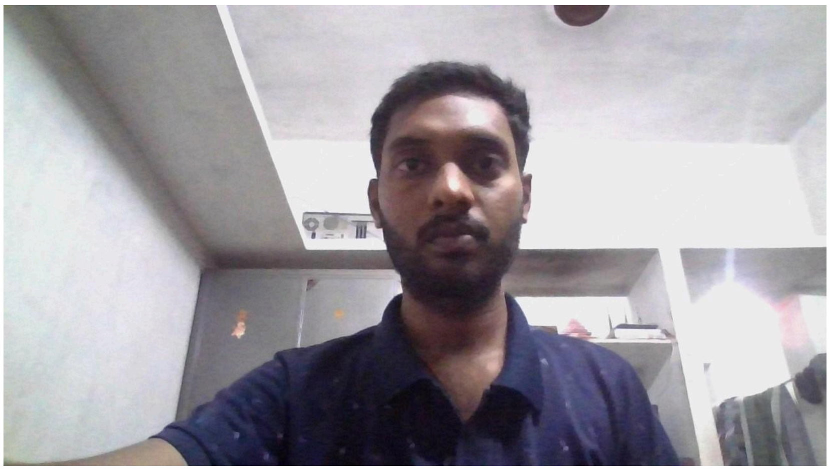
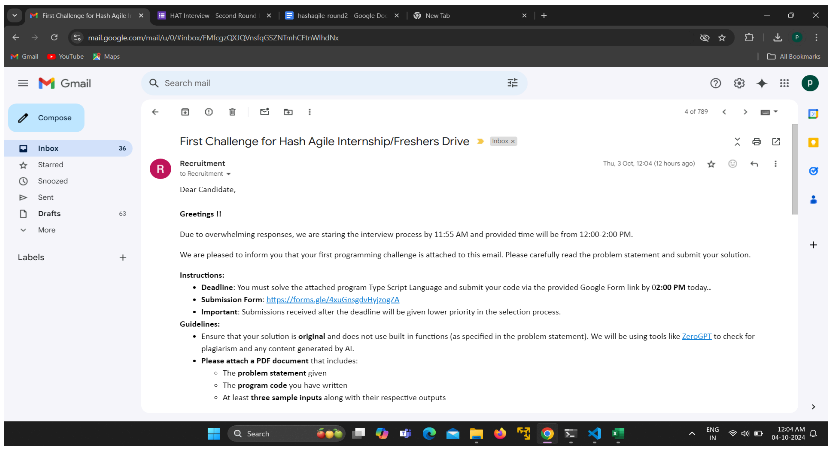
Hashagile Round-2 [submission]

A. Full Name: Venkata Ramprasad Pade

B. Selfie:



C. First Round Mail:



First Challenge for Hash Agile Internship/Freshers Drive

Recruitment
to Recruitment

Thu, 3 Oct, 12:04 (12 hours ago)

Dear Candidate,

Greetings !!

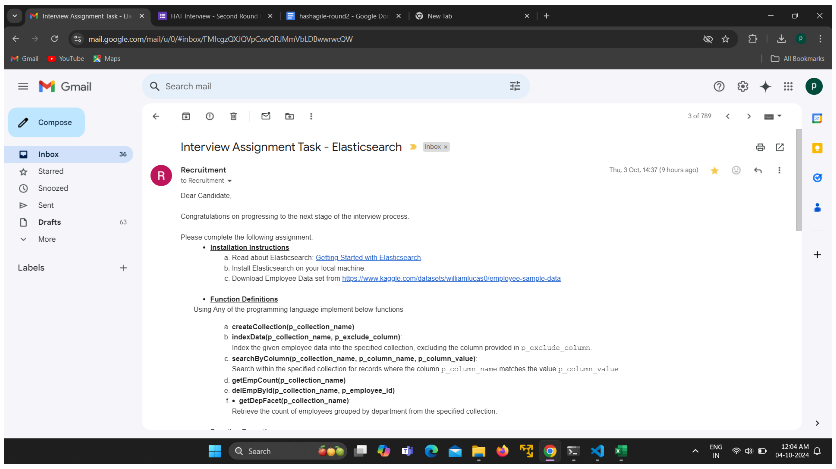
Due to overwhelming responses, we are staring the interview process by 11:55 AM and provided time will be from 12:00-2:00 PM.

We are pleased to inform you that your first programming challenge is attached to this email. Please carefully read the problem statement and submit your solution.

Instructions:
- **Deadline**: You must solve the attached program Type Script Language and submit your code via the provided Google Form link by 0**2:00 PM** today..
- **Submission Form**: https://forms.gle/4xuGnsgdvHyjzogZA
- **Important**: Submissions received after the deadline will be given lower priority in the selection process.

Guidelines:
- Ensure that your solution is **original** and does not use built-in functions (as specified in the problem statement). We will be using tools like ZeroGPT to check for plagiarism and any content generated by AI.
- **Please attach a PDF document** that includes:
  - The **problem statement** given
  - The **program code** you have written
  - At least **three sample inputs** along with their respective outputs

D. Second Round Mail



Interview Assignment Task - Elasticsearch

Recruitment
to Recruitment

Thu, 3 Oct, 14:37 (9 hours ago)

Dear Candidate,

Congratulations on progressing to the next stage of the interview process.

Please complete the following assignment:
- **Installation Instructions**
  a. Read about Elasticsearch: Getting Started with Elasticsearch.
  b. Install Elasticsearch on your local machine.
  c. Download Employee Data set from https://www.kaggle.com/datasets/williamlucas0/employee-sample-data

- **Function Definitions**
  Using Any of the programming language implement below functions

  a. **createCollection(p_collection_name)**
  b. **indexData(p_collection_name, p_exclude_column)**:
  Index the given employee data into the specified collection, excluding the column provided in `p_exclude_column`.
  c. **searchByColumn(p_collection_name, p_column_name, p_column_value)**:
  Search within the specified collection for records where the column `p_column_name` matches the value `p_column_value`.
  d. **getEmpCount(p_collection_name)**
  e. **delEmpById(p_collection_name, p_employee_id)**
  f. • **getDepFacet(p_collection_name)**:
  Retrieve the count of employees grouped by department from the specified collection.

