In [1]:
```
import pandas as pd
import networkx as nx
import json
```

In [2]:
```
df = pd.read_csv("/Users/jisusingh/Downloads/force_layout_d3js/data_scopus.csv"
df = df.fillna(0)
```

In [3]:
```
df.head()
```

Out[3]:

	Title	Year	EID	Abstract	Publisher	Conference name	Conference date	Author
0	Virtual reality applications for the built env...	2020	2-s2.0-85086464158	With its advanced capabilities of immersive an...	Elsevier B.V.	0	0	Zhang Y. Liu H. Kang S. C., Al Hussei M
1	Self-tracking while doing sport: Comfort, moti...	2020	2-s2.0-85082875828	The spread of wearable technologies is paving ...	Academic Press	0	0	Rapp A. Tirabeni L
2	Bridge damage: Detection, IFC-based semantic e...	2020	2-s2.0-85078194587	Building Information Modeling (BIM) representa...	Elsevier B.V.	0	0	Isailovi D. Stojanovi V., Trapp M. Richter..
3	VR system for spatio-temporal visualization of...	2019	2-s2.0-85075706132	Social media analysis is helpful to understand...	Springer	0	0	Okada K. Yoshid M., Itoh T. Czaudern T., S..
4	DiseaSE: A biomedical text analytics system fo...	2019	2-s2.0-85074886243	Due to increasing volume and unstructured natu...	Academic Press Inc.	0	0	Abulaish M. Parwe M.A. Jahiruddi

In [4]:
```
G = nx.Graph()

for _, row in df.iterrows():
    authors = row['Authors'].split(', ')
    eid = row['EID']

    for i in range(len(authors)):
        for j in range(i + 1, len(authors)):
            G.add_edge(authors[i], authors[j], publication=eid)

data = nx.readwrite.json_graph.node_link_data(G)

output_path = "/Users/jisusingh/Downloads/force_layout_d3js/author_network.jso

with open(output_path, 'w') as f:
```

```
    json.dump(data, f, indent=2)

print(f"Network data saved as JSON: {output_path}")
```

```
Network data saved as JSON: /Users/jisusingh/Downloads/force_layout_d3js/autho
r_network.json
```

