User Keywords Full Text Search

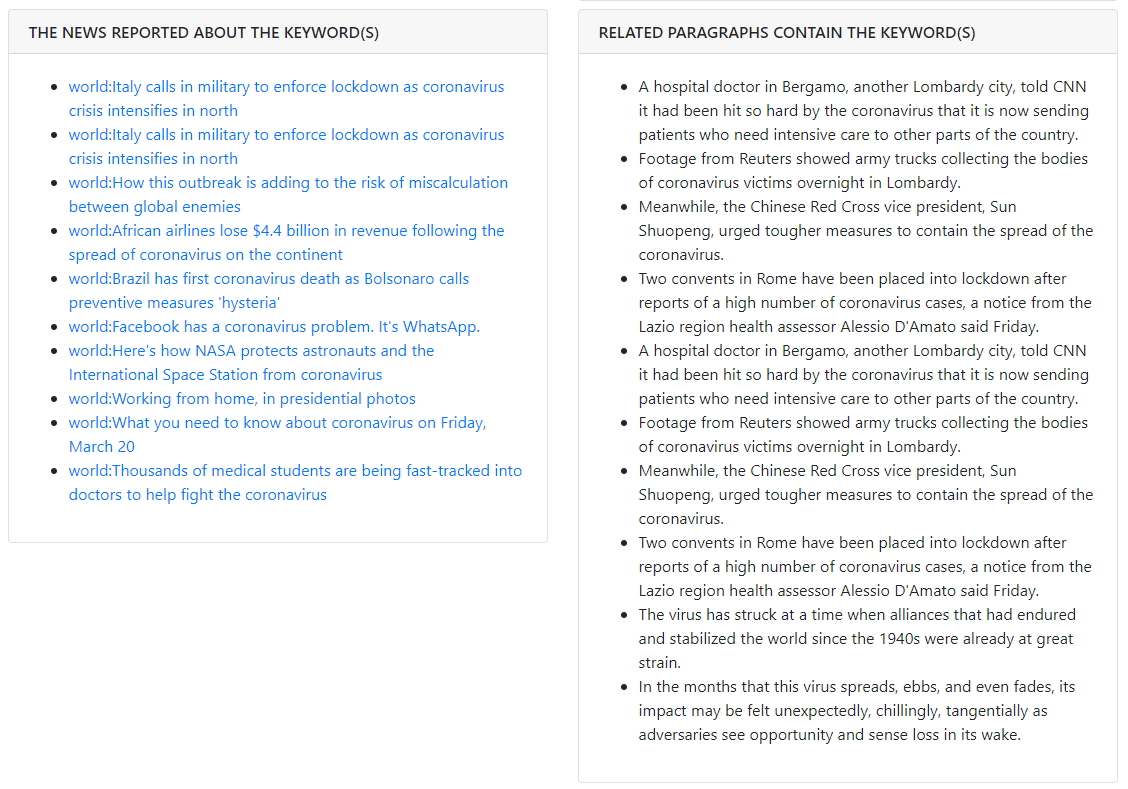
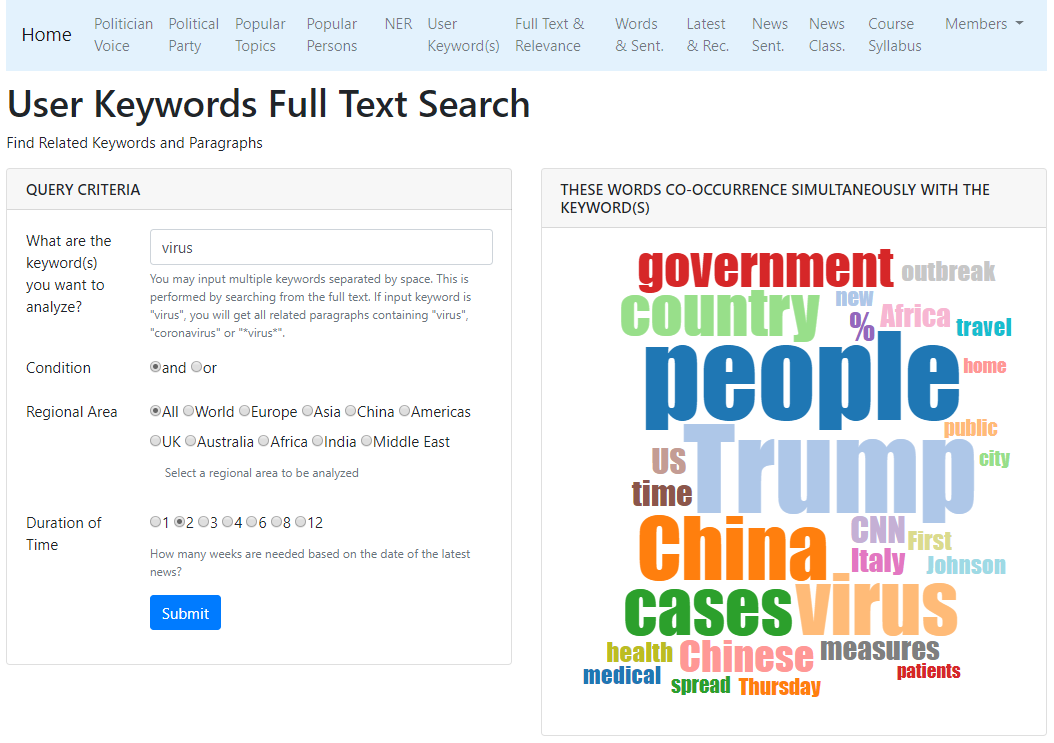
Find Related Keywords and Paragraphs