E. Github Url for first round:
https://github.com/ramprasad-13/hashagile-round1

F. Github Url for second round:
https://github.com/ramprasad-13/hashagile-round2

Problem Statement

- **Installation Instructions**
  1. Read about Elasticsearch: Getting Started with Elasticsearch.
  2. Install Elasticsearch on your local machine.
  3. Download Employee Data set from https://www.kaggle.com/datasets/williamlucas0/employee-sample-data

- **Function Definitions**

Using Any of the programming language implement below functions

1. **createCollection(p_collection_name)**
2. **indexData(p_collection_name, p_exclude_column)**:
   Index the given employee data into the specified collection, excluding the column provided in `p_exclude_column`.
3. **searchByColumn(p_collection_name, p_column_name, p_column_value)**:
   Search within the specified collection for records where the column `p_column_name` matches the value `p_column_value`.
4. **getEmpCount(p_collection_name)**
5. **delEmpById(p_collection_name, p_employee_id)**
6. · **getDepFacet(p_collection_name)**:
   Retrieve the count of employees grouped by department from the specified collection.

- **Function Executions**

Once the functions are implemented, execute the functions in the given order with the parameters mentioned

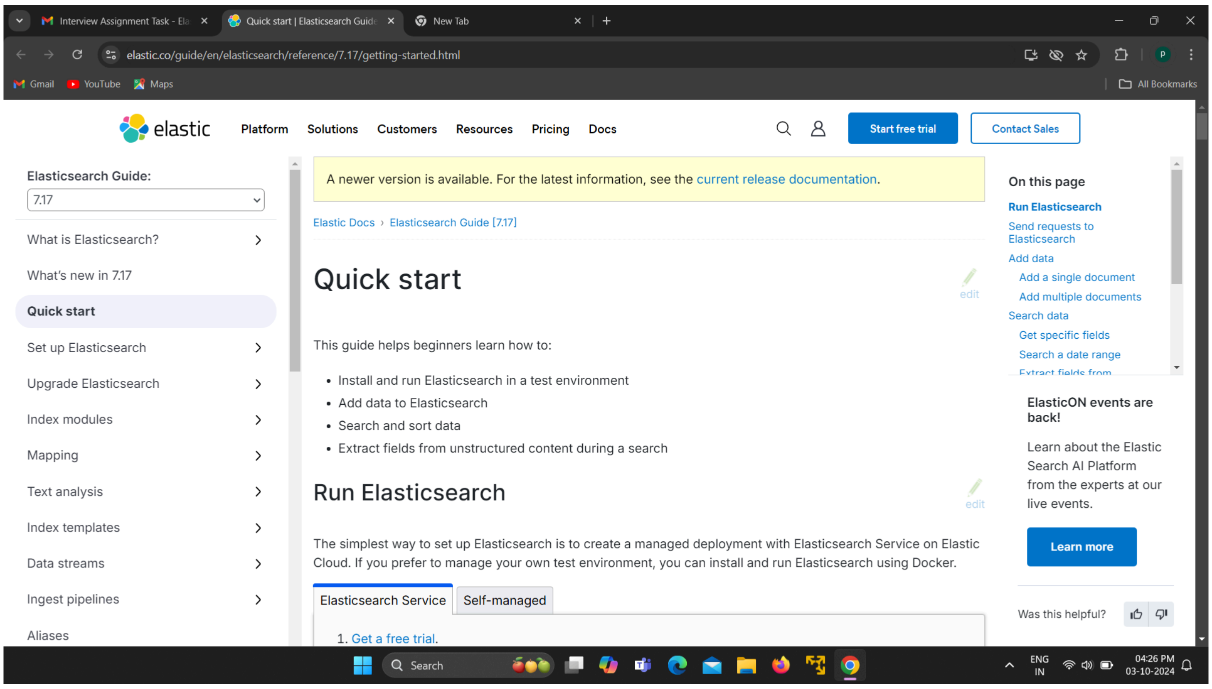
1. Var v_nameCollection = 'Hash_<Your Name>'
2. Var v_phoneCollection ='Hash_<Your Phone last four digits'
3. createCollection(v_nameCollection)
4. createCollection(v_phoneCollection)
5. getEmpCount(v_nameCollection)
6. indexData(**v**_nameCollection,'Department')
7. indexData(**v_** phoneCollection, 'Gender')
8. delEmpById (v_ nameCollection ,'E02003')
9. getEmpCount(v_nameCollection)
10. searchByColumn(v_nameCollection,'Department','IT')
11. searchByColumn(v_nameCollection,'Gender' ,'Male')
12. searchByColumn(v_ phoneCollection,'Department','IT')
13. getDepFacet(v_ nameCollection)
14. getDepFacet(v_ phoneCollection)

Assignment - Result(step-by-step):

- **Installation Instructions**
  1. Read about Elasticsearch: Getting Started with Elasticsearch.

2. Install Elasticsearch on your local machine.
3. Download Employee Data set from https://www.kaggle.com/datasets/williamlucas0/employee-sample-data

Getting started with ElasticSearch:


Interview Assignment Task - Ela | Quick start | Elasticsearch Guid | New Tab

elastic.co/guide/en/elasticsearch/reference/7.17/getting-started.html

Gmail YouTube Maps All Bookmarks

elastic Platform Solutions Customers Resources Pricing Docs Start free trial Contact Sales

Elasticsearch Guide:
7.17
What is Elasticsearch?
What's new in 7.17
Quick start
Set up Elasticsearch
Upgrade Elasticsearch
Index modules
Mapping
Text analysis
Index templates
Data streams
Ingest pipelines
Aliases

A newer version is available. For the latest information, see the current release documentation.

Elastic Docs › Elasticsearch Guide [7.17]

Quick start

This guide helps beginners learn how to:

- Install and run Elasticsearch in a test environment
- Add data to Elasticsearch
- Search and sort data
- Extract fields from unstructured content during a search

Run Elasticsearch

The simplest way to set up Elasticsearch is to create a managed deployment with Elasticsearch Service on Elastic Cloud. If you prefer to manage your own test environment, you can install and run Elasticsearch using Docker.

