Assignment No. 11

6CS371 : Advanced Database System Lab

Neo4j Graph Database

Name : Jay Shirgupe
PRN: 21510026
Batch: T-7
TY CSE

Aim

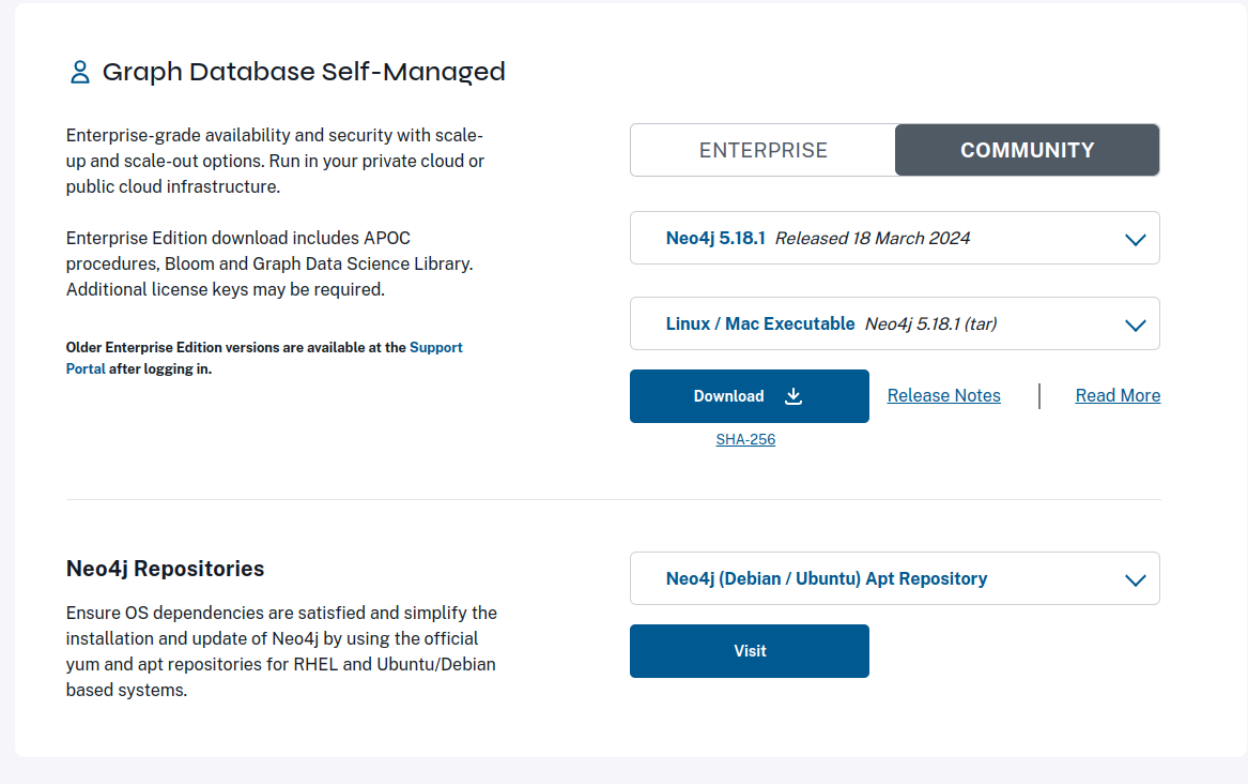
Design a Neo4j graph database for a "Research Papers Database" scenario. Develop a Python desktop application for efficient querying, including tasks like checking citations and retrieving paper classifications.