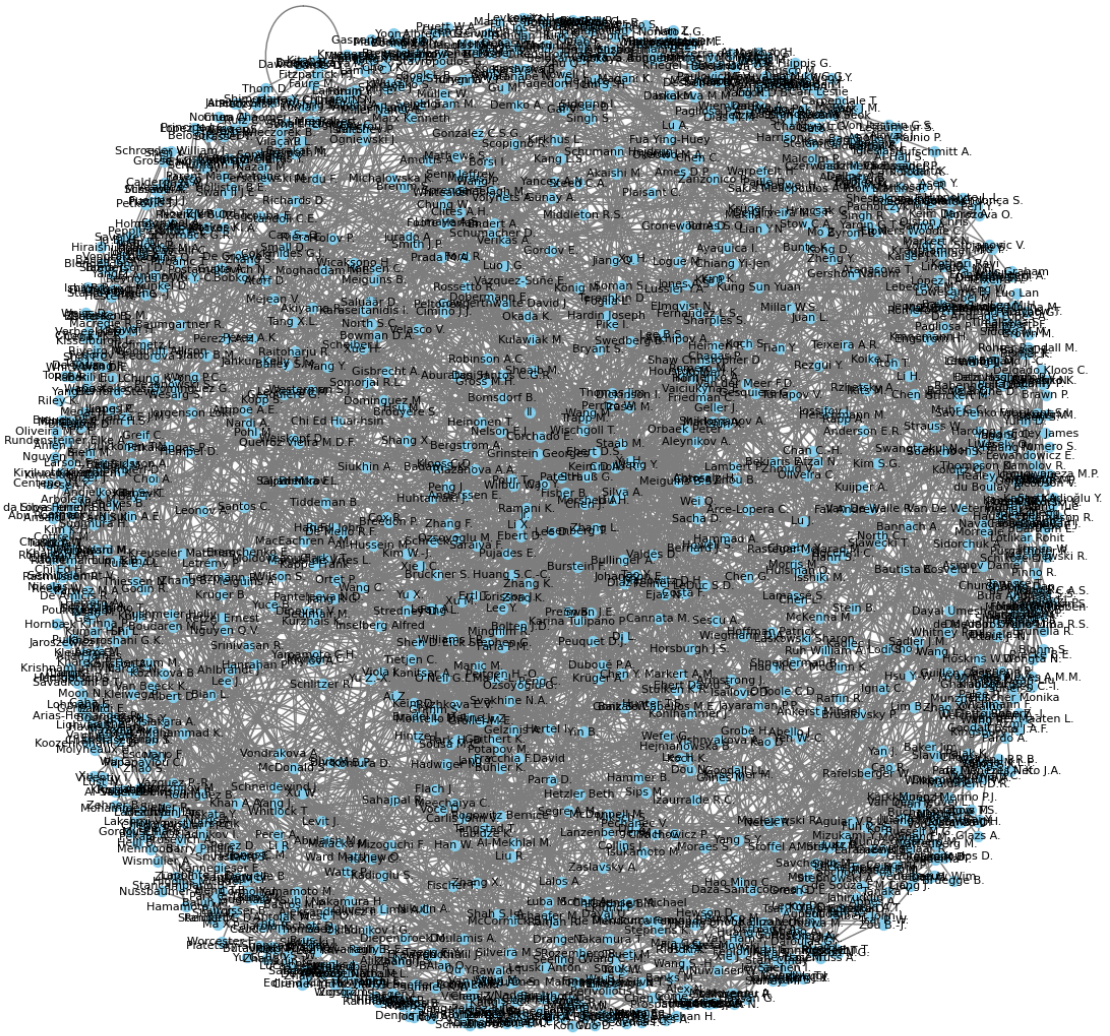
In [5]:

```
import json
import matplotlib.pyplot as plt
from networkx.readwrite import json_graph

with open("/Users/jisusingh/Downloads/force_layout_d3js/author_network.json") a
    data = json.load(f)

G = json_graph.node_link_graph(data)

plt.figure(figsize=(12, 12))
pos = nx.spring_layout(G, k=0.5)
nx.draw(G, pos, with_labels=True, node_size=50, font_size=8, edge_color="gray"
plt.title("Author Network Graph")
plt.show()
```



In [6]:
```
df['Affiliation Country'] = df['Authors with affiliations'].apply(lambda x: x.s

top_countries = df['Affiliation Country'].value_counts().nlargest(10).index.to
```

In [7]:
```
top_countries
```

Out[7]:
```
['United States',
 'Germany',
 'United Kingdom',
 'South Korea',
 'China',
 'Canada',
 'Russian Federation',
 'Japan',
 'Brazil',
 'Australia']
```

In [8]:
```
import matplotlib.colors as mcolors
```

```
color_map = {country: color for country, color in zip(top_countries, plt.cm.tal
default_color = "#A9A9A9"
```

In [9]:

```
G = nx.Graph()

for _, row in df.iterrows():
    authors = row['Authors'].split(', ')
    eid = row['EID']
    country = row['Affiliation Country']
    node_color = color_map.get(country, default_color)

    for author in authors:
        G.add_node(author, color=node_color)

    for i in range(len(authors)):
        for j in range(i + 1, len(authors)):
            G.add_edge(authors[i], authors[j], publication=eid)

plt.figure(figsize=(12, 12))
pos = nx.spring_layout(G, k=0.5)
node_colors = [G.nodes[author]["color"] for author in G.nodes]
nx.draw(G, pos, with_labels=True, node_size=50, font_size=8, edge_color="gray"
plt.title("Author Network Graph Colored by Affiliation Country")
plt.show()
```

Author Network Graph Colored by Affiliation Country

In [ ]: