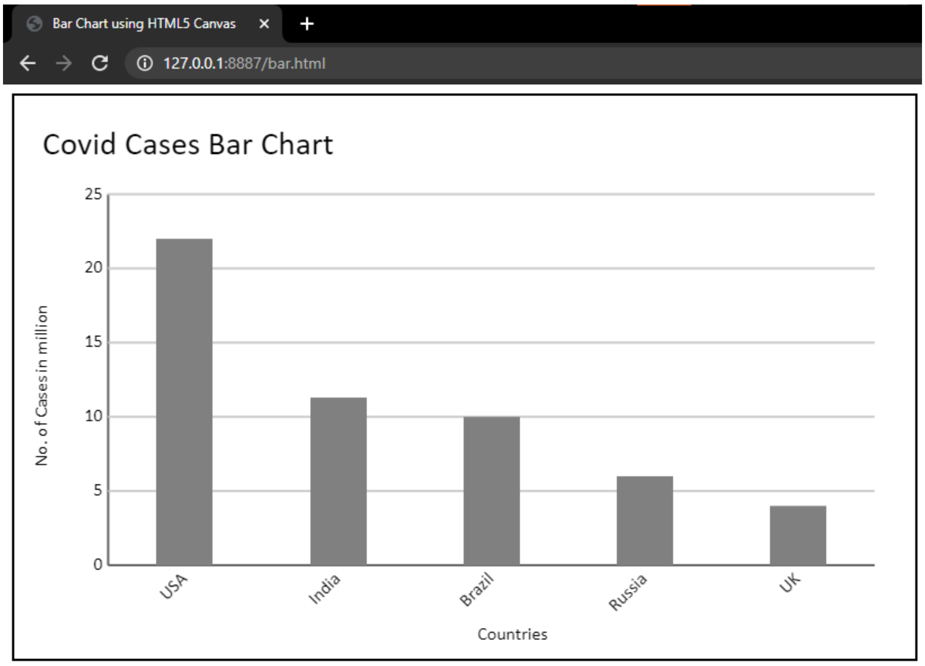
```
<html>
<head><title>Bar Chart using HTML5 Canvas</title></head>
<body>
   <canvas width="800" height="500" style="border: 2px solid black;"/>
   <script>
      var label = ["USA","India","Brazil","Russia","UK"];
      var data = [22,11.3,10,6,4];

      const canvas = document.querySelector('canvas');
      const color = "grey";
      const barWidth = 50;
      const step = 5;
      const max = 25;
      const maxValue = max - Math.floor(max % step);
      const minValue = 0;
      const title = "Covid Cases Bar Chart";
      const ytitle = "No. of Cases in million";
      const xtitle = "Countries";
      const font = "12pt Calibri";
      const axisColor = "#666";
      const gridColor = "#CCC";
      const padding = 25;

      const context = canvas.getContext("2d");
      const range = maxValue - minValue;
      const gridLines = Math.round(range / step);
      const maxWidth = context.measureText(maxValue - (gridLines/2 * step)).width;
      const x = padding + (maxWidth*3);
      const y = padding * 3.5;
      const width = canvas.width - (maxWidth + padding * 4);
      const height = canvas.height - (context.measureText(label[0]).width + padding * 6);

      //Title of the chart
      context.save();
      context.font = "22pt Calibri";
      context.fillStyle = "black";
      context.textAlign = "right";
      context.textBaseline = "middle";
      context.fillText(title, context.measureText(title).width + padding, padding + 20);
      context.restore();
```

```
        //Y axis line
        context.save();
        context.beginPath();
        context.moveTo(x, y);
        context.lineTo(x, height + y);
        context.strokeStyle = axisColor;
        context.lineWidth = 2;
        context.stroke();
        context.restore();

        //Y axis label
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
        context.rotate(-1 * Math.PI / 2);
        context.fillText(ytitle, height - (context.measureText(title).width * 4 ) + padding*2, 
padding);
        context.restore();

        //Y axis ticks
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
        for (var n = 0; n <= gridLines; n++) {
          var value = maxValue - (n * step);
          var thisY = (n * height / gridLines) + y;
          context.fillText(value, x - 5, thisY);
        }
        context.restore();

        //X axis line
        context.save();
        context.beginPath();
        context.moveTo(x, y + height);
        context.lineTo(x + width, y + height);
        context.strokeStyle = axisColor;
        context.lineWidth = 2;
        context.stroke();
        context.restore();
```

```
//X axis label
context.save();
context.font = font;
context.fillStyle = "black";
context.textAlign = "right";
context.textBaseline = "middle";
context.fillText(xtitle, x + width/2 + padding*2, y + height + padding*2.5);
context.restore();

//X axis ticks
context.save();
var barSpacing = width / label.length;
for (var n = 0; n < label.length; n++) {
  var l = label[n];
  context.save();
  context.translate(x + ((n + 1 / 2) * barSpacing), y + height + 10);
  context.rotate(-1 * Math.PI / 4);
  context.font = font;
  context.fillStyle = "black";
  context.textAlign = "right";
  context.textBaseline = "middle";
  context.fillText(l, 0, 0);
  context.restore();
}
context.restore();

//Inner Grid of the chart
context.save();
context.strokeStyle = gridColor;
context.lineWidth = 2;
for (var n = 0; n < gridLines; n++) {
  var ty = (n * height / gridLines) + y;
  context.beginPath();
  context.moveTo(x, ty);
  context.lineTo(x + width, ty);
  context.stroke();
}
context.restore();

//Bars
context.save();
var barSpacing = width / data.length;
var unitHeight = height / range;
for (var n = 0; n < data.length; n++) {
  var barHeight = (data[n] - minValue) * unitHeight;
```

```
     if (barHeight > 0) {
      context.save();
      context.translate(Math.round(x + ((n + 1 / 2) * barSpacing)), Math.round(y + height));
      context.scale(1, -1);
      context.beginPath();
      context.rect(-barWidth / 2, 0, barWidth, barHeight);
      context.fillStyle = color;
      context.fill();
      context.restore();
     }
    }
    context.restore();
  </script>
</body>
</html>
```

