PROJECT REPORT

NAME: ROHITH KRISHNA

REGNO:18BCE0537

1.INTRODUCTION :

Education is the basic requirement for human development. With education, employment opportunities are broadened and income levels are increased. The development of an individual and the progress of a nation depend on education.

Proficiency and level of training are fundamental pointers of the level of improvement accomplished by a general public.

Spread of literacy is by and large connected with vital attributes of present day development for example, modernization, urbanization, trade and industrialization.

Literacy shapes a vital contribution to generally improvement of society empowering them to understand their social, political and social condition better and react to it appropriately

Hence we need to properly analyze such important criteria and provide proper picture to the audience so as to ensure that any issues that rise can be taken care of.

2.PROBLEM :

We in the Course of study are going to depict the Statistical data collected in the form of visuals so as to ensure that the target audience are given a clear cut picture and are able to exactly infer proper facts from the data.

Main Aim is to create a project-specific dashboard and visualizations to aid in decision making to **target audience. The Governmental agencies** and the education motto driven organizations like **Make A Difference** (MAD) **foundation** and **Teach for India** (TFI).

We are going to analyze literacy rate in India across the states using interactive visual techniques and visualizing the data in the form of informative plots.

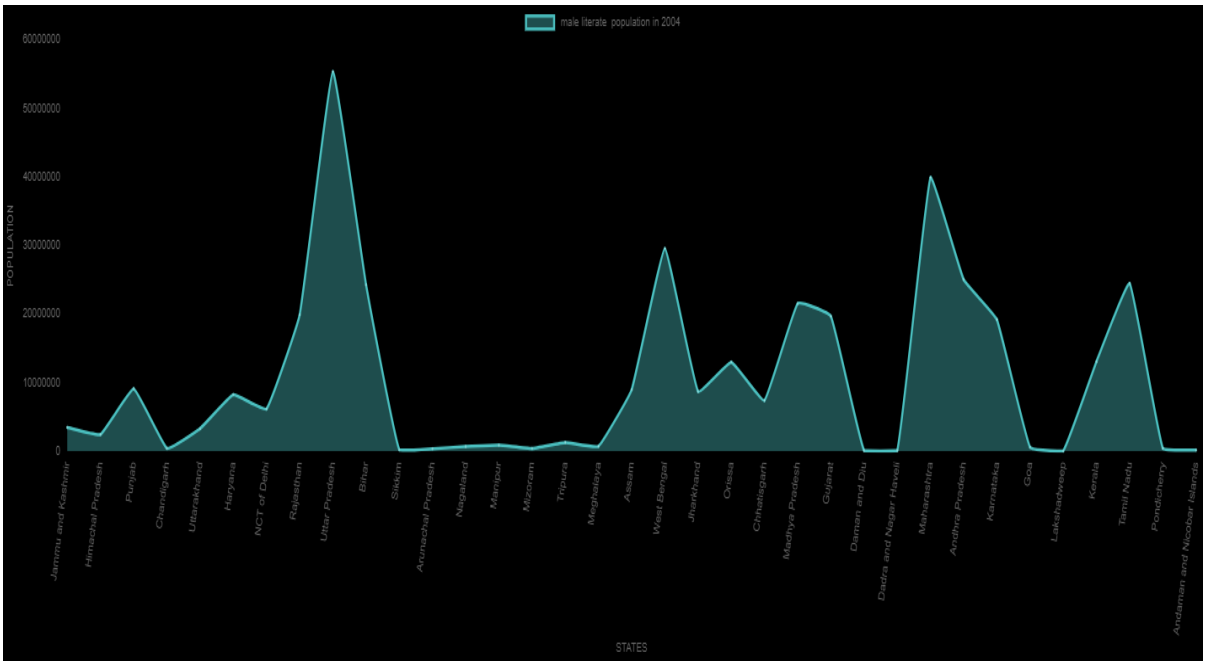
3.DATA COLLECTION :

The dataset consists of attributes: **Year**, **States**, **Male Literate Population**, **Female Literate Population**, **Total Literate Population**, **Male Literacy Rate**, **Female Literacy Rate**, **and Total Literate Population**.

They were collected from various websites like data.gov.in, mospi.nic.in/ and other surveys conducted by various authorities, collected and stored in the form of csv in an year-wise manner

4.TASKS:

1. Single line chart depicting attribute chosen by the user



CODE:

```
<script type="text/javascript">
  var v= {{p|safe}}
  var h= {{l|safe}}
  var j= {{k|safe}}
  var l={{label|safe}}
  Chart.defaults.global.responsive = false;
 //single line chart
     var chartData =
     {
       labels : [{% for item in label %}
                  "{{item}}",
                 {% endfor %}],
       datasets : [{
           label: 'Relative growth graph depicted between  {{at1}} in 200{{a}
} and {{at2}} in 200{{b}}',
           fill: true,
           lineTension: 0.1,
           backgroundColor: "rgba(75,192,192,0.4)",
           borderColor: "rgba(75,192,192,1)",
           borderCapStyle: 'butt',
           borderDash: [],
           borderDashOffset: 0.0,
           borderJoinStyle: 'miter',
```

