**NEWS TRACKER APPLICATION**

**TEAM ID :** PNT2022TMID07300

**DOMAIN NAME :** APPLICATION DEVELOPMENT

|  |  |  |
| --- | --- | --- |
| **NAME** | **REG NO** | **TEAM** |
| Aisvarya P | 711519BEC004 | Team Leader |
| Poomalar K | 711519BEC305 | Team member 1 |
| Aisvarya pandi k | 711519BEC033 | Team member 2 |
| Jayakumar p | 711519BEC043 | Team member 3 |

**FACULTY MENTOR(S) INDUSTRY MENTOR(S)**

Mrs.M.VAISHNAVI SAI PRIYA

**TABLE OF FIGURES**

1. **INTRODUCTION**

1.1 Project Overview

1.2 Purpose

2. **LITERATURE SURVEY**

2.1 References

2.2 Problem Statement Definition

3. **IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. **REQUIREMENT ANALYSIS**

4.1 Functional requirement

4.2 Non-Functional requirements

5. **PROJECT DESIGN**

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. **PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. **CODING & SOLUTIONING**

1. **INTRODUCTION**

1.1 **PROJECT OVERVIEW**

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...

1.2 **PURPOSE**

Newspapers are one of the most popular and most needed commodities in our daily life. In today's busy world, reading newspapers has become one of the traditional ways of getting news. News is produced every minute and distributed via television, radio and the Internet, so the news updated the next morning is already outdated. So newspaper and magazine publishers have a hard time keeping up with the pace. Change is needed and publishers must embrace mobile.

2. **LITERATURE SURVEY**

2.1 **REFERENCE**

1.https://flask.palletsprojects.com/en/2.1.x/tutorial/ 2.https://www.ibm.com/docs/en/db2-for-zos/11?topic=zos-db2-basics-tutorial-working-db2-interactiv ely

3.https://www.ibm.com/in-en/products/watson-assistant/docs-resources 4.https://www.ibm.com/blogs/cloud-archive/2019/03/video-containerization-explained/

5. https://developer.ibm.com/components/cloud-foundry/tutorials/

2.2 **PROBLEM STATEMENT DEFINITION**

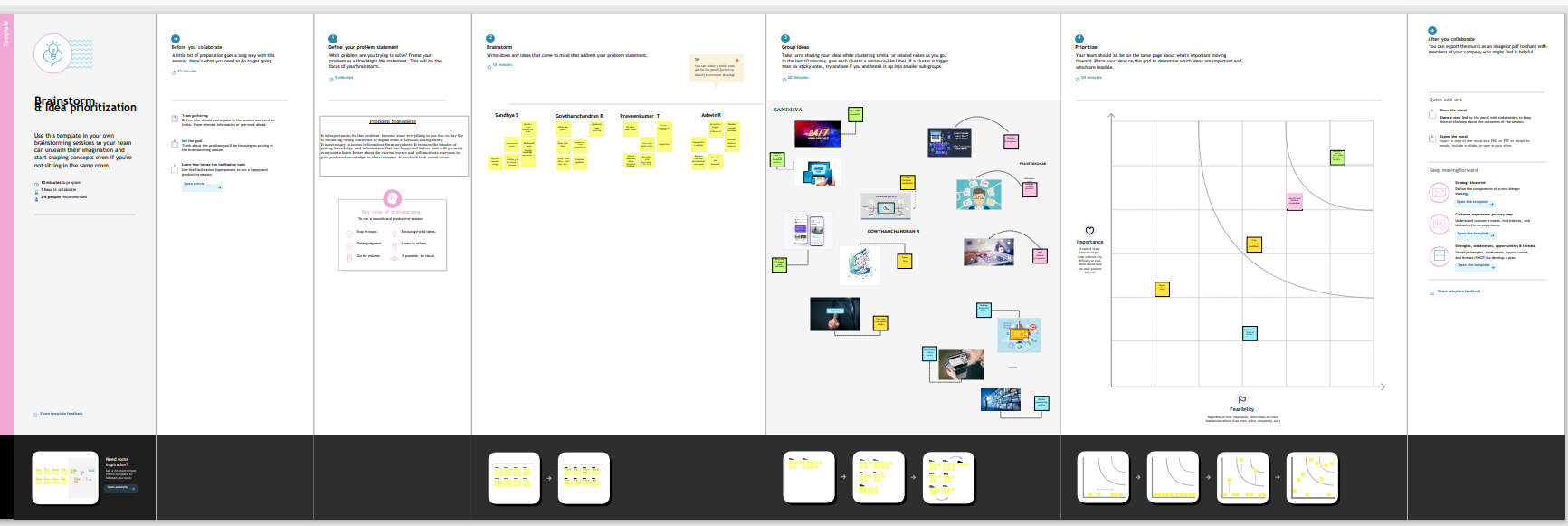
In today’s world we have a lot of work and there is no time to cope up with everything in the schedule. So, it is not possible to read the newspaper. By using this application, the user can access the news they are interested in and get a quick peek for the day. There are multiple news-sharing apps used by a single user and are often spammed with notifications. There is also a lot of fake news which gets shared. A news sharing app wants to help users find relevant and important news easily every day and also understand explicitly that the news is not fake but from proper sources. This news tracker application helps the user to get all information about commodities, sports, technology, education etc. The user can register using their personal details and interact with the application and save their time.Creating an app that benefits with the smart news filtering functionality where it allows its users to segregate various news sections, comprising business, sports, political, international, and more. The users can opt for filters like Breaking News, Most Popular, and more. Creating a user-friendly news app that would not just have good number of features, but will also be able to get accessed by different types of users keeping the experience delightful.