- Output:



2. Read the data from a text file and draw simple bar chart.
   - Program:

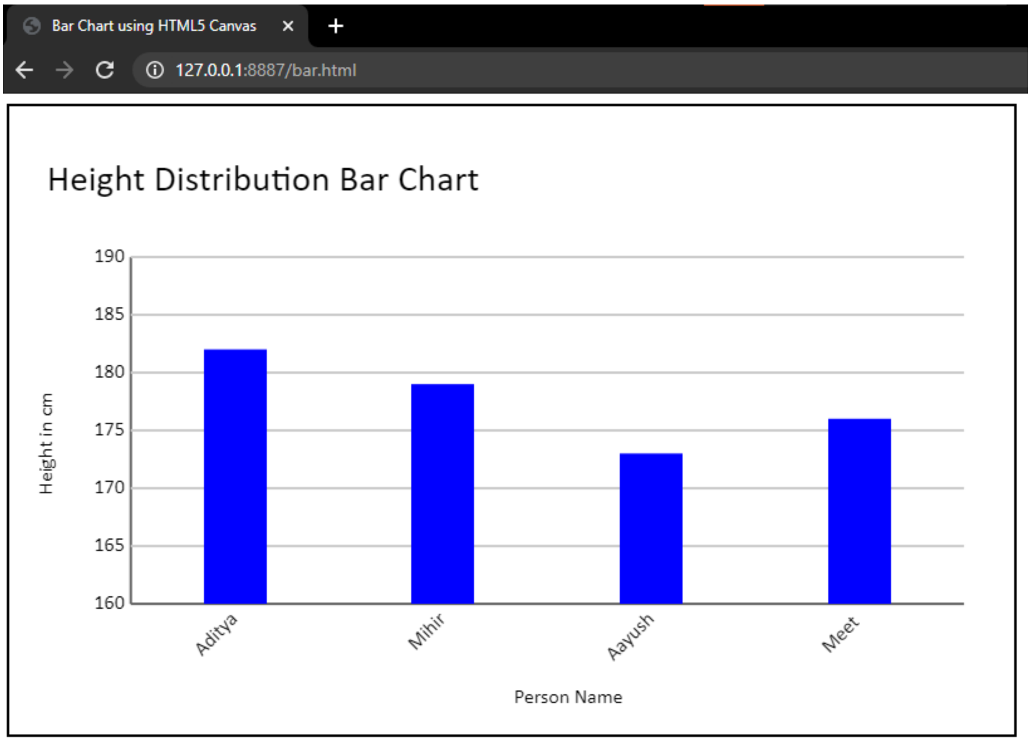
```
<html>
<head>
   <title>Bar Chart using HTML5 Canvas</title>
   <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
</head>
<body>
   <canvas width="800" height="500" style="border: 2px solid black;"/>
   <script>
      var label = [];
      var data = [];

      $.get('http://127.0.0.1:8887/barText.txt', function(content) {
         var lines = content.split('\n');
         label = lines[0].split(' ');
         data = lines[1].split(' ');
      }).done(function(){

            const canvas = document.querySelector('canvas');
            const color = "blue";
            const barWidth = 50;
            const step = 5;
            const max = 190;
            const maxValue = max - Math.floor(max % step);
            const minValue = 160;
            const title = "Height Distribution Bar Chart";
            const ytitle = "Height in cm";
            const xtitle = "Person Name";
            const font = "12pt Calibri";
            const axisColor = "#666";
            const gridColor = "#CCC";
            const padding = 30;

            const context = canvas.getContext("2d");
            const range = maxValue - minValue;
            const gridLines = Math.round(range / step);
            const maxWidth = context.measureText(maxValue - (gridLines/2 * step)).width;
            const x = padding + (maxWidth*4);
            const y = padding * 4;
            const width = canvas.width - (maxWidth + padding * 4);
            const height = canvas.height - (context.measureText(label[0]).width + padding * 6);

            //Title of the chart
            context.save();
            context.font = "22pt Calibri";
            context.fillStyle = "black";
            context.textAlign = "right";
```

