# IBM - NEWS TRACKER APPLICATION

## PROJECT REPORT


***Submitted by***

**PRANAVA RAMAN B M S (2019103555)**

**SACHIN RAGHUL T (2019103573)**

**SANJEEV K M (2019103576)**

**SRIVATSAV R (2019103066)**


**TEAM ID : PNT2022TMID35300**


**ANNA UNIVERSITY :: CHENNAI**

# TABLE OF FIGURES

**9. ADVANTAGES & DISADVANTAGES**

**10.CONCLUSION**

**11.FUTURE SCOPE**

**12.APPENDIX**

**Source Code**

**GitHub Link**

# 1.INTODUCTION

## 1.1 Project Overview

As our lives are very busy these days, we often feel we need more than 24 hrs. a day to cope up with everything we have in our schedule. Well, that's not possible but reducing the time by changing the conventional method of reading news can help. Just tell us what market news you're interested in and get a quick peek for the day. Only read what you feel is relevant and save your time. This app helps you to query for all information about Indices, Commodities, Currencies, Future Rates, Bonds, etc.

## 1.2 Purpose

Newspapers are one of the most popular and most needed commodities in our daily life. In today's busy world, reading newspapers has become one of the traditional ways of getting news. News is produced every minute and distributed via television, radio and the Internet, so the news updated the next morning is already outdated. So newspaper and magazine publishers have a hard time keeping up with the pace. Change is needed and publishers must embrace mobile

# 3. LITRATURE SURVEY

## 2.1 Existing problem

We've collected all the relevant negative factors of lack of news tracking statistics, along with results from studies and reports that have analysed the issue. This guide will give you an unbiased look at why the media reports negative news. We'll provide you with an informed and educated overview of the subject in general.

## 2.2 Reference

| S No. | Paper Title | Author (s) | Month / Year | Methods / Implementation Techniques | Resource Link |
|---|---|---|---|---|---|
| 1. | News Keyword Extraction for Topic Tracking | Sungjick Lee, Han-joon Kim | Sept. 2008 | Keyword extraction technique is used to extract main features in studies such as information retrieval, text categorization, topic detection, and document summarization. To extract keywords, TF-IDF (Term Frequency-Inverse Document Frequency) weighting model has been widely used. | https://ieeexplore.ieee.org/document/4624203 |
| 2. | Breaking News Detection and Tracking in Twitter | Swit Phuvipadawat, Tsuyoshi Murata | Mar. 2010 | The breaking news can be categorized by a method to collect, group, rank and track breaking news from Twitter. To improve the similarity comparison for short-length messages, an emphasis is put on proper nouns. | https://ieeexplore.ieee.org/abstract/document/5616930 |

| | | | | | |
|---|---|---|---|---|---|
| | | | | Reliability,popularity and freshness for the ranking factors are used. | |
| 3. | Learning approaches for detecting and tracking news events | Yiming Yang, Jaime Q. Carbonell, Ralf D. Brown | June 1999 | Extending Supervised Learning and Unsupervised Clustering Algorithms to allow document classification based on content and temporal aspects of news events. | https://ieeexplore.ieee.org/abstract/document/784083 |
| 4. | Using Cloud Computing Capabilities – On the example of implementing a news application. | Olga Miknovich, Oksana Golubeva | 2019 | The possibilities of cloud computing technologies are considered on the example of the application implementation, which is a function that receives a news feed through the NewsApi service. The cloud computing model FaaS (Function as a Service), the Microsoft Azure cloud platform and the Azure Functions solution are used for implementation. | https://elib.psu.by/bitstream/123456789/31517/1/160-163.pdf |

## 2.3 Problem Statement Definition

The majority of people rarely read the news until something really significant occurs in their area of interest or around the world. It is possible to get the information you need through traditional newspapers and news sources, but it takes a lot of time and is not practical everywhere. Users of the News Tracker Application may quickly scan news stories that are tailored to their interests.

| | |
|---|---|
| **Who does the problem affect?** | People who are employed, students and anyone else who are generally busy and don’t have time to keep up with the daily news |
| **What are the boundaries of the problem?** | News recommendations are not tailored to each user's interests. |
| **What is the issue?** | People don't follow the news since it takes too long and can't keep their interest |
| **When does this issue occur?** | When the news is overrun with intricate and pointless information regarding the occurrences. |
| **Where is the issue occurring?** | In print and television, as well as other traditional media |
| **Why is it important that we fix the problem?** | People could catch up on everyday events without spending a lot of time if this issue could be fixed. |

- Raj, an astrophysicist who also enjoys Cricket, is unable to watch the whole game or even the highlights due to time constraints. He will be able to follow along easily if the news is presented to him in a condensed and ordered manner.

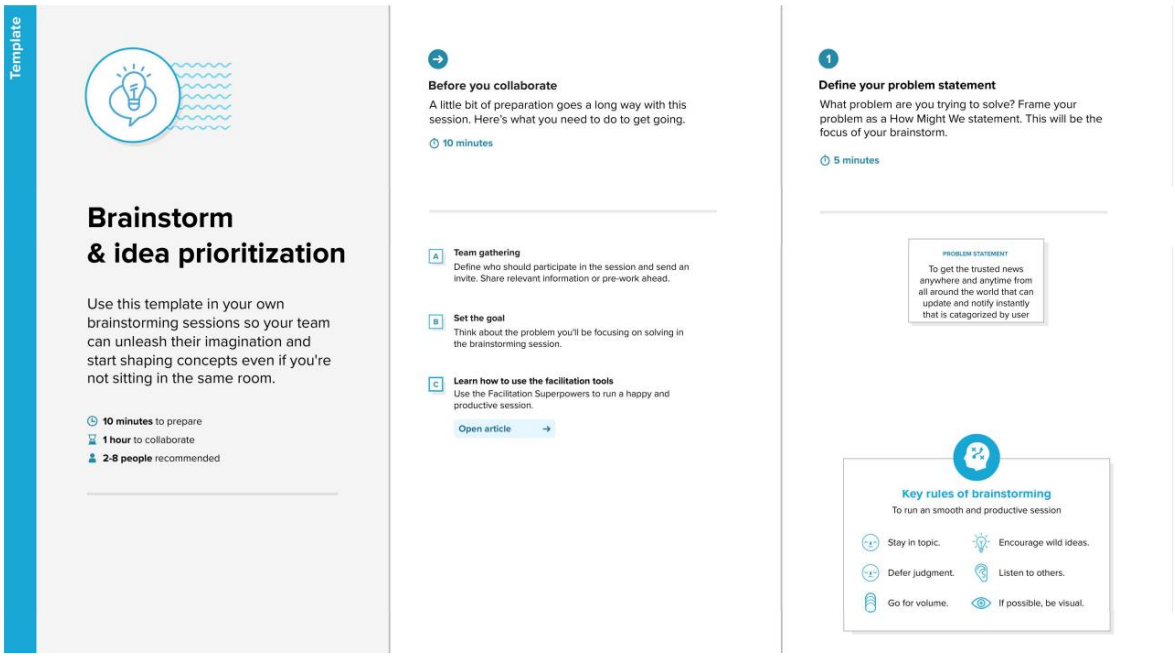
- Dwight, an assistant manager at a paper company, is a determined and hardworking employee with a very tight schedule and doesn’t have free time to read newspaper to catch up with the happenings at Scranton that his co-workers are usually talking about in the coffee breaks.

