NEWS AGGREGATOR

USING DJANGO

A PROJECT REPORT

Submitted by

GANGA DEVI N (715517104021)

JAYASHNI K (715517104027)

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PSG INSTITUTE OF TECHNOLOGY AND APPLIED RESEARCH, COIMBATORE 641 062

ANNA UNIVERSITY: CHENNAI - 600 025

MAY 2021

ANNA UNIVERSITY: CHENNAI - 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"NEWS AGGREGATOR USING DJANGO"** is the bonafide work of **"JAYASHNI K (715517104027), GANGA DEVI N (715517104021)"** who carried out the project work under my supervision.

----------	**----------**
SIGNATURE	**SIGNATURE**
Dr. R. MANIMEGALAI	Dr. A.C. SUMATHI
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Computer Science and Engineering	Assistant Professor (Sl. Gr.)
PSG Institute of Technology and	Computer Science and Engineering,
Applied Research,	PSG Institute of Technology and
	Applied Research,
Coimbatore – 641 062	Coimbatore – 641 062

Submitted for the project viva-voce Examination held on_____

----------	**----------**
INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

I am very grateful to **Dr. G. Chandramohan, Principal** for providing me with an environment to complete our project successfully.

I also take the privilege of expressing my gratitude to **Dr. P.V Mohanram, Secretary** in extending his support to carry out my study.

I am greatly indebted to **Dr. R. Manimegalai, Head of the Department, Computer Science and Engineering** for her guidance which was instrumental in the completion of my project.

I express my sincere thanks to **Dr.A.C.Sumathi, ProjectGuide** for her constant encouragement and support throughout the course.

Also, I earnestly thank my Project Coordinator **Mr. S. Thivaharan, AP(SG)** for his consistent support throughout the course.

Finally, I take this opportunity to extend my deepest appreciation to my **family and friends**, who supported me during the crucial times of my project.

GANGA DEVI N

JAYASHNI K

PLAGIARISM REPORT

VeriGuide Originality Report

Background Information [what is this?]

Batch file name: NewsAggregatorReportt.docx

Report generated on: 28/05/2021, 05:59:15 AM

Checking Parameters [what is this?]

Matching scope(s): Within submission, Internet

Leniency: Detailed matching with threshold 70%

Minimum sentence length: Sentences with more than or equal to 3 meaningful words were checked

Similarity Statistics

Similarity Statistics [what is this?]

Total number of documents: 1

Number of documents which can be processed: 1

Number of documents which cannot be processed: 0

Show 10 entries Search:

Entry	Document	Status	Similarity	Action
1	NewsAggregatorReportt.docx	processed	24/301=7.90%	View details

Showing 1 to 1 of 1 entries First Previous 1 Next Last

ABSTRACT

News Aggregator is a web application that combines news articles from many websites and presents the collected information in a location. There are numerous publications and news sites online. They publish their content on many platforms on regular basis. Instead of visiting 10-20 news sites to gain current affairs, the news aggregator collects the articles for the reader. In a news aggregator, the reader can select the websites they want to follow. And it is just one click away to get information from various websites.

Web crawlers and web applications together forms a news aggregator. These technologies have their implementation purely in Python. The news aggregator can be implemented in 3 steps:

1. The website is scraped for the articles.

2. Images, links, and title of the article are stored in the database

3. Stored objects in the database are served to the reader. The reader gets the information in the form of template.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

MVT	Model View Template
HTML	Hypertext Markup Language
JS	JavaScript
OS	Operating System
API	Application Programming Interface
URL	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol
CSS	Cascading Style Sheets
UI	User Interface

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The automated collection of data from the internet is old as the internet itself. Web scraping is the practice of extracting data other than the interaction made by the program with an API. This is accomplished by writing an automated program that queries a web server, requests data, and then parses that data to extract required information. In practice, web scraping encompasses a large kind of programming techniques and technologies such as data analysis and information security.

1.2 OBJECTIVE

The main objective of this project is to develop a website to scrape the news articles from major news agencies and organize them in one site. The project is aimed at providing all news at one place, to develop a dynamic news portal which updates its daily content, to allow redirection for brief news.

The web application allows user to get news on multiple languages depending upon on their preference like English, Tamil. This website allows the user to get a time-consuming insight in just a second.

1.3 PROBLEM STATEMENT

News Aggregator is a web application that combines news article from various websites and displays the data in a location. There are many publications and news websites online. They publish their content on various platforms. Instead of visiting 10-20 news sites to get information, the news aggregator

fetches the articles for the user. In a news aggregator, user can select the websites they want to follow. Web crawlers and web applications are combined to form news aggregator. These technologies have their implementation in Python.

1.4 PROJECT OVERVIEW

News aggregator, online platform or an application that collects news stories and other information and publishes and organizes the information in a particular manner. A user requests a resource from Django. Django acts as a controller and checks all the resources in the URL. If the URL maps, a view is called which is a python code for scraping the news content that interacts with the model and template which is created using HTML. Django then responds back to the user with a template. Beautiful soup library in python is used for scraping the news content. It is done by retrieving HTML data from a domain name and parsing the data for target information. Then the target information is stored in SQLite3 database. These information are displayed in the website.

1.5 SCOPE AND MOTIVATION

There are many news websites, which cover up news on many broad topics, out of which only a few of them are of reader's interest. A news aggregator can be an application to save a lot of time and with some alteration and filtration programmer can fine tune the application to show only news of user's interest. The News Aggregator is a system that can automatically tell where the content (i.e., object) on the Web comes from and who is responsible for the content after it has aggregated the content.

1.6 INTRODUCTION TO DJANGO

DJANGO is a python web framework that encourages rapid development and clean design and secures websites. Django simplifies the development process of complex database used for web application like a new oriented website.

Django has a built-in support for Ajax, RSS, caching and various other frameworks. Django provides a connection between a data model and the database engine and supports a many set of database systems which includes MySQL, Oracle, Postgres. It also supports multilingual websites through its built-in internationalization system.

Django is a well-designed framework that includes three major parts such as model, view and template.

