NEWS TRACKER APPPLICATION

USING CLOUD

A Project report submitted in partial fulfilment of 7th semester in degree of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

Team ID:

ABIRAMI A	810019106003
DHESIKA K	810019106021
GUNAMUKHI S	810019106025
HARIPRABHA S	810019106028

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING,

BIT CAMPUS,THIRUCHIRAPALLI

ANNA UNIVERSITY: CHENNAI 600025

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

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

SURVEY ON PAPERS RELATED TO NEWS TRACKER APPLICATION
NAME : Exploring Mobile News Reading Interactions for News app Personalization
AUTHOR : MARIOS CONSTANTINIDES
PUBLISHED YEAR : 2015

BASIC DESCRIPTION:

The 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.

HIGHLIGHTS OF THE PROJECT :

Mobile news access perfectly complements the continuously updating, 24-hour nature of digital news services. But if users are now never out of range of the news, they need more than ever for that access to be adaptive and personalised. Personalised news services are already able to help people find news that is relevant to them, to recommend the right news to the right users, and to help users keep abreast of news by aggregation over multiple sources.

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.This adaptively is achieved through several methods [5] including: news content personalisation by pushing filtered articles predicted to match the user's interests; adaptive news browsing by changing the order of news categories; contextual news access by offering users access to additional information related to the news they are reading; and news aggregation, by automatically identifying main news topics emerging from multiple sources.This previous work on adaptively in digital news access has focused on recommendation of news content.

OVERVIEW OF THE PROJECT:

The Person who likes to be informed About the Latest stories and any updates to stories he or she is following, usually reading the news for up to 10 minutes at a time and several times a day at intervals, for example, when travelling. Due to her limited time she prefers to extract the important bits of a story (i.e. reading by skimming).

User: A person who likes to catch up on the day's news, preferably at home.

He likes an in-depth analysis of the stories he reads and will read at length to fully understand the story (i.e. a detailed reading).He usually reads the news once a day, spending more than 10 minutes to get through all the stories of interest and likes being informed on a variety of topics.Personalized news recommendations systems,

recommend news from enormous collection of news articles based user preferences and interest that help to optimize the user reading ability in the targeted news.

RESULT :

User interests and preferences can be identified in various ways, e.g. using social networks [71], [93] (section V-A), mobile apps [3], [15] (section V-C), Google's news personalization [16], [75], constructing user profiles by click behavior or history logs [30], [75], mobility based news personalization [114] etc. Existing personalized news recommendation (PNR) systems proposed in a number of research articles for effective recommendations of news as per user's need, can broadly be classified based on approaches adapted.

CONCLUSION :

We Explore the feasibility of recognizing patterns of news reading interactions and evaluated three adaptive interface designs for different news reader types. We show that from their interaction log, a specific user can be recognized as one of three kinds. The reader types emerging from the online survey are well defined and distinct. The evaluation of the three variant interfaces suggests that different news reader types need different user interfaces. We have Demonstrated a method for monitoring users' news reading behavior and inferring news reader type from it. In the future we will further explore the design of adaptive interfaces, in order to be in a position to demonstrate a complete Adaptive Mobile News Framework Providing Automatic Personalization Of the News App.

DISADVANTAGES OF THE PROJECT :

- Less Security
- Internet Connection Required

LIMITATIONS OF THE PROJECT :

- The site could crash if a lot of people try accessing it at the same time.
- There are some news sites available online access to readers only after they pay for daily news.

REFERENCES :

1.Ofcom, News consumption in the UK, Public report (2014).