Introduction

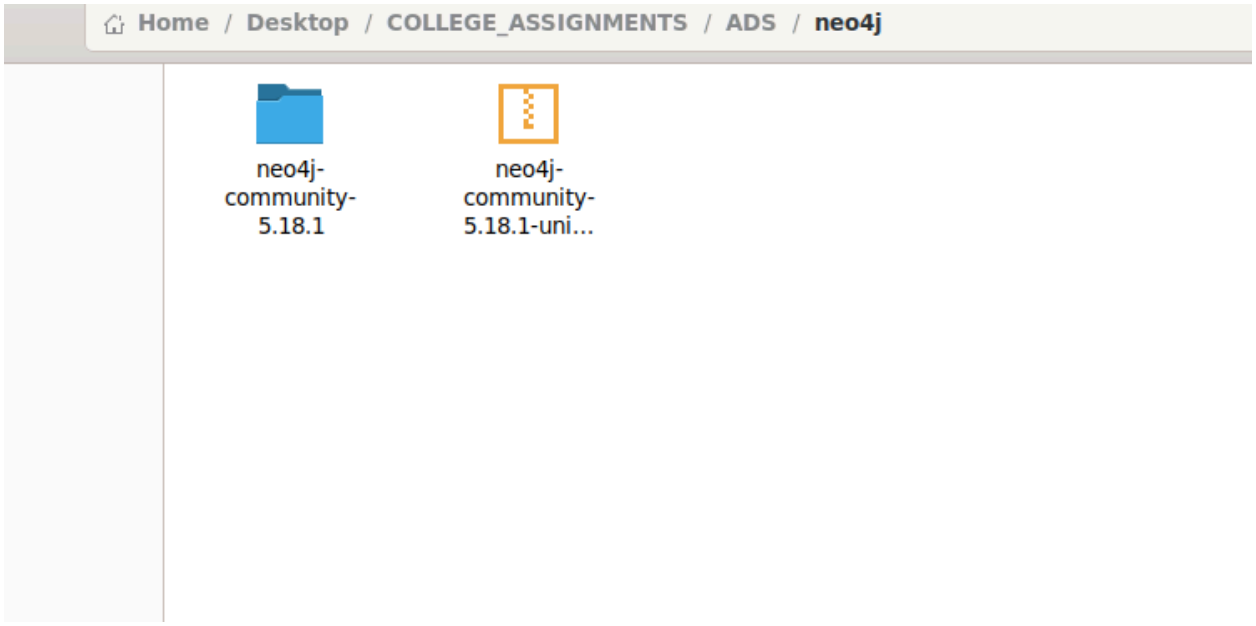
The "Research Papers Database" assignment involves the design and implementation of a graph database solution using Neo4j, a popular graph database management system. The database aims to address the complex relationships inherent in research papers, including authors, classifications, and citations. With an extensive dataset obtained from the Cora Research Paper Classification Project, comprising approximately 25,000 authors, 37,000 papers, and 220,000 relationships, the task is to create a robust database model capable of efficiently storing and querying research paper data. Furthermore, the assignment involves the development of a Python-based desktop application to provide users with a convenient interface for interacting with the database. The application should support various search functionalities, such as checking paper citations and retrieving paper classifications, empowering users to explore the interconnected nature of research papers effortlessly. In summary, this assignment presents an opportunity to apply graph database concepts in real-world scenarios, showcasing the power and versatility of Neo4j in managing complex relationships within large datasets. Additionally, the development of a user-friendly desktop application enhances the usability and accessibility of the database, facilitating seamless interaction and exploration of research paper data.

Procedure

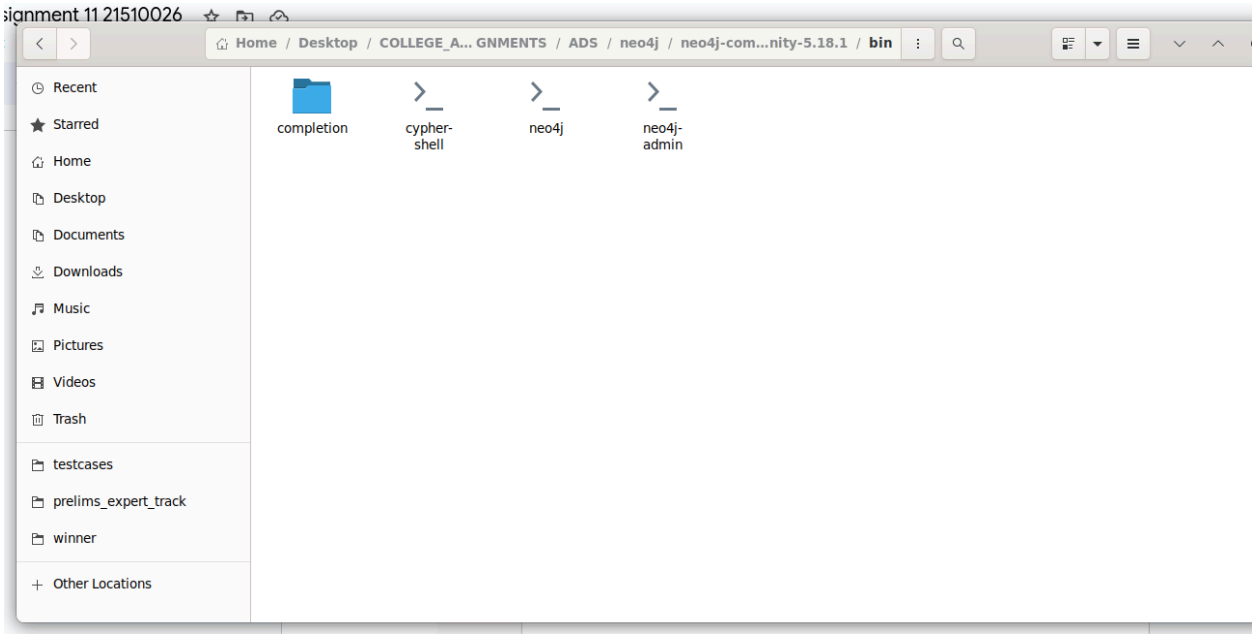
1. **Download Neo4j**
    a. Install OpenJDK 18
    b. Download the neo4j community server (.tar for linux, .zip for windows)



