[Rahul Annapandi](mailto:rahul.annapandi@somaiya.edu)

**lyit-53**

**Experiment no.10**

**Aim**- Design web pages using suitable graphs (Google charts + D3.js) for explaining the cognitive application development steps

**Objective:** To understand and implement applications using concepts of AI and DS.

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Cognitive Application Development Steps</title>

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

<script src="https://d3js.org/d3.v7.min.js"></script>

<style>

body {

font-family: Arial, sans-serif;

margin: 20px;

}

.chart-container {

display: flex;

justify-content: space-around;

flex-wrap: wrap;

}

.chart {

width: 45%;

height: 400px;

margin-bottom: 50px;

}

.d3-container {

width: 100%;

height: 500px;

}

.title {

text-align: center;

margin-top: 20px;

}

</style>

</head>

<body>

<h1 class="title">Cognitive Application Development Steps</h1>

<!-- Google Chart: Bar Chart for Development Phases -->

<div class="chart-container">

<div id="bar\_chart\_div" class="chart"></div>

<div id="line\_chart\_div" class="chart"></div>

</div>

<!-- Google Chart: Pie Chart for Development Phases Distribution -->

<div id="pie\_chart\_div" class="chart"></div>

<!-- D3.js Tree Diagram -->

<h2 class="title">Hierarchical Overview of Development Stages</h2>

<div id="d3-container" class="d3-container"></div>

<h2 class="title">Force-Directed Graph of Steps</h2>

<div id="force-directed" class="d3-container"></div>

<!-- Load Google Charts -->

<script type="text/javascript">

google.charts.load('current', {'packages':['corechart']});

google.charts.setOnLoadCallback(drawCharts);

// Drawing all charts

function drawCharts() {

drawBarChart();

drawLineChart();

drawPieChart();

}

// Google Bar Chart

function drawBarChart() {

var data = google.visualization.arrayToDataTable([

['Development Step', 'Effort (hrs)', { role: 'style' }],

['Data Collection', 10, 'color: #76A7FA'],

['Data Preprocessing', 15, 'color: #76A7FA'],

['Modeling', 20, 'color: #76A7FA'],

['Training & Testing', 25, 'color: #76A7FA'],

['Deployment & Monitoring', 30, 'color: #76A7FA']

]);

var options = {

title: 'Cognitive Application Development Effort per Phase',

chartArea: {width: '50%'},

hAxis: {

title: 'Effort (hrs)',

minValue: 0

},

vAxis: {

title: 'Development Step'

}

};

var chart = new google.visualization.BarChart(document.getElementById('bar\_chart\_div'));

chart.draw(data, options);

}

// Google Line Chart

function drawLineChart() {

var data = google.visualization.arrayToDataTable([

['Development Step', 'Completion %'],

['Data Collection', 20],

['Data Preprocessing', 40],

['Modeling', 60],

['Training & Testing', 80],

['Deployment & Monitoring', 100]

]);

var options = {

title: 'Progress of Development Phases',

curveType: 'function',

legend: { position: 'bottom' }

};

var chart = new google.visualization.LineChart(document.getElementById('line\_chart\_div'));

chart.draw(data, options);

}

// Google Pie Chart

function drawPieChart() {

var data = google.visualization.arrayToDataTable([

['Phase', 'Effort (hrs)'],

['Data Collection', 10],

['Data Preprocessing', 15],

['Modeling', 20],

['Training & Testing', 25],

['Deployment & Monitoring', 30]

]);

var options = {

title: 'Effort Distribution Across Phases'

};

var chart = new google.visualization.PieChart(document.getElementById('pie\_chart\_div'));

chart.draw(data, options);

}

</script>

<!-- D3.js Tree Diagram -->

<script>

const data = {

"name": "Development Steps",

"children": [

{

"name": "Data Collection",

"children": [

{ "name": "Sources", "value": 100 },

{ "name": "Sensors", "value": 200 }

]

},

{

"name": "Data Preprocessing",

"children": [

{ "name": "Cleaning", "value": 150 },

{ "name": "Transformation", "value": 120 }

]

