

# NEWS TRACKER APPLICATION

## 1. INTRODUCTION

- 1.1 Project Overview.....
- 1.2 Purpose.....

## 2. LITERATURE SURVEY

- 2.1 Existing problem.....
- 2.2References.....
- 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.....

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

- 7.1 Feature 1.....
- 7.2 Feature 2.....
- 7.3 Database Schema (if Applicable).....

## **8. TESTING**

8.1 Test Cases.....

## **9. RESULTS**

9.1 Performance Metrics.....

## **10. ADVANTAGES&DISADVANTAGES.....**

## **11. CONCLUSION.....**

## **12. FUTURE SCOPE.....**

## **13. APPENDIX.....**

Source Code

GitHub & Project Demo Link

# **1. INTRODUCTION**

## **1.1 Project Overview**

News Tracker is an application designed to enhance and optimize the way a user interacts with news stories. This is achieved primarily using a search engine that connects to News API which allows the user to narrow-down the content of their results. Additional search parameters such as Polarity and Subjectivity use Natural Language Processing through Python's NLTK and Text blob libraries to parse through the html data given by the URL returned by News API , which will order a user's search results by the degree to which an article is objective or positive, with the corresponding score listed for each article. Other features available to users include the ability to save stories that they want to refer back to later, save multiple search queries for easy-access through an accordion drop-down, and the ability to set one of said queries as a user's default so that their headline feed will be base its results off that query default so that the main headline page will show results based off of the user's default search query

## **1.2 Purpose**

As world's technology is rapidly growing we has fast connection and network to instantly connect to other person. Day to day use in mobile, tablets and laptop is increasing, most of the people already have this facilities. In this fast and information oriented world we need to stay updated with every incidents and news too. This News app is android mobile application where user have access to latest news from 120+ newspapers from 50+ countries. The main focus of this application is to connect news articles from all around the world and deliver it to user as fast as possible in best visualize way. Android provides simple application structure and requires Java and Mark-up languages knowledge to work with. Such as, an discrete movement delivers a solitary screen for a user interface and a service whole completes work in the contextual . We can work on different module separately and can combine at the end, we can also add future modules easily afterwards

## **2. LITERATURE SURVEY**

### **2.1 Existing problem**

There are multiple news-sharing apps used by a single user and are often spammed with notifications. There is also a lot of fake news which gets shared. A news-sharing app wants to help users find relevant and important news easily every day and also understand explicitly that the news is not fake but from proper sources. As I felt that there is a considerable hike in the news reading population due to COVID 19 and Work at home. My team members found it acceptable. Due to changes brought about by the pandemic, people started depending on the news a lot to get updates about the situation around them. We thought that this behavior change would be very interesting to study. Based on my suggestion my team made it a criterion.

### **2.2 References**

#### **1. An Approach to News Event Detection and Tracking Based on Stream of Online News Source:**

IEEE Xplore Authors: Yajie Qi, Li Zhou, Huayou Si, Jian Wan, Ting Jin.

Websites: <https://ieeexplore.ieee.org/document/8048142>

#### **About the Paper:**

Once an event occurs, usually there are a large number of online news to be released. How to quickly and accurately detect the hot events from the huge amount of online news is the focus and hotspot. Event detection and tracking technology is as a key technology to solve this problem. In this paper, we propose an approach to detect hot events from the online news stream in a timely manner and track the hot events. Based on the idea of single-pass clustering algorithm, this approach addresses the weight of keywords and proposes a new method to calculate similarity among news to track event. Through the analysis of the experimental results, we can find that this algorithm has a good effect on hot event detection.

#### **2. Exploring Mobile News Reading Interactions for News App Personalization Source:**

ResearchGate Authors: Marios Constantinides, John Dowell, David Johnson, Sylvain Malacria.

Websites :<https://www.researchgate.net/publication/299870645>

#### **About the Paper:**

As news is increasingly accessed on smartphones and tablets, the need for personalizing news app interactions is apparent. We report a series of three studies addressing key issues in the development of adaptive news app interfaces. We first surveyed users' news reading preferences and behaviors;

analysis revealed three primary types of reader. We then implemented and deployed an Android news app that logs users' interactions with the app. We used the logs to train a classifier and showed that it is able to reliably recognize a user according to their reader type. Finally, we evaluated alternative, adaptive user interfaces for each reader type. The evaluation demonstrates the differential benefit of the adaptation for different users of the news app and the feasibility of adaptive interfaces for news apps.

### **3.A News Event Detection and Tracking Algorithm Based on Dynamic Evolution Model:**

Via analyzing news data on the Internet, an algorithm is presented for news event detection and tracking based on a dynamic evolution model, which borrows the idea of single-pass clustering and combines the specialties of news. The dynamic model is given based on the living characteristics of news event, including similarity computing model based on time distance between news story and news event, event model evolution algorithm, and dynamic threshold idea. This algorithm can automatically organize news data into news special topics ,and furthermore provide personalized service for users. Finally ,experimental results are used to indicate the validity of the algorithm.

### **2.3 Problem Statement Definition:**

A problem statement is a concise description of the problem or issues a project seeks to address. The problem statement identifies the current state, the desired future state and any gaps between the two. A problem statement is an important communication tool that can help ensure everyone working on a project knows what the problem they need to address is and why the project is important.

#### **Customer Problem Statement:**

The customer needs a way to get relevant news based on his choices so that the customer does not have to spend a lot of time on searching news. News is filled with ads and spams annoys and irritates the customer and affects the user experience. Market is full of news with all categories. Customers 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.

### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas:


An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



## 3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. 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.

### Step-1: Team Gathering, Collaboration and Select the Problem Statement



## 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

➔

#### 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

---

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

**B** Set the goal  
Think about the problem you'll be focusing on solving in the brainstorming session.

**C** 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

How might we [your problem statement]?




#### 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.

## Step-2: Brainstorm, Idea Listing and Grouping

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

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 set to the pencil (switch to sketch) icon to start drawing

**RASHEED**

Much important news is shared these days through social media

**KATHIR**

An news app enables users to not only receive the information, but to interact with it, share it or discuss it in real time.

**MANO**

Members are also being drawn to the app through custom and controlled push notifications instead of those that are sporadic and general through social media platforms.

**MANIC**

news feed is a perfect way to keep your attendees connected with real-time agendas and links to session materials.



Step-3: Idea Prioritization

Template



### Idea prioritization

Use this framework to rank ideas based on their feasibility and impact to visually compare the merits of multiple ideas. Deliver a set of ideas that your team wants to try out, and identify which of them need to be prioritized.

Share template feedback

1

Collect your ideas in one place

Jot down different ideas your team is interested in trying out. These could be different solutions, or different approaches to the same solution. As a team, go through the ideas in the idea bank one by one and place them on the grid. Take the time to discuss each idea and come to a consensus on where it should go.

Idea bank

What's the impact of this idea?

What's the effort to implement this idea?

What's the complexity of this idea?

High

Low

Importance

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

Low

High

Feasibility

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

### 3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Most people don't like to carry a newspaper with them. Some people want them to be updated only in the area they are interested in.
2.	Idea / Solution description	An application needs to be developed in which users can read news whenever they want and they will be able to customize their area of interest. So that they will be notified, if any new news is updated in their interested areas.
3.	Novelty / Uniqueness	<ol style="list-style-type: none"> <li>1. A user can read news only from their interested fields rather than reading all the news.</li> <li>2. This application provides users with a trusted and secured ecosystem. News shared through the application is original and spam free.</li> </ol>
4.	Social Impact / Customer Satisfaction	This application encourages its users to provide feedback. Based on that feedback, developments were made eventually.
5.	Business Model (Revenue Model)	<ol style="list-style-type: none"> <li>1. Add advertisements to the application, so that we can get revenue from those advertisement-sponsored organizations. More advertisements may irritate the user.</li> <li>2. Add premium subscription, users who subscribe for premium won't get advertisements.</li> </ol>
6.	Scalability of the Solution	As it was an application-based project, correct ideation and execution can develop an application with no bugs and errors, so that the user might like our application and some might suggest and share it to their surroundings, resulting in an increase in our application insights.