Take a look at the demonstration website.

In this app, users input their keywords and find related keywords and paragraphs based on their input keywords.

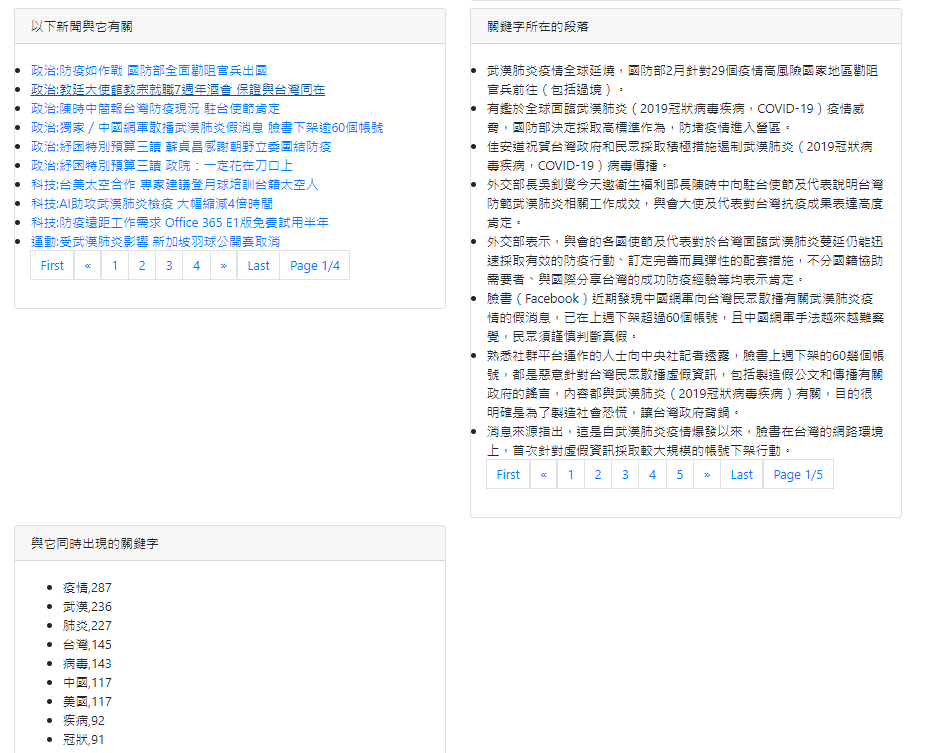
Let’s get started!

<http://163.18.22.32:8002/userkeyword/userkey_associate>



<http://163.18.23.21:8000/userkeyword/userkey_associate>





Step 0: Create a project by duplicating previous project

We will continue from our previous project.

I suggest you rename your project folder with a meaningful name such as:

“website\_news\_analysis\_user\_keyword\_association”

Step1: Open the project with your favorite Editor

Open our project using VS Code.

