Project Report

TEAM ID : PNT2022TMID44807

TEAM MEMBERS :

DINESH S (LEADER),
PRADEEP K,
PRAVINKUMAR R,
VIGNESHWARAN G,
KALPANA R.

PROJECT : NEWS TRACKER APPLICATION

10. **FUTURE SCOPE**

11. **11. APPENDIX**

    Source Code

1. INTRODUCTION

1.1 Project Overview

News Tracker is a full stack web application which allows users to register along with their favourite topics, upon login the app displays the news based on the user's interest. The news displayed in the app is based on the New catcher API and Cric buzz API from Rapid API site. A news-sharing app wants to help users find relevant and important news easily every day and also provide explicitly news from that users locality/region which may of help to the user.

1.2 Purpose

Enabling users to view news from anywhere at anytime. It also helps to reduce the time to get information about a specific topic. Also enables a person to get an updated news which may help Business people to make business related decisions quickly and correctly.

2. LITERATURE SURVEY

2.1 Existing problem

Physical newspapers are old fashioned in this digital era. They cost money to buy, can easily be damaged, limited amount of information, not flexible to modifications, poor quality. Sometimes may show irrelevant and updated news.

2.2 Survey

Sl. No	TITLE	AUTHORS	YEAR	TECHNIQUES	MERITS	DEMERITS
1	**Design and Implementation of News Collecting and Filtering System Based on RSS**	Zheng, R., & Zhang, Y.	**2012**	Using RSS to collect News with enhanced search system	System can automatically collect the latest news information from the subscribe site, then parsing and Storing the information into database.	Graphics and photos do not always appear .Posts are easily deformatted or fully erased. RSS might make the process a bit unpleasant.
2	**News Event Detection and Tracking Based on Stream of Online News**	Yajie Qi Li Zhou Huayou Si JianWan Ting Jin	**2017**	Single-pass clustering algorithm for event detection and tracking	Extraction of news content on particular field by searching keywords.	In some cases it can't analyse the keyword brings original news or not
3	**Deep News Event Ranker Based on User Relevant Query**	Kong, X., Kong, Q., Mao, W., & Tang, S.	**2018**	Word embedding technology using Global vector of word representation	Top news will be ranked according to the user query	The model is trained on the cooccurrence matrix of words, which takes a lot of memory for

						storage
4	**Exploring mobile news reading interactions for news app personalisation**	Marios Constantinides, John Dowell, David Johson, Sylvain Malacria	**2015**	1. Identification of news reader types 2. Interaction logging and classification study 3. Deployment and data collection 4. Predicting News reader types 5. Adaptive UI	The adaptive user interface changes according to the type of task you want to perform. This will increase the stability of the system.	The overall code of the website and the size of the app increases if the user interface is adaptive. There is a lot of code to be written for making the user interface adaptive
5	**Detection and Tracking in News Articles**	Sagar Patel, Sanket Suthar, Sandip Patel, Neha Patel	**2015**	1. Pre-processing 2. Tokenization 3. Stemming/Lemmizat - ion 4. Vector Space Model 5. Topic tracking	Allows the computing for a continuous degree of similarities between queries and document	1. Suffers from synonym and polysemy 2. It theoretically assumes that terms are statistically independent.