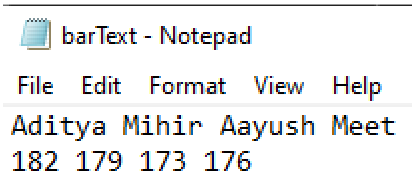
```
    context.textBaseline = "middle";
    context.fillText(title, context.measureText(title).width + padding, padding*2);
    context.restore();

    //Y axis line
    context.save();
    context.beginPath();
  context.moveTo(x, y);
    context.lineTo(x, height + y);
    context.strokeStyle = axisColor;
    context.lineWidth = 2;
    context.stroke();
    context.restore();

    //Y axis label
    context.save();
    context.font = font;
    context.fillStyle = "black";
    context.textAlign = "right";
    context.textBaseline = "middle";
    context.rotate(-1 * Math.PI / 2);
    context.fillText(ytitle, height - (context.measureText(title).width * 3 ) + padding*2,
padding);
    context.restore();

    //Y axis ticks
    context.save();
    context.font = font;
    context.fillStyle = "black";
    context.textAlign = "right";
    context.textBaseline = "middle";
    for (var n = 0; n <= gridLines; n++) {
      var value = maxValue - (n * step);
      var thisY = (n * height / gridLines) + y;
      context.fillText(value, x - 5, thisY);
    }
    context.restore();

    //X axis line
    context.save();
    context.beginPath();
    context.moveTo(x, y + height);
    context.lineTo(x + width, y + height);
    context.strokeStyle = axisColor;
    context.lineWidth = 2;
    context.stroke();
  context.restore();
```

```
    //X axis label
    context.save();
    context.font = font;
    context.fillStyle = "black";
    context.textAlign = "right";
    context.textBaseline = "middle";
    context.fillText(xtitle, x + width/2 + padding*2, y + height + padding*2.5);
    context.restore();

    //X axis ticks
    context.save();
    var barSpacing = width / label.length;
    for (var n = 0; n < label.length; n++) {
      var l = label[n];
      context.save();
      context.translate(x + ((n + 1 / 2) * barSpacing), y + height + 10);
      context.rotate(-1 * Math.PI / 4);
      context.font = font;
      context.fillStyle = "black";
      context.textAlign = "right";
      context.textBaseline = "middle";
      context.fillText(l, 0, 0);
      context.restore();
    }
    context.restore();

  //Inner Grid of the chart
    context.save();
    context.strokeStyle = gridColor;
    context.lineWidth = 2;
    for (var n = 0; n < gridLines; n++) {
      var ty = (n * height / gridLines) + y;
      context.beginPath();
      context.moveTo(x, ty);
      context.lineTo(x + width, ty);
      context.stroke();
    }
    context.restore();

  //Bars
    context.save();
    var barSpacing = width / data.length;
    var unitHeight = height / range;
    for (var n = 0; n < data.length; n++) {
      var barHeight = (data[n] - minValue) * unitHeight;
```

```
         if (barHeight > 0) {
          context.save();
          context.translate(Math.round(x + ((n + 1 / 2) * barSpacing)), Math.round(y + height));
          context.scale(1, -1);
          context.beginPath();
          context.rect(-barWidth / 2, 0, barWidth, barHeight);
          context.fillStyle = color;
          context.fill();
          context.restore();
         }
        }
        context.restore();
    });
  </script>
</body>
</html>
```