Model – It is a single definitive data source which contains an essential field and behavior of the data. One model is only one table in the database. Django provides a set of automatically generated database APIs for the convenience of users.

View – It is file containing python function which takes web requests and returns web responses. A response can be HTML content or an XML documents or a “404 error” and so on.

Template – Django’s template is a simple text file which can generate HTML and XML. Tags play an important role and controls the logic of a template.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEMS

The number of papers dealing with news aggregator using Django in literature is growing exponentially. Several researchers have played a significant role in the development of aggregator.

Paliouras et al. [9] developed a personalized news aggregator web application. Their main objective was to develop content-based filtering on important keyword. As a result, the user can get personalized news based on their interest. The main advantage of their project was the combination of RSS feed and HTML. The future work of the project was to improve the functionality of the system, enhance presentation, user friendly interface.

Alaa Mohamed et al. [1] developed a news aggregator and efficient summarization system that collects news from various sources all in one place. Their main objective was to aggregate news from cloud service. As a result, they were able to aggregate relevant article of a certain input keyword and summarized it efficiently. The merit of this project was text enhancement and text summarization. The future work of the project was to aggregate personalized news based on user's interest.

A PERSONALIZED NEWS AGGREGATOR ON THE WEB(PNS) Georgios Paliouras, Alexandros Mouzakidis, Vassileios Moustakas, Christos Skourlas	
Problem solved	A personalized news aggregator on the web is created
Proposed solution	Content based filtering is done based on important keyword.
results	Personalized news is viewed.
merit	Combination of RSS feed and HTML
demerit	Only English and Greek is supported.
Future work	Improve functionality of the system,enhance presentation,user friendly

Table 2.1 Personalized News Aggregator

NEWS AGGRAGATOR AND EFFICIENT SUMMARIZATION SYSTEM Alaa Mohamed , Marwan Ibrahim , Mayar Yasser, Mohamed Ayman, Menna Gamil, Walaa Hassan	
Problem Solved	Collect news from various sources all in one place
Proposed solution	Aggregate news from cloud service
Result	Aggregate relevant article of a certain input keyword and summarized it efficiently.
merit	Text enhancement, Text summarization
demerit	Word frequency algorithm
Future work	Aggregate Personalized news based on user's interest

Table 2.2 News Aggregator and Summarization system

2.2 DRAWBACKS IN THE EXISTING SYSTEM

1.Multi languages support was not provided. The user's intended to read news only in the language in which news aggregator was created.

2.For Summarization system, a summary evaluation tool named 'Rouge" for summarization comparisons between Text Rank and Word Frequency algorithms turns out to be inefficient.

3.Sites that have unorganized content were the real problem behind the low traffic.

4.Broken links or links that do not direct to the real story can create a problem.

5.Readers won't visit the site if the news aggregator concentrates more on international news that might not be related to the reader who is intended to read local news.

2.3 PROPOSED SYSTEM

There are multiple news websites, they also cover news on several broad topics, out of which only a few of them are of our interest. A news aggregator can be a tool which can provide news on various domains such as Technology, Science, Climate etc. It can save a lot of time by providing both local and international news on one site. Since Django supports multilingual websites, user can view the content based on their known language. The application displays the news in short and crisp manner so that user won't waste time in reading the news briefly and even if the user wants to read the news elaborately then the user can access the link that redirects to the original webpage. The total size of the web

application is very small so the website won’t take much time to load the content. It has a simple user interface with easy navigation to all the other links.

2.4 BENEFITS OF THE PROPOSED SYSTEM

1.The web application doesn’t need any extra APIs or plugins to extract news from various sites.

2.The news aggregator contains all the valuable information in the main page. The user can read the news based on their interest.

3.The mobile friendly news aggregator site allows the user to check the news feed using smartphone.

4.The news is scraped from the trusted sources so there won’t be any fake news

CHAPTER 3

SYSTEM DESCRIPTION

3.1 SYSTEM DESCRIPTION

The system's implementation is developed using Python and the project runs as a web application developed using Django and BeautifulSoup library.

3.2 PYTHON 3.6.3

Python is an object-oriented programming language that can be used for various kind of software development. It offers active support for integration with other languages and tools. This comes with extensive standard libraries, and can be learned in a few days. Many Python programmers report substantial productivity gains and feel the language encourages the development of high quality and more accurate code. Python runs on Windows, Linux, Mac OS X, OS/2, Unix, Amiga, Palm Handhelds, and Nokia mobile phones. Python has also been combined to the Java and .NET virtual machines. Python disseminates under an OSI-approved open-source license which makes it free usage, even for commercial products.

3.3 DJANGO

Django is a Python web framework that provides pragmatic design, clean and rapid development. Django helps to write software that is complete, secure, maintainable, versatile sand portable. To develop News aggregator, three major

parts are required. They are database, user interface and the functions to interact in between. Django framework provide sufficient functionalities to implement these three parts in the form of model, view and template.

3.4 WEB SCRAPING

Web scraping is defined as extraction of data from a website. This information is collected and then exported into a format such as spreadsheet or an API and then the data can be retrieved and analyze and use the data the way we want. It can extract data from any website no matter how large is the data in the website. Web scraping software uses programming language such as JavaScript, Python, Go or PHP.

Fig 3.1 Web Scraping

To know whether a website allows web scraping or not, website's "robots.txt" file can be checked. The file can be found by appending "/robots.txt" to the URL of the scraped website.

3.4.1 Features of python for web scraping

1.Collection of Libraries:

Python has a variety of libraries such as NumPy, Pandas that provides methods and services for various purposes. It is suitable for web scraping and for manipulation of extracted data.

2.Dynamically typed:

In Python, without defining the datatypes, the variables can be directly used wherever required. So, it is time efficient.

3.Easily Understandable Syntax:

Python syntax are written in English so it is easily understandable. The indentation used in Python helps the programmer to differentiate between different blocks in the code.

4.Small code, large task:

In Python, the code for web scraping is small. These small codes do large tasks and saves time.

3.4.2 Scraping data from website

A request is sent to the URL while running the code for web scraping. The server sends the data as a response to the request and allows the programmer to read the HTML page. The code then, parses the HTML page and finds the data. Then the data is extracted.