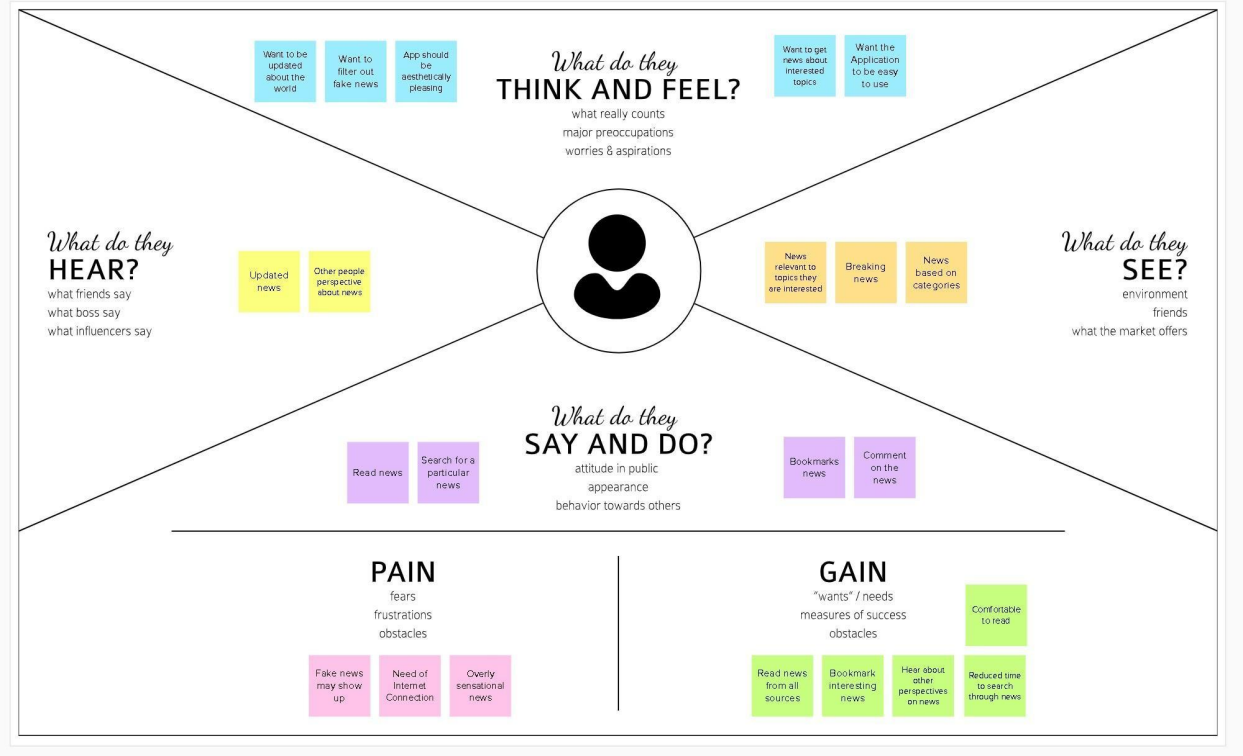
6	**Following the Fed with a News Tracker**	Michael William McCracken	**2012**	The paper is not a technical paper but is essentially a statistical paper on how should one conclude whether the data have come in stronger, weaker or as expected. This is based on the Citi Group U.S Economic Surprise Index.	Tracks whether a core set of economic data series has been coming in under expectations, at expectations, or over expectations	, the surprise indexes tend to have the right sign and be significant: a positive change in the U.S. surprise index (i.e. the U.S. economy doing better than expected) appreciates the U.S. dollar versus the foreign currency, whereas a positive change in the foreign surprise index depreciated the U.S. dollar.

2.3 Problem Statement Definition

Newspaper contains limited, nonuser/reader specific, Location specific news. There are multiple news-sharing apps available which can be used by a single user and are often spammed with notifications. There is also a lot of unwanted news which gets shared. So it may take a lot of time for the user to find the news he/she likes. A news-sharing app wants to help users find relevant and important news easily every day and also provide explicitly news from that users locality/region which may of help to the user.

Vijay is a busy business man who needs to read news on the go without any hassles while travelling because he considers carrying around a physical newspaper a nuisance to him and the people around him. Vijay needs to read needs to read news in such a way that he doesn't have to worry about ever buying physical newspapers or carrying with him everywhere. Something which should fit in the palm of his hands, which he could carry everywhere, access from everywhere, something digital such as an Application hosted on the internet which could be accessed from any device that is connected to the Internet. Such as smartphones and computers.