### 3.4 Problem Solution fit

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> Who is your customer? i.e. working parents of 0-5 y.o. kids  1. News reader 2. People	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.  1. It will consume more time 2. It will consume more cost 3. Network connection	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking  People may use either newspaper or social media or youtube channels to know the news	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one, explore different sides.  1. People can get simultaneous breaking news 2. We can avoid fake news 3. News received at correct time	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations.  In a busy world people not have allocate time for reading newspaper and watching news channels	<b>7. BEHAVIOUR</b> <span>BE</span> What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)  People follow youtube channels but this will not possible to know all news. People buy a news paper they don't read all news because of time cons	
Identify strong TR & EM	<b>3. TRIGGERS</b> <span>TR</span> What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.  Reading about a more efficient solution in the news	<b>10. YOUR SOLUTION</b> <span>SL</span> If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.  Making separate space for each category of news. people select the news category and know all news about that.	<b>8. CHANNELS of BEHAVIOUR</b> <span>CH</span> <b>8.1 ONLINE</b> What kind of actions do customers take online? Extract online channels from #7  In online people know news faster through network  <b>8.2 OFFLINE</b> What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.  In offline people must allocate time for reading newspaper	Extract online & offline CH of BE
	<b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in control - use it in your communication strategy & design.  People will know the news in faster			



Problem-Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 license.  
 Created by Daria Nepriakhina / Amaltama.com

## 4. REQUIREMENT ANALYSIS

Requirements analysis, also called requirements engineering, is the process of determining user expectations for a new or modified product. These features, called requirements, must be quantifiable, relevant and detailed. In software engineering, such requirements are often called functional specifications.

### 4.1 Functional requirement

#### Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through online application Registration through Gmail Registration through website
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User login	Login through browser directly by entering username and password Login through Login through email
FR-4	User interaction	Done through user interface between client and server View the related news by subscribed or requested page
FR-5		Application have tools to share this news in social networks

## 4.2 Non-Functional requirements

### Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	End users can receive push updates for new content on a site by subscribing to the site's news feed
NFR-2	Security	How well are the system and its data protected against attacks
NFR-3	Reliability	How often does the system experience critical failures? How much time does it take to fix the issue when it arises ?And how is user availability time compared to downtime?
NFR-4	Performance	Performance is the core non-functional requirements no system can do without. It defines how fast a software system or a particular piece of it responds to certain users actions under a certain workload. In most cases, this metric explains how long a user must wait before the target operation happens (the page renders, a transaction is processed, etc.) given the overall number of users at the moment. But it's not always like that. Performance requirements may describe background



## 5. PROJECT DESIGN

Project design is an early phase of the project lifecycle where ideas, processes, resources, and deliverables are planned out. A project design comes before a project plan as it's a broad overview whereas a project plan includes more detailed information.

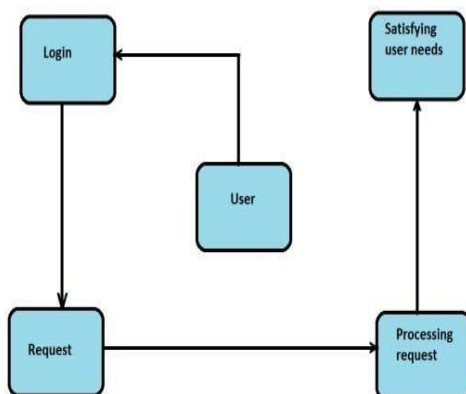
There are seven steps involved when creating a project design, including defining goals and using a visual aid to communicate objectives

These visual elements include a variety of methods such as Gantt charts, Kanban boards, and flowcharts. Providing a visual representation of your project strategy can help create transparency between stakeholders and clarify different aspects of the project, including its overall feasibility.

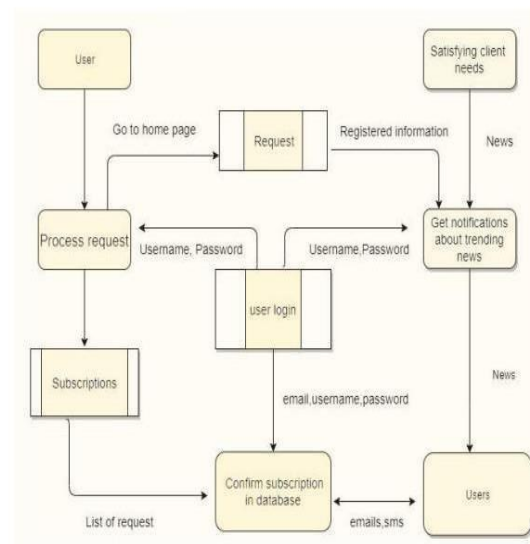
## 5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear 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.

Data Flow Diagram for the proposed solution:



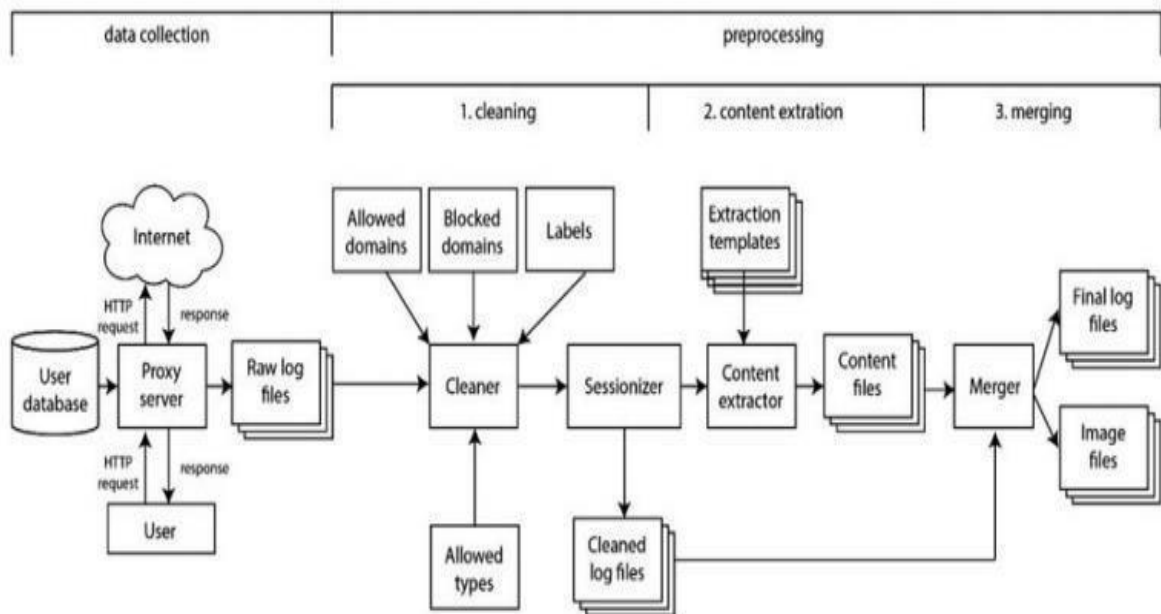
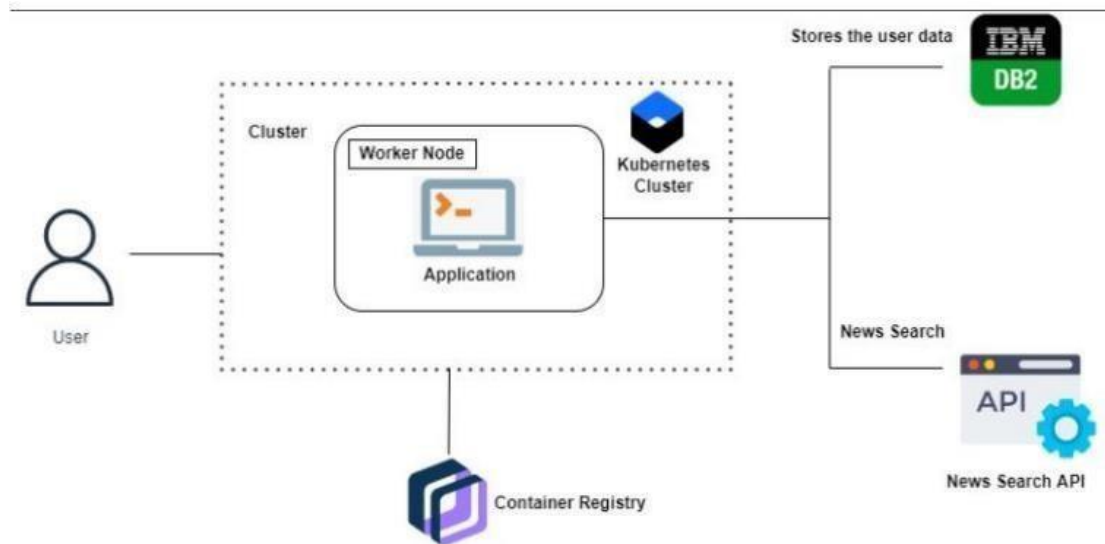
**DFD Level 0 (Industry Standard):**



