Hash Agile Task

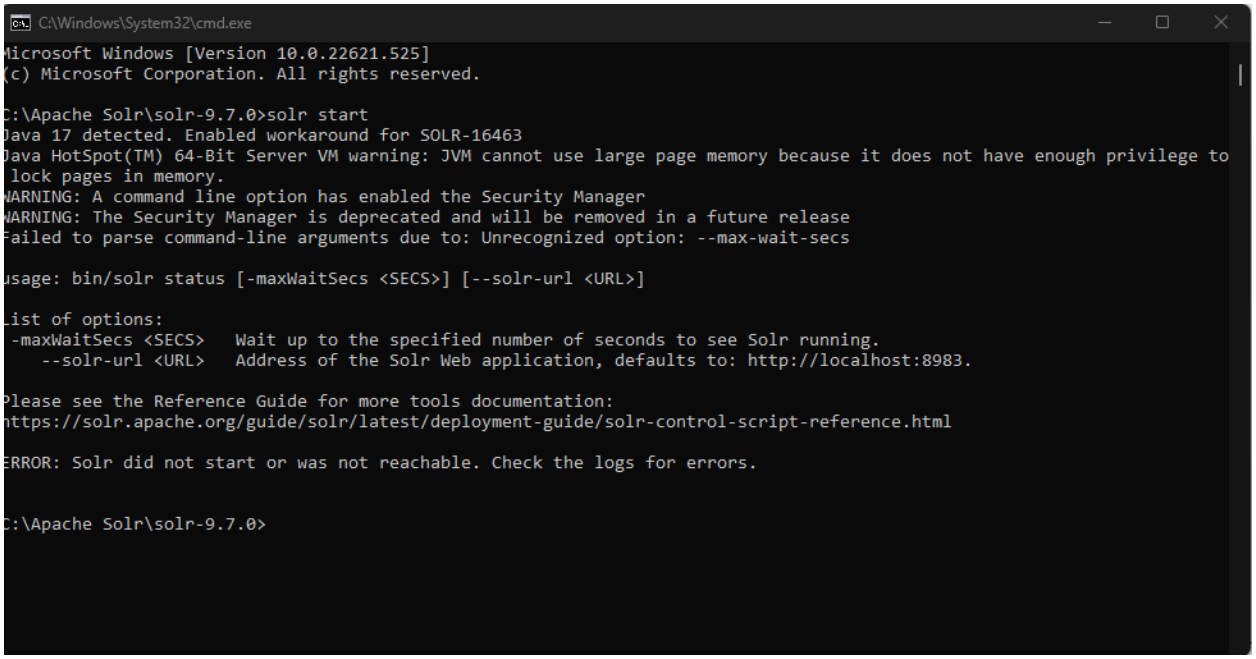
Question:

Indexing Using Apache Solr
Dataset: [Link]

Steps:

Installed Solr From Official Website

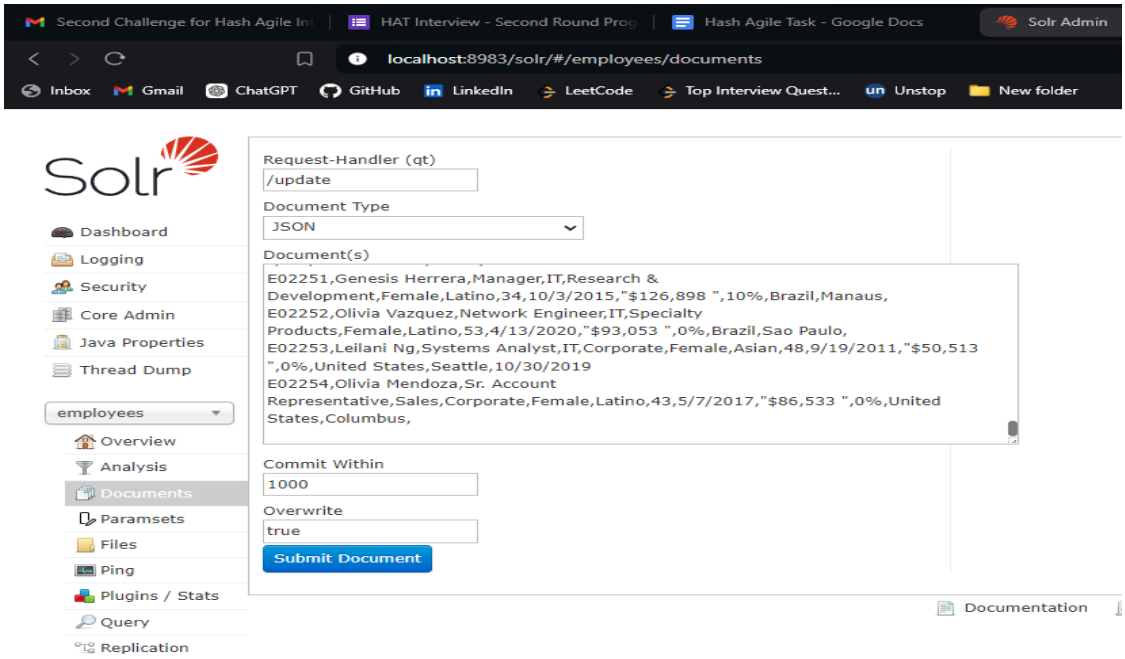
Then using cmd start The solr using solr start command