- Popular NYC chef Monica enjoys gathering newspaper articles about food and keeping them organised in her files. However, there aren't many of these pieces to be found in news benches. She would be pleased with a system that made access to such material easier.

# 3. IDEATION & PROPOSED SOLUTION

## 3.1 Empathy Map Canvas


## 3.2 Ideation And Brain Storming



2

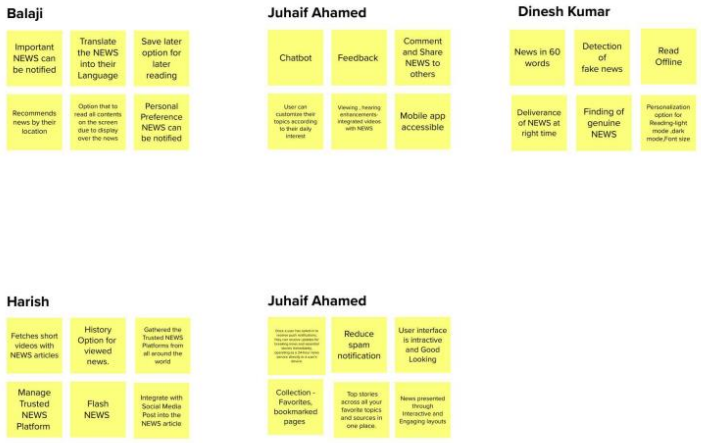
## Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

**TIP**

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!




3

## Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes


**TIP**

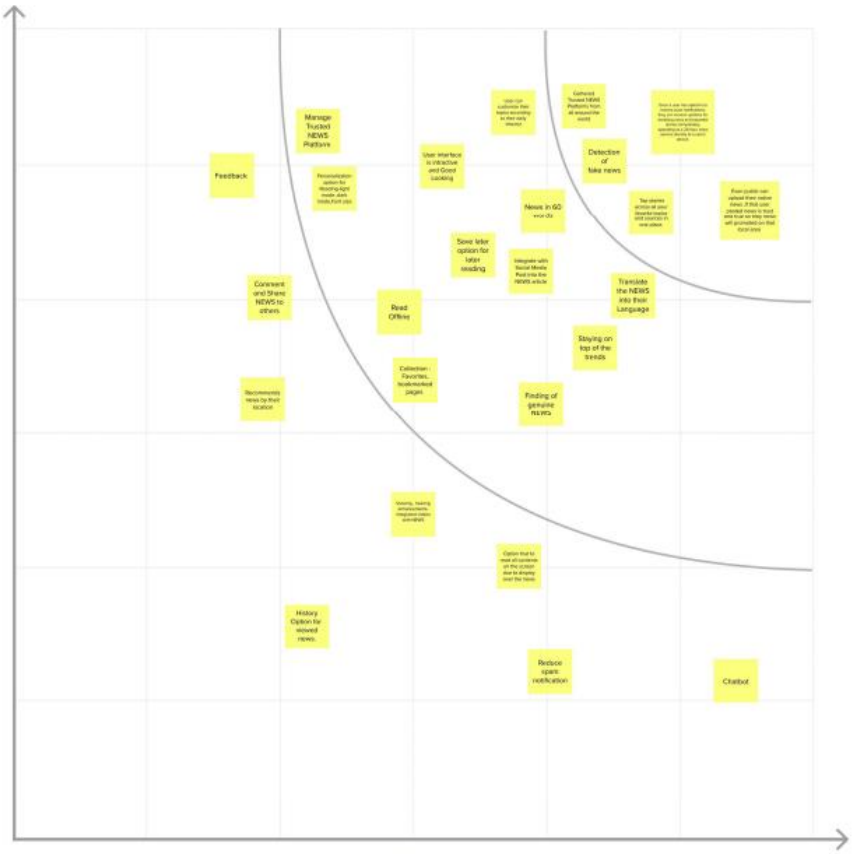
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

