

**NEWS TRACKER APPLICATION  
USING CLOUD**

*A Project report submitted in partial fulfilment of 7<sup>th</sup> semester in degree of*

**BACHELOR OF ENGINEERING  
IN  
ELECTRONICS AND COMMUNICATION ENGINEERING  
*Submitted by***

Team ID: PNT2022TMID41666

KUMARAVEL M

PRADEEP E

KARTHICK PS

KAVINRAJA M

GOKUL S



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION  
ENGINEERING MUTHAYAMMAL COLLEGE OF  
ENGINEERING, RASIPURAM**

NAMAKKAL 637408

**NOV-2022**

## **ABSTRACT**

Everyone has the right to freedom of speech. However, this right is being misused to differentiate and attack others, physically or verbally, in the name of free speech. This discrimination is known as hate speech. Hate speech can be well-defined as language used to express hate towards a person or a group of people based on characteristics such as race, religion, ethnicity, gender, nationality, disability and sexual orientation. The increasing usage of social sites and information sharing has specified major benefits to humanity. However, this has also assumed rise to a variety of challenges including the spreading and sharing of hate speech messages. Thus, to solve this emerging issue in social media sites, recent studies employed a variety of machine learning and deep learning algorithms with text mining algorithm to automatically detect the hate speech messages on real time datasets. Hence, the aim of this Project is to analyses the comments on social networks using Natural Language processing technique (NLP) and Deep learning algorithm named as Back propagation neural network algorithm. Using NLP technique, can extract the keywords from user generated content and implement Back Propagation neural network to classify the text whether it is positive or negative. If it is negative means, automatically block the comments as per user wish and also block the friends based on pre-defined threshold values. Experimental results shows that the proposed framework implemented in real time social network site with improved notification .

## **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**

### **6. PROJECT PLANNING & SCHEDULING 6.1 Sprint Planning & Estimation**

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

### **7. CODING & SOLUTIONING**

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

### **11. CONCLUSION**

### **12. FUTURE SCOPE**

### **13. APPENDIX**

Source Code GitHub & Project Demo Link

# **News Tracker Application**

## **1.INTRODUCTION**

Mobile app ecosystems are transforming patterns of news consumption. Until quite recently, reading the news was a niche use for smartphones [12], mostly for when users were ‘on the go’; now however, two in every three users of mobile devices in the US regularly access news and as many as one in five read in-depth news articles daily [2]; a similar picture is found in the UK [1]. This growth in mobile news access continues the migration of news consumers to the Internet.

### **1.1 Project overview**

As news is increasingly accessed on smartphones and tablets, the need for personalising 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 behaviours; 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 recognise 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.

### **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**

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.

## **2.2 References**

Like articles, websites, blogs.

<https://www.researchgate.net/>.