    c. Extract it to a folder



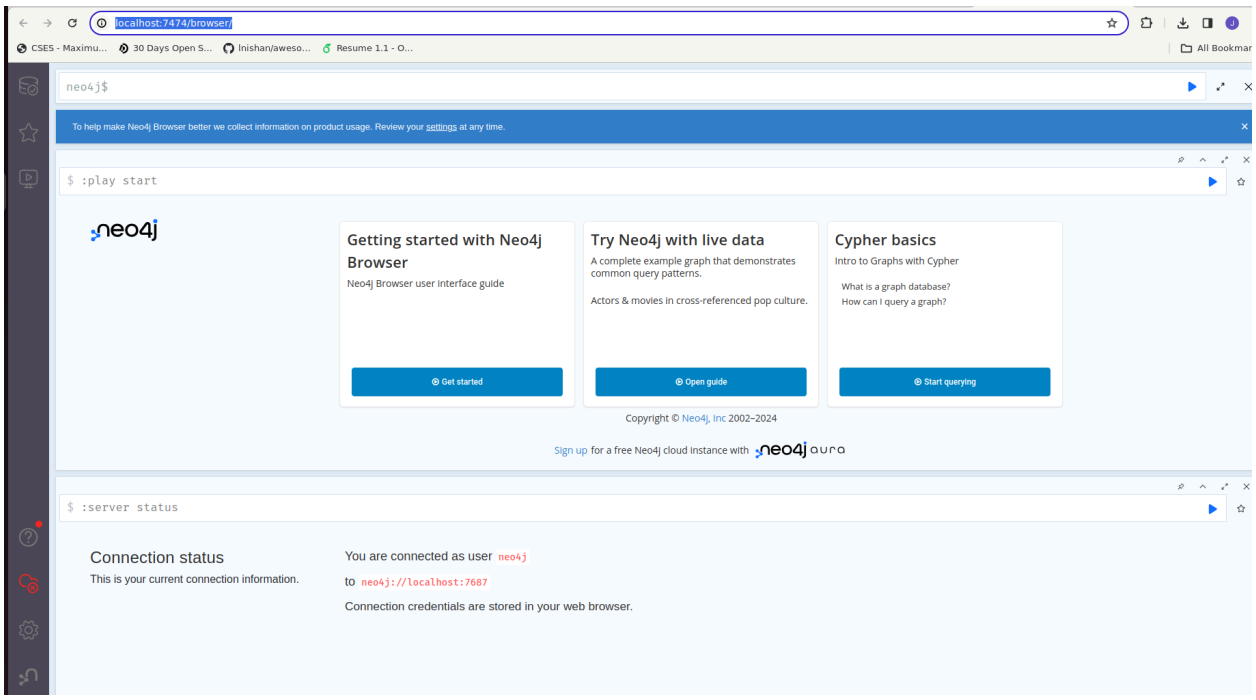
2. **Start Neo4j Server**
   a. Navigate to the /bin folder in neo4j server folder



   b. Run the neo4j executable with start parameter

```
oneautumleaf@oneautumleaf-IdeaPad-Gaming-3-15IHU6:~/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/bin$ ./neo4j start
Directories in use:
home:         /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1
config:       /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/conf
logs:         /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/logs
plugins:      /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/plugins
import:       /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/import
data:         /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/data
certificates: /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/certificates
licenses:     /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/licenses
run:          /home/oneautumleaf/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/run
Starting Neo4j.
Started neo4j (pid:7482). It is available at http://localhost:7474
There may be a short delay until the server is ready.
oneautumleaf@oneautumleaf-IdeaPad-Gaming-3-15IHU6:~/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5.18.1/bin$
```