Elasticsearch Service | Self-managed

1. Get a free trial.

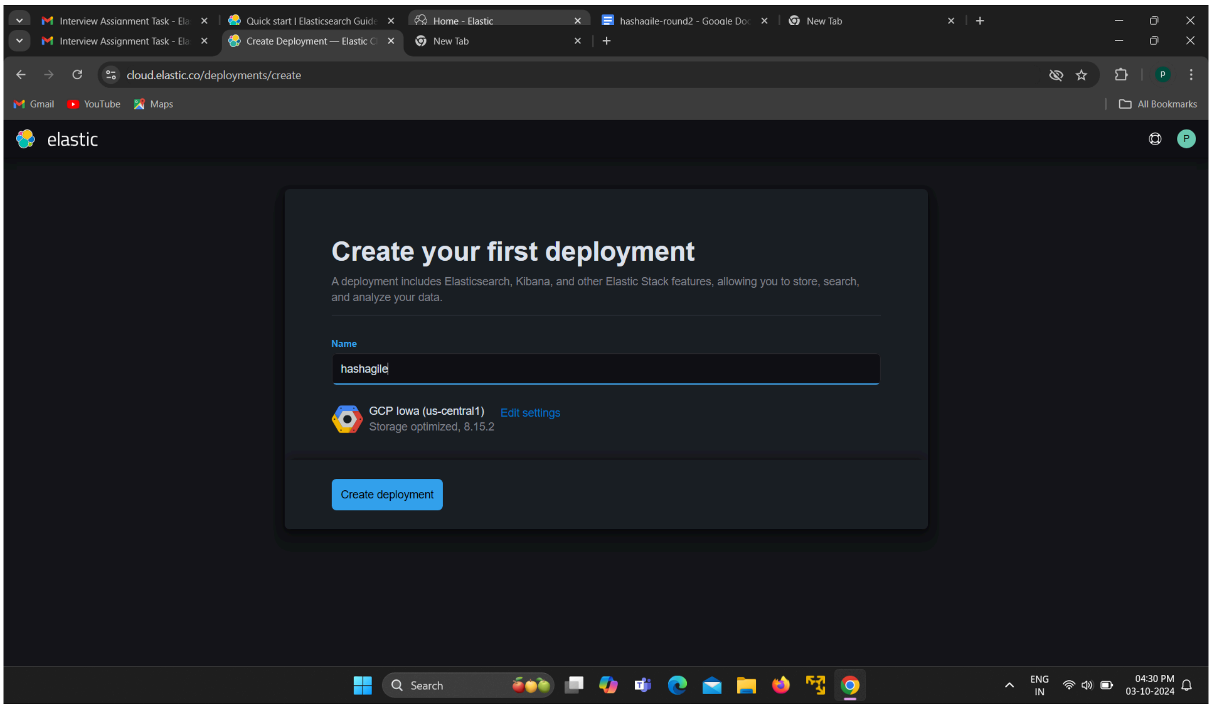
On this page
Run Elasticsearch
Send requests to Elasticsearch
Add data
Add a single document
Add multiple documents
Search data
Get specific fields
Search a date range
Extract fields from

ElasticON events are back!
Learn about the Elastic Search AI Platform from the experts at our live events.
Learn more

Was this helpful?

04:26 PM 03-10-2024

Making use of Elastic cloud(Alternative to local installation)


Interview Assignment Task - Ela | Quick start | Elasticsearch Guid | Home - Elastic | hashagile-round2 - Google Doc | New Tab
Interview Assignment Task - Ela | Create Deployment — Elastic C | New Tab

cloud.elastic.co/deployments/create

Gmail YouTube Maps All Bookmarks

elastic

Create your first deployment

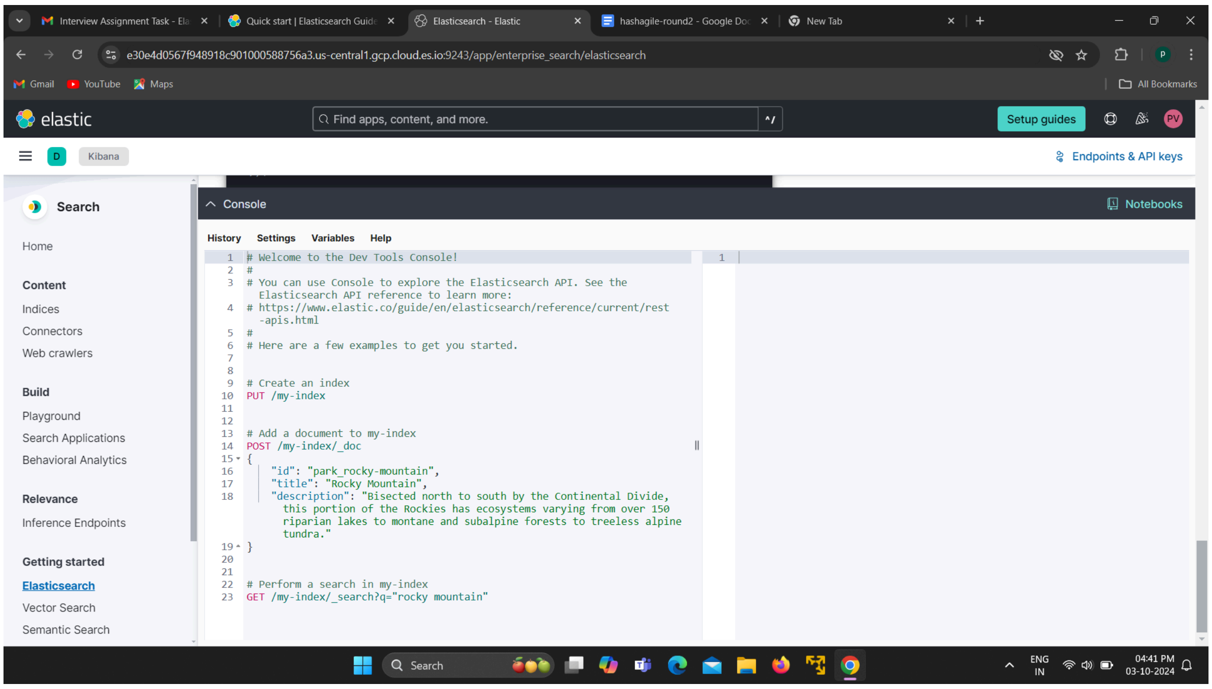
A deployment includes Elasticsearch, Kibana, and other Elastic Stack features, allowing you to store, search, and analyze your data.