3.**IDEATION & PROPOSED SOLUTION**

3.1.**EMPATHY MAP CANVAS**



3.2.**IDEATION & BRAINSTORMING**



3.3.**PROPOSED SOLUTION**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **PARAMETER** | **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 touches, providing authorized news, cloud-based app, With tons of features to make it exciting. |
| 3. | Novelty/Solution description | Profile based account, like and share, download blogs, links to original publishers are provided. |
| 4. | Social Impact/Customer Satisfication | 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 tothe customers need. |

3.4 **PROBLEM SOLUTION FIT**



4.**REQUIREMENT ANALYSIS**

4.1. **FUNCTIONAL REQUIREMENTS**

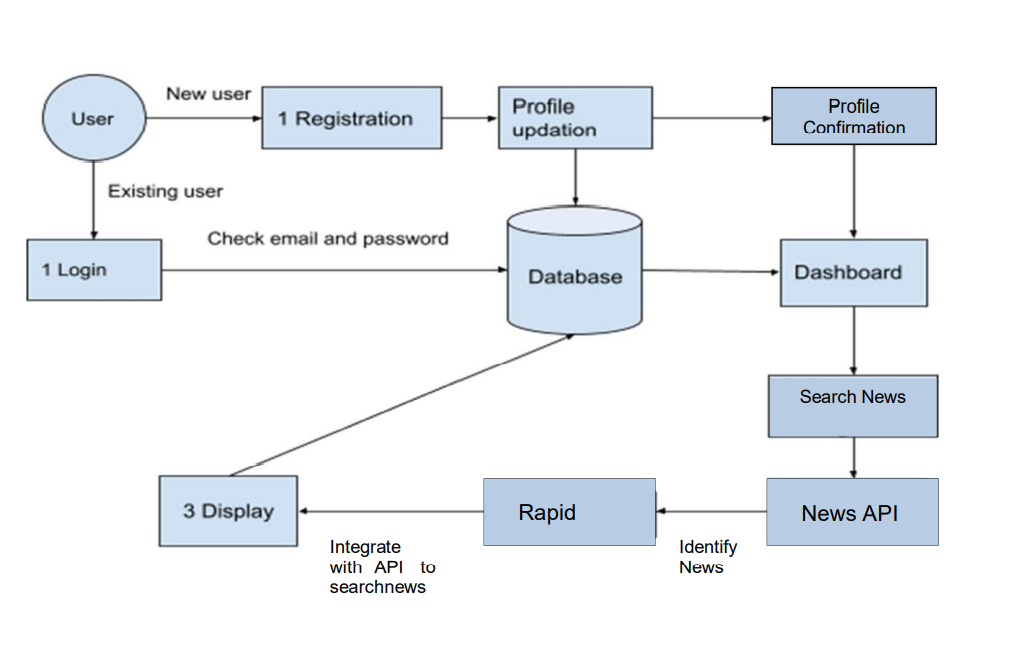
|  |  |  |
| --- | --- | --- |
| **FR NO.** | **FUNCTIONAL REQUIREMENTS**  **(EPIC)** | **SUB REQUIREMENT (STORY/SUB-TASK)** |
| FR-1 | User Registration | Registration through Email  Registration through Phone |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | User Profile Completion | Get personal details like name, news reading plan, phone number, gender, etc.. |
| FR-4 | Display news | Integrate Rapid API and display the headline news |
| FR-5 | Search news | User can search their favorite category of news |

