News Aggregator Application

A Major/Minor Project Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

BY
Aman Chauhan (EN17EL301012)
Abhishek Patidar (EN17CS301012)
Aayush Purohit (EN17CS301006)

Under the Guidance of
Prof. Rakesh Giri Goswami

Department of Computer Science and Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

AUG-DEC 2020

Report Approval

The project work News Aggregator Application is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approved any statement made, opinion expressed, or conclusion drawn there in; but approve the "Project Report" only for the purpose for which it has been submitted.

Internal Examiner

Name: Prof. Ruchi Patel

Designation: Professor at Medi-Caps University

External Examiner

Name:

Designation:

Declaration

We hereby declare that the project entitled News Aggregator Application submitted in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science and Engineering completed under the supervision of Prof. Rakesh Giri Goswami, Professor, Department of Computer Science and Engineering, Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Aman Chauhan(EN17EL301012)

Abhishek Patidar(EN17CS301012)

Aayush Purohit(EN17CS301006)

Date: 16/11/2020

Certificate

I Rakesh Giri Goswami certify that the project entitled News Aggregator Application submitted in partial fulfillment for the award of the degree of Bachelor of Technology by Aman Chauhan (EN17EL301012), Abhishek Patidar (EN17CS301012), Aayush Purohit (EN17CS301006). It is carried out by them under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

Prof. Rakesh Giri Goswami

Department of Computer Science and Engineering

Medi-Caps University, Indore

Prof. Ruchi Patel

Department of Computer Science and Engineering

Medi-Caps University, Indore

Dr. Suresh Jain

Head of the Department

Computer Science and Engineering

Medi-Caps University, Indore

Acknowledgements

We would like to express our deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided us with every facility to successfully carry out this project, and our profound indebtedness to **Prof. Sunil K Somani,** Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up our morale. We also thank **Prof. D K Panda,** Dean, Faculty of Engineering, Medi-Caps University, for giving us a chance to work on this project. We would also like to thank our Head of the Department **Prof. Suresh Jain** for his continuous encouragement towards the betterment of the project.

We express our heartfelt gratitude to our project guide, **Prof. Rakesh Giri Goswami**, Professor, Department of Computer Science and Engineering, Medicaps University, without whose continuous help and support, this project would never have reached to the completion.

We would also like to thank our project coordinators **Prof. Ruchi Patel** and **Prof. Sachin Solanki** who extended their kind support and help towards the completion of this project.

It is their help and support, due to which we became able to complete the project and technical report.

Without their support this report would not have been possible.

Aman Chauhan (EN17EL301012)
Abhishek Patidar (EN17CS301012)
Aayush Purohit (EN17CS301006)
B.Tech. IV Year
Department of Computer Science and Engineering
Faculty of Engineering
Medi-Caps University, Indore

Abstract

Aggregation technology helps to consolidate many websites into one page that can show only the new or updated information. News aggregator sites help to collect the latest information and stories which are happening all around the world. It is an easy way to read all the important news in a single web space.

This is to facilitate all people who are busy with their works and have no time to get their desired information. We provide an interactive and multiple feature oriented news aggregator site which helps every user to gain information all around the world and read the article in a concise manner.

This is used to produce news articles from different sources of news portal websites and provide the summary of those long articles. This helps the readers to gain a lot of information in a small span of time.

The main goal is to provide the users or readers of this aggregator portal to obtain the news articles in a customized format. Users can choose the category of news that they prefer the most. The categories of the news articles can be World, India, Entertainment, Technology and Sports. It performs sentiment analysis on those articles to divide them into positive, negative or neutral news. Recommended news articles are displayed to the readers to provide them information about popular news articles. Reader can also check its weather status through its IP address.

Some of the problems faced by the current news aggregator sites are the sentiment analysis and recommender system. Majority of them does not differentiate the positive, negative or neutral news. They mainly focus on categorical news articles and do not provide recommender system for the same. We provided these features to take aggregation technology to the next step.

This document fully and formally describes the requirements of the proposed said project system. It sets out the functional and non-functional requirements and includes a description of the user interface, documentation and training requirements.

Keywords: news aggregator; summary; sources; categories; sentiment analysis; articles; recommender system

Table of Contents

List of Figures

List of Tables

Introduction

1.1 Introduction of the Project

In recent years, the World Wide Web has experienced a self-feeding increase in the number of users and the quantity of content, data and services. More content makes the Web more interesting for more users, who in turn create more content. A typical example of this is the news industry, which seems to be turning fully online and trying to follow the developments in Web publishing. Most of the news publishers have introduced electronic versions of their content, which in many cases are much richer in structure than the traditional paper versions. Furthermore, certain new websites have emerged which acts as an intermediary to provide the news from different sources and put it in one frame. These websites provide us a news aggregating services. This helps the end user to gain access to an enormous volume of information, which apart from its clear positive side brings along the problem of information overload. The task of finding interesting information in all that is within reach is as daunting and frustrating for the non-expert user as looking for a needle in a hay-stack. Thus, we need to help provide a simple mechanism which enables the end user to get the popular and categorical based news articles based on the preference.

A content aggregator is a website that collects different content including news articles, social media posts, images, and videos on particular issues from around the web and makes them accessible in one place. Aggregation features are frequently built into web portal sites, in the web browsers themselves, in email applications or in application software designed specifically for reading feeds

News aggregator websites is a type of content aggregator website which allows users to view

news and updates from various sources at one convenient location. The feeds are often in the RSS or Atom formats which use Extensible Markup Language (XML) to structure pieces of information to be aggregated in a feed reader that displays the information in a user-friendly interface.

1.2 Literature Review

A news aggregator provides and updates information from different sources in a systematized way. "Some news aggregator services also provide update services, whereby a user is regularly updated with the latest news on a chosen topic." Websites such as Google News, Huffington Post, World News (WN) Network and Daily Beast where aggregation is entirely automatic, using algorithms which carry out contextual analysis and group similar stories together, while other sites supplement automatically aggregated news headline RSS feeds from a number of reputable mainstream and alternative news outlets, while including their own articles in a separate section of the website. Some news aggregators offer subscription services to professionals.

News aggregation websites began with content selected and entered by humans; while automated selection algorithms were eventually developed to fill the content from a range of either automatically selected or manually added sources. There are many features that can be added to the news aggregator application.

1.2.1 Web Scraping

Web Scraping is a technique employed to extract large amounts of data from websites whereby the data is extracted and saved to a local file in your computer or to a database in table format.

Modern media can create outstanding value or an existential threat to your business - in a single news cycle. If you're a company that depends on timely news analyses, or a company that frequently appears in the news, web scraping is the ultimate solution for monitoring, aggregating and parsing the most critical news stories from your industry.

- Investment Decision Making
- Online Public Sentiment Analysis
- Competitor Monitoring
- Political Campaigns

Scaping the news articles can help to summarize that long-form article in a single paragraph with keywords, it would be easier to learn the context of that news quickly in a single frame.

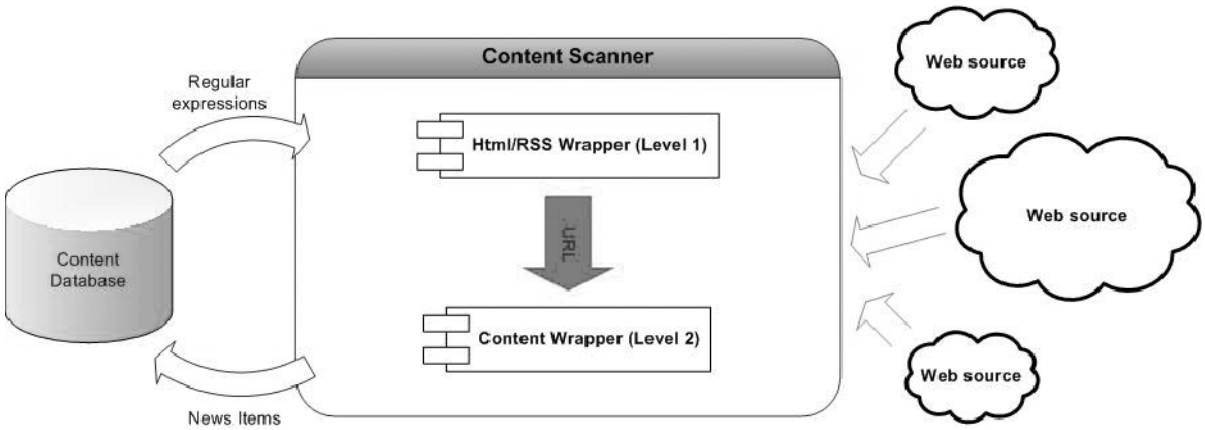

Figure 1.1 Content Aggregator

1.2.2 Sentiment Analysis and Text Summarization

The objective of sentiment analysis is to categorize the sentiment of public opinions by sorting them into positive, neutral, and negative. Sentiment analysis is often used in business to detect sentiment in social data, gauge brand reputation, and understand customers. It helps to produce the articles based on their neutrality. Users can decide and select the most appropriate articles based on their interest.

Text summarization refers to the technique of shortening long pieces of text. The intention is to create a coherent and fluent summary having only the main points outlined in the document. Furthermore, applying text summarization reduces reading time, accelerates the process of researching for information, and increases the amount of information that can fit in an area.

1.2.3 Weather forecasting application

Weather can be displayed easily on webpage using existing APIs suck as openweathermap, weathermap or weatherstack api. This can further work with geolocation APIs to display the weather of the user's location. Certain APIs are a free weather and geolocation API provider that provides extensive APIs that range from the weather forecast, historical weather, IP lookup, and astronomy through to sports, time zone, and geolocation.

1.2.4 Recommender System

Recommender systems are the systems that are designed to recommend things to the user based on many different factors. These systems predict the most likely product that the users are most likely to purchase and are of interest to.