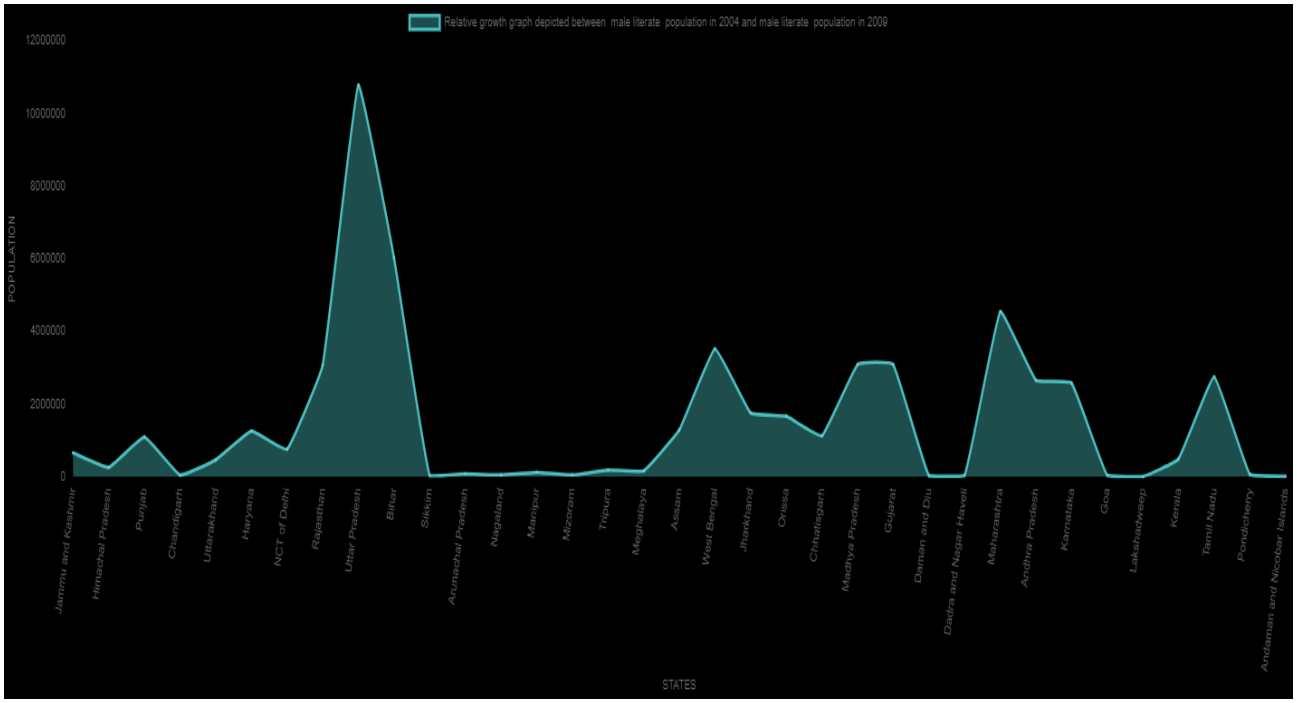
```
            pointBorderColor: "rgba(75,192,192,1)",
            pointBackgroundColor: "#fff",
            pointBorderWidth: 1,
            pointHoverRadius: 5,
            pointHoverBackgroundColor: "rgba(75,192,192,1)",
            pointHoverBorderColor: "rgba(220,220,220,1)",
            pointHoverBorderWidth: 2,
            pointRadius: 1,
            pointHitRadius: 10,
            data :v,
            spanGaps: false
          }]
    };
var option={
  scales: {
      xAxes: [{
        ticks: {
          maxRotation: 90,
          minRotation: 80
        },
        scaleLabel:{
          display:true,
          labelString:"STATES"
        }
      }],
      yAxes: [{
        ticks: {
          beginAtZero: true
        },
        scaleLabel:{
          display:true,
          labelString:"POPULATION"
        }
      }]
    }
}
  // get chart canvas
      var ctx = document.getElementById("myChart1").getContext("2d");
// create the chart using the chart canvas
      var myChart = new Chart(ctx, {
        type: 'line',
        data: chartData,
        options:option
      });
// next
var chartData2 =
      {
        labels : [{% for item in label %}
```

```
                    "{{item}}",
                  {% endfor %}],
        datasets : [{
            label: ' {{at1}} in 200{{a}}',
            fill: true,
            lineTension: 0.1,
            backgroundColor: "rgba(75,192,192,0.4)",
            borderColor: "rgba(75,192,192,1)",
            borderCapStyle: 'butt',
            borderDash: [],
            borderDashOffset: 0.0,
            borderJoinStyle: 'miter',
            pointBorderColor: "rgba(75,192,192,1)",
            pointBackgroundColor: "#fff",
            pointBorderWidth: 1,
            pointHoverRadius: 5,
            pointHoverBackgroundColor: "rgba(75,192,192,1)",
            pointHoverBorderColor: "rgba(220,220,220,1)",
            pointHoverBorderWidth: 2,
            pointRadius: 1,
            pointHitRadius: 10,
            data :j,
            spanGaps: false
          }]
    };
// get chart canvas
      var ctx2 = document.getElementById("myChart2").getContext("2d");
// create the chart using the chart canvas
      var myChart2 = new Chart(ctx2, {
        type: 'line',
        data: chartData2,
        options:option
      });
// next
var chartData3 =
      {
        labels : [{% for item in label %}
                    "{{item}}",
                  {% endfor %}],
        datasets : [{
            label: '{{at2}} in 200{{b}}',
            fill: true,
            lineTension: 0.1,
            backgroundColor: "rgba(75,192,192,0.4)",
            borderColor: "rgba(75,192,192,1)",
            borderCapStyle: 'butt',
            borderDash: [],
            borderDashOffset: 0.0,
```

```
            borderJoinStyle: 'miter',
            pointBorderColor: "rgba(75,192,192,1)",
            pointBackgroundColor: "#fff",
            pointBorderWidth: 1,
            pointHoverRadius: 5,
            pointHoverBackgroundColor: "rgba(75,192,192,1)",
            pointHoverBorderColor: "rgba(220,220,220,1)",
            pointHoverBorderWidth: 2,
            pointRadius: 1,
            pointHitRadius: 10,
            data :h,
            spanGaps: false
          }]
    };
 // get chart canvas
 var ctx3 = document.getElementById("myChart3").getContext("2d");
  // create the chart using the chart canvas
  var myChart3 = new Chart(ctx3, {
        type: 'line',
        data: chartData,
        options:option
      });
</script>
```

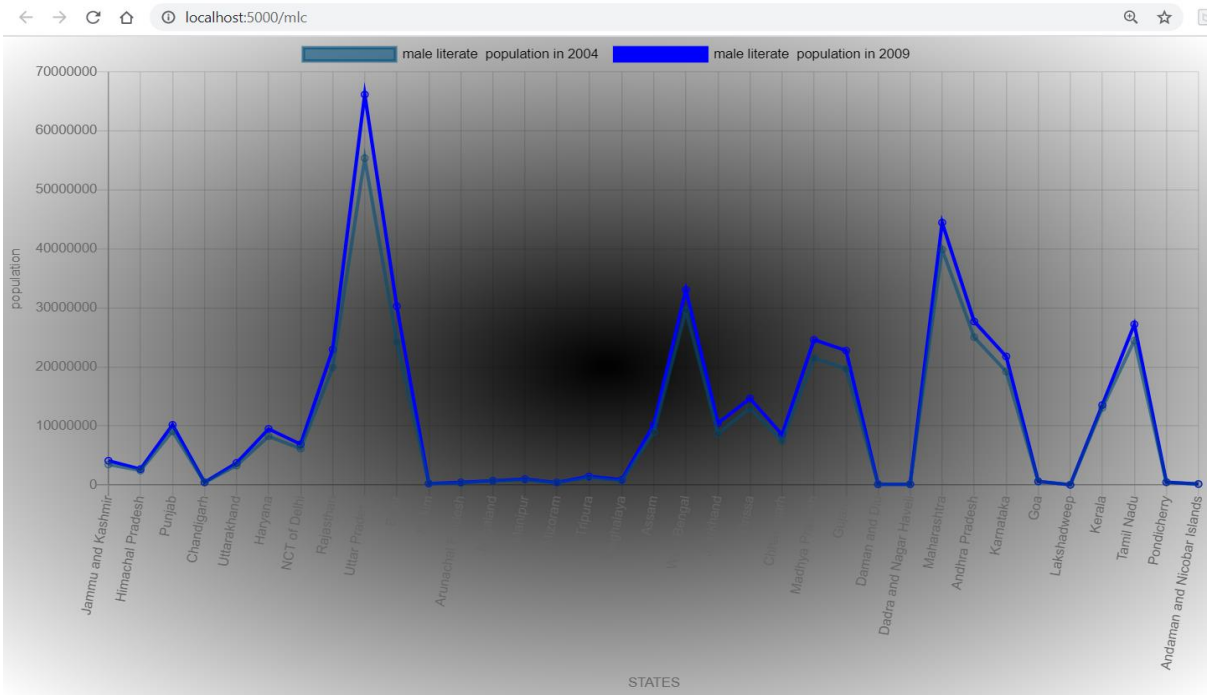
Respective labels for the axes are also shown with legend.