## 5.2 Solution & Technical

### Architecture

#### 5.3 SOLUTION ARCHTECTURE:



## TECHNICAL ARCHITECTURE:

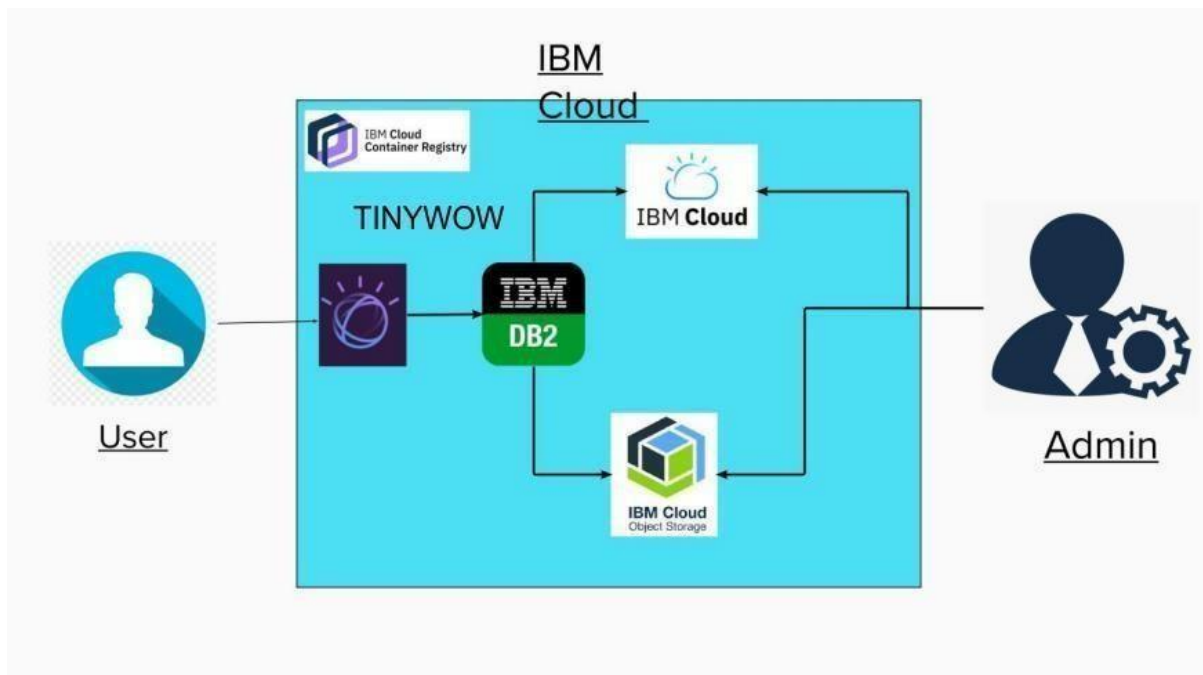


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL
6.	Cloud Database	Database Service on Cloud	IBM DB2
7.	File Storage	File storage requirements	IBM Block Storage
8.	Infrastructure (Server / Cloud)	Application Deployment on Cloud Cloud Server Configuration : Db2 /python	Kubernetes,



**Table-2: Application Characteristics:**

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask	Python
2.	encryption hashing and salting	Encryption hashing and salting	Encryptions
3.	Scalable Architecture	Getting resources to different parts of the system that need it	Microservices Architecture
4.	Availability	The Application available 24/7	IBM Cloud
5.	Performance	1000 request per day	IBM Watson

## 5.4 User Stories

A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming 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	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail	I can register through Gmail by OTP authentication	Medium	Sprint-2
	Login	USN-4	As a user, I can log into the application by entering email & password	I can view all types of information through this application	High	Sprint-1
	Dashboard	USN-5	To see their histories about recently viewed, updates for search related news, current progress, feedback		Medium	Sprint-2
Customer (Web user)	Browser	USN-6	Works as an interactive medium between client and server	I can access the resources through browser	High	Sprint-1
Customer Care Executive	Chat bot	USN-7	Rectify the customer's issues related to account, subscription and customization	Chat bot can resolve simple issues for customers	Low	Sprint-2
Feedback	Feedback Form	USN-8	Getting feedback from customers helps application's administrator to improve the quality of the application	Customers can tell their opinions	High	Sprint-1
Administrator	Admin module	USN-9	As an admin, I will modify the application as per customer requirements and fix the bugs to give customers a bug free service	I can modify the entire application	High	Sprint-2

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation

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.	5	High	Rasheed Mano Kathir Manic
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	5	High	Rasheed Kathir
Sprint-1		USN-3	As a user, I can register for the application through Gmail	5	Medium	Mano Manic
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	5	High	Rasheed Kathir
Sprint-2	Dashboard	USN-5	As a user, I can enter the interests and choices of news I want to see for the first time in dashboard.	10	High	Rasheed Kathir Mano Manic
Sprint-2	Dashboard User Interface	USN - 11	Administrator designing the user interface	10	Medium	Rasheed Mano
Sprint-3		USN-6	As a user I can go through the feed of news filtered according to my wish.	10	High	Kathir Manic
Sprint-3		USN-7	As a user, I can log out my account in settings.	10	Medium	Rasheed Kathir
Sprint-4		USN-8	As a user, I can update my interests and choice in account settings.	10	Medium	Rasheed Kathir

### 6.2 Sprint Delivery Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-4	Chat bot / Query	USN-9	Solve issues brought up by client	5	Medium	Rasheed Manic
Sprint-4		USN-10	Roll out updates and bug fixes	5	High	Rasheed