Step 2: Create “app\_user\_keyword\_association”

Step 2: Create a new APP named “app\_user\_keyword\_association” in your website root.

python manage.py startapp app\_user\_keyword\_association

Step3: Copy data file to the dataset folder

We will use the preprocessed data file, so copy it into the dataset folder.

We have done this in our previous app.

Step 4: settings.py

|  |  |
| --- | --- |
| |  | | --- | | ALLOWED\_HOSTS = ['localhost', '127.0.0.1']  INSTALLED\_APPS = [  …..  ' app\_top\_keyword',  ' app\_top\_person',  'app\_top\_ner\_analysis',  'app\_user\_keyword',  'app\_user\_keyword\_association',  ]  TEMPLATES = [  {  'BACKEND': 'django.template.backends.django.DjangoTemplates',  'DIRS': [os.path.join(BASE\_DIR, 'templates')],  'APP\_DIRS': True,  'OPTIONS': …..  ……  ……  },  ]  # Static files (CSS, JavaScript, Images)  # https://docs.djangoproject.com/en/2.2/howto/static-files/  STATIC\_URL = '/static/'  # we should add static folder into path for css and images  STATICFILES\_DIRS = (  os.path.join(BASE\_DIR, 'static'),  ) | |

Step 4.5: static css and JS

1. Create a folder “static” in the project folder.
2. Put your own JS “pagination\_bs4.js” into the static folder.

Step 5: website\_configs/urls.py

website\_configs/urls.py

|  |  |
| --- | --- |
| |  | | --- | | from django.contrib import admin  from django.urls import path  from django.urls import include  from django.views.generic import TemplateView  urlpatterns = [  # top keywords  path('topword/', include('app\_top\_keyword.urls')),  # top persons  path('topperson/', include('app\_top\_person.urls')),  # top name entity keyword  path('topner/', include('app\_top\_ner.urls')),  # user keyword analysis  path('userkeyword/', include('app\_user\_keyword.urls')),  # full text search and associated keyword display  path('userkeyword\_assoc/', include('app\_user\_keyword\_association.urls')),  ] | |

Step 6: app\_user\_keyword\_association/urls.py

In folder app\_user\_keyword\_association, create a Python file named “urls.py”.

app\_user\_keyword\_association/urls.py

|  |  |
| --- | --- |
| |  | | --- | | from django.urls import path  from app\_user\_keyword\_association import views  app\_name="app\_user\_keyword\_association"  urlpatterns = [      path('', views.home, name='home'),      path('api\_get\_userkey\_associate/', views.api\_get\_userkey\_associate),  ] | |

Step 7: views.py

(1) We need the following data to display on website:

response = {

'newslinks': newslinks,

'related\_words': related\_words,

'same\_paragraph': same\_paragraph\_top10,

'clouddata':clouddata,

'num\_articles': len(df\_query),

}

return JsonResponse(response)

