

## **NEWS TRACKER APPLICATION**

**IBM NALAIYATHIRAN (HX8001)**

**PROJECT REPORT**

**SUBMITTED BY**

**TEAM\_ID:PNT2022TMID26501**

**KAMALESH D      211719106034**

**LATHA D          211719106040**

**LAVANYA S       211719106041**

**GOWTHAM S     211719106019**

*in*

*partial fulfillment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**RAJALAKSHMI INSTITUTE OF TECHNOLOGY**

**ANNA UNIVERSITY: CHENNAI 600 025**

**NOVEMBER 2022**

**ANNA UNIVERSITY: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**NEWS TRACKER APPLICATION**” is the bonafide work of “**KAMALESH D (211719106034), LATHA D (211719106040), LAVANYA S (211719106041) and GOWTHAM S (211719106019)**”, who carried out the project work under my supervision.

**SIGNATURE:**

**Dr. S.MANJULA, M.E., Ph.D.,**  
**HEAD OF THE DEPARTMENT,**

Dept. of Electronics and  
Communication Engg.,  
Rajalakshmi Institute of  
Technology,  
Kuthambakkam Post,  
Chennai - 600 124

**SIGNATURE:**

**Mr. A.BALAJI, M.E., (Ph.D).,**  
**MENTOR,**

Dept. of Electronics and  
Communication Engg.,  
Rajalakshmi Institute of  
Technology,  
Kuthambakkam Post,  
Chennai - 600 124

The viva-voce is held on \_\_\_\_\_.

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

# TABLE OF FIGURES

1. INTRODUCTION
  - 1.1 Project Overview
  - 1.2 Purpose
2. LITERATURE SURVEY
  - 2.1 Existing problem
  - 2.2 References
  - 2.3 Problem Statement Definition
3. IDEATION & PROPOSED SOLUTION
  - 3.1 Empathy Map Canvas
  - 3.2 Ideation & Brainstorming
  - 3.3 Proposed Solution
  - 3.4 Problem Solution fit
4. REQUIREMENT ANALYSIS
  - 4.1 Functional requirement
  - 4.2 Non-Functional requirements
5. PROJECT DESIGN
  - 5.1 Data Flow Diagrams
  - 5.2 Solution & Technical Architecture
  - 5.3 User Stories
6. PROJECT PLANNING & SCHEDULING
  - 6.1 Sprint Planning & Estimation
  - 6.2 Sprint Delivery Schedule
  - 6.3 Reports from JIRA
7. CODING & SOLUTIONING
  - 7.1 Feature
8. TESTING
  - 8.1 Test Cases
  - 8.2 User Acceptance Testing
9. RESULTS
10. ADVANTAGES & DISADVANTAGES
11. CONCLUSION
12. FUTURE SCOPE
13. APPENDIX(Source Code, GitHub & Project Demo Link)

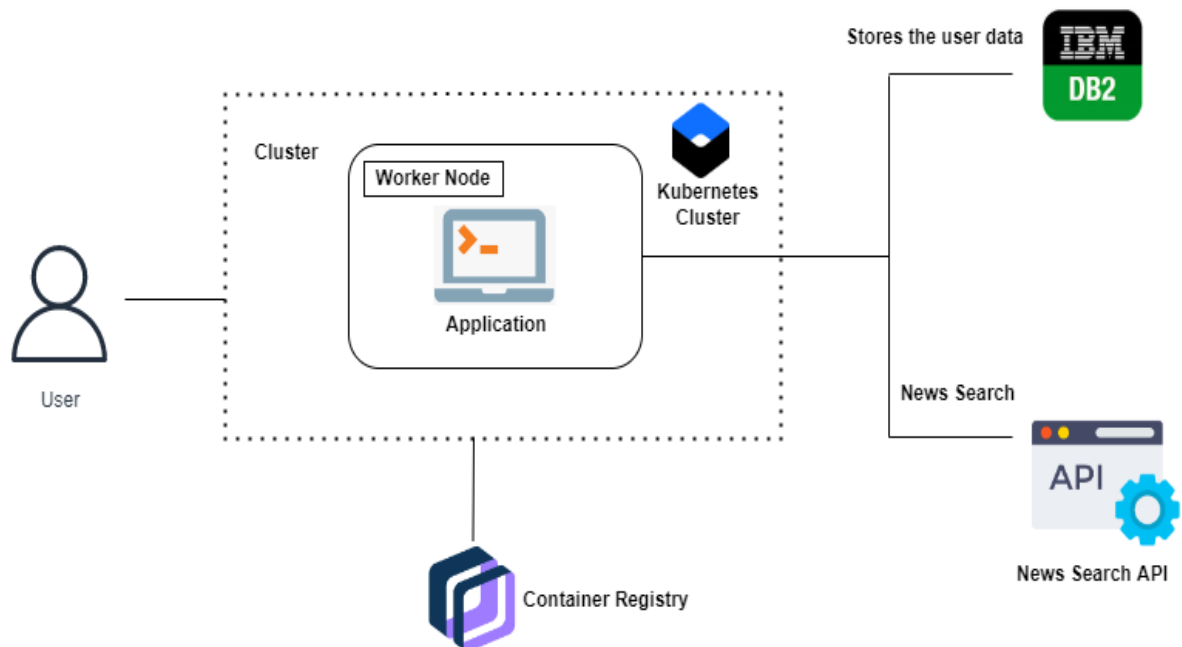
## 1. INTRODUCTION :

Nowadays, the news media are subject to the social scrutiny because of the lack of credibility, the manipulative media, the misinformation campaigns, and the propagation of fake news point out that the main difference between fake news and true news relies on lack editorial norms and processes that ensure the accuracy and credibility of the information. Thus, to arrange a way that allows guaranteeing these editorial processes (or at least part of them) can suppose a big step in the fight against the above-mentioned issues. To track the evolution of the news reports and the relevant data and information it contains as they change over time, and therefore to trace how the related news evolves, constitute other instruments to face the previous issues. This is not only useful for end readers but for fact-checking agencies and those tools that perform automatic indexing and extraction of relevant information of news. Once the information has been verified and fact-checked, these tools need a way to guarantee that the extracted data have not changed.

### 1.1 Project Overview

#### News Tracker Application

##### Technical Architecture:



### **Project Workflow:**

- The user interacts with the application.
- Registers by giving the details.
- Integrate the application with news APIs and store the data in the database.
- The database will have all the details and the user can search the news by using a search bar.

## **1.2 Purpose**

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.... as on official websites.

## **2. LITERATURE SURVEY**

### **2.1 Existing problem**

News is one of the primary source of gaining information about the actions and events that happen all around. It may be an event that happened in the past, happening now or going to happen in the future. In the present days where there is a rapid increase in the development and adaptability of technologies throughout all the demographic of people, it is necessary to provide news in such a way that it is interconnected with the current technological trends. 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.... as on official websites

## 2.2 References

| S.No | Paper Title                                                             | Author(s)                                                          | Month/ Year   | Method/ Implementation Technique(s)                                                                                                                                                                                             | Resource Link                                                                                                                                                                                                                                                       |
|------|-------------------------------------------------------------------------|--------------------------------------------------------------------|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | Exploring mobile news reading interactions for news app personalization | Marios Constantinides, John Dowell, David Johson, Sylvain Malacria | August, 2015  | 1. Identification of news reader types<br>2. Interaction logging and classification study<br>3. Deployment and data collection<br>4. Predicting News readertypes<br>5. Adaptive UI                                              | <a href="https://www.researchgate.net/publication/299870645_Exploring_mobile_news_reading_interactions_for_news_app_personalisation">https://www.researchgate.net/publication/299870645_Exploring_mobile_news_reading_interactions_for_news_app_personalisation</a> |
| 2    | Detection and Tracking in News Articles                                 | Sagar Patel, Sanket Suthar, Sandip Patel, Neha Patel               | March, 2015   | 1. Preprocessing<br>2. Tokenization<br>3. Stemming/L emmization<br>4. Vector SpaceModel<br>5. Topic tracking                                                                                                                    | <a href="https://www.researchgate.net/publication/315657099_Topic_Detection_and_Tracking_in_News_Articles">https://www.researchgate.net/publication/315657099_Topic_Detection_and_Tracking_in_News_Articles</a>                                                     |
| 3    | Following the Fed with a News Tracker                                   | Michael William McCracken                                          | January, 2012 | The paper is not a technical paper but is essentially a statistical paper on how should one conclude whether the data have come in stronger, weaker or as expected. This is based on the CitiGroup U.S Economic Surprise Index. | <a href="https://www.researchgate.net/publication/227438253_Following_the_Fed_with_a_News_Tracker">https://www.researchgate.net/publication/227438253_Following_the_Fed_with_a_News_Tracker</a>                                                                     |
| 4    | An End-to-end Weaklysupervised News Aggregation Framework               | Xijin Tang, Xiaohui Huang                                          | June, 2022    | The framework combines Snorkelbased weaklysupervised classification, Latent Dirichlet Allocation (LDA) topic modeling, and topic signal detection model to classify and aggregate unlabeled                                     | <a href="https://www.researchgate.net/publication/361087328_An_End-to-end_Weakly-supervised_News_Aggregation_Framework">https://www.researchgate.net/publication/361087328_An_End-to-end_Weakly-supervised_News_Aggregation_Framework</a>                           |

|  |  |  |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |  |
|--|--|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
|  |  |  |  | news texts and ultimately generate visualized results containing news categories, news topics, and temporal topic relationships. This paper uses constructed knowledge thesaurus and the Snorkel method to weakly supervise the classification of unlabeled news with no manual tagging. Subsequently, we utilize LDA to generate the topics and obtain the signal value of each topic based on the topic signal detection function. Finally, we establish the temporal topic relationships and get the visualized results of news aggregation. |  |
|--|--|--|--|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|

### 2.3 Problem Statement Definition

Mr. John is a 52 years old man, who frequently reads news, he wishes to read the pertinent market news and needs to get help with queries about indices, commodities, currencies, future rates, bond, etc... Through News Tracker Application.

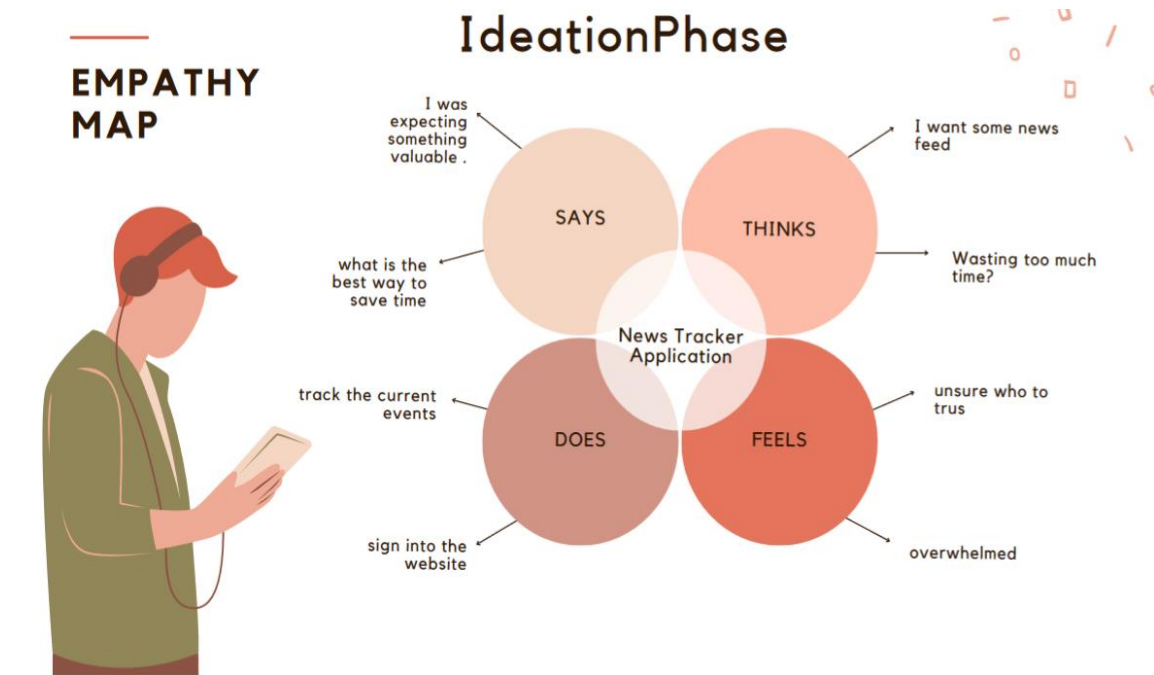
|                                         |                                                                                                                                                                                                                                                                                                                   |
|-----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Who does the problem affect?            | Person who searches for authentic news.                                                                                                                                                                                                                                                                           |
| What are the boundaries of the problem? | People with lack of internet facility and facing the difficulty in operating the software.                                                                                                                                                                                                                        |
| What is the issue?                      | As our lives are very busy now-a-days, we often feel we need more than 24 hrs. A day to come up with everything to be updated ourself via news in our schedule. Well, that's not possible but reducing the time by using the conventional method of reading news can help at any time in any place to be updated. |
| When does the issue occur?              | When we search for pertinent news.                                                                                                                                                                                                                                                                                |



|                                              |                                                                                                                                                                                                                                                                                                                                        |
|----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Where does the issue occur?                  | This issue occurs for common people, business men.                                                                                                                                                                                                                                                                                     |
| Why is it important that we fix the problem? | It is important to read the relevant news and save time.                                                                                                                                                                                                                                                                               |
| What solution to solve this issue?           | This could be solved by collecting relevant information from the user through a news tracker application. App will provide a quick snap to save your time.                                                                                                                                                                             |
| What methodology was used to the issue?      | <ul style="list-style-type: none"> <li>➤ The User interacts with the application.</li> <li>➤ Registers by giving the details.</li> <li>➤ Integrate the application with News APIs and store the data in Database.</li> <li>➤ The database will have all the details and the user can search the news by using a search bar.</li> </ul> |

### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas






## 3.2 Ideation & Brainstorming

Template



### Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

-  **10 minutes** to prepare
-  **1 hour** to collaborate
-  **2-8 people** recommended

 [Share template feedback](#)



## Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes



### Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.



### Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.



### Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#)



1

## Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes

### PROBLEM

The user needs a way to get relevant news based on his choices so that the user does not have to spend a lot of time on searching news. News is filled with ads and spams, annoys and irritates the user and affects the user experience. Market is full of news with all categories. Users are not interested in all the categories and will be more interested with their personal choices. Traditional way of tracking news is slow and obsolete, user needs a new innovative application to track news with personal based choices. Lack of quick updates for the day results in staying behind in the trend in today's world and tracking the news regularly should be done.



### Key rules of brainstorming

To run a smooth and productive session



Stay in topic.



Encourage wild ideas.



Defer judgment.



Listen to others.



Go for volume.



If possible, be visual.

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!

### KAMALESH D

|                             |                   |                       |
|-----------------------------|-------------------|-----------------------|
| Sign UP                     | Responsive UI     | Add to Favourites     |
| News Sharing                | Maintain Database | Hash User credentials |
| User can add their own news | Modify News       | Security              |

### LATHA D

|                   |                          |                   |
|-------------------|--------------------------|-------------------|
| Accurate Results  | Engaging Cloud App       | Automation of App |
| 24/7 Availability | Easy to Maintain         | No Ads            |
| Flask Backend     | Better Backup & Security | Regular Updates   |

### LAVANYA S

|                     |                         |                       |
|---------------------|-------------------------|-----------------------|
| Choice Based result | Less Distraction        | Regular Notifications |
| Reliable news       | Write maintainable code | User Data Privacy     |
| Lightweight         | Simple UI               | IBM Cloud             |

### GOWTHAM S

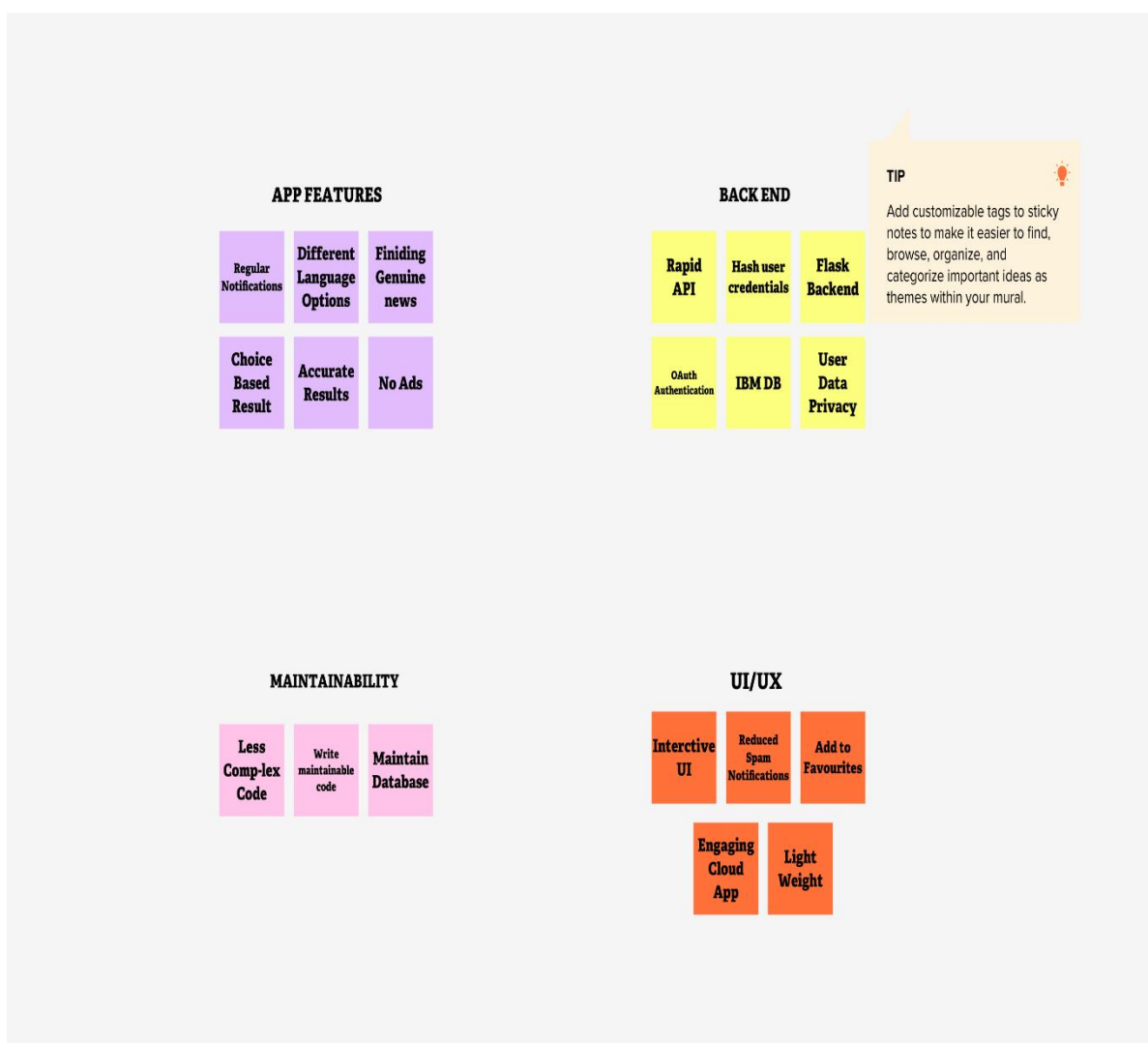
|                      |                            |                      |
|----------------------|----------------------------|----------------------|
| Rapid API            | Interactive UI             | Containerize Docker  |
| Faster Data Fetching | APIs For Data Presentation | OAuth Authentication |
| IBM DB               | Less Complex code          | User Friendly        |

3

## Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

🕒 20 minutes

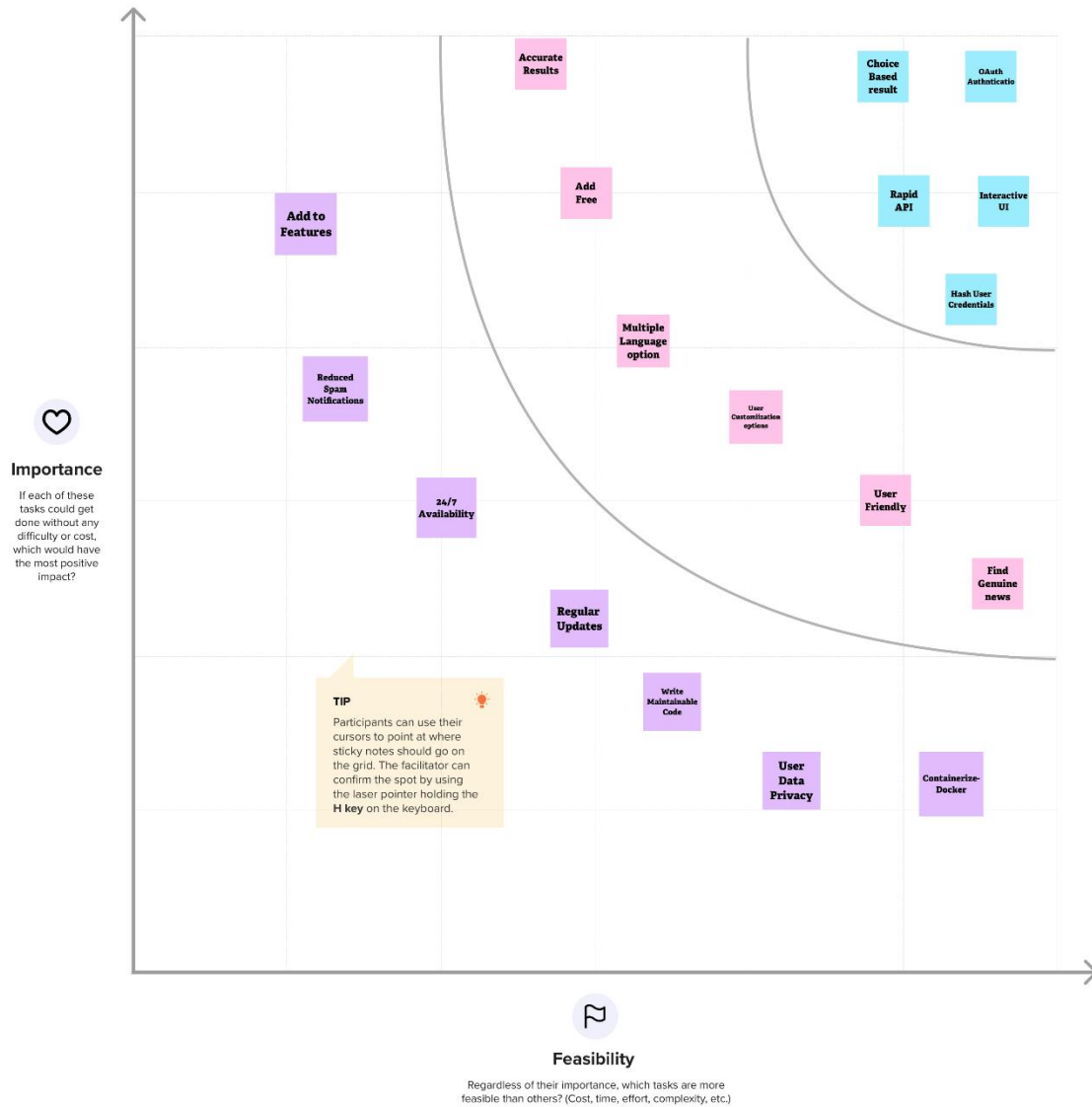


4

## 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





## After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

---

### Quick add-ons



#### Share the mural

**Share a view link** to the mural with stakeholders to keep them in the loop about the outcomes of the session.



#### Export the mural

Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

---

### Keep moving forward



#### Strategy blueprint

Define the components of a new idea or strategy.