In our project, we are using Popularity based recommender system. It is a type of recommendation system which works on the principle of popularity and or anything which is in trend. These systems check about the product or movie which are in trend or are most popular among the users and directly recommend those.

The basic merit of this recommender system is that it does not suffer from cold start problems which mean on day 1 of the business also it can recommend products on various different filters. There is no need for the user's historical data.

1.3 Objectives

The objective of this project is to design and develop a News Aggregator Application using Web Scraping and Natural Language Processing. The system will be designed using django to control the flow of the project and display the front end events by using the Python on its back end.

This project has the features of displaying the news articles based on different categories. Sentiment Analysis is also done to divide the news articles based on positive, negative or neutral news. News Recommendation is developed to produce the news articles based on popularity- based recommendation system.

1.4 Scope

The purpose of this project is to select and gather several news articles from different sources and merge them in a single web page. It helps the users to view the news articles based on his/her preferences. Popular news articles are also recommended to its users.

The major benefit of using an aggregator websites over particular news websites is to produce all the data around the world in a single place. It is not required for users to search all the websites to find a particular article.

This project also helps to summarize a long article into a short summary. Users will be able to read and evaluate the article in short span of time. Articles are also categorically divided based on positive, negative and neutral news. It helps the users to decide the news articles he/she required to view based on sentiment choices.

1.5 Problem in existing system

Majority of news aggregator sites provides no extra feature other than news aggregation. News articles are only displayed based on different categories. News aggregating application such as Inshorts provides the text summarization feature but other than that, it doesn't have any other different feature. Certain aggregating websites do not provide recommendation system and sentiment analysis feature. Users cannot choose to read the news articles based on sentiments i.e. positive, negative and neutral.

Our project deals with the existing problems and provides the recommendation system, text summarization and sentiment analysis features in a single space.

System Requirement Analysis

2.1 Information Gathering

Information Gathering is a very key part of the feasibility analysis process. In order to build a news recommendation system, we gathered information about various approaches to build a news aggregator application and we also gather information about various features that can be added in the application. We had discussed with every team member about its applications and also observed various aspects of existing applications like Inshorts, Dailyhunt etc .After proper information gathering, we found out that we can build the web application in python Django in which we had thought of aggregating various news articles from various news websites like India Today, Times of India and Indian Express by the help of beautiful soup library in python. We also found that features like sentiment analysis, news recommendation and text summarization can be added.

There is some more required data which needs to be added to build our project. Collection of twitter users information and converting them into csv file is also done to achieve the development to recommender system. Our machine required to train and test using Naïve Bayes classifier with many positive and negative sentences which we had collected to produce sentiment analysis feature in our project. Certain articles about news aggregator were also read by us to learn about every aspect of the project.

These steps were taken out by every team member in order gather information and were stored for future purposes.

2.2 System Feasibility

A feasibility analysis usually involves a thorough assessment of the operational (need), financial and technical aspects of a proposal. Feasibility study is the test of the system proposal made to identify whether the user needs may be satisfied using the current software and hardware technologies, whether the system will be cost effective from a business point of view and whether it can be developed with the given budgetary constraints. A feasibility study should be relatively cheap and done at the earliest possible time. Depending on the study, the decision is made whether to go ahead with a more detailed analysis.

When a new project is proposed, it normally goes through feasibility assessment. Feasibility study is carried out to determine whether the proposed system is possible to develop with available resources and what should be the cost consideration.

The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility of the system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

2.2.1 Technical Feasibility

Technical Feasibility deals with the hardware as well as software requirements. Technology is not a constraint to type system development. We have to find out whether the necessary technology, the proposed equipment have the capacity to hold the data, which is used in the project, should be checked to carry out this technical feasibility.

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipment's have the technical capacity to hold the data required to use the new system?
- Can the system be upgraded if developed?

- Are there technical guarantees of accuracy, reliability, ease of access and data security?

The current system developed is technically feasible. It is a web based system for reading news articles from different websites in a single space. Thus, it provides an easy access to the users. The development of the project only requires a single computer system backed by the internet connection. All users are permitted to the users and can look to particular information according to their requirement. Therefore, it provides the technical guarantee of accuracy, reliability and security. The work for the project is done with the current equipment and existing software technology.

2.2.2 Economical Feasibility

The feasibility study presents the tangible and intangible benefits from the project by comparing the development and operational cost. The technique of cost benefit analysis is often used as a basis for assessing economic feasibility. This system needs some more initial investment than the existing system, but it can be justifiable that it will improve quality of service.

Thus feasibility study should center along the following points:

- Improvement resulting over the existing method in terms of accuracy, timeliness.
- Cost comparison
- Estimate on the life expectancy of the hardware
- Overall objective

Our project is economically feasible. It does not require much cost to be involved in the overall process. It also does not require any addition hardware or software as the interface for this system is developed using the existing resources and technologies available. The overall objectives are in easing out the requirement processes.

2.2.3 Behavioural/ Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following:

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This analysis involves how it will work when it is installed and the assessment of political and managerial environment in which it is implemented. The system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The new proposed system is very much useful to the useful to the users and there for it will accept broad audience from around the world. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

2.3 Platform Specification (Development and Deployment)

A technical specification is a normative document on which the project development is based. It contains detailed requirements for the components and characteristics of the future web product.

2.3.1 Hardware Description

The selection of hardware is very important in the existence and proper working of any software. When selecting hardware, the size and requirements are also important. The minimum requirements which are required to develop the project are as follows:

Processor: Intel Core i5 processor 1,333 MHz

RAM: 8GB

Hard Disk Drive: 500GB

Monitor: Display Panel (1920 X 1080)

Display Adapter: Intel® UHD Graphics 620

Network Adapter: Microsoft Kernel Debug Network Adapter

2.3.2 Software Implementation Technology

The following software technology is required to develop the project:

Operating System: Windows 10 Home

Front- End: CSS, HTML, Java-Script, Bootstrap4

Back-End: Python3.8

Django is used to interlink the front-end and back-end and to test the project in the local server.