4

## Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes




**Importance**

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

**Feasibility**

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

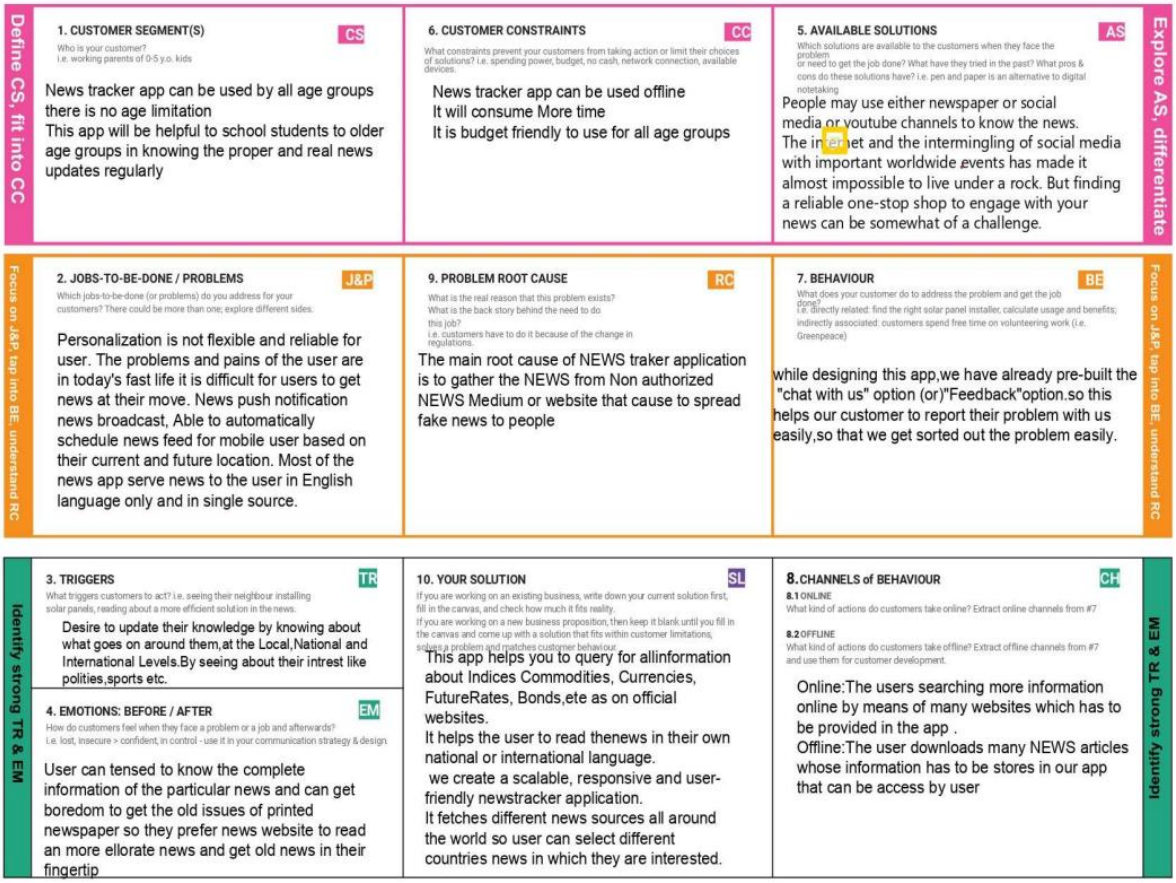
**TIP**

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the

## 3.3 Proposed Solution:

| S.No. | Parameter | Description |
|---|---|---|
| 1. | Problem Statement (Problem to be solved) | User needs a way to get a interested marketing in a quick peek of the day. So that we give relevant news where the user more interested on that day and save their time. |
| 2. | Idea / Solution description | The app should include all of the trustworthy sources from across the world and then specify in each article which source has validated this news because I'm assuming that customers can't discern the difference between real and fake news. |
| 3. | Novelty / Uniqueness | • Saving and downloading the interested articles<br>• Access Both Local and International News instantly<br>• Even Public can Upload their native news, if that user posted news is trust and true their news will promoted on that local area category<br>• Translate the NEWS articles in their native language<br>• User can access and customize their topics by their Interest then that News should be notify 24/7 |
| 4. | Social Impact / Customer Satisfaction | When news feed are customized/personalized to the user, the time spend to learn about the day will be minimized and will be completely productive. .For customer Satisfaction ,we Track all kind of news .it will be more Useful to get knowledge in Today's World. |
| 5. | Business Model (Revenue Model) | Our First and most important Business is to deliver a good content or good news to the people apart from that We are going to promote advertisement about good products. |

# 3.4 Problem Solution Fit



# 4.REQUIREMENT ANALYSIS

## 4.1 Functional requirement

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

| FR No. | Functional Requirement (Epic) | Sub Requirement (Story / Sub-Task) |
|---|---|---|
| FR-1 | User Installation | User can Browse the app from the website |
| FR-2 | User Registration | Registration through Form Registration through Gmail |
| FR-3 | User Confirmation | Confirmation via Email Confirmation via OTP |
| FR-4 | User Login | User should login the app with the user's name and password |

## 4.2 Non-Functional requirements

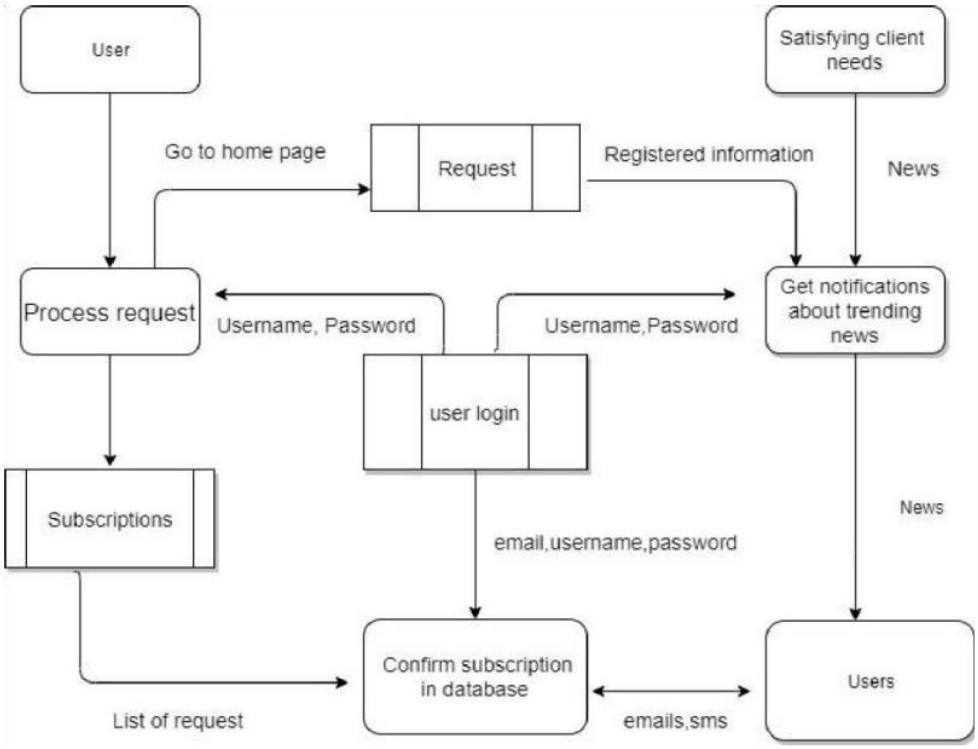
**Non-functional Requirements:**

Following are the non-functional requirements of the proposed solution.

| FR No. | Non-Functional Requirement | Description |
|---|---|---|
| NFR-1 | **Usability** | Everyone can understand the process of using the app easily by the commands given in the app. |
| NFR-2 | **Security** | This can be access only by the authorized API. so Security can be improved and No fake news can be shared. |
| NFR-3 | **Performance** | The updation of trending news occurs without any interruption. So, it performance is good |
| NFR-4 | **Availability** | This application will be available to the all the user who are using this application |

# 5.PROJECT DESIGN

## 5.1 Data Flow Diagrams



## 5.2 Solution &Technical Architecture

| S.No | Component | Description | Technology |
|---|---|---|---|
| 1. | User Interface | The user can interact with the application toknow about the trending news | HTML,CSS, JavaScript/ Angular Js/ ReactJs etc. |
| 2. | Application Logic-1 | The application contains this resource gives you basic understanding of Flask | Flask |
| 3. | Application Logic-2 | The application contains the news sub-division like geographical news, economicnews and society news | IBM Watson STT service |
| 4. | Application Logic-3 | The user can view the growth of the economy in industry through graph | IBM Watson Assistant |
| 5. | Database | Updation of trending news are stored in the MySQL database | MySQL, NoSQL, etc. |
| 6. | Cloud Database | With the use of cloud, media coverage issue cannot be occurred | IBM DB2, IBM Cloudant etc. |

| S.No | Characteristics | Description | Technology |
|---|---|---|---|
| 1. | Open-Source Frameworks | Flask is flexible and doesn't require touse any particular project or code layout used in this application | Python-Flask |
| 2. | Security Implementations | This can be access only by the journalist. So, It is a high Security | Container Registry, Kubernetes Cluster. |
| 3. | Scalable Architecture | News Tracker is a socio-economic access because helps to know aboutthe daily activity of the world | Container Registry, Kubernetes Cluster. |
| 4. | Availability | This application will be available to the all the user who are using this application | Container Registry, Kubernetes Cluster. |
| 5. | Performance | The updation of trending news occurs without any interruption. So,it performance is good | Kubernetes Cluster. |

## 5.3 User Stories

| User Type | Functional Requirement (Epic) | User Story Number | User Story / Task | Acceptance criteria | Priority | Release |
|---|---|---|---|---|---|---|
| Customer (Web user) | Browsing | USN-1 | Enter the web site on the browser | I can even login through browser | | |
| Customer (Web user) | | USN-2 | As a user, I can register for the application by entering my email, password, confirming my password and phone number. | I can access my account / dashboard | High | Sprint-1 |
| | | USN-3 | As a user, I will receive conformation email once I have registered for the | I can receive conformation email & | High | Sprint-1 |
| | | USN-4 | As a user, I can register for the application through Gmail | I can register & access the dashboard with Gmail account Login | Medium | Sprint-1 |
| | Login | USN-6 | As a user, I can log into the application by entering email & password | I can login to the official page | High | Sprint-1 |
| | Dashboard | USN-7 | Day to day news, feeds, categories, tech news and other updates after we save the news article for later read | I can see all the news which I wanted | High | Sprint-2 |
| | Local News Post | USN-8 | As a user can enter post their News article based on user's location , It can be authorized by that local people | I can post my local news which people want | High | Sprint-3 |
| | Local News authorized page | USN-9 | Local users can authorize the news and push into the local news category in their location | User can Authorize their news based on their location | High | Sprint-4 |

# 6.PROJECT PLANNING & SCHEDULING

## 6.1 Sprint Planning & Estimation

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority |
|---|---|---|---|---|---|
| Sprint-1 | Registration(Admin and Customer) | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 3 | High |
| Sprint-1 | | USN-2 | As a user, I will receive confirmation email once I have registered for the application | 5 | High |
| Sprint-1 | | USN-3 | As a user, I can register for the application through Facebook | 1 | Low |
| Sprint-1 | | USN-4 | As a user, I can register for the application through Gmail | 2 | Medium |
| Sprint-1 | Login (Admin and Customer) | USN-5 | As a user, I can log into the application by entering email & password | 1 | High |
| Sprint-2 | Dashboard(Admin and Customer) | USN-6 | As a user I should be able to navigate and access all the features hassle free | 5 | High |
| Sprint-2 | Layout | USN-7 | As a user I should be able to access the portal with different devices with the same comfort | 3 | High |
| Sprint-3 | Data Store,Retrieval and Authentication | USN-8 | Get Data from API and store as JSON in DB2 | 5 | High |
| Sprint-3 | | USN-9 | Get bin data from API and store in DFS | 3 | High |
| Sprint-3 | Local News Dashboard | USN-10 | Create a Option of post and authorize the news by User's location | 2 | High |
| Sprint-4 | User Segregation | USN-11 | As a CC executive I should be able to | 3 | Low |

| Sprint | Functional Requirement (Epic) | User Story Number | User Story / Task | Story Points | Priority |
|---|---|---|---|---|---|
| | and data access | | uniquely identify the customer and offer help | | |
| Sprint-4 | Change code | USN-12 | As a administrator I should be able to modify code according to the future requirements. | 2 | Medium |
| Sprint-4 | Monitor the system And Testing | USN-13 | As a administrator I should be able to monitor the cloud system and fix errors before customer. | 1 | High |
| Sprint-4 | Depolyment with Docker | USN-14 | As a User,I will deploy the entire Application using Docker. | 2 | Medium |
| Sprint-4 | Orchest with Kubernates | USN-15 | As a User,I will allocate the server nodes and balance the work loads in server. | 2 | Medium |

# 6.2 Sprint Delivery Schedule

**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

| Sprint | Total Story Points | Duration | Sprint Start Date | Sprint End Date (Planned) | Story Points Completed (as on Planned End Date) | Sprint Release Date (Actual) |
|---|---|---|---|---|---|---|
| Sprint-1 | 12 | 6 Days | 24 Oct 2022 | 29 Oct 2022 | 12 | 29 Oct 2022 |
| Sprint-2 | 8 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 8 | 05 Nov 2022 |
| Sprint-3 | 10 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 10 | 12 Nov 2022 |
| Sprint-4 | 10 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 10 | 19 Nov 2022 |

**Velocity:**
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

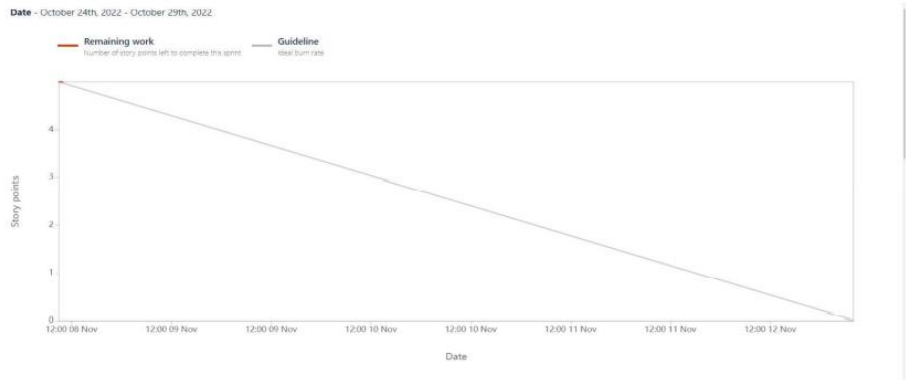
Average Velocity of Sprint-1 = 12/6 =2.0

Average Velocity of Sprint-2 = 8/6 =1.3

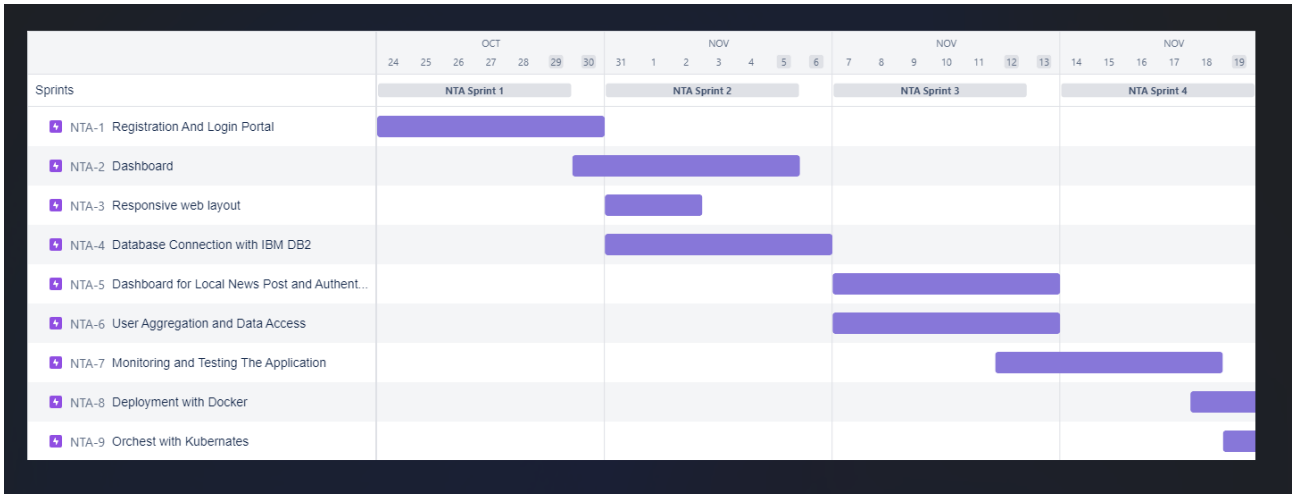
Average Velocity of Sprint-3 = 10/6 =1.6

Average Velocity of Sprint-4 = 10/6 =1.6

**Burndown Chart:**



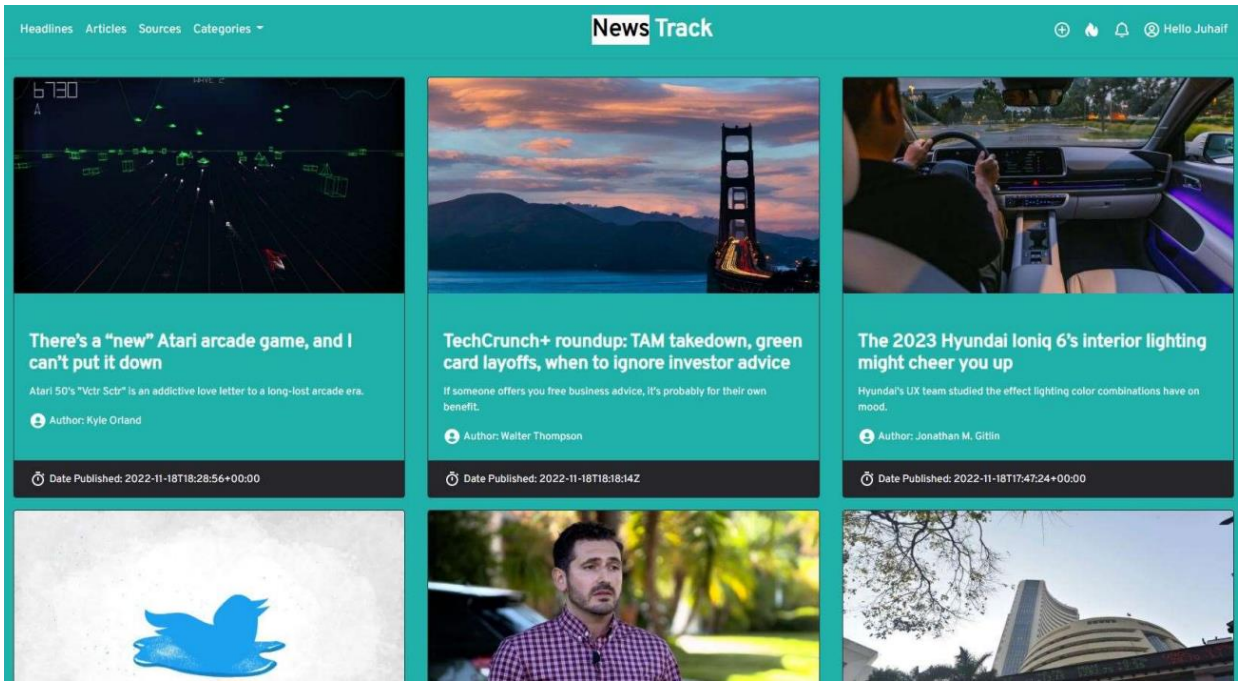

# 6.3 Reports from JIRA

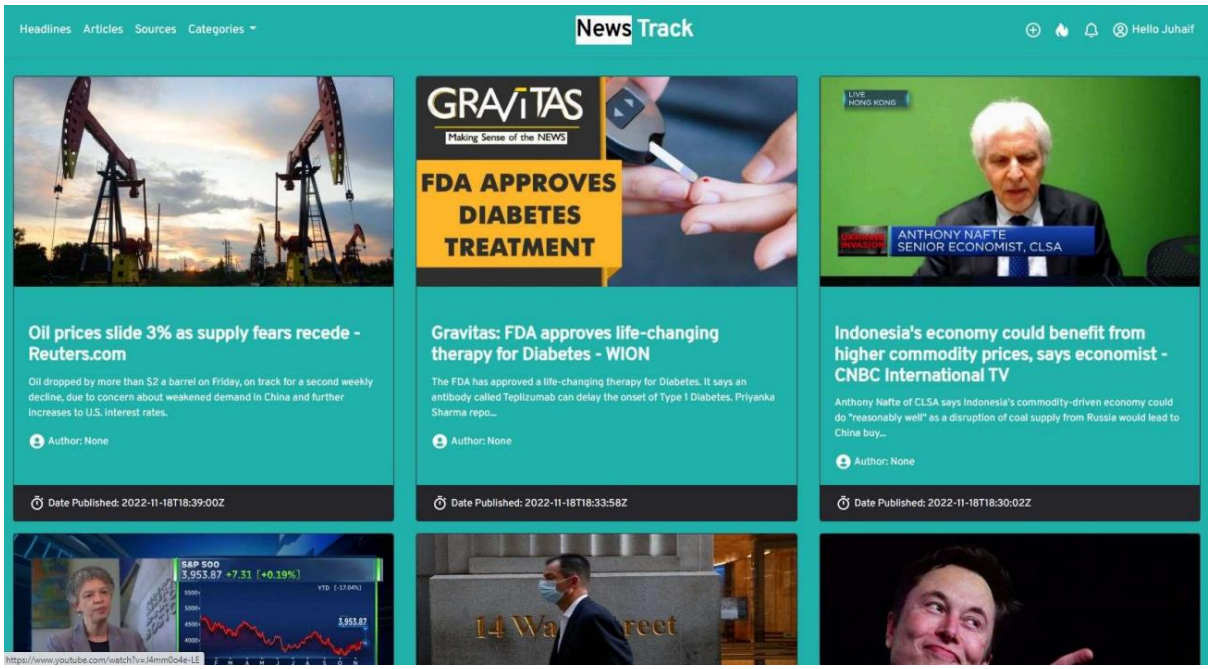


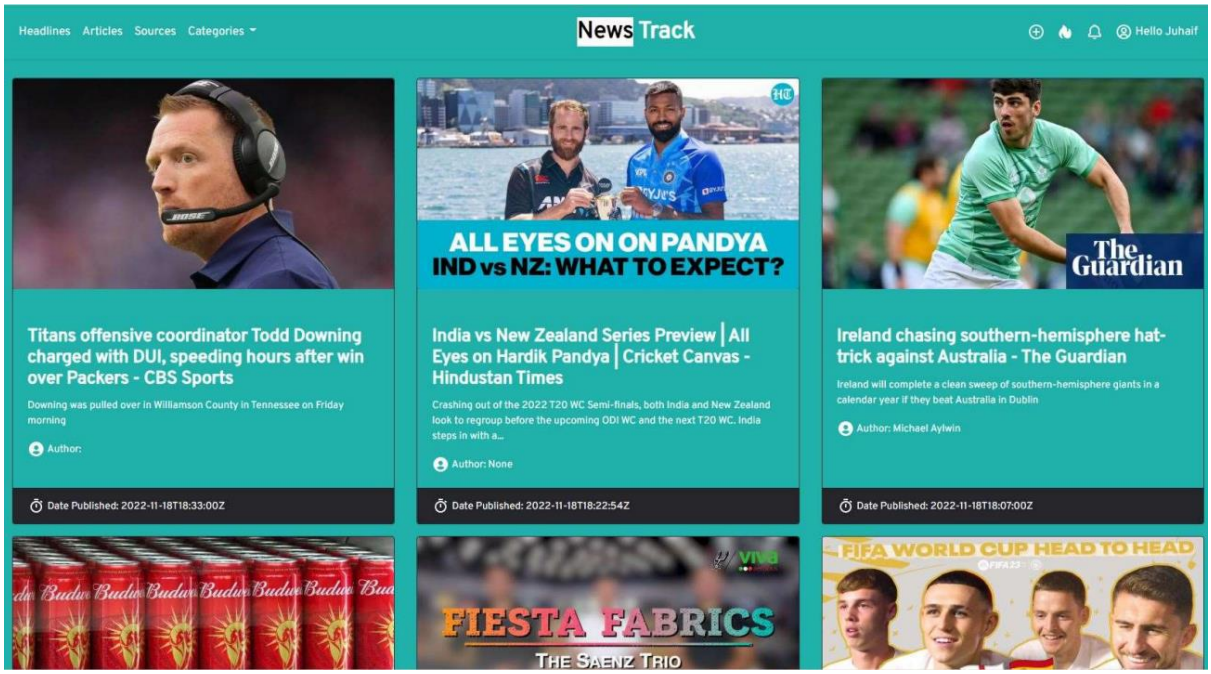
# 7. CODING AND SOLUTION

## 7.1 Feature 1

Access All Local and International News Around the word by categories

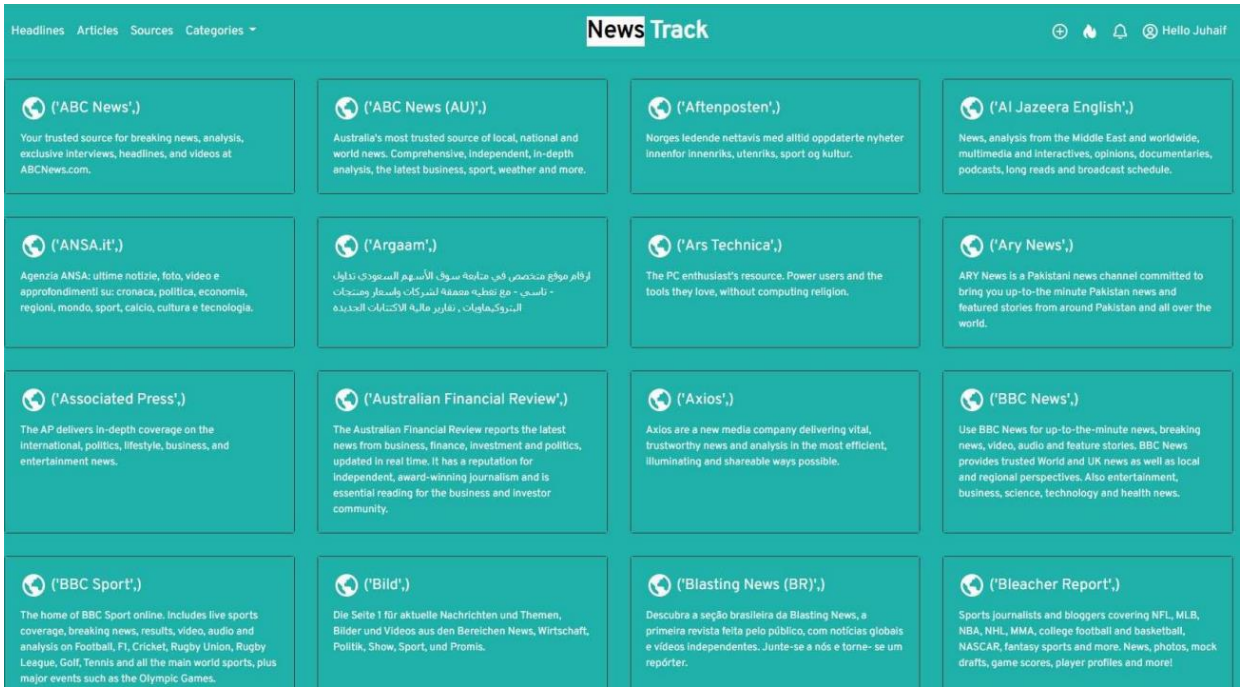


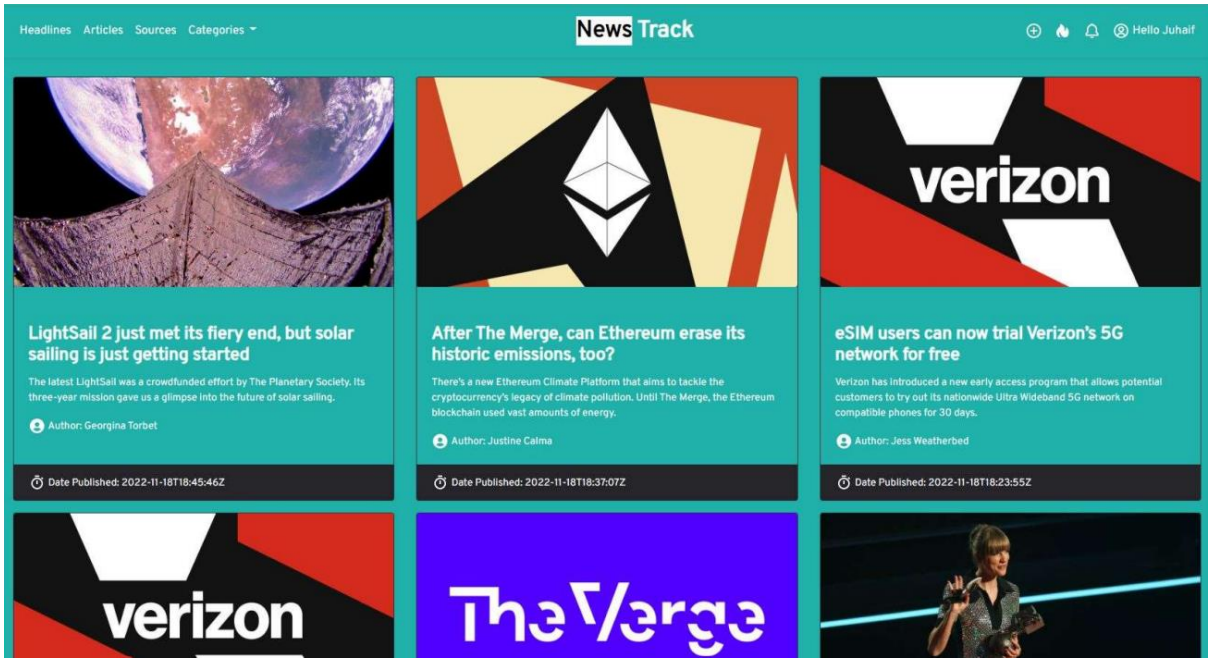






## 7.2 Feature 2

Trusted News Sources only accessed by using NewsAPI and We can View some Trending Articles from Popular Websites like as Verge,Forbes etc




Headlines Articles Sources Categories
News Track
Hello Juhaif
LightSail 2 just met its fiery end, but solar sailing is just getting started
The latest LightSail was a crowdfunded effort by The Planetary Society. Its three-year mission gave us a glimpse into the future of solar sailing.
Author: Georgina Torbet
Date Published: 2022-11-18T18:45:46Z
After The Merge, can Ethereum erase its historic emissions, too?
There's a new Ethereum Climate Platform that aims to tackle the cryptocurrency's legacy of climate pollution. Until The Merge, the Ethereum blockchain used vast amounts of energy.
Author: Justine Calma
Date Published: 2022-11-18T18:37:07Z
verizon
eSIM users can now trial Verizon's 5G network for free
Verizon has introduced a new early access program that allows potential customers to try out its nationwide Ultra Wideband 5G network on compatible phones for 30 days.
Author: Jess Weatherbed
Date Published: 2022-11-18T18:23:55Z
verizon
The Verge