2. Pew Research Centre, The Future of Mobile News, Public report (2012).

3. Reuters Institute, Tracking the future of news, Public Report (2014)

4. Bill sus, D. & Pazzani, M. A hybrid user model for news story classification. Springer Vienna (1999),
99108.

5. Bill sus, D. & Pazzani, M. Adaptive news access. In The Adaptive Web, Springer Berlin Heidelberg (2007)

2.3 Problem Statement Definition

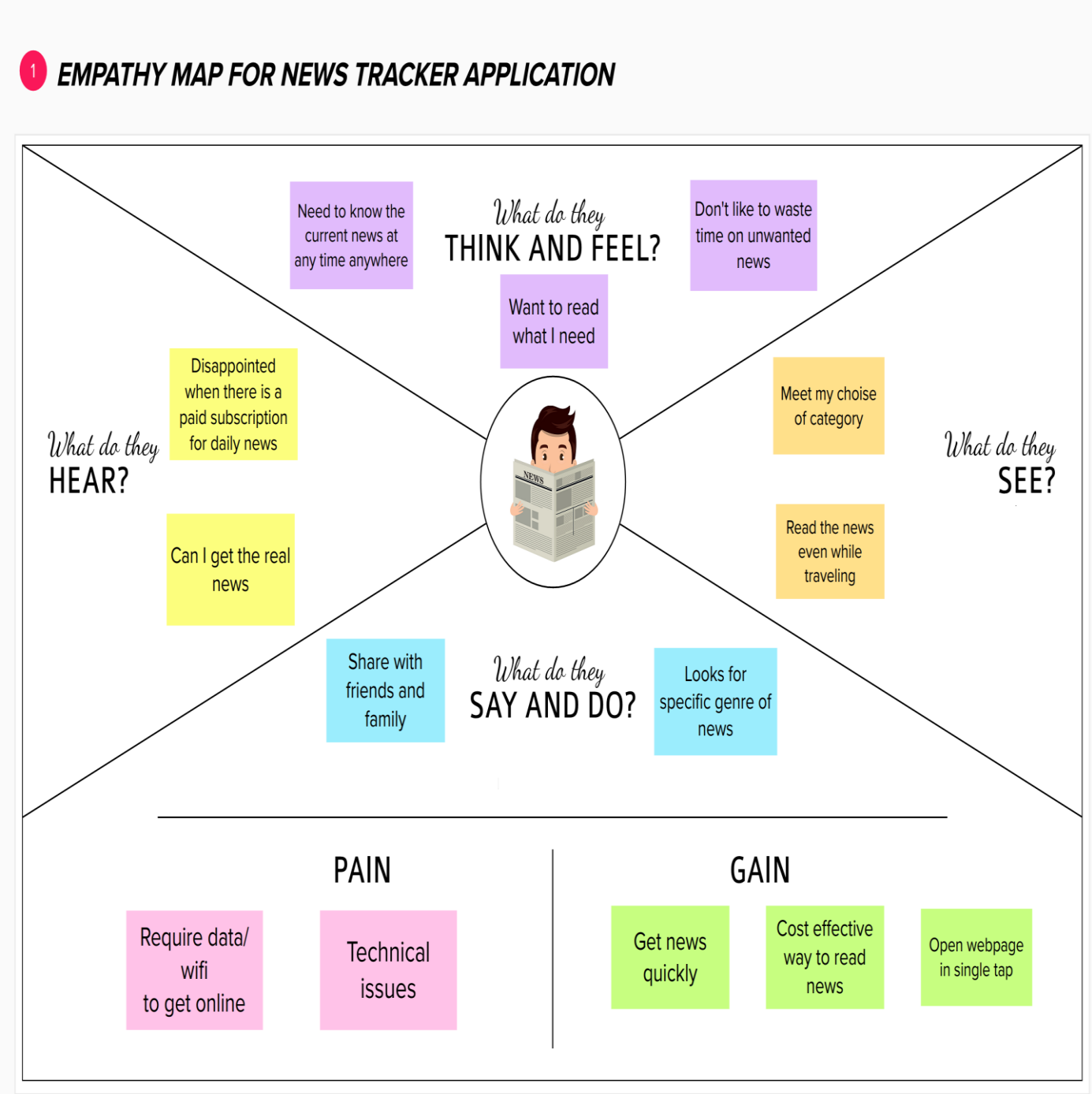
- Ram is a busy manager who wants a way to read the news he needs because he doesn't want to waste his time on unwanted news.

- Arjun is a bus conductor who needs a way to read the daily news on his free time because he can't able to spend his working hours on reading news.

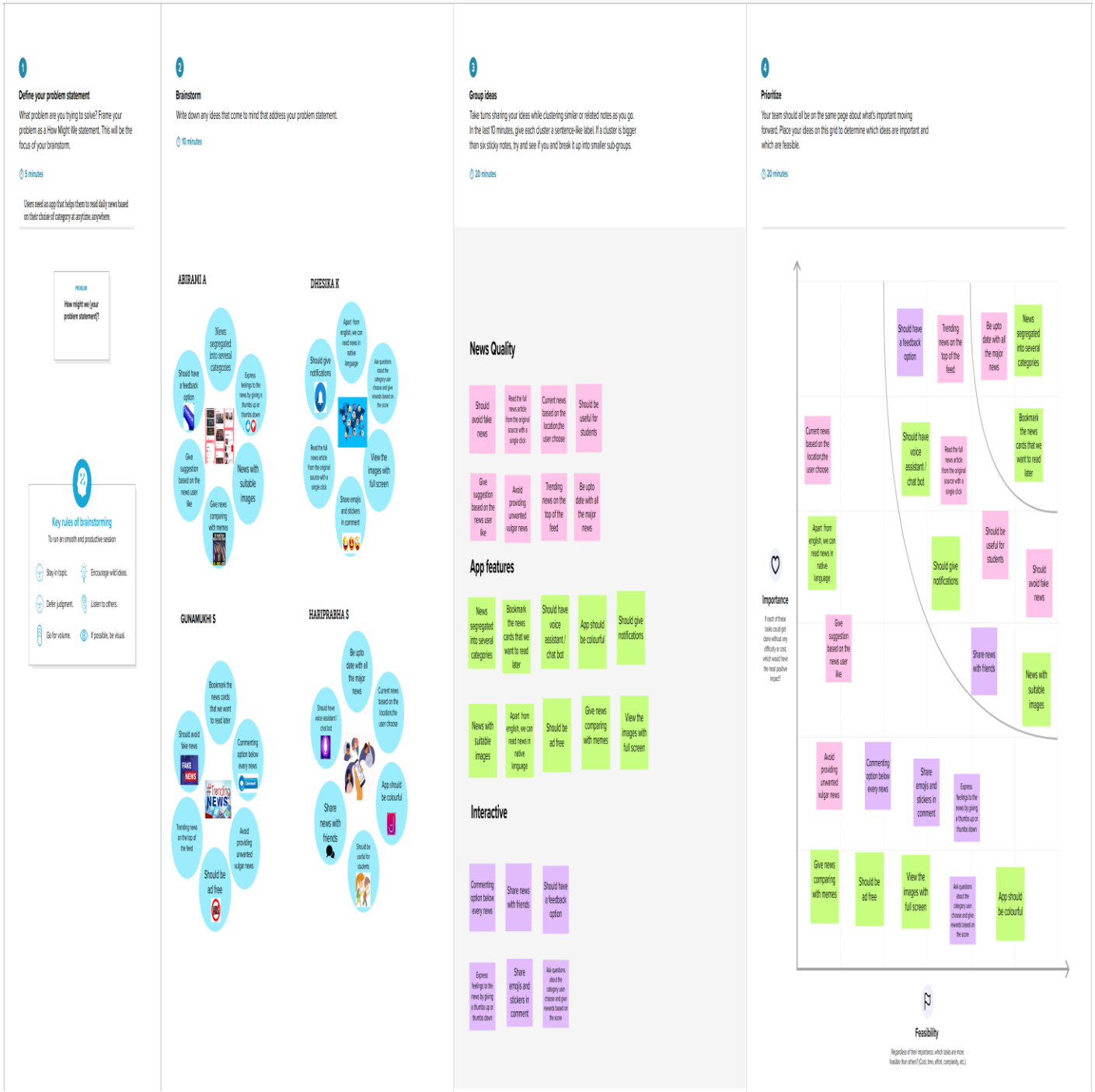
- Meera is an Indian woman working in USA needs a way to read the news about her motherland so that she gets daily updates about her motherland.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



3.3 Proposed Solution

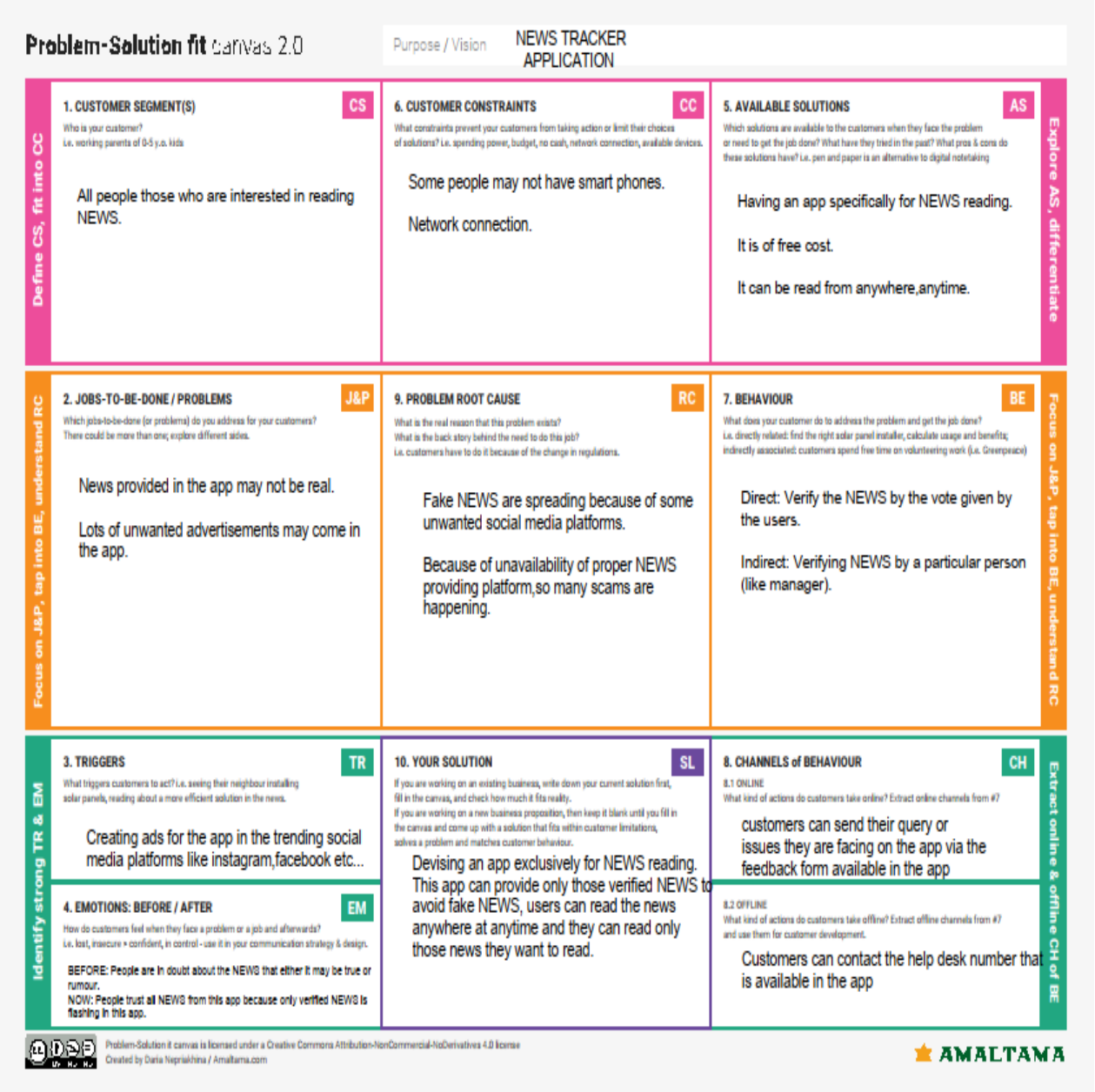
Project Design Phase-I
Proposed Solution

Date	30 September 2022
Team ID	PNT2022TMID32390
Project Name	NEWS TRACKER APPLICATION
Maximum Marks	2 Marks

Proposed Solution :

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The objective of this project is to provide an app that helps the user to read the daily news based on their choice of category at anytime, anywhere.
2.	Idea / Solution description	We intend to collaborate our application with the cloud in order to provide the news that belong to the user's category of interest. The database will have all the details and the user can search the news by using a search bar.
3.	Novelty / Uniqueness	This app pulls news sources from all across the world, so we will likely be able to find our favourite news. This app can set up a daily briefing to give us the most important stories of the day.
4.	Social Impact / Customer Satisfaction	Helps news readers to get required information within less time. Provide a feedback form to the user to know their need in order to improve the customer satisfaction. Provide frequent updates to overcome and fix the bugs.
5.	Business Model (Revenue Model)	We can include the ads in portal. We can provide subscription option to users to have premium membership.
6.	Scalability of the Solution	To increase the number of users we shall advertise our app in some other popular websites. When the load on the app increases, we can scale up the software to meet the needs. By focusing on application testing we can find bugs and fix the problems.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

Project Design Phase-II
Solution Requirements (Functional & Non-functional)

Date	03 October 2022
Team ID	PNT2022TMID32390
Project Name	NEWS TRACKER APPLICATION
Maximum Marks	4 Marks

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 Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Help Users Find Content With Categories	If the App Readers want to Access Different types of Content,We Need to Provide an Easy way to them. Adding 'Content Categories' to the Main App Menu is the Perfect way to do this.
FR-4	Make Ads User-Friendly	The Advertising Methods,Such as Native in -Feed Advertisements are UX-Friendly that helps us Generate more Revenue, and Provide a Better user Experience to the App Readers
FR-5	Increase Subscriptions By Allowing Registrations	Including a Feature of 'Create Accounts' in the News Apps Enable the users to Register on the App and use their Accounts to read the Most Happening Content First. It will help us to Gather the Most useful's user data , Such as the App Readers Loyalty and the Type of Content they are interested in.
FR-6	Display the Most Relevant Story First	It is an important Step to Display the Most Recent or top Story first to the App users.

Non-functional Requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	People Who are all Not Able to Buy the News Paper can Use Our App. It is Cost Effective.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

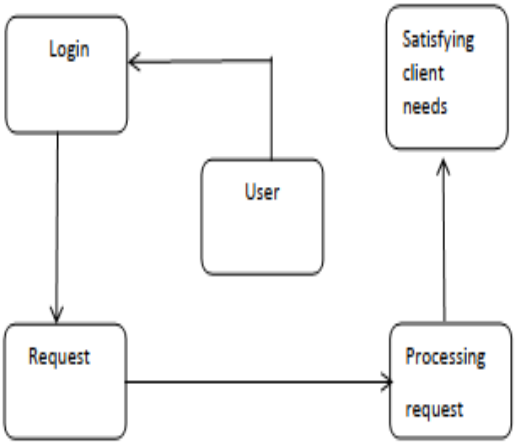
Project Design Phase-II
Data Flow Diagram &User Stories

Date	15 October 2022
Team ID	PNT2022TMID32390
Project Name	NEWS TRACKER APPLICATION
Maximum Marks	4 Marks

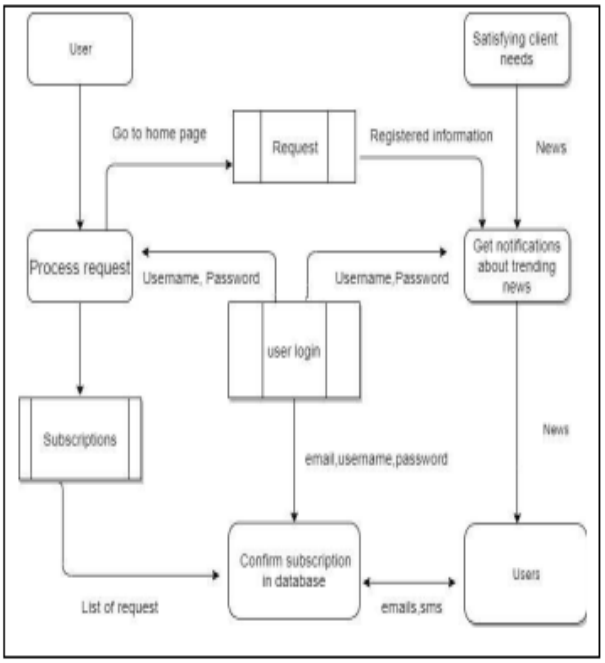
Data Flow Diagrams:

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.

Example: (Simplified)



Example: DFD Level 0 (Industry Standard)

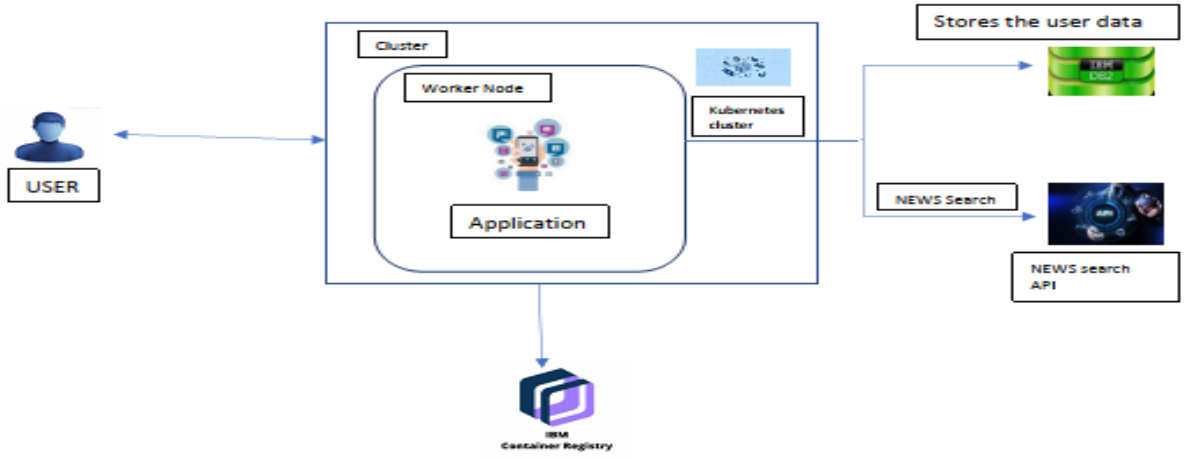


5.2 Solution & Technical Architecture

Project Design Phase-I
Solution Architecture

Date	19 September 2022
Team ID	PNT2022TMID32390
Project Name	NEWS TRACKER APPLICATION
Maximum Marks	4 Marks

Solution Architecture Diagram:

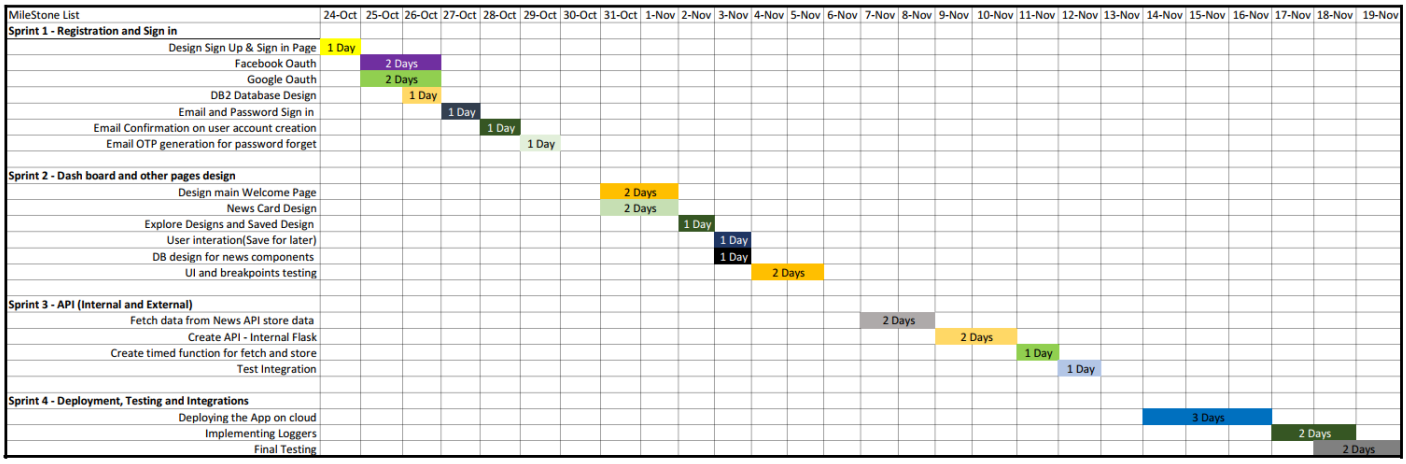


5.3 User Stories

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 datas in the app	Product behaviour/Improvement purposes	Low	

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation



6.2 Sprint Delivery Schedule

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	24 October 2022
Team ID	PNT2O22TMID32390
Project Name	Project – News Tracker Application
Maximum Marks	8 Marks

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	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Abirami ,Hariprabha
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Gunamukhi, Dhesika
Sprint-1		USN-3	As a user, I can register for the application through Facebook	2	Low	Abirami, Gunamukhi
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Gunamukhi
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Dhesika, Hariprabha
Sprint-2	Dashboard	USN-6	As a user I should be able to navigate and access all the features hassle free	2	High	Abirami, Dhesika
Sprint-2	Layout	USN-7	As a user I should be able to access the portal with different devices with the same comfort	2	High	Gunamukhi, Hariprabha
Sprint-3	Data Store and retrieval	USN-8	Get Data from API and store as JSON in DB2	3	High	Abirami, Hariprabha
Sprint-3		USN-9	Get bin data from API and store in DFS	2	High	Dhesika, Abirami
Sprint-4	User Segregation and data access	USN-10	As a CC executive I should be able to uniquely identify the customer and offer help	1	Low	Dhesika

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/signin/">

    <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="sha384iYQeCzEYFbKjA/T2uDLTpkwGzCiq6soy8tYaI1GyVh/UjpbCx/TYkiZhlZB6+ fzT" crossorigin="anonymous">
    <!-- Favicons -->
<link rel="apple-touch-icon" href="/docs/5.0/assets/img/favicons/apple-touchicon.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-pinnedtab.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-anchor: 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-bstoggle="offcanvas" data-bs-target="#offcanvasDarkNavbar" ariacontrols="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-bsdismiss="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">
    <img class="mb-4" src="static/Images/house-user-solid.svg" alt="" width="72" height="57">
    <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">Change Password</button>

</form>

</main>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-u1OknCvxWvY5kfmNBILK2hRnQC3Pr17a+RTT6rIHI7NnikvbZlHgTPOOmMi466C8" 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"></script>

<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('.btnedit').getAttribute('data-sb-js-snippet')
    StackBlitzSDK.openBootstrapSnippet(htmlSnippet, jsSnippet, classes)
  })
 })

 StackBlitzSDK.openBootstrapSnippet = (htmlSnippet, jsSnippet, classes) => {
const markup = `<!doctype html>
<html lang="en">
 <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link
href="https:\/\/cdn.jsdelivr.net\/npm\/bootstrap@5.2.1\/dist\/css\/bootstrap.min.c
ss" rel="stylesheet">
  <link href="https://getbootstrap.com/docs/5.2/assets/css/docs.css"
rel="stylesheet">
  <title>Bootstrap Example</title>
  <${'script'}
src="https:\/\/cdn\.jsdelivr\.net\/npm\/bootstrap@5\.2\.1\/dist\/js\/bootstrap\.bun
dle\.min\.js"></${'script'}>
 </head>
 <body class="p-3 m-0 border-0 ${classes}">

  <!-- Example Code -->
${htmlSnippet.replace(/^/gm, '    ')}
  <!-- End Example Code -->
 </body>
```

</html>`

const jsSnippetContent = jsSnippet ? '\/\/ NOTICE!!! Initially embedded in our docs this JavaScript\n\/\/ file contains elements that can help you create reproducible\n\/\/ use cases in StackBlitz for instance\.\n\/\/ In a real project please adapt this content to your needs\.\n\/\/ \u002b\n\n\/*!\n * JavaScript for Bootstrap\u0027s docs \(https:\/\/getbootstrap\.com\/\)\n * Copyright 2011\2022 The Bootstrap Authors\n * Copyright 2011\-2022 Twitter, Inc\.\n * Licensed under the Creative Commons Attribution 3\.0 Unported License\.\n * For details, see https:\/\/creativecommons\.org\/licenses\/by\/3\.0\/\.\n *\/\n\n\/* global bootstrap: false *\/\n\n\(\(\) =\u003e \{\n \u0027use strict\u0027\n\n \/\/ \-\-\-\-\-\-\-\-\n \/\/ Tooltips\n \/\/ \-\-\-\-\-\-\-\-\n \/\/ Instantiate all tooltips in a docs or StackBlitz page\n document\.querySelectorAll\(\u0027\[data\-bs\toggle=\u0022tooltip\u0022\]\u0027\)\n \.forEach\(tooltip =\u003e \{\n new bootstrap\.Tooltip\(tooltip\)\n \}\)\n\n \/\/ \-\-\-\-\-\-\-\-\n \/\/ Popovers\n \/\/ \\-\-\-\-\-\-\-\n \/\/ Instantiate all popovers in a docs or StackBlitz page\n document\.querySelectorAll\(\u0027\[data\-bs\toggle=\u0022popover\u0022\]\u0027\)\n \.forEach\(popover =\u003e \{\n new bootstrap\.Popover\(popover\)\n \}\)\n\n \/\/ \-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\\-\-\-\-\-\-\-\-\-\-\-\-\-\n \/\/ Toasts\n \/\/ \-\n \/\/ Used by \u0027Placement\u0027 example in docs or StackBlitz\n const toastPlacement = document\.getElementById\(\u0027toastPlacement\u0027\)\n if \(toastPlacement\) \{\n document\.getElementById\(\u0027selectToastPlacement\u0027\)\.addEventListener\(\u0027change\u0027, function \(\) \{\n if \(!toastPlacement\.dataset\.originalClass\) \{\n toastPlacement\.dataset\.originalClass = toastPlacement\.className\n \}\n\n toastPlacement\.className = `\$\{toastPlacement\.dataset\.originalClass\} \$\{this\.value\}`\n \}\)\n \}\n\n \/\/ Instantiate all toasts in a docs page only\n document\.querySelectorAll\(\u0027\.bd\-example \.toast\u0027\)\n \.forEach\(toastNode =\u003e \{\n const toast = new

bootstrap\.Toast\(toastNode, \{\n autohide: false\n \}\)\n\n
toast\.show\(\)\n \}\)\n\n \/\/ Instantiate all toasts in a docs page only\n const
toastTrigger = document\.getElementById\(\u0027liveToastBtn\u0027\)\n const
toastLiveExample = document\.getElementById\(\u0027liveToast\u0027\)\n if
\(toastTrigger\) \{\n toastTrigger\.addEventListener\(\u0027click\u0027, \(\)
=\u003e \{\n const toast = new bootstrap\.Toast\(toastLiveExample\)\n\n
toast\.show\(\)\n \}\)\n \}\n\n \/\/ \-\\-
\-\n \/\/ Alerts\n \/\/ \-\n \/\/ Used
in \u0027Show live toast\u0027 example in docs or StackBlitz\n const
alertPlaceholder =
document\.getElementById\(\u0027liveAlertPlaceholder\u0027\)\n const
alertTrigger = document\.getElementById\(\u0027liveAlertBtn\u0027\)\n\n
const appendAlert = \(message, type\) =\u003e \{\n const wrapper =
document\.createElement\(\u0027div\u0027\)\n wrapper\.innerHTML = \[\n
`\u003cdiv class=\u0022alert alert\-\$\{type\} alert\-dismissible\u0022
role=\u0022alert\u0022\u003e`,\n `
\u003cdiv\u003e\$\{message\}\u003c\/div\u003e`,\n \u0027 \u003cbutton
type=\u0022button\u0022 class=\u0022btn\-close\u0022 data\-
bs\dismiss=\u0022alert\u0022 aria\-
label=\u0022Close\u0022\u003e\u003c\/button\u003e\u0027,\n
\u0027\u003c\/div\u003e\u0027\n
\]\.join\(\u0027\u0027\)\n\n alertPlaceholder\.append\(wrapper\)\n
\}\n\n if \(alertTrigger\) \{\n
alertTrigger\.addEventListener\(\u0027click\u0027, \(\) =\u003e \{\n
appendAlert\(\u0027Nice, you triggered this alert message!\u0027,
\u0027success\u0027\)\n \}\)\n \}\n\n \/\/ \-
\-\-\-\-\-\-\-\-\n \/\/ Checks \u0026 Radios\n \/\/ \-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\\-
\-\-\-\-\-\-\-\-\n \/\/ Indeterminate checkbox example in docs and StackBlitz\n
document\.querySelectorAll\(\u0027\.bd\-example\-indeterminate
\[type=\u0022checkbox\u0022\]\u0027\)\n \.forEach\(checkbox =\u003e \{\n
if \(checkbox\.id\.includes\(\u0027Indeterminate\u0027\)\) \{\n
checkbox\.indeterminate = true\n \}\n \}\)\n\n \/\/ \-\-\-\-\-\-\-\-\-\-\-\-\-\-\-
\-\-\-\-\-\-\-\-\-\-\-\-\-\-\-\n \/\/ Links\n \/\/ \-
\-\-\-\-\-\-\n \/\/ Disable empty links in docs examples
only\n document\.querySelectorAll\(\u0027\.bd\-content

\[href=\u0022#\u0022\]\u0027\)\n \.forEach\(link =\u003e \{\n link\.addEventListener\(\u0027click\u0027, event =\u003e \{\n event\.preventDefault\(\)\n \}\)\n \}\)\n\n \/\/ \- \\-\-\-\-\-\-\-\-\-\-\-\-\n \/\/ Modal\n \/\/ \- \-
\-\-\n \/\/ Modal \u0027Varying modal content\u0027 example in docs and StackBlitz\n const exampleModal = document\.getElementById\(\u0027exampleModal\u0027\)\n if \(exampleModal\) \{\n exampleModal\.addEventListener\(\u0027show\.bs\.modal\u0027, event =\u003e \{\n \/\/ Button that triggered the modal\n const button = event\.relatedTarget\n \/\/ Extract info from data\-bs\-* attributes\n const recipient = button\.getAttribute\(\u0027data\-bs\-whatever\u0027\)\n\n \/\/ Update the modal\u0027s content\.\n
const modalTitle = exampleModal\.querySelector\(\u0027\.modal\-title\u0027\)\n const modalBodyInput = exampleModal\.querySelector\(\u0027\.modal\-body input\u0027\)\n\n modalTitle\.textContent = `New message to \$\{recipient\}`\n modalBodyInput\.value = recipient\n \}\)\n \}\n\n \/\/ \-\\-\n \/\/ Offcanvas\n \/\/ \-\-\-\-\-\-\-\\-\n \/\/ \u0027Offcanvas components\u0027 example in docs only\n const myOffcanvas = document\.querySelectorAll\(\u0027\.bd\-example\-offcanvas \.offcanvas\u0027\)\n if \(myOffcanvas\) \{\n myOffcanvas\.forEach\(offcanvas =\u003e \{\n offcanvas\.addEventListener\(\u0027show\.bs\.offcanvas\u0027, event =\u003e \{\n event\.preventDefault\(\)\n \}, false\)\n \}\)\n \}\n\}\)\(\)\n' : null

const project = { files: {

'index.html': markup,

'index.js': jsSnippetContent

},

title: 'Bootstrap Example', description: `Official example from ${window.location.href}`, template: jsSnippet ? 'javascript' : 'html', tags: ['bootstrap']

```
  }

  StackBlitzSDK.openProject(project, { openFile: 'index.html' })
 }
</script>
 </body>
</html>
```

8. TESTING

8.1 Test Cases

Test case

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	-	Abishek
Sign up	Functional	Login page	Verify user has the option to sign up	Can visible	Yes visible	pass	Successful	-	Abishek
Forgot password	Functional	Login page	Verify user has the option to forgot password	Yes the option is available	Option is available	pass	Successful	-	Manikand-an

Result:

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

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 Advertisment disturb the user.
- ✞ Sometimes the news gives brief information.
- ✞ Prevailance 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.tokener 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,resend_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.tokener 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"}        respone =
requests.request(
         "GET", url=self.__url, headers=self.__headers, params=querystring)
respone = respone.json()        retArr = []        for x in respone["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"}          respone =
requests.request("GET", url=self.__url, headers=self.__headers, params=querystring)        respone =
respone.json()      retArr = []      for x in respone["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">

         <img id="ham_icon" src="../assets/hamburger 1.svg" alt="ham_icon">

         <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> -->
        <img id="user_icon" src="../assets/user 1.svg" alt="user_icon">
        <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">
        <img class="bookmark" src="../assets/bookmark-solid.svg" alt="" />
        <a class="news_cont" href="./news.html">
          <div class="img_cont">
            <img src="../assets/pexels-cottonbro-3944418 1.png" alt=""
class="news_thumbnail">
          </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>
</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">
```

```
        <img id="ham_icon" src="../assets/hamburger 1.svg" alt="ham_icon">
        <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> -->
        <img id="user_icon" src="../assets/user 1.svg" alt="user_icon">
        <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">
            <img class="bookmark" src="../assets/bookmark-regular.svg" alt=""
/>
            <a class="news_cont" href="./news.html">
                <div class="img_cont">
                    <img src="../assets/pexels-cottonbro-3944418 1.png" alt=""
class="news_thumbnail">
                </div>
                <div class="news_content">
                    <h2 class="news_heading">JSV effect</h2>
                </div>
            </a>
        </div>
        <div class="news_wrapper">
            <img class="bookmark" src="../assets/bookmark-regular.svg" alt="" />
            <a class="news_cont" href="./news.html">
                <div class="img_cont">
                    <img src="../assets/pexels-cottonbro-3944418 1.png" alt=""
class="news_thumbnail">
                </div>
                <div class="news_content">
                    <h2 class="news_heading">JSV effect</h2>
                </div>
            </a>
        </div>
        <div class="news_wrapper">
            <img class="bookmark" src="../assets/bookmark-regular.svg" alt=""
/>
```

```
            <a class="news_cont" href="./news.html">
              <div class="img_cont">
                <img src="../assets/pexels-cottonbro-3944418 1.png" alt="" class="news_thumbnail">
              </div>
              <div class="news_content">
                <h2 class="news_heading">JSV effect</h2>
              </div>
            </a>
          </div>
          <div class="news_wrapper">
            <img class="bookmark" src="../assets/bookmark-regular.svg" alt="" />
            <a class="news_cont" href="./news.html">
              <div class="img_cont">
                <img src="../assets/pexels-cottonbro-3944418 1.png" alt="" class="news_thumbnail">
              </div>
              <div class="news_content">
                <h2 class="news_heading">JSV effect</h2>
              </div>
            </a>
          </div>
        </div>
        <div class="news_wrapper">
          <img class="bookmark" src="../assets/bookmark-regular.svg" alt="" />
          <a class="news_cont" href="./news.html">
            <div class="img_cont">
```

```
                <img src="../assets/pexels-cottonbro-3944418 1.png" alt="" class="news_thumbnail">
            </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/fontawesome/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 registed ?</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>

    <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">
            <img id="ham_icon" src="../assets/hamburger 1.svg" alt="ham_icon">
            <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> -->
            <img id="user_icon" src="../assets/user 1.svg" alt="user_icon">
            <div class="profile_div display_none">
                <h3 class="logout">Logout</h3>
            </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-9449-1659008276

Project demo link

https://drive.google.com/file/d/1KBsRno3EbPechCkOlKaFTL2ef_chjsSq/view?usp=sharing