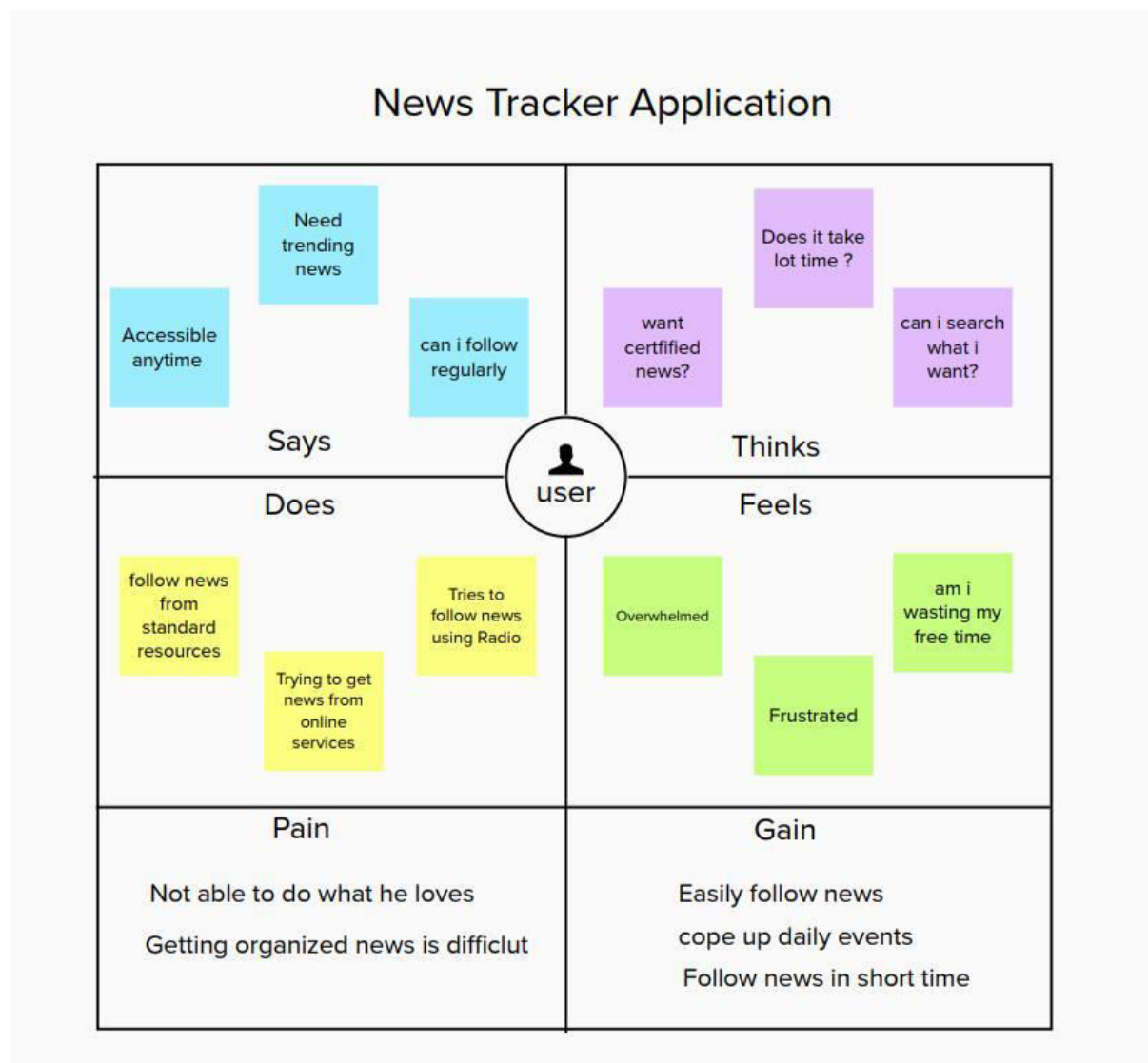
<https://techvidvan.com/tutorials/android-news-app-project-source-code/>

<https://www.ripublication.com/>

## **2.3 Problem Statement Definition**

- 1.The user needs a way to organize the news on daily basis so that he can cope up with daily events in his tight schedule.
- 2.The user wants to read news only about particular topics so that he can be informed about his interest.
- 3.The user wants to get informed from only certified news outlets .
- 4.The user needs a way to search about the news on topics he wants to.
- 5.The user wants to know the news about his surrounding using GPS location.

### 3. IDEATION & PROPOSED SOLUTION 3.1 Empathy Map Canvas



## 3.2 Ideation & Brainstorming

### PRADEEA

All in one place	Receive only the information you need
Use only trusted sources	Weather updates

### KAVIN

Very little resource usage	Curated and focused
Market trends tracking	Track how often user visit area

### KUMARAVEL M

Keep the news relevant and short	Customisable profile
Bookmark news	Deliver news, according to the persons internet

### GOKULS

Saves user Time	Create better and more targeted content
Deliver news in daily basis	Browsing with keyword

### KARTHICK PS

Improved OnTime Performance	Focus on lead
Syndicate your content	Competitive

### 3.3 Proposed Solution

S.NO	PARAMETERS	DESCRIPTION
1.	Problem statement (problems to be solved)	To help customer to access NEWS using Cloud Application.
2.	Idea/solution description	Making the customer to access news in few touch, providing authorized news, cloud based app, With tons of features to make it exciting
3.	Novelty/Uniqueness	Profile based account, like and share, download blogs, links to original publishers are provided.
4.	Social Impact/Customer satisfaction	Customer Satisfaction, Customer can follow NEWS publishers, like and share feature.
5.	Business Model (Revenue)	Showing ad, Showing a particular publishers NEWS and blogs.
6.	Scalability of the solution	Enabling the system to scalable according to the customers need



