**CS5600: Programming Assignment (50 points)**

In this assignment, you will learn and implement CRUD(Create, Read, Update, Delete). These are the basic APIs to perform at most functions on web applications. You will develop the backend web framework using any programming languages such as Python, Java, C#, …, etc. The backend application must connect to Neo4j-a graph DBMS developed by Neo4j, inc. (https://neo4j.com). To test the backend web framework, you must use API platforms such as Postman API Platform(<https://www.postman.com>) or Swagger UI such as FastAPI(https://fastapi.tiangolo.com/).

The CSV file (IMDB-Movie-Data.csv-Movies on IMDB.com) is provided in this assignment with the column names including:

**Ids**

**Title**

**Genre**

**Description**

**Director**

**Actors**

**Year**

**Runtime (Minutes)**

**Rating**

**Votes**

**Revenue (Millions)**

First, the instructor will guide you how to generate Neo4j Database on local and cloud using Neo4j Desktop and Neo4j Aura(https://console.neo4j.io). You will import the sample csv file to create graph database on Neo4j DBMS.

This is a database model for the assignment:

**Person**

[ :ACTED\_IN ]

**Movie**

[ :DIRECTED ]

[ :IN ]

**Genres**

**Nodes:**

Movie [ ids, title, description, year, runtime, rating, votes, revenue ]

Person [ name ]

Genres [ type ]

**Relationships:**

ACTED\_IN

DIRECTED

IN

The Neo4j Database Configuration:

url = “bolt://localhost:7687” for local database

usr = “neo4j” //default username

pwd = “Admin1234” //the password

…

This is an example of configuration in Python Flask:

Text

Description automatically generated

These are the example of API interfaces in Python file:

In xxxx.py file, you must perform the following main functions:

1. Insert the new movie information.

@app.route('/imdb', methods=['POST'])

1. Update the movie information using title. (By update only title, description, and rating)

@app.route('/imdb/<string:fname>', methods=['PATCH'])

1. Delete the movie information using title.

@app.route('/imdb/<string:fname>', methods=['DELETE'])

1. Retrieve all the movies in database.

@app.route('/imdb', methods=['GET'])

1. Display the movie’s details includes actors, directors and genres using title.

@app.route('/imdb/<string:fname>', methods=['GET'])

**Submission:**

**Turn in your codes in folder name “PROG\_ASSIGN\_XXXXX\_YYYYYY” where XXXXX is course number (CRN) and YYYYYY is 700# student id and zip the folder before submitting your assignment.**

**Notes:**

**If you want to use other web frameworks such as Java, JavaScript, PHP, or Microsoft Visual Studio .NET, etc. You must contact me beforehand and let me know how to execute your web framework. You can use the internet cloud application platforms such as Heroku (**[**https://www.heroku.com**](https://www.heroku.com)**). However, in your assignment submission, you must provide me the links for execute the web application and source codes.**

LOAD CSV WITH HEADERS FROM "<https://raw.githubusercontent.com/khomsun2013/ADB/main/IMDB-Movie-Data.csv>" AS row  
WITH row.Ids AS i, row.Title AS t, row.Description AS d, row.Year AS y, row.Runtime AS r, row.Rating AS rt, row.Votes AS v, row.Revenue AS rv  
WHERE rv IS NOT NULL  
MERGE (:Movie{ids:i, title:t, description:d, year:y, runtime:r, rating:rt, votes: v, revenue:rv})

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

LOAD CSV WITH HEADERS FROM "<https://raw.githubusercontent.com/khomsun2013/ADB/main/IMDB-Movie-Data.csv>" AS row  
WITH row.Ids AS i, row.Title AS t, row.Description AS d, row.Year AS y, row.Runtime AS r, row.Rating AS rt, row.Votes AS v, row.Revenue AS rv  
WHERE rv IS NULL  
MERGE (m:Movie{ids:i, title:t, description:d, year:y, runtime:r, rating:rt, votes: v})  
ON CREATE SET m.revenue = NULL

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

LOAD CSV WITH HEADERS FROM "<https://raw.githubusercontent.com/khomsun2013/ADB/main/IMDB-Movie-Data.csv>" AS row  
WITH row.Ids AS t, SPLIT(row.Actors,',') AS a  
UNWIND t AS t1  
UNWIND a AS a1  
WITH \*, trim(a1) AS a2  
MERGE (:Person{name:a2})  
WITH \*  
MATCH(mo:Movie{ids:t1}),(ac:Person{name:a2})  
MERGE (ac)-[:ACTED\_IN]->(mo)  
RETURN ac, mo