[Open the template →](#)



#### Customer experience journey map

Understand customer needs, motivations, and obstacles for an experience.

[Open the template →](#)



#### Strengths, weaknesses, opportunities & threats

Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.

[Open the template →](#)



[Share template feedback](#)



### **3.3 Proposed Solution**

#### **ABSTRACT**

News Tracker is to collect a variety of sources of misinformation on social media and aggregate the stories published so that we could have a clearer sense of the different types of misinformation out there. We wanted to know how frequently it was published and what strategies and narratives were employed to engage audiences on social media. To do this, we built a database of misinformation sites, identified their social media pages, and found any related to that also shared the content. We used the social media API to collect all the stories posted to these pages as well as their engagement data — how many times users liked, shared, and commented on the stories. We continuously monitored this feed to identify patterns in and assess the substance of the posts. Fake news is a false piece of information. Any piece of Fake news can be created to deliberately misinform or deceive readers, promote a biased point of view, particular cause or agenda, and even for the entertainment. Fake news can be promoted by unauthenticated user ID, social media, printing of fake news in newspapers may be due to political pressure and many more. Spreading of fake news can cause discontent among people, riots, and even cause loss of trust between two people and even nations. It also has a huge possibility to exploit general public thinking in a completely different manner. News and media coverage gets hugely distorted due to initialization and spread of fake news.

#### **PROPOSED SYSTEM**

The proposed system we are using a hybrid support vector machine and Random forest model for training. We are combining the two machine learning techniques so that we can able to get the accuracy score of above 98% for large datasets also.

### 3.4 Problem Solution fit

Project Title: NEWS TRACKER APPLICATION

Project Design Phase-I - Solution Fit Template

Team ID: PNT2022TMD26501

|                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                            |                                                                                                                                                                                                                                                             |                           |                                          |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|------------------------------------------|
| Define CS, fit into CC                                                                                                                                                                                                                | <b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span><br><p>The one who can't find time to read news paper</p> <p>People who just want to read relevant news.</p>                                                                                                                                                                                            | <b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span><br><p>Network issues while scrolling for news</p> <p>Responsivity of application</p> <p>Giving relevant news according to their search.</p>                                                                 | <b>5. AVAILABLE SOLUTIONS</b> <span>AS</span><br><p>They may choose newspaper or YouTube for news.</p> <p>Various news applications are available in various languages.</p>                                                                                 | Explore AS, differentiate |                                          |
|                                                                                                                                                                                                                                       | <b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span><br><p>They can't reschedule / give separate time to read news paper</p> <p>The problem while searching for the relevant news</p> <p>The particular news can't be searched in case of newspaper.</p> <p>The interests of the customer can't be managed rather than digital application.</p> | <b>9. PROBLEM ROOT CAUSE</b> <span>RC</span><br><p>Newspapers give day-to- day news only.</p> <p>A day to cope with everything we have in our schedule</p>                                                                                                 | <b>7. BEHAVIOUR</b> <span>BE</span><br><p>People may use applications but, they have constraints</p> <p>People may surf the internet but, won't get the apt news.</p> <p>YouTube channels may be a solution but, they can't get all the news available.</p> |                           | Focus on J&P, tap into BE, understand RC |
|                                                                                                                                                                                                                                       | <b>3. TRIGGERS</b> <span>TR</span><br><p>Customer may install the app by seeing ads.</p> <p>Through friends' suggestion.</p>                                                                                                                                                                                                                         | <b>10. YOUR SOLUTION</b> <span>SL</span><br><p>User should login with their details in the application in which we are providing with databases to manage their interest.</p> <p>Easy to use</p> <p>Various languages and regional news will be there.</p> | <b>8. CHANNELS of BEHAVIOUR</b> <span>CH</span><br><p>8.1 ONLINE</p> <p>People can find the particular news and can get what they want</p> <p>8.2 OFFLINE</p> <p>People need to buy newspapers to read.</p>                                                 |                           |                                          |
| <b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span><br><p>Before installing they feel it difficult to read news</p> <p>After using, they feel comfortable and user-friendly to get what they want.</p> <p>It conserves their time.</p> |                                                                                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                            |                                                                                                                                                                                                                                                             |                           |                                          |

## 4. REQUIREMENT ANALYSIS

### 4.1 Functional requirement

Following are the functional requirements of the proposed solution.

| FR No. | Functional Requirement (Epic) | Sub Requirement (Story / Sub-Task)                                                                                                                             |
|--------|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| FR-1   | User Registration             | <ul style="list-style-type: none"><li>• Registration through online</li><li>• Registration through Gmail</li><li>• Registration through mobile phone</li></ul> |
| FR-2   | User installation             | <ul style="list-style-type: none"><li>• Application can download by play store</li><li>• Or it can be viewed through website</li></ul>                         |
| FR-2   | User Confirmation             | <ul style="list-style-type: none"><li>• Confirmation via Email</li><li>• Confirmation via OTP</li></ul>                                                        |
| FR-3   | User login                    | <ul style="list-style-type: none"><li>• Login through email</li><li>• Login through OTP</li><li>• Login by username and password</li></ul>                     |
| FR-4   | User information              | <ul style="list-style-type: none"><li>• It can pop up their preferred news article by setting the application settings</li></ul>                               |

## 4.2 Non-Functional requirements

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

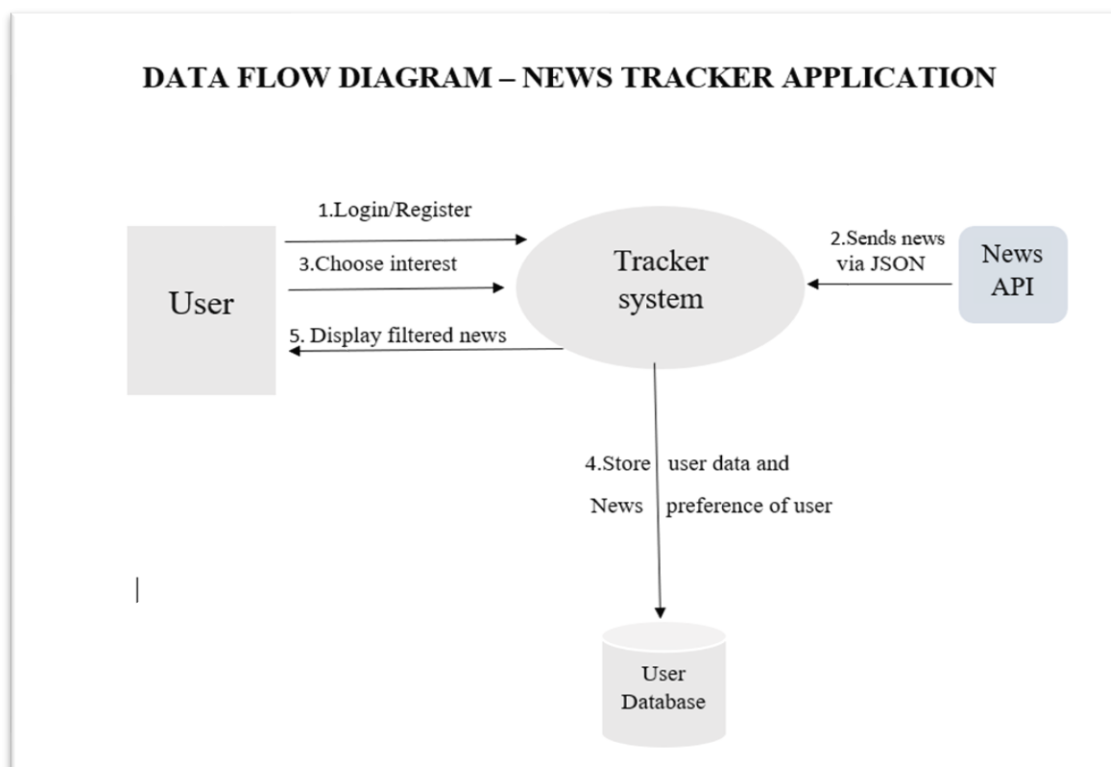
| FR No. | Non-Functional Requirement | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------|----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NFR-1  | Usability                  | <ul style="list-style-type: none"><li>• The option of filtering the content based on different category should be incorporated in the app to provide the audience wide taste and sensibilities. So, the users can read news that matters to them.</li><li>• This will help the users to share news on various platforms such as Twitter and Facebook. This will not only give an amazing user experience and also will also increase the views.</li><li>• When a user is not online due to some reason, he/she should have to access to the internet. Whenever the user is online the news content is downloaded in the cache memory of the app, this is how a user can access to the content offline.</li></ul> |
| NFR-2  | Security                   | <ul style="list-style-type: none"><li>• Authorised person can only manage database which contains the news article</li><li>• The information about the users is secured safely.</li><li>• People cannot watch news who are not registered with the application.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| NFR-3  | Reliability                | <ul style="list-style-type: none"><li>• This application publishes only the reliable content only with the help of API</li><li>• How much time it takes to solve an issue</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NFR-4 | <b>Performance</b> <ul style="list-style-type: none"> <li>• Application performance refers to how well an app runs on a mobile device under various loads and circumstances.</li> <li>• It might require measuring how long an app takes to load, mitigating crashes during peaks in user activity, or monitoring battery usage.</li> <li>• App performance is an essential component of app development.</li> <li>• Potential users won't just uninstall an app when annoyed; they will likely seek a competitor to address their needs.</li> </ul> |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

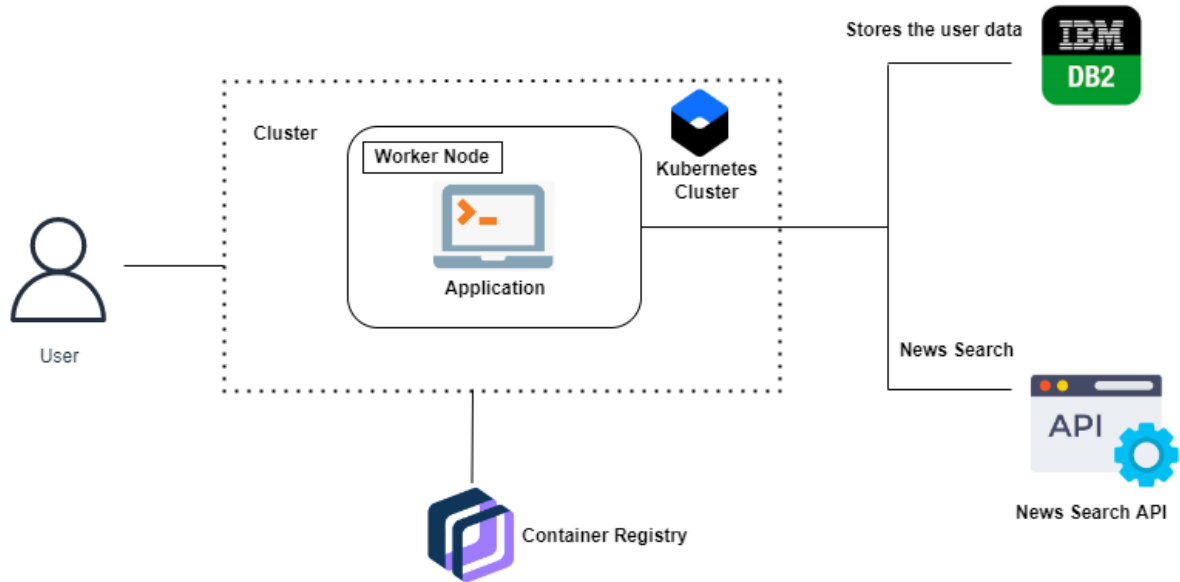
## 5. PROJECT DESIGN

### 5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clean DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system. What changes the information and where data is stored.