- Output:





```
Aditya Mihir Aayush Meet
182 179 173 176
```

3. Read the data from a csv file and draw simple column bar chart.
    - Program:

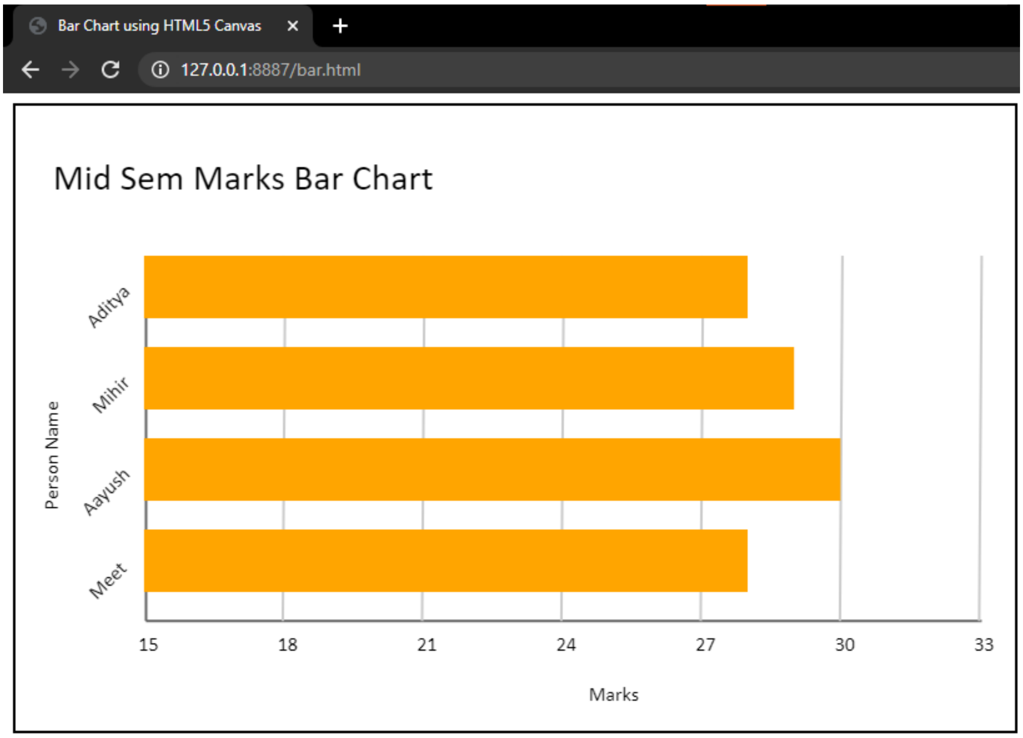
```
<html>
<head>
   <title>Bar Chart using HTML5 Canvas</title>
   <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
</head>
<body>
   <canvas width="800" height="500" style="border: 2px solid black;"/>
   <script>
      var label = [];
      var data = [];

     $.get('http://127.0.0.1:8887/barCSV.csv', function(content) {
         var lines = content.split('\n');
         label = lines[0].split(',');
         data = lines[1].split(',');
     }).done(function(){

            const canvas = document.querySelector('canvas');
            const color = "orange";
            const barHeight = 50;
            const step = 3;
            const max = 35;
            const maxValue = max - Math.floor(max % step);
            const minValue = 15;
            const title = "Mid Sem Marks Bar Chart";
            const xtitle = "Person Name";
            const ytitle = "Marks";
            const font = "12pt Calibri";
            const axisColor = "#666";
            const padding = 30;

            const context = canvas.getContext("2d");
            const range = maxValue - minValue;
            const gridLines = Math.round(range / step);
            const maxWidth = context.measureText(maxValue - (gridLines/2 * step)).width;
            const x = padding*2 + (maxWidth*4);
            const y = padding * 4;
            const width = canvas.width - (maxWidth + padding * 4);
            const height = canvas.height - (context.measureText(label[0]).width + padding * 6);

            //Title of the chart
            context.save();
            context.font = "22pt Calibri";
            context.fillStyle = "black";
            context.textAlign = "right";
            context.textBaseline = "middle";
```