2. Single Line chart of relative difference between attributes chosen between selected two years



Respective labels for the axes are also shown with legend

3. Multi Line chart of the attributes selected and year acting as differentiating factor respectively



Labels for the axes and the legends respectively are shown

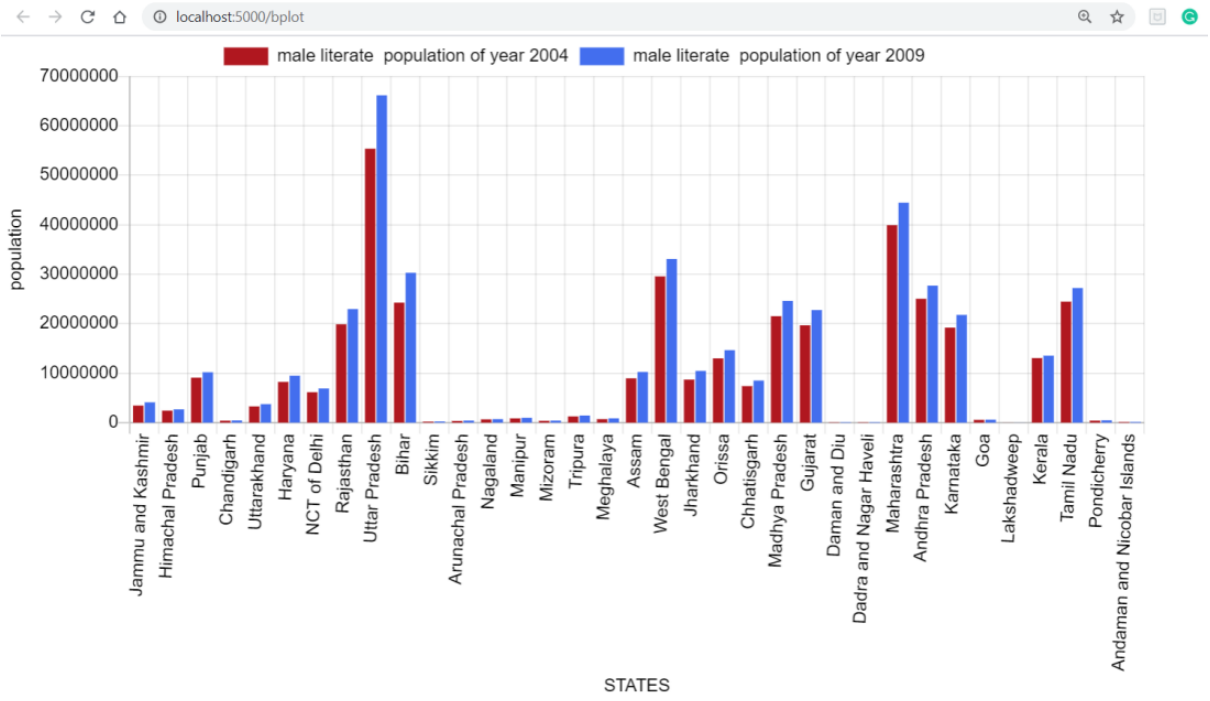
CODE:

```
<script type="text/javascript">

  var v= {{p|safe}}
  var h= {{l|safe}}
  var j= {{k|safe}}
  var l={{label|safe}}
  Chart.defaults.global.responsive = false



  var mc2 = document.getElementById("myChart2").getContext("2d");


var dataFirst = {
    label: "{{at1}} in 200{{a}} ",
    data: j,
    lineTension: 0,
    fill: false,
    borderColor: "rgba(0,75,120,.6)"
  };
```

```
var dataSecond = {
    label: "{{at2}} in 200{{b}}",
    data: h,
    lineTension: 0,
    fill: false,
  borderColor: 'blue'
  };

var sData = {
  labels: l ,
  datasets: [dataFirst, dataSecond]
};

var chartOptions = {
  legend: {
    display: true,
    position: 'top',
    labels: {
      boxWidth: 80,
      fontColor: 'black'
    }
  },
  scales: {
      xAxes: [{
        ticks: {
          maxRotation: 90,
          minRotation: 80
        },
        scaleLabel:{
          display:true,
          labelString:"STATES"
        }
      }],
      yAxes: [{
        ticks: {
          beginAtZero: true
        },
        scaleLabel:{
          display:true,
          labelString:"population"
        }
      }]
    }
};

var lineChart = new Chart(mc2, {
  type: 'line',
  data: sData,
```

```
  options: chartOptions
});
</script>
```

4.Multi stack chart depicting both the attributes chosen by the user



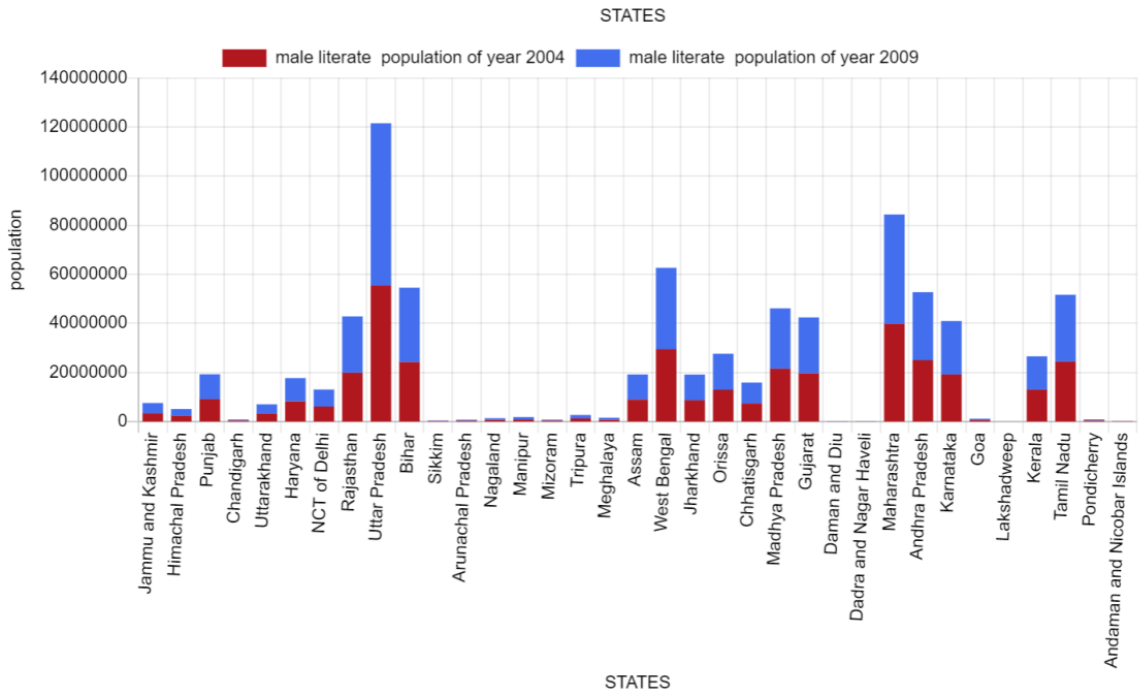
Axes are labelled and the legend is provided as shown

CODE:

```
<script type="text/javascript">
    var v= {{p|safe}}
    var h= {{l|safe}}
    var j= {{k|safe}}
    var l={{label|safe}}

var c1=document.getElementById("myChart1");
var ct1=c1.getContext("2d");

Chart.defaults.global.defaultFontColor = 'black';
Chart.defaults.global.defaultFontSize = 16;

var d1={
    label:"{{at1}} of year 200{{a}}",
    borderColor:"rgba(122,75,32,.4)",
    data : j,
    backgroundColor:'#B0171F',
```

```
}

var d2={
    label:"{{at2}} of year 200{{b}}",
    borderColor:"rgba(122,75,32,.4)",
    data:h,
    backgroundColor:'#436EEE',

}

var fd={
    datasets:[d1,d2],
    labels:l
};

var options={
    responsive: false,
    scales: {
      xAxes: [{
        ticks: {
          maxRotation: 90,
          minRotation: 80
        },
        scaleLabel:{
          display:true,
          labelString:"STATES"
        }
      }],
      yAxes: [{
        ticks: {
          beginAtZero: true
        },
        scaleLabel:{
          display:true,
          labelString:"population"
        }
      }]
    }
  };

var ch= new Chart(ct1,{
    type:"bar",
    data:fd,
    options:options
})
```

5. Stacked chart of the attributes chosen by the user



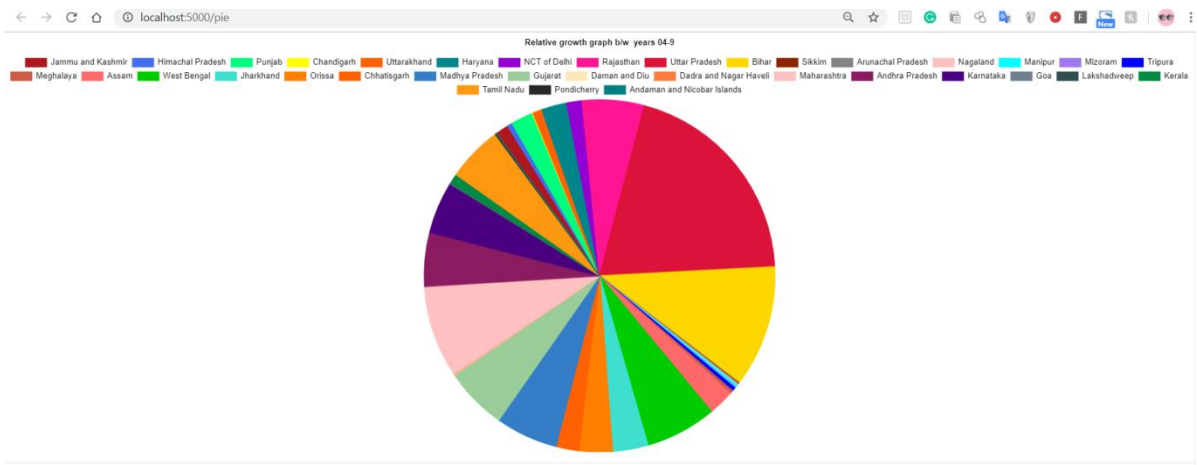
Axes are labelled and the legend is provided as shown