## 5.2 Solution & Technical Architecture



## 5.3 User Stories

| User Type                  | Functional Requirement (Epic) | User Story Number | User Story/Task                                                                                       | Acceptance Criteria                | Priority | Release  |
|----------------------------|-------------------------------|-------------------|-------------------------------------------------------------------------------------------------------|------------------------------------|----------|----------|
| Customer (Mobile/web user) | Registration                  | USN-1             | As a user, I can register for the application by entering my email, password and confirm my password. | I can access my account/ dashboard | High     | Sprint-1 |

|  |           |       |                                                                                                        |                                                    |        |          |
|--|-----------|-------|--------------------------------------------------------------------------------------------------------|----------------------------------------------------|--------|----------|
|  |           | USN-2 | As a user, I will receive confirmation email once I have registered for the application through Gmail. | I can receive confirmation email & click confirm   | High   | Sprint-1 |
|  |           | USN-3 | As a user, I can register for the application through Gmail.                                           |                                                    | Medium | Sprint-1 |
|  | Login     | USN-4 | As a user, I can log into the application by entering email & password.                                |                                                    | High   | Sprint-1 |
|  | Dashboard | USN-5 | As a user, I can enter the interests & choices of news I want to see for the first time in dashboard.  |                                                    | High   | Sprint-2 |
|  |           | USN-6 | As a user I can go through the feed of news filtered according to my wish.                             |                                                    | High   | Sprint-3 |
|  |           | USN-7 | As a user, I can logout my account in settings.                                                        | I can click confirm to log out and end the session | Medium | Sprint-3 |

|                         |                          |       |                                                                       |  |        |          |
|-------------------------|--------------------------|-------|-----------------------------------------------------------------------|--|--------|----------|
|                         | Settings                 | USN-8 | As a user, I can update my interests and choices in account settings. |  | Medium | Sprint-4 |
| Customer Care Executive | Chat Bot / Query Section | USN-9 | Solve issues brought up by client.                                    |  | Medium | Sprint-4 |
| Admin                   |                          | USN10 | Roll out updates and bug fixes.                                       |  | High   | Sprint-4 |

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation

| TITLE                                       | DESCRIPTION                                                                                                                      | DATE              |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|-------------------|
| Literature Survey and Information Gathering | Literature survey on the selected project & gathering information by referring the technical papers, research publications, etc. | 3 September 2022  |
| Prepare Empathy Map                         | Prepare Empathy Map Canvas to capture the user pains & gains, prepare the list of problem statements.                            | 10 September 2022 |
| Ideation Brain Storming                     | List by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility and importance.             | 16 September 2022 |



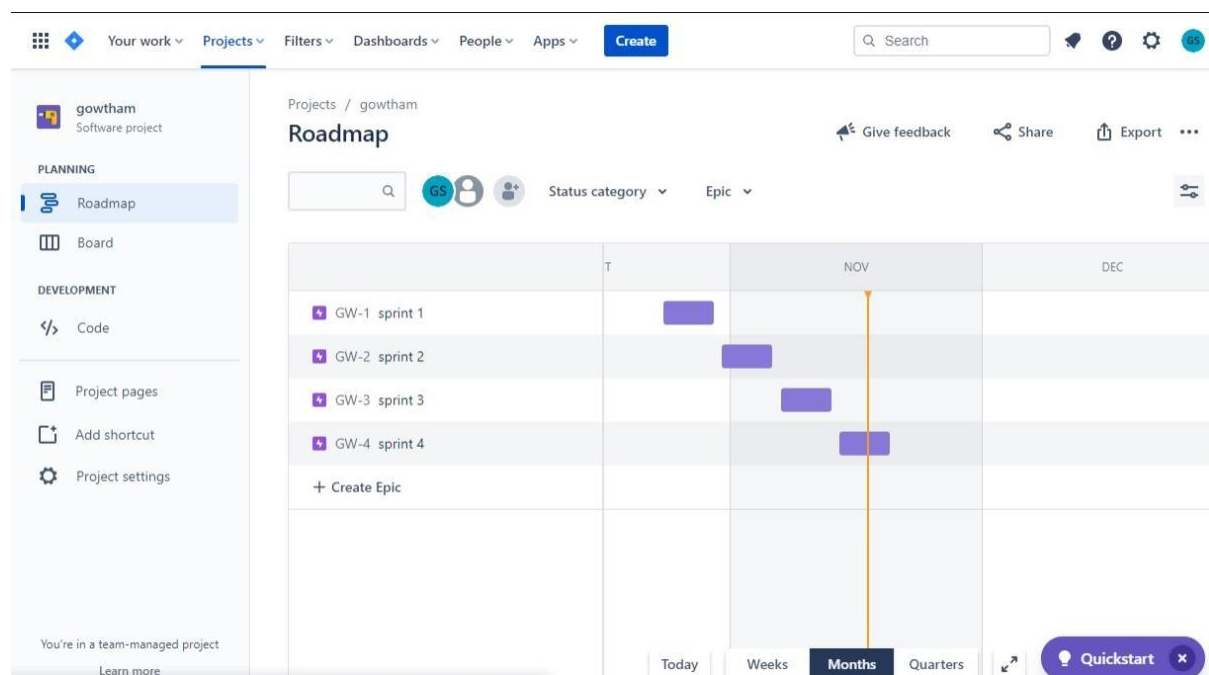
|                                                     |                                                                                                                                                       |                   |
|-----------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| Proposed Solution                                   | Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc. | 24 September 2022 |
| Problem Solution Fit                                | Prepare problem solution fit document.                                                                                                                | 28 September 2022 |
| Solution Architecture                               | Prepare solution architecture document.                                                                                                               | 6 October 2022    |
| Customer Journey                                    | Prepare the customer journey maps, understand the user interactions and experiences with the application.                                             | 8 October 2022    |
| Solution Requirement                                | Prepare the functional requirement document.                                                                                                          | 10 October 2022   |
| Data Flow Diagram                                   | Draw the data flow diagrams and submit for review.                                                                                                    | 15 October 2022   |
| Technology Architecture                             | Prepare the technology architecture diagram.                                                                                                          | 20 October 2022   |
| Prepare Milestone & Activity List                   | Prepare the milestones and activity list of the project.                                                                                              | 28 October 2022   |
| Project Development<br>Delivery of Sprint-1,2,3 & 4 | Develop and submit the developed code by testing it.                                                                                                  | 19 October 2022   |

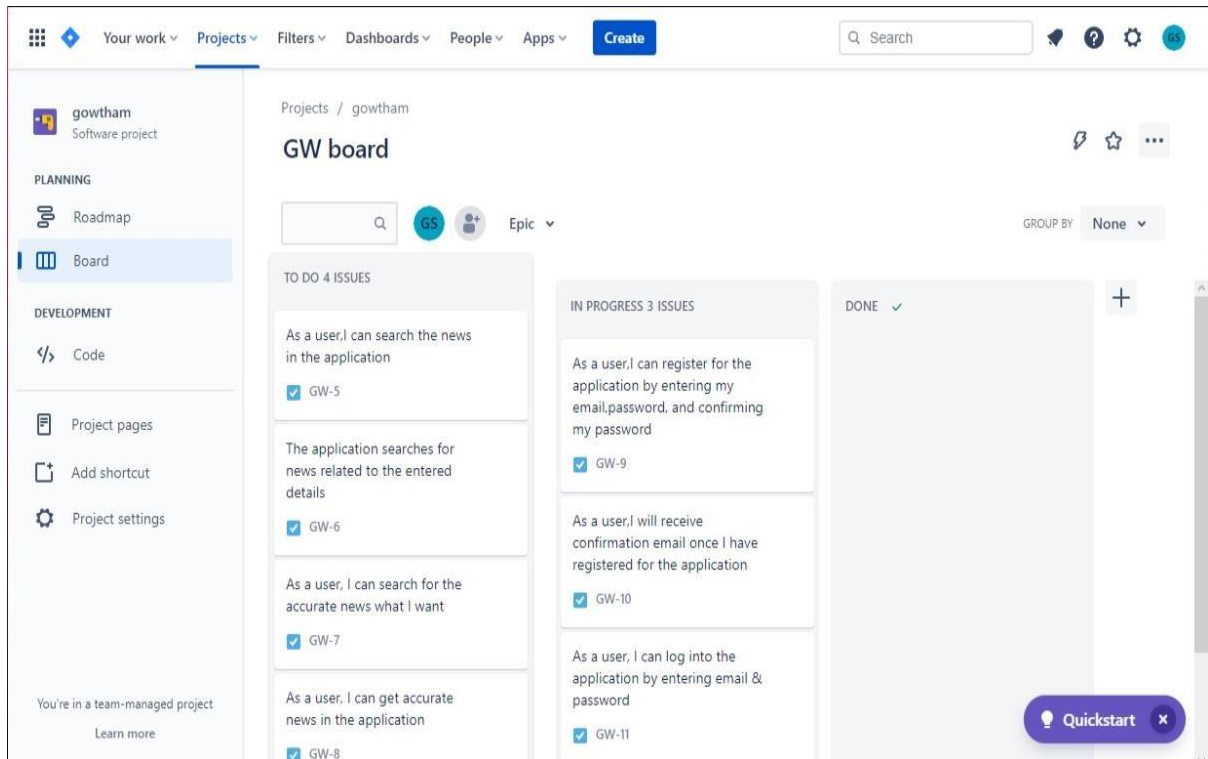
## 6.2 Sprint Delivery Schedule

| Sprint   | Functional Requirement (Epic)    | User Story Number | User Story / Task                                                                                         | Story Points | Priority | Team Members |
|----------|----------------------------------|-------------------|-----------------------------------------------------------------------------------------------------------|--------------|----------|--------------|
| Sprint-1 | Registration                     | USN-1             | As a user, I can register for the application by entering my email, password, and confirming my password. | 2            | High     | Kamalesh     |
| Sprint-1 |                                  | USN-2             | As a user, I will receive confirmation email once I have registered for the application                   | 1            | High     | Kamalesh     |
| Sprint-1 |                                  | USN-3             | As a user, I can register for the application through Facebook                                            | 2            | Low      | Latha        |
| Sprint-1 |                                  | USN-4             | As a user, I can register for the application through Gmail                                               | 2            | Medium   | Latha        |
| Sprint-1 | Login                            | USN-5             | As a user, I can log into the application by entering email & password                                    | 1            | High     | Kamalesh     |
| Sprint-2 | Dashboard                        | USN-6             | As a user I should be able to navigate<br>And access all the features hassle free                         | 2            | High     | Lavanya      |
| Sprint-2 | Layout                           | USN-7             | As a user I should be able to access the portal with different devices with the same comfort              | 2            | High     | Lavanya      |
| Sprint-3 | Data Store and retrieval         | USN-8             | Get Data from API and store as JSON in DB2                                                                | 3            | High     | Gowtham      |
| Sprint-3 |                                  | USN-9             | Get bin data from API and store in DFS                                                                    | 2            | High     | Gowtham      |
| Sprint-4 | User Segregation and data access | USN-10            | As a CC executive I should be able to uniquely identify the customer and offer help                       | 1            | Low      | Latha        |
| Sprint-4 | Change code                      | USN-11            | As a administrator I should be able to modify code according to the future requirements.                  | 2            | Medium   | Kamalesh     |
| Sprint-4 | Monitor the system               | USN-12            | As a administrator I should be able to monitor the cloud system and fix errors before customer.           | 1            | High     | Latha        |

| 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 | 20                 | 6 Days   | 24 Oct 2022       | 29 Oct 2022               | 8                                               | 29 Oct 2022                  |
| Sprint-2 | 20                 | 6 Days   | 31 Oct 2022       | 05 Nov 2022               | 4                                               | 05 Nov 2022                  |
| Sprint-3 | 20                 | 6 Days   | 07 Nov 2022       | 12 Nov 2022               | 5                                               | 12 Nov 2022                  |
| Sprint-4 | 20                 | 6 Days   | 14 Nov 2022       | 19 Nov 2022               | 4                                               | 19 Nov 2022                  |

