mainly 2 way to extract data from website,

(1) API

(2) Access html page info, is called webscraping (or) web harvesting

understanding,

* import requests library
* Then specify url of the web want to scrap
* Send HTTP request to the specified URL and save response.
* Now print r.Content to get raw html Content. It is of 'string' type.

beautifulsoup web scrape.

```
import requests
from bs4 import BeautifulSoup
web_url = "link of the site"
response = requests.get(url = web.url)
data = response.text
soup = BeautifulSoup(data, 'html.parser')
print(soup)  (or) print(soup.prettify)
```

instead of using html.parser also can use htmlslip

need to change a request as a text (or) content.

(e.g) response.text (or) response.content

It will print the pages html content.

find() method

```
find_syntax = soup.find("widget name", {"id":
              "id name of widget in which you want
               to edit"}).get_text()
```

find_all() method.

find all tags which specified tag name (or

Syntax

```
for word in soup.find_all('id'):
    find_all_syntax = word.get_text()
print(find_all_syntax)
```

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
ser_obj = Service("path to chrome driver")
driver = webdriver.Chrome(service = ser_obj)
driver.get(url = "url of the page link")
```

} importing basic

1) Find_element , 2) Find_elements

```
from selenium.webdriver.common.by import by
```

→ import this before findinging elements

```
find_element(By.ID, "id")
    ''      (By.NAME, "name")
    ''      (By.XPATH, "Xpath")
    ''      (By.LINK_TEXT, "link text")
    ''  (By.PARTIAL_LINK_TEXT, "partial link text")
    ''  (By.TAG_NAME, "tagname")
    ''  (By.CLASS_NAME, "class name")
    ''  (By.CSS_SELECTOR, "CSS selector")
```

'By' class is used to specify which attribut
is used to locate elements.

54) Flask (one of the be framework).

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def home():
    return 'helloworld'
if __name__ == '__main__':
    app.run(debug = True)
```

basic syntax.

@app.route('/') → decorative fun.

return hello world in browser.

app.run(debug = True) → run the flask in debug mode.

installation in terminal,

pip install Flask → $env:FLASK_APP = "hello.py" → flask run.

(filename)

using jinja template in Flask.

```
from flask import Flask, render_template.
@app.route('/')
def home():
    return render_template('index.html')
```

instead of printing the text link the html file using Jinja.

Flask (one of the python frameworks)

```
@app.route('/blog/<num>')
def get_blog(num):
    result = response.json()
    return render_template(blog.html, blogs = result)
```

passing variable (< >) if want to change the type <int:num> use like this.

pass the fun varai.. to the page. cant pass directly, use the variable name and pass the value.

use the (blogs) name into the html file for showing the result.

(e.g) html used flask.

```
<body>
    {%. for blog in blogs:%.}
        {%. if blog['id'] == 2:%.}
        <h1> {{ blog['title'] }} </h1>
        <h2> {{ blog['subtitle'] }} </h2>
        {%.endif%.}
    {%. endfor%.}
</body>
```

{%. %.} → open and close like this in every condition.

if we use 'for' loop condition, in end use {%. endfor %.}

'if condition' in end use {%. endif %}

the variable name use {{ . }} this to wrapp the variable.

all above a route is a differend pages. we dont have a link in index page to redirecting to /blog route.

```
<a href="{{url_for('get_blog', num=2)}}"> click </a>
```

in index.html spefy like this to get a /blog route. also put the variable

```
<div class = "row">
    <div class = "col">.col </div>
    <div class = "col">.col </div>
    <div class = "col">.col </div>
</div>
```

Three equal column.

↑ specify style background color.

.col	.col	.col

```
<div class = "row">
    <div class = "col-sm-3">.col-sm-3 </div>
                ''
                ''
                ''
</div>
```

responsive column.

```
<div class = "row">
    <div class = "col-sm-4">.col-sm-4 </div>
    <div class = "col-sm-8">.col-sm-8 </div>
</div>
```

two unequal responsive columns.

.col-sm-4	.col-sm-8

go and do practice only seeing the code not gonna be work at all.
So write it your own creativity

Containers.

Container → make content full width not take full width.

Container-fluid → take 100% of the width.

```
<div style="background-color: red;" class="container">
    <h1> hello </h1>
</div>
```

hello.

color will be red.

instead of the if we use Container-fluid the result is →

hello

(Take the full width of the page.

Carousel : → slideshow for cycling through elements.