The steps to be followed to extract data are,

Step 1: Identify the URL to scrape the data

Step 2: Inspect the Webpage

Step 3: The data to be extracted is identified

Step 4: Write the code for scraping the data

Step 5: Run the code and extract the data from the webpage

Step 6: Store the data in the database

3.5 LIBRARIES FOR WEB SCRAPING

3.5.1 Beautifulsoup

The Beautiful Soup is a python library which comes from the name Lewis Carroll poem which contain the same name in “Alice’s Adventures in the Wonderland”. Beautiful Soup is a Python library that makes the programmer to scrape the information from web pages easily. It acts on top of an HTML or XML parser, which helps to iterate, search and modify the parse tree.

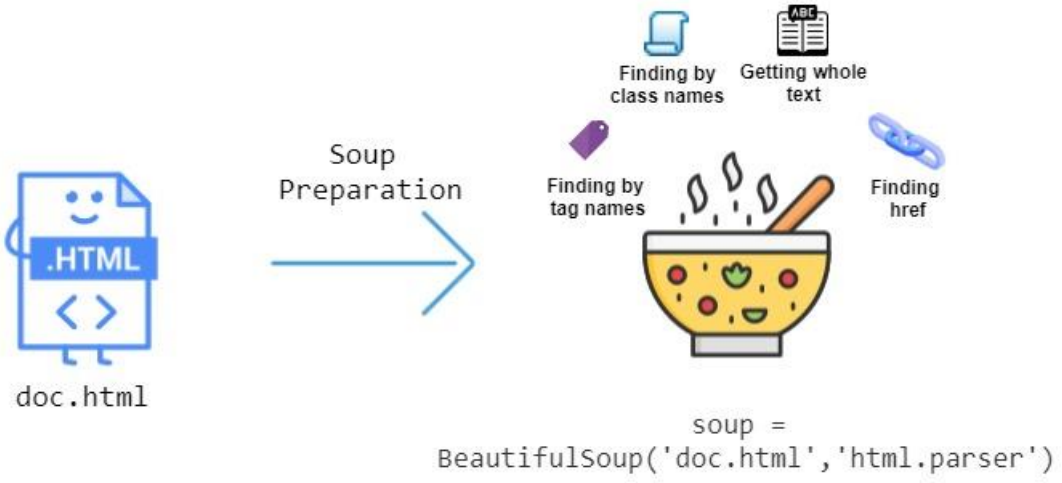

Fig 3.2 Working of BeautifulSoup

Beautiful Soup has been developed as a Python 2 library. Then the library is automatically converted to Python 3 code.

3.5.2 Html5lib

Html5lib is a python library for parsing HTML. It conforms the WHATWG HTML specification, as it is implemented in all web browsers. It can parse all the elements in HTML doc, breaking down the different tags and pieces which can be filtered for many use cases. It parses the text in the same way done in major web browsers. It tackles broken HTML tags and add some needed tags to complete the structure. It is written in python code.

3.5.3 Requests Module

Requests is a Python library that sends all kinds of HTTP requests. It has lot of features ranging from passing parameters to the URLs to send custom headers and SSL Verification.

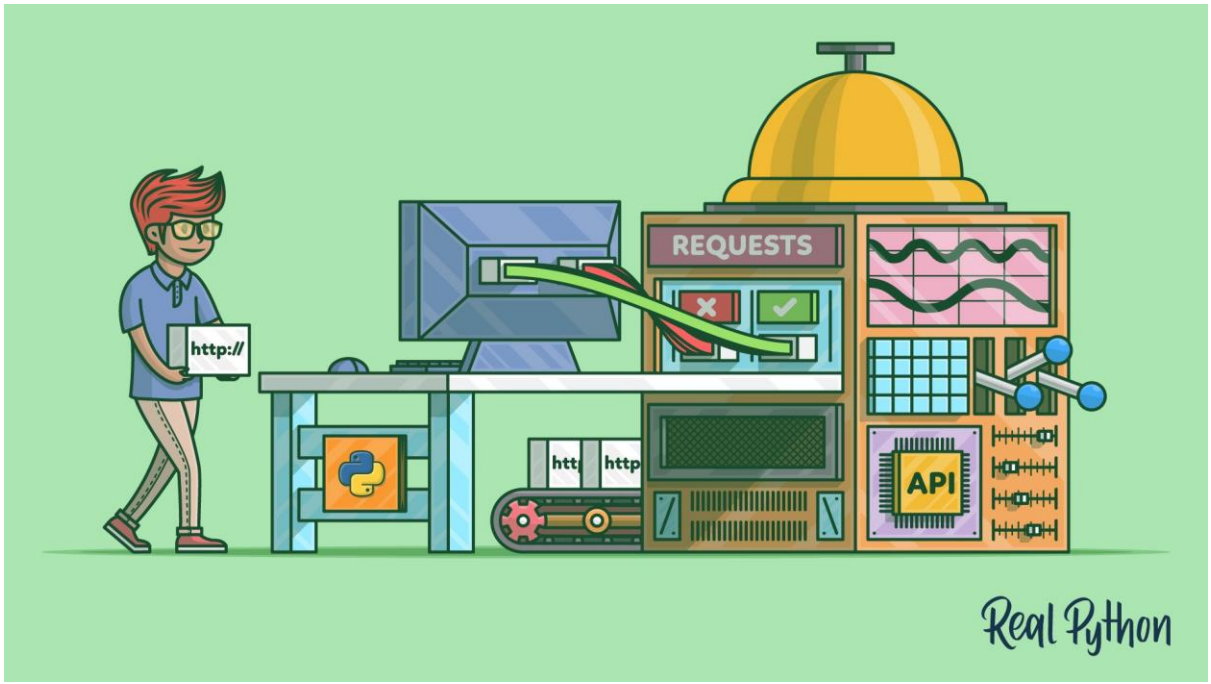

Fig 3.3 Working of Requests module

It abstracts the complexity of making requests behind the simple API so that it helps to interact with services and consume data in application. Requests

is an Apache2 Licensed HTTP library. This permit to send HTTP/1.1 requests with the help of Python. The method for installation of requests on any OS is to download the base files and install requests.