# 8. TESTING

## 8.1 Test Case

| Test case ID | Feature Type | Component | Test Scenario | Pre-Requisite | Steps To Execute | Test Data | Expected Result | Actual Result | Status | Commnets | TC for Automation(Y/N) | BUG ID |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Functional | Login Page | Verify user is able to Login into the Application | | 1) Open the News tracker application.<br>2) Login with user Credentials<br>3) Verify logged in to user account | Email: jfad@gmail.com<br>Password: 1234 | Login Successful | Working as expected | Pass | | N | |
| 2 | Functional | Signup Page | Verify user is able to Signup in the Application | | 1) Open the news tracker<br>2) Enter the Details and Create a new User<br>3) Verify if user is created and inserted into DB Table | Email: jfadad@gmail.com<br>Password: 1234 | Account Created Successfully | Working as expected | Pass | | N | |
| 3 | Functional | Dashboard page | Verify if all the user details are stored in Database | | 1) Open the News tracker application.<br>2) Enter the Details and Create a new User<br>3) Verify if user is created and inserted into DB Table | Username:<br>jfadad@gmail.com<br>password: 1234 | User should navigate to user account homepage | Working as expected | Pass | | | |
| 4 | Functional | Login page | Verify user is able to log into application with InValid credentials | | 1.Enter URL and click go<br>2.Click on Sign IN button<br>3.Enter InValid username/email in Email text box<br>4.Enter valid password in password text box<br>5.Click on login button | Username:balaji@gmail.com<br>password: 592001 | Application should show 'Incorrect email or password ' validation message. | Working as expected | Pass | | | |
| 5 | Functional | Login page | Verify user is able to log into application with InValid credentials | | 1.Enter URL and click go<br>2.Click on Sign IN button<br>3.Enter InValid username/email in Email text box<br>4.Enter valid password in password text box<br>5.Click on login button | Username:harish@gmail.com password:32002 | Application should show 'Incorrect email or password ' validation message. | Working as expected | Pass | | | |

## 8.2 User Acceptance Testing

1.PURPOSE OF DOCUMENT

The purpose of this document is to briefly explain the test coverage and open issues of the News Tracker Application project at the time of the release to User Acceptance Testing(UAT).

2.DEFECT ANALYSIS

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved.

| Resolution | Severity 1 | Severity 2 | Severity 3 | Severity 4 | Subtotal |
|---|---|---|---|---|---|
| By Design | 3 | 1 | 2 | 1 | 7 |
| Duplicate | 1 | 0 | 0 | 0 | 1 |
| External | 1 | 0 | 0 | 1 | 2 |
| Fixed | 2 | 1 | 1 | 1 | 5 |
| Not Reproduced | 0 | 0 | 0 | 0 | 1 |
| Skipped | 0 | 0 | 0 | 0 | 0 |
| Won't Fix | 0 | 0 | 0 | 0 | 0 |
| Totals | 7 | 2 | 3 | 3 | 16 |

3.TEST CASE ANALYSIS

This report shows the number of test cases that have passed, failed, and untested.

| Section | Total Cases | Not Tested | Fail | Pass |
|---|---|---|---|---|
| Login Page | 4 | 0 | 0 | 4 |
| Registration Page | 1 | 0 | 0 | 1 |
| Home Page | 2 | 0 | 0 | 2 |

## 9. ADVANTAGES

1.Viewers can get their news straight off their smartphone or tablet computer.

2. News is at their fingertips in an instant. An online newspaper can be read more elaborate than a printed newspaper.

3. You can read the Popular News Articles too very easily at the click of the mouse.

4. Access the News By Categories

## DISADVANTAGES

1. It can limited by time.

2. It may rely too heavily on personalities, emotions, opinions… not facts.

3. It can shortchange complex stories or avoid them altogether.

## 10. CONCLUSION

The Motivation and scope behind this project are to connect people through this application and provide a medium to share their views on the topic/news/information. Then, People with the same interest can interact with each other. However, they can even share more information on the topic. This app while cross-checks the redundancy in the information along with the false and misleading information, which later results in panic in the people.

## 11. FUTURE SCOPE

Location feature with automation can be implemented which means as user move from one city to other local news will change as per it. Offline Reading can be improve will more efficient way on full articles. Data quality check needed. If API can't reach to certain article source it gives null value which can cause problem in JSON parsing.

## 12. APPENDIX

## SOURCE CODE :

### main.py

```
from app import app


if __name__ == "__main__":
    app.run()
```

### __init__.py

```
from flask import Flask


app = Flask(__name__)


from app import views
```

### views.py

```
from app import app
from flask import render_template, redirect
from flask import url_for
from flask import request
from .request import businessArticles, entArticles, get_news_source,
healthArticles, publishedArticles, randomArticles, scienceArticles,
sportArticles, techArticles, topHeadlines
```

```
import ibm_db

import re

from sendgrid import SendGridAPIClient

from sendgrid.helpers.mail import Mail


app.secret_key = 'a'

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=19af6446-6171-4641-8aba-
9dcff8e1b6ff.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30699;SECURIT
Y=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=vdw12720;PWD=2C3yBJCDv
rFURLPQ",'','')


@app.route('/', methods=['GET', 'POST'])
def login():

    if request.method == 'POST':

        # getting user data

        email = request.form.get('email')

        password = request.form.get('password')

        sql_check_query = "SELECT * FROM user WHERE email = ?"

        stmt = ibm_db.prepare(conn, sql_check_query)

        ibm_db.bind_param(stmt, 1, email)

        ibm_db.execute(stmt)

        account = ibm_db.fetch_assoc(stmt)

        print(account)

        if account:

            # email id exists

            # checking if the password is correct

            if not account['PASSWORD'] == password:

                flash('Invalid password', category='error')


            else:

                # user entered the correct password

                # redirecting the user to the dashboard
```

```
                session['user_id'] = account['EMAIL']
                return redirect(url_for('home'))

        else:
            # email id does not exist in the database
            flash('Email invalid... Try Again', category='error')

        return render_template('auth/login.html')

    return render_template('auth/login.html')
    # return render_template('login.html')

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        # getting user data
        email = request.form.get('email')
        password = request.form.get('password')
        # checking: user already exists or not
        sql_check_query = "SELECT * FROM user WHERE email = ?"
        stmt = ibm_db.prepare(conn, sql_check_query)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)

        account = ibm_db.fetch_assoc(stmt)
        # email id does not exist in the database
        if not account:
            # inserting the data into the database
            sql_insert_query = "INSERT INTO user VALUES (?, ?)"
            stmt = ibm_db.prepare(conn, sql_insert_query)
            ibm_db.bind_param(stmt, 1, email)
```

```
            ibm_db.bind_param(stmt, 2, password)
            ibm_db.execute(stmt)


            # user data inserted into the database
            # redirecting to login page
            flash('User created successfully! Please Login',
category='success')
            return redirect('/')


        else:
            flash('Email id already exists! Try another one',
category='error')


        return render_template('auth/register.html')


    return render_template('auth/register.html')
    # return render_template('register.html')


@app.route('/home')
def home():
    articles = publishedArticles()


    return  render_template('home.html', articles = articles)


@app.route('/headlines')
def headlines():
    headlines = topHeadlines()


    return  render_template('headlines.html', headlines = headlines)


@app.route('/articles')
def articles():
```

```
    random = randomArticles()

    return  render_template('articles.html', random = random)

@app.route('/sources')
def sources():
    newsSource = get_news_source()

    return  render_template('sources.html', newsSource = newsSource)

@app.route('/category/business')
def business():
    sources = businessArticles()

    return  render_template('business.html', sources = sources)

@app.route('/category/tech')
def tech():
    sources = techArticles()

    return  render_template('tech.html', sources = sources)

@app.route('/category/entertainment')
def entertainment():
    sources = entArticles()

    return  render_template('entertainment.html', sources = sources)

@app.route('/category/science')
def science():
    sources = scienceArticles()
```

```
    return  render_template('science.html', sources = sources)

@app.route('/category/sports')
def sports():
    sources = sportArticles()

    return  render_template('sport.html', sources = sources)

@app.route('/category/health')
def health():
    sources = healthArticles()

    return  render_template('health.html', sources = sources)
```