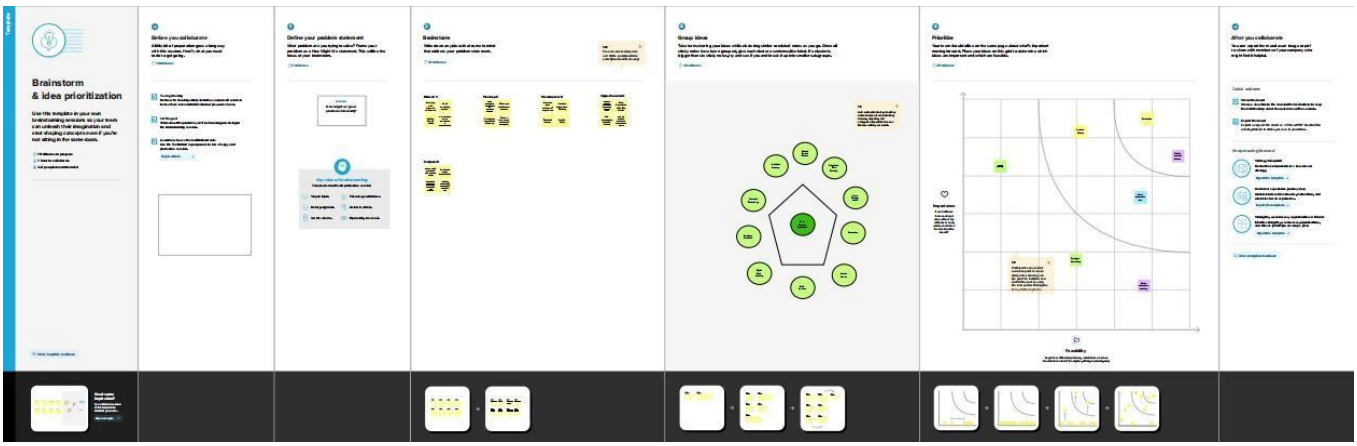
3. IDEATION AND PROPOSED SOLUTION



3.1 Empathy map canvas

3.2 Ideation and Brainstorming

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

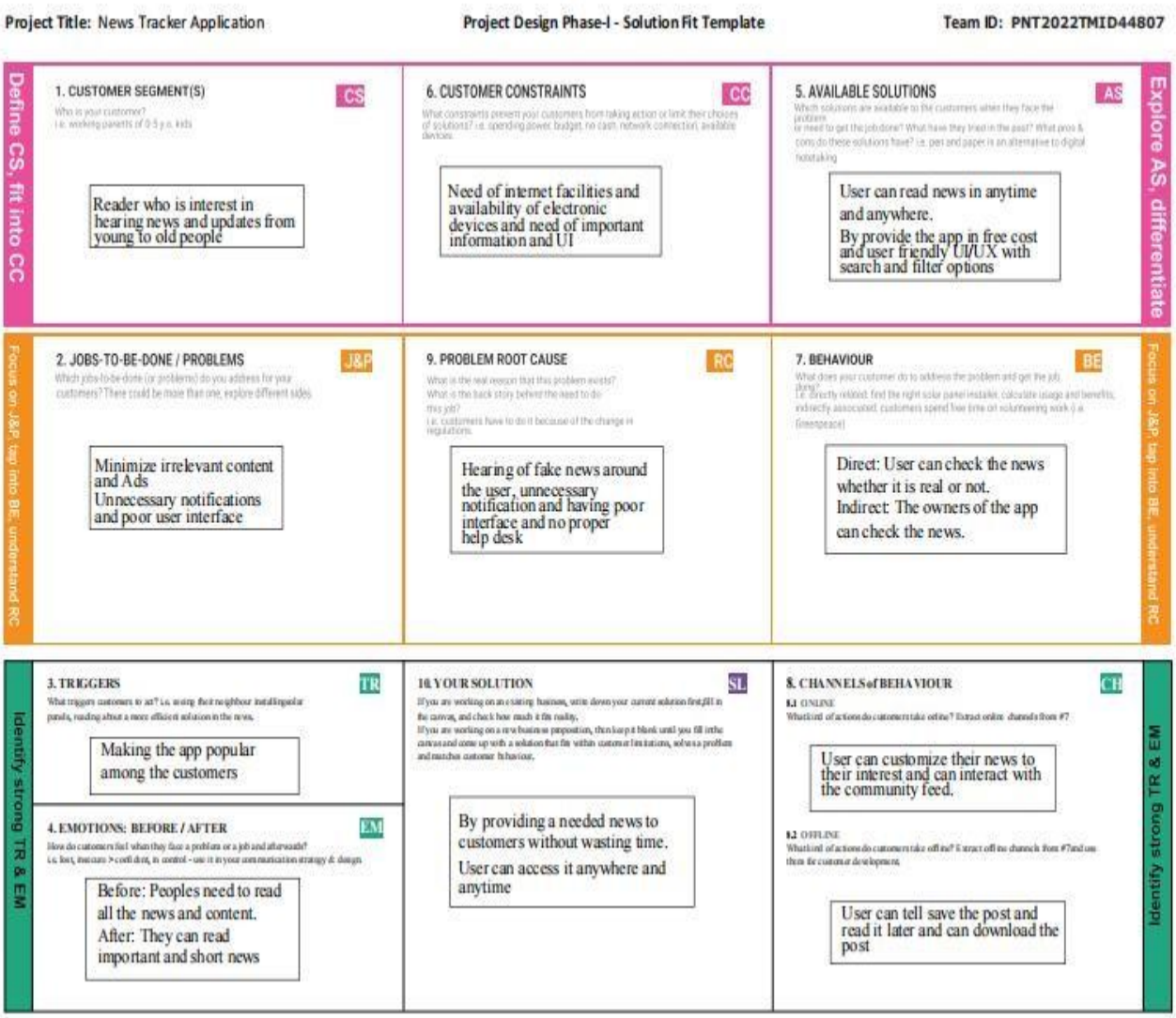


3.3 Proposed Solution

S.NO.	Parameter	Description
1.	Problem Statement (Problem to be solved)	**Statement:** Everyday, a lot of events happen world-wide and we rely on newspapers, television and news articles to get the reliable and trust-worthy information about these events.**Description:** As a result, we created a platform that offers such news from reliable sources worldwide, in an organized and efficient manner.
2.	Idea / Solution description	One platform for all local and worldwide news. Trustworthy and Reliable News. Fast and efficient system. Preventing spread of False information. Data Storage and Backup. Communication.
3.	Novelty / Uniqueness	A cloud computing-based news application that generates news and reports about the happenings around the world using computers and network (Internet). News based on most reliable and trustworthy resources around the world. Developing the Eco- Friendly & sustainability based on centre.

4.	Social Impact / Customer Satisfaction	Cloud computing offers a way to create coordinate, and share information across the globe. The adoption of cloud-based services gives access to a wider range of data and sharing the important information in an efficient way. Our platform eliminates the spread of false news and exposes the injustice and wrongdoings done by false groups. Eliminating the fake news provides better understanding of the real-events happening in the world and the spread of knowledge.
5.	Business Model (Revenue Model)	Our application covers a range of topics including politics, business, criminal justice, environment, technology etc. Our business model will be monetized and generate income by showing advertisements and Operating on monthly and yearly subscription model.
6.	Scalability of the Solution	Scalability is one of the benchmarks of the cloud services and its adoption with businesses. Cloud scalability will help to increase the user-base by increasing the resource allocation and meeting the changing demands without sacrificing the efficiency or quality of our customer service and internal operations. Providing fast and reliable news while maintaining positive relationships with your customers.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through online application Registration through Gmail

FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User login	Login through browser directly by entering username and password Login through Login through email

FR-4	User interaction	Done through user interface between client and server View the related news by subscripted or requested page

Non-functional Requirements:

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

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	End users can receive push updates for new content on a site by subscribing to the site's news feed
NFR-2	**Security**	How well are the system and its data protected against attacks
NFR-3	**Reliability**	How often does the system experience critical failures? How much time does it take to fix the issue when it arises ?And how is user availability time compared to downtime?

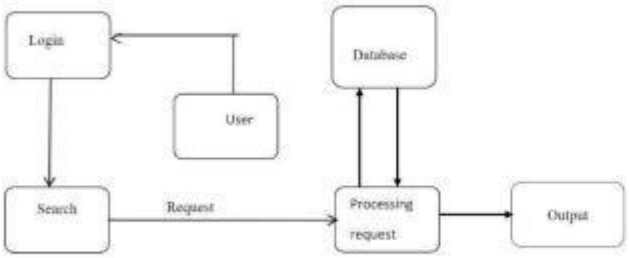
NFR-4	**Performance**	Performance is the core non-functional requirements no system can do without. It defines how fast a software system or a particular piece of it responds to certain users actions under a certain workload. In most cases, this metric explains how long a user must wait before the target operation happens (the page renders, a transaction is processed, etc.) given the overall number of users at the moment. But it's not always like that. Performance requirements may describe background
		processes invisible to users, e.g. backup. But let's focus on user-centric performance.

NFR-5	**Availability**	Availabilitydescribes how likely the system is accessible to a user at a given point in time. While it can be expressed as an expected percentage of successful requests, you may also define it as a percentage of time the system is accessible for operation during some time period. For instance, the system may be available 98 percent of the time during a month. Availability is perhaps the most business-critical requirement, but to define it, you also must have estimations for reliability and maintainability.
NFR-6	**Scalability**	Scalabilityassesses the highest workloads under which the system will still meet the performance requirements. There are two ways to enable your system scale as the workloads get higher: horizontal and vertical scaling.

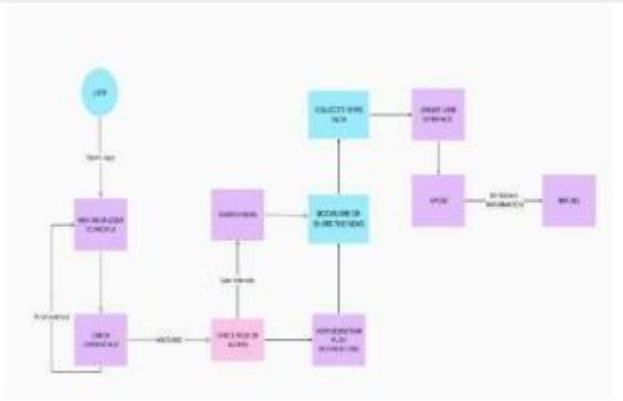
5. PROJECT DESIGN

5.1 Data Flow Diagrams.

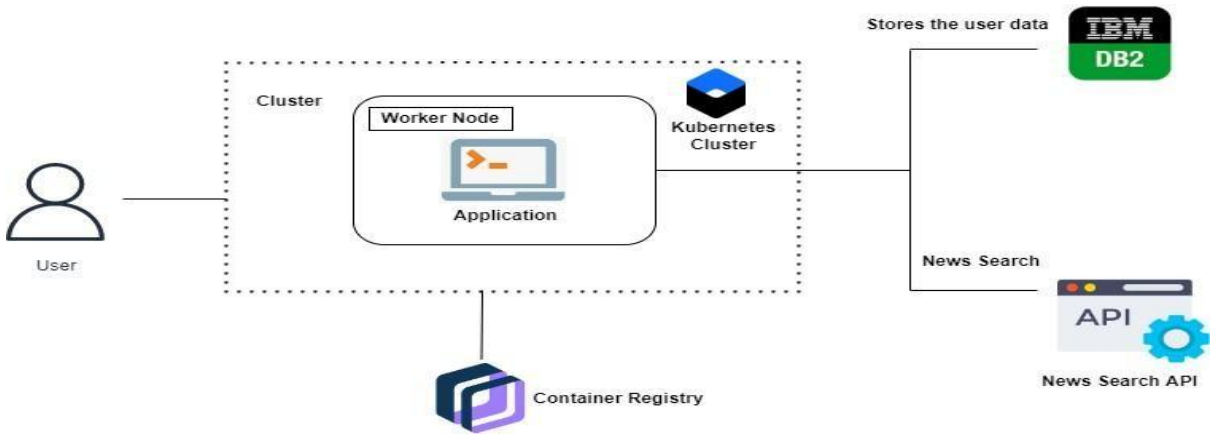
Flow Chart



DFD - News Tracking Application



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, Web 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
	Confirmation	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	Medium	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I can access my account / dashboard	High	Sprint-2
	Dashboard	USN-4	As a user, I can search NEWS and a quick snap is displayed in the dashboard	I can view the NEWS	High	Sprint-2
	Chatbot	USN-5	As a user, I can Chart with the bot so that my questions are clarified	I can access the chat	High	Sprint-3
	Profile	USN-6	As a user I can edit my interest so that I can get news accordingly	I can access Profile	High	Sprint-3
	Notification	USN-7	As a user, I will receive notification to my email so that I'll be updated on the news	I can receive notification mail	Medium	Sprint-4

6. PROJECT PLANNING & SCHEDULING

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	Creating Login page Creating Registration page	10	High	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint-1	Database Connectivity	USN-2	To Store details of the customer Connecting UI with Database	10	Medium	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint-2	News Tracker UI	USN-3	Building UI News Tracker Application	10	High	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint-2	API	USN-4	Connecting UI with News API, Google News API	10	High	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint-3	SendGrid Integration	USN-5	SendGrid Integration With Python Code	10	Low	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint-3	News Reader (Voice)	USN-6	Building Voice Assistant to read the news	10	Medium	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint-4	Containerization	USN-7	Containerizing the app	10	High	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran
Sprint -4	Upload image and deployment	USN-8	Upload Docker image to the IBM Registry and deploy it in the Kubernetes Cluster	10	High	Dinesh, Pradeep, Kalpana, Pravin Kumar, Vigneshwaran

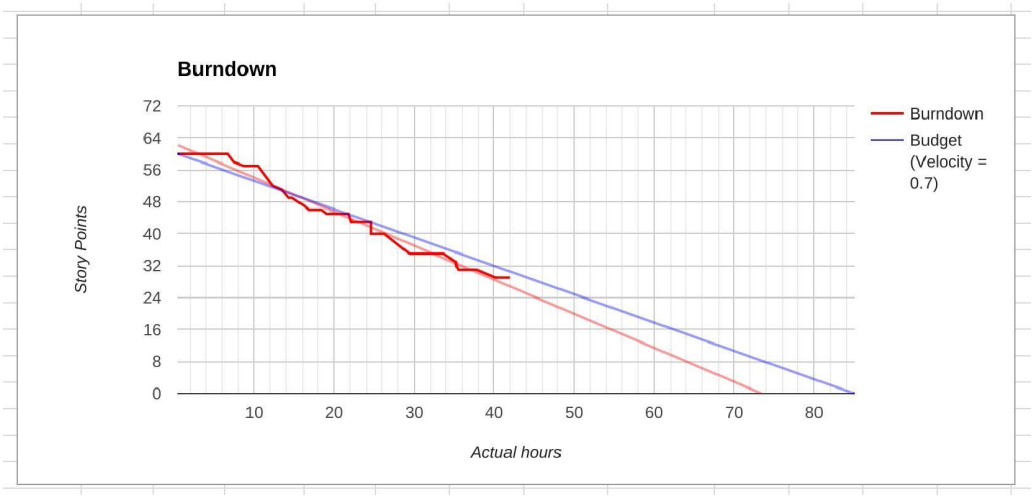
Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	4 Days	01 Nov 2022	04 Nov 2022	20	17 Nov 2022
Sprint-2	20	4 Days	04 Nov 2022	08 Nov 2022	20	17 Nov 2022
Sprint-3	20	4 Days	08 Nov 2022	12 Nov 2022	20	17 Nov 2022
Sprint-4	20	4 Days	12 Nov 2022	16 Nov 2022	20	17 Nov 2022

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

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

Burndown Chart:



7. CODING

app.py

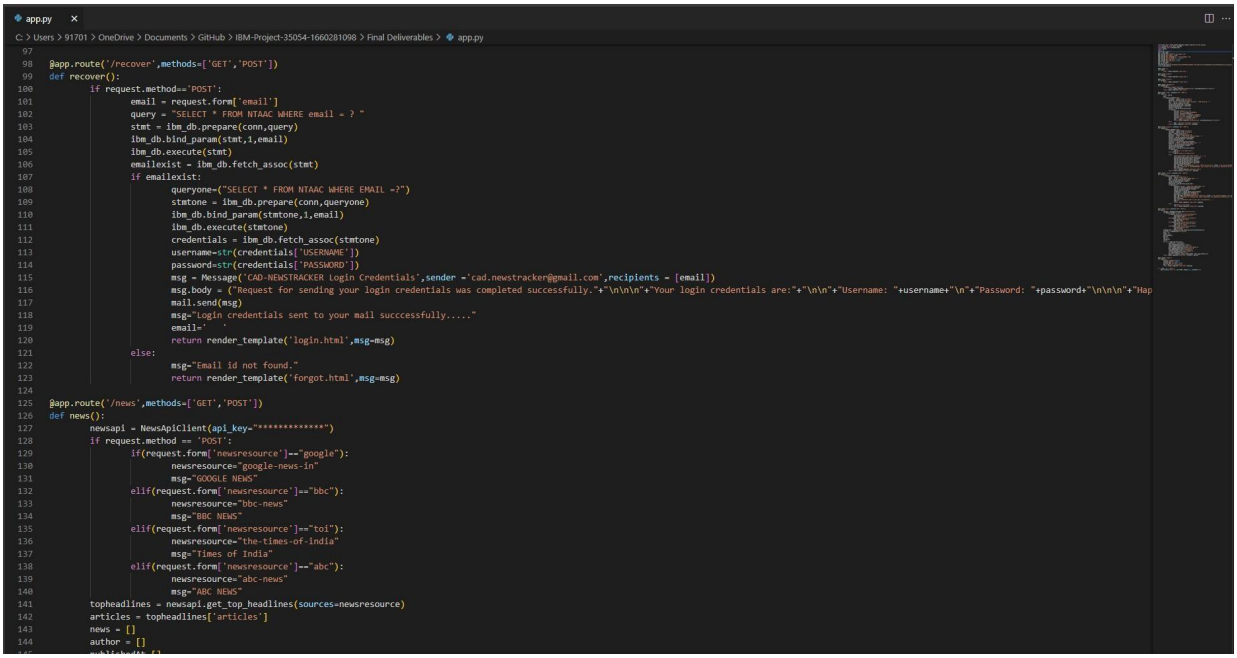
C: > Users > 91701 > OneDrive > Documents > GitHub > IBM-Project-35054-1660281098 > Final Deliverables > app.py

```python
from flask import Flask,render_template,request,redirect,url_for,session
from flask_mail import Mail, Message
from newsapi import NewsApiClient
import ibm_db
import re
app=Flask(__name__)
mail = Mail(app)
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = '********@gmail.com'
app.config['MAIL_PASSWORD'] = '*******'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
app.secret_key = 'a'
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32328;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=dqw60999;PWD=I
global newsresource

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

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

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

@app.route('/dashboard')
def dashboard():
        if(session['loggedin']==True):
                return render_template('dashboard.html',username=session['fullname'])
        return render_template('login.html')

@app.route('/login',methods=['GET','POST'])
def login():
        global userid
        msg=' '
        if request.method=='POST':
                username = request.form['username']
                password = request.form['password']
                sql = "SELECT * FROM NTAAC WHERE username = ? AND password = ?"
                stmt = ibm_db.prepare(conn,sql)
                ibm_db.bind_param(stmt,1,username)
                ibm_db.bind_param(stmt,2,password)
                ibm_db.execute(stmt)
```

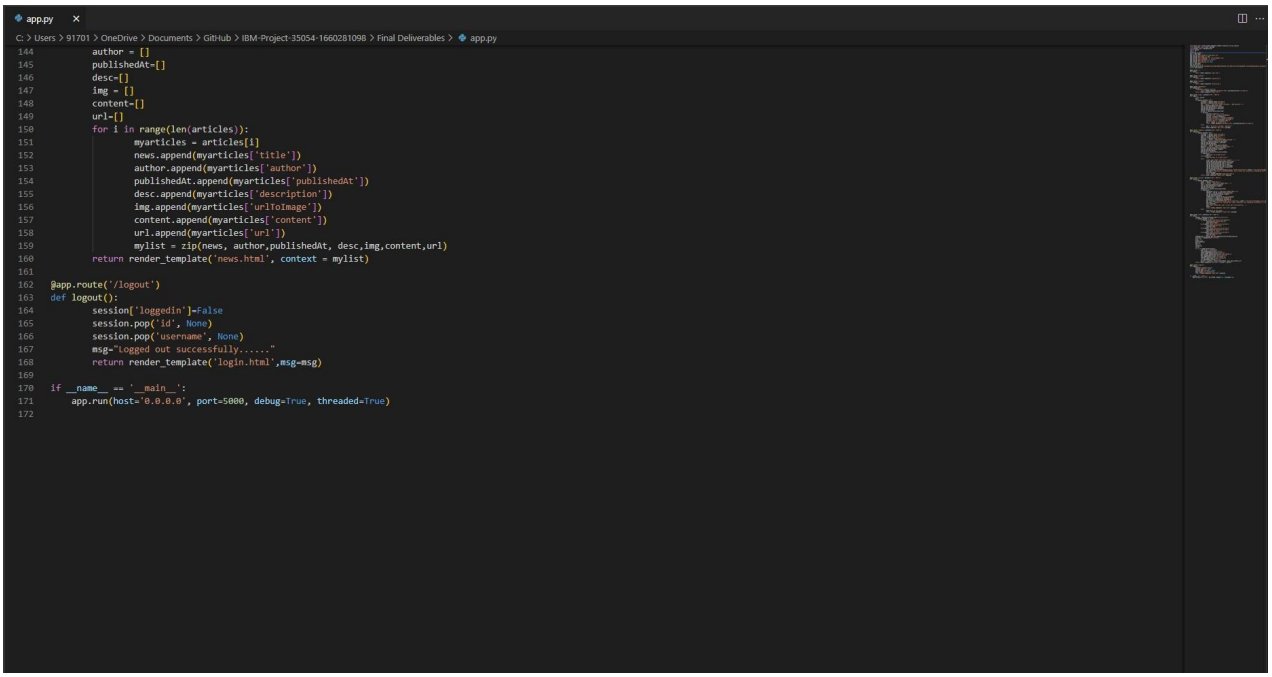
app.py

C: > Users > 91701 > OneDrive > Documents > GitHub > IBM-Project-35054-1660281098 > Final Deliverables > app.py

```python
from flask import Flask,render_template,request,redirect,url_for,session
from flask_mail import Mail, Message
from newsapi import NewsApiClient
import ibm_db
import re
app=Flask(__name__)
mail = Mail(app)
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = '********@gmail.com'
app.config['MAIL_PASSWORD'] = '*******'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
app.secret_key = 'a'
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32328;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=dqw60999;PWD=I
global newsresource

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

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

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

@app.route('/dashboard')
def dashboard():
        if(session['loggedin']==True):
                return render_template('dashboard.html',username=session['fullname'])
        return render_template('login.html')

@app.route('/login',methods=['GET','POST'])
def login():
        global userid
        msg=' '
        if request.method=='POST':
                username = request.form['username']
                password = request.form['password']
                sql = "SELECT * FROM NTAAC WHERE username = ? AND password = ?"
                stmt = ibm_db.prepare(conn,sql)
                ibm_db.bind_param(stmt,1,username)
                ibm_db.bind_param(stmt,2,password)
                ibm_db.execute(stmt)
```



```
@app.route('/recover',methods=['GET','POST'])
def recover():
        if request.method=='POST':
                email = request.form['email']
                query = "SELECT * FROM NTAAC WHERE email = ? "
                stmt = ibm_db.prepare(conn,query)
                ibm_db.bind_param(stmt,1,email)
                ibm_db.execute(stmt)
                emailexist = ibm_db.fetch_assoc(stmt)
                if emailexist:
                        queryone=("SELECT * FROM NTAAC WHERE EMAIL =?")
                        stmtone = ibm_db.prepare(conn,queryone)
                        ibm_db.bind_param(stmtone,1,email)
                        ibm_db.execute(stmtone)
                        credentials = ibm_db.fetch_assoc(stmtone)
                        username=str(credentials['USERNAME'])
                        password=str(credentials['PASSWORD'])
                        msg = Message('CAD-NEWSTRACKER Login Credentials',sender ='cad.newstracker@gmail.com',recipients = [email])
                        msg.body = ("Request for sending your login credentials was completed successfully."+"\n\n\n"+"Your login credentials are:"+"\n\n"+"Username: "+username+"\n"+"Password: "+password+"\n\n\n"+"Hap
                        mail.send(msg)
                        msg="Login credentials sent to your mail succcessfully....."
                        email=''
                        return render_template('login.html',msg=msg)
                else:
                        msg="Email id not found."
                        return render_template('forgot.html',msg=msg)

@app.route('/news',methods=['GET','POST'])
def news():
        newsapi = NewsApiClient(api_key="************")
        if request.method == 'POST':
                if(request.form['newsresource']=="google"):
                        newsresource="google-news-in"
                        msg="GOOGLE NEWS"
                elif(request.form['newsresource']=="bbc"):
                        newsresource="bbc-news"
                        msg="BBC NEWS"
                elif(request.form['newsresource']=="toi"):
                        newsresource="the-times-of-india"
                        msg="Times of India"
                elif(request.form['newsresource']=="abc"):
                        newsresource="abc-news"
                        msg="ABC NEWS"
        topheadlines = newsapi.get_top_headlines(sources=newsresource)
        articles = topheadlines['articles']
        news = []
        author = []
```



```
        author = []
        publishedAt=[]
        desc=[]
        img = []
        content=[]
        url=[]
        for i in range(len(articles)):
                myarticles = articles[i]
                news.append(myarticles['title'])
                author.append(myarticles['author'])
                publishedAt.append(myarticles['publishedAt'])
                desc.append(myarticles['description'])
                img.append(myarticles['urlToImage'])
                content.append(myarticles['content'])
                url.append(myarticles['url'])
                mylist = zip(news, author,publishedAt, desc,img,content,url)
        return render_template('news.html', context = mylist)

@app.route('/logout')
def logout():
        session['loggedin']=False
        session.pop('id', None)
        session.pop('username', None)
        msg="Logged out successfully....."
        return render_template('login.html',msg=msg)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True, threaded=True)
```

8. ADVANTAGES & DISADVANTAGES

1. This app can be accessed anywhere and anytime, So that the user can view the news
2. Its ad free
3. The news is only based on the API

4. It may contain some unwanted content but we don't have control over it ● The user can bookmark their favourite news.

9. CONCLUSION

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

10. FUTURE SCOPE

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

11. APPENDIX

- *https://github.com/IBM-EPBL/IBM-Project-45195-1660728763*

DEMO LINK:
https://drive.google.com/file/d/13ghcIOJ8hhHKNYGNy12Fd2oHzgS3ueGS/view?usp=share_link