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

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

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

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

### Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

[illegible]

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

### 7.1 Feature 1

#### VERIFICATION EMAIL SENDER

```
def emailSender(email, token):
    configuration = sib_api_v3_sdk.Configuration()
    configuration.api_key['api-key'] = app.data['mail_api_key']
    api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
        sib_api_v3_sdk.ApiClient(configuration))
    now = datetime.now()
    dt_string = now.strftime("%d/%m/%Y %H:%M:%S")
    msg = {}
    msg['Subject'] = "Verfiy your NewsTracker Account"
    msg['From'] = {"name": "News Tracker Dev Team",
                  "email": "verify@news"
                  "tracker.com"}
    msg['To'] = [{"email": email}]
    msg['Text'] = f'Please click this <a href="http://127.0.0.1:5500/frontend/pages/verify.html?token={token}">link</a> to verify your account'
    html = f"""
    <html>
    <head></head>
    <body>
    <p>நன்றி, for joining NewsTracker </p>
    <br>
    <p>Hurray🎉, you just registered at NewsTracker<br><br>
    Please click the following link to verify your account:<br>
    <a href="http://127.0.0.1:5500/frontend/pages/verify.html?token={token}">Click Here to Verify 🎉</a>
    </p>
    <br>
    <p>⚠️Note: This link expires within one hour from the time sent</p>
    <br><br>
    <p>Regrads,<br></p>
    <p><a href="https://localhost:5000">NewsTracker Dev Team</a></p>
    <br><br>
    <p>Email sent at {dt_string}</p>
    </body>
    </html>
    """
    send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
        to=msg['To'], html_content=html,
        sender=msg['From'], subject=msg['Subject'], text_content=msg['Text'])
    try:
        api_response = api_instance.send_transac_email(send_smtp_email)
        print(api_response)
    except ApiException as e:
        print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
```

The above function is used to send the verification code to the desired email.

## 7.2 Feature 2

### Cookie Checker

```
1 def token_required(f):
2     @wraps(f)
3     def decorated(*args, **kwargs):
4         token = request.cookies.get("access_token")
5         try:
6             data = jwt.decode(token, app.app.config['SECRET_KEY'], algorithms=['HS256'])
7             ip=request.headers.get("ip")
8             cookieIp=data['ip']
9             if(ip!=cookieIp):
10                 resp={"status":"not logged in"}
11                 @after_this_request
12                 def deleter(response):
13                     response.delete_cookie("access_token",path="/")
14                     response.delete_cookie("email",path="/")
15                     return response
16                 return resp,401
17         except:
18             resp = {"status":"not logged in"}
19             @after_this_request
20             def deleter(response):
21                 response.delete_cookie("access_token",path="/")
22                 response.delete_cookie("email",path="/")
23                 return response
24             return resp, 401
25         return f(data['email'],*args, **kwargs)
26     return decorated
```

This code is used to check the cookie from the client side and checks whether the user is signed in or not.

## 7.3 Database Schema (if Applicable)

### User Table

Table definition

USER					Approximate 2 rows (4.03 MB) Updated on 2022-10-17 15:43:28
Name	Data type	Nullable	Length	Scale	
ID	INTEGER	N		0	
NAME	VARCHAR	N	255	0	
EMAIL	VARCHAR	N	255	0	
PASSWORD	VARCHAR	N	255	0	
FAVOURITES	CLOB	Y	1048576	0	
BOOKMARKS	CLOB	Y	1048576	0	
VERIFIED	BOOLEAN	Y	1	0	
RESEND_TIME	VARCHAR	N	255	0	

### Bookmarks Table:

Table definition

BOOKMARK					Approximate -1 rows (4.00 MB) Updated on
Name	Data type	Nullable	Length	Scale	
ID	INTEGER	N		0	
DATA	CLOB	N	1048576	0	



## 8. TESTING

### 8.1 TEST CASES:

Test case	feature	component	Test scenario	Expected result	Actual result	status	comments	bug	Executed by
Sign in	Functional	Login page	Verify user can see the sign in option	can visible	Yes visible	pass	successful	-	RASHEED KATHIR MANO MANIC
Sign up	Functional	Login page	Verify user has the option to sign up	Can visible	Yes visible	pass	Successful	-	RASHEED KATHIR MANO MANIC
Forgot password	Functional	Login page	Verify user has the option to forgot password	Yes the option is available	Option is available	pass	Successful	-	RASHEED KATHIR MANO MANIC

Fetch news	Functional	Home page	Verify user can get the news	News will be feed to the app	404 error	Fail	unsuccessful	App integration problem	Arun,Ram
Types of news available in the fetch news page	Functional	Fetch news	Types of news available	Weather, Sport, Economy.	Hover buttons will be shown.	Yes	successful	-	RASHEED, KATHIR MANO MANIC

## 9. RESULTS

### 9.1 Performance Metrics

The performance of a recommendation algorithm is evaluated by using some specific metrics that indicate the accuracy of the system. The type of metric used depends on the type of filtering technique. Root Mean Square Error (RMSE), Receiver Operating Characteristics (ROC), Area Under Cover (AUC), Precision, Recall and F1 score is generally used to evaluate the performance or accuracy of the recommendation algorithms.

**Root-mean square error (RMSE).** RMSE is widely used in evaluating and comparing the performance of a recommendation system model compared to other models. A lower RMSE value indicates higher performance by the recommendation model. RMSE, as mentioned by [69], can be as represented as follows:

$$RMSE = \sqrt{\frac{1}{N_p} \sum_{u,i} (p_{ui} - r_{ui})^2} \quad (1)$$

where,  $N_p$  is the total number of predictions,  $p_{ui}$  is the predicted rating that a user  $u$  will select an item  $i$  and  $r_{ui}$  is the real rating.

**Precision.** Precision can be defined as the fraction of correct recommendations or predictions (known as True Positive) to the total number of recommendations provided, which can be as represented as follows:

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)} \quad (2)$$

It is also defined as the ratio of the number of relevant recommended items to the number of recommended items expressed as percentages.

**Recall.** Recall can be defined as the fraction of correct recommendations or predictions (known as True Positive) to the total number of correct relevant recommendations provided, which can be as represented as follows:

$$Recall = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)} \quad (3)$$

It is also defined as the ratio of the number of relevant recommended items to the total number of relevant items expressed as percentages.

**F1 Score.** F1 score is an indicator of the accuracy of the model and ranges from 0 to 1, where a value close to 1 represents higher recommendation or prediction accuracy. It represents precision and recall as a single metric and can be as represented as follows:

$$F1 \text{ score} = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall}) \quad (4)$$

**Coverage.** Coverage is used to measure the percentage of items which are recommended by the algorithm among all of the items.

**Accuracy.** Accuracy can be defined as the ratio of the number of total correct recommendations to the total recommendations provided, which can be as represented as follows:

$$\text{Accuracy} = \frac{TP + FN}{TP + FN + TN + FP} \quad (5)$$

**Intersection over union (IoU).** It represents the accuracy of an object detector used on a specific dataset [70].

$$\text{IoU} = \frac{TP}{TP + FN + FP} \quad (6)$$

**ROC.** ROC curve is used to conduct a comprehensive assessment of the algorithm's performance [57].

**AUC.** AUC measures the performance of recommendation and its baselines as well as the quality of the ranking based on pairwise comparisons [5].

**Rank aware top-N metrics.** The rank aware top-N recommendation metric finds some of the interesting and unknown items that are presumed to be most attractive to a user [71]. Mean reciprocal rank (MRR), mean average precision (MAP) and normalized discounted cumulative gain (NDCG) are three most popular rank aware metrics.