view.py

|  |
| --- |
| from django.shortcuts import render  from django.views.decorators.csrf import csrf\_exempt  from django.http import JsonResponse  from datetime import datetime, timedelta  import pandas as pd  import math  import re  from collections import Counter  # (1) we can load data using read\_csv() 自己app的csv檔案  # global variable  # df = pd.read\_csv('dataset/cna\_news\_200\_preprocessed.csv', sep='|')  # (2) we can load data using reload\_df\_data() function 隔壁app的csv檔案  # global variable  def load\_df\_data\_v1():  # global variable  global df  df = pd.read\_csv('app\_user\_keyword/dataset/cna\_news\_200\_preprocessed.csv', sep='|')  # (3) df can be import from app\_user\_keyword 隔壁app的變數  # To save memory, we just import df from the other app as follows.  # from app\_user\_keyword.views import df  # (4) df can be import from app\_user\_keyword 隔壁app的變數  import app\_user\_keyword.views as userkeyword\_views  def load\_df\_data():  # import and use df from app\_user\_keyword  global df # global variable  df = userkeyword\_views.df  load\_df\_data()  # For the key association analysis  def home(request):  return render(request, 'app\_user\_keyword\_association/home.html')  # df\_query should be global  @csrf\_exempt  def api\_get\_userkey\_associate(request):  userkey = request.POST.get('userkey')  cate = request.POST['cate'] # This is an alternative way to get POST data.  cond = request.POST.get('cond')  weeks = int(request.POST.get('weeks'))  key = userkey.split()  #global df\_query # global variable It's not necessary.  df\_query = filter\_dataFrame\_fullText(key, cond, cate,weeks)  print(key)  print(len(df\_query))  if len(df\_query) != 0: # df\_query is not empty  newslinks = get\_title\_link\_topk(df\_query, k=15)  related\_words, clouddata = get\_related\_word\_clouddata(df\_query)  same\_paragraph = get\_same\_para(df\_query, key, cond, k=10) # multiple keywords  else:  newslinks = []  related\_words = []  same\_paragraph = []  clouddata = []  response = {  'newslinks': newslinks,  'related\_words': related\_words,  'same\_paragraph': same\_paragraph,  'clouddata':clouddata,  'num\_articles': len(df\_query),  }  return JsonResponse(response)  # Searching keywords from "content" column  # Here this function uses df.content column, while filter\_dataFrame() uses df.tokens\_v2  def filter\_dataFrame\_fullText(user\_keywords, cond, cate, weeks):  # end date: the date of the latest record of news  end\_date = df.date.max()  # start date  start\_date = (datetime.strptime(end\_date, '%Y-%m-%d').date() -  timedelta(weeks=weeks)).strftime('%Y-%m-%d')  # (1) proceed filtering: a duration of a period of time  # 期間條件  period\_condition = (df.date >= start\_date) & (df.date <= end\_date)  # (2) proceed filtering: news category  # 新聞類別條件  if (cate == "全部"):  condition = period\_condition # "全部"類別不必過濾新聞種類  else:  # category新聞類別條件  condition = period\_condition & (df.category == cate)  # (3) proceed filtering: news category  # and or 條件  if (cond == 'and'):  # query keywords condition使用者輸入關鍵字條件and  condition = condition & df.content.apply(lambda text: all(  (qk in text) for qk in user\_keywords)) # 寫法:all()  elif (cond == 'or'):  # query keywords condition使用者輸入關鍵字條件  condition = condition & df.content.apply(lambda text: any(  (qk in text) for qk in user\_keywords)) # 寫法:any()  # condiction is a list of True or False boolean value  df\_query = df[condition]  return df\_query  # get titles and links from k pieces of news  def get\_title\_link\_topk(df\_query, k=25):  items = []  for i in range( len(df\_query[0:k]) ): # show only 10 news  category = df\_query.iloc[i]['category']  title = df\_query.iloc[i]['title']  link = df\_query.iloc[i]['link']  photo\_link = df\_query.iloc[i]['photo\_link']  # if photo\_link value is NaN, replace it with empty string  if pd.isna(photo\_link):  photo\_link=''    item\_info = {  'category': category,  'title': title,  'link': link,  'photo\_link': photo\_link  }  items.append(item\_info)  return items  # Get related keywords by counting the top keywords of each news.  # Notice: do not name function as "get\_related\_keys",  # because this name is used in Django  def get\_related\_word\_clouddata(df\_query):  # wf\_pairs = get\_related\_words(df\_query)  # prepare wf pairs  counter=Counter()  for idx in range(len(df\_query)):  pair\_dict = dict(eval(df\_query.iloc[idx].top\_key\_freq))  counter += Counter(pair\_dict)  wf\_pairs = counter.most\_common(20) #return list format  # cloud chart data  # the minimum and maximum frequency of top words  min\_ = wf\_pairs[-1][1] # the last line is smaller  max\_ = wf\_pairs[0][1]  # text size based on the value of word frequency for drawing cloud chart  textSizeMin = 20  textSizeMax = 120  # Scaling frequency value into an interval of from 20 to 120.  clouddata = [{'text': w, 'size': int(textSizeMin + (f - min\_) / (max\_ - min\_) \* (textSizeMax - textSizeMin))}  for w, f in wf\_pairs]  return wf\_pairs, clouddata  # Step1: split paragraphs in text 先將文章切成一個段落一個段落  def cut\_paragraph(text):  paragraphs = text.split('。') # 遇到句號就切開  #paragraphs = re.split('。', text) # 遇到句號就切開  #paragraphs = re.split('[。！!？?]', text) # 遇到句號(也納入問號、驚嘆號、分號等)就切開  paragraphs = list(filter(None, paragraphs))  return paragraphs  # Step2: Select all paragraphs where multiple keywords occur.  def get\_same\_para(df\_query, user\_keywords, cond, k=30):  same\_para = []  for text in df\_query.content:  #print(text)  paragraphs = cut\_paragraph(text)  for para in paragraphs:  para += "。"  if cond == 'and':  if all([re.search(kw, para) for kw in user\_keywords]):  same\_para.append(para)  elif cond == 'or':  if any([re.search(kw, para) for kw in user\_keywords]):  same\_para.append(para)  return same\_para[0:k]    print("app\_user\_keyword\_association was loaded!") |

