**PROJECT**

**ON**

**Basic Social Network Analysis Tool**

**(Data Structures and Algorithms)**

**Overview:**

This document outlines a basic social network analysis tool implemented in Python. The tool focuses on graph representation, degree centrality calculation, and basic visualization.?

**Code Structure:**

Python:

import matplotlib.pyplot as plt

import networkx as nx

G = nx.Graph()

G.add\_node("charvik")

G.add\_node("sushma")

G.add\_node("venky")

G.add\_node("charan")

G.add\_edge("charvik", "sushma")

G.add\_edge("sushma", "venky")

G.add\_edge("venky", "charan")

G.add\_edge("charan", "charvik")

pos = nx.spring\_layout(G)

nx.draw\_networkx\_nodes(G, pos, node\_size=700, node\_color="lightblue")

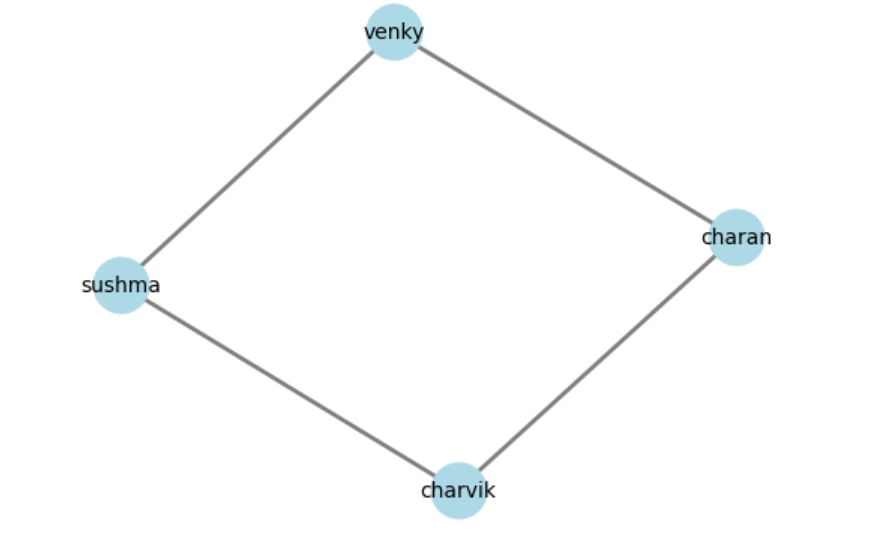
nx.draw\_networkx\_edges(G, pos, width=2, edge\_color="gray")

nx.draw\_networkx\_labels(G, pos, font\_size=10, font\_family="sans-serif")

plt.axis("off")

plt.show()

**Output:**



**Explanation of the Project:**

Here's a step-by-step explanation of the project:

Step 1: Importing Libraries

* import matplotlib.pyplot as plt: Imports the Matplotlib library, which is used for creating static, animated, and interactive visualizations. The as plt part assigns the alias plt to the library for easier access.
* import networkx as nx: Imports the NetworkX library, which is used for creating and analyzing complex networks. The as nx part assigns the alias nx to the library for easier access.

Step 2: Creating a Graph:-

* G = nx.Graph(): Creates a new, empty graph object G using NetworkX.

Step 3: Adding Nodes:-

* G.add\_node("charvik"): Adds a node with the label "charvik" to the graph.
* G.add\_node("sushma"): Adds a node with the label "sushma" to the graph.
* G.add\_node("venky"): Adds a node with the label "venky" to the graph.
* G.add\_node("charan"): Adds a node with the label "charan" to the graph.

Step 4: Adding Edges:-

* G.add\_edge("charvik", "sushma"): Adds an edge between the nodes "charvik" and "sushma".
* G.add\_edge("sushma", "venky"): Adds an edge between the nodes "sushma" and "venky".
* G.add\_edge("venky", "charan"): Adds an edge between the nodes "venky" and "charan".
* G.add\_edge("charan", "charvik"): Adds an edge between the nodes "charan" and "charvik".

Step 5: Positioning Nodes:-

* pos = nx.spring\_layout(G): Uses the Fruchterman-Reingold force-directed algorithm to position the nodes in the graph. This algorithm tries to position nodes in a way that minimizes edge crossings and keeps nodes at a similar distance from each other.

Step 6: Drawing Nodes:-

* nx.draw\_networkx\_nodes(G, pos, node\_size=700, node\_color="lightblue"): Draws the nodes in the graph using the positions calculated in Step 5. The nodes are colored light blue, and their size is set to 700.

Step 7: Drawing Edges:-

* nx.draw\_networkx\_edges(G, pos, width=2, edge\_color="gray"): Draws the edges in the graph using the positions calculated in Step 5. The edges are colored gray, and their width is set to 2.

Step 8: Drawing Labels:-

* nx.draw\_networkx\_labels(G, pos, font\_size=10, font\_family="sans-serif"): Draws the labels for each node in the graph using the positions calculated in Step 5. The labels are displayed in sans-serif font with a font size of 10.

Step 9: Showing the Plot

* plt.axis("off"): Turns off the axis for the plot, so only the graph is displayed.
* plt.show(): Displays the plot in a window.

This project creates a simple social network visualization using NetworkX and Matplotlib.