## 3.4 Problem Solution fit

<p><b>1. CUSTOMER SEGMENT(S)</b></p> <p>Who is your customer? i.e. working parents of 0-5 y.o. kids</p> <div> <p>Person who wants the facility to access NEWS in digital, with minimum amount of time.</p> </div>	<p><b>6. CUSTOMER CONSTRAINTS</b></p> <p>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.</p> <div> <p>User have only limited amount of time i.e his/her leisure time is only few minutes he needs to access NEWS.</p> </div>	<p><b>5. AVAILABLE SOLUTIONS</b></p> <p>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 &amp; cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking</p> <div> <p>They have solutions like reading NEWS in any hard copy or having to access internet for the NEWS.</p> </div>
<p><b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span></p> <p>Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.</p> <div> <p>Problem should be fixed is making an app which gathers NEWS using Internet.</p> </div>	<p><b>9. PROBLEM ROOT CAUSE</b> <span>RC</span></p> <p>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.</p> <div> <p>In our current daily life we have only Few minutes of leisure time using the time for reading NEWS is better to now about our society. Hence an app for NEWS gathering need to be developed.</p> </div>	<p><b>7. BEHAVIOUR</b> <span>BE</span></p> <p>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)</p> <div> <p>User can access NEWS with in few minutes to get informed about his needs.</p> </div>
<p><b>3. TRIGGERS</b> <span>TR</span></p> <p>What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.</p> <div> <p>The need to now about society or his personal interest.</p> </div> <p><b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span></p> <p>How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure &gt; confident, in control - use it in your communication strategy &amp; design.</p> <div> <p>To hard get informed about his needs After this he can enjoy his interest in no time.</p> </div>	<p><b>10. YOUR SOLUTION</b> <span>SL</span></p> <p>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.</p> <div> <p>An app Interacting with different websites is being created.</p> </div>	<p><b>8. CHANNELS OF BEHAVIOUR</b></p> <p><b>8.1 ONLINE</b> What kind of actions do customers take online? Extract online channels from #7</p> <p><b>8.2 OFFLINE</b> What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.</p> <div> <p>In online he need to create a account in the app, so only he can access NEWS from different sources through online. In offline he can only access hard copies of NEWS with only limited access.</p> </div>

## 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	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Login	User using his /her device to login to the app/website for reading news
FR-4	User search	For getting news he/she must need to search the news in the given search bar, Which will collect the news according to his search from the online sources like blog, e-news paper,Articles etc
FR-5	User Reads	Gathered news will be shown to the user in short lines/ summary manner. Along with news Pictures, hyperlinks for the blog, article Or original publisher of the news is given
FR-6	User features	User can have lot of features like Like and share and Post features.

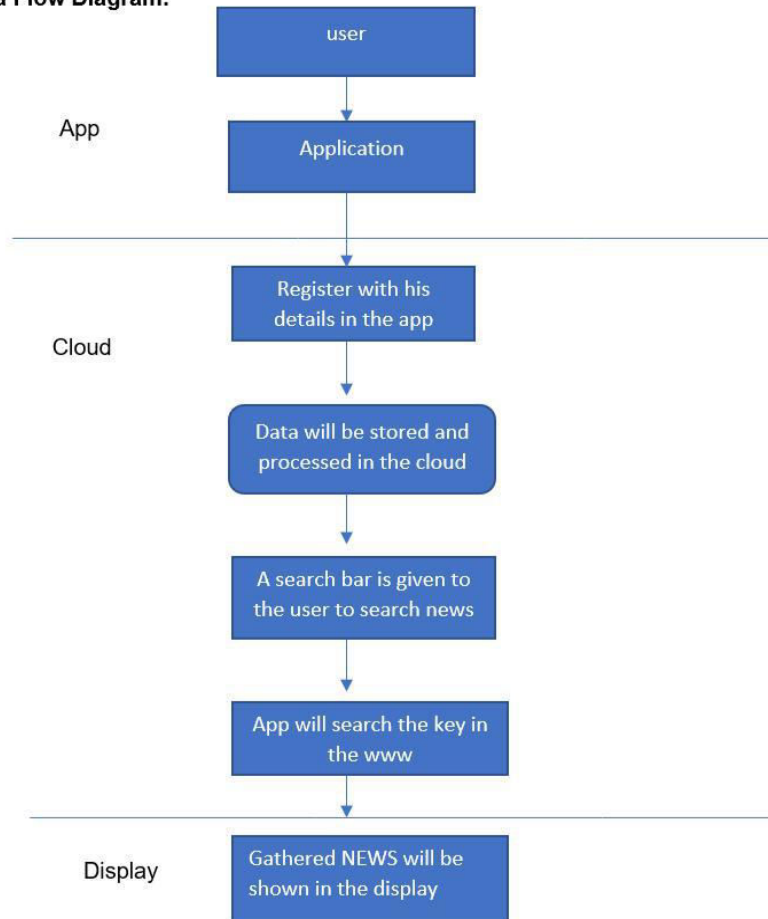
## 4.2 Non-functional Requirements:

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

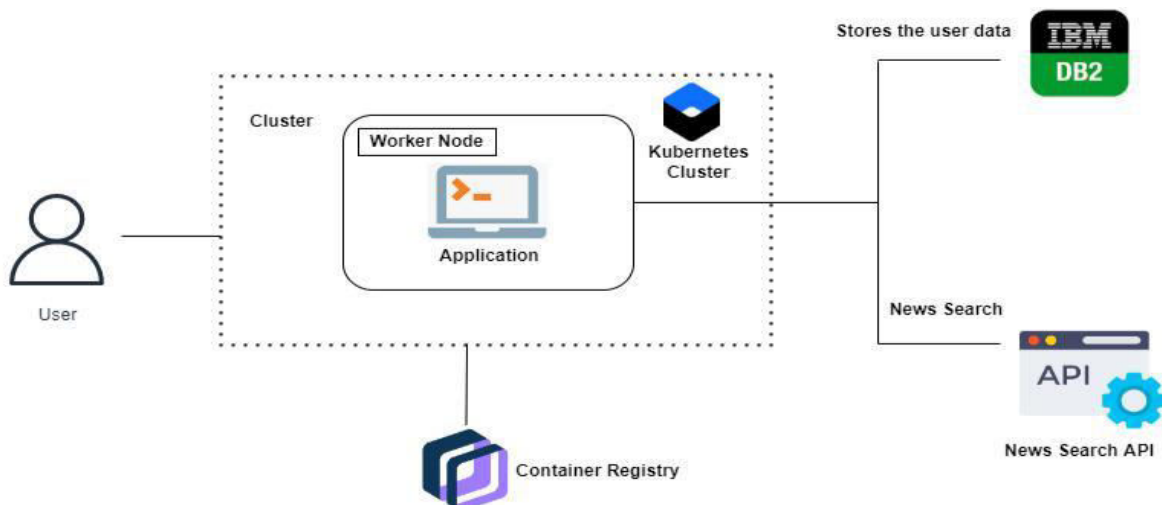
<b>FR No.</b>	<b>Non-Functional Requirement</b>	<b>Description</b>
NFR-1	<b>Usability</b>	For getting news he/she must need to search the news in the given search bar, Which will collect the news according to his search from the online sources like blog, e-news paper,Articles etc. Gathered news will be shown to the user in short lines/ summary manner. Along with news Pictures, hyperlinks for the blog, article Or original publisher of the news is given
NFR-2	<b>Security</b>	User's data for the Registration and data collected from his activity are maintained in high security and must be used for only product improvement. Disclosing of his/her detail is considered as a crime hence they can take any legal actions against the service provider.
NFR-3	<b>Reliability</b>	News gathered and presented to the user are from only authorized publishers, and the sources are authentic. User can relay on this app for original news .
NFR-4	<b>Performance</b>	24*7 service with best cloud service, There will be no performance issue to the users.
NFR-5	<b>Availability</b>	Can download from Playstore and online websites.
NFR-6	<b>Scalability</b>	Hence it is cloud based system Scalability will be no issue because Scalability is the prior feature of the cloud based services.