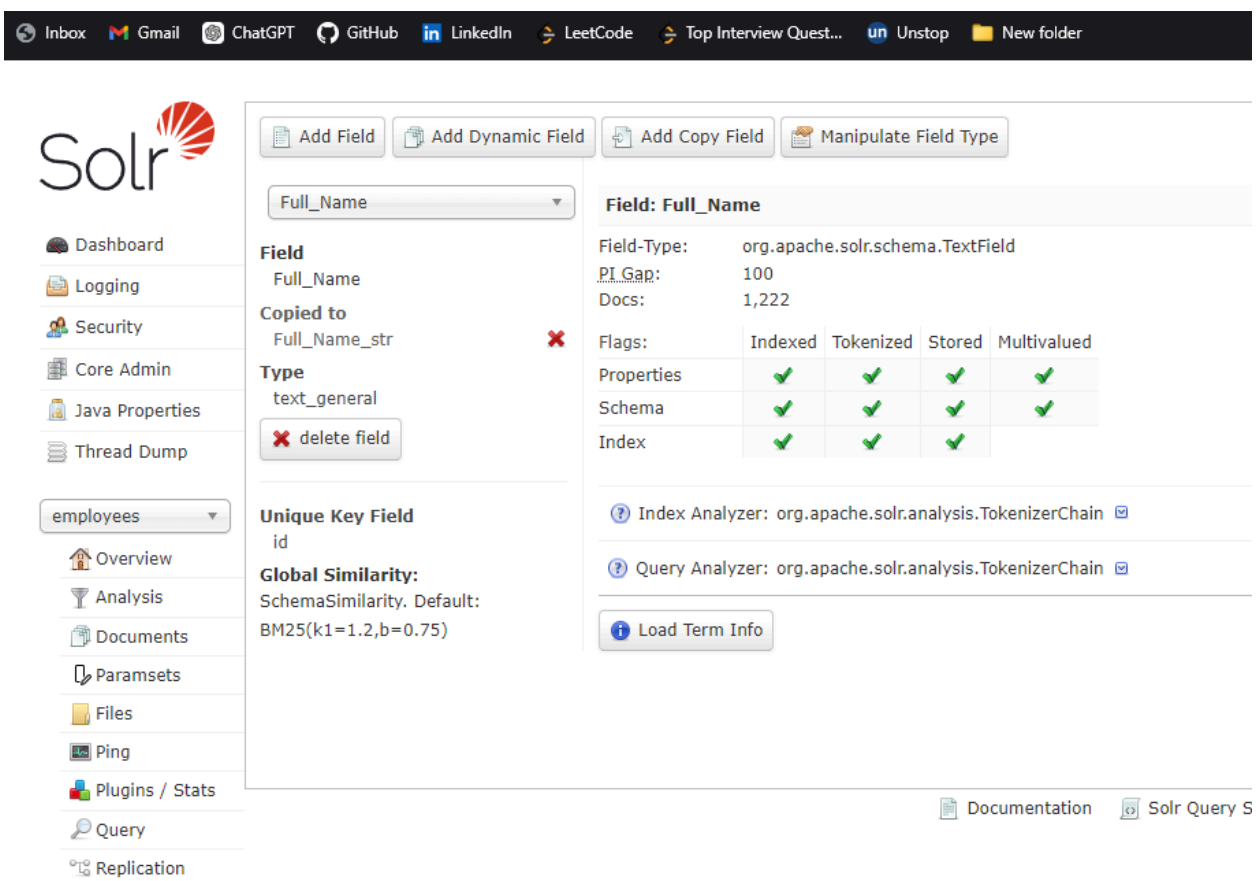
Then Go to the mentioned port [http://localhost:8983]

Then go to the documents tab to create the index for the downloaded CSV dataset file.