CODE:

```
<script type="text/javascript">
    var v= {{p|safe}}
    var h= {{l|safe}}
    var j= {{k|safe}}
    var l={{label|safe}}

c2=document.getElementById("myChart2");
ct2=c2.getContext('2d');

var sfd={
  dataset:[d1,d2],
  label:l
};


var opt2={
    responsive: false,
    scales: {
      xAxes: [{stacked:true,
```

```
        ticks: {
          maxRotation: 90,
          minRotation: 80
        },
        scaleLabel:{
          display:true,
          labelString:"STATES"
        }
      }],
      yAxes: [{stacked:true,
        ticks: {
          beginAtZero: true
        },
        scaleLabel:{
          display:true,
          labelString:"population"
        }
      }]
    }
  };
var ch2= new Chart(ct2,{

 type:'bar',

  data:fd,
  options:opt2
});
</script>
```

6. Pie chart of relative difference between attributes chosen



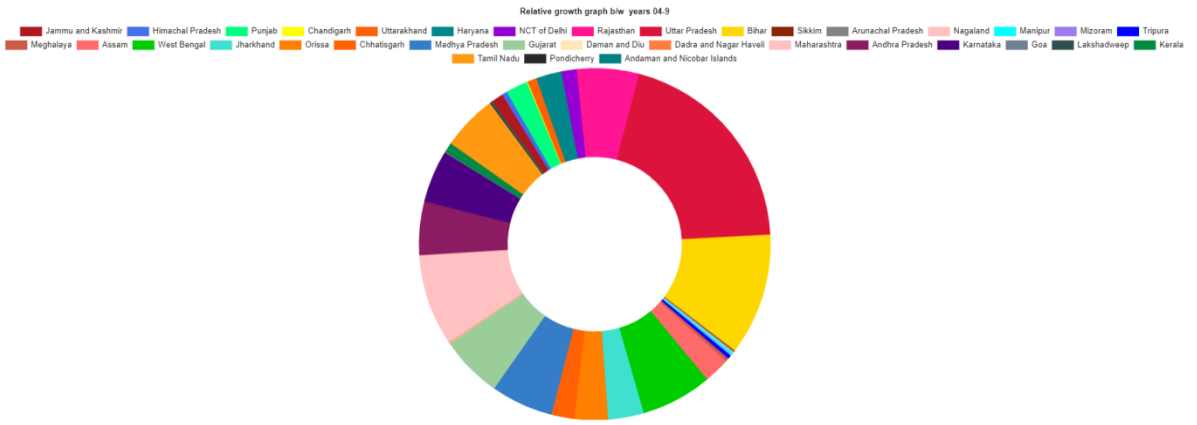
Legend is given as required

CODE:

```
<script type="text/javascript">
       var v= {{p|safe}}
       var h= {{l|safe}}
       var j= {{k|safe}}
       var l={{label|safe}}

       var canvas = document.getElementById("myChart1");
       var ctx = canvas.getContext('2d');
// Global Options:
       Chart.defaults.global.defaultFontColor = 'black';
       Chart.defaults.global.defaultFontSize = 16;

         var data = {
          labels: l,
           datasets: [
                        {
                fill: true,
                backgroundColor:['#B0171F','#436EEE','#00FF7F','#FFFF00','#FF6
103','#00868B','#9400D3','#FF1493','#DC143C','#FFD700','#8B2500','#848484','#F
FC1C1','#00FFFF','#9F79EE','#0000FF','#CD5B45','#FF6A6A','#00CD00','#40E0D0','
#FF7F00','#FF6103','#357EC7','#9BCD9B','#FFE7BA','#FF7D40','#FFC1C1','#8B1C62'
,'#4B0082','#708090','#2F4F4F','#008B45','#FF9912','#282828','#008080'] ,
                data: v,

                borderColor:['#B0171F','#436EEE','#00FF7F','#FFFF00','#FF6103'
,'#00868B','#9400D3','#FF1493','#DC143C','#FFD700','#8B2500','#848484','#FFC1C
1','#00FFFF','#9F79EE','#0000FF','#CD5B45','#FF6A6A','#00CD00','#40E0D0','#FF7
F00','#FF6103','#357EC7','#9BCD9B','#FFE7BA','#FF7D40','#FFC1C1','#8B1C62','#4
B0082','#708090','#2F4F4F','#008B45','#FF9912','#282828','#008080'],
                borderWidth: [2,2]
        }
    ]
};
var options = {
        title: {
                  display: true,
                  text: 'Relative growth graph b/w  years 0{{a}}-{{b}}',
                  position: 'top'
              },
        rotation: -0.7 * Math.PI
};
// Chart declaration:
var myBarChart = new Chart(ctx, {
    type: 'pie',
```

```
    data: data,
    options: options
});
```