CHAPTER 4

SYSTEM DESIGN

4.1 COLLECTION OF DATA

The dataset for this project has been collected from various open-source Websites. The request module allows us to send http request using python. The HTTP request returns a response object with all the response data. The content is scraped using html5lib parser. This is done by using beautifulsoup library. The scraped content is stored in the SQLite3 database. Then the intended data is extracted from the whole scraped content. The information is scraped from the news website and it is stored as object in the database.

4.1.1 ATTRIBUTES REQUIRED FOR WEB SCRAPING

First, beautifulsoup has to be installed for scraping the data. The terms and condition of the site is to be checked before scraping as they likely have some rules to govern the data. The site from which the data is scraped might change the layout. So, the website has to be visited at a regular interval of time to make sure that a layout is unchanged. In case if there is a change the code has to be rewritten.

4.1.2 DESCRIPTION OF SQLITE DATABASE

SQLite is an embedded database for client storage in application such as web browsers. It is the most widely deployed database engine, as it is used by several widespread browsers, operating systems, and embedded systems (such as

mobile phones), among others. SQLite is included in many programming languages. One such language is Python. SQLite stores the entire databases such as definitions, tables, indices etc. on a host machine as a single cross-platform file. It is implemented by locking the entire database file during writing. Read operations in SQLite can be multitasked, but write operations can be performed in sequential order only. Applications using SQLite require less configuration than client–server databases due to the server-less design.

HEADLINE	URL	TIMES TAMP
Passport office has refused to issue my passport, says Mehbo.	http://timesofindia.indiatimes.com/articleshow/81744672.cms?utm_source=contentofinterest&utm_medium=text&utm_campaign=cppst	Updated: Mar 29, 2021, 17:07 IST
8 states account for over 84 per cent of India's fresh Covid ...	http://timesofindia.indiatimes.com/articleshow/81744674.cms?utm_source=contentofinterest&utm_medium=text&utm_campaign=cppst	Mar 29, 2021, 14:13 IST
Can't imagine an Indian side without Rishabh Pant: Ian Bell	http://timesofindia.indiatimes.com/articleshow/81745106.cms?utm_source=contentofinterest&utm_medium=text&utm_campaign=cppst	Mar 29, 2021, 15:00 IST

Guterres says UN negotiating with China on unfettered access.	http://timesofindia.indiatimes.com/articleshow/81745808.cms?utm_source=contentofinterest&utm_medium=text&utm_campaign=cppst	Mar 29, 2021, 16:31 IST

Table 4.1 Sample Data

4.2 ARCHITECTURE OF DJANGO

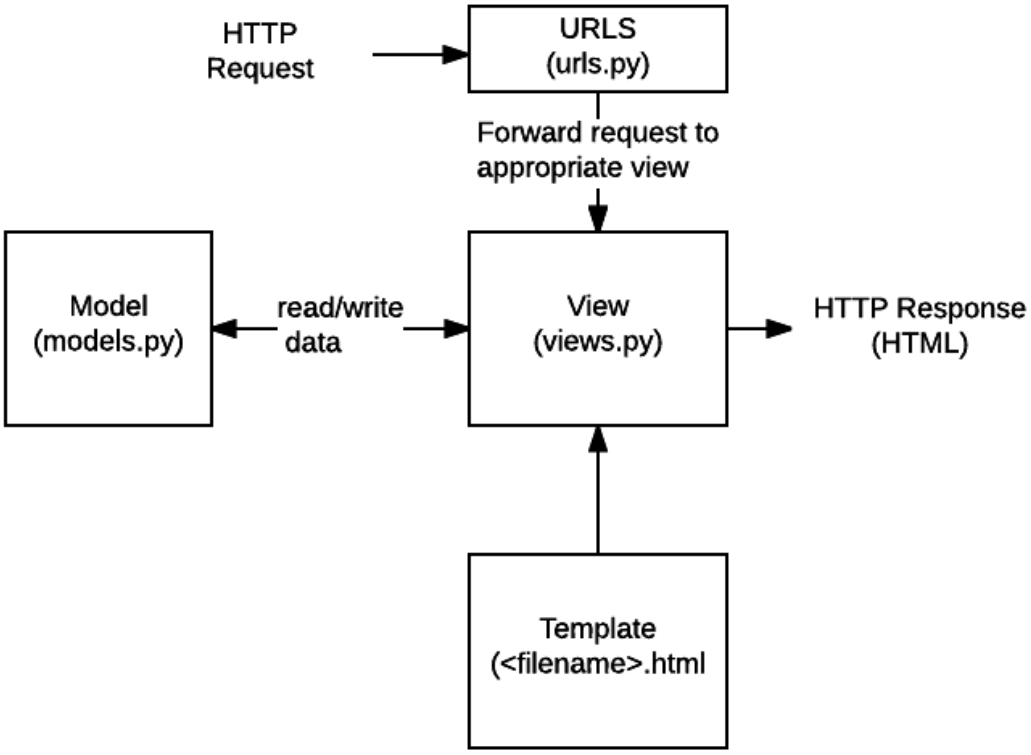

Fig 4.1 Architecture diagram of Django

Django is framed on MVT architecture. The MVT architecture is a software design pattern for developing a web application. MVT has three parts.

4.2.1 Model

The model act as the interface of data. It is responsible for maintaining data. It can be imagined as the logical data structure behind the complete application. It is represented by a database (generally relational database such as MySQL, Postgres).

4.2.2 View

The view is a user interface. It is the one which is seen in the browser when the website is rendered. View is nothing but look and feel of a website. Whenever the website is open the UI which is used for interaction is all due to the view function. It is represented by HTML, CSS, JS and jinja files. A view is a python function. It sends a web request and gets web content as a response. The response can be an HTML content of a web page or a redirected site link, or a “404 error”, or an XML document, or an image etc. But this page has to be associated with a URL to view it as a webpage.

4.2.3 Template

A template consists of static parts to display the output. It is nothing but the HTML file. It also contains syntax for describing how dynamic content will be inserted.