System Analysis

3.1 Overall Flow of the System

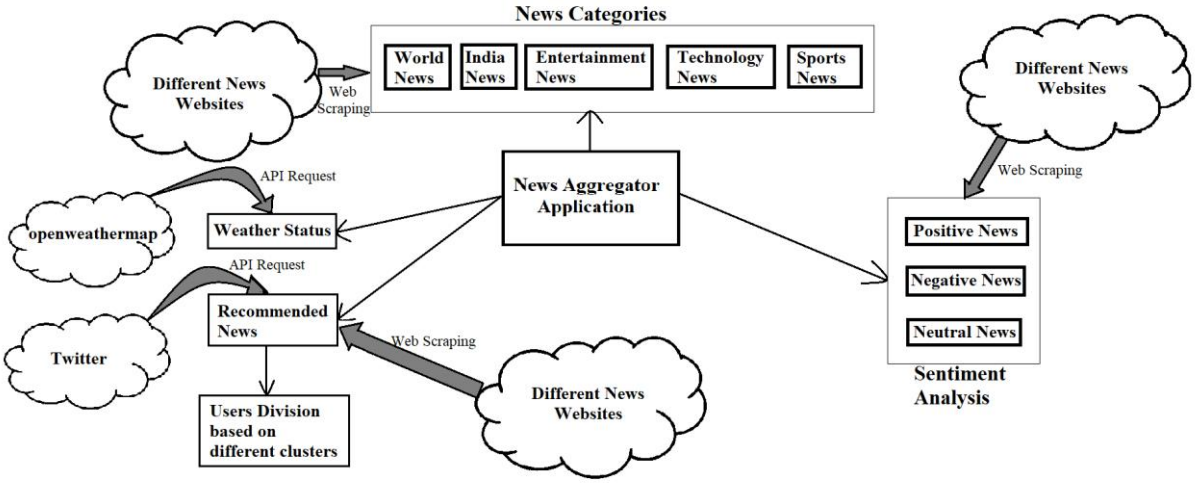

Figure 3.1 Overall Flow of the System

3.2 Use Case Diagram

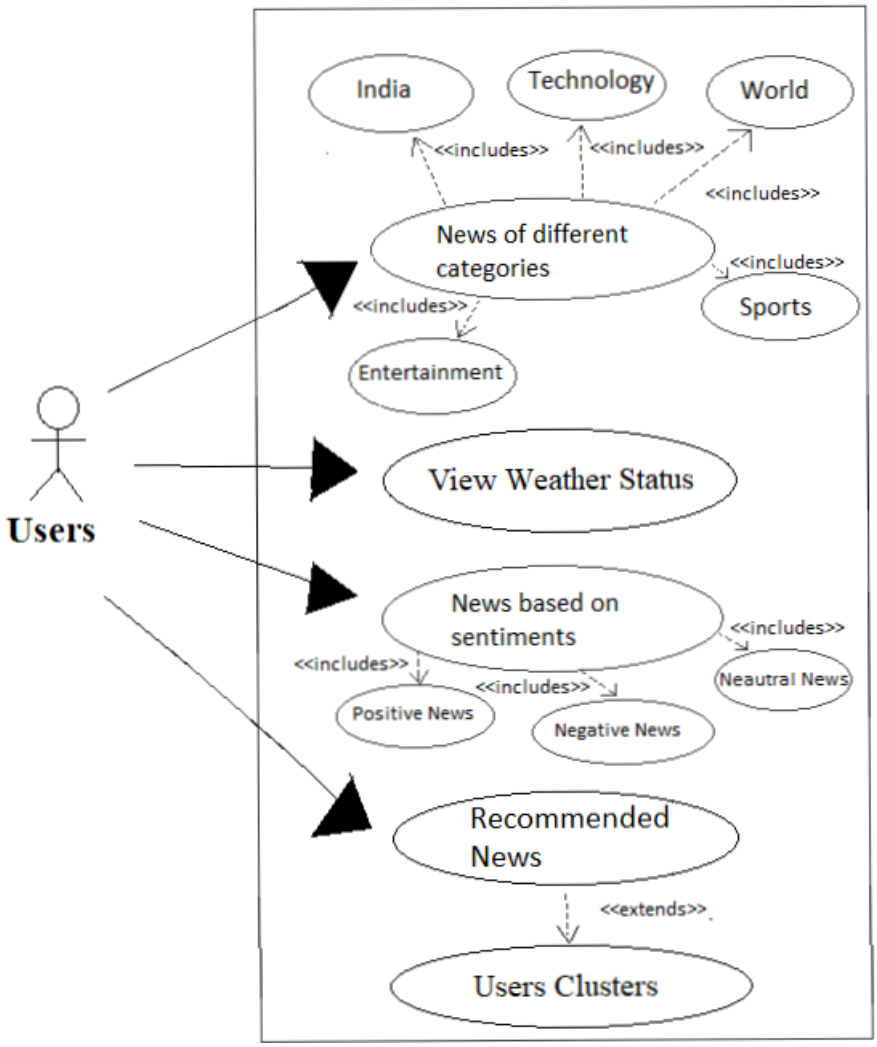

Figure 3.2 Use Case Diagram

3.3 Activity Diagram

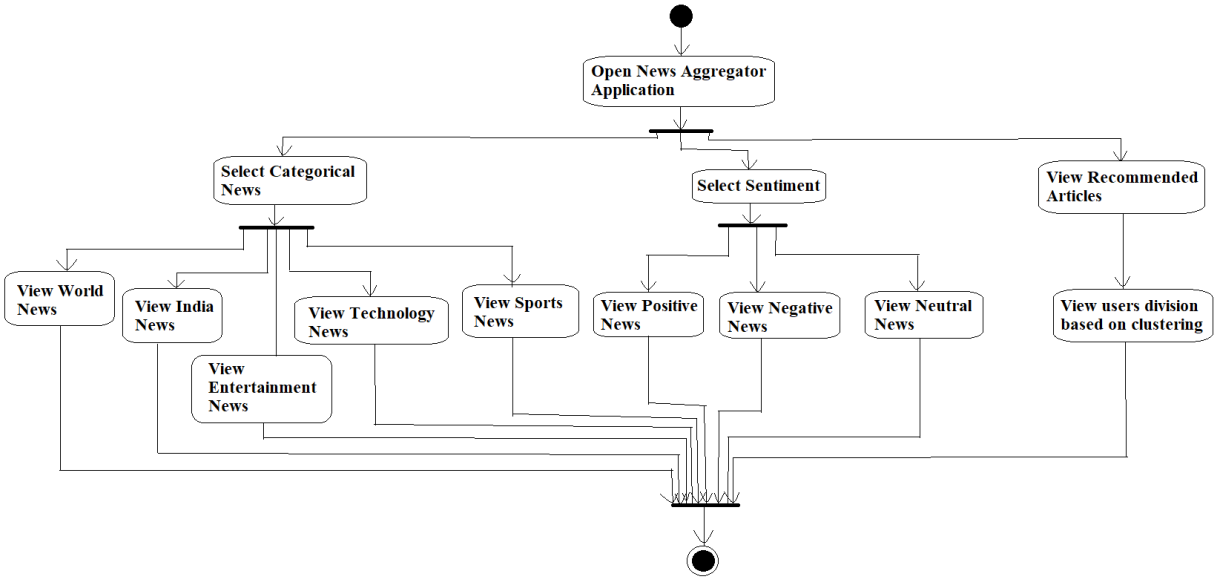

Figure 3.3 Activity Diagram

3.4 Sequence Diagram

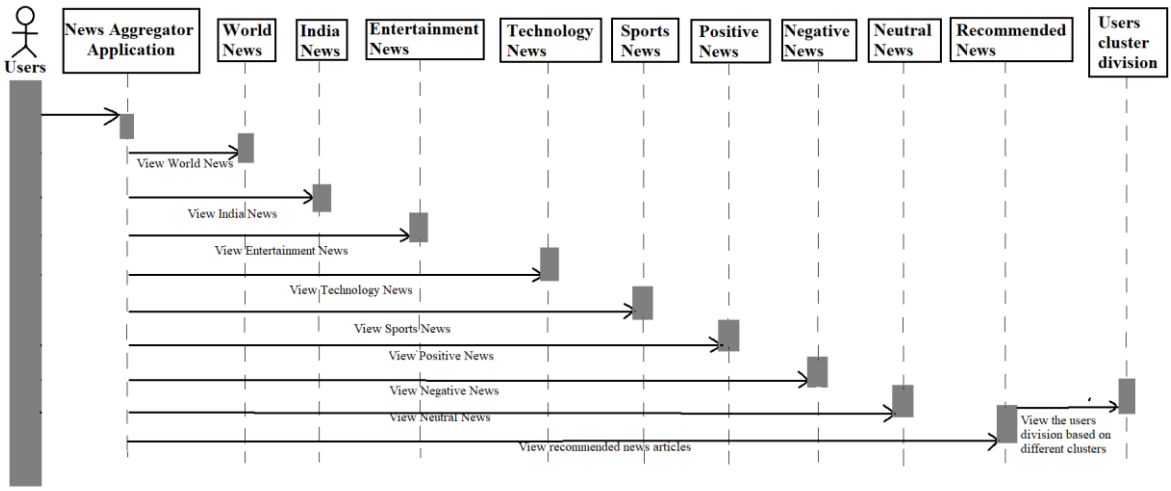

Figure 3.4 Sequence Diagram

3.5 Collaboration Diagram

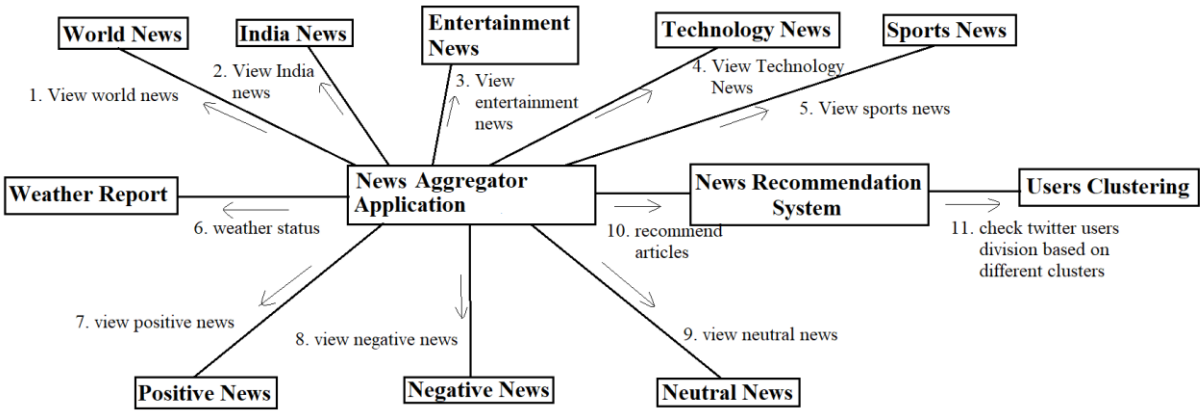

Figure 3.5 Collaboration Diagram

Design

4.1 Architectural Design

The software needs the architectural design to represents the design of software. The software that is built for computer-based systems can exhibit one of these many architectural styles. Each style will describe a system category that consists of:

- A set of components (eg: a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication, and cooperation between the components.
- Conditions that how components can be integrated to form the system.
- Semantic models that help the designer to understand the overall properties of the system.

The use of architectural styles is to establish a structure for all the components of the system.