7. doughnut chart of relative difference between attributes chosen



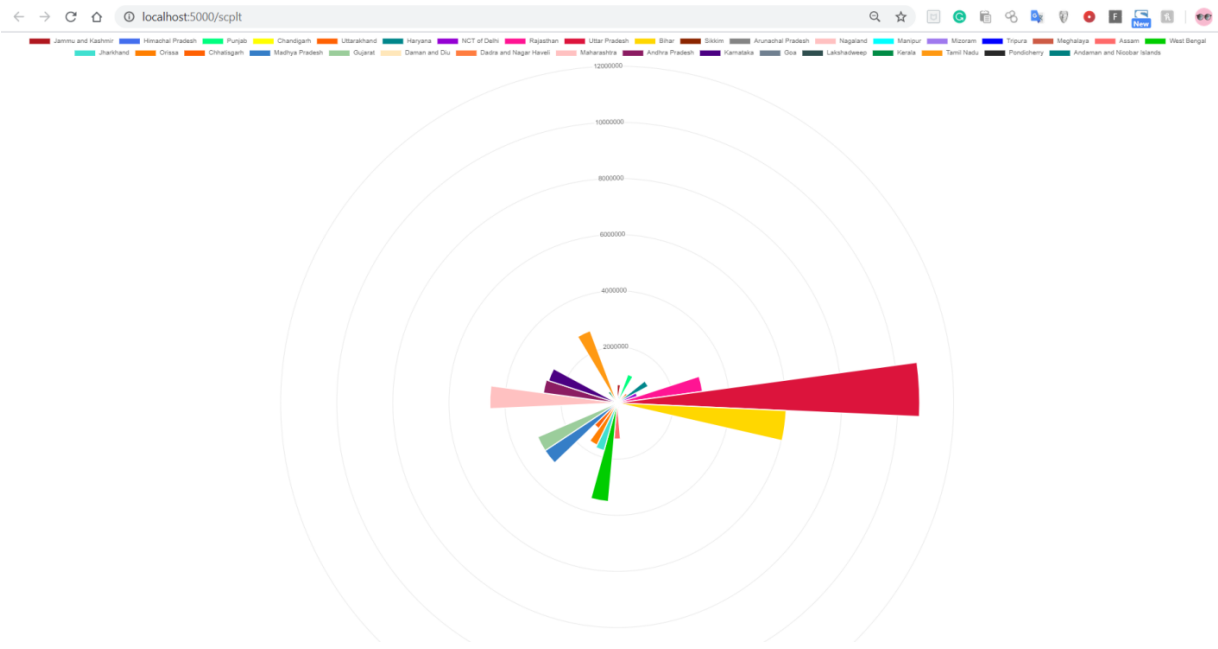
Legend is shown as required

CODE:

```
 <script type="text/javascript">
       var v= {{p|safe}}
       var h= {{l|safe}}
       var j= {{k|safe}}
       var l={{label|safe}}
// doughnut chart
var c2 = document.getElementById("myChart2");
var ct2 = c2.getContext('2d');
// Global Options:
 Chart.defaults.global.defaultFontColor = 'black';
 Chart.defaults.global.defaultFontSize = 16;
var data = {
    labels: l,
      datasets: [
        {
            fill: true,
            backgroundColor:['#B0171F','#436EEE','#00FF7F','#FFFF00','#FF6103'
,'#00868B','#9400D3','#FF1493','#DC143C','#FFD700','#8B2500','#848484','#FFC1C
1','#00FFFF','#9F79EE','#0000FF','#CD5B45','#FF6A6A','#00CD00','#40E0D0','#FF7
F00','#FF6103','#357EC7','#9BCD9B','#FFE7BA','#FF7D40','#FFC1C1','#8B1C62','#4
B0082','#708090','#2F4F4F','#008B45','#FF9912','#282828','#008080'] ,
            data: v,

            borderColor:['#B0171F','#436EEE','#00FF7F','#FFFF00','#FF6103','#0
0868B','#9400D3','#FF1493','#DC143C','#FFD700','#8B2500','#848484','#FFC1C1','
#00FFFF','#9F79EE','#0000FF','#CD5B45','#FF6A6A','#00CD00','#40E0D0','#FF7F00'
```

```
,'#FF6103','#357EC7','#9BCD9B','#FFE7BA','#FF7D40','#FFC1C1','#8B1C62','#4B008
2','#708090','#2F4F4F','#008B45','#FF9912','#282828','#008080'],
            borderWidth: [2,2]
        }
    ]
};
var options = {
        title: {
                    display: true,
                    text: 'Relative growth graph b/w  years 0{{a}}-{{b}}',
                    position: 'top'
                },
        rotation: -0.7 * Math.PI
};
// Chart declaration:
var myBarChart = new Chart(ct2, {
    type: 'doughnut',
    data: data,
    options: options
});
</script>
```

8. Polar area chart to depict the relative difference between attributes chosen



Legend is shown as required

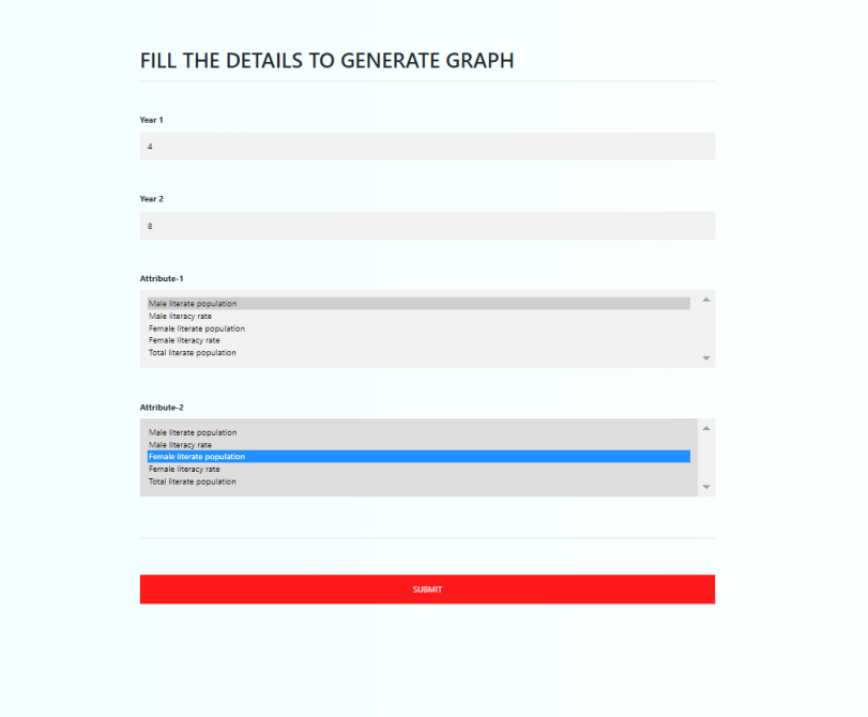
CODE:

```
<script type="text/javascript">
    var v= {{p|safe}}
  var h= {{l|safe}}
  var j= {{k|safe}}
```

```
   var l={{label|safe}}
   var c1=document.getElementById("myChart1");
   var ct1=c1.getContext("2d");
var data = {
    datasets: [{
        data: v,
        backgroundColor: ['#B0171F','#436EEE','#00FF7F','#FFFF00','#FF6103','#
00868B','#9400D3','#FF1493','#DC143C','#FFD700','#8B2500','#848484','#FFC1C1',
'#00FFFF','#9F79EE','#0000FF','#CD5B45','#FF6A6A','#00CD00','#40E0D0','#FF7F00
','#FF6103','#357EC7','#9BCD9B','#FFE7BA','#FF7D40','#FFC1C1','#8B1C62','#4B00
82','#708090','#2F4F4F','#008B45','#FF9912','#282828','#008080'
        ],
       label: 'Polar area chart depicting growth rate' // for legend
    }],
    labels:l
};
var ctx = $("#myChart");
new Chart(ct1, {
    data: data,
    type: 'polarArea'
});
</script>
</body>
</html>
```

5. ACTIONS :

1. Creating a dashboard where the user is given the choice of the database and the attributes which the user wants to analyze



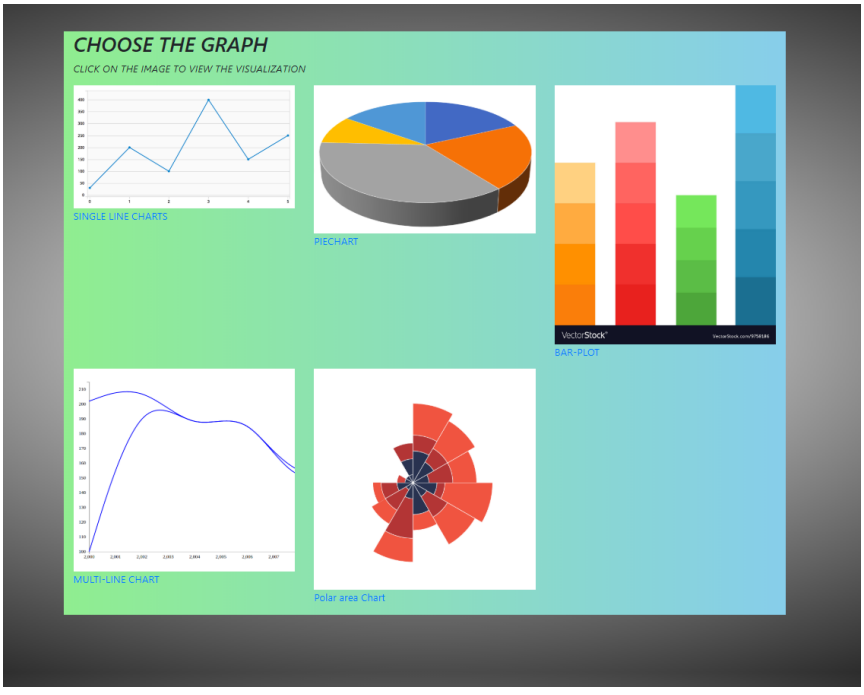
CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>

<meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>data</title>
    <script type="text/javascript" src="{{ url_for('static',filename='jquery-
2.1.4.js')}}"></script>
    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstr
ap.min.css" rel="stylesheet" integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin=
"anonymous">
    <link rel="stylesheet" href="{{ url_for('static',filename='data.css')}}">
   </head>
<nav class="navbar navbar-default">
</nav>
<form action = "http://localhost:5000/result" method = "POST">
      <div class="container">
        <h1>FILL THE DETAILS TO GENERATE GRAPH</h1>
       <hr>