},

{

"name": "Modeling",

"children": [

{ "name": "Feature Engineering", "value": 300 },

{ "name": "Algorithm Selection", "value": 250 }

]

},

{

"name": "Training & Testing",

"children": [

{ "name": "Train Model", "value": 200 },

{ "name": "Evaluate Model", "value": 100 }

]

},

{

"name": "Deployment & Monitoring",

"children": [

{ "name": "Deploy Model", "value": 180 },

{ "name": "Monitor Model", "value": 220 }

]

}

]

};

const width = 600;

const height = 400;

const svg = d3.select("#d3-container").append("svg")

.attr("width", width)

.attr("height", height)

.append("g")

.attr("transform", "translate(50,50)");

const root = d3.hierarchy(data);

const treeLayout = d3.tree().size([width - 100, height - 100]);

treeLayout(root);

// Nodes

svg.selectAll('circle')

.data(root.descendants())

.enter()

.append('circle')

.attr('cx', d => d.x)

.attr('cy', d => d.y)

.attr('r', 5)

.style('fill', 'steelblue');

// Links

svg.selectAll('line')

.data(root.links())

.enter()

.append('line')

.attr('x1', d => d.source.x)

.attr('y1', d => d.source.y)

.attr('x2', d => d.target.x)

.attr('y2', d => d.target.y)

.style('stroke', '#ccc');

// Labels

svg.selectAll('text')

.data(root.descendants())

.enter()

.append('text')

.attr('x', d => d.x)

.attr('y', d => d.y - 10)

.attr('text-anchor', 'middle')

.text(d => d.data.name);

</script>

<!-- D3.js Force-Directed Graph -->

<script>

const forceData = {

nodes: [

{ id: "Data Collection" },

{ id: "Data Preprocessing" },

{ id: "Modeling" },

{ id: "Training & Testing" },

{ id: "Deployment & Monitoring" }

],

links: [

{ source: "Data Collection", target: "Data Preprocessing" },

{ source: "Data Preprocessing", target: "Modeling" },

{ source: "Modeling", target: "Training & Testing" },

{ source: "Training & Testing", target: "Deployment & Monitoring" }

]

};

const forceSvg = d3.select("#force-directed").append("svg")

.attr("width", width)

.attr("height", height);

const simulation = d3.forceSimulation(forceData.nodes)

.force("link", d3.forceLink(forceData.links).id(d => d.id).distance(150))

.force("charge", d3.forceManyBody().strength(-400))

.force("center", d3.forceCenter(width / 2, height / 2));

const link = forceSvg.append("g")

.selectAll("line")

.data(forceData.links)

.enter().append("line")

.attr("stroke-width", 2)

.attr("stroke", "#999");

const node = forceSvg.append("g")

.selectAll("circle")

.data(forceData.nodes)

.enter().append("circle")

.attr("r", 10)

.attr("fill", "steelblue")

.call(d3.drag()

.on("start", dragStarted)

.on("drag", dragged)

.on("end", dragEnded));

const text = forceSvg.append("g")

.selectAll("text")

.data(forceData.nodes)

.enter().append("text")

.attr("x", 12)

.attr("y", 3)

.text(d => d.id);

simulation.on("tick", () => {

link

.attr("x1", d => d.source.x)

.attr("y1", d => d.source.y)

.attr("x2", d => d.target.x)

.attr("y2", d => d.target.y);

node

.attr("cx", d => d.x)

.attr("cy", d => d.y);

text

.attr("x", d => d.x + 12)

.attr("y", d => d.y + 3);

});

function dragStarted(event, d) {

if (!event.active) simulation.alphaTarget(0.3).restart();

d.fx = d.x;

d.fy = d.y;

}

function dragged(event, d) {

d.fx = event.x;

d.fy = event.y;

}

function dragEnded(event, d) {

if (!event.active) simulation.alphaTarget(0);

d.fx = null;

d.fy = null;