**MRR.** MRR is calculated as a mean of the reciprocal of the position or rank of first relevant recommendation [72,73]. MRR as mentioned by [72,73] can be expressed as follows:

$$\text{MRR} = \frac{1}{N_u} \sum_{u \in N_u} \frac{1}{L_{nu}[k] \in R_u} \quad (7)$$

where  $u$ ,  $N_u$  and  $R_u$  indicate specific user, total number of users and the set of items rated by the user, respectively.  $L$  indicates list of ranking length ( $n$ ) for user ( $u$ ) and  $k$  represents the position of the item found in the lists  $L$ .

**MAP:** MAP is calculated by determining the mean of average precision at the points where relevant products or items are found. MAP as mentioned by [73] can be expressed as follows.

$$\text{MAP} = \frac{1}{N_u} \sum_{u \in N_u} \frac{1}{|R_u|} \sum_{k=1}^n \frac{1}{L_{nu}[k] \in R_u} P_u @ k \quad (8)$$

where  $P_u$  represents precision in selecting relevant item for the user.

**NDCG:** NDCG is calculated by determining the graded relevance and positional information of the recommended items, which can be expressed as follows [73].

$$\text{NDCG}_u = \frac{\sum_{k=1}^n G(u, n, k) D(k)}{\sum_{k=1}^n G^*(u, n, k) D(k)} \quad (9)$$

where  $D(k)$  is a discounting function,  $G(u, n, k)$  is the gain obtained recommending an item found at  $k$ -th position from the list  $L$  and  $G^*(u, n, k)$  is



the gain related to  $k$ -th item in the ideal ranking of  $n$  size for  $u$  user.

## **10. ADVANTAGES AND DISADVANTAGES:**

### **10.1 ADVANTAGES:**

#### **1. Own Your Channels:**

What's nice about building a news app is that you control the channel.

You do rely on Google and social media networks to decide when it makes the most sense to put it in front of your readers, but you've built a great and valuable asset you can count on, especially if users are registered, for the site or a newsletter.

#### **2. Better User Experience:**

Like all of us, your readers have come to expect a native app experience from all the brands they trust to consume content from. They want fast loading articles, offline use, no intrusive banners or popups, easy navigation designed for a thumb.

#### **3. Push notifications:**

With personalized push notifications from your news mobile app, you have a more effective means to communicate with readers and get stories in front of them when it counts.

### **10.2 DISADVANTAGES:**

#### **1. No comprehensive news coverage:**

Your organization would want a news API that delivers comprehensive news coverage and presents news articles in various languages. However, due to different news sites across several niches in the market, your organization may find it hard to choose the desired news API. Google News API follows its own algorithm while crawling and indexing sites, so news and niche sites might get missed.

#### **2. Data isn't always machine-readable and ready to integrate into your solution:**

Apart from that, numerous news APIs provide news data that isn't machine-readable or can ingest into your solution. Besides, news data isn't continuous most of the time. You may detect high latency for every site, or prime sites are crawled more regularly than small sites.

#### **3. It's not scalable and includes only current news data, no past data:**

Challenge most organizations face is when they expect to scale news data, and the news API they've chosen isn't advanced for their one-time project. Hence the news API fails to gauge the data. Likewise, some news APIs offer only real-time data and don't focus much on archive news data.

## **11. CONCLUSION :**

Thus we have developed a full stack application based on the plans and within the given time. We have tested the application in both desktop and mobile and it worked well, Overall it was a great experience.

## **12. FUTURE SCOPE:**

In future we may integrate our own news API instead of third party APIs and may develop a mobile native application so that it can be used in both android and IOS.



## 13. APPENDIX

### Source Code:

#### Backend\_

#### App.py

```
from dotenv import dotenv_values
from apiFetch import *
from flask import Flask
from flask_cors import CORS
from routes.register import Register
from routes.checkEmail import CheckEmail
from routes.verify import Verify
from routes.getnews import *
from routes.islogin import IsLogin
from routes.login import Login
from flask_restful import Api
from utils.apiFetch import apiRunner
app = Flask(__name__)
api = Api(app)
data = dotenv_values(".env")
app.config['SECRET_KEY'] = data["secret_key"]
app.config['SECURITY_PASSWORD_SALT'] =
data["secured_password_salt"]
apiRunner()
CORS(app, supports_credentials=True)
api.add_resource(Login, '/login')
api.add_resource(IsLogin, '/islogin')
api.add_resource(CheckEmail, '/register/check')
api.add_resource(Register, '/register')
api.add_resource(Verify, '/register/verify')
api.add_resource(Personal, '/news/recommended')
api.add_resource(News, '/news/<topic>')
```

## **Routes\_**

### **CheckEmail.py**

```
from flask import request
from flask_restful import Resource
from utils.dbQuery import selectQuery
class CheckEmail(Resource):
    def post(self):
        data=request.json
        email=str(data["email"])
        res=selectQuery("SELECT email FROM USER WHERE
EMAIL=?", (email,))
        print(res)
        if(not res):
            return {"status":True},200
        return {"status":False},400
```

### **Getnews.py**

```
from random import random
from flask_restful import Resource
from utils.cookieChecker import token_required
from utils.dbQuery import selectQuery
from utils.apiFetch import apiData
import random
from utils.testData import retArray
class Personal(Resource):
    @token_required
    def get(email, self):
        topicsArr = ["sport", "tech", "world", "finance", "politics", "business",
                    "economics", "entertainment", "beauty", "travel", "music", "food",
                    "science", "cricket"]
        fav = selectQuery("SELECT favourites from user where email=?",
(email,))[
    'FAVOURITES']
```

```

fav = fav.split(',')
for x in fav:
    topicsArr.remove(x)
retArr = []
favList = []
# favList = retArray
nonFavList = []
for x in fav:
    try:
        data = apiData(x)
    except:
        data=None
    if (data is not None):
        for y in data:
            favList.append(y)
try:
    random.shuffle(favList)
    favList = sorted(favList, key=lambda k: k['date'], reverse=True)
except:
    favList=[]
for x in topicsArr:
    try:
        data = apiData(x)
    except:
        data=None
    if(data is not None):
        for y in data:
            nonFavList.append(y)
try:
    random.shuffle(nonFavList)
    nonFavList = sorted(nonFavList, key=lambda k: k['date'], reverse=True)
except:
    nonFavList=[]
retArr = favList+nonFavList
return {"data": retArr}, 200

class News(Resource):
    @token_required

```

```

def get(email,self,topic):
    topicsArr = ["headline", "sport", "tech", "world", "finance", "politics",
"business",
                "economics", "entertainment", "beauty", "travel", "music", "food",
"science", "cricket"]
    if(topic not in topicsArr):
        return {"status": "Not a valid topic"}, 404
    try:
        retArr=apiData(topic)
        random.shuffle(retArr)
        retArr=sorted(retArr, key=lambda k: k['date'], reverse=True)
    except:
        retArr=[]
    return {"data": retArr}, 200

```

### **Islogin.py**

```

from flask import request, after_this_request
from utils.password import checkPassword
from flask_restful import Resource
from utils.cookieChecker import token_required
class IsLogin(Resource):
    @token_required
    def get(email, self):
        return {"status": "Logged in"}, 200

```

### **login.py**

```

from datetime import datetime
from flask_restful import Resource
from flask import request, after_this_request
from utils.password import checkPassword
from utils.dbQuery import *
from dateparser import parse
from utils.tokenener import generate_confirmation_token
from utils.emailSender import emailSender
import app
import jwt
class Login(Resource):
    def post(self):