        <label for="dataset-1"><b>Year 1</b></label>
      <p><input type="number" placeholder="1-11" name="dataset-
1"  min="1" max="11" required ></p>
        <br>

        <label for="dataset-2"><b>Year 2</b></label>
        <p><input type="number" placeholder="1-11" name="dataset-
2" min="1" max="11" required></p>
    <br>

        <span><b> Attribute-1 </b></span>

     <p> <select name="Attribute-1" size="6">
          <option value="1">Male literate population</option>
          <option value="2">Male literacy rate</option>
          <option value="3">Female literate population</option>
          <option value="4">Female literacy rate</option>
          <option value="5">Total literate population</option>
        </select>
       </p>
        <br>
```

```
        <span><b>Attribute-2</b></span>

       <p><select name="Attribute-2" size="6">
            <option value="1">Male literate population</option>
            <option value="2">Male literacy rate</option>
            <option value="3">Female literate population</option>
            <option value="4">Female literacy rate</option>
            <option value="5">Total literate population</option>
          </select>
    </p>
<br>
        <hr>

        <button type="submit" class="registerbtn">SUBMIT</button>
      </div>

          </form>

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

2. After the query is processed, we give the user the type of visualization based upon his need,

So that he can infer the task at hand from the result of the visualization , where on clicking the icon you are redirected to a new page



Once one of the image is clicked , the respective visualization opens in a new tab

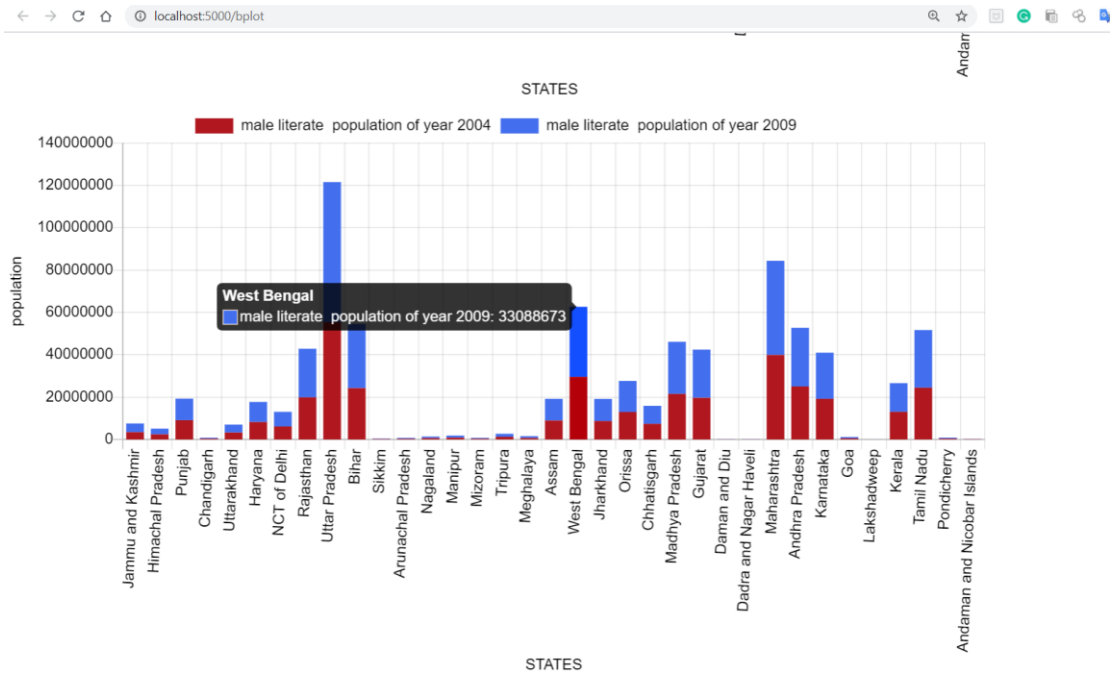
CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>result</title>
    <script type="text/javascript" src="{{url_for('static',filename='jquery-
2.1.4.js')}}"></script>
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs
/Chart.js/2.9.3/Chart.min.js"></script>
    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstr
ap.min.css" rel="stylesheet" integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin=
"anonymous">
     <link rel="stylesheet" href="{{url_for('static',filename='result.css')}}"
>
</head>

<body>
    <div class="container">
  <h2><i>CHOOSE THE GRAPH</i></h2>
  <p><i>CLICK ON THE IMAGE TO VIEW THE VISUALIZATION</i></p>
  <div class="row">
    <div class="col-md-4">
      <div class="thumbnail">
        <a href="http://localhost:5000/lc" target="_blank">
          <img src="https://www.patternfly.org/v3/pattern-library/data-
visualization/line-chart/img/single-line-
chart.png" alt="Lights" style="width:100%">
          <div class="caption">
            <p>SINGLE LINE CHARTS</p>
          </div>
        </a>
      </div>
    </div>
    <div class="col-md-4">
      <div class="thumbnail">
        <a href="http://localhost:5000/pie" target="_blank">
          <img src="https://miro.medium.com/max/629/1*rzNBRBt1RCr4NsN4Lb7V1Q.p
ng" alt="Nature" style="width:100%">
          <div class="caption">
            <p>PIECHART</p>
          </div>
        </a>
```

```
      </div>
    </div>
    <div class="col-md-4">
      <div class="thumbnail">
        <a href="http://localhost:5000/bplot" target="_blank">
          <img src="https://cdn2.vectorstock.com/i/1000x1000/81/86/stacked-
bar-chart-vector-9758186.jpg" alt="Fjords" style="width:100%">
          <div class="caption">
            <p>BAR-PLOT</p>
          </div>
        </a>
      </div>
    </div>

  </div>

  <div class="row">
    <div class="col-md-4">
      <div class="thumbnail">
        <a href="http://localhost:5000/mlc" target="_blank">
          <img src="https://cms-
assets.tutsplus.com/uploads/users/442/posts/22973/image/skjdhd636363.png" alt=
"Lights" style="width:100%">
          <div class="caption">
            <p> MULTI-LINE CHART</p>
          </div>
        </a>
      </div>
    </div>
    <div class="col-md-4">
      <div class="thumbnail">
        <a href="http://localhost:5000/scplt" target="_blank">
          <img src="https://datavizproject.com/wp-
content/uploads/2016/06/DVP_1_100-40.png" alt="Nature" style="width:100%">
          <div class="caption">
            <p>Polar area Chart</p>
          </div>
        </a>
      </div>
    </div>