4.3 WORKING OF DJANGO

A user request for resources from Django. Django works as a controller and check the available resource in the URL. If URL maps a view is called that interacts with a model and template. It renders the template. Django then response back to the user and sends the template as a response.

4.4 URL MAPPING

When a user makes a request to a webpage, Django controller takes over to look for the corresponding view via the url.py file. It will return an HTML content as a response or a 404 not found error, if the page is not found. In urls.py the important thing is the “urlpatterns” tuple. It is where the mapping between URLs and views are defined. The mapping is composed of three elements.

1.Pattern

A regular expression matching the URL which is to be resolved and mapped. Everything that can work with a python re module is eligible for the pattern. It is useful when a parameter is passed via URL.

2.The python path to the view

The python path to the view is same as importing a module.

3.Name

For URL reversing, the named urlpatterns are used.

4.5 BENEFITS OF DJANGO ARCHITECTURE

The Django Framework is an architecture which communicates between all the three components without the need of complex code. The advantages of Django are

1.Rapid Development

Django has many different components that makes it easy for simultaneous development of same application at the same time by multiple developers.

2.Loosely coupled

Django has many components which require one another at some part of the application, that leads to rise of security of the overall website.

3.Ease of Modification

This is the main advantage of Django architecture. If there is a change in any component, other component change is not required. It provides more adaptability for website than other frameworks.

4.6 ALGORITHM

Step 1: The template for the news aggregator web application is created which is nothing but the HTML files which is going to serve the scraped data.

Step 2: The HTML files are created for homepage, English news and Tamil news.

Step 3: In views.py file, the code for scraping each website is written.

Step 4: For English news, the data is scraped from NDTV news website("https://www.ndtv.com/world-news") and Times of India news("https://timesofindia.indiatimes.com/briefs") website.

Step 5: For Tamil news, the data is scraped from Dailythanthi news website("https://www.dailythanthi.com/") and Puthiyathalaimurai news website("http://www.puthiyathalaimurai.com/").

Step 6: For home page, the data is scraped from mashable.com website for Technology news ("https://in.mashable.com/tech/")
and science news ("https://in.mashable.com/science/").

Step 7: Using request module, the HTTP request for websites is made. The content of the website is returned as the response.

Step 8: With the help of html5lib parser, the parse tree for corresponding website is created.

Step 9: From the parse tree, the tag which contains the news headlines is found and the data is extracted from the respective tags.

Step 10: Now the text is formatted and it is sent to the corresponding html page.

Step 11: In urls.py file, provide the corresponding mapping for the URL and the view function corresponding to it.

Step 12: Finally, open the command prompt from the root folder and execute the command “python manage.py runserver” for the execution of the news aggregator web application on a local host.

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 SYSTEM FLOW

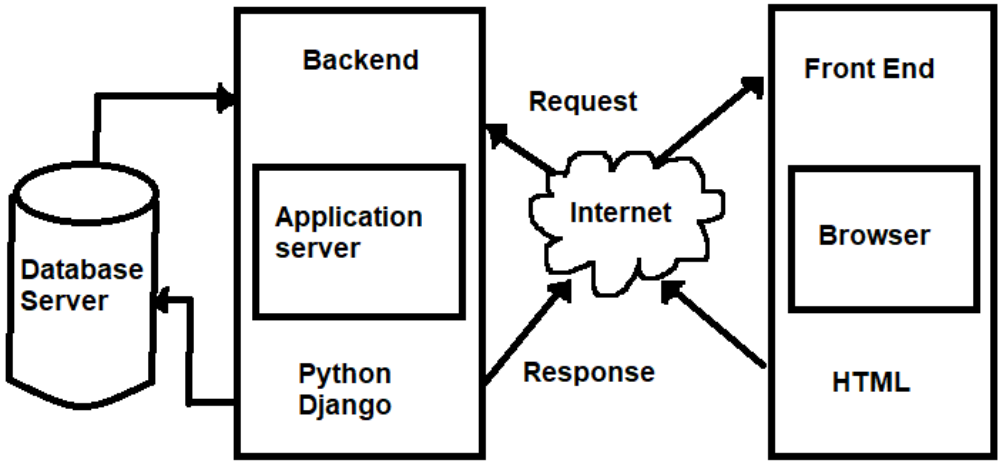

Fig 5.1 Flow diagram of News Aggregator

The system workflow for new aggregator involves three stages. They are scraping the website, storing the data and serving the stored database objects in the webpage.

5.2 SETTING UP ENVIRONMENT

Django is installed. BeautifulSoup library is installed using pip. The webpage is created using HTML and CSS. Using “python manage.py runserver” command, the emulated server on the local computer is run.

5.3 IMPLEMENTATION

Step 1: First run the server on the local computer.

Step 2: The request is sent for the selected websites using Internet.

Step 3: The website will accept the request and respond to it.

Step 4: All the data from the website are scraped using web scraper.

Step 5: The information is stored in the database.

Step 6: The news headlines are extracted from the stored data using BeautifulSoup library.

Step 7: The HTML pages for new aggregator is created. Buttons are created for navigation to different language news.

Step 8: The technology and science news are summarized in two different partitions in the main page.

Step 9: On clicking the button for English news, the headlines from “Times of India” and NDTV are displayed. On clicking the button for Tamil news, the headlines from “Dailythanthi” and “Puthiyathalaimurai” are displayed. Further CSS is used for enhancing the text for better readability.