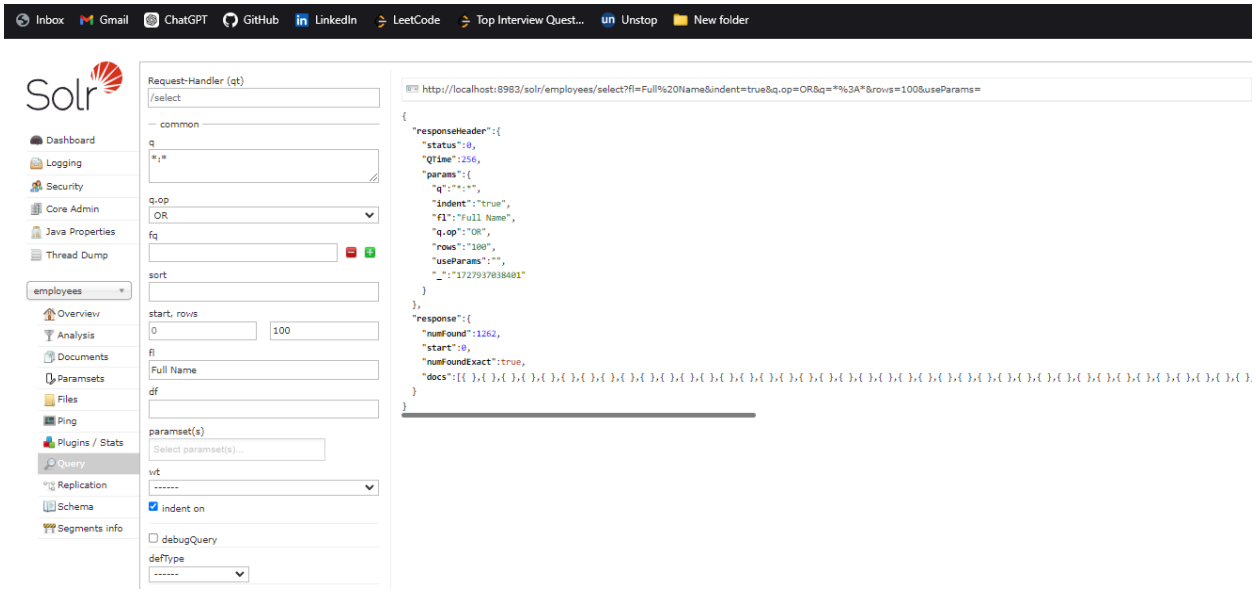
Copy the contents from the CSV file and manually paste it in the Document field and submit it.



Then it will create indexes for each field and verify them by going to schema tab and by selecting a field.



Then go to the Query tab and give appropriate fields and check the indexing working correctly.



Request-Handler (qt): /select

common

q: *:*

q.op: OR

fq:

sort:

start, rows: 0 100

fl: Full Name

df:

paramset(s): Select paramset(s)...

wt: ------

☑ indent on

☐ debugQuery

defType: ------

http://localhost:8983/solr/employees/select?fl=Full%20Name&indent=true&q.op=OR&q=*%3A*&rows=100&useParams=

```
{
  "responseHeader":{
    "status":0,
    "QTime":256,
    "params":{
      "q":"*:*",
      "indent":"true",
      "fl":"Full Name",
      "q.op":"OR",
      "rows":"100",
      "useParams":"",
      "_":"1727937038401"
    }
  },
  "response":{
    "numFound":1262,
    "start":0,
    "numFoundExact":true,
    "docs":[{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ },{ }.
  }
}
```

Then Additionally I have done an interface using Python Tkinter and retrieved the particular person data by their name field indexing the code for Python Tkinter is below.

(i have used Tkinter but At first i try to use HTML and jS but its giving CORS error, Cross-Origin Requests: When a web application tries to make requests to a server that is on a different origin (domain, protocol, or port) than its own, the browser blocks this request for security reasons unless the server explicitly allows it.) That's why I used tkinter since it is run in local desktop.So I thought it would work and it worked.