4.1.1 Architectural Context Diagram

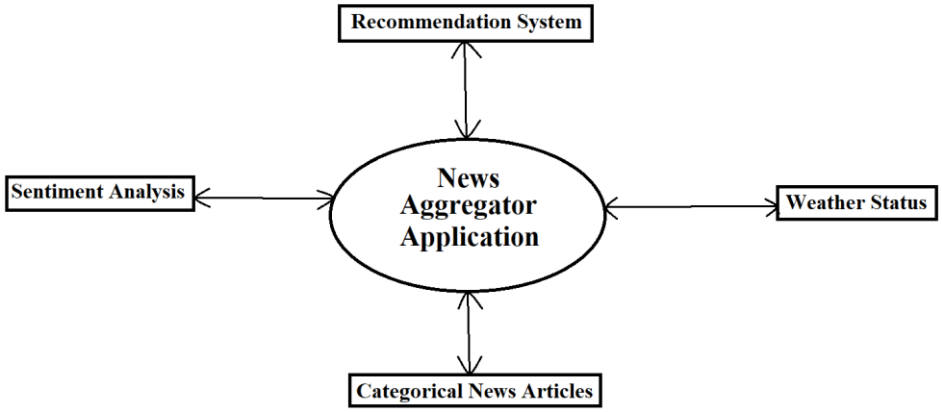

Figure 4.1 Architectural Context Diagram

4.1.2 Architectural Behavioral Diagram

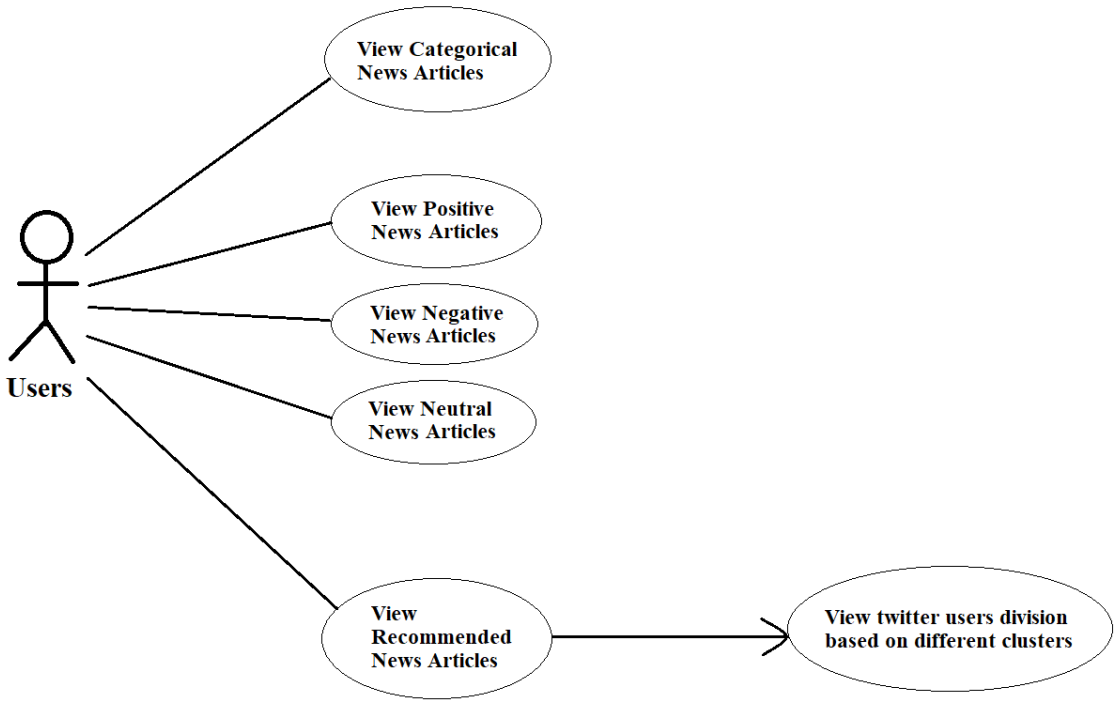

Figure 4.2 Architectural Behavioral Diagram

4.1.3 Description of Architectural Diagram

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap.

The architectural context diagram of our project depicts the system 'News Aggregator Application' and its flow of information towards its external entities. In the given diagram, News Aggregator Application shows the flow of information and has the features of Sentiment Analysis, Recommendation System, Weather Status and Categorical News Articles.

The architectural behavioral diagram of our project depicts the behavior of the project and its direction of flow of information. Users can view the categorical news articles i.e. world news, India news, entertainment news, technology news and sports news. Users can also view the news articles that are positive, negative or neutral and can check the popular recommended news articles. The recommended news articles can further display the twitter users division based on different clusters.

4.1.5 Control Hierarchy

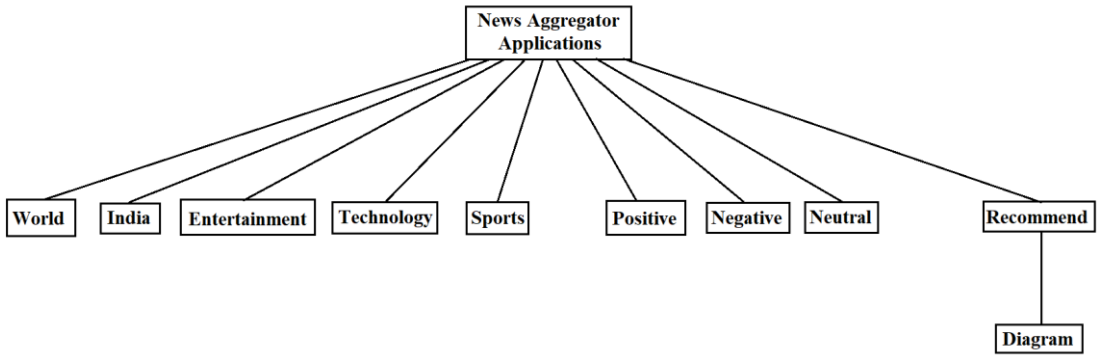

Figure 4.3 Control Hierarchy

Control hierarchy is also called as the program structure. It basically represents the hierarchy of control. It basically display the web pages designed for different news category such as world, India, entertainment, technology, sports, positive, negative neutral and recommended news articles. Recommended news articles further depicts the diagram of the users division.

4.2 Procedural/ Modular Approach

4.2.1 Modules Used

A module is a collection of source files and builds settings that allow you to divide your project into discrete units of functionality. Your project can have one or many modules and one module may use another module as a dependency. Each module can be independently built, tested, and debugged.

Our project News Aggregator Application consists of the following modules:

- Index
- World
- India
- Entertainment
- Technology
- Sports
- Positive
- Negative
- Neutral
- Recommend
- Diagram

4.2.2 Internal Data Structures

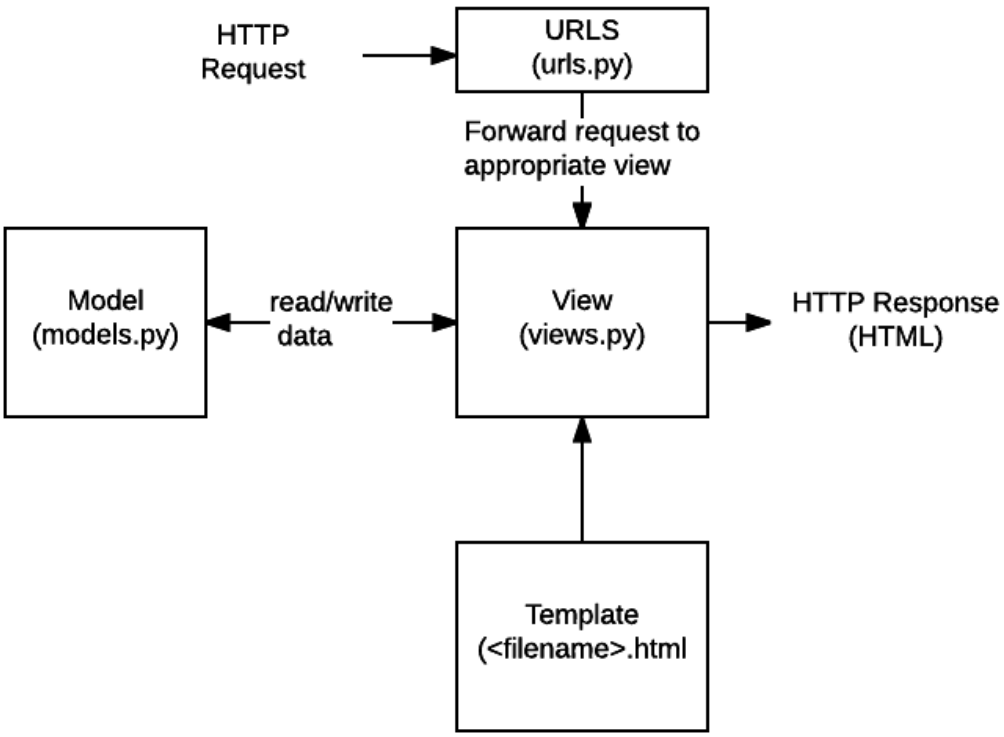

Figure 4.4 Internal Django working

- **URLs:** While it is possible to process requests from every single URL via a single function, it is much more maintainable to write a separate view function to handle each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data.
- **View:** A view is a request handler function, which receives HTTP requests and returns HTTP responses. Views access the data needed to satisfy requests via *models*, and delegate the formatting of the response to *templates*.

- **Models:** Models are Python objects that define the structure of an application's data, and provide mechanisms to manage (add, modify, delete) and query records in the database.

- **Templates:** A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A *view* can dynamically create an HTML page using an HTML template, populating it with data from a *model*. A template can be used to define the structure of any type of file; it doesn't have to be HTML!

4.3 Data Design

Data design is the first design activity, which results in fewer complexes, modular and efficient program structure. The information domain model developed during analysis phase is transformed into data structures needed for implementing the software. The data objects, attributes, and relationships depicted in entity relationship diagrams and the information stored in data dictionary provide a base for data design activity. During the data design process, data types are specified along with the integrity rules required for the data.

4.3.1 Data objects and Resultant Data Structures

The news.csv file is used to evaluate and analyse the top users and their tweets to produce a recommendation system. This news.csv file is generated by accessing the twitter API and requests the information from different news portals official IDs.

Other dataset such as sample text to perform sentiment analysis is downloaded directly during the compilation. It basically contains 5000 positive sample texts and 5000 negative sample texts.

4.4 Interface Design

Django is used to provide a connection between the front end and back end and deploy the web page. HTML, CSS, Javascript and bootstrap is used to develop the interface of the project. Python is used at the back end to generate a possible solution to complete the objective of our project. The following versions are used in our project:

- Python 3.8.6
- HTML5
- CSS 3
- Bootstrap4
- Django 3.1.3

To establish the connection between several web pages, we had provided the connection establishment link and their path in the Django. This further establishes a connection and run the project in its local server.

Implementation and Testing

5.1 Implementation

5.1.1 Implementation Model

A system must be implemented for its use. The system consists of the service layer for processing the business logic, and an interface. It is required to migrate the python code to the server that is intended to serve the application. The establishment of connection between the interface and the python code helps the project to work according to the requirements. Then, we can host the site; the system is ready to be used.

5.1.2 Experimental Setup

News aggregation systems rely on a large corpus of documents. Many large corpora are available for aggregation research. However, our interest here is in news aggregration. Since, the goal of our project work was to create a model for aggregating news articles. So, we have to rely on web documents in our experiments.

Corpus

The document corpus for carrying out our project study consisted of the news articles collected from three different news portals site.

Dataset

The dataset was constructed by retrieving the structured text from xml file which consisted of the heading, location, description and the content of the news. Twitter users information is also retrieved using twitter API to generate a csv file for the development of recommender system. Positive and negative sentences (5000 each) are also gathered to train our model for sentiment analysis.

5.1.3 Preprocessing

Generating API keys

We had generated API keys from openweathermap and twitter to obtain the relevant information which will be helpful in generating weather status report and recommender system. Openweathermap API helps the developers to get the information about user's weather status using their location or IP address. Twitter API helps the developer to get twitter users information, their tweets and their followers. It helped us to choose the top users of news official twitter handle of news websites and to gain knowledge about their interests.

Aggregating

A content aggregator website is a site that collects data from other sources across the internet and puts the information in one place where users can access it. The data collected can be based on many things, depending on the channel or platform it's pulling from...

- A certain hashtag
- A certain user account
- A keyphrase
- A channel
- A playlist

The process followed by the web aggregator is called Web aggregating. Many sites, in particular search engines, use aggregating as a means of providing up-to-date data. Web aggregators are mainly used to create a copy of all the visited pages for later processing by a search engine that will index the downloaded pages to provide fast searches. Aggregators can also be used for automating maintenance tasks on a Web site, such as checking links or validating HTML code. Also, aggregators can be used to gather specific types of information from Web pages, such as harvesting e-mail addresses (usually for sending spam).

The large volume implies that the aggregator can only download a limited number of the Web pages within a given time, so it needs to prioritize its' downloads. The high rate of change implies that the pages might have already been updated or even deleted.

The number of possible aggregatable URLs being generated by server-side software has also made it difficult for web aggregators to avoid retrieving duplicate content. Endless combinations of HTTP GET (URL-based) parameters exist, of which only a small selection will actually return unique content

Some potential issues:

- Links encountered during parsing may be relative paths.
- Normalization needed.
- Pages of a given web site may contain several duplicated links.
- Some links may point to robot-free areas.

HTML Parsing

An HTML parser analyzes HTML markup tags. The parser is the front-end for a markup analyzer, the entity designed for the identification of document structure. The markup analyzer contains the core logic for identifying different document parts and assigning levels of significance for them according to the scheme described earlier.

The parser also takes care of decoding encoded character entities, removing whitespaces, detecting block-level structures, inline structures, and stripping tags.

5.1.4 Tools Used

The tools used for the design and developments of the system are as follows:

A) Development Environment

- Python3: For business logic and interface design.
- HTML, CSS, JavaScript and Bootstrap 4: For validation, styling and presenting document.
- Django: For establishing interconnection and run the developed project in local server.

B) Development Tool

- Sublime Text: For text editing.
- Jupiter Notebook and Anaconda: To test the python code.

5.1.5 Program Code

5.1.5.1 Libraries used

```
from django.shortcuts import render
import requests
import nltk
from bs4 import BeautifulSoup
from newspaper import Article
from textblob import TextBlob
import geocoder
import json
import pandas as pd
import numpy as np
import time
#to scrape Twitter
import tweepy
```

```
from tweepy import OAuthHandler
import csv
import matplotlib.pyplot as plt
import io
import urllib,base64
import operator
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.corpus import twitter_samples, stopwords
from nltk.tag import pos_tag
from nltk.tokenize import word_tokenize
from nltk import FreqDist, classify, NaiveBayesClassifier
import re, string, random
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
from nltk.tokenize import RegexpTokenizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.cluster import KMeans
from sklearn.manifold import MDS
import os
import warnings
warnings.filterwarnings('ignore')
```

5.1.5.2 Weather Status

```
g = geocoder.ip('me')
lat=str(g.latlng[0])
lon=str(g.latlng[1])
city=str(g[0])
city=city[1:-3]+"India"
api_key = "96d2e493c472b35ec2a8976bdb6d83bd"
base_url = "http://api.openweathermap.org/data/2.5/weather?"
```

```
complete_url = base_url + "appid=" + api_key + "&lat=" + lat + "&lon=" + lon
response = requests.get(complete_url)
x = response.json()
if x["cod"] != "404":
        y = x["main"]
        current_temperature = y["temp"]
        current_pressure = y["pressure"]
        current_humidiy = y["humidity"]
        z = x["weather"]
        weather_description = z[0]["description"]
        w_temp=str(current_temperature)
        w_press=str(current_pressure)
        w_hum=str(current_humidiy)
        w_desc=str(weather_description.title())
```

5.1.5.3 Sentiment Analysis

```
def remove_noise(tweet_tokens, stop_words = ()):
        cleaned_tokens = []
        for token, tag in pos_tag(tweet_tokens):
                token = re.sub('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+#]|[!*\(\),]|'\
                        '(?:%[0-9a-fA-F][0-9a-fA-F]))+','', token)
                token = re.sub("(@[A-Za-z0-9_]+)","", token)
                if tag.startswith("NN"):
                        pos = 'n'
                elif tag.startswith('VB'):
                        pos = 'v'
                else:
                        pos = 'a'
                lemmatizer = WordNetLemmatizer()
                token = lemmatizer.lemmatize(token, pos)
```

```
            if len(token) > 0 and token not in string.punctuation and token.lower() not in
            stop_words:
                cleaned_tokens.append(token.lower())
                return cleaned_tokens
def get_all_words(cleaned_tokens_list):
    for tokens in cleaned_tokens_list:
        for token in tokens:
            yield token
def get_tweets_for_model(cleaned_tokens_list):
    for tweet_tokens in cleaned_tokens_list:
        yield dict([token, True] for token in tweet_tokens)

positive_tweets = twitter_samples.strings('positive_tweets.json')
negative_tweets = twitter_samples.strings('negative_tweets.json')
text = twitter_samples.strings('tweets.20150430-223406.json')
tweet_tokens = twitter_samples.tokenized('positive_tweets.json')[0]
stop_words = stopwords.words('english')
positive_tweet_tokens = twitter_samples.tokenized('positive_tweets.json')
negative_tweet_tokens = twitter_samples.tokenized('negative_tweets.json')
positive_cleaned_tokens_list = []
negative_cleaned_tokens_list = []
for tokens in positive_tweet_tokens:
    positive_cleaned_tokens_list.append(remove_noise(tokens, stop_words))
for tokens in negative_tweet_tokens:
    negative_cleaned_tokens_list.append(remove_noise(tokens, stop_words))
all_pos_words = get_all_words(positive_cleaned_tokens_list)
freq_dist_pos = FreqDist(all_pos_words)
positive_tokens_for_model = get_tweets_for_model(positive_cleaned_tokens_list)
negative_tokens_for_model = get_tweets_for_model(negative_cleaned_tokens_list)
positive_dataset = [(tweet_dict, "Positive") for tweet_dict in positive_tokens_for_model]
negative_dataset = [(tweet_dict, "Negative") for tweet_dict in negative_tokens_for_model]
dataset = positive_dataset + negative_dataset
```

```
random.shuffle(dataset)
train_data = dataset[:7000]
test_data = dataset[7000:]
classifier = NaiveBayesClassifier.train(train_data)
def sentiment(txt):
        tokens = remove_noise(word_tokenize(txt))
        p=str(classifier.classify(dict([token, True] for token in tokens)))
        if p=="Negative":
                p=-1
                return p
        elif p=="Positive":
                p=1
                return p
```

5.1.5.4 Web Scraping the news articles

```
pos=[]
pos_link=[]
pos_img=[]
pos_art=[]
neg=[]
neg_link=[]
neg_img=[]
neg_art=[]
neu=[]
neu_link=[]
neu_img=[]
neu_art=[]
headline=[]
web="https://www.indiatoday.in"
web1="https://timesofindia.indiatimes.com/briefs"
it_r = requests.get("https://www.indiatoday.in/world")
it_soup = BeautifulSoup(it_r.content, 'html5lib')
```

```
it_headings  = it_soup.find_all('h2')
it_headings  = it_headings[0:4]
it=[]
for hm in it_headings:
        it.append(hm.text)
        headline.append(hm.text)
link=[]
for l in range(0,len(it_headings)):
        txt=str(it_headings[l])
        for i in range(0, len(txt)):
        if(txt[i]=="a"  and txt[i+1]==" " and txt[i+2]=="h"  and txt[i+3]=="r"  and txt[i+4]=="e"
        and txt[i+5]=="f"  and txt[i+6]=="="):
                new=""
                for j in range(i+8,len(txt)):
                        if(txt[j]==""):
                        break
                new=new+txt[j]
                j=j+1
        link.append(web+new)
art=[]
img=[]
for i in range(0,len(link)):
        article=Article(link[i])
        article.download()
        article.parse()
        nltk.download('punkt')
        article.nlp()
        img.append(article.top_image)
        art.append(article.summary)
for i in range(0,len(it)):
        edu=TextBlob(it[i])
        x=edu.sentiment.polarity
```

```
        z=sentiment(it[i])
        if x==0:
                neu.append(it[i])
                neu_link.append(link[i])
                neu_img.append(img[i])
                neu_art.append(art[i])
                continue
        if z==1:
                pos.append(it[i])
                pos_link.append(link[i])
                pos_img.append(img[i])
                pos_art.append(art[i])
        elif z==-1:
                neg.append(it[i])
                neg_link.append(link[i])
                neg_img.append(img[i])
                neg_art.append(art[i])
it=tuple(zip(it,link,img,art))
```

5.1.5.5 News Recommender System

```
consumer_key = 'EOxjQUZBztiTRBQRG4SewSp0l'
consumer_secret = 'jo4h3qW9DSwIlGpAZ12pP0xHDa0qePPkKX2uswbpa6xrAQoxAj'
access_token = '1348835624-apgxylphWsCprKdGwG6DFhp0tViFpvet1nEfxua'
access_secret = '9h7BFke7lfI1hLHMv1Rv2NE69NEXaTwy9DBr11wIWZJxa'
auth = OAuthHandler(consumer_key,  consumer_secret)
auth.set_access_token(access_token, access_secret)
api = tweepy.API(auth)
usersData=pd.read_csv('News.csv')
usersData.drop_duplicates(inplace  = True)
usersData = usersData[(usersData.totalTweets > 10) & (usersData.Popularity > 1)]
usersData = usersData.reset_index(drop=True)
def getTweets(user):
```

```
        twitterUser = api.get_user(user)
        tweets = api.user_timeline(screen_name  = user, count = 10,tweet_mode='extended')
        tentweets = []
        for tweet in tweets:
                if tweet.full_text.startswith("RT  @") == True:
                        tentweets.append(tweet.retweeted_status.full_text)
                else:
                        tentweets.append(tweet.full_text)
        return tentweets
vfunc = np.vectorize(getTweets)
usersData["tweets"] = usersData['ActiveNewsReaders'].apply(lambda x: getTweets(x))
ps = nltk.PorterStemmer()
wn = nltk.WordNetLemmatizer()
stop = set(stopwords.words('english'))
exclude = set(string.punctuation)
words = set(nltk.corpus.words.words())
def processing(tweets):
        cleanedTweets = []
        for tweet in tweets:
                tw = re.sub('http\S+', '', tweet)
                tw = re.sub('RT', '', tw)
                tw = re.sub('@[^\s]+','',tw)
                tw = "".join([char  for char in tw if char not in string.punctuation])
                tw = tw.lower()
                tw = ' '.join([word  for word in tw.split() if word not in (stop)])
                tw = ' '.join([word  for word in tw.split() if len(word)>2])
                cleanedTweets.append(tw)
        cleanedTweets = ' '.join(cleanedTweets)
        ProcessedTweets = nltk.word_tokenize(cleanedTweets)
        ProcessedTweets = [ps.stem(word) for word in ProcessedTweets]
        ProcessedTweets = [wn.lemmatize(word)  for word in ProcessedTweets]
        ProcessedTweets = [word for word in ProcessedTweets if len(word)>2]
```

```
        ProcessedTweets = ' '.join(w  for w in ProcessedTweets if w in words)
        return ProcessedTweets
usersData["ptweets"] = usersData['tweets'].apply(lambda  x : processing(x))
tfidf_vectorizer  = TfidfVectorizer(max_df=0.9,  max_features=200000,min_df=0.1,
use_idf=True)
tfidf_matrix  = tfidf_vectorizer.fit_transform(usersData.ptweets)
tfidf_matrix.toarray()
pd.DataFrame(tfidf_matrix.toarray(),  columns=tfidf_vectorizer.get_feature_names(),  index =
usersData.ActiveNewsReaders)
num_clusters  = 5
km = KMeans(n_clusters=num_clusters,  init='k-means++',  max_iter=100,  n_init=1)
km.fit(tfidf_matrix)
clusters = km.labels_.tolist()
km.cluster_centers_.argsort()[:, ::-1]
top_keywords=[]
order_centroids  = km.cluster_centers_.argsort()[:, ::-1]
terms = tfidf_vectorizer.get_feature_names()
for i in range(num_clusters):
        for ind in order_centroids[i, :10]:
                top_keywords.append(terms[ind])
def processArticles(articles):
        cleanedarticles  = []
        for article in articles:
                article = re.sub("[^a-zA-Z]","  ", str(article))
                article = article.lower()
                article = ' '.join([word  for word in article.split()  if word not in (stop)])
                article = ' '.join([word  for word in article.split()  if len(word)>2])
        #tokenization
        article = nltk.word_tokenize(article)
        #stemming
        article = [ps.stem(word) for word in article]
        #lammitization
```

```
        article = [wn.lemmatize(word)  for word in article]
        article = [word for word in article if len(word)>2]
        article = ' '.join(w  for w in article  if w in words)
        cleanedarticles.append(article)
        return cleanedarticles
headlines = processArticles(headline)
top_keywords = list(dict.fromkeys(top_keywords))
recommended_headlines=[]
for xl in range(0,len(headline)):
        splitting=str(headlines[xl]).split()
        for bq in range(0,len(splitting)):
                for resq in top_keywords:
                        if(splitting[bq]==resq):
                                recommended_headlines.append(headline[xl])
def countele(lst,  x):
        count = 0
        for ele in lst:
                if (ele == x):
                        count = count + 1
        return count
counting=[]
for xl in range(0,len(recommended_headlines)):
        counting.append(countele(recommended_headlines,recommended_headlines[xl]))
head_count = dict(zip(recommended_headlines,counting))
head_count = dict(sorted(head_count.items(),  key=operator.itemgetter(1),reverse=True))
recommended_headlines  = []
for k in head_count.keys():
        recommended_headlines.append(k)
recommended_headlines=recommended_headlines[0:6]
```

5.2 Testing

5.2.1 Testing Objective

The primary objective of the testing is to ensure software reliability, software quality and system assurance of the project. It helps us to gain knowledge about error and bugs that may occur during the implementation of the project.

The optimum performance and capacity utilization is decided based on its working condition. It helps in the prevention of errors and adds value to the product by ensuring conformity to client requirements.

5.2.2 Testing Scope

The scope of a test defines what areas of a customer's product are supposed to get tested, what functionalities to focus on, what bug types the customer is interested in, and what areas or features should not be tested by any means.

Following are the scope of the testing for this project:

- The proper working and link connection of the project on the developer's system.
- No error is generated during the execution of the project.
- Check the weather status and compare it with current report.
- Check the working of the hyperlinks provided in the news articles. If there is any broken link, work on correcting it.
- Check the execution time of the project and ensure it to be optimal.
- Ensure relevant output generation on implementing the sentiment analysis and recommender system.

5.2.3 Testing Principles

For testing an application or software, we need to follow some principles to make our product defects free, and that also helps the test engineers to test the software with their effort and time.

Following testing principles are defined to ensure smooth functionality:

- **Efficient test management for effective results**

  → Test management helps in developing a robust risk management process. Effective test strategy and planning is needed for developing effective applications.

- **Cross platform Test environment to avoid uncertainties**

  → It would be expensive to develop a test environment for news aggregator site, since it includes multiple platforms. Cross-platform testing is important for news aggregator system in this respect. In news aggregator sites, cross-platform development also includes cross-browser adaptability. Users should be able to use the site concurrently and securely on multiple browsers and platforms.

- **End User focused user acceptance testing to ensure adoption**

  → At the end of the development process, it is vital to pass the user acceptance assessment. For continued performance, there should be security testing performed to make sure that the user is securely able to perform actions on the website. Usability and experience are two of the most important aspects affecting user behavior and hence, user acceptance testing is a must for news aggregator systems.

- **Regression Testing for predictability and robustness**

  → User expectations are always increasing and they are now spending more time searching for news articles of their interests. New feature enhancements with frequent releases to match user expectations have boosted user experience to another level. Regression testing helps in ensuring that faults are uncovered with the number of changes. It helps drastically when one can ensure that the functionality would not change, even after there are continuous changes in environment.

5.2.4 Testing Methods Used

Unit Testing

Unit Testing concentrates on each unit of software as implemented in the source code. It only tests the functionality of the units themselves.

We have used different test cases in this project for unit testing. We used unit testing to test:

- Existence of classes
- Existence of methods
- Exception thrown properly or not
- Actual implementation works or not.

5.2.5 Test Cases

Unit Testing

Table 5.1 Unit Testing

Test Case ID	Test Case Objective	Prerequisite	Expected Output	Actual Output	Status
T1	Check web pages and their connections	Establish connections between different web pages with Django	All web pages must display without any error.	All web pages must display without any error.	Pass
T2	Test Weather Status	API key is to be extracted	Temperature→ 302.48 Kelvin Atmospheric Pressure→ 1016 hPa Humidity→ 14% Description→ Clear Sky Location→ Ajmer, Rajasthan, India	Temperature→ 302.48 Kelvin Atmospheric Pressure→ 1016 hPa Humidity→ 14% Description→ Clear Sky Location→ Ajmer, Rajasthan, India	Pass
T3	Check news articles and their information displayed	Web scraping must be performed	All news articles headlines, their image and text must be displayed without any error.	All news articles headlines, their image and text must be displayed without any error.	Pass

Sentiment Analysis Testing

Table 5.2 Sentiment Analysis Testing

S. No	Test Case ID	Algorithm used	Expected Output	Actual Output	Status
1.	Input: "COVID vaccine excitement builds as Moderna reports third positive result."				
	T4	Naïve Bayes	Positive	Positive	Pass
	T5	Sentiment.polarity	Positive	Positive	Pass
2.	Input: "Dozens to be deliberately infected with coronavirus in UK 'human challenge' trials"				
	T6	Naïve Bayes	Negative	Negative	Pass
	T7	Sentiment.polarity	Negative	None	Fail
3.	Input: "Memory fail was predicted by attention lapsing and media multitasking."				
	T8	Naïve Bayes	Negative	Negative	Pass
	T9	Sentiment.polarity	Negative	Negative	Pass
4.	Input: "Team of scientists from Coimbatore bags first prize at National Water Awards"				
	T10	Naïve Bayes	Positive	Positive	Pass
	T11	Sentiment.polarity	Positive	Positive	Pass