5.4 CODE:

Views.py

```
from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
import requests
from bs4 import BeautifulSoup

#ENGLISH
# Getting news from Times of India

toi_r = requests.get("https://timesofindia.indiatimes.com/briefs")
toi_soup = BeautifulSoup(toi_r.content, 'html5lib')
toi_headings = toi_soup.find_all('h2')
toi_headings = toi_headings[0:-13]
```

```
toi_news = []

for th in toi_headings:
    toi_news.append(th.text)

toi_news=toi_news[2:]

#Getting news from NDTV
hint = requests.get("https://www.ndtv.com/world-news")
hint_soup = BeautifulSoup(hint.content, 'html5lib')
hint_headings = hint_soup.find_all("h2",{"class":"newsHdng"})
hint_news = []

for h in hint_headings:
    hint_news.append(h.text)
    if len(hint_news)>20:
        break

#TAMIL

#Getting news from Dailythanthi
ht_r = requests.get("https://www.dailythanthi.com/")
ht_soup = BeautifulSoup(ht_r.content, 'html5lib')
ht_head=ht_soup.find("div",{"class":"NewsWithTopImage"})
ht_headings = ht_soup.findAll("span", {"class": "abstract"})
ht_news = []
ht_news.append(ht_head.get_text())

for hth in ht_headings:
    ht_news.append(hth.text)

#Getting news from Puthiyathalaimurai
pt = requests.get("http://www.puthiyathalaimurai.com/")
pt_soup = BeautifulSoup(pt.content, 'html5lib')
pt_headings = pt_soup.find_all("div",{"class":"banner-desc"})
pt_tech=pt_soup.find_all("div",{"class":"text"})
pt_news = []
```

```
for p in pt_headings:
   pt_news.append(p.text)
for t in pt_tech:
        pt_news.append(t.text)

#Technology
g=requests.get("https://in.mashable.com/tech/")
g_soup = BeautifulSoup(g.content, 'html5lib')
g_headings = g_soup.find_all("a",{"class":"lnk"})

g_news = []

for i in g_headings:
   g_news.append(i.text)

#Science
s=requests.get("https://in.mashable.com/science/")
s_soup = BeautifulSoup(s.content, 'html5lib')
s_headings = s_soup.find_all("a",{"class":"lnk"})

s_news = []

for i in s_headings:
   s_news.append(i.text)


def index(req):
        return render(req, 'news/index.html', {'g_news':g_news, 's_news':
s_news})

def eng(req):
        return render(req, 'news/english.html',
{'toi_news':toi_news,'hint_news':hint_news})

def tam(req):
```

```
    return render(req, 'news/tamil.html', {'ht_news': ht_news,'pt_news':pt_news})
```

index.html

```
<!DOCTYPE html>
<html>
<head>
   <title></title>
   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

   <style type="text/css">

       body{
      background-color: rgb(235, 235, 224);
   }

     .jumbotron{
      background-color: rgb(255, 255, 102);
   }

      .zoom {
  padding: 10px;
  background-color: rgb(204, 255, 255);
  transition: transform .2s;
  width: 450px;
  margin: 0 auto;
  border-color: black;
  border-radius: 25px;
  border-style: solid;
}
```

```
.zoom:hover {
  -ms-transform: scale(1.5); /* IE 9 */
  -webkit-transform: scale(1.5); /* Safari 3-8 */
  transform: scale(1.5);
}


    </style>
</head>
<body>
    <div class="jumbotron">
        <center><h1>News Aggregator</h1>
          <a href="/" class="btn btn-danger">Refresh News</a>
          <a href="{% url 'english'%}" class="btn btn-danger">English</a>
          <a href="{% url 'tamil'%}" class="btn btn-danger">Tamil</a>

    </center>
    </div>
    <div class="container">
        <div class="row">
            <div class="col-6">
                    <h3 class="text-centre">Technology</h3>
                    {% for n in g_news %}
                    <h5 class="zoom"> -  {{n}} </h5>
                    <hr>
                    {% endfor %}
                    <br>
            </div>
            <div class="col-6">
                    <h3 class="text-centre">Science</h3>
                    {% for s in s_news %}
                    <h5 class="zoom"> - {{s}} </h5>
                    <hr>
                    {% endfor %}
                    <br>
            </div>
        </div>
```

```
</div>
   <script
src="http://code.jquery.com/jquery-3.3.1.min.js"
integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
   crossorigin="anonymous"></script>
   <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvf
a0b4Q" crossorigin="anonymous"></script>
   <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PV
CmYl" crossorigin="anonymous"></script>
</body>
</html>
```

english.html

```
<!DOCTYPE html>

<html>

<head>

   <title></title>

   <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/d
AiS6JXm" crossorigin="anonymous">



   <style type="text/css">
```

```
        body{
        background-color: rgb(235, 235, 224);
    }

     .jumbotron{
        background-color: rgb(255, 255, 102);
    }

        .zoom {
  padding: 10px;
  background-color: rgb(204, 255, 255);
  transition: transform .2s;
  width: 450px;
  margin: 0 auto;
  border-color: black;
  border-radius: 25px;
  border-style: solid;
}

.zoom:hover {
  -ms-transform: scale(1.5); /* IE 9 */
  -webkit-transform: scale(1.5); /* Safari 3-8 */
  transform: scale(1.5);
    </style>
```

```
</head>
<body>
  <div class="jumbotron">
    <center><h1>English News</h1>
    <a href="/" class="btn btn-danger" >Home</a>
    <button type="button" onClick="reloadThePage()" class="btn btn-danger">Refresh</button>
  </center>
  </div>
  <div class="container">
    <div class="row">
      <div class="col-6">
          <h3 class="text-centre"> News from Times of india</h3>
          {% for n in toi_news %}
          <h5 class="zoom"> -  {{n}} </h5>
          <hr>
          {% endfor %}
          <br>
      </div>
      <div class="col-6">
          <h3 class="text-centre">News from NDTV</h3>
          {% for htn in hint_news %}
          <h5 class="zoom"> - {{htn}} </h5>
          <hr>
          {% endfor %}
```

```
                <br>
            </div>
        </div>



</div>
    <script
src="http://code.jquery.com/jquery-3.3.1.min.js"
integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
    crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvf
a0b4Q" crossorigin="anonymous"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PV
CmYl" crossorigin="anonymous"></script>
    <script>
function reloadThePage(){
    window.location.reload();
}
</script>
</body>
</html>
```

urls.py

```
from django.contrib import admin

from django.urls import path

from news import views



urlpatterns = [

    #path('admin/', admin.site.urls),

    path('', views.index, name = "home"),

    path('english-page', views.eng, name="english"),

    path('tamil-page',views.tam,name="tamil")

]
```

settings.py

```
import os
from pathlib import Path

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent



# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '9956#!4sz4i7drr%y1=3u318b#+s64d5y+26ktm-oi^#q*26r1'
```

```
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'news',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'news_agg.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
```

```
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'news_agg.wsgi.application'


# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}


# Password validation
# https://docs.djangoproject.com/en/3.1/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
```

```
        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/3.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.1/howto/static-files/

STATIC_URL = '/static/'
```

apps.py

```
from django.apps import AppConfig
class NewsConfig(AppConfig):
    name = 'news'
```