## 6.3 Reports from JIRA

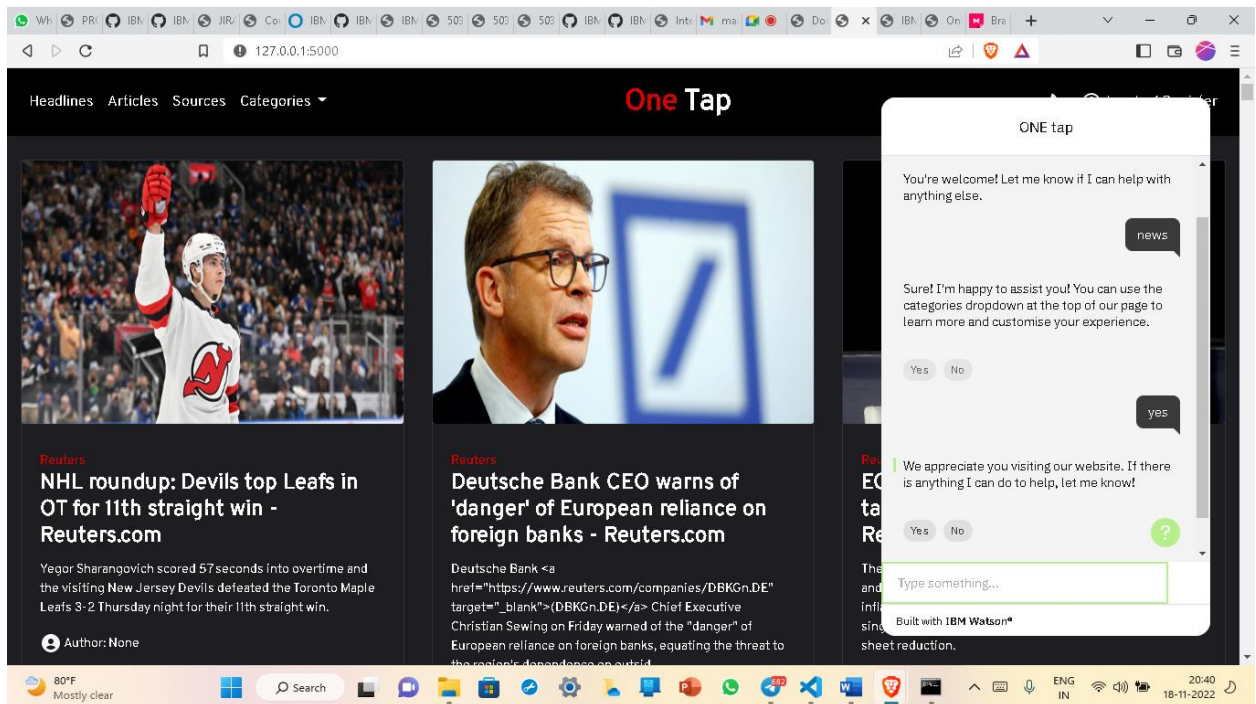




## 7. CODING & SOLUTIONING (Explain the features added in the project along with code)

### 7.1 Feature:

```
<script>
  window.watsonAssistantChatOptions = {
    integrationID: "fa18fb77-6c75-43f6-b1a1-a4f29fbbbc16", // The ID of
this integration.
    region: "eu-gb", // The region your integration is hosted in.
    serviceInstanceID: "87be378b-e186-46e4-9050-a97b5f524d5b", // The ID of
your service instance.
    onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');
    t.src="https://web-
chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>
```



## 8. TESTING

### 8.1 Test Cases

#### News Test Case:

```
import unittest
from ..models import Articles

class NewsTest(unittest.TestCase):
    def setUp(self):
        self.articles = Articles(
            "Techcrunch",
            "Steve Njuguna",
            "Nike acquires NFT collectibles studio RTFKT",
            "Nike is taking a plunge deeper into the world of crypto collectibles, announcing that they're acquiring the NFT studio RTFKT (pronounced 'artifact'). The acquisition announcement ...",
            "https://techcrunch.com/2021/12/13/nike-acquires-nft-collectibles-studio-rtfkt/",
            "https://techcrunch.com/wp-content/uploads/2021/12/FC3_itGXEEAA6z5g.jpg?w=1390&crop=1",
            "2021-12-13T20:19:00Z"
        )
```

```
def test_instance(self):
    self.assertTrue(isinstance(self.articles, Articles))

if __name__ == "__main__":
    unittest.main()
```

### **Source Test Case:**

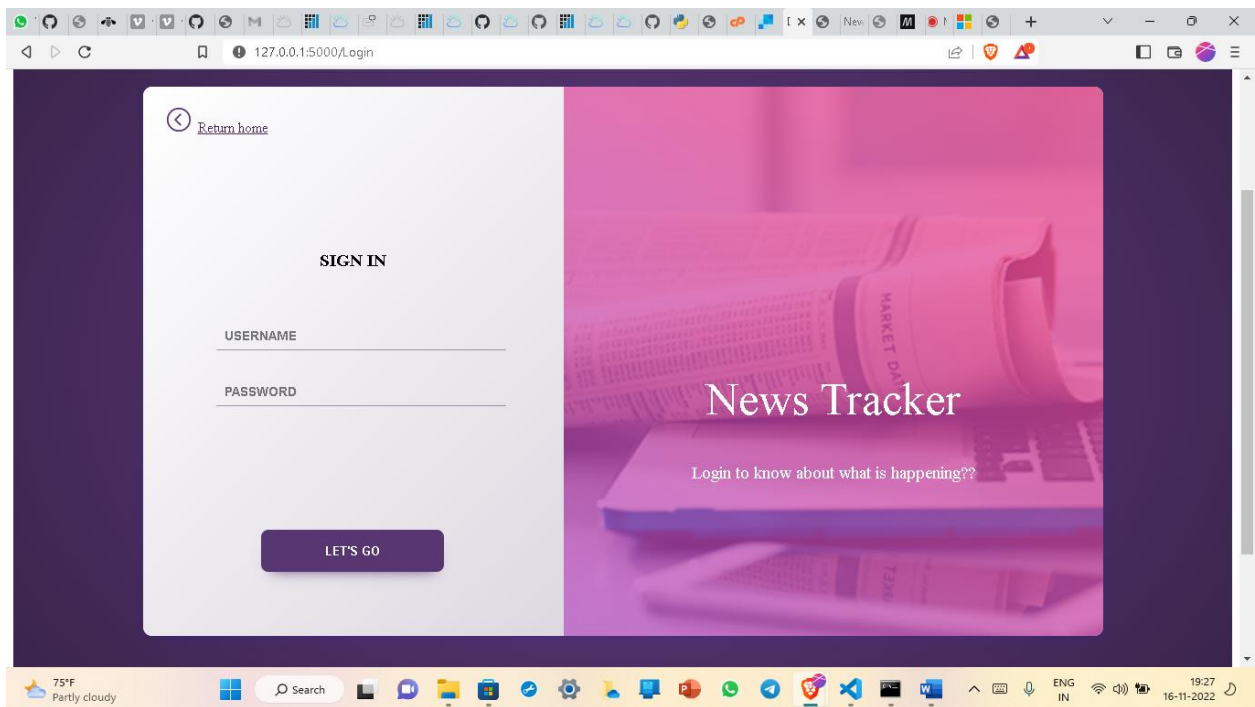
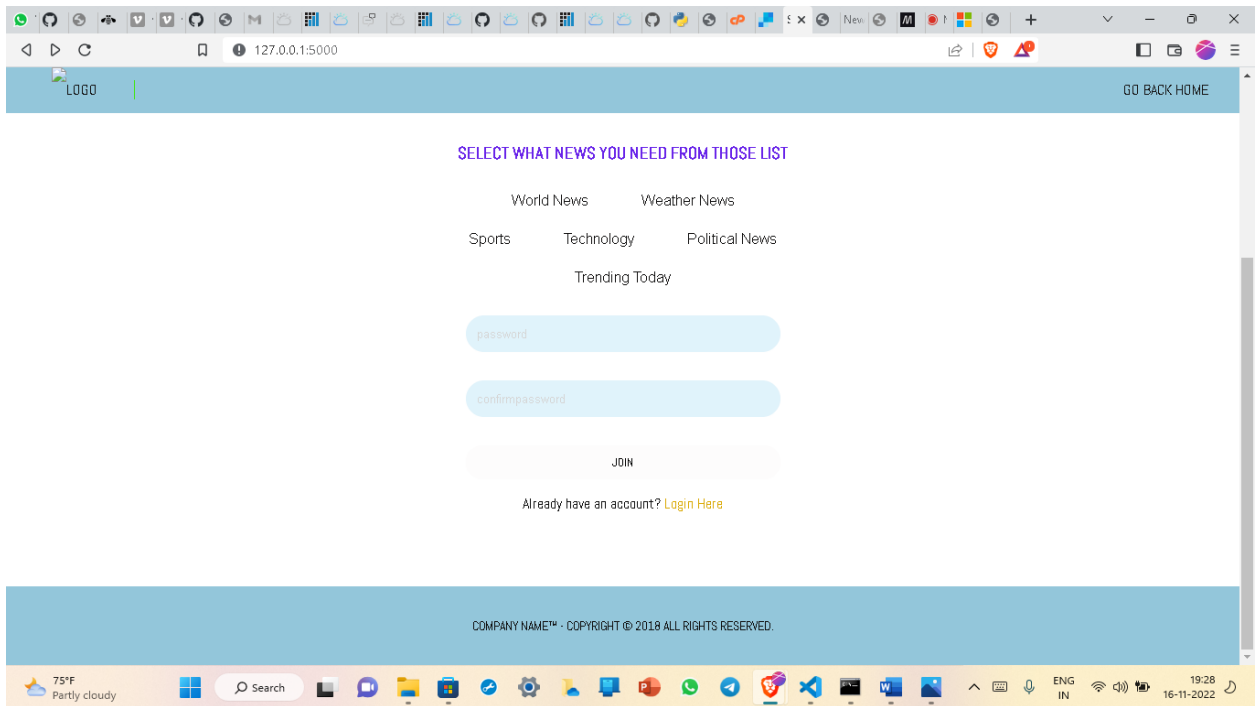
```
import unittest
from ..models import Sources

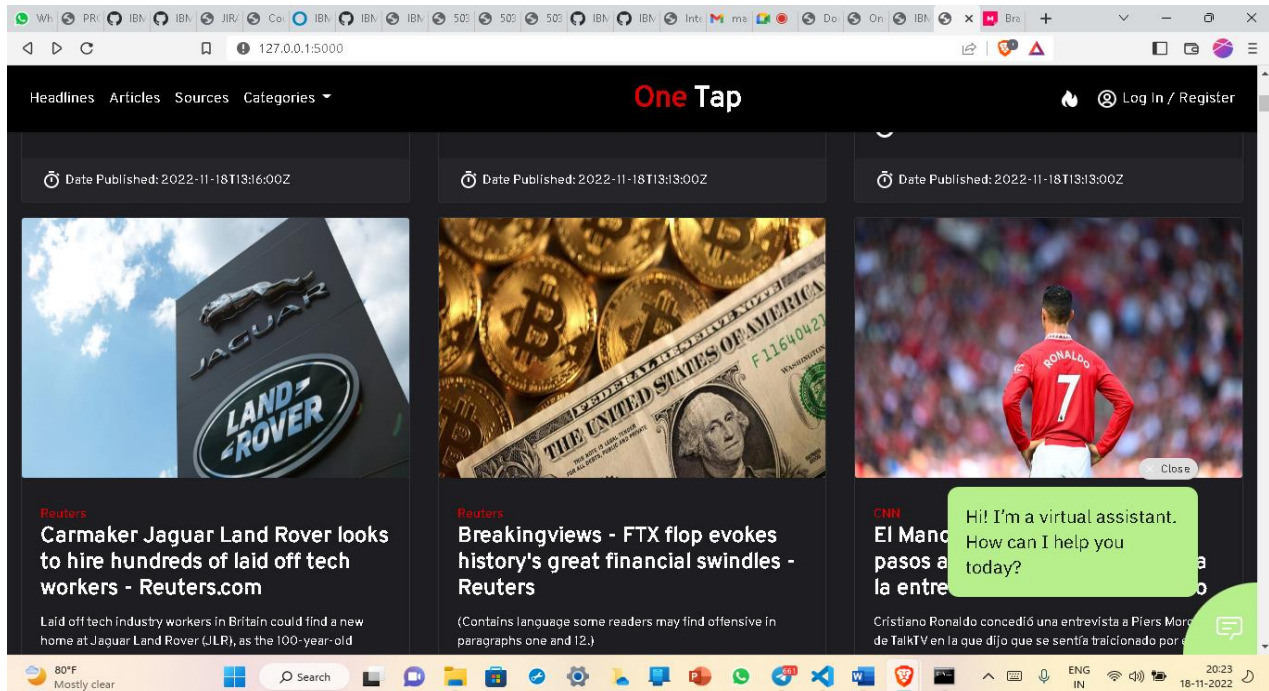
class SourceTest(unittest.TestCase):
    def setUp(self):
        self.source = Sources(
            "ABC News",
            "Your trusted source for breaking news, analysis, exclusive
interviews, headlines, and videos at ABCNews.com.",
            "https://abcnews.go.com/"
        )

    def test_instance(self):
        self.assertTrue(isinstance(self.source, Sources))

if __name__ == "__main__":
    unittest.main()
```