5.2.6 Output

A) Home Page

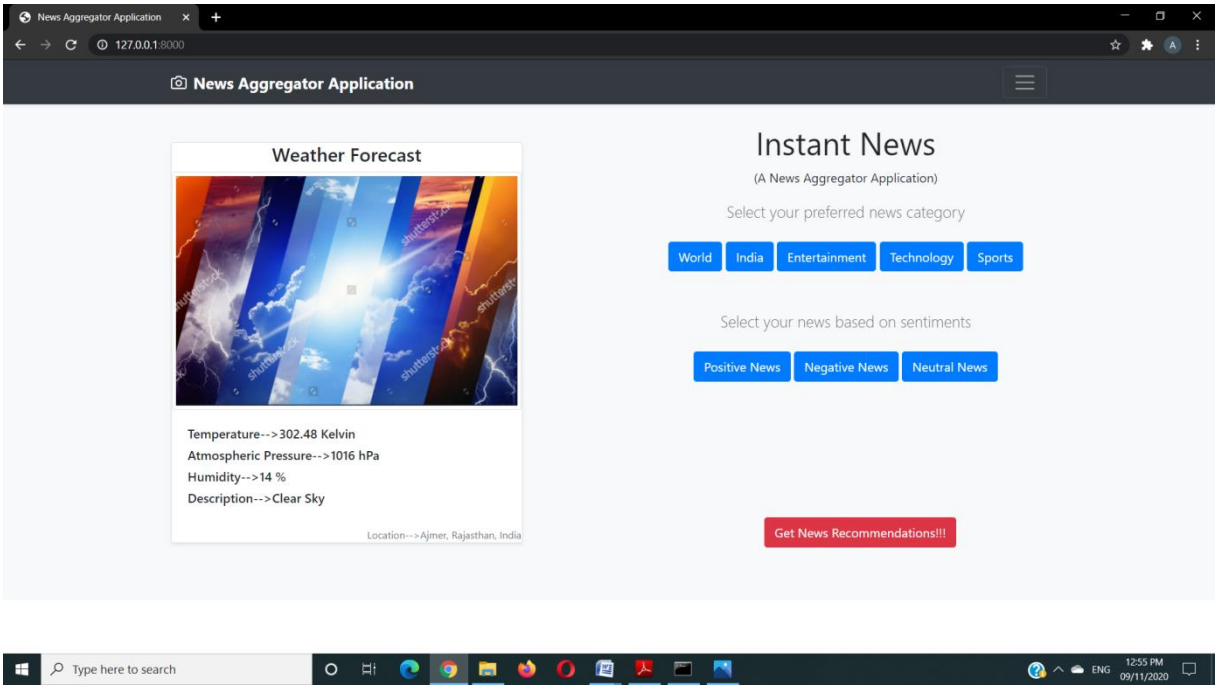

Figure 5.1 Home Page

B) World News

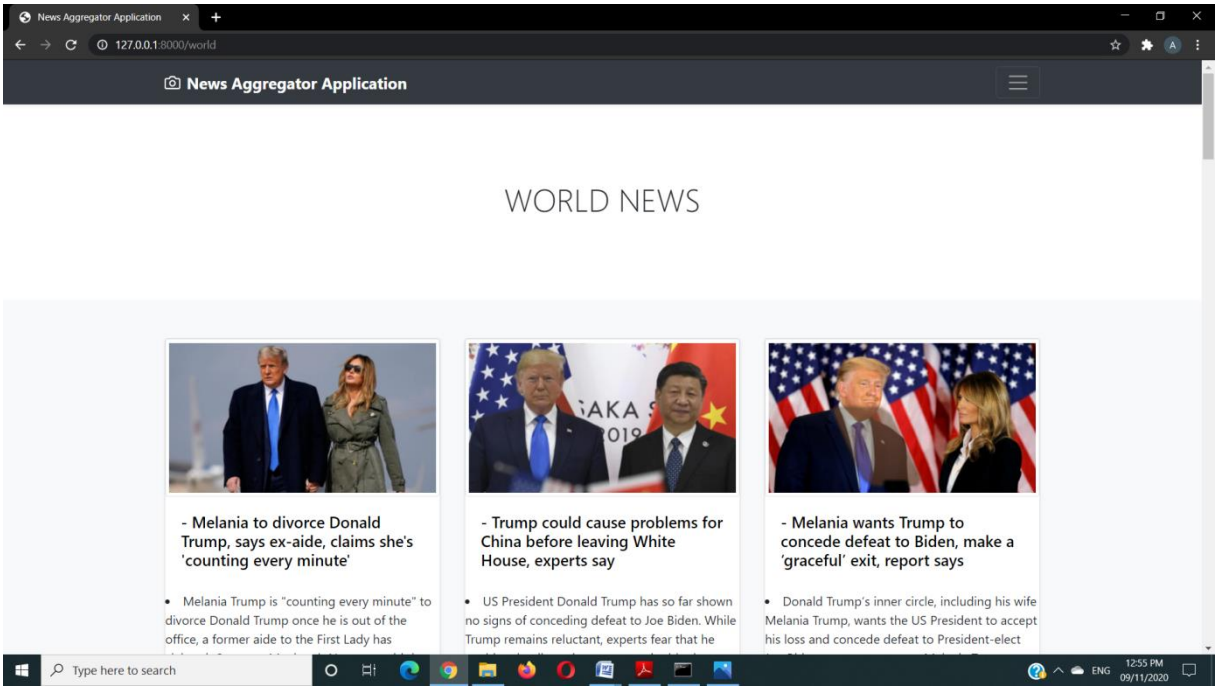

Figure 5.2 World News

C) Sports News

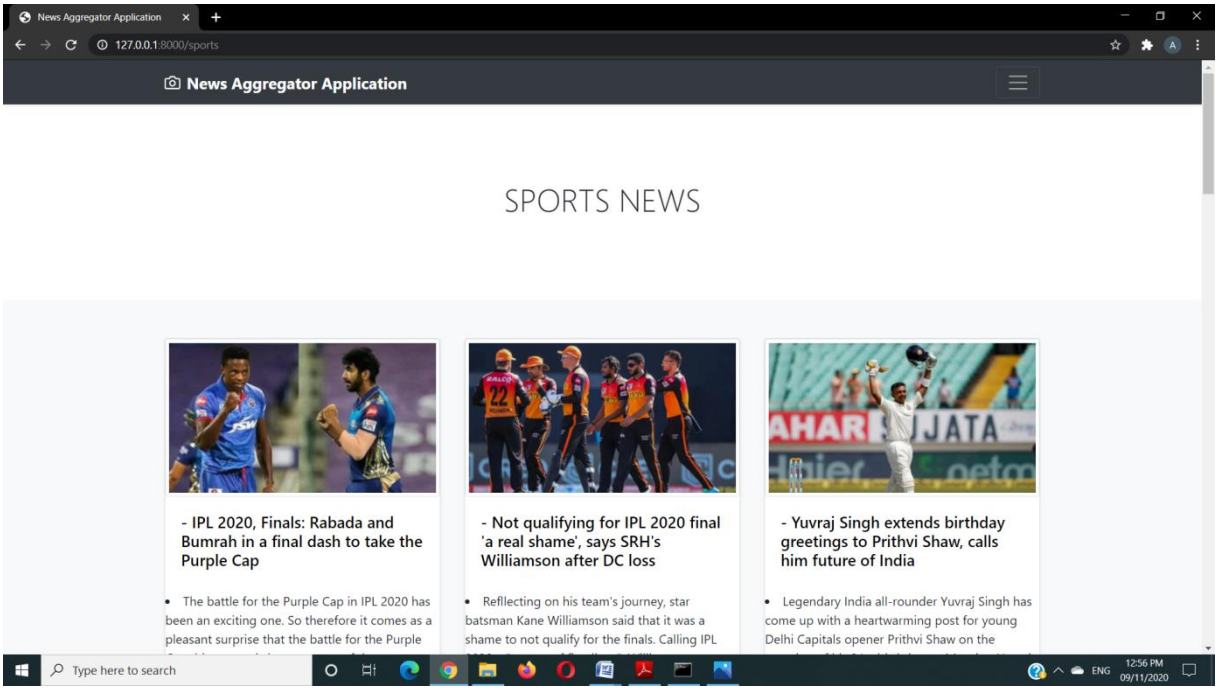

Figure 5.3 Sports News

D) Positive News

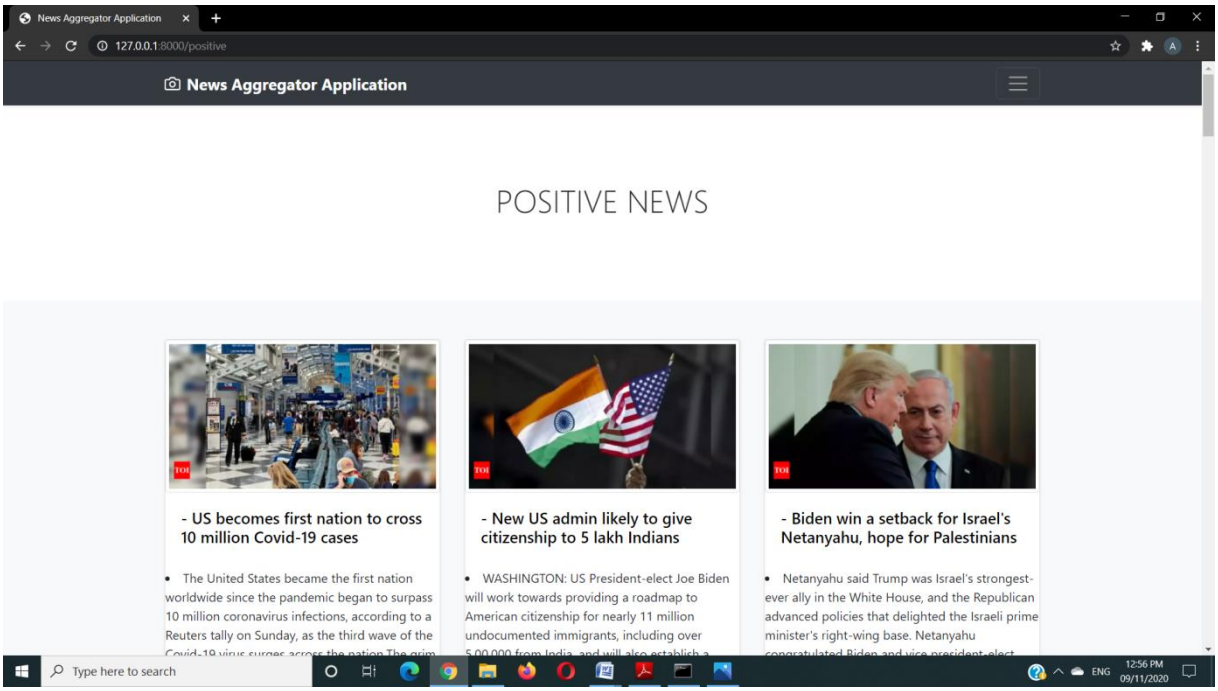

Figure 5.4 Positive News

D) Recommended News Headlines

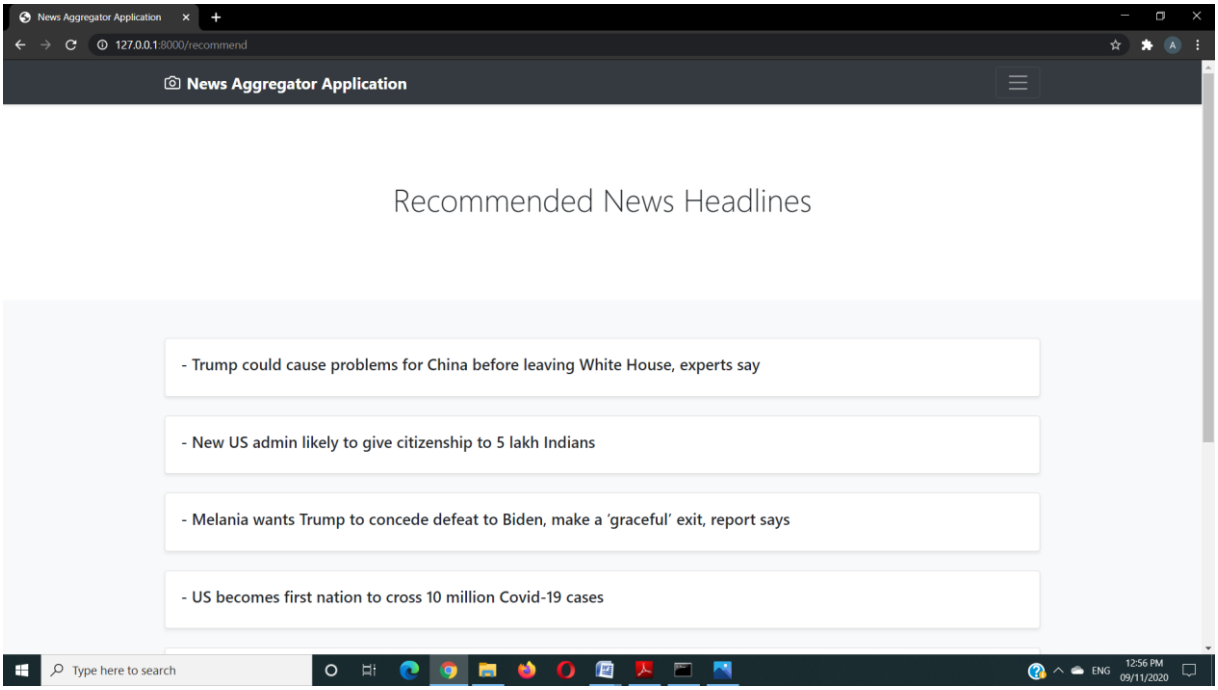

Figure 5.5 Recommended News Headlines

E) Twitter users division based on different clusters

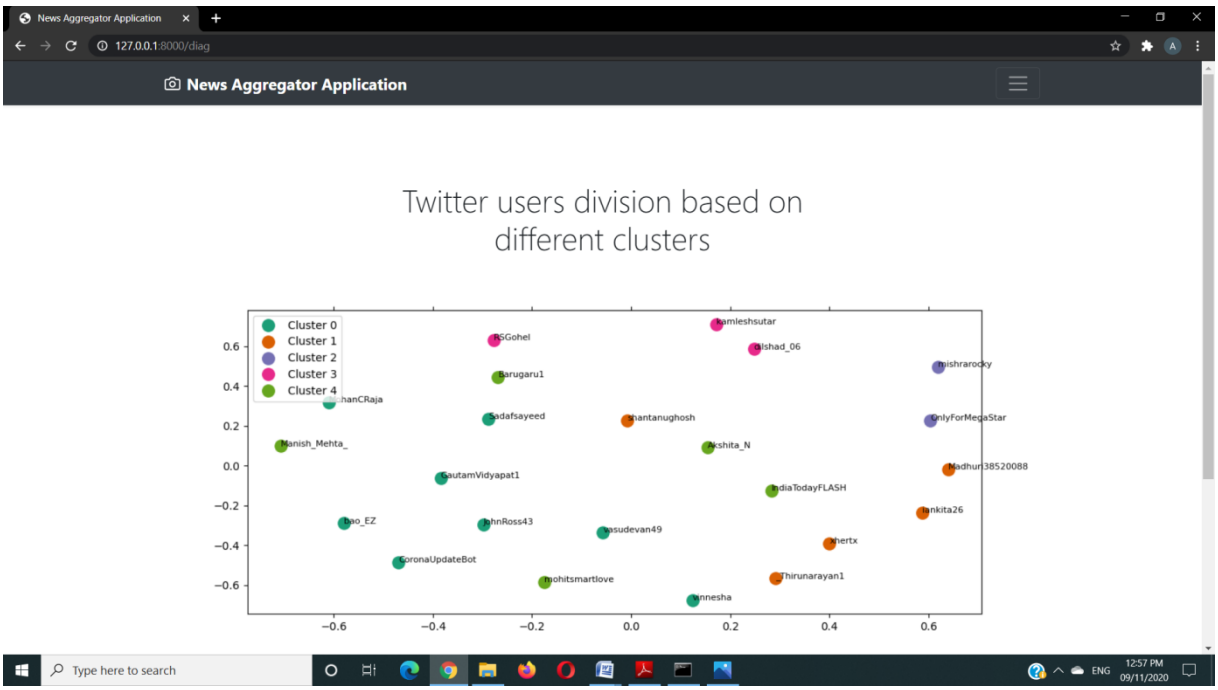

Figure 5.6 Twitter users division based on different clusters

5.2.7 Results

All the output generated matched with expected output. All the test cases have been passed. The project is compatible and can be accessed in multiple browsers such as Google Chrome, Mozilla Firefox etc. All the modules had been integrated with each other to ensure the proper flow of the project.

The news articles are scraped from different news portals and are merged in single space to divide them into different categories. These news articles headlines and their display image are shown in the final output.

Sentiment analysis is performed using Naives Bayes classifier which uses 5000 positive statements and 5000 negative statements to train the model. Sentiment Analysis generated the output with 99.5% accuracy. Using sentiment.polarity from Textblob, accuracy of the output positive news turns out to be 98.11% while negative news turns out to be 94.78%.

News Recommendation system is developed using the technique of popularity based recommendation system. Most popular twitter users' interests are gathered and based on their interests, recommendation system is generated.

Limitations

Our project news aggregator application consists of following limitations:

- We had used only three news websites (India Today, Indian Express and Times of India) as the data source of our project. To develop a news aggregator site and deploy on Internet, it requires many news portals as their data source which helps them to generate a good quantity of news articles.

- It is a desktop based web application. Mobile users can access this page only on desktop-mode in their web browsers.

- On developing this application in large scale, it will require high computation speed as it aggregates news information from different sources. Cost will increase while building it on large scale.

- It does not verify the news articles. If any wrong news is provided in it, then the particular website is solely responsible.

- News aggregator sites mainly works automatically but if the source website changes their webpage or its interface, then human intervention will be required.

Future Scope

Future scopes of our project news aggregator application are as follows:

- Fake news can be detected using Machine Learning in the news aggregator sites.
- Different languages other than English can also be included in this project. Natural Language Processing will be required to produce this feature.
- Login system can be included to produce different sessions for different users. This may help to develop hybrid recommender system.
- Artificial Intelligence driven unbiased news portal can be developed to generate neutral news which doesn't show any bias.
- Other medium such as audio and video view functionality can also be included in the news aggregator sites.

Conclusion

After having detail study of News Aggregator Application one can see a great change in the behavior of users or readers in many manners like their attitude, reading pattern. Users want to read information or articles in less span of time.

The central concept of application is to allow the readers to read most of the information in a single source and in a concise manner. The end user of this application is the readers who obtain the information from this portal. The users can choose the articles from different categories as well as different sentiments. Recommended news articles will also be displayed based on the most popular choice. Weather status is displayed using the user's IP address.

The news aggregator application is having lots of scope in future world. The time is most important factor in an individual life. It goes without saying that the discovery of content has changed the structure of the internet, and the market has warmed up gradually. Without content, the internet is a world with no meaning. If you don't find content on a site, then gradually you will shift to somewhere which will generate the right kind of material for you.

Finding the best content aggregators on the internet can be hard, and it entirely depends on the individual's choice and how they perceive it. News aggregator application helps the users to choose the articles based on his/her interests and saves a lot of time.

Bibliography and References

- Ten best News Aggregators of 2020
  https://www.lifewire.com/best-news-aggregators-4584410
- Naive Bayes Classifier
  https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
- Openweathermap API
  https://openweathermap.org/api
- Twitter API
  https://developer.twitter.com/en/docs/twitter-api
- Recommendation Systems
  https://www.analyticssteps.com/blogs/what-are-recommendation-systems-machine-learning
- Sentiment analysis in python
  https://towardsdatascience.com/a-beginners-guide-to-sentiment-analysis-in-python-95e354ea84f6
- Stemming and Lemmatization in Python
  https://www.datacamp.com/community/tutorials/stemming-lemmatization-python
- NLP in Python
  https://towardsdatascience.com/how-to-use-nlp-in-python-a-practical-step-by-step-example-bd82ca2d2e1e