Step 8: home.html

In the app folder, create a HTML file named “home.html”

app\_user\_keyword\_association/templates/app\_user\_keyword\_association/home.html

home.html

|  |
| --- |
| {% extends 'base.html' %} {% block title %}  全文檢索與你關心的關鍵詞關聯分析  {% endblock %} {% block content %}  <div class="col-lg-12">  <h1>全文檢索與你關心的關鍵詞關聯分析</h1>  <p>  對你想要了解的議題進行全文檢索，找出有哪些詞與你的關鍵詞一起出現?  </p>  </div>  <div class="col-lg-6 mb-2">  <!-- 輸入條件區塊開始 -->  <div class="card">  <div class="card-header">  <h3 class="h6 text-uppercase mb-0">輸入條件</h3>  </div>  <div class="card-body">  <div class="row mb-3">  <label class="col-md-3 col-form-label">關心哪個關鍵詞?</label>  <div class="col-md-9">  <input  id="input\_keyword"  name="userkey"  value="烏克蘭 俄羅斯"  class="form-control"  />  <div class="form-text">  全文搜尋，可輸入多個關鍵詞或片段詞句，以空白隔開。  </div>  </div>  </div>  <div class="row mb-3">  <label class="col-sm-3 col-form-label">條件</label>  <div class="col-md-9">  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="and"  name="condradio"  id="condradio1"  checked  />  <label class="form-check-label" for="condradio1">and</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="or"  name="condradio"  id="condradio2"  />  <label class="form-check-label" for="condradio2">or</label>  </div>  </div>  </div>  <div class="row mb-3">  <label class="col-sm-3 col-form-label">新聞類別</label>  <div class="col-md-9">  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="全部"  name="cateradio"  id="cateradio1"  checked  />  <label class="form-check-label" for="cateradio1">全部</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="政治"  name="cateradio"  id="cateradio2"  />  <label class="form-check-label" for="cateradio2">政治</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="科技"  name="cateradio"  id="cateradio3"  />  <label class="form-check-label" for="cateradio3">科技</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="運動"  name="cateradio"  id="cateradio4"  />  <label class="form-check-label" for="cateradio4">運動</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="證卷"  name="cateradio"  id="cateradio5"  />  <label class="form-check-label" for="cateradio5">證卷</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="產經"  name="cateradio"  id="cateradio6"  />  <label class="form-check-label" for="cateradio6">產經</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="娛樂"  name="cateradio"  id="cateradio7"  />  <label class="form-check-label" for="cateradio7">娛樂</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="生活"  name="cateradio"  id="cateradio8"  />  <label class="form-check-label" for="cateradio8">生活</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="國際"  name="cateradio"  id="cateradio9"  />  <label class="form-check-label" for="cateradio9">國際</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="社會"  name="cateradio"  id="cateradio10"  />  <label class="form-check-label" for="cateradio10">社會</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="文化"  name="cateradio"  id="cateradio11"  />  <label class="form-check-label" for="cateradio11">文化</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="兩岸"  name="cateradio"  id="cateradio12"  />  <label class="form-check-label" for="cateradio12">兩岸</label>  </div>  </div>  </div>  <div class="row mb-3">  <label class="col-md-3 col-form-label">最近多少周?