## 8.2 User Acceptance Testing





## 9. RESULTS

News tracker application using cloud is developed and executed at the level of completed progress.

## 10.ADVANTAGES & DISADVANTAGES

### Advantages:

- It provides higher accuracy through cross validation.
- High stability compare to Existing system.
- It is an user-friendly application.
- To explore and discover trending news and topics.
- Easily accessible and portable.
- Better user experience.
- Minute by minute updates of news.



## Disadvantage:

- Occurrence of Advertisement disturb the user.
- Sometimes the news gives brief information.
- It works only through internet.
- Device fault may affect the application.
- Fake news may mislead the readers.

## 11.CONCLUSION:

We explored the feasibility of recognizing patterns of news reading interactions and evaluated three adaptive interface designs for different news reader types. We show that from their interaction log, a specific user can be recognized as one of three kinds. The reader types emerging from the online survey are well defined and distinct. The evaluation of the three variant interfaces suggests that different news reader types need different user interfaces. We have demonstrated a method for monitoring users' news reading behavior and inferring news reader type from it. In the future we will further explore the design of adaptive interfaces, in order to be in a position to demonstrate a complete adaptive mobile news framework providing automatic personalization of news apps.

## 12.FUTURE SCOPE:

In the future we will further explore the design of adaptive interfaces, in order to be in a position to demonstrate a complete adaptive mobile news framework providing automatic personalization of news apps.

## 13. APPENDIX

### Source Code

#### \_\_init\_\_.py:

```
from flask import Flask

app = Flask(__name__)

from app import views
```

## 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 = "367756a948444eb3becb814af0e4bcb8"

class ProdConfig(Config):
    pass

class DevConfig(Config):
    DEBUG = True

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

## 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
```

## request.py:

```
from .models import Articles
from .models import Sources
from newsapi.newsapi_client 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

```

## views.py:

```

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

@app.route('/')
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)

@app.route('/Login')
def Login():
    return render_template('Login.html')

@app.route('/Signup')
def Signup():
    return render_template('Signup.html')
```

### main.py:

```
from app import app

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

### news\_test.py:

```
import unittest
from ..models import Articles

class NewsTest(unittest.TestCase):
    def setUp(self):
        self.articles = Articles(
            "Techcrunch",
            "Steve Njuguna",
            "Nike acquires NFT collectibles studio RTFKT",
            "Nike is taking a plunge deeper into the world of crypto collectibles, announcing that they're acquiring the NFT studio RTFKT (pronounced 'artifact'). The acquisition announcement ...",
            "https://techcrunch.com/2021/12/13/nike-acquires-nft-collectibles-studio-rtfkt/",
            "https://techcrunch.com/wp-content/uploads/2021/12/FC3_itGXEEAA6z5g.jpg?w=1390&crop=1",
            "2021-12-13T20:19:00Z"
        )

    def test_instance(self):
        self.assertTrue(isinstance(self.articles, Articles))

if __name__ == "__main__":
    unittest.main()
```

### source\_test.py:

```
import unittest
from ..models import Sources

class SourceTest(unittest.TestCase):
    def setUp(self):
        self.source = Sources(
            "ABC News",
            "Your trusted source for breaking news, analysis, exclusive
interviews, headlines, and videos at ABCNews.com.",
            "https://abcnews.go.com/"
        )

    def test_instance(self):
        self.assertTrue(isinstance(self.source, Sources))

if __name__ == "__main__":
    unittest.main()
```

### articles.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
    <div class="row">
        {% for source, title, desc, author, img, p_date, url in random %}
        <div class="col-md-4 d-flex justify-content-center">
            <div class="card mb-3 bg-dark text-white">
                
                <div class="card-body">
                    <p class="card-text source">{{ source.name }}</p>
                    <a href="{{ url }}">
                        <h2 class="card-title"><b> {{ title }} </b></h2>
                    </a>
                    <p class="card-text">{{ desc }}</p>
                    <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10-
4.579 10-10s17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
```

```

4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
    </div>
    <div class="card-footer">
        <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8-3.589-8-8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
    </div>
</div>
</div>
    {% endfor %}
</div>
</div>
{% endblock %}

```

## Base.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <meta name="Description" content="Enter your description here"/>
    <!-- Favicons -->
    <link href="{{ url_for('static', filename='/favicon.ico') }}" rel="icon">

    <link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Overpass">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.6.0/css/bootstrap.min.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css">
    <link rel="stylesheet" href="{{ url_for('static',
filename='assets/css/style.css') }}">
    <title>One Tap</title>
</head>
<body>

    {% block navbar %}

```

```

<div class="container-fluid" style="background-color: #000;">
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top shadow-sm">

    <!-- Show this only on mobile to medium screens -->
    <a class="navbar-brand d-lg-none" href="{{ url_for('home') }}"><b
class="lg"><span class="logo">One</span> Tap</b></a>

    <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarToggle" aria-controls="navbarToggle" aria-expanded="false"
aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>

    <!-- Use flexbox utility classes to change how the child elements
are justified -->
    <div class="collapse navbar-collapse justify-content-between"
id="navbarToggle">

      <ul class="navbar-nav">
        <li class="nav-item">
          <a class="nav-link" href="{{ url_for('headlines')
}}">Headlines</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ url_for('articles')
}}">Articles</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="{{ url_for('sources')
}}">Sources</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" href="#"
id="navbarDropdown" role="button" data-toggle="dropdown" aria-expanded="false">
            Categories
          </a>
          <div class="dropdown-menu" aria-
labelledby="navbarDropdown">
            <a class="dropdown-item" href="{{ url_for('business')
}}">Business</a>
            <a class="dropdown-item" href="{{ url_for('tech')
}}">Technology</a>
          </div>
        </li>
      </ul>
    </div>
  </nav>
</div>

```



```

        <a class="dropdown-item" href="{ {
url_for('entertainment') }}">Entertainment</a>
        <a class="dropdown-item" href="{ { url_for('science')
}}">Science</a>
        <a class="dropdown-item" href="{ { url_for('sports')
}}">Sport</a>
        <a class="dropdown-item" href="{ { url_for('health')
}}">Health</a>
    </div>
</li>
</ul>

<!-- Show this only lg screens and up -->
    <a class="navbar-brand d-none d-lg-block" href="{ {
url_for('home') }}"><b class="lg"><span class="logo">One</span> Tap</b></a>

    <ul class="navbar-nav">
        <li class="nav-item">
            <a class="nav-link" href="/headlines">
                <svg class="svg" xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M16.5,8c0,1.5-0.5,3.5-2.9,4.3c0.7-1.7,0.8-
3.4,0.3-5c-0.7-2.1-3-3.7-4.6-4.6C8.9,2.4,8.2,2.8,8.3,3.4c0,1.1-0.3,2.7-
2,4.4 C4.1,10,3,12.3,3,14.5C3,17.4,5,21,9,21c-4-4-1-7.5-1-
7.5c0.8,5.9,5,7.5,7,7.5c1.7,0,5-1.2,5-6.4c0-3.1-1.3-5.5-2.4-
6.9 C17.3,7.2,16.6,7.5,16.5,8"></path></svg>
            </a>
        </li>
        <li class="nav-item">
            <a class="nav-link" href="/Login">
                <svg class="svg" xmlns="http://www.w3.org/2000/svg"
viewBox="0 0 24 24" width="24" height="24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2A10.13 10.13 0 0 0 2 12a10 10 0 0 0 4
7.92V20h.1a9.7 9.7 0 0 0 11.8 0h.1v-.08A10 10 0 0 0 22 12 10.13 10.13 0 0 12
2zM8.07 18.93A3 3 0 0 1 11 16.57h2a3 3 0 0 1 2.93 2.36 7.75 7.75 0 0 1-7.86
0zm9.54-1.29A5 5 0 0 0 13 14.57h-2a5 5 0 0 0-4.61 3.07A8 8 0 0 1 4 12a8.1 8.1 0 0
1 8-8 8.1 8.1 0 0 1 8 8 8 8 0 0 1-2.39 5.64z"></path><path d="M12 6a3.91 3.91 0 0
0-4 4 3.91 3.91 0 0 0 4 4 3.91 3.91 0 0 0 4-4 3.91 3.91 0 0 0-4-4zm0 6a1.91 1.91
0 0 1-2-2 1.91 1.91 0 0 1 2-2 1.91 1.91 0 0 1 2 2 1.91 1.91 0 0 1-2
2z"></path></svg> Log In / Register
            </a>
        </li>

```

```

        </ul>
    </div>
</nav>
</div>

{% endblock %}

{% block content %}

{% endblock %}

{% block footer %}

<div class="container-fluid footer">
    <p class="text-center footer-text">
        © <span class="logo">News</span> NEWS TRACKER APPLICATION
    </p>
</div>

{% endblock %}

<script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.5.1/jquery.slim.min.js"></sc
ript>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.1/umd/popper.min.js"><
/script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.6.0/js/bootstrap.min.js"></script>
<script>
    window.watsonAssistantChatOptions = {
        integrationID: "fa18fb77-6c75-43f6-b1a1-a4f29fbbbc16", // The ID of
this integration.
        region: "eu-gb", // The region your integration is hosted in.
        serviceInstanceID: "87be378b-e186-46e4-9050-a97b5f524d5b", // The ID of
your service instance.
        onLoad: function(instance) { instance.render(); }
    };
    setTimeout(function(){
        const t=document.createElement('script');
        t.src="https://web-
chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
        document.head.appendChild(t);

```

## Business.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
    <div class="row">
        {% for source, title, desc, author, img, p_date, url in sources %}
        <div class="col-md-4 d-flex justify-content-center">
            <div class="card mb-3 bg-dark text-white">
                
                <div class="card-body">
                    <p class="card-text source">{{ source.name }}</p>
                    <a href="{{ url }}">
                        <h2 class="card-title"><b> {{ title }} </b></h2>
                    </a>
                    <p class="card-text">{{ desc }}</p>
                    <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:;"><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10-
4.579 10-10S17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
                </div>
                <div class="card-footer">
                    <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:;"><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8 8-3.589-8-8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
                </div>
            </div>
        </div>
        {% endfor %}
    </div>
</div>
```