  </div>
```

```
</div>


</body>
</html>
```

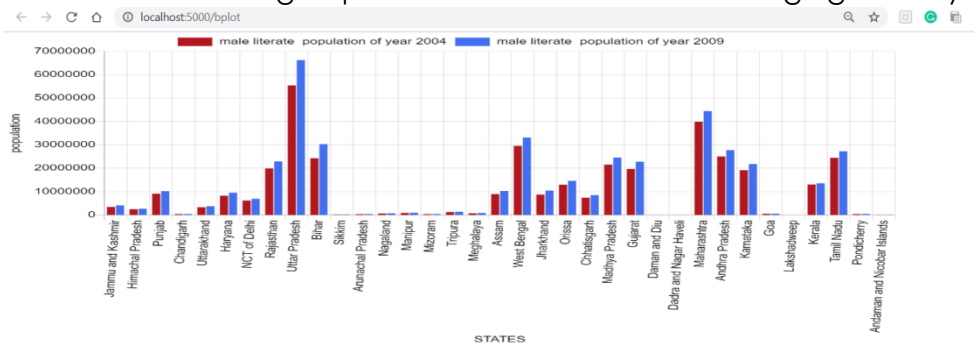
3. Details of the component of the visual can be seen when a cursor is placed on it



6. VISUAL IDOMS :

Based upon the visual chosen there were 2-3 visual encoding idioms were used

1. One is the usage of states to separate the data values from the set of values and group the data values belonging to the same state
2. Second is the column height to differentiate the attribute values
3. Third is the color to group the values of attributes belonging to the year



As shown the color is used to differentiate data values belonging to a different years , our first visual encoding algorithm , height of column to separate the data values – second visual encoding algorithm and states to group the same – third visual encoding algorithm

7. IMPLEMENTATION :

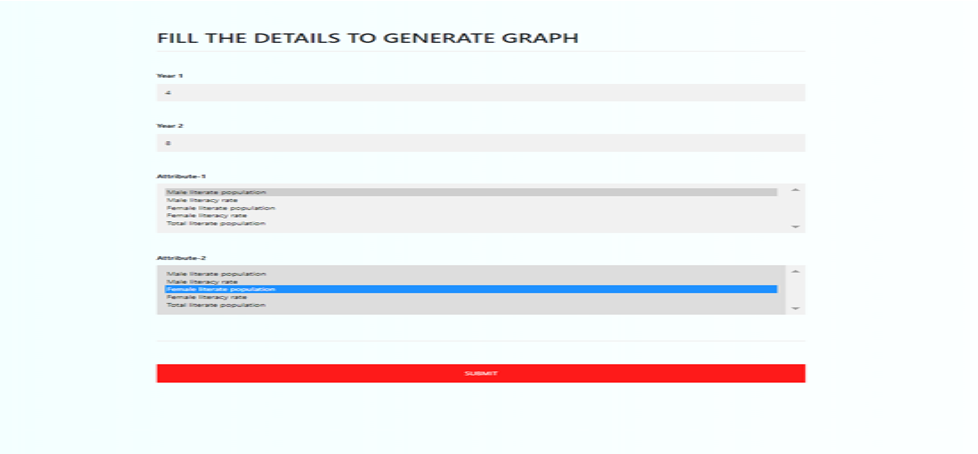
We execute this application in Vscode editor.

We will first start by running the app.py folder

```
(work_env) PS D:\dtp\files> python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 255-953-427
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [06/Jun/2020 11:37:15] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:37:15] "GET /static/jquery-2.1.4.js HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:37:15] "GET /static/data.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:37:15] "GET /static/tr.js HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:37:26] "POST /result HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:37:26] "GET /static/jquery-2.1.4.js HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:37:26] "GET /static/result.css HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:42:57] "GET /lc HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:50:46] "GET /mlc HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 11:54:07] "GET /bplot HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 12:01:44] "GET /mlc HTTP/1.1" 200 -
127.0.0.1 - - [06/Jun/2020 12:01:50] "GET /scplt HTTP/1.1" 200 -
```

Then copy paste the url that appears on the screen to in your browser connect to the server

Then the starting page opens up requesting for the details



Then after pressing submit we reach the page where our choice for visualization is given

Then on clicking on the visualization user intends to view , the respective visualization opens up in a new page showing the exact visualization with proper visual encoding applied along with labelled axes and legends to depict what the visualization represents and to which attribute it belongs to.

MODULES OF CODE:

app.py

```
import xlrd as xl
import numpy as np
import matplotlib.pyplot as plt
from flask import Flask, jsonify, request, render_template, Markup, json, sess
ion, redirect, url_for,session
import io
import base64
from  funf import ds1,ds2
from dc import att1,att2
import json

app = Flask(__name__)
app.config["TEMPLATES_AUTO_RELOAD"]=True

@app.route('/')
def data():
   return render_template('data.html')


@app.route('/result',methods = ['POST', 'GET'])
def result():
   if request.method == 'POST':
      global result
      result = request.form
      global a
      global b
```

```
      global c
      global d
      a=int(result["dataset-1"])
      b=int(result["dataset-2"])
      c=int(result["Attribute-1"])
      d=int(result["Attribute-2"])
      global l
      global k
      f=ds1(a)
      g=ds2(b)
      k=att1(f,c)
      l=att2(g,d)
      global p
      global k1
      global k2
      k1=l.copy()
      k2=k.copy()
      k1=np.array(k1)
      k2=np.array(k2)
      p=np.subtract(k1,k2)
      p=list(p)
      global label
      label=['Jammu and Kashmir','Himachal Pradesh','Punjab','Chandigarh','Utt
arakhand','Haryana','NCT of Delhi', 'Rajasthan','Uttar Pradesh','Bihar','Sikki
m','Arunachal Pradesh','Nagaland','Manipur','Mizoram','Tripura','Meghalaya','A
ssam','West Bengal','Jharkhand','Orissa','Chhatisgarh','Madhya Pradesh','Gujar
at','Daman and Diu','Dadra and Nagar Haveli','Maharashtra','Andhra Pradesh','K
arnataka','Goa','Lakshadweep','Kerala','Tamil Nadu','Pondicherry','Andaman and
 Nicobar Islands']
      global at1,at2
      if(c==1):
         at1="male literate  population"
      elif(c==2):
         at1="male literacy rate"

      elif(c==3):
         at1="female literate population"

      elif(c==4):
         at1="female literacy rate"

      elif(c==5):
         at1="total literate population"

      if(d==1):
         at2="male literate  population"

      elif(d==2):
```

```
         at2="male literacy rate"

      elif(d==3):
         at2="female literate population"

      elif(d==4):
         at2="female literacy rate"

      elif(d==5):
         at2="total literate population"


      return render_template('result.html',result = result,a=a,b=b,c=c,d=d,p=p
,k=k1,l=k2,label=label)



@app.route('/lc')
def lc():
   return render_template("gr.html",k=k,l=l,p=p,label=label,a=a,b=b,c=c,d=d,at
1=at1,at2=at2)

@app.route('/pie')
def hst():
   return render_template("pie.html",k=k,l=l,label=label,p=p,at1=at1,at2=at2,a
=a,b=b,c=c,d=d)

@app.route('/bplot')
def bplot():
   return render_template("bplot.html",k=k,l=l,p=p,label=label,at1=at1,at2=at2
,a=a,b=b,c=c,d=d)

@app.route('/mlc')
def mlc():
   return render_template("mlc.html",k=k,l=l,p=p,label=label,at1=at1,at2=at2,a
=a,b=b,c=c,d=d)

@app.route('/scplt')
def scplt():
   return render_template("scplt.html",k=k,l=l,p=p,label=label,at1=at1,at2=at2
,a=a,b=b,c=c,d=d)
if __name__ == '__main__':
   app.run(debug=True)
```

funf.py-Module containing function used for database selection

it opens the necessary xlsx file and starts indexing it from zero

```
import xlrd as xl
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px


def ds1(a):
    if(a==1):
        d1=xl.open_workbook('l1.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==2):
        d1=xl.open_workbook('l2.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==3):
        d1=xl.open_workbook('l3.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==4):
        d1=xl.open_workbook('l4.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==5):
        d1=xl.open_workbook('l5.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==6):
        d1=xl.open_workbook('l6.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==7):
        d1=xl.open_workbook('l7.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==8):
        d1=xl.open_workbook('l8.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==9):
        d1=xl.open_workbook('l9.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==10):
        d1=xl.open_workbook('l10.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==11):
        d1=xl.open_workbook('l11.xlsx')
        a1=d1.sheet_by_index(0)
    else:
        exit()
    return(a1)

def ds2(b):
    if(b==1):
        d2=xl.open_workbook('l1.xlsx')
```

```
        a2=d2.sheet_by_index(0)
    elif(b==2):
        d2=xl.open_workbook('l2.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==3):
        d2=xl.open_workbook('l3.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==4):
        d2=xl.open_workbook('l4.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==5):
        d2=xl.open_workbook('l5.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==6):
        d2=xl.open_workbook('l6.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==7):
        d2=xl.open_workbook('l7.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==8):
        d2=xl.open_workbook('l8.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==9):
        d2=xl.open_workbook('l9.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==10):
        d2=xl.open_workbook('l10.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==11):
        d2=xl.open_workbook('l11.xlsx')
        a2=d2.sheet_by_index(0)
    else:
        exit()
    return(a2)
```

dc.py-Module containing function used for attribute selection and collection

It returns the list of values of attributes chosen by the user

```
import xlrd as xl
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
l1=[]
def att1(f,c):
```

