Functionality 1 = Scrape the website: Scraping a website in order to later be able to thematisis it into different categories and compare them to eachother. I.e. which website has the most articles about sports.

S1: To be able to do anything with a website we first need its contents.
Psuedo Code:

```
    PyScrape.Scrape(webvariable1)
```

S2: To be able to compare two websites we need to know what the website contains. Images, articles (very long strings/tags) etc.
Psuedo Code:

```
#Scrapes the website for all its contents
Website1 = PyScraper.Scrape(webVariable1)
    #Organises the website into logical pieces based on tags and contents. I.e. an
    #article will be x number of charaters long, an image will be in an img tag etc.
    WebsiteOrganised = PyScraper.OrganizeTags(website1)
```

S3: Gather information from diffrent websites and put the contents into a nested dictionary in order to store all related values.
Psuedo code:

```
    Def organizeArticles(listOfArticles):
        For articleTitles in articles:
    For theme in articles:
    for url in articles:
                    DictOfArticles = {articleTitle : {theme : url}}
```

Functionality 2 = CountPhrases: In order to make the scraper useful right from the box, the webscraper will come prepackaged with some logic to categorize articles based on their contents. I.e. if "Ukraine war" is repeated several times it most likely is an article about the war in Ukraine, which is foreign affairs.

S1: To see the diffrence in coverage between two websites
Psuedo code:

```
        Print(pyScraper.themeify(webVariable1)
```

S2: Find the website that has the most referances to the war in Ukraine.
Psuedo code:

```
phraseCount1 = pyScraper.countPhrase(webVariable1, "krigen i
    Ukraina")
        phraseCount2 = pyScraper.countPhrase(webVariable2, "krigen i Ukraina")

        If  phraseCount1 > phraseCount2:
            Print(f”{webVariable1.name} writes more articles about the war in Ukraine then {webVariable2.name}”)
        Else if phraseCount1 < phraseCount2:
            Print(f”{webVariable2.name} writes more articles about the war in Ukraine then {webVariable1.name}”)
        Else:
        Print(“they have the same amount of coverage”)
```

Functionality 3 = Categorize articles (themeify):

S1: What website has the most sports articles?

Psuedo code:

```
For each article in articles:
themeSpreadW1 = pyScraper.themeify(webVariable1, theme=sports)
themeSpreadW2 = pyScraper.themeify(webVariable2, theme=sports)
    Print(themeSpreadW1)
    Print(themeSpreadW2)
```

S2: What website has most foreign affairs articles?

Psuedo code:

```
    For each article in articles:
themeSpreadW1 = pyScraper.themeify(webVariable1, theme=”election”)
themeSpreadW2 = pyScraper.themeify(webVariable2, theme=”election”)
    Print(themeSpreadW1)
    Print(themeSpreadW2)
```

Functionality 4 = Compare websites:

S1: I want to compare two websites to see how their coverage varries

Psuedo code:

```
    PyScraper.compareArticles(webVariable1, webVariable2)
```

S2: I want to compare two websites in order to see which website has the most coverage from a given part of the world

Psuedo code:

```
Pyscraper.compareArticles(webVariable1, webVariable2, search=”geography”)
```

S3: i want to compare the two websites to find what political party they mention the most

Pusedo code:

```
Pyscraper.compareArticles(webVariable1, webVariable2, search=”political parties”)
```

API Specifications

Our api will solve problems using web scraper's knowledge. This means that we streamline what web scrapers do by adapting it to what the user wants to solve.

Our first functional area of use is to compare websites. It gives us many possible scenarios we can compare between websites.

1. comparison of how many html headers are on each website.

Client code 1

make a variable that holds the max headers amount int. (empty one as long)

also another variable that contains the url to the one with the most headers. (empty one so long).

Next, we create a for loop that goes through each web page and counts the headers.

Also if max headers is less headerscounts then print that url.

client code 2

defines and functions to tell the number of headers on a web page.

Then create an if test that compares who has the most.

Also elif if the other is more.

1. scenario 2 which one has better seo between the webpages

    Client code 1

```
function compareHeaders(url1, url2):
    headers1 = fetchHeaders(url1) # function to fetch headers of webpage at url1
    headers2 = fetchHeaders(url2) # function to fetch headers of webpage at url2

    if len(headers1) > len(headers2):
        print(url1, "has more headers than", url2)
    elif len(headers1) < len(headers2):
        print(url2, "has more headers than", url1)
    else:
        print("Both webpages have the same number of headers")
```

client code 2

```
Function compareMetadata(page1, page2):
    metaCount1 = countMetadata(page1)
    metaCount2 = countMetadata(page2)

    If metaCount1 > metaCount2:
        Return "Page 1 has more metadata"
    ElseIf metaCount2 > metaCount1:
        Return "Page 2 has more metadata"
    Else:
        Return "Both pages have the same amount of metadata"
      End Function
```