```
{% endblock %}
```

## Entertainment.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source, title, desc, author, img, p_date, url in sources %}
    <div class="col-md-4 d-flex justify-content-center">
      <div class="card mb-3 bg-dark text-white">
        
        <div class="card-body">
          <p class="card-text source">{{ source.name }}</p>
          <a href="{{ url }}">
            <h2 class="card-title"><b> {{ title }} </b></h2>
          </a>
          <p class="card-text">{{ desc }}</p>
          <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10 10-
4.579 10-10s17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
        </div>
        <div class="card-footer">
          <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8 3.589-8-8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
        </div>
      </div>
    </div>
    </div>
    </div>
    {% endfor %}
  </div>
</div>

{% endblock %}
```

## Headlines.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source, title, desc, author, img, p_date, url in headlines %}
    <div class="col-md-4 d-flex justify-content-center">
      <div class="card mb-3 bg-dark text-white">
        
        <div class="card-body">
          <p class="card-text source">{{ source.name }}</p>
          <a href="{{ url }}">
            <h2 class="card-title"><b> {{ title }} </b></h2>
          </a>
          <p class="card-text">{{ desc }}</p>
          <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10-
4.579 10-10S17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
          </div>
          <div class="card-footer">
            <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8 8-3.589-8-8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 2.691 6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>
</div>

{% endblock %}
```

## Health.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source, title, desc, author, img, p_date, url in sources %}
    <div class="col-md-4 d-flex justify-content-center">
      <div class="card mb-3 bg-dark text-white">
        
        <div class="card-body">
          <p class="card-text source">{{ source.name }}</p>
          <a href="{{ url }}">
            <h2 class="card-title"><b> {{ title }} </b></h2>
          </a>
          <p class="card-text">{{ desc }}</p>
          <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10-
4.579 10-10S17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
          </div>
          <div class="card-footer">
            <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8 8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>

{% endblock %}
```

## Home.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source, title, desc, author, img, p_date, url in articles %}
      <div class="col-md-4 d-flex justify-content-center">
        <div class="card mb-3 bg-dark text-white">
          
          <div class="card-body">
            <p class="card-text source">{{ source.name }}</p>
            <a href="{{ url }}">
              <h2 class="card-title"><b> {{ title }} </b></h2>
            </a>
            <p class="card-text">{{ desc }}</p>
            <p class="card-text"><svg
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 2C6.579
2 2 6.579 2 12s4.579 10 10 10 4.579 10-10s17.421 2 12 2zm0 5c1.727 0 3 1.272 3
3s-1.273 3-3 3c-1.726 0-3-1.272-3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2
4.106-2.2h2c1.714 0 3.209.88 4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-
5.106-2.228z"></path></svg> Author: {{ author }}</p>
          </div>
          <div class="card-footer">
            <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8 8-3.589 8-8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>
</div>

{% endblock %}
```

## Science.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source, title, desc, author, img, p_date, url in sources %}
    <div class="col-md-4 d-flex justify-content-center">
      <div class="card mb-3 bg-dark text-white">
        
        <div class="card-body">
          <p class="card-text source">{{ source.name }}</p>
          <a href="{{ url }}">
            <h2 class="card-title"><b> {{ title }} </b></h2>
          </a>
          <p class="card-text">{{ desc }}</p>
          <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10-
4.579 10-10S17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
          </div>
          <div class="card-footer">
            <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8 8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>

{% endblock %}
```



## Source.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source in newsSource %}
      <div class="col-md-3 d-flex justify-content-center source-body">
        <div class="card bg-dark">
          <div class="card-body">
            <a href="{{ source.url }}" target="_blank">
              <h5 class="card-title"><svg
xmlns="http://www.w3.org/2000/svg" width="36" height="36" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 2C6.486
2 2 6.486 2 12s4.486 10 10 10s17.514 2 12 2zM4 12c0-.899.156-
1.762.431-2.569L6 11L2 2v2L2 2 1v1.931C7.061 19.436 4 16.072 4 12zm14.33
4.873C17.677 16.347 16.687 16 16 16v-1a2 2 0 0 0-2-2h-4v-3a2 2 0 0 0 2-2V7h1a2 2
0 0 0 2-2v-.411C17.928 5.778 20 8.65 20 12a7.947 7.947 0 0 1-1.67
4.873z"></path></svg> {{ source.name }}</h5>
              <p class="card-text">{{ source.description }}</p>
            </a>
          </div>
        </div>
      </div>
    {% endfor %}
  </div>
{% endblock %}
```

### Sport.html:

```
{% extends 'base.html' %}

{% block content %}

<div class="container-fluid landing">
  <div class="row">
    {% for source, title, desc, author, img, p_date, url in sources %}
    <div class="col-md-4 d-flex justify-content-center">
      <div class="card mb-3 bg-dark text-white">
        
        <div class="card-body">
```

```

        <p class="card-text source">{{ source.name }}</p>
        <a href="{{ url }}">
            <h2 class="card-title"><b> {{ title }} </b></h2>
        </a>
        <p class="card-text">{{ desc }}</p>
        <p class="card-text"><svg xmlns="http://www.w3.org/2000/svg"
width="24" height="24" viewBox="0 0 24 24" style="fill: rgba(255, 255, 255,
1);transform: ;msFilter:; "><path d="M12 2C6.579 2 2 6.579 2 12s4.579 10 10 10-
4.579 10-10S17.421 2 12 2zm0 5c1.727 0 3 1.272 3 3s-1.273 3-3 3c-1.726 0-3-1.272-
3-3s1.274-3 3-3zm-5.106 9.772c.897-1.32 2.393-2.2 4.106-2.2h2c1.714 0 3.209.88
4.106 2.2C15.828 18.14 14.015 19 12 19s-3.828-.86-5.106-2.228z"></path></svg>
Author: {{ author }}</p>
    </div>
    <div class="card-footer">
        <p class="card-text published"><svg class="p-svg"
xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"
style="fill: rgba(255, 255, 255, 1);transform: ;msFilter:; "><path d="M12 5c-4.411
0-8 3.589-8 8s3.589 8 8 8-3.589 8-8-3.589-8-8-8zm0 14c-3.309 0-6-2.691-6-
6s2.691-6 6-6 6-2.691 6-6 6z"></path><path d="M11 9h2v5h-2zM9
2h6v2H9zm10.293 5.707-2-2 1.414-1.414 2 2z"></path></svg> Date Published: {{
p_date }}</p>
    </div>
</div>
    </div>
    <div>
        {{% endfor %}}
    </div>
</div>
{{% endblock %}}

```

## Signup.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Signup</title>
    <link rel="Stylesheet" href="{{ url_for('static', filename='Signup.css') }}">
</head>
<body>
    <nav>
        <ul class="navigation">

```

```

        <div class="brand">
<!-- Making menu icon clickable to display the navigation menu on smaller screens
-->
        <i onclick="navToggle()" id="nav-icon" class="fa fa-navicon"
style="font-size:24px"></i>
    </div>

    <div id="toggle" class="nav-container">
        <a class="left" href="/"><li>GO BACK HOME</li></a>
    </div>
</ul>
</nav>
<!-- NAVIGATION END HERE -->

<section class="form">
<div class="center">
    <h1>JOIN OUR <b style="color: #7a44dd ;">NEWS TRACKING APPLICATION</b> TO
KNOW ABOUT THE NEWS</h1>
    <hr width="20%" style="border: 1px solid #6a29e4;">
    <br>

    <form action="">
        <input class="name-surname" type="text" name="name"
placeholder="firstname">
        <input class="name-surname" type="text" name="surname"
placeholder="lastname"><br>
        <input type="text" name="email" placeholder="emailid"><br>
        <br> <input type="password" name="password" placeholder="password"><br>
        <input type="password" name="conf_password"
placeholder="confirmpassword"><br>

        <button formaction="/">JOIN</button>
        <p>Already have an account? <a href="/Login">Login Here</a></p>
    </form>
</div>

</section>

<!-- FOOTER STARTS HERE -->

    <footer>
        <div class="footer_container">
            <p class="bottom_text">ONE TAP™ - Copyright © 2018 All Rights
Reserved.</p>

```

```

        </div>
    </footer>

<!-- FOOTER ENDS HERE -->
</body>
</html>

```

## Login.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="Stylesheet" href="{ url_for('static', filename='Login.css') }">
</head>
<body>

    <section class="login">
        <div class="signup">
            <form action="">
                <h5>Click here to create a new Account</h5>
                <button formaction="/Signup">SignUp</button>
            </form>
        </div>
        <div class="login_box">
            <div class="left">
                <div class="top_link"><a href="/">Return home</a></div>
                <div class="contact">
                    <form action="">
                        <h3>SIGN IN</h3>
                        <input type="text" placeholder="USERNAME">
                        <input type="password" placeholder="PASSWORD">
                        <button class="submit" formaction="/">LET'S GO</button>
                    </form>
                </div>
            </div>
            <div class="right">
                <div class="right-text">
                    <h2>News Tracker</h2>
                    <h5>Login to know about what is happening??</h5>

```

```
        </div>

    </div>
</section>
</body>
</html>
```

## Signup.css:

```
@import url('https://fonts.googleapis.com/css?family=Abel');

body {
    background-color: #ffffff;
    font-family: 'Abel', sans-serif;
    color: #333;
    margin: 0;
}

a {
    color: #daa800;
    text-decoration: none;
    transition: 1s;
}

a:hover {
    text-decoration: none;
    color: #daa800;
    opacity: 0.8;
    transition: 1s;
}

p {
    color: #0f0f0f;
}

h1 {
    font-weight: normal;
}

/*
=====
*/

section.form {
```

```
    opacity: 1;
    background-color: transparent;
    padding: 20px;
    text-align: center;
    color: #161515;
}

.center {
    margin-right: auto;
    margin-left: auto;
    display: block;
    width: 400px;
    margin-top: 100px;
    margin-bottom: 60px;
}

input {
    opacity: 0.3;
    border-radius: 20px;
    padding: 10px;
    width: 80%;
    display: block;
    margin-right: auto;
    margin-left: auto;
    margin-bottom: 10px;
    box-shadow: none;
    color: rgb(24, 22, 22);
    border: none;
    background-color: #97d6f3;
    transition: 1s;
    border: 2px solid transparent;
}

input:focus, input:hover {
    transition: 1s;
    border: 2px solid #5d5d5d;
    background-color: transparent;
}

.name-surname {
    width: 36%;
    display: inline;
}

.checkbox{
    display: flex;
```

```
    flex-wrap: wrap;
    align-items: center;
    justify-content: center;
}
input[type="checkbox"]{
    -webkit-appearance: none;
    appearance: none;
    display: none;
}
label{
    position: relative;
    display: block;
    padding: 2px 25px;
    border-radius: 20px;
    text-align: center;
    line-height: 30px;
    color: #0e0d0d;
    margin: 4px;
    font-family: 'poppins', sans-serif;
    transition: .2s;
    background-color: transparent;
}
label::after{
    content: "";
    position: absolute;
    top: -2px;
    left: -2px;
    width: 100%;
    height: 100%;
    z-index: -1;
    border: 2px solid orange;
    border-radius: 20px;
    transform: scale(.9);
    opacity: 0;
    transition: .2s;
}
input:checked + label::after{
    border-color: #daa800;
    opacity: 1;
    transform: scale(1);
}
input:checked + label{
    color: #daa800;
    border: none;
    background: none;
```

```
}  
button {  
    width: 86%;  
    border-radius: 20px;  
    background-color: #fdfcfc;  
    margin-left: auto;  
    margin-right: auto;  
    display: block;  
    box-shadow: none;  
    text-shadow: none;  
    border: none;  
    padding: 10px;  
    font-family: 'Abel', sans-serif;  
    transition: 1s;  
}
```

```
button:hover {  
    background-color: #daa800;  
    transition: 1s;  
}
```