4.2. **NON-FUNCTIONAL REQUREMENTS**

|  |  |  |
| --- | --- | --- |
| **NFR NO.** | **NON-FUNCTIONAL REQUIREMENT** | **DESCRIPTION** |
| NFR-1 | Usability | Provide user friendly UI  Simple and intuitive design. |
| NFR-2 | Security | Comprehensive authorization and authentication  scheme for each system actor. |
| NFR-3 | Reliability | The system must perform without failure in 95  percent of use cases. |
| NFR-4 | Performance | The landing page supporting several users must  provide 5 second or less response time. |
| NFR-5 | Availability | Uninterrupted services must be available all time  except the time of server updating. |
| NFR-6 | Scalability | Provide horizontal or vertical scaling for higher worloads. |

5.**PROJECT DESIGN**

5.1. **DATA FLOW DIAGRAM**



5.2. **SOLUTION & TECHNICAL ARCHITECTURE**

● Find the best tech solution to solve existing business problems.

● Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.

● Define features, development phases, and solution requirements.

● Provide specifications according to which the solution is defined, managed, and delivered.

5.3.**USER STORIES**

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.

6.**PROJECT PLANNING & SCHEDULING**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional requriment**  **(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 | Aisvarya |
| Sprint-1 |  | USN-2 | As a user, I will receive confirmation email  once I have  registered for the  application. | 1 | High | Aisvarya |
| Sprint-1 |  | USN-3 | As a user, I can register for the application  through Facebook.  I can register for the  application through Gmail. | 2 | Low | Poomalar |
| Sprint-1 | Login | USN-4 | As a user, I can log into the  application by entering email  &password. | 2 | Low | Aisvarya |
| Sprint-2 | Dashboard | USN-5 | As a user I should be able to navigate  and access all the features hassle free. | 1 | Medium | Jaya kumar |
| Sprint-2 | Layout | USN-6 | As a user I should be able to access the portal with different devices  with the same  comfort. | 2 | High | Jaya kumar |
| Sprint-3 | Data store and retrieval | USN-7 | Get Data from API and store as JSON in DB2. | 2 | High | Gowtham pandi |
| Sprint-3 |  | USN-8 | Get bin data from API and store in DFS. | 3 | High | Gowtham pandi |
| Sprint-4 | User Segregation and data access | USN-9 | As a CC executive I should be able to  uniquely identify the customer and offer  help. | 1 | Low | Poomalar |