```

```

data=request.json
ip=request.headers.get("ip")
email=data["email"]
password=data["password"]
queryRes=selectQuery("SELECT * from user where email=?", (email,))
res=checkPassword(password,queryRes["PASSWORD"])
if(not res):
    return {"status":"Wrong credentials"},400
if(not queryRes["VERIFIED"]):
    lastTime=parse(queryRes["RESEND_TIME"])
    currTime=datetime.now()
    diff=currTime-lastTime
    diff=diff.total_seconds()
    if(diff>3600):
        token=generate_confirmation_token(email)
        emailSender(email,token)
        insertQuery("UPDATE user set RESEND_TIME=? where
email=?",(datetime.now(),email))
        return {"status":"Not verified"},400
    @after_this_request
    def cookieSender(response):
        access_token=jwt.encode(
            {"email":email,"ip":ip},app.app.config['SECRET_KEY']
        )
        #
response.set_cookie("access_token",str(access_token),httponly=True,samesite=
None,path="/")
        #
response.set_cookie("email",str(email),httponly=True,samesite=None,path="/")
        response.headers.add('Set-Cookie',f'access_token={str(access_token)};
SameSite=None; Secure; HttpOnly; Path=/'
        response.headers.add('Set-Cookie',f'email={str(email)};
SameSite=None; Secure; HttpOnly; Path=/'
        return response
    return {"status":"Successfully Logged in"},200

```

## Register.py

```

from datetime import datetime

```

```

from flask import request,after_this_request
from flask_restful import Resource
from utils.dbQuery import insertQuery
from utils.emailSender import newEmailSender
from utils.password import genHash
class Register(Resource):
    def post(self):
        req=request.json
        name=req['name']
        email=req['email']
        password=req['password']
        favourite=req['favourite']
        fav=', '.join(favourite)
        if(name==" or email==" or password==" or fav==" ):
            return {"status":"Missing data"},404
        password=genHash(password)
        t=(name,email,password,fav,datetime.now())
        res=insertQuery('INSERT INTO user
(name,email,password,favourites,resent_time) values (?,?,?,?,?)',t)
        if(not res):
            return {"status":"Error while registering"},400

        @after_this_request
        def emailer(response):
            newEmailSender(email)
            return response
        return {"status":"Successfully registered"},200

```

## **verify.py**

```

from flask import request
from flask_restful import Resource
from utils.tokenener import confirm_token
from utils.dbQuery import insertQuery

class Verify(Resource):
    def post(self):
        data=request.json
        token=data["token"]
        email=confirm_token(token)
        if(not email):
            return {"status":"Couldn't verify"},400

```



```

        res=insertQuery("UPDATE user set verified=True where
email=?", (email,))
        if(not res):
            return {"status": "Couldn't Update"}, 400
        return {"status": "verified successfully"}, 200

```

## Utils\_

### apiFetch.py

```

from datetime import datetime
from time import sleep
import warnings
from dotenv import dotenv_values
from threading import *
import requests
from dateparser import parse
class Api:
    warnings.simplefilter('ignore')
    __key = dotenv_values(".env")
    __key = __key["key"]
    __apiMap = {}
    __mainApiMap = {}
    __url = "https://newscatcher.p.rapidapi.com/v1/latest_headlines"
    __headers = {
        "X-RapidAPI-Key": str(__key),
        "X-RapidAPI-Host": "newscatcher.p.rapidapi.com"
    }
    def __newCatcherRunner(self, title):
        querystring = {"topic": title, "lang": "en",
            "media": "True", "country": "IN"}
        response = requests.request(
            "GET", url=self._url, headers=self._headers, params=querystring)
        response = response.json()
        retArr = []
        for x in response["articles"]:
            newJson = {}
            newJson["url"] = x["link"]
            newJson["title"] = x["title"]
            newJson["img"] = x["media"]
            newJson["topic"] = x["topic"]
            currTime = parse(x["published_date"])
            newJson["date"] = currTime.strftime("%d/%m/%Y")
            retArr.append(newJson)
        return retArr
    def __topHeadlinesFetcher(self):
        querystring = {"topic": "news", "lang": "en", "media": "True", "country": "IN"}

```

```

        response = requests.request("GET", url=self._url, headers=self._headers,
params=querystring)
        response = response.json()
        retArr = []
        for x in response["articles"]:
            newJson = {}
            newJson["url"] = x["link"]
            newJson["title"] = x["title"]
            newJson["img"] = x["media"]
            newJson["topic"] = x["topic"]
            currTime = parse(x["published_date"])
            newJson["date"] = currTime.strftime("%d/%m/%Y")
            retArr.append(newJson)
        self.__apiMap["headline"]=retArr
        print("headline fetched at "+str(datetime.now()))
def __newsCatcherApiFetcher(self):
    arr = ["sport", "tech", "world", "finance", "politics", "business",
        "economics", "entertainment", "beauty", "travel", "music", "food", "science"]
    for x in arr:
        self._apiMap[x] = self._newCatcherRunner(x)
    print("NewsCatcher fetched at "+str(datetime.now()))
def __cricketFetcher(self):
    url = "https://cricbuzz-cricket.p.rapidapi.com/news/v1/index"
    headers = {
        "X-RapidAPI-Key": self.__key,
        "X-RapidAPI-Host": "cricbuzz-cricket.p.rapidapi.com"
    }
    response = requests.request("GET", url, headers=headers)
    response = response.json()
    response = response["storyList"]
    retArr = []
    for x in response:
        try:
            x = x["story"]
            newJson = {}
            newJson["url"] = f'https://www.cricbuzz.com/cricket-news/{x["id"]}/newsTrakcer'
            newJson["title"] = x["hline"]
            newJson["image"] =
f'https://www.cricbuzz.com/a/img/v1/500x500/i1/c/{x["id"]}/abc.jpg'
            currTime = datetime.fromtimestamp(int(x["pubTime"])/1e3)
            newJson["date"] = currTime.strftime("%d/%m/%Y")
            newJson["topic"] = "cricket"
            retArr.append(newJson)
        except:
            pass
    self.__apiMap["cricket"] = retArr
    print("Cricbuzz fetched at "+str(datetime.now()))
def newsCatcherThreader(self):

```

```

while True:
    print("NewsCatcher fetching.... at "+str(datetime.now()))
    try:
        self.__newsCatcherApiFetcher()
        self._mainApiMap = self._apiMap
    except:
        print("Error NewsCatcher fetching.... at "+str(datetime.now()))
        pass
    sleep(30*60)
def topHeadlinesThreader(self):
    while True:
        print("Headline fetching. .. at "+str(datetime.now()))
        try:
            self.__topHeadlinesFetcher()
            self._mainApiMap = self._apiMap
        except:
            print("Error headline fetching. ...at "+str(datetime.now()))
            pass
        sleep(30*60)
def cricbuzzThreader(self):
    while True:
        print("Cricbuzz fetching. .. at "+str(datetime.now()))
        try:
            self.__cricketFetcher()
            self._mainApiMap = self._apiMap
        except:
            print("Error Cricbuzz fetching.... at "+str(datetime.now()))
            pass
        sleep(15*60)
def dataGetter(self, topic):
    return self._mainApiMap[str(topic)]
a = Api()
def apiRunner():
    t1 = Thread(target=a.topHeadlinesThreader)
    t2 = Thread(target=a.newsCatcherThreader)
    t3 = Thread(target=a.cricbuzzThreader)
    t1.daemon=True
    t2.daemon=True
    t3.daemon=True
    t1.start()
    t2.start()
    t3.start()
def apiData(topic):
    return a.dataGetter(topic)