**request.py**

```
from .models import Articles
from .models import Sources
from newsapi import NewsApiClient
from .config import Config
import urllib.request,json

api_key=None
base_url=None
base_url_for_everything=None
base_url_top_headlines=None
base_source_list=None

def publishedArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)
```

```
    get_articles = newsapi.get_everything(sources= 'cnn, reuters, cnbc, the-
verge, gizmodo, the-next-web, techradar, recode, ars-technica')


    all_articles = get_articles['articles']


    articles_results = []


    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []


    for i in range(len(all_articles)):
        article = all_articles[i]


        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])


        article_object = Articles(source, title, desc, author, img, p_date,
url)


        articles_results.append(article_object)
```

```
        contents = zip(source, title, desc, author, img, p_date, url)


    return  contents


def topHeadlines():
    newsapi = NewsApiClient(api_key= Config.API_KEY)


    top_headlines = newsapi.get_top_headlines(sources= 'cnn, reuters, cnbc,
techcrunch, the-verge, gizmodo, the-next-web, techradar, recode, ars-
technica')


    all_headlines = top_headlines['articles']


    articles_results = []


    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []


    for i in range(len(all_headlines)):
        headline = all_headlines[i]


        source.append(headline['source'])
        title.append(headline['title'])
        desc.append(headline['description'])
        author.append(headline['author'])
        img.append(headline['urlToImage'])
        p_date.append(headline['publishedAt'])
```

```
        url.append(headline['url'])


        article_object = Articles(source, title, desc, author, img, p_date,
url)


        articles_results.append(article_object)


        contents = zip(source, title, desc, author, img, p_date, url)


    return  contents


def randomArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)


    random_articles = newsapi.get_everything(sources= 'the-verge, gizmodo,
the-next-web, recode, ars-technica')


    all_articles = random_articles['articles']


    articles_results = []


    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []


    for i in range(len(all_articles)):
        article = all_articles[i]
```

```
        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        articles_results.append(article_object)

        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents

def businessArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)

    business_articles = newsapi.get_top_headlines(category='business')

    all_articles = business_articles['articles']

    business_articles_results = []

    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
```

```
    url = []

    for i in range(len(all_articles)):
        article = all_articles[i]

        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        business_articles_results.append(article_object)

        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents

def techArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)

    tech_articles = newsapi.get_top_headlines(category='technology')

    all_articles = tech_articles['articles']

    tech_articles_results = []

    source = []
```

```
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []

    for i in range(len(all_articles)):
        article = all_articles[i]

        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        tech_articles_results.append(article_object)

        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents

def entArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)

    ent_articles = newsapi.get_top_headlines(category='entertainment')
```

```
    all_articles = ent_articles['articles']

    ent_articles_results = []

    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []

    for i in range(len(all_articles)):
        article = all_articles[i]

        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        ent_articles_results.append(article_object)

        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents
```

```
def scienceArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)

    science_articles = newsapi.get_top_headlines(category='science')

    all_articles = science_articles['articles']

    science_articles_results = []

    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []

    for i in range(len(all_articles)):
        article = all_articles[i]

        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        science_articles_results.append(article_object)
```

```
        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents

def sportArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)

    sport_articles = newsapi.get_top_headlines(category='sports')

    all_articles = sport_articles['articles']

    sport_articles_results = []

    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []

    for i in range(len(all_articles)):
        article = all_articles[i]

        source.append(article['source'])
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
```

```
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        sport_articles_results.append(article_object)

        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents

def healthArticles():
    newsapi = NewsApiClient(api_key= Config.API_KEY)

    health_articles = newsapi.get_top_headlines(category='health')

    all_articles = health_articles['articles']

    health_articles_results = []

    source = []
    title = []
    desc = []
    author = []
    img = []
    p_date = []
    url = []

    for i in range(len(all_articles)):
        article = all_articles[i]

        source.append(article['source'])
```

```
        title.append(article['title'])
        desc.append(article['description'])
        author.append(article['author'])
        img.append(article['urlToImage'])
        p_date.append(article['publishedAt'])
        url.append(article['url'])

        article_object = Articles(source, title, desc, author, img, p_date,
url)

        health_articles_results.append(article_object)

        contents = zip(source, title, desc, author, img, p_date, url)

    return  contents

def get_news_source():
  '''
  Function that gets the json response to our url request
  '''
  get_news_source_url = 'https://newsapi.org/v2/sources?apiKey=' +
Config.API_KEY
  with urllib.request.urlopen(get_news_source_url) as url:
    get_news_source_data = url.read()
    get_news_source_response = json.loads(get_news_source_data)

    news_source_results = None

    if get_news_source_response['sources']:
      news_source_results_list = get_news_source_response['sources']
      news_source_results = process_sources(news_source_results_list)
```

```
  return news_source_results


def process_sources(source_list):
  '''
  function that process the news articles and transform them to a list of objects
  '''
  news_source_result = []
  for news_source_item in source_list:
    name = news_source_item.get('name')
    description = news_source_item.get('description')
    url = news_source_item.get('url')


    if name:
      news_source_object = Sources(name, description,url)
      news_source_result.append(news_source_object)
  return news_source_result
```

**models.py**

```
class Sources:
    def __init__(self, name, description, url):
        self.name=name,
        self.description=description
        self.url=url


class Articles:
    '''Define article model'''
    def __init__(self, source, author, title, description, url, urlToImage, publishedAt):
        self.source = source
        self.author = author
        self.title = title
```

```
        self.description = description
        self.url = url
        self.urlToImage = urlToImage
        self.publishedAt = publishedAt
```

**config.py**

```
class Config:


    NEWS_BASE_URL_SOURCES = 'https://newsapi.org/v2/top-
headlines/sources?apiKey={}'
    NEWS_BASE_EVERYTHING_URL =
'https://newsapi.org/v2/everything?domains={}&apiKey={}'
    NEWS_BASE_HEADLINES_URL = 'https://newsapi.org/v2/top-
headlines?country=us&apiKey={}'
    NEWS_BASE_SOURCE = 'https://newsapi.org/v2/top-
headlines?sources={}&apiKey={}'
    API_KEY = "12d02fd71ab3406d9ba3b36454e7f092"


class ProdConfig(Config):
    pass


class DevConfig(Config):
    DEBUG = True


config_options= {
    'development': DevConfig,
    'production': ProdConfig
}
```

**GITHUB LINK**

https://github.com/IBM-EPBL/IBM-Project-51210-1660975785