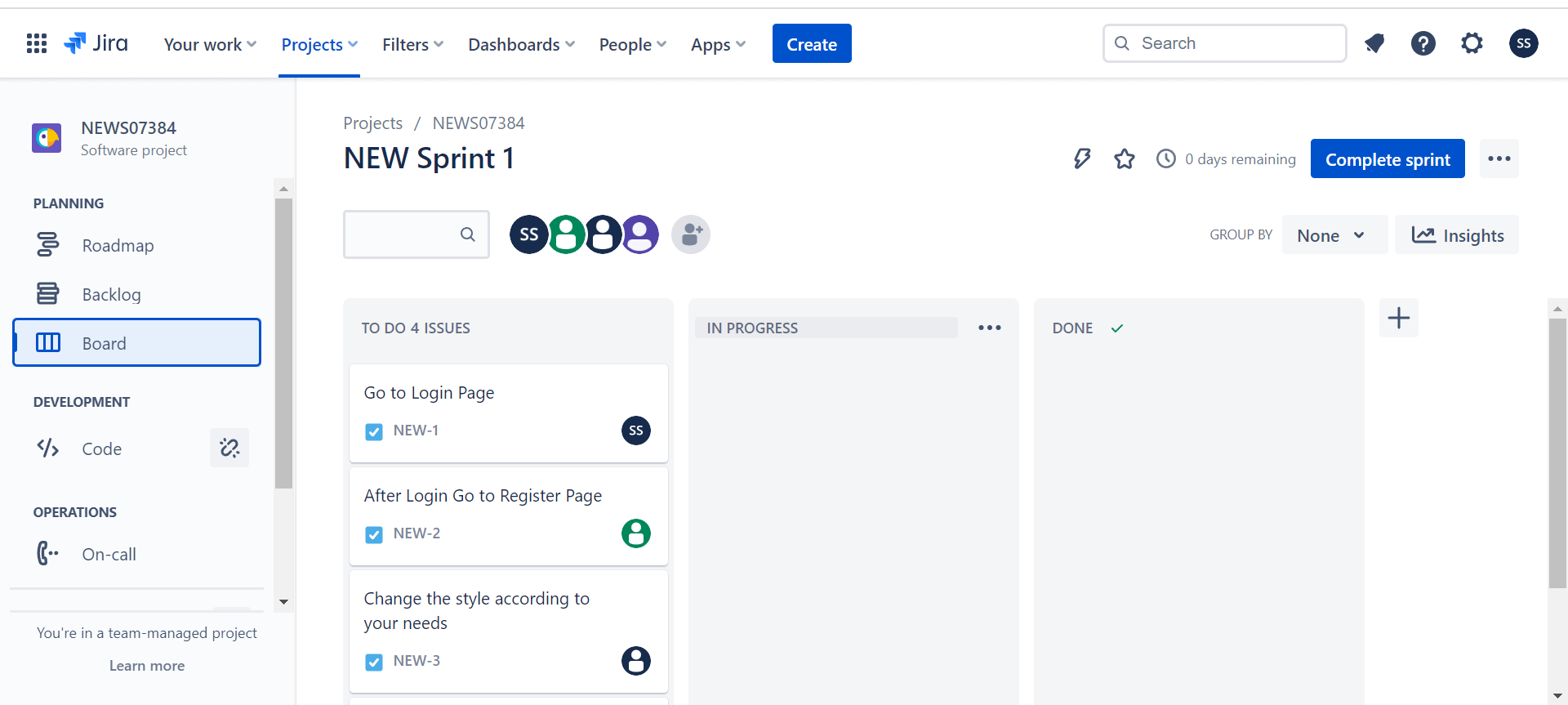
**SPRINT DELIVERY SCHEDULE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date** | **Story Point Complete** | **Sprint Release Date** |
| Sprint-1 | 20 | 7 days | 21 Oct 2022 | 28 Oct 2022 | 8 | 21 Oct 2022 |
| Sprint-2 | 20 | 7 days | 28 Oct 2022 | 04 Nov 2022 | 3 | 28 Oct 2022 |
| Sprint-3 | 20 | 7 days | 04 Nov 2022 | 11 Nov 2022 | 5 | 04 Nov 2022 |
| Sprint-4 | 20 | 7 days | 11 Nov 2022 | 18 Nov 2022 | 5 | 11 Nov 2022 |