CHAPTER 6

TESTING

6.1 TESTING THE WEB PAGE

Testing demands that the developer discards the presumed notions of correctness of the software just developed and overcome a conflict of interest that occurs when errors are uncovered. There are several rules that can serve well as testing objectives. The aim of testing is to find the errors on executing the program. A good testcase is found to be the one which has a greater probability of finding an undiscovered error. A successful test is one that uncovers an as-yet undiscovered error. Initially the environment checking is done to see if all the necessary requirements are installed or not.

Installation of Python 3.6.3, beautifulSoup, request, html5lib are confirmed. By performing web application testing, it is assured that the applications will work smoothly and it will be easily accepted by the end-users. A wide range of testing techniques are used to test the web apps for their quality and function. Such techniques test the web application for quality, and also test the web application's browser compatibility, load testing, scalability testing and so on.

6.1.1 Functional Testing

The main aim of functional testing is to check all the functions within a web application are working smoothly without any technical problems. In a web application, functional testing is done to check whether all the links are working

properly or not, testing cookies, validating HTML or CSS, testing database for the security and so on.

6.1.2 Web UI Testing

The most important interfaces in a web application are web server and application server interface and database server interface. Web UI testing will confirm that all the components in a web application are connected appropriately.

6.1.3 Compatibility Testing

Compatibility testing is the most crucial thing while testing the application. Compatibility testing will check the website for browser compatibility, operating system compatibility, mobile browsing and printing options.

6.2 RESULT ANALYSIS

The system has been tested and web application opens properly when it is executed on the local host. The content has been extracted from the right tags mentioned in the scraping program. The extracted data is displayed based on the topics like technology, science. By selecting the language, the news from the specified website is scraped and the content is displayed in the form of blocks.