Solr_interface.py

```
import tkinter as tk
from tkinter import messagebox
import requests

# Solr server details
SOLR_URL = 'http://localhost:8983/solr/employees/select'

# Create main window
root = tk.Tk()
root.title("Employee Search")

# Create input field for search query
tk.Label(root, text="Enter Full Name:").pack(pady=10)
```

```
entry = tk.Entry(root, width=50)
entry.pack(pady=5)

# Create a text area to display results
results_text = tk.Text(root, height=20, width=80)
results_text.pack(pady=10)

# Function to search employees in Solr
def search_employees():
    query = entry.get()
    if not query:
        messagebox.showwarning("Input Error", "Please enter a name to search.")
        return

    params = {
        'q': f'Full_Name:"{query}"',  # Use quotes to handle names with spaces
        'wt': 'json',
        'indent': 'true'
    }

    try:
        response = requests.get(SOLR_URL, params=params)
        data = response.json()

        results_text.delete(1.0, tk.END)  # Clear previous results

        if data['response']['docs']:
            results = ""
            for doc in data['response']['docs']:
                # Extract fields
                full_name = doc.get('Full_Name', 'N/A')
                job_title = doc.get('Job_Title', 'N/A')
                department = doc.get('Department', 'N/A')
                business_unit = doc.get('Business_Unit', 'N/A')
                gender = doc.get('Gender', 'N/A')
                ethnicity = doc.get('Ethnicity', 'N/A')
                age = doc.get('Age', 'N/A')
                hire_date = doc.get('Hire_Date', 'N/A')
                annual_salary = doc.get('Annual_Salary', 'N/A')
                bonus = doc.get('Bonus %', 'N/A')
                country = doc.get('Country', 'N/A')
                city = doc.get('City', 'N/A')
                exit_date = doc.get('Exit_Date', 'N/A')
```

```
            # Format the result string
            results += (
                f"Full Name: {full_name}, Job Title: {job_title}, Department: {department}, "
                f"Business Unit: {business_unit}, Gender: {gender}, Ethnicity: {ethnicity}, "
                f"Age: {age}, Hire Date: {hire_date}, Annual Salary: {annual_salary}, "
                f"Bonus %: {bonus}, Country: {country}, City: {city}, Exit Date: {exit_date}\n"
            )
        results_text.insert(tk.END, results)
    else:
        results_text.insert(tk.END, "No results found.")
  except Exception as e:
    messagebox.showerror("Error", f"An error occurred: {e}")

# Create search button
search_button = tk.Button(root, text="Search", command=search_employees)
search_button.pack(pady=10)

# Run the application
root.mainloop()
```

Explanation(Main Functionality)

User Interface:

Input Field: A text box to enter the employee's full name.

Search Button: A button to initiate the search.

Functionality:

When the user clicks the Search button:

The app sends a request to a Solr server to find the employee by name.

It uses the entered name to search the employee database.

Display Results:

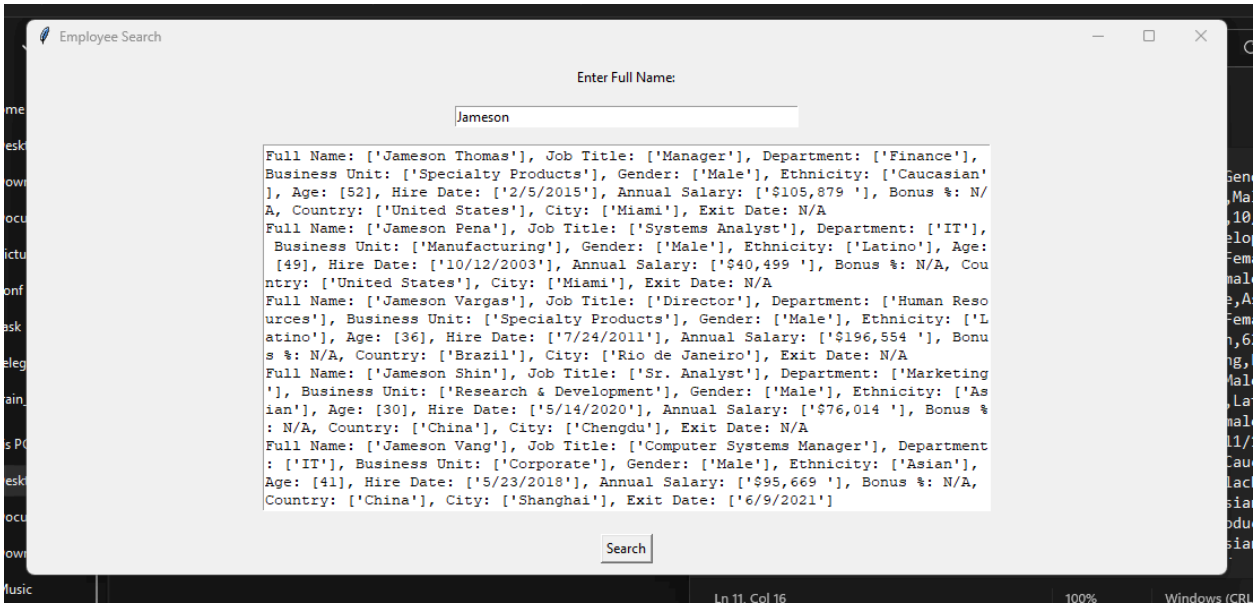
If an employee is found:

Their details (like job title, department, salary, etc.) are displayed.

If no employee is found:

A message indicates that no results were found.

Output:



I understand Apache Solr is an open-source search platform that enables fast and powerful full-text search and indexing of large datasets. It is designed for scalability and can handle complex search queries efficiently.

Whenever there is a query of data, Apache Solr says Hi :)