```
    if(c==1):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,2)))

    elif(c==2):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,5)))

    elif(c==3):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,3)))
    elif(c==4):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,6)))
    elif(c==5):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,4)))
    else:
        exit()
    return(l1)
l2=[]
def att2(g,d):
    if(d==1):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,2)))

    elif(d==2):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,5)))

    elif(d==3):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,3)))
    elif(d==4):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,6)))
    elif(d==5):
        i=7
        for i in range(7,g.nrows):
```

```
            l2.append(int(g.cell_value(i,4)))
    else:
        exit()
    return(l2)
```

Code for graphs to appear to web page were previously shown under the tasks section

8.VALIDATION :

Validation is an automated check, performed to guarantee that the data input is rational and acceptable. It does not check the correctness of the data itself.

In this project , user is given the choice of the database and its validated using functions **ds1** and **ds2** programmed in a module designed named as **funf.py**,in which the choice made by the user is checked and corresponding database is returned , that is in this project the year for which the user intends to analyze

MODULE funf.py:

```
import xlrd as xl
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px


def ds1(a):
    if(a==1):
        d1=xl.open_workbook('l1.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==2):
        d1=xl.open_workbook('l2.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==3):
        d1=xl.open_workbook('l3.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==4):
        d1=xl.open_workbook('l4.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==5):
        d1=xl.open_workbook('l5.xlsx')
        a1=d1.sheet_by_index(0)
```

```
    elif(a==6):
        d1=xl.open_workbook('l6.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==7):
        d1=xl.open_workbook('l7.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==8):
        d1=xl.open_workbook('l8.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==9):
        d1=xl.open_workbook('l9.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==10):
        d1=xl.open_workbook('l10.xlsx')
        a1=d1.sheet_by_index(0)
    elif(a==11):
        d1=xl.open_workbook('l11.xlsx')
        a1=d1.sheet_by_index(0)
    else:
        exit()
    return(a1)

def ds2(b):
    if(b==1):
        d2=xl.open_workbook('l1.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==2):
        d2=xl.open_workbook('l2.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==3):
        d2=xl.open_workbook('l3.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==4):
        d2=xl.open_workbook('l4.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==5):
        d2=xl.open_workbook('l5.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==6):
        d2=xl.open_workbook('l6.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==7):
        d2=xl.open_workbook('l7.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==8):
        d2=xl.open_workbook('l8.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==9):
```

```
        d2=xl.open_workbook('l9.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==10):
        d2=xl.open_workbook('l10.xlsx')
        a2=d2.sheet_by_index(0)
    elif(b==11):
        d2=xl.open_workbook('l11.xlsx')
        a2=d2.sheet_by_index(0)
    else:
        exit()
    return(a2)
```

Then to collect the values of all the attribute chosen , from the database using function**att1**and **att2** programmed in another module **dc.py**designed, then it returns the list of values of the attribute chosen by the user

MODULE dc.py

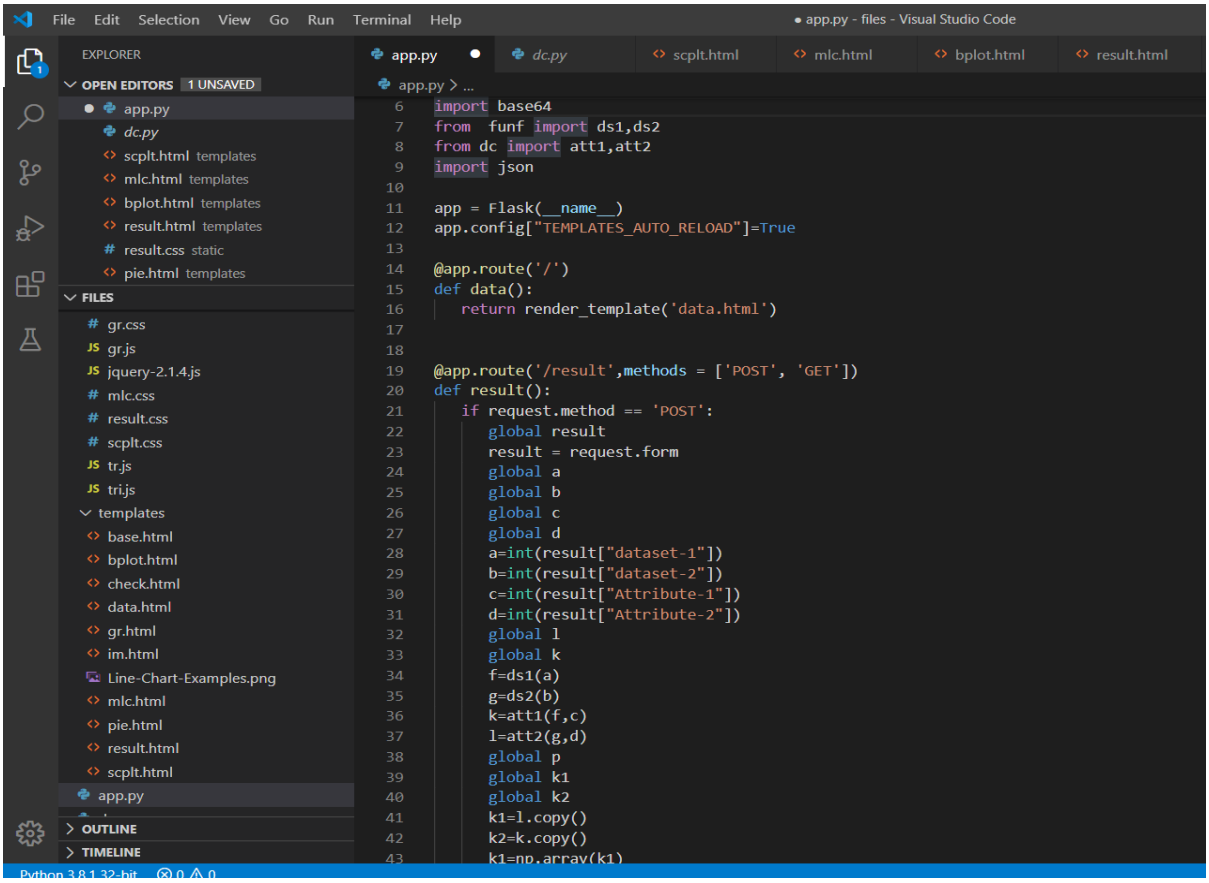
```
import xlrd as xl

import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
l1=[]
def att1(f,c):
    if(c==1):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,2)))

    elif(c==2):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,5)))

    elif(c==3):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,3)))
    elif(c==4):
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,6)))
    elif(c==5):
```

```
        i=7
        for i in range(7,f.nrows):
            l1.append(int(f.cell_value(i,4)))
    else:
        exit()
    return(l1)
l2=[]
def att2(g,d):
    if(d==1):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,2)))

    elif(d==2):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,5)))

    elif(d==3):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,3)))
    elif(d==4):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,6)))
    elif(d==5):
        i=7
        for i in range(7,g.nrows):
            l2.append(int(g.cell_value(i,4)))
    else:
        exit()
    return(l2)
```

THEIR USAGE IN MAIN MODULE app.py where in respective function are called and used and libraries included

The functions ds1( )and ds2( ) select the xlsx file chosen the by the user and index it starting from zero

The functions att1( ) and att2( )



9.CONCLUSION :

From the above execution of application we were successful in building an interactive data visualization dashboard using visual code as an editor and using flask framework to host it on the local server in order to make an user interactive dashboard .

We Used minimum of two visual encoding algorithms for each data visual the user chooses and to a maximum of 3 depending on the visualization.

We were successful in providing the exact picture to the target audience by revealing facts through data visuals what may otherwise be missed when viewed in form of datasets

We were able to give the target audience the choice to choose the years for analysis along with the attributes which help them in zeroing down the issue they would like to work on, hence by reducing the time wastage in going through resources which are bulky and inference to be made can rarely be made accurately without delay , By using this application user will easily identify the shortcomings of effort they've been making over the years and also helps to select the right target, thereby helping in reaching their objective in optimal amount of time, which helps in overall growth of the nation.