```

**CookieChecker.py**

```

import app
from functools import wraps
import jwt
from flask import request,after_this_request
def token_required(f):
    @wraps(f)
    def decorated(*args, **kwargs):
        token = request.cookies.get("access_token")
        try:
            data = jwt.decode(token, app.app.config['SECRET_KEY'],algorithms=['HS256'])
            ip=request.headers.get("ip")
            cookieIp=data['ip']
            if(ip!=cookieIp):
                resp={"status":"not logged in"}
                @after_this_request
                def deleter(response):
                    response.delete_cookie("access_token",path="/")
                    response.delete_cookie("email",path="/")
                    return response
                return resp,401
        except:
            resp = {"status":"not logged in"}
            @after_this_request
            def deleter(response):
                response.delete_cookie("access_token",path="/")
                response.delete_cookie("email",path="/")
                return response
            return resp, 401
        return f(data['email'],*args, **kwargs)
    return decorated

```

## dbConfig.py

```

from dotenv import dotenv_values
def getDbCred():
    data=dotenv_values(".env")
    dsn_hostname=data["ibm_host_name"]
    dsn_uid=data["ibm_user_id"]
    dsn_pwd=data["ibm_password"]
    dsn_driver=data["ibm_driver"]
    dsn_database=data["ibm_db_name"]
    dsn_port=data["ibm_port"]
    dsn_protocol=data["ibm_protocol"]
    dsn=(
        "DATABASE={1};"
        "HOSTNAME={2};"
        "PORT={3};"

```

```

        "PROTOCOL={4};"
        "UID={5};"
        "PWD={6};"
        "SECURITY=SSL").format(dsn_driver,dsn_database,dsn_hostname,dsn_p
ort,dsn_protocol,dsn_uid,dsn_pwd)
    return dsn

```

### **dbQuery.py**

```

import ibm_db
from utils.dbConfig import getDbCred
conn=ibm_db.connect(getDbCred(),"","")
def selectQuery(query,params=None):
    try:
        stmt=ibm_db.prepare(conn,query)
        if(params==None):
            ibm_db.execute(stmt)
            data=ibm_db.fetch_assoc(stmt)
            return data
        ibm_db.execute(stmt,params)
        data=ibm_db.fetch_assoc(stmt)
        return data
    except:
        return False
def insertQuery(query,params):
    try:
        stmt=ibm_db.prepare(conn,query)
        ibm_db.execute(stmt,params)
        return True
    except:
        return False

```

### **emailSender.py**

```

from __future__ import print_function
from utils.tokener import generate_confirmation_token
import sib_api_v3_sdk
import app
from sib_api_v3_sdk.rest import ApiException
from datetime import datetime
def emailSender(email, token):
    configuration = sib_api_v3_sdk.Configuration()

```

```

configuration.api_key['api-key'] = app.data['mail_api_key']
api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
    sib_api_v3_sdk.ApiClient(configuration))
now = datetime.now()
dt_string = now.strftime("%d/%m/%Y %H:%M:%S")
msg = { }
msg['Subject'] = "Verfiy your NewsTracker Account"
msg['From'] = { "name": "News Tracker Dev Team",
    "email": "verify@newstracker.com" }
msg['To'] = [ { "email": email } ]
msg['Text']=f'Please click this <a
href="http://127.0.0.1:5500/frontend/pages/verify.html?token={ token} ">link</a
> to verify your account'
html = f"""\
<html>
    <head></head>
    <body>
        <p>நன்றி, for joining NewsTracker 🙏</p>
        <br>
        <p>Hurray , you just registerd at NewsTracker<br><br>
        Please click the following link to verify your account:<br>
        <a
href="http://127.0.0.1:5500/frontend/pages/verify.html?token={ token} ">Click
Here to Verify 🏠</a>
        </p>
        <br>
        <p>    Note: This link expires within one hour from the time sent</p>
        <br><br>
        <p>Regrads,<br></p>
        <p><a href="https://localhost:5000">NewsTracker Dev Team</a></p>
        <br><br>
        <p>Email sent at { dt_string}</p>
    </body>
</html>
"""

send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
    to=msg['To'], html_content=html, sender=msg['From'],
subject=msg['Subject'],text_content=msg['Text'])
try:
    api_response = api_instance.send_transac_email(send_smtp_email)
    print(api_response)
except ApiException as e:

```

```

        print("Exception when calling SMTPApi->send_transac_email: %s\n" %
e)
def newEmailSender(email):
    token = generate_confirmation_token(email)
    emailSender(email, token)

```

### **password.py**

```

import bcrypt

def genHash(password):
    salt=bcrypt.gensalt()
    bytes=password.encode('utf-8')
    hash=bcrypt.hashpw(bytes,salt)
    print(hash)
    return hash

def checkPassword(password,hash):
    hash=hash.encode('utf-8')
    bytes=password.encode('utf-8')
    res=bcrypt.checkpw(bytes,hash)
    return res

```

### **tokener.py**

```

from itsdangerous import URLSafeTimedSerializer
import app

def generate_confirmation_token(email):
    serializer=URLSafeTimedSerializer(app.app.config['SECRET_KEY'])
    return
serializer.dumps(email,salt=app.app.config['SECURITY_PASSWORD_SALT']
)

def confirm_token(token,expiration=3600):
    serializer=URLSafeTimedSerializer(app.app.config['SECRET_KEY'])
    try:
        email=serializer.loads(
            token,
            salt=app.app.config['SECURITY_PASSWORD_SALT'],
            max_age=expiration

```

```
)  
except:  
    return False  
return email
```

## Frontend\_

### Bookmarks.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">  
  
    <link rel="stylesheet" href="../css/bookmarks.css">  
  
    <title>Document</title>  
  
</head>  
  
<body>  
  
    <nav id="head_nav">  
  
        <div class="title_container">  
  
              
  
            <h1 id="app_name">News App</h1>  
  
            <!-- <div class="search_container">  
  
                <input type="text"  
  
                    id="search_box"  
  
                    placeholder="Search for topics, locations and resources"  
  
                >  
  
            </div> -->  
  
            
```



```
<div class="profile_div display_none">

    <a class="bookmarked_link" href="/bookmarks.html"><h3
class="bookmarked">Bookmarks</h3></a>

    <h3 class="logout">Logout</h3>

</div>

</div>

<div class="menu_container">

    <h3 class="nav_button selected_nav_item">Home</h3>

    <h3 class="nav_button">India</h3>

    <h3 class="nav_button">World</h3>

    <h3 class="nav_button">Technology</h3>

    <h3 class="nav_button">Entertainment</h3>

    <h3 class="nav_button">Sports</h3>

    <h3 class="nav_button">Science</h3>

    <h3 class="nav_button">Health</h3>

</div>

</nav>

<section id="home">

    <div class="news_wrapper">

        <a class="news_cont" href="/news.html">

            <div class="img_cont">

            </div>

            <div class="news_content">

                <h2 class="news_heading">JSV effect</h2>

            </div>

        </a>

    </div>

</section>

</div>
```

<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Rerum eveniet quibusdam autem iste, eligendi nihil quod molestias facere incidunt quia officia atque distinctio consequuntur qui, suscipit voluptates quasi minus ipsum.</p>

</div>

</a>

</div>

</section>

<script src="../../js/main.js"></script>

</body>

</html>

## **Index.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">

<link rel="stylesheet" href="../../css/index.css">

<title>Document</title>

</head>

<body>

<nav id="head\_nav">

<div class="title\_container">



<h1 id="app\_name">News App</h1>