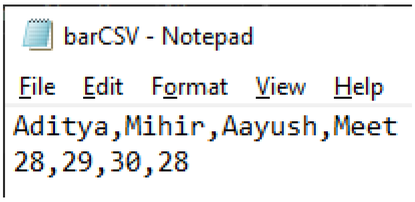
```
        context.fillText(title, context.measureText(title).width + padding, padding*2);
        context.restore();

        //X axis line
        context.save();
        context.beginPath();
      context.moveTo(x, y);
        context.lineTo(x, height + y);
        context.strokeStyle = axisColor;
        context.lineWidth = 2;
        context.stroke();
        context.restore();

        //X axis label
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
        context.rotate(-1 * Math.PI / 2);
        context.fillText(xtitle, height - (context.measureText(title).width*3) - padding,
padding);
        context.restore();

        //X axis ticks
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
      for (var n = 0; n<= gridLines; n++) {
          var value = minValue + (n * step);
          var thisX = (n * width / gridLines) + x;
          context.fillText(value, thisX + 10, y + height + 20);
        }
        context.restore();

        //Y axis line
        context.save();
        context.beginPath();
        context.moveTo(x, y + height);
        context.lineTo(x + width, y + height);
        context.strokeStyle = axisColor;
        context.lineWidth = 2;
        context.stroke();
      context.restore();
```

```
  //Y axis label
  context.save();
  context.font = font;
  context.fillStyle = "black";
  context.textAlign = "right";
  context.textBaseline = "middle";
  context.fillText(ytitle, x + width/2 + padding*2, y + height + padding*2);
  context.restore();

  //Y axis ticks
  context.save();
  var barSpacing = height / label.length;
for (var n = 0; n < label.length; n++) {
    var l = label[n];
    context.save();
    context.translate(x-15,y + ((n + 1 / 2) * barSpacing-10));
    context.rotate(-1 * Math.PI / 4);
    context.font = font;
    context.fillStyle = "black";
    context.textAlign = "right";
    context.textBaseline = "middle";
    context.fillText(l, 0, 0);
    context.restore();
  }
  context.restore();

//Inner Grid of the chart
  context.save();
  context.strokeStyle = gridColor;
  context.lineWidth = 2;
for (var n = 1; n <=gridLines; n++) {
    var tx = (n * width / gridLines) + x;
    context.beginPath();
    context.moveTo(tx, y);
    context.lineTo(tx-2, y + height);
    context.stroke();
  }
  context.restore();

//Bars
  context.save();
var barSpacing = height / data.length;
  var unitWidth = width / range;
  for (var n = 0; n < data.length; n++) {
    var barWidth = (data[n] - minValue) * unitWidth;
```

```
            if (barWidth > 0) {
              context.save();
              context.translate(x + 17,Math.round(y + 13 + ((n + 1 / 2) * barSpacing)));
              context.scale(1, -1);
              context.beginPath();
              context.rect(-unitWidth / 2, 0, barWidth, barHeight);
              context.fillStyle = color;
              context.fill();
              context.restore();
            }
          }
          context.restore();
      });
   </script>
</body>
</html>
```

- Output:





```
Aditya,Mihir,Aayush,Meet
28,29,30,28
```

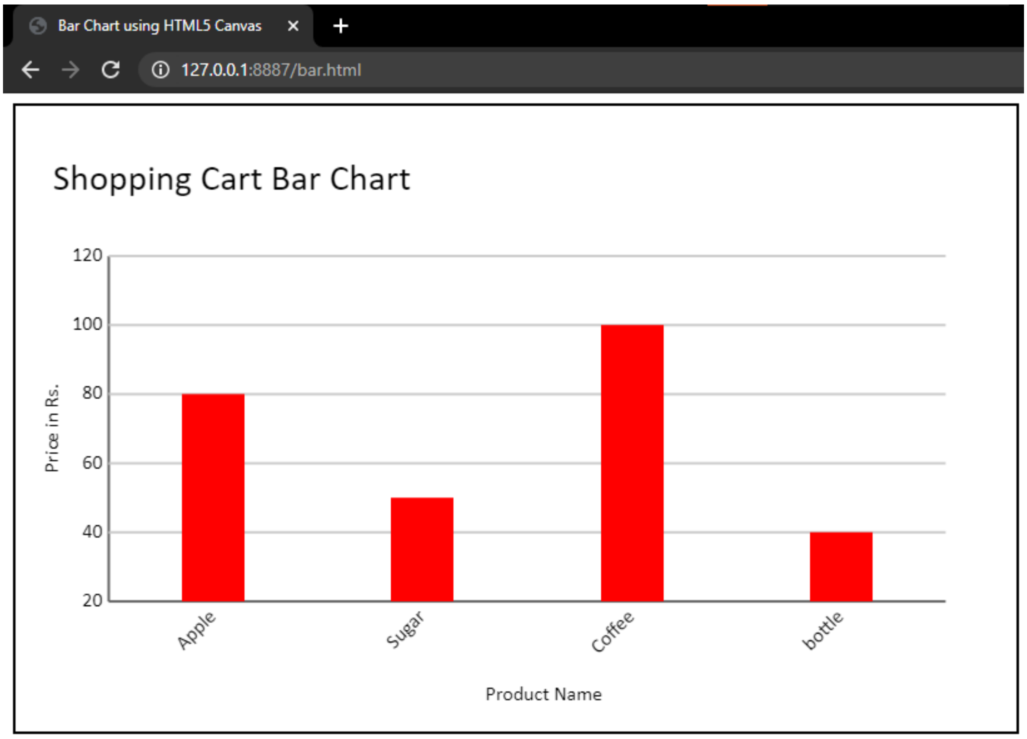
4. Read the data from a xml file and draw simple bar chart.
    - Program:

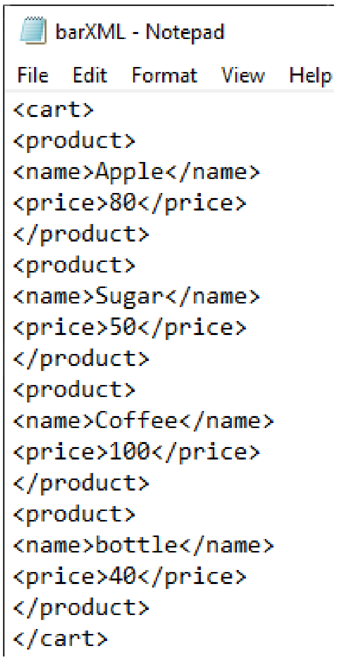
```
<html>
<head>
   <title>Bar Chart using HTML5 Canvas</title>
   <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
</head>
<body>
   <canvas width="800" height="500" style="border: 2px solid black;"/>
   <script>
      var label = [];
      var data = [];

     $.get('http://127.0.0.1:8887/barXML.xml', function(content) {
        var $products = $(content).find('product');
        $products.each(function(index){
                 label.push($(this).find('name').text());
                 data.push($(this).find('price').text());
           });
     }).done(function(){

           const canvas = document.querySelector('canvas');
           const color = "red";
           const barWidth = 50;
           const step = 20;
           const max = 120;
           const maxValue = max - Math.floor(max % step);
           const minValue = 20;
           const title = "Shopping Cart Bar Chart";
           const ytitle = "Price in Rs.";
           const xtitle = "Product Name";
           const font = "12pt Calibri";
           const axisColor = "#666";
           const gridColor = "#CCC";
           const padding = 30;

           const context = canvas.getContext("2d");
           const range = maxValue - minValue;
           const gridLines = Math.round(range / step);
           const maxWidth = context.measureText(maxValue - (gridLines/2 * step)).width;
           const x = padding + (maxWidth*4);
           const y = padding * 4;
           const width = canvas.width - (maxWidth + padding * 4);
           const height = canvas.height - (context.measureText(label[0]).width + padding * 6);

           //Title of the chart
           context.save();
           context.font = "22pt Calibri";
           context.fillStyle = "black";
           context.textAlign = "right";
           context.textBaseline = "middle";
           context.fillText(title, context.measureText(title).width + padding, padding*2);
           context.restore();

           //Y axis line
           context.save();
           context.beginPath();
        context.moveTo(x, y);
           context.lineTo(x, height + y);
           context.strokeStyle = axisColor;
           context.lineWidth = 2;
           context.stroke();
           context.restore();

           //Y axis label
           context.save();
           context.font = font;
           context.fillStyle = "black";
           context.textAlign = "right";
           context.textBaseline = "middle";
           context.rotate(-1 * Math.PI / 2);
     context.fillText(ytitle, height - (context.measureText(title).width * 3 ) - padding, padding);
           context.restore();

           //Y axis ticks
           context.save();
           context.font = font;
           context.fillStyle = "black";
           context.textAlign = "right";
           context.textBaseline = "middle";
           for (var n = 0; n <= gridLines; n++) {
             var value = maxValue - (n * step);
             var thisY = (n * height / gridLines) + y;
             context.fillText(value, x - 5, thisY);
           }
           context.restore();

           //X axis line
           context.save();
           context.beginPath();
           context.moveTo(x, y + height);
           context.lineTo(x + width, y + height);
           context.strokeStyle = axisColor;
           context.lineWidth = 2;
           context.stroke();
           context.restore();
```

```
        //X axis label
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
        context.fillText(xtitle, x + width/2 + padding*2, y + height + padding*2.5);
        context.restore();

        //X axis ticks
        context.save();
        var barSpacing = width / label.length;
        for (var n = 0; n < label.length; n++) {
          var l = label[n];
          context.save();
          context.translate(x + ((n + 1 / 2) * barSpacing), y + height + 10);
          context.rotate(-1 * Math.PI / 4);
          context.font = font;
          context.fillStyle = "black";
          context.textAlign = "right";
          context.textBaseline = "middle";
          context.fillText(l, 0, 0);
          context.restore();
        }
        context.restore();

    //Inner Grid of the chart
        context.save();
        context.strokeStyle = gridColor;
        context.lineWidth = 2;
        for (var n = 0; n < gridLines; n++) {
          var ty = (n * height / gridLines) + y;
          context.beginPath();
          context.moveTo(x, ty);
          context.lineTo(x + width, ty);
          context.stroke();
        }
        context.restore();

    //Bars
        context.save();
        var barSpacing = width / data.length;
        var unitHeight = height / range;
        for (var n = 0; n < data.length; n++) {
          var barHeight = (data[n] - minValue) * unitHeight;

          if (barHeight > 0) {
            context.save();
```

```
                    context.translate(Math.round(x + ((n + 1 / 2) * barSpacing)), Math.round(y +
height));
                    context.scale(1, -1);
                    context.beginPath();
                    context.rect(-barWidth / 2, 0, barWidth, barHeight);
                    context.fillStyle = color;
                    context.fill();
                    context.restore();
                  }
                }
                context.restore();
        });
    </script>
</body>
</html>
```