Name
hashagile

GCP Iowa (us-central1) Edit settings
Storage optimized, 8.15.2

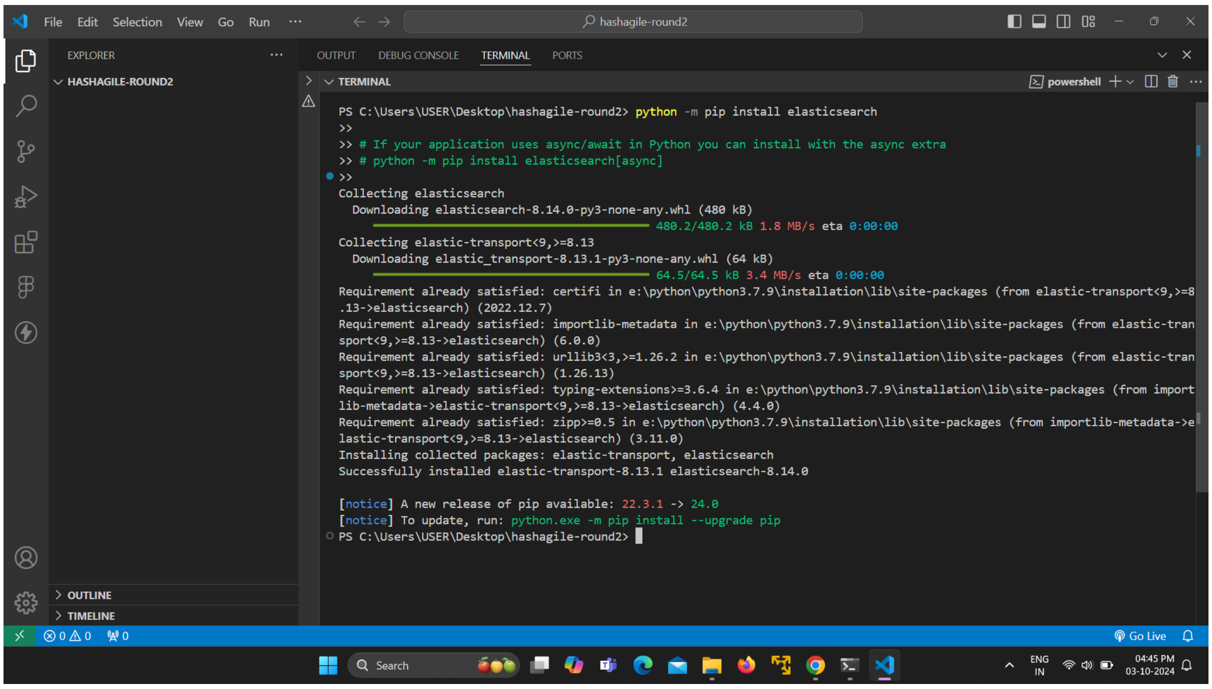
Create deployment

04:30 PM 03-10-2024

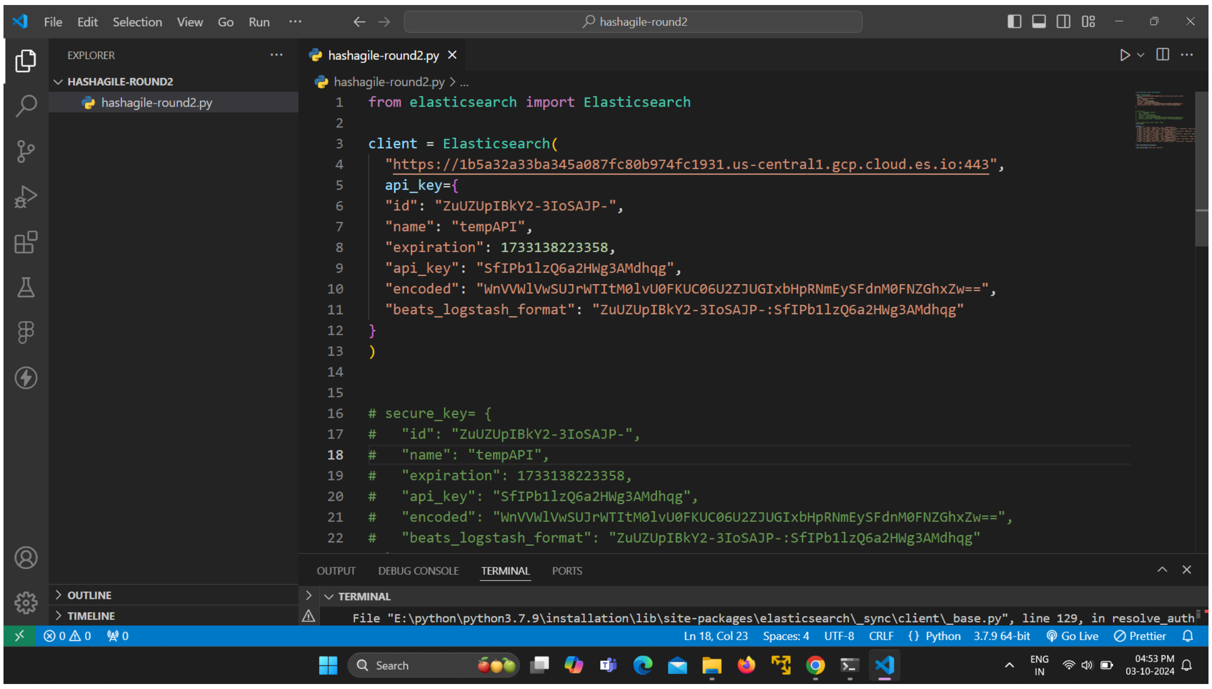


```
# Welcome to the Dev Tools Console!
#
# You can use Console to explore the Elasticsearch API. See the
  Elasticsearch API reference to learn more:
# https://www.elastic.co/guide/en/elasticsearch/reference/current/rest
  -apis.html
#
# Here are a few examples to get you started.


# Create an index
PUT /my-index


# Add a document to my-index
POST /my-index/_doc
{
    "id": "park_rocky-mountain",
    "title": "Rocky Mountain",
    "description": "Bisected north to south by the Continental Divide,
      this portion of the Rockies has ecosystems varying from over 150
      riparian lakes to montane and subalpine forests to treeless alpine
      tundra."
}


# Perform a search in my-index
GET /my-index/_search?q="rocky mountain"
```