</label>  <div class="col-md-9">  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="1"  name="wkradio"  id="wkradio1"  />  <label class="form-check-label" for="wkradio1">1</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="2"  name="wkradio"  id="wkradio2"  checked  />  <label class="form-check-label" for="wkradio2">2</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="3"  name="wkradio"  id="wkradio3"  />  <label class="form-check-label" for="wkradio3">3</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="4"  name="wkradio"  id="wkradio4"  />  <label class="form-check-label" for="wkradio4">4</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="6"  name="wkradio"  id="wkradio6"  />  <label class="form-check-label" for="wkradio6">6</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="8"  name="wkradio"  id="wkradio8"  />  <label class="form-check-label" for="wkradio8">8</label>  </div>  <div class="form-check form-check-inline">  <input  class="form-check-input"  type="radio"  value="12"  name="wkradio"  id="wkradio12"  />  <label class="form-check-label" for="wkradio12">12</label>  </div>  <div class="form-text">  以最新資料時間為準，往前推多少周?  </div>  </div>  </div>  <div class="row mb-3">  <div class="col-md-9 ms-auto">  <button type="button" id="btn\_ok" class="btn btn-primary">  查詢  </button>  </div>  </div>  </div>  </div>  </div>  <!-- 輸入區塊結束-->  <!-- 繪圖區塊-->  <div class="col-lg-6 mb-2">  <div class="card">  <div class="card-header">  <h3 class="h6 text-uppercase mb-0">這些詞與它同時出現喔!</h3>  </div>  <div class="card-body">  <div id="cloud"></div>  </div>  </div>  </div>  <!-- 區塊結束-->  <!-- 新聞連結區塊-->  <div class="col-lg-6 mb-2">  <div class="card">  <div class="card-header">  <h3 class="h6 text-uppercase mb-0">以下新聞與它有關(取數篇展示)</h3>  </div>  <div class="card-body">  <h2 id="num\_articles"></h2>  <ul class="list-group" id="newslinks"></ul>  </div>  </div>  </div>  <!-- 區塊結束-->  <!-- 同時出現的關鍵字區塊-->  <div class="col-lg-6 mb-2">  <div class="card">  <div class="card-header">  <h3 class="h6 text-uppercase mb-0">關鍵字所在的段落(取數段展示)</h3>  </div>  <div class="card-body">  <!-- 這個標籤顯示包含該關鍵詞的段落 -->  <ul class="list-group" id="same\_paragraph"></ul>  </div>  </div>  </div>  <!-- 區塊結束-->  <!-- 同時出現的關鍵字區塊-->  <div class="col-lg-6 mb-2">  <div class="card">  <div class="card-header">  <h3 class="h6 text-uppercase mb-0">與它同時出現的關鍵字</h3>  </div>  <div class="card-body">  <ul id="related\_words"></ul>  </div>  </div>  </div>  <!-- 區塊結束-->  {% endblock %} {% block extra\_js %}  <!-- chartjs-->  <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.3/Chart.min.js"></script>  <!-- cloud chart  Here is the cloud chart script.  reference:http://bl.ocks.org/joews/9697914  reference:https://jsfiddle.net/x827g61m/  reference:https://www.d3-graph-gallery.com/wordcloud  src="https://rawgit.com/jasondavies/d3-cloud/master/build/d3.layout.cloud.js">  -->  <script src="https://cdnjs.cloudflare.com/ajax/libs/d3/3.4.11/d3.min.js"></script>  <script src="https://cdn.rawgit.com/jasondavies/d3-cloud/v1.2.1/build/d3.layout.cloud.js"></script>  <!-- Here are your codes -->  <script>  // Show the page with default setting when page is initialized.  call\_ajax();  // btn submit  $("#btn\_ok").on("click", function () {  call\_ajax();  }); //event function  // category radio button  $("input[name='cateradio']").on("change", function () {  call\_ajax();  }); //event function  // weeks radio button  $("input[name='wkradio']").on("change", function () {  call\_ajax();  }); //event function  // condition radio button  $("input[name='condradio']").on("change", function () {  call\_ajax();  }); //event function  function call\_ajax() {  const userkey = $("#input\_keyword").val();  const weeks = $("input[name='wkradio']:checked").val();  const cate = $("input[name='cateradio']:checked").val();  const cond = $("input[name='condradio']:checked").val();  if (userkey.length < 2) {  alert("輸入關鍵字不可空白或小於兩個中文字!");  return 0;  }  $.ajax({  type: "POST",  url: "api\_get\_userkey\_associate/",  data: {  userkey: userkey,  cate: cate,  weeks: weeks,  cond: cond,  }, // pass to server  success: function (received) {    // display number of articles or stories  const num\_articles = received["num\_articles"];  $("#num\_articles").empty();  $("#num\_articles").append(`<h2 style="color:red">總篇數:${num\_articles}</h2>`);    // show news title and link  const newslinks = received["newslinks"];  $("#newslinks").empty();  if (newslinks.length == 0) {  alert("No result returned!");  }  // show news title and link  for (let i = 0; i < newslinks.length; i++) {  const items =`  <li class="list-group-item py-2 border-bottom">  <div class="d-flex align-items-center">  <span class="badge bg-secondary me-2 px-2 py-1">${newslinks[i].category}</span>  <a href="${newslinks[i].link}" class="text-decoration-none" target="\_blank">${newslinks[i].title}</a>  </div>  </li>`;  $("#newslinks").append(items);  }  // show related words  const related\_words = received["related\_words"];  $("#related\_words").empty();  for (let i = 0; i < related\_words.length; i++) {  $("#related\_words").append( `<li>${related\_words[i]}</li>` );  }  // show paragraphs containing the user keywords  const same\_paragraph = received["same\_paragraph"];  $("#same\_paragraph").empty();  for (let i = 0; i < same\_paragraph.length; i++) {  $("#same\_paragraph").append(`<li class="list-group-item py-2 border-bottom">${same\_paragraph[i]}</li>`);  }  // draw word cloud for related words  topWordToDraw = received.clouddata;  $("#cloud").empty();  drawCloud(topWordToDraw, "#cloud");  }, //success function  error: function (msg, status) {  console.log(msg);  console.log(status);  }, //print status and msg when ajax goes wrong  }); //ajax  } //function call\_ajax()  function drawCloud(topWordToDraw, element\_id) {  // You should set a proper box size to show cloud chart  const width = 500;  const height = 500;  // First define your cloud data, using `text` and `size` properties:  // Next you need to use the layout script to calculate the placement, rotation and size of each word:  // Constructs a new cloud layout instance.  // Wordcloud features that are different from one word to the other  d3.layout  .cloud()  .size([width, height])  .words(topWordToDraw) //data for cloud chart  .rotate(function () {  //return ~~(Math.random() \* 2) \* 90; //~~1.5 => 1 (same as Math.floor(1.5))  return 0; // don't rotate  })  .font("Impact")  .fontSize(function (d) {  return d.size;  })  .on("end", draw) //call function draw()  .start();  // Finally implement `draw`, which performs the D3 drawing  // This function takes the output of 'layout' above and draw the words  // Wordcloud features that are THE SAME from one word to the other can be here  function draw(words) {  const fill = d3.scale.category20();  // append the svg object to the body of the page  d3.select(element\_id)  .append("svg") // element\_id such as "#cloud"  .attr("width", width)  .attr("height", height)  .append("g")  .attr(  "transform",  "translate(" + ~~(width / 2) + "," + ~~(height / 2) + ")"  )  .selectAll("text")  .data(words)  .enter()  .append("text")  .style("font-size", function (d) {  return d.size + "px";  })  .style("-webkit-touch-callout", "none")  .style("-webkit-user-select", "none")  .style("-khtml-user-select", "none")  .style("-moz-user-select", "none")  .style("-ms-user-select", "none")  .style("user-select", "none")  .style("cursor", "default")  .style("font-family", "Impact")  .style("fill", function (d, i) {  return fill(i);  })  .attr("text-anchor", "middle")  .attr("transform", function (d) {  return "translate(" + [d.x, d.y] + ")rotate(" + d.rotate + ")";  })  .text(function (d) {  return d.text;  });  } //draw  } //drawCloud()  </script>  {% endblock %} |

Step 9: base.html

<a class="nav-link" href="{% url 'app\_user\_keyword\_association:home' %}">全文檢索與關聯新聞分析</a>

Step 10: Run server

python manage.py runserver 8000

What does your web page look like?