## 5. PROJECT DESIGN 5.1 Data Flow Diagrams

Data Flow Diagram:



## 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 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 Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		High	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)	log in	USN-6	As a user, I can access the application through website	I can access my account / dashboard	High	
Customer Care Executive	24x7 service	USN-7	As a User, I may need help in using the app/website	Any help need	High	
Administrator	Owner/Chief executive	-	He/she will access the data in the app	Product behaviour/improvement purposes	Low	

## 6. PROJECT PLANNING & SCHEDULING 6.1 Sprint Planning & Estimation

TITLE	DESCRIPTION	DAT E
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers, research publications etc.	2 Sept 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	10 Sept 2022

Ideation Brain Storming	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	15 Sept 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	23 Sept 2022
Problem Solution Fit	Prepare problem solution fit document	28 Sept 2022
Solution Architecture	Prepare solution architecture document.	5 October 2022
Customer Journey	Prepare the customer journey maps to	7 October 2022

	understand the user interactions & experiences with the application (entry to exit).	
Solution Requirement	Prepare the functional requirement document	8 October 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	10 October 2022
Technology Architecture	Prepare the technology architecture diagram.	15 October 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	26 October 2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS..

## 6.2 Sprint Delivery Schedule

### Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
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
Sprint-2	Confirmation Email	USN-2	As a user, I will receive confirmation email once I have registered for the application	2	High
Sprint-3	User profile	USN-3	Once the registration done,a separate profile will be created for a user and they can access the information securely from their profile.	2	High
Sprint-4	Search the information	USN-4	After the profile creation as a user,I can search the news information to be needed.	2	High
Sprint-5	Category	USN-5	A user can search the information by selecting their category like sports,food,politics,weather etc...	2	High

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-6	Location	USN-6	As per user location or selected location by user,the information will be feeded.	1	Medium
Sprint-7	Language	USN-7	As a user,I can read or view the information as my needed language.	2	High

## 6.3 Reports from JIRA

## 7. CODING & SOLUTIONING

Forgot password-

Forpass.html

```
<!doctype html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<meta name="description" content="">
```

```
<meta name="author" content="Mark Otto, Jacob Thornton, and
Bootstrap contributors">

<meta name="generator" content="Hugo 0.84.0">
<title>Sign In</title>

<link rel="canonical" href="https://getbootstrap.com/docs/5.0/examples/sign-
in/">

<link          href="https://getbootstrap.com/docs/5.0/assets/css/docs.css"
rel="stylesheet">

<!-- Bootstrap core CSS -->

<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
rel="stylesheet"                                integrity="sha384-
iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+
fzT" crossorigin="anonymous">

<!-- Favicons -->

<link rel="apple-touch-icon" href="/docs/5.0/assets/img/favicons/apple-touch-
icon.png" sizes="180x180">

<link rel="icon" href="/docs/5.0/assets/img/favicons/favicon-32x32.png"
sizes="32x32" type="image/png">

<link rel="icon" href="/docs/5.0/assets/img/favicons/favicon-16x16.png"
sizes="16x16" type="image/png">

<link rel="manifest" href="/docs/5.0/assets/img/favicons/manifest.json">

<link rel="mask-icon" href="/docs/5.0/assets/img/favicons/safari-pinned-
tab.svg" color="#7952b3">

<link rel="icon" href="/docs/5.0/assets/img/favicons/favicon.ico">

<meta name="theme-color" content="#7952b3">
```



```
<style>

.bd-placeholder-img {
  font-size: 1.125rem;
  text-align: middle;
  -webkit-user-select: none;
  -moz-user-select: none;
  user-select: none;
}

@media (min-width: 768px) {
  .bd-placeholder-img-lg {
    font-size: 3.5rem;
  }
}

</style>
```

```
<!-- Custom styles for this template -->

<link href="static/sheets/signin.css" rel="stylesheet">
<link href="static/sheets/colors.css" rel="stylesheet">

</head>

<body class="text-center">

  <nav class="navbar navbar-dark bg-dark fixed-top">
    <div class="container-fluid">
      <a class="navbar-brand" href="#">News Tracker</a>
```

```
        <button class="navbar-toggler" type="button" data-bs-  
toggle="offcanvas" data-bs-target="#offcanvasDarkNavbar" data-bs-  
controls="offcanvasDarkNavbar">
```

```
        <span class="navbar-toggler-icon"></span>
```

```
</button>
```

```
<div class="offcanvas offcanvas-end text-bg-dark" tabindex="-1"  
id="offcanvasDarkNavbar" aria-labelledby="offcanvasDarkNavbarLabel">
```

```
<div class="offcanvas-header">
```

```
<h5 class="offcanvas-title"  
id="offcanvasDarkNavbarLabel">Profile</h5>
```

```
<button type="button" class="btn-close btn-close-white" data-bs-  
dismiss="offcanvas" aria-label="Close"></button>
```

```
</div>
```

```
<div class="offcanvas-body">
```

```
<ul class="navbar-nav justify-content-end flex-grow-1 pe-3">
```

```
<li class="nav-item">
```

```
<a class="nav-link active" aria-current="page"  
href="home.html">Home</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link" href="base.html">Fetch News</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link" href="about.html">About Us</a>
```

```
</li>
```

```
<li class="nav-item">
```

```
<a class="nav-link" href="signin.html">Sign In</a>
```

```
</li>
```

```

        <li class="nav-item">
            <a class="nav-link" href="signup.html">Sign Up</a>
        </li>
    </div>
</div>
</div>
</nav>

<main class="form-signin ">
    <form action = "{ { url_for("getUser") } }" method="POST">
        
        <h1 class="h3 mb-3 fw-normal white">Change your password</h1>

        <div class="form-floating">
            <input type="email" class="form-control" id="floatingInput" name =
"uname" placeholder="name@example.com">
            <label for="floatingInput">Email address</label>
        </div>

        <br>

        <button class="w-100 btn btn-lg btn-warning btn-outline-warning"
type="submit"><a href="static/templates/changepass.html">Change
Password</a></button><br><br>

    </form>
</main>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.j
s" integrity="sha384-
u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOm
Mi466C8" crossorigin="anonymous"></script>