Using this default code i am trying to stimulate in my local machine

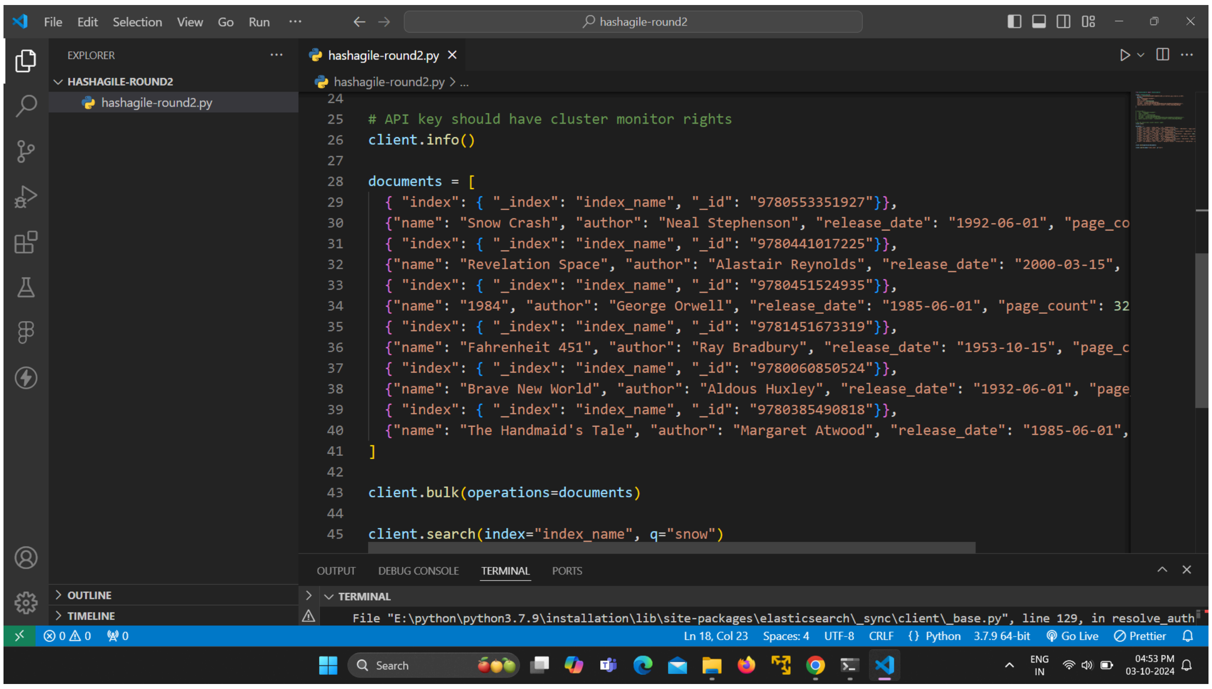


```
PS C:\Users\USER\Desktop\hashagile-round2> python -m pip install elasticsearch
>>
>> # If your application uses async/await in Python you can install with the async extra
>> # python -m pip install elasticsearch[async]
>>
Collecting elasticsearch
  Downloading elasticsearch-8.14.0-py3-none-any.whl (480 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 480.2/480.2 kB 1.8 MB/s eta 0:00:00
Collecting elastic-transport<9,>=8.13
  Downloading elastic_transport-8.13.1-py3-none-any.whl (64 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 64.5/64.5 kB 3.4 MB/s eta 0:00:00
Requirement already satisfied: certifi in e:\python\python3.7.9\installation\lib\site-packages (from elastic-transport<9,>=8
.13->elasticsearch) (2022.12.7)
Requirement already satisfied: importlib-metadata in e:\python\python3.7.9\installation\lib\site-packages (from elastic-tran
sport<9,>=8.13->elasticsearch) (6.0.0)
Requirement already satisfied: urllib3<3,>=1.26.2 in e:\python\python3.7.9\installation\lib\site-packages (from elastic-tran
sport<9,>=8.13->elasticsearch) (1.26.13)
Requirement already satisfied: typing-extensions>=3.6.4 in e:\python\python3.7.9\installation\lib\site-packages (from import
lib-metadata->elastic-transport<9,>=8.13->elasticsearch) (4.4.0)
Requirement already satisfied: zipp>=0.5 in e:\python\python3.7.9\installation\lib\site-packages (from importlib-metadata->e
lastic-transport<9,>=8.13->elasticsearch) (3.11.0)
Installing collected packages: elastic-transport, elasticsearch
Successfully installed elastic-transport-8.13.1 elasticsearch-8.14.0

[notice] A new release of pip available: 22.3.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
PS C:\Users\USER\Desktop\hashagile-round2>
```



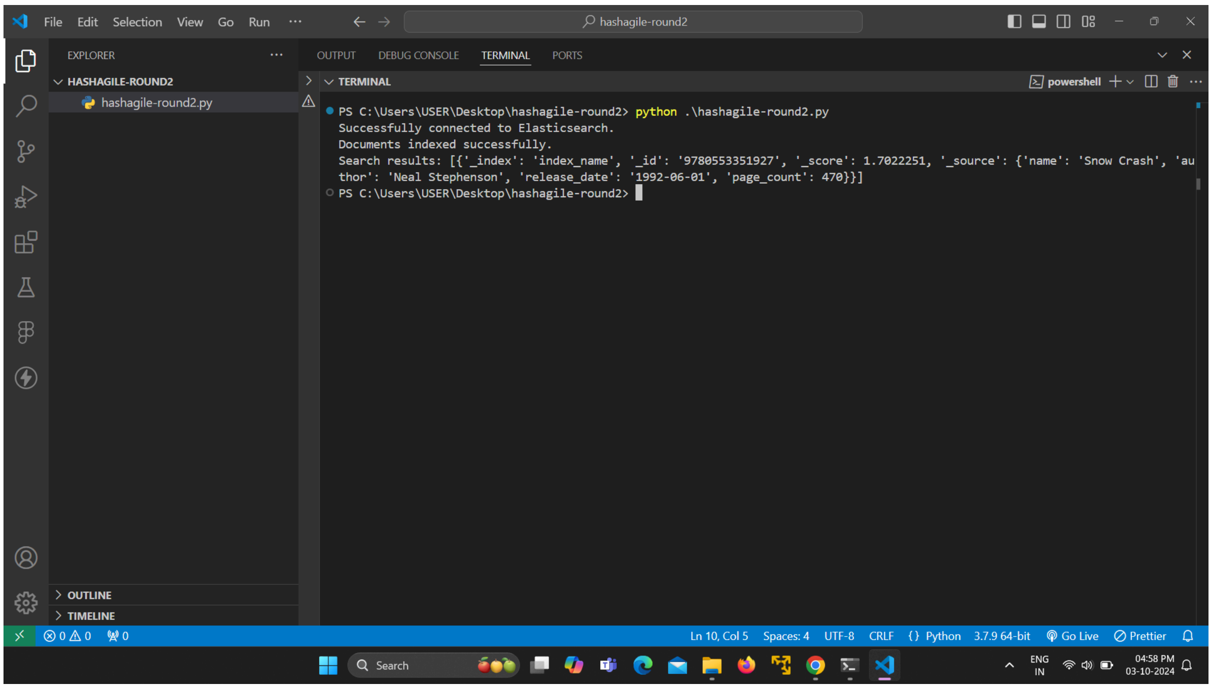
```python
from elasticsearch import Elasticsearch

client = Elasticsearch(
  "https://1b5a32a33ba345a087fc80b974fc1931.us-central1.gcp.cloud.es.io:443",
  api_key={
  "id": "ZuUZUpIBkY2-3IoSAJP-",
  "name": "tempAPI",
  "expiration": 1733138223358,
  "api_key": "SfIPb1lzQ6a2HWg3AMdhqg",
  "encoded": "WnVVWlVwSUJrWTItM0lvU0FKUC06U2ZJUGIxbHpRNmEySFdnM0FNZGhxZw==",
  "beats_logstash_format": "ZuUZUpIBkY2-3IoSAJP-:SfIPb1lzQ6a2HWg3AMdhqg"
}
)


# secure_key= {
#   "id": "ZuUZUpIBkY2-3IoSAJP-",
#   "name": "tempAPI",
#   "expiration": 1733138223358,
#   "api_key": "SfIPb1lzQ6a2HWg3AMdhqg",
#   "encoded": "WnVVWlVwSUJrWTItM0lvU0FKUC06U2ZJUGIxbHpRNmEySFdnM0FNZGhxZw==",
#   "beats_logstash_format": "ZuUZUpIBkY2-3IoSAJP-:SfIPb1lzQ6a2HWg3AMdhqg"
```

```
File "E:\python\python3.7.9\installation\lib\site-packages\elasticsearch\_sync\client\_base.py", line 129, in resolve_auth
```



```python

# API key should have cluster monitor rights
client.info()

documents = [
  { "index": { "_index": "index_name", "_id": "9780553351927"}},
  {"name": "Snow Crash", "author": "Neal Stephenson", "release_date": "1992-06-01", "page_co
  { "index": { "_index": "index_name", "_id": "9780441017225"}},
  {"name": "Revelation Space", "author": "Alastair Reynolds", "release_date": "2000-03-15",
  { "index": { "_index": "index_name", "_id": "9780451524935"}},
  {"name": "1984", "author": "George Orwell", "release_date": "1985-06-01", "page_count": 32
  { "index": { "_index": "index_name", "_id": "9781451673319"}},
  {"name": "Fahrenheit 451", "author": "Ray Bradbury", "release_date": "1953-10-15", "page_c
  { "index": { "_index": "index_name", "_id": "9780060850524"}},
  {"name": "Brave New World", "author": "Aldous Huxley", "release_date": "1932-06-01", "page
  { "index": { "_index": "index_name", "_id": "9780385490818"}},
  {"name": "The Handmaid's Tale", "author": "Margaret Atwood", "release_date": "1985-06-01",
]

client.bulk(operations=documents)

client.search(index="index_name", q="snow")
```

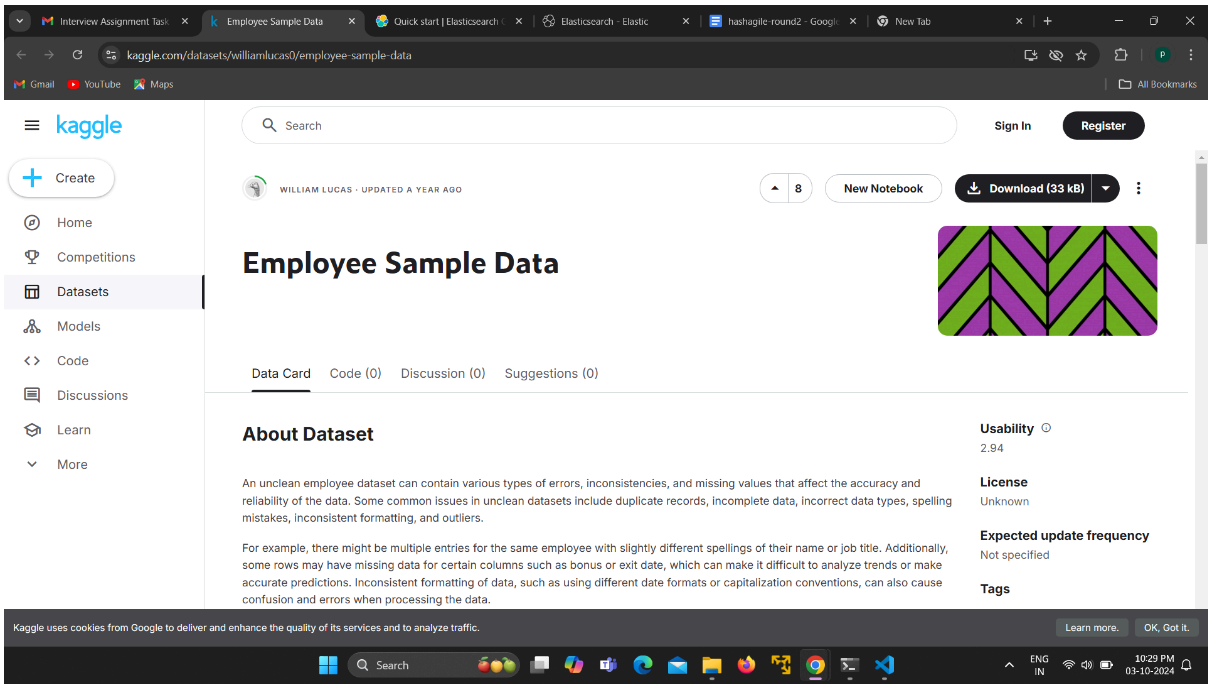
```
File "E:\python\python3.7.9\installation\lib\site-packages\elasticsearch\_sync\client\_base.py", line 129, in resolve_auth
```



```
PS C:\Users\USER\Desktop\hashagile-round2> python .\hashagile-round2.py
Successfully connected to Elasticsearch.
Documents indexed successfully.
Search results: [{'_index': 'index_name', '_id': '9780553351927', '_score': 1.7022251, '_source': {'name': 'Snow Crash', 'au
thor': 'Neal Stephenson', 'release_date': '1992-06-01', 'page_count': 470}}]
PS C:\Users\USER\Desktop\hashagile-round2>
```

File Edit Selection View Go Run
hashagile-round2
EXPLORER
HASHAGILE-ROUND2
hashagile-round2.py
OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
OUTLINE
TIMELINE
Ln 10, Col 5 Spaces: 4 UTF-8 CRLF Python 3.7.9 64-bit Go Live Prettier
04:58 PM 03-10-2024

C. Downloading employee dataset from kaggle



kaggle.com/datasets/williamlucas0/employee-sample-data

WILLIAM LUCAS · UPDATED A YEAR AGO

Employee Sample Data

Data Card Code (0) Discussion (0) Suggestions (0)

About Dataset

An unclean employee dataset can contain various types of errors, inconsistencies, and missing values that affect the accuracy and reliability of the data. Some common issues in unclean datasets include duplicate records, incomplete data, incorrect data types, spelling mistakes, inconsistent formatting, and outliers.

For example, there might be multiple entries for the same employee with slightly different spellings of their name or job title. Additionally, some rows may have missing data for certain columns such as bonus or exit date, which can make it difficult to analyze trends or make accurate predictions. Inconsistent formatting of data, such as using different date formats or capitalization conventions, can also cause confusion and errors when processing the data.

Usability
2.94

License
Unknown

Expected update frequency
Not specified

Tags

Kaggle uses cookies from Google to deliver and enhance the quality of its services and to analyze traffic.

Extract csv file from archive.zip and place in hashagile folder along with code file

Function Definitions:

1. **createCollection(p_collection_name):**

```
def create_collection(collection_name):
    if not client.indices.exists(index=collection_name):
        client.indices.create(index=collection_name)
        print(f"Collection '{collection_name}' created.")
    else:
        print(f"Collection '{collection_name}' already exists.")
```

2. **indexData(p_collection_name, p_exclude_column)**:

```
def index_data(collection_name, exclude_column):
    # Load the employee data
    try:
        df = pd.read_csv('employee_data.csv', encoding='ISO-8859-1')  # Adjust
encoding as needed
    except Exception as e:
        print(f"Error reading CSV file: {e}")
        return

    df = df.drop(columns=[exclude_column])  # Exclude the specified column
    df.fillna('', inplace=True)  # Replace NaNs with empty strings

    # Prepare documents for indexing
    documents = df.to_dict(orient='records')
    print(f"Preparing to index {len(documents)} documents in '{collection_name}'.")

    # Index data
    actions = []
    for i, doc in enumerate(documents):
        action = {
            '_op_type': 'index',
            '_index': collection_name,
            '_id': str(i),
            '_source': doc
        }
        actions.append(action)

    # Bulk index documents
    try:
        helpers.bulk(client, actions)
        print(f"Data indexed in '{collection_name}' excluding column
'{exclude_column}'.")
```

```
    except helpers.BulkIndexError as e:
        print(f"Error indexing documents: {e.errors}")
```

Index the given employee data into the specified collection, excluding the column provided in `p_exclude_column`.

3. **searchByColumn(p_collection_name, p_column_name, p_column_value)**: Search within the specified collection for records where the column `p_column_name` matches the value `p_column_value`.

```
def search_by_column(collection_name, column_name, column_value):
    query = {
        "query": {
            "match": {
                column_name: column_value
            }
        }
    }
    results = client.search(index=collection_name, body=query)
    return results['hits']['hits']
```

4. **getEmpCount(p_collection_name):**

```
def get_emp_count(collection_name):
    count = client.count(index=collection_name)
    return count['count']
```

5. **delEmpById(p_collection_name, p_employee_id):**

```
def del_emp_by_id(collection_name, employee_id):
    try:
        client.delete(index=collection_name, id=employee_id)
        print(f"Employee with ID '{employee_id}' deleted from '{collection_name}'.")
    except Exception as e:
        print(f"Error deleting employee: {e}")
```

6. **· getDepFacet(p_collection_name)**:

```
def get_dep_facet(collection_name):
    query = {
        "size": 0,
        "aggs": {
            "departments": {
                "terms": {
                    "field": "Department.keyword"
                }
            }
        }
    }
```

```
result = client.search(index=collection_name, body=query)
return result['aggregations']['departments']['buckets']
```

Function Execution:

1. Var v_nameCollection = ‘Hash_<Your Name>’
   name_collection = 'hash_ram’

2. Var v_phoneCollection =’Hash_<Your Phone last four digits’
   phone_collection = 'hash_2179'

3. createCollection(v_nameCollection)
   create_collection(name_collection)

4. createCollection(v_phoneCollection)
   create_collection(phone_collection)

5. getEmpCount(v_nameCollection)
   print("Employee Count in Name Collection:",
   get_emp_count(name_collection))

6. indexData**(v**_nameCollection,’Department’)
   index_data(name_collection, 'Department')

7. indexData**(v**_ phoneCollection, ‘Gender’)
   index_data(phone_collection, 'Gender')

8. delEmpById (v_ nameCollection ,‘E02003’)
   del_emp_by_id(name_collection, 'E02003')

9. getEmpCount(v_nameCollection)
   print("Employee Count in Name Collection after deletion:",
   get_emp_count(name_collection))