```
<!-- <div class="search_container">
  <input type="text"
    id="search_box"
    placeholder="Search for topics, locations and resources"
  >
</div> -->



<div class="profile_div display_none">
  <a class="bookmarked_link" href="/bookmarks.html"><h3
class="bookmarked">Bookmarks</h3></a>
  <h3 class="logout">Logout</h3>
</div>

</div>

<div class="menu_container">
  <h3 class="nav_button selected_nav_item">Home</h3>
  <h3 class="nav_button">India</h3>
  <h3 class="nav_button">World</h3>
  <h3 class="nav_button">Technology</h3>
  <h3 class="nav_button">Entertainment</h3>
  <h3 class="nav_button">Sports</h3>
  <h3 class="nav_button">Science</h3>
  <h3 class="nav_button">Health</h3>
</div>

</nav>

<section id="home">
  <div class="horizontal_content">
    <div class="news_wrapper">
```

```

```

```
<a class="news_cont" href="/news.html">
  <div class="img_cont">
    
```

```
</div>
```

```
<div class="news_content">
```

```
<h2 class="news_heading">JSV effect</h2>
```

```
</div>
```

```
</a>
```

```
</div>
```

```
<div class="news_wrapper">
```

```

```

```
<a class="news_cont" href="/news.html">
  <div class="img_cont">
    
```

```
</div>
```

```
<div class="news_content">
```

```
<h2 class="news_heading">JSV effect</h2>
```

```
</div>
```

```
</a>
```

```
</div>
```

```
<div class="news_wrapper">
```

```

```

```
<a class="news_cont" href="./news.html">
  <div class="img_cont">
    
  </div>
  <div class="news_content">
    <h2 class="news_heading">JSV effect</h2>
  </div>
</a>
</div>
<div class="news_wrapper">
  
  <a class="news_cont" href="./news.html">
    <div class="img_cont">
      
    </div>
    <div class="news_content">
      <h2 class="news_heading">JSV effect</h2>
    </div>
  </a>
</div>
</div>
<div class="news_wrapper">
  
  <a class="news_cont" href="./news.html">
```

```
<div class="img_cont">

</div>

<div class="news_content">

    <h2 class="news_heading">JSV effect</h2>

    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Rerum
eveniet quibusdam autem iste, eligendi nihil quod molestias facere incidunt quia
officia atque distinctio consequuntur qui, suscipit voluptates quasi minus
ipsum.</p>

</div>

</a>

</div>

</section>

<script src="../js/main.js"></script>

<script src="../js/index.js" type="module"></script>

</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>Login and Sign Up Page</title>
```

```
<!--font awesome-->

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.2.0/css/all.min.css">

<!--custom css file link-->

<link rel="stylesheet" href="../css/login.css">

</head>

<body>

<div class="container" id="container">

  <div class="form-container sign-up-container">

    <form id="signup-form" onsubmit="return false;">

      <h1>Create Account</h1>

      <span>You can use your custom email</span>

      <input type="text" placeholder="NAME" class="box" required
autocomplete="name">

      <input type="email" placeholder="E-mail" class="box" required
autocomplete="email">

      <input type="password" placeholder="PASSWORD" class="box"
required autocomplete="current-password">

      <input type="password" placeholder="RE-ENTER PASSWORD"
class="box" required autocomplete="current-password">

      <button class="btn">Sign Up</button>

      <h2></h2>

    </form>

  </div>

  <div class="form-container sign-in-container">

    <form onsubmit="return false;">

      <h1>Sign In</h1>

      <span>Enter your sign in credentials</span>
```

```
<input type="email" placeholder="E-mail" class="box"
autocomplete="email">

<input type="password" placeholder="PASSWORD" class="box"
autocomplete="current-password">

<button class="btn">Sign In</button>

<h2></h2>

</form>

</div>

<div class="overlay-container">

  <div class="overlay">

    <div class="overlay-panel overlay-left">

      <div class="container-checkbox">

        <ul class="ks-cboxtags">

          <li><input type="checkbox" id="checkboxOne"
value="sport" checked><label for="checkboxOne">sport</label></li>

          <li><input type="checkbox" id="checkboxTwo"
value="tech" checked><label for="checkboxTwo">tech</label></li>

          <li><input type="checkbox" id="checkboxThree"
value="world"><label for="checkboxThree">world</label></li>

          <li><input type="checkbox" id="checkboxFour"
value="finance"><label for="checkboxFour">finance</label></li>

          <li><input type="checkbox" id="checkboxFive"
value="politics" checked><label for="checkboxFive">politics</label></li>

          <li><input type="checkbox" id="checkboxSix"
value="business"><label for="checkboxSix">business</label></li>

          <li><input type="checkbox" id="checkboxSeven"
value="economics"><label for="checkboxSeven">economics</label></li>

          <li><input type="checkbox" id="checkboxEight"
value="entertainment"><label
for="checkboxEight">entertainment</label></li>
```



```
        <li><input type="checkbox" id="checkboxNine"
value="beauty"><label for="checkboxNine">beauty</label></li>

        <li><input type="checkbox" id="checkboxTen"
value="travel"><label for="checkboxTen">travel</label></li>

        <li><input type="checkbox" id="checkboxEleven"
value="music"><label for="checkboxEleven">music</label></li>

        <li><input type="checkbox" id="checkboxTwelve"
value="food"><label for="checkboxTwelve">food</label></li>

        <li><input type="checkbox" id="checkboxThirteen"
value="science"><label for="checkboxThirteen">science</label></li>

        <li><input type="checkbox" id="checkboxFourteen"
value="cricket"><label for="checkboxFourteen">cricket</label></li>

    </ul>

</div>

<h1>Already registred ?</h1>

<button class="btn" id="signin">sign in</button>

</div>

<div class="overlay-panel overlay-right">

<h1>Welcome to latest world updates!</h1>

<button class="btn" id="signup">sign up</button>

</div>

</div>

</div>

<div>

<script src="../js/login.js" type="module"></script>

</body>

</html>
```

## News.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">

  <link rel="stylesheet" href="../css/news.css">

  <title>Document</title>

</head>

<body>

  <nav id="head_nav">

    <div class="title_container">

      <h1 id="app_name">News App</h1>

      <!-- <div class="search_container">

        <input type="text"

          id="search_box"

          placeholder="Search for topics, locations and resources"

        >

      </div> -->

      <div class="profile_div display_none">

        <h3 class="logout">Logout</h3>

      </div>

    </div>

  </nav>

</body>

</html>
```

```
<div class="menu_container">
  <h3 class="selected_nav_item">Home</h3>
  <h3>India</h3>
  <h3>World</h3>
  <h3>Technology</h3>
  <h3>Entertainment</h3>
  <h3>Sports</h3>
  <h3>Science</h3>
  <h3>Health</h3>
</div>
</nav>
<iframe src="" frameborder="0">
  <!-- news API -->
</iframe>
<script src="../js/main.js"></script>
</body>
</html>
```

## Verify.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" />
    <link rel="stylesheet" href="../css/verify.css" />
```

```
<title>Verify</title>
</head>
<body>
  <nav id="head_nav">
    <div class="title_container">
      <h1 id="app_name">News Tracker</h1>
    </div>
  </nav>
  <div class="status-cont">
    <div class="welcome-text">Please wait... 🚧</div>
  </div>
</body>
<script src="../js/verify.js" type="module"></script>
</html>
```

GITHUB LINK:

<https://github.com/IBM-EPBL/IBM-Project-24987-1659951676/tree/main>

DEMO LINK:

<http://159.122.183.93:31965/>

---

---