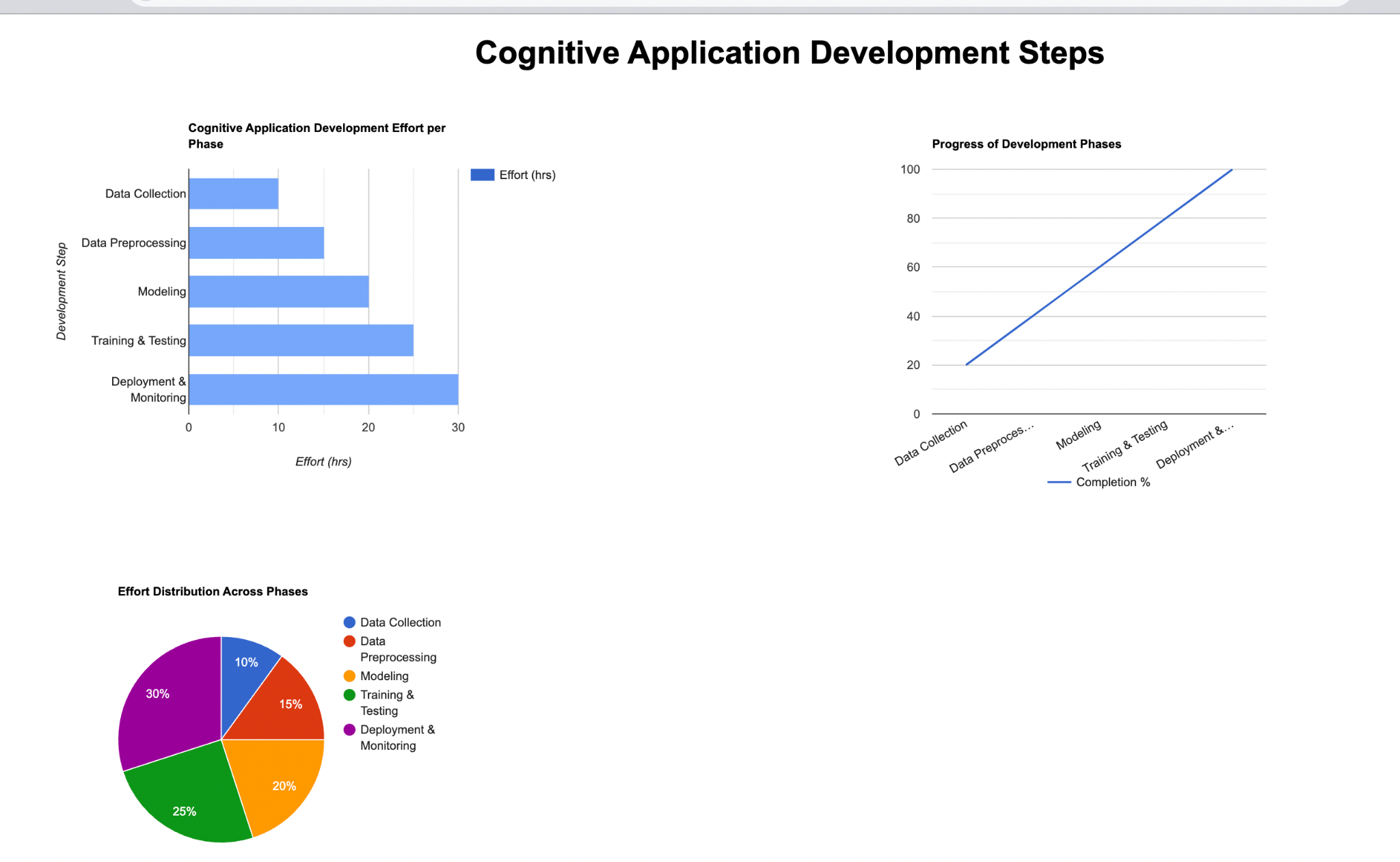
}

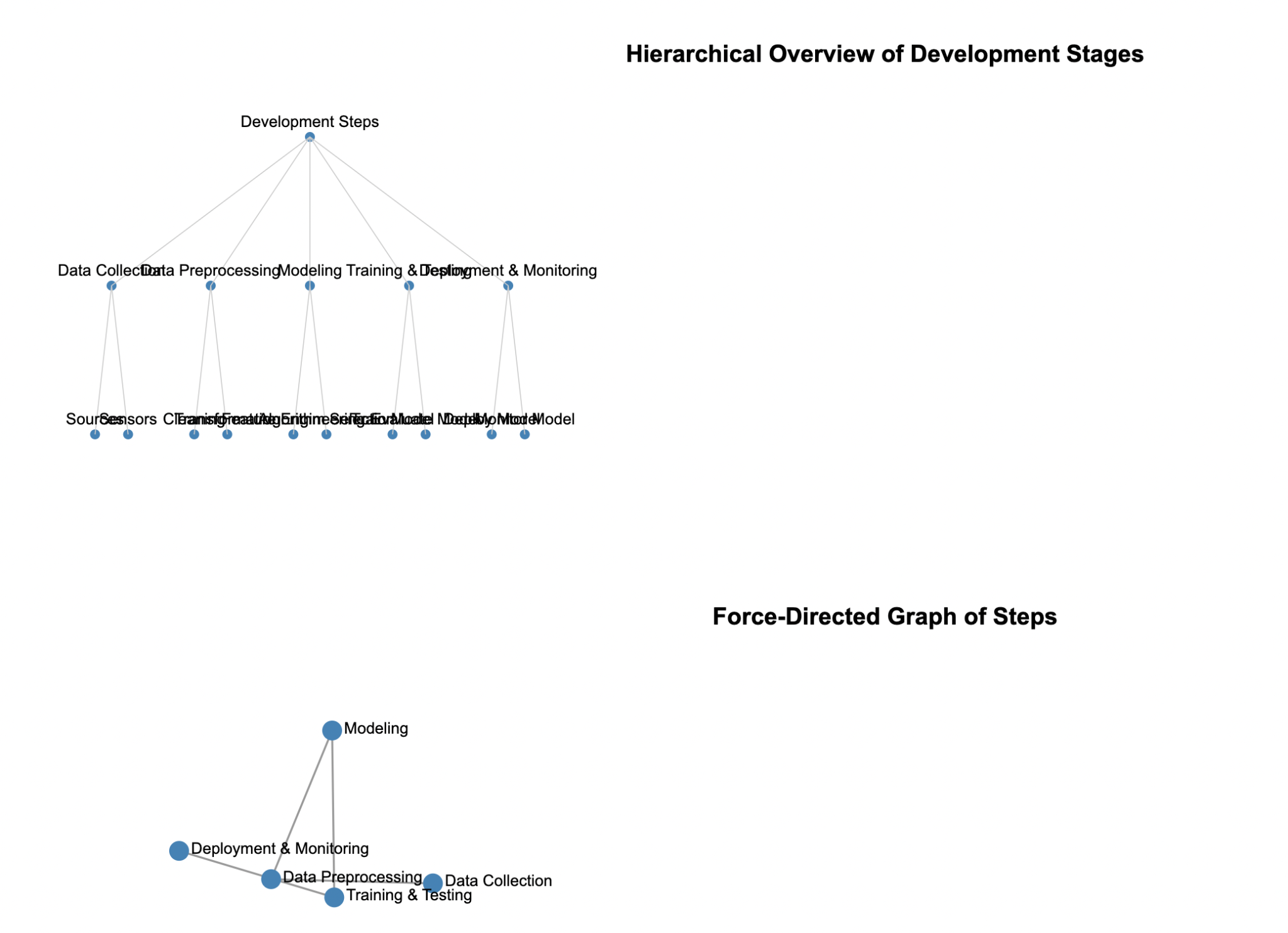
</script>

</body>

</html>

**Screenshot:**

****

****

**Outcome:** Implement applications using concepts of AI and DS.

**Conclusion:** I want to conclude that using a combination of Google Charts and D3.js provides a powerful, interactive way to visualize the steps and relationships in cognitive application development. These tools help simplify complex data into intuitive and meaningful insights for better understanding and decision-making.We implemented this experiment successfully and got our respective output.