10. searchByColumn(v_nameCollection,’Department’,’IT’)
    print("Search results for 'IT' in Department:",
    search_by_column(name_collection, 'Department', 'IT'))

11. searchByColumn(v_nameCollection,’Gender’ ,’Male’)
    print("Search results for 'Male' in Gender:",
    search_by_column(name_collection, 'Gender', 'Male'))

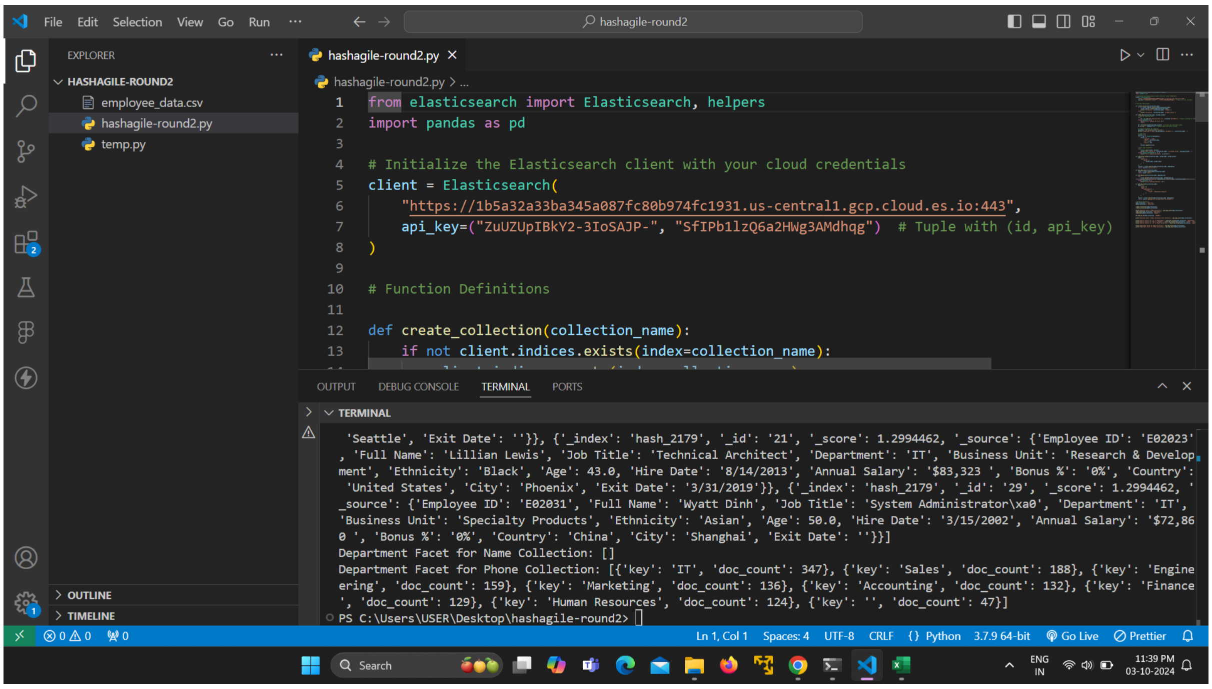
12. searchByColumn(v_ phoneCollection,’Department’,’IT’)
    print("Search results for 'IT' in Phone Collection:",
    search_by_column(phone_collection, 'Department', 'IT'))

13. getDepFacet(v_ nameCollection)

print("Department Facet for Name Collection:",
get_dep_facet(name_collection))

14. getDepFacet(v_ phoneCollection)
print("Department Facet for Phone Collection:",
get_dep_facet(phone_collection))

Result:



```python
from elasticsearch import Elasticsearch, helpers
import pandas as pd

# Initialize the Elasticsearch client with your cloud credentials
client = Elasticsearch(
    "https://1b5a32a33ba345a087fc80b974fc1931.us-central1.gcp.cloud.es.io:443",
    api_key=("ZuUZUpIBkY2-3IoSAJP-", "SfIPb1lzQ6a2HWg3AMdhqg")  # Tuple with (id, api_key)
)

# Function Definitions

def create_collection(collection_name):
    if not client.indices.exists(index=collection_name):
```

```
 'Seattle', 'Exit Date': ''}}, {'_index': 'hash_2179', '_id': '21', '_score': 1.2994462, '_source': {'Employee ID': 'E02023'
, 'Full Name': 'Lillian Lewis', 'Job Title': 'Technical Architect', 'Department': 'IT', 'Business Unit': 'Research & Develop
ment', 'Ethnicity': 'Black', 'Age': 43.0, 'Hire Date': '8/14/2013', 'Annual Salary': '$83,323 ', 'Bonus %': '0%', 'Country':
 'United States', 'City': 'Phoenix', 'Exit Date': '3/31/2019'}}, {'_index': 'hash_2179', '_id': '29', '_score': 1.2994462, '
_source': {'Employee ID': 'E02031', 'Full Name': 'Wyatt Dinh', 'Job Title': 'System Administrator\xa0', 'Department': 'IT',
'Business Unit': 'Specialty Products', 'Ethnicity': 'Asian', 'Age': 50.0, 'Hire Date': '3/15/2002', 'Annual Salary': '$72,86
0 ', 'Bonus %': '0%', 'Country': 'China', 'City': 'Shanghai', 'Exit Date': ''}}]
Department Facet for Name Collection: []
Department Facet for Phone Collection: [{'key': 'IT', 'doc_count': 347}, {'key': 'Sales', 'doc_count': 188}, {'key': 'Engine
ering', 'doc_count': 159}, {'key': 'Marketing', 'doc_count': 136}, {'key': 'Accounting', 'doc_count': 132}, {'key': 'Finance
', 'doc_count': 129}, {'key': 'Human Resources', 'doc_count': 124}, {'key': '', 'doc_count': 47}]
PS C:\Users\USER\Desktop\hashagile-round2>
```

File Edit Selection View Go Run ···
hashagile-round2
EXPLORER
HASHAGILE-ROUND2
employee_data.csv
hashagile-round2.py
temp.py
OUTLINE
TIMELINE
OUTPUT DEBUG CONSOLE TERMINAL PORTS
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.7.9 64-bit Go Live Prettier
Search
ENG IN
11:39 PM 03-10-2024