```

```
<script src="https://cdn.jsdelivr.net/npm/@docsearch/js@3"></script>
```

```
<script  
src="https://cdn.jsdelivr.net/npm/@stackblitz/sdk@1/bundles/sdk.umd.js"></sc  
ript>
```

```
<script src="/docs/5.2/assets/js/docs.min.js"></script>
```

```
<script>
```

```
document.querySelectorAll('.btn-edit').forEach(btn => {  
  btn.addEventListener('click', event => {  
    const htmlSnippet = event.target.closest('.bd-code-  
snippet').querySelector('.bd-example').innerHTML
```

```
    const classes = Array.from(event.target.closest('.bd-code-  
snippet').querySelector('.bd-example').classList).join(' ')
```

```
    const jsSnippet = event.target.closest('.bd-code-snippet').querySelector('.btn-  
edit').getAttribute('data-sb-js-snippet')
```

```
    StackBlitzSDK.openBootstrapSnippet(htmlSnippet, jsSnippet, classes)  
  })  
})
```

```
StackBlitzSDK.openBootstrapSnippet = (htmlSnippet, jsSnippet, classes) => {
```

```
  const markup = `<!doctype html>
```

```
<html lang="en">
```

```
<head>
```



[illegible]

[illegible]





## **10. ADVANTAGES & DISADVANTAGES Advantages:-**

- ❖ In this app news is already categorization.
- ❖ Easily accessible and portable.
- ❖ Better user experience.
- ❖ Apps help you convert visitors to loyal readers.
- ❖ Minute by minute updates of news.
- ❖ This app helps you to get local news updates instantly.
- ❖ To explore and discover trending news and topics.
- ❖ News feeds for you based on your interest.

### **Disadvantages:-**

- ❖ Some apps demand premium subscription from user.
- ❖ Occurance of Advertisement disturb the user.
- ❖ Sometimes the news gives brief information.
- ❖ Prevalance of fake and uncertain news can confuse the user lead to misconception.
- ❖ Fake news may mislead the readers.

## **11. CONCLUSION**

We explored the feasibility of recognising 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 recognised 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 behaviour 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 personalisation 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 personalisation of news apps.

### 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 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>

  </div>
```

```
<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**

<https://github.com/IBM-EPBL/IBM-Project-28670-1660115021>

## **Project demo link**

[https://drive.google.com/file/d/1bWBYLUBYE-hEPxPK\\_SRh0PwB1FcBOfZ9/view?usp=sharing](https://drive.google.com/file/d/1bWBYLUBYE-hEPxPK_SRh0PwB1FcBOfZ9/view?usp=sharing)