3. **Access Neo4j Browser**
    a. Access neo4j browser from http://localhost:7474/browser/



4. **Login and Configure**
    a. For the first time, the following credentials are to be use
        Username: neo4j
        Password: neo4j
    b. On the first login password has to be changed.
5. **Load data for research paper**
    a. Load the data using the following commands

Load the nodes

```
LOAD CSV WITH HEADERS FROM
'https://raw.githubusercontent.com/ngshya/datasets/master/cora/cora_content.csv'
AS line FIELDTERMINATOR ','
CREATE (:Paper {id: line.paper_id, class: line.label});
```

Table

Code

Added 2708 labels, created 2708 nodes, set 5416 properties, completed after 24766 ms.

Added 2708 labels, created 2708 nodes, set 5416 properties, completed after 24766 ms.

Load the relationships

```
LOAD CSV WITH HEADERS FROM
'https://raw.githubusercontent.com/ngshya/datasets/master/cora/cora_cites.csv'
AS line FIELDTERMINATOR ','
MATCH (citing_paper:Paper {id: line.citing_paper_id}),(cited_paper:Paper {id: line.cited_paper_id})
CREATE (citing_paper)-[:CITES]→(cited_paper);
```

Table

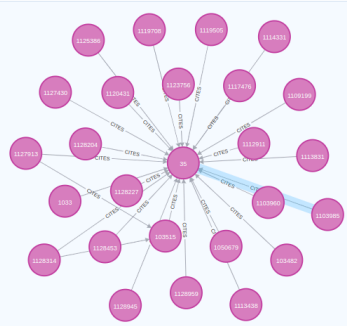
Warn

Code

Created 5429 relationships, completed after 9646 ms.

Created 5429 relationships, completed after 9646 ms.

```
neo4j$ match (a)-[r]-(b) return a, r, b limit 50;
```



Relationship properties

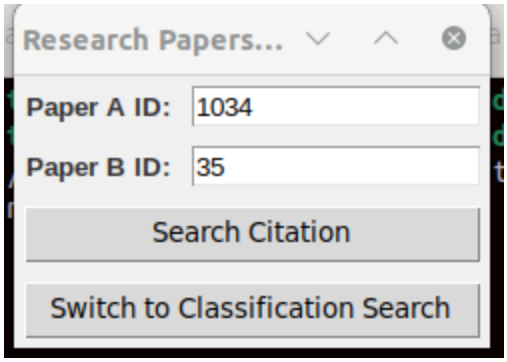
CITES

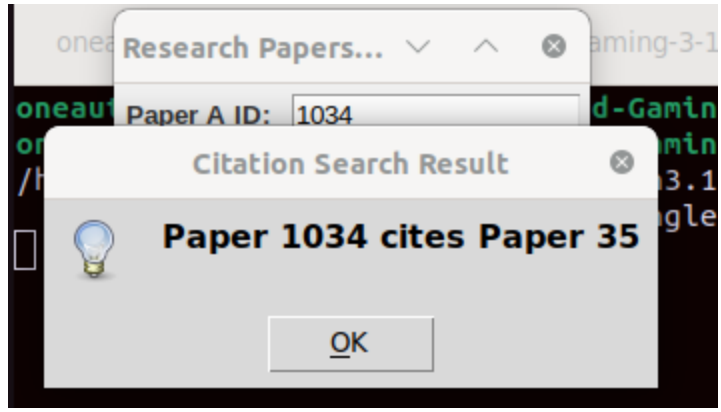
<elementId>	5:a8022d78-db08-473e-9730-1d749b48e54a:5
<id>	5

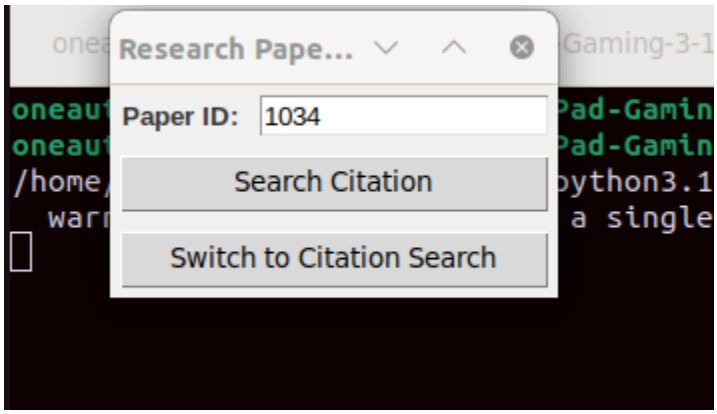
6. **Create a python desktop application**

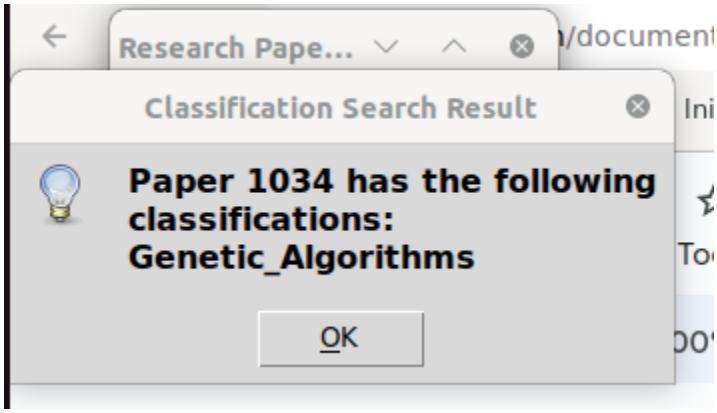
   We build a python application using tkinter and neo4j library.

   For this application we have created a new user with the username ads.


Research Papers...
Paper A ID:
1034
Paper B ID:
35
Search Citation
Switch to Classification Search


Research Papers...
Paper A ID:
1034
Citation Search Result
Paper 1034 cites Paper 35
OK


Research Pape...
Paper ID:
1034
Search Citation
Switch to Citation Search


Research Pape...
Classification Search Result
Paper 1034 has the following classifications:
Genetic_Algorithms
OK

Code

oneautumleaf@oneautumleaf-IdeaPad-Gaming-3-15IHU6: ~/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5....

```python
import tkinter as tk
from tkinter import messagebox
from tkinter import ttk
from neo4j import GraphDatabase

# Custom style class for tkinter application
class CustomStyle:
    def __init__(self):
        self.background_color = '#f0f0f0'
        self.foreground_color = '#333333'
        self.highlight_color = '#4285f4'
        self.font = ('Arial', 10)
        self.label_font = ('Arial', 10, 'bold')

# Database manager class remains unchanged
class DatabaseManager:
    def __init__(self, uri, username, password):
        self._uri = uri
        self._username = username
        self._password = password
        self._driver = None

    def connect(self):
        self._driver = GraphDatabase.driver(self._uri, auth=(self._username, self._password))

    def disconnect(self):
        if self._driver is not None:
            self._driver.close()

    def check_citation(self, paper_a_id, paper_b_id):
        with self._driver.session() as session:
            result = session.run("""
            MATCH path = (p1:Paper {id: $paper_a_id})-[:CITES*]->(p2:Paper {id: $paper_b_id})
            RETURN relationships(path) AS citations
            """, paper_a_id=paper_a_id, paper_b_id=paper_b_id)
            paths = result.single()
            if paths:
                return [(rel.start_node['id'], rel.end_node['id']) for rel in paths['citations']]
            else:
                return None

    def get_paper_classification(self, paper_id):
        with self._driver.session() as session:
            result = session.run("""
            MATCH (p:Paper {id: $paper_id})
            RETURN p.class as classification
            """, paper_id=paper_id)
            return [record['classification'] for record in result]

# Application class with custom UI elements
```

oneautumleaf@oneautumleaf-IdeaPad-Gaming-3-15IHU6: ~/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5....

```python
            MATCH (p:Paper {id: $paper_id})
            RETURN p.class as classification
            """, paper_id=paper_id)
            return [record['classification'] for record in result]

# Application class with custom UI elements
class Application(tk.Tk):
    def __init__(self, db_manager, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.title("Research Papers Database Search")
        self.db_manager = db_manager
        self.custom_style = CustomStyle()
        self.configure(background=self.custom_style.background_color)
        self.current_search = "citation"
        self.create_widgets()

    def create_widgets(self):
        self.label_paper_a = ttk.Label(self, text="Paper A ID:", font=self.custom_style.label_font, backgro
        self.label_paper_a.grid(row=0, column=0, padx=5, pady=5, sticky='w')
        self.entry_paper_a = ttk.Entry(self, font=self.custom_style.font)
        self.entry_paper_a.grid(row=0, column=1, padx=5, pady=5)

        self.label_paper_b = ttk.Label(self, text="Paper B ID:", font=self.custom_style.label_font, backgro
        self.label_paper_b.grid(row=1, column=0, padx=5, pady=5, sticky='w')
        self.entry_paper_b = ttk.Entry(self, font=self.custom_style.font)
        self.entry_paper_b.grid(row=1, column=1, padx=5, pady=5)

        self.label_paper_id = ttk.Label(self, text="Paper ID:", font=self.custom_style.label_font, backgrou
        self.label_paper_id.grid(row=2, column=0, padx=5, pady=5, sticky='w')
        self.entry_paper_id = ttk.Entry(self, font=self.custom_style.font)
        self.entry_paper_id.grid(row=2, column=1, padx=5, pady=5)

        self.btn_search = ttk.Button(self, text="Search Citation", command=self.search, style='Custom.TButt
        self.btn_search.grid(row=3, column=0, columnspan=2, padx=5, pady=5, sticky='ew')

        self.toggle_search_btn = ttk.Button(self, text="Switch to Classification Search", command=self.togg
        self.toggle_search_btn.grid(row=4, column=0, columnspan=2, padx=5, pady=5, sticky='ew')

    def toggle_search(self):
        if self.current_search == "citation":
            self.current_search = "classification"
            self.toggle_search_btn.config(text="Switch to Citation Search")
            self.label_paper_a.grid_forget()
            self.entry_paper_a.grid_forget()
            self.label_paper_b.grid_forget()
            self.entry_paper_b.grid_forget()
            self.label_paper_id.grid(row=0, column=0, padx=5, pady=5, sticky='w')
            self.entry_paper_id.grid(row=0, column=1, padx=5, pady=5)
        else:
            self.current_search = "citation"
```

oneautumleaf@oneautumleaf-IdeaPad-Gaming-3-15IHU6: ~/Desktop/COLLEGE_ASSIGNMENTS/ADS/neo4j/neo4j-community-5....

```python
        if self.current_search == "citation":
            self.current_search = "classification"
            self.toggle_search_btn.config(text="Switch to Citation Search")
            self.label_paper_a.grid_forget()
            self.entry_paper_a.grid_forget()
            self.label_paper_b.grid_forget()
            self.entry_paper_b.grid_forget()
            self.label_paper_id.grid(row=0, column=0, padx=5, pady=5, sticky='w')
            self.entry_paper_id.grid(row=0, column=1, padx=5, pady=5)
        else:
            self.current_search = "citation"
            self.toggle_search_btn.config(text="Switch to Classification Search")
            self.label_paper_a.grid(row=0, column=0, padx=5, pady=5, sticky='w')
            self.entry_paper_a.grid(row=0, column=1, padx=5, pady=5)
            self.label_paper_b.grid(row=1, column=0, padx=5, pady=5, sticky='w')
            self.entry_paper_b.grid(row=1, column=1, padx=5, pady=5)
            self.label_paper_id.grid_forget()
            self.entry_paper_id.grid_forget()

    def search(self):
        if self.current_search == "citation":
            paper_a_id = self.entry_paper_a.get()
            paper_b_id = self.entry_paper_b.get()
            cited = self.db_manager.check_citation(paper_a_id, paper_b_id)
            if cited:
                messagebox.showinfo("Citation Search Result", f"Paper {paper_a_id} cites Paper {paper_b_id}
            else:
                messagebox.showinfo("Citation Search Result", f"Paper {paper_a_id} does not cite Paper {pap
        else:
            paper_id = self.entry_paper_id.get()
            classifications = self.db_manager.get_paper_classification(paper_id)
            messagebox.showinfo("Classification Search Result", f"Paper {paper_id} has the following classi

if __name__ == "__main__":
    # Neo4j connection details
    uri = "bolt://localhost:7687"
    username = "ads"
    password = "Jay@1234"

    # Create and connect to database manager
    db_manager = DatabaseManager(uri, username, password)
    db_manager.connect()

    # Create and run the application
    app = Application(db_manager)
    app.mainloop()

    # Disconnect from the database
    db_manager.disconnect()
```

Conclusion

In conclusion, this assignment successfully implemented a Neo4j graph database for managing research papers and developed a Python-based desktop application for querying citation relationships and classifications. By combining theoretical knowledge with practical implementation, we demonstrated efficient data handling and accurate querying capabilities. Moving forward, potential enhancements include optimizing query performance and refining user interaction. Overall, this assignment enriched our understanding of graph databases and advanced database systems.

References

https://neo4j.com/download/