

Graph based search

Documentation



July 16, 2019

Andy Ang Yong Kiat

Contents

[File Directory Explanation 2](#_Toc14186697)

[How to Run the Program 2](#_Toc14186698)

[How to Store Data into Neo4j 2](#_Toc14186699)

# File Directory Explanation

\***Bold** words are folders\*

**app**

* \_\_init\_\_.py – Contains all the methods
* requirements.txt  - Contains all our venv dependencies
* run.py - Where our flask app will be run from, the main method
* clustering.py - Perform document clustering and output processeddf dataframe
* processeddf - Contains clustering results in dataframe
* neo4j.py - Inserting clustering results into Neo4j
* **documents/documents2** - Contains documents scraped for clustering
* **scrapyapp** – Contains scrapyapp to scrape HR policy website
* **static**
  + **node\_modules** - Contains external modules, installed using *npm install*
  + **css/vendor** - Contains our CSS files
  + **img** - Contains images used in app
  + html files - Contains all the templates
  + package.json - Contains all our npm dependencies
  + package-lock.json - Auto generated file from package.json
* **databases** – Contains data used for Neo4j

# How to Run the Program

**Assuming Python Version**: Python 3.7.3

**Steps to Follow – Initialisation:**

1. **Open command prompt and install Virtual Environment:** pip install virtualenv
2. **Creates folder:** virtualenv venv
3. **Git Clone inside venv folder:** https://github.com/Astarcorp/app.git
4. **Activate Virtual Environment:** venv\Scripts\activate
5. **Go inside app folder:** cd app
6. **Install Requirements:** pip install -r requirements.txt
7. **Turn on Neo4j Server**
8. **Run Flask:** Python run.py
9. **You can access our webpage here:** http://localhost:5000/
10. **Download node.js:** https://nodejs.org/en/ the one on the left side (LTS)
11. **Open a second command prompt**
12. **Go inside app folder:** cd app\static
13. **Install dependencies:** npm install

# How to Store Data into Neo4j

**Neo4j Commands:**

LOAD CSV WITH HEADERS FROM 'file:///searches.csv' AS line

MATCH (node:User {username: line.username})

MATCH (p:Policy) WHERE ID(p) = toInt(line.policyId)

MERGE (node)-[s:SEARCH {numsearch: toInt(line.numsearch)}]->(p)

LOAD CSV WITH HEADERS FROM 'file:///users.csv' AS line

CREATE (:User { username: line.username, email: line.email, dateofbirth: line.dateofbirth, jobtype: line.jobtype, dateofhire: line.dateofhire, password: line.password})

CREATE (node:Image {image: './static/img/charts.png'})

MATCH (ee:Person) WHERE ee.name = "Emil" RETURN ee

MATCH (ee:PERSON)-[:KNOWS]-(friends) WHERE ee.name = "Emil" RETURN ee, friends

WITH -> Manipulate the output before it is passed on

COLLECT -> Group by

REDUCE -> Iterate in each element and run the expression

MATCH (n) DETACH DELETE n