```
/*
```

```
===== */
```

```
/* FOOTER STARTS HERE */
```

```
footer {  
    background-color: #93c6da;  
    color: #fff;  
    width: 100%;  
}
```

```
.footer_container {  
    padding: 20px;  
    text-align: center;  
    float: none;  
}
```

```
.footer_container .bottom_text {  
    font-size: 14px;  
    text-transform: uppercase;  
}
```

```
.social_icons {  
    margin-left: auto;
```



```
margin-right: auto;
display: block;
}
/* FOOTER ENDS HERE */

/* NAVIGATION STARTS HERE */
.navigation {
margin: 0;
}

.brand { /* Your brand logo */
border-right: 0.5px solid #40e60e;
font-size: 16px;
line-height: 50px;
display: inline;
margin-right: 30px;
padding-right: 40px;
}

.nav-container {
display: inline;
}

/* Added font-size: 0; to get rid of the space between lists */
nav {
width: 100%;
background-color: #93c6da;
margin: 0;
font-size: 0;
padding-left: 10px;
padding-right: 10px;
color: rgb(19, 18, 18);
position: fixed;
z-index: 1;
}

/* Declaring font-size for the anchor element to be 16px since it is 0 on the
above code */
nav a {
font-size: 16px;
}

ul {
display: inline;
margin: 0;
```

```
        overflow: hidden;
        list-style-type: none;
    }

    /* Adding background color to the li element and vertically centering the words
    in the li element */
    ul li {
        display: inline;
        margin: 0;
        padding: 14px;
        line-height: 50px;
        background-color: #1b1a1b;
        transition: 1s;
    }

    ul li:hover {
        transition: 1s;
        background-color: transparent;
    }

    ul a { /* Removing the 'underline' from the link */
        text-decoration: none;
    }

    ul a:hover { /* Removing the 'underline' from the link on the 'hover effect' */
        text-decoration: none;
    }

    ul a li {
        display: inline;
        text-decoration: none;
        color: rgb(22, 20, 20);
        margin: 0;
        text-transform: uppercase;
        background-color: transparent;
        transition: 1s;
    }

    ul a li:hover {
        transition: 1s;
        background-color: #87cde9;
        color: #daa800 ;
    }

    .left {
```

```
    float: right;
    margin-right: 30px;
}

#nav-icon {
    display: none;
    transition: 1s;
}

@media only screen and (max-width: 816px) {
    nav, .navigation {
        padding: 0;
    }

    .brand {
        line-height: 50px;
        border: none;
        margin-right: auto;
        margin-right: auto;
        display: block;
        width: 100%;
        padding-top: 7px;
        padding-bottom: 7px;
        padding-left: 20px;
        padding-right: 20px;
    }

    .left {
        float: none;
        margin-right: 0;
        transition: 1s;
    }

    nav ul li {
        display: block;
        text-align: center;
        width: 100%;
        transition: 1s;
        padding: 7px;
    }

    #nav-icon {
        display: inline;
        float: right;
        line-height: 50px;
    }
}
```

```
    }  
}  
  
/* NAVIGATION ENDS HERE */
```

## Login.css:

```
    @import url('https://fonts.googleapis.com/css?family=Abel');  
  
body {  
    background-color: #ffffff;  
    font-family: 'Abel', sans-serif;  
    color: #333;  
    margin: 0;  
}  
  
a {  
    color: #daa800;  
    text-decoration: none;  
    transition: 1s;  
}  
  
a:hover {  
    text-decoration: none;  
    color: #daa800;  
    opacity: 0.8;  
    transition: 1s;  
}  
  
p {  
    color: #0f0f0f;  
}  
  
h1 {  
    font-weight: normal;  
}  
  
/*  
=====
```

```
padding: 20px;
text-align: center;
color: #161515;
}

.center {
margin-right: auto;
margin-left: auto;
display: block;
width: 400px;
margin-top: 100px;
margin-bottom: 60px;
}

input {
opacity: 0.3;
border-radius: 20px;
padding: 10px;
width: 80%;
display: block;
margin-right: auto;
margin-left: auto;
margin-bottom: 10px;
box-shadow: none;
color: rgb(24, 22, 22);
border: none;
background-color: #97d6f3;
transition: 1s;
border: 2px solid transparent;
}

input:focus, input:hover {
transition: 1s;
border: 2px solid #5d5d5d;
background-color: transparent;
}

.name-surname {
width: 36%;
display: inline;
}

.checkbox{
display: flex;
flex-wrap: wrap;
align-items: center;
```

```
        justify-content: center;
    }
    input[type="checkbox"]{
        -webkit-appearance: none;
        appearance: none;
        display: none;
    }
    label{
        position: relative;
        display: block;
        padding: 2px 25px;
        border-radius: 20px;
        text-align: center;
        line-height: 30px;
        color:#0e0d0d;
        margin: 4px;
        font-family: 'poppins',sans-serif;
        transition: .2s;
        background-color: transparent;
    }
    label::after{
        content: "";
        position: absolute;
        top: -2px;
        left: -2px;
        width: 100%;
        height: 100%;
        z-index: -1;
        border: 2px solid orange;
        border-radius: 20px;
        transform: scale(.9);
        opacity: 0;
        transition: .2s;
    }
    input:checked + label::after{
        border-color: #daa800;
        opacity: 1;
        transform: scale(1);
    }
    input:checked + label{
        color: #daa800;
        border: none;
        background: none;
    }
    button {
```

```
width: 86%;
border-radius: 20px;
background-color: #fdfcfc;
margin-left: auto;
margin-right: auto;
display: block;
box-shadow: none;
text-shadow: none;
border: none;
padding: 10px;
font-family: 'Abel', sans-serif;
transition: 1s;
}
```

```
button:hover {
    background-color: #daa800;
    transition: 1s;
}
```

```
/*
```

```
=====
===== */
```

```
/* FOOTER STARTS HERE */
```

```
footer {
    background-color: #93c6da;
    color: #fff;
    width: 100%;
}
```

```
.footer_container {
    padding: 20px;
    text-align: center;
    float: none;
}
```

```
.footer_container .bottom_text {
    font-size: 14px;
    text-transform: uppercase;
}
```

```
.social_icons {
    margin-left: auto;
    margin-right: auto;
    display: block;
```

```
}
/* FOOTER ENDS HERE */

/* NAVIGATION STARTS HERE */
.navigation {
  margin: 0;
}

.brand { /* Your brand logo */
  border-right: 0.5px solid #40e60e;
  font-size: 16px;
  line-height: 50px;
  display: inline;
  margin-right: 30px;
  padding-right: 40px;
}

.nav-container {
  display: inline;
}

/* Added font-size: 0; to get rid of the space between lists */
nav {
  width: 100%;
  background-color: #93c6da;
  margin: 0;
  font-size: 0;
  padding-left: 10px;
  padding-right: 10px;
  color: rgb(19, 18, 18);
  position: fixed;
  z-index: 1;
}

/* Declaring font-size for the anchor element to be 16px since it is 0 on the
above code */
nav a {
  font-size: 16px;
}

ul {
  display: inline;
  margin: 0;
  overflow: hidden;
  list-style-type: none;
```



```
}

/* Adding background color to the li element and vertically centering the words
in the li element */
ul li {
    display: inline;
    margin: 0;
    padding: 14px;
    line-height: 50px;
    background-color: #1b1a1b;
    transition: 1s;
}

ul li:hover {
    transition: 1s;
    background-color: transparent;
}

ul a { /* Removing the 'underline' from the link */
    text-decoration: none;
}

ul a:hover { /* Removing the 'underline' from the link on the 'hover effect' */
    text-decoration: none;
}

ul a li {
    display: inline;
    text-decoration: none;
    color: rgb(22, 20, 20);
    margin: 0;
    text-transform: uppercase;
    background-color: transparent;
    transition: 1s;
}

ul a li:hover {
    transition: 1s;
    background-color: #87cde9;
    color: #daa800 ;
}

.left {
    float: right;
    margin-right: 30px;
}
```

```
}

#nav-icon {
  display: none;
  transition: 1s;
}

@media only screen and (max-width: 816px) {
  nav, .navigation {
    padding: 0;
  }

  .brand {
    line-height: 50px;
    border: none;
    margin-right: auto;
    margin-right: auto;
    display: block;
    width: 100%;
    padding-top: 7px;
    padding-bottom: 7px;
    padding-left: 20px;
    padding-right: 20px;
  }

  .left {
    float: none;
    margin-right: 0;
    transition: 1s;
  }

  nav ul li {
    display: block;
    text-align: center;
    width: 100%;
    transition: 1s;
    padding: 7px;
  }

  #nav-icon {
    display: inline;
    float: right;
    line-height: 50px;
  }
}
```

```
/* NAVIGATION ENDS HERE */
```

## Style.css:

```
* {  
  
  font-family: 'Overpass', Tahoma, Geneva, Verdana, sans-serif;  
  background-color: #1e1e21;  
  color: #fff;  
}  
  
p {  
  font-size: 14px;  
}  
  
a {  
  text-decoration: none;  
  color: #fff;  
}  
  
a: hover {  
  text-decoration: none;  
  color: #fff;  
}  
  
.landing {  
  margin-top: 100px;  
}  
  
#navbarToggle {  
  background-color: #000;  
}  
  
.navbar.navbar-dark .nav-item .nav-link {  
  color: #fff !important;  
  background-color: #000;  
}  
  
.navbar.navbar-dark .nav-item .nav-link: hover {  
  color: #fff !important;  
  background-color: #000;  
}  
  
.navbar {
```

```
    background-color: #000!important;
}

.navbar-nav {
    background-color: #000;
}

.navbar-brand {
    background-color: #000;
}

.lg {
    background-color: #000;
    font-size: xx-large;
}

h1 {
    font-size: 33px;
    color: #dfdcd9;
}

h2 {
    font-size: 1.5em;
    color: #dfdcd9;
}

.author {
    font-size: 12px;
}

.date {
    font-size: 12px;
}

.article-card {
    position: relative;
    display: -ms-flexbox;
    display: flex;
    -ms-flex-direction: column;
    flex-direction: column;
    min-width: 0;
    color: #000;
    background-color: #fff;
    word-wrap: break-word;
    background-clip: border-box;
```

```
border-radius: 10px;
margin-top: 15px;
}

.img {
width: 100%;
}

.image-fluid {
height: 18em;
}

.cardfooter {
padding: 0.75rem 1.25rem;
padding-bottom: 0px;
border-radius: 0px 0px 10px 10px;
}

.article_name {
color: #dfdcd9;
}

.source-body {
margin-bottom: 30px;
}

.source {
margin-top: 5px;
padding-top: 5px;
padding-bottom: 0px;
margin-bottom: 0px;
color: #dd0000;
}

.logo {
color: #dd0000;
background-color: #000;
}

.svg {
background-color: #000;
}

.p-svg {
background-color: #26262a;
```

```
}

.footer {
  background-color: #000;
  padding-top: 20px;
  padding-bottom: 15px;
}

.footer-text {
  background-color: #000;
  margin-bottom: 0px;
}

.dropdown-menu {
  background-color: #1e1e21;
}

.dropdown-item {
  color: #fff;
}

.dropdown-item:hover {
  color: #fff;
  background-color: #26262a;
}

.published {
  background-color: #26262a;
}

.card-footer {
  background-color: #26262a;
}
```

## **GitHub & Project Demo Link**

### **GitHub Link:**

<https://github.com/IBM-EPBL/IBM-Project-22813-1659858704>

### **Project Demo Link:**

- [https://cloud-object-storage-so-cos-standard-oen.s3.jp-tok.cloud-object-storage.appdomain.cloud/onetap\\_signup.mp4.mp4](https://cloud-object-storage-so-cos-standard-oen.s3.jp-tok.cloud-object-storage.appdomain.cloud/onetap_signup.mp4.mp4)
  
- <https://onetap.s3.eu.cloud-object-storage.appdomain.cloud/OneTapDemo.mp4>