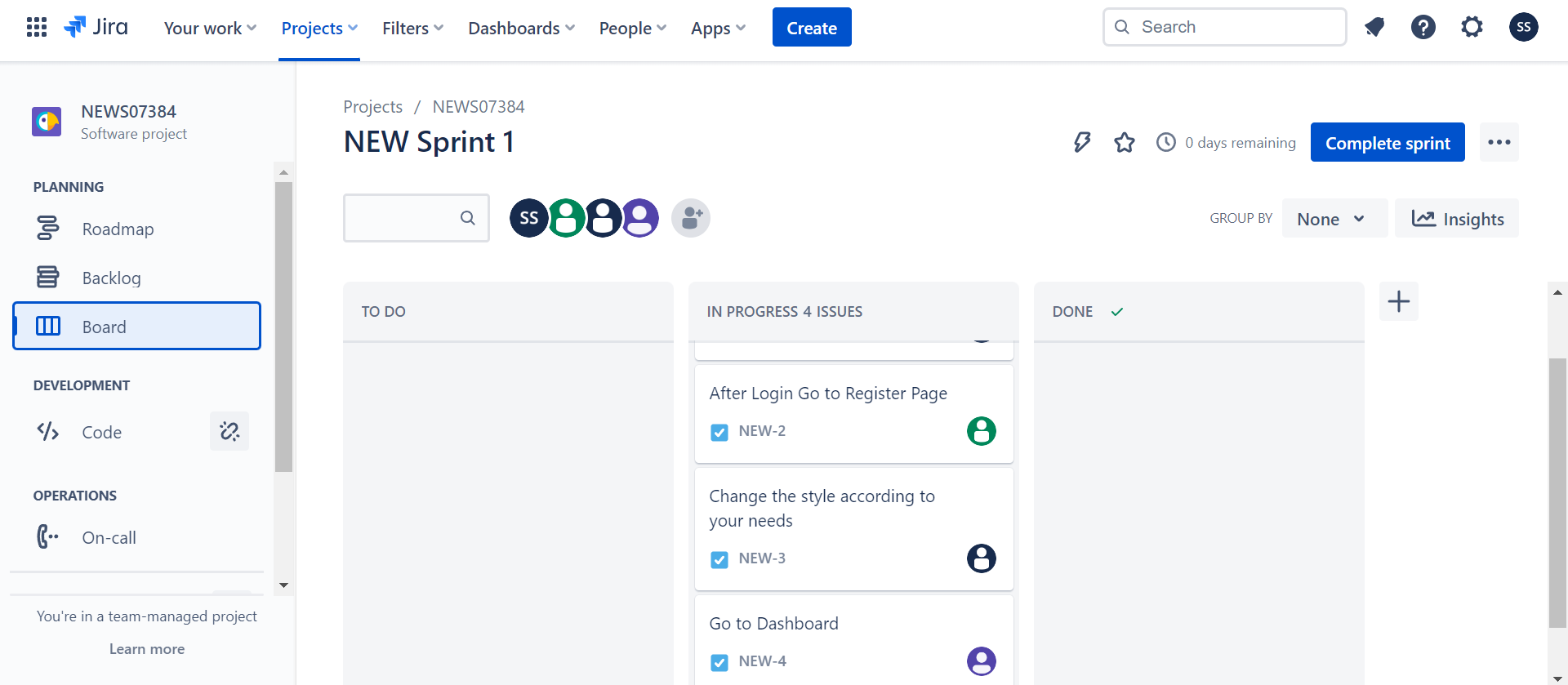
**REPORTS FROM JIRA**

**Sprint - 1**

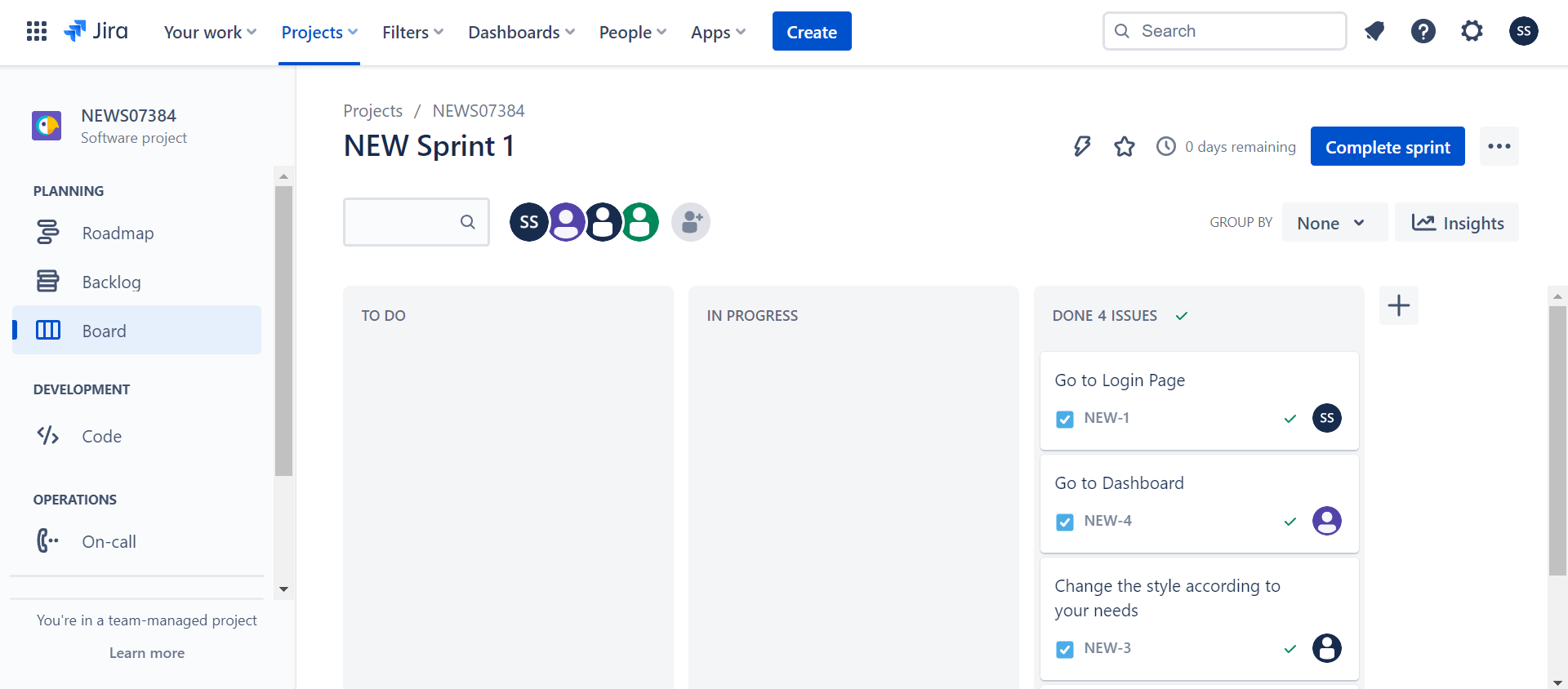