- Output:





```
<cart>
<product>
<name>Apple</name>
<price>80</price>
</product>
<product>
<name>Sugar</name>
<price>50</price>
</product>
<product>
<name>Coffee</name>
<price>100</price>
</product>
<product>
<name>bottle</name>
<price>40</price>
</product>
</cart>
```

5. Read the data from a json file and draw simple bar chart.
    - Program:

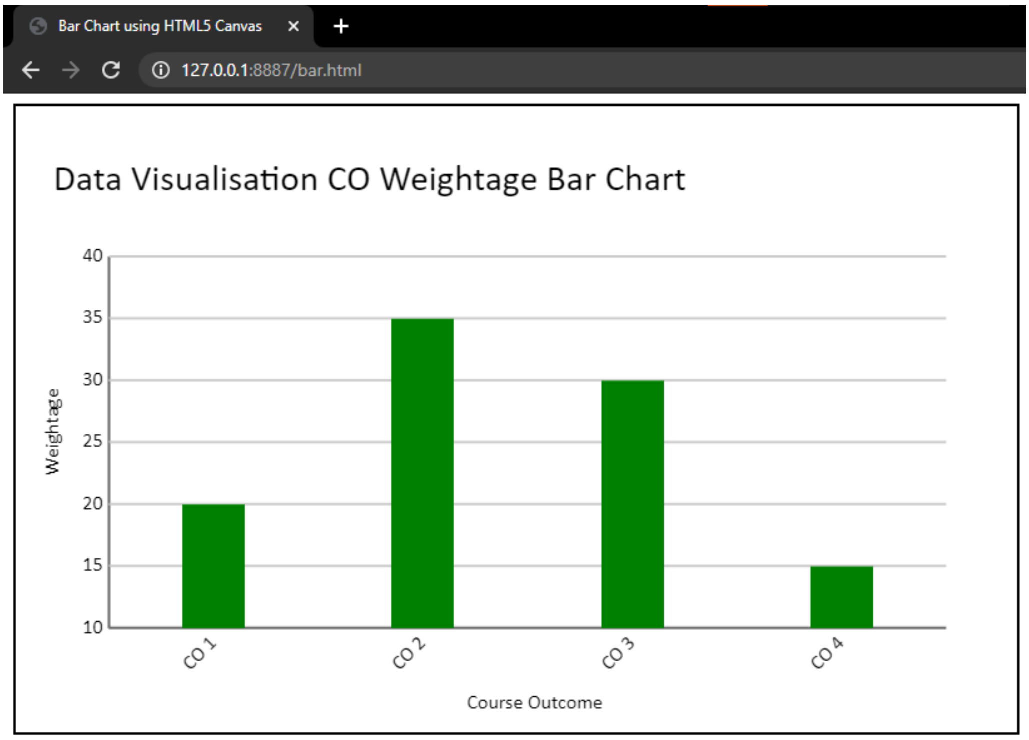
```
<html>
<head>
   <title>Bar Chart using HTML5 Canvas</title>
   <script src="https://code.jquery.com/jquery-3.5.1.js"></script>
</head>
<body>
   <canvas width="800" height="500" style="border: 2px solid black;"/>
   <script>
      var label = [];
      var data = [];

      $.getJSON('http://127.0.0.1:8887/barJSON.json', function(content) {
         $.each(content,function(k,v){
                  label.push(v.label);
                  data.push(v.value);
            });
      }).done(function(){

            const canvas = document.querySelector('canvas');
            const color = "green";
            const barWidth = 50;
            const step = 5;
            const max = 40;
            const maxValue = max - Math.floor(max % step);
            const minValue = 10;
            const title = "Data Visualisation CO Weightage Bar Chart";
            const ytitle = "Weightage";
            const xtitle = "Course Outcome";
            const font = "12pt Calibri";
            const axisColor = "#666";
            const gridColor = "#CCC";
            const padding = 30;

            const context = canvas.getContext("2d");
            const range = maxValue - minValue;
            const gridLines = Math.round(range / step);
            const maxWidth = context.measureText(maxValue - (gridLines/2 * step)).width;
            const x = padding + (maxWidth*4);
            const y = padding * 4;
            const width = canvas.width - (maxWidth + padding * 4);
            const height = canvas.height - (context.measureText(label[0]).width + padding * 6);

            //Title of the chart
            context.save();
            context.font = "22pt Calibri";
            context.fillStyle = "black";
```