6.3 OUTPUT

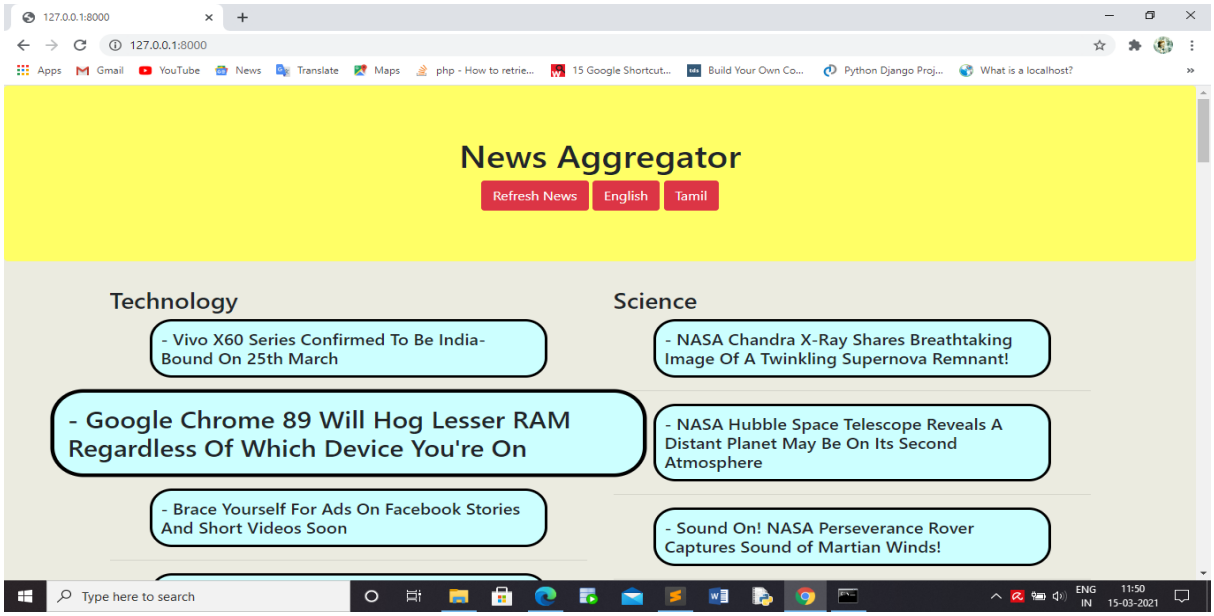

Fig 6.1 News Aggregator Home Page

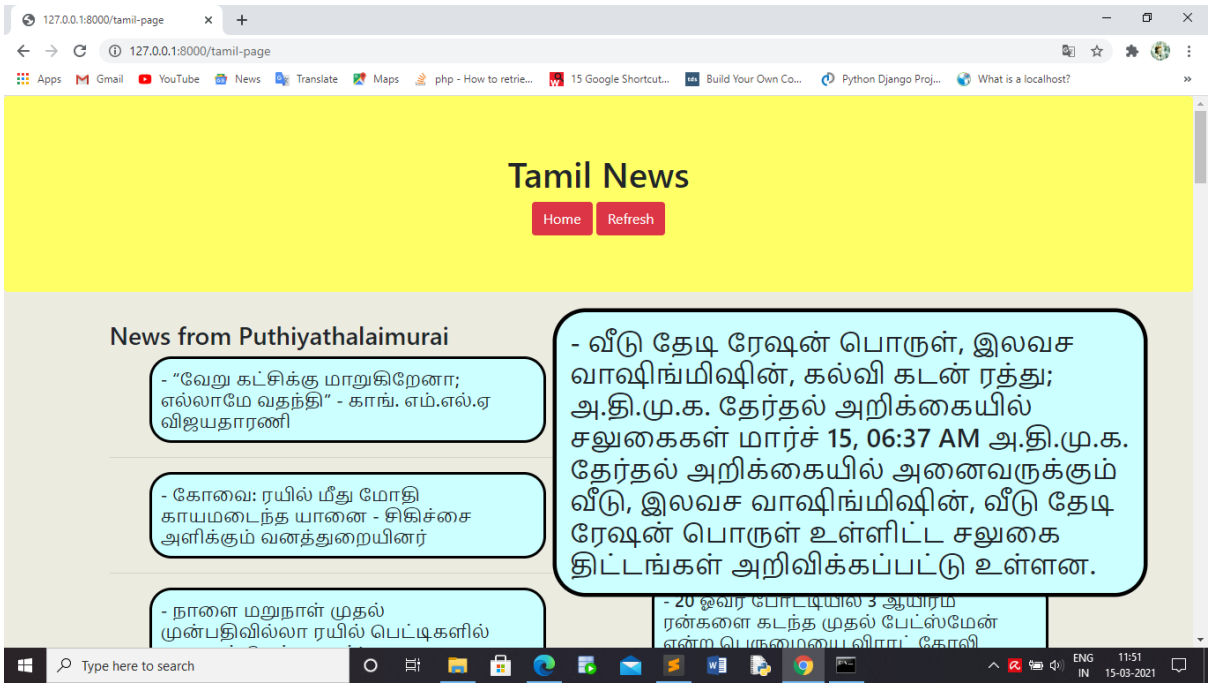

Fig 6.2 News Aggregator Tamil News Page

Fig 6.3 News Aggregator English News Page

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

The application created can successfully scrape the news and display the headlines of the news. The whole size of the application is very small and can be opened in any browser. The news aggregator web application can save lot of time for the user who need a news in short and crisp manner. In aggregation process, the data which is gathered from the website are the latest that is available. The reader can click on the news which the reader wants to read in brief. This is a generalized project and compatible with all the system.

7.2 FUTURE WORK

1.The user can subscribe to their favourite news website and articles so that they get personalized content based on their interest.

2.Search bar can be included so that user can search the news which the user wants to read.

3.Depending on the preference, the reader can choose the location so that the news from that particular location should be displayed.

4.Weather and climate condition of the user's location should be displayed in the webpage.

5.Search Engine Optimization can be done to increase the traffic to the web application and to the publisher.

6.Web application can be customized with images for the corresponding news.

REFERENCES

1. Alaa Mohamed, Marwan Ibrahim, Mayar Yasser, Mohamed Ayman, Menna Gamil and Walaa Hassan, (2020), “News Aggregator and Efficient Summarization System” International Journal of Advanced Computer Science and Applications (IJACSA),11(6).

2. Bahana R, R. Adinugroho, B. S. Abbas;"Webcrawler and backend for news aggregator application (Noox project), " (2017) IEEE International Conference on Computational IntelligencePhuket,2017, pp.561doi:10.1109/CYBERNETICSCOM.2017.8311684.

3. Deepak Mahto, March (2014), “A Web Scraper World”, IEEE Transaction on Computing for Sustainable Development, Vol.2, No.1.

4. Fister, S. Fong, Yan Zhuang, (2014), "Data Reconstruction of Abandoned Websites", IEEE 2nd International Symposiumon Computational and Business Intelligence (ISCBI).

5. Karale B, March (2016) “News headline extraction from Internet”, IEEE Transaction on Computing for Sustainable Development, Vol.45, No.1

6. Kolari P, (2014), " Scraping web using Google API service”, IEEE Transactions on Knowledge, Data Engineering, Vol.6, No.4.

7. Malik S.K, S.M. Ravi, (2011),"Information Extraction Using Web Usage Mining Web Scrapping and Semantic Annotation", IEEE International Conference on Computational Intelligence and Communication Networks (CICN), 2011.

8. Natalia LF, (2015) "The use of web scraping in computer parts assembly price comparison," 3rd International Conference New Media, Indonesia, 2015, /CONMEDIA.2015.7449152.

9. Paliouras, Georgios & Mouzakidis, Alexandros & Moustakas, Vassileios & Skourlas, Christos. (1970). PNS: A Personalized News Aggregator on the Web. 10.1007/978-3-540-77471-6_10.

10. Pratiba D, A. Dua, and U. Singh, (2018)"Web Scraping &Data Extraction with Google Scholar," 3rd International Conference on Computational Systems& Information Technology for Sustainable Solutions, Bengaluru, India, 2018, pp. 277-281, doi: 10.1109/CSITSS.2018.8768777.

11. Quang Thai Le, D Pishva, (2015),"Application of Web Scraping and Google API service to optimize convenience stores distribution", 17th IEEE International Conference on Advanced Communication Technology (ICACT), 2015.

12. Sandeep and Vinay, 23 June (2016), “Extraction of contents from news web pages using pattern matching” IEEE Transaction on Web Research (ICWR)

13. Sandeep Sirsat and Vinay Chavan , 23 June (2016), “ Pattern matching for extraction of core contents from news web pages” IEEE Transaction on Web Research (ICWR).

14. TehPohey Lee, Abdul Azim Abdul Ghani and Chang Yu Huang,(2008), ”Survey on application tools of Really Simple Syndication (RSS): A case study at Klang Valley”, Vol.3.

15. Zhao, Bo. (2017). Web Scraping. 10.1007/978-3-319-32001-4_483-1.

16. Zixuan Zhuang Mehdi Bahrami, Mukesh Singhal, (2015), "A cloud-based web crawler architecture", 18th International Conference on Intelligence in Next Generation Networks 978-1-4799-1866-9, pp. 216- 223, 2015.

17. https://www.hackersfriend.com/articles/building-news-aggregator-web-app-with-django-using-python-web-scraping

18. https://data-flair.training/blogs/django-project-news-aggregator-app/

19. https://www.edureka.co/blog/web-scraping-with-python/

20. https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/

21. Video reference for URL mapping in news aggregator https://www.youtube.com/watch?v=gvdSkBmjpbY

22. Video reference for database connection https://www.youtube.com/watch?v=1jcjuV2Stx