**Sprint in To Do**



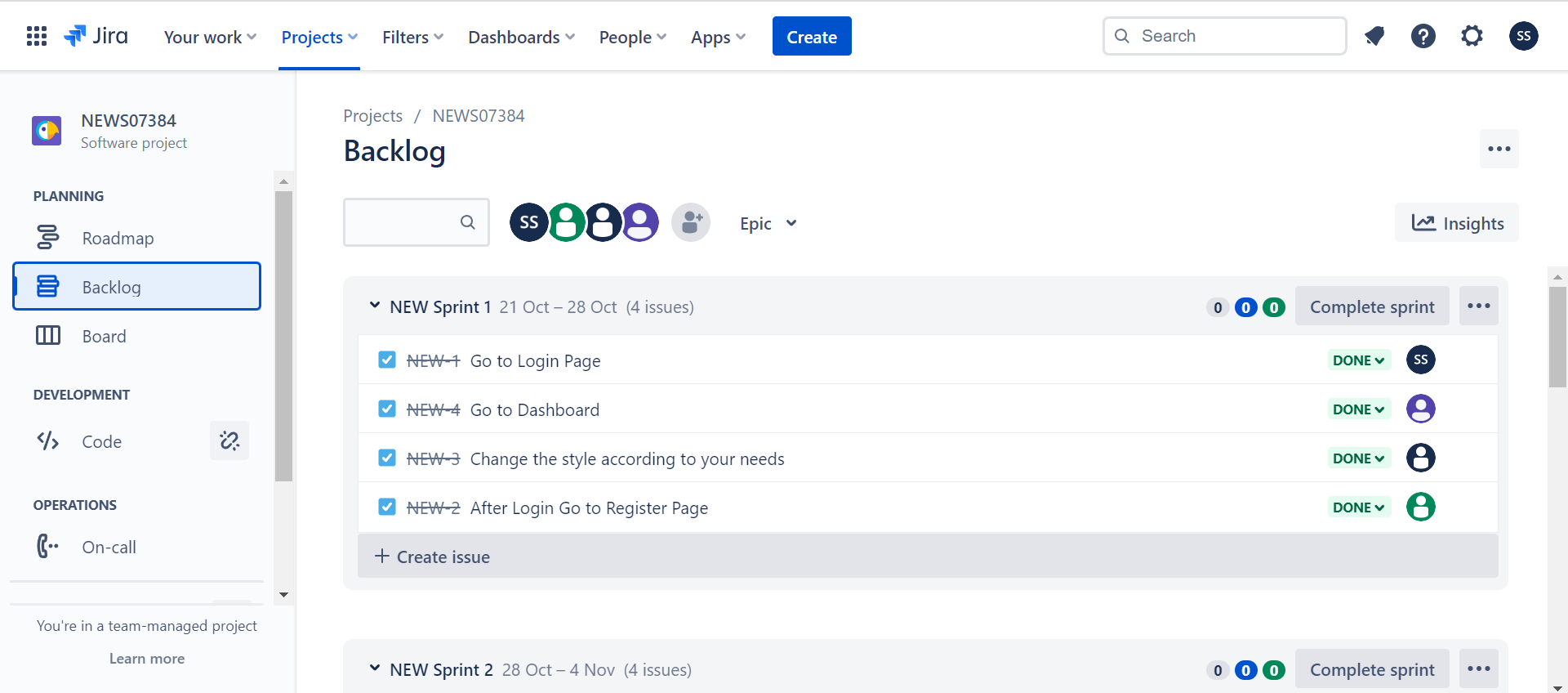
**Sprint in Progress**



**Sprint is Done**

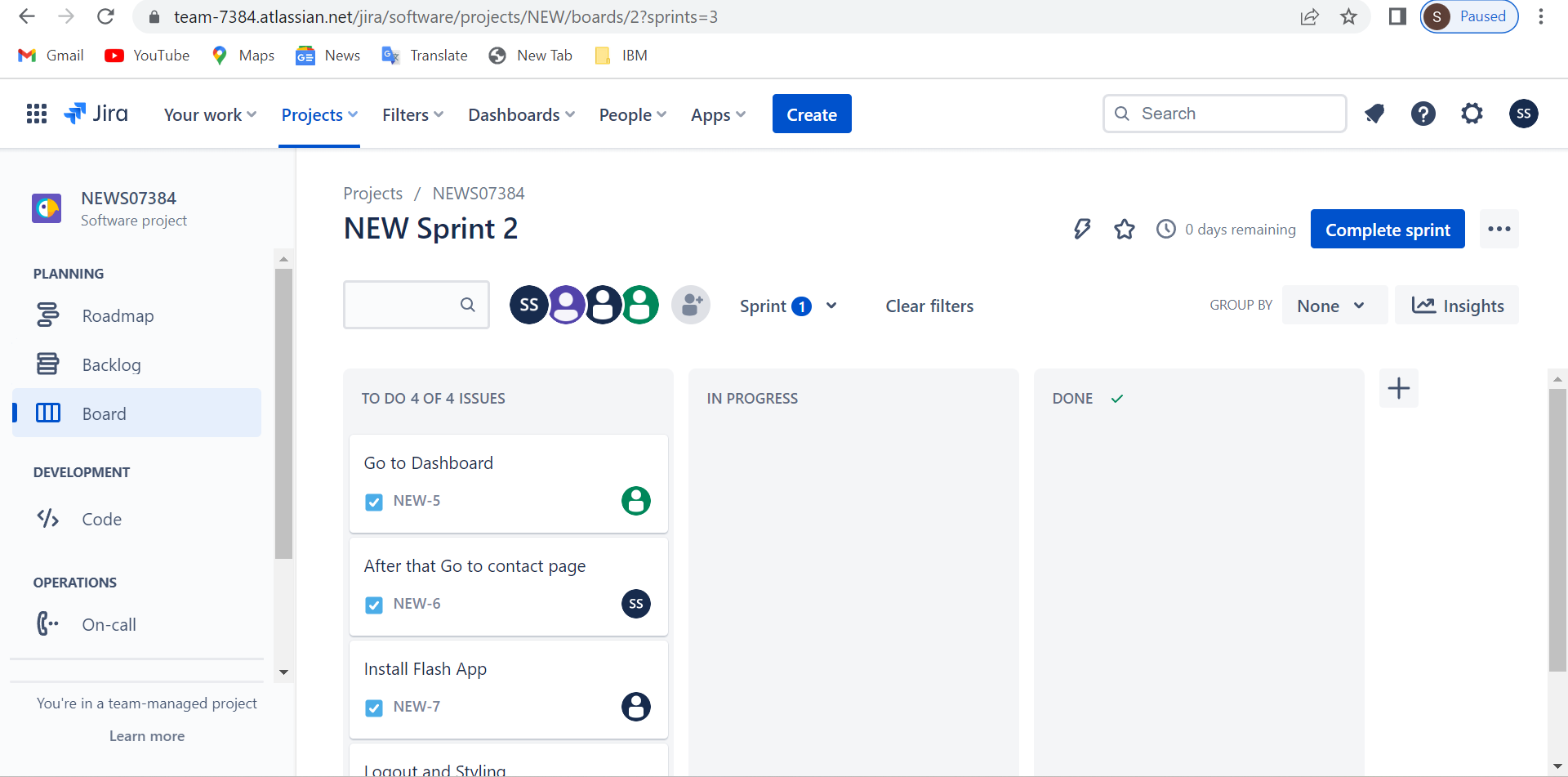


**Final Result - Sprint 1**