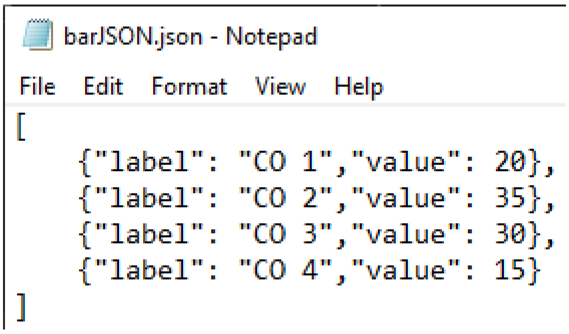
```
        context.textAlign = "right";
        context.textBaseline = "middle";
        context.fillText(title, context.measureText(title).width + padding, padding*2);
        context.restore();

        //Y axis line
        context.save();
        context.beginPath();
      context.moveTo(x, y);
        context.lineTo(x, height + y);
        context.strokeStyle = axisColor;
        context.lineWidth = 2;
        context.stroke();
        context.restore();

        //Y axis label
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
        context.rotate(-1 * Math.PI / 2);
        context.fillText(ytitle, height - (context.measureText(title).width*2) + padding,
padding);
        context.restore();

        //Y axis ticks
        context.save();
        context.font = font;
        context.fillStyle = "black";
        context.textAlign = "right";
        context.textBaseline = "middle";
        for (var n = 0; n <= gridLines; n++) {
          var value = maxValue - (n * step);
          var thisY = (n * height / gridLines) + y;
          context.fillText(value, x - 5, thisY);
        }
        context.restore();

        //X axis line
        context.save();
        context.beginPath();
        context.moveTo(x, y + height);
        context.lineTo(x + width, y + height);
        context.strokeStyle = axisColor;
        context.lineWidth = 2;
        context.stroke();
        context.restore();
```

```
    //X axis label
    context.save();
    context.font = font;
    context.fillStyle = "black";
    context.textAlign = "right";
    context.textBaseline = "middle";
    context.fillText(xtitle, x + width/2 + padding*2, y + height + padding*2);
    context.restore();

    //X axis ticks
    context.save();
    var barSpacing = width / label.length;
    for (var n = 0; n < label.length; n++) {
      var l = label[n];
      context.save();
      context.translate(x + ((n + 1 / 2) * barSpacing), y + height + 10);
      context.rotate(-1 * Math.PI / 4);
      context.font = font;
      context.fillStyle = "black";
      context.textAlign = "right";
      context.textBaseline = "middle";
      context.fillText(l, 0, 0);
      context.restore();
    }
    context.restore();

//Inner Grid of the chart
    context.save();
    context.strokeStyle = gridColor;
    context.lineWidth = 2;
    for (var n = 0; n < gridLines; n++) {
      var ty = (n * height / gridLines) + y;
      context.beginPath();
      context.moveTo(x, ty);
      context.lineTo(x + width, ty);
      context.stroke();
    }
    context.restore();

//Bars
    context.save();
    var barSpacing = width / data.length;
    var unitHeight = height / range;
    for (var n = 0; n < data.length; n++) {
      var barHeight = (data[n] - minValue) * unitHeight;
```

```
                if (barHeight > 0) {
                  context.save();
                  context.translate(Math.round(x + ((n + 1 / 2) * barSpacing)), Math.round(y + height));
                  context.scale(1, -1);
                  context.beginPath();
                  context.rect(-barWidth / 2, 0, barWidth, barHeight);
                  context.fillStyle = color;
                  context.fill();
                  context.restore();
                }
              }
              context.restore();
      });
   </script>
</body>
</html>
```

- Output:





```
[
    {"label": "CO 1","value": 20},
    {"label": "CO 2","value": 35},
    {"label": "CO 3","value": 30},
    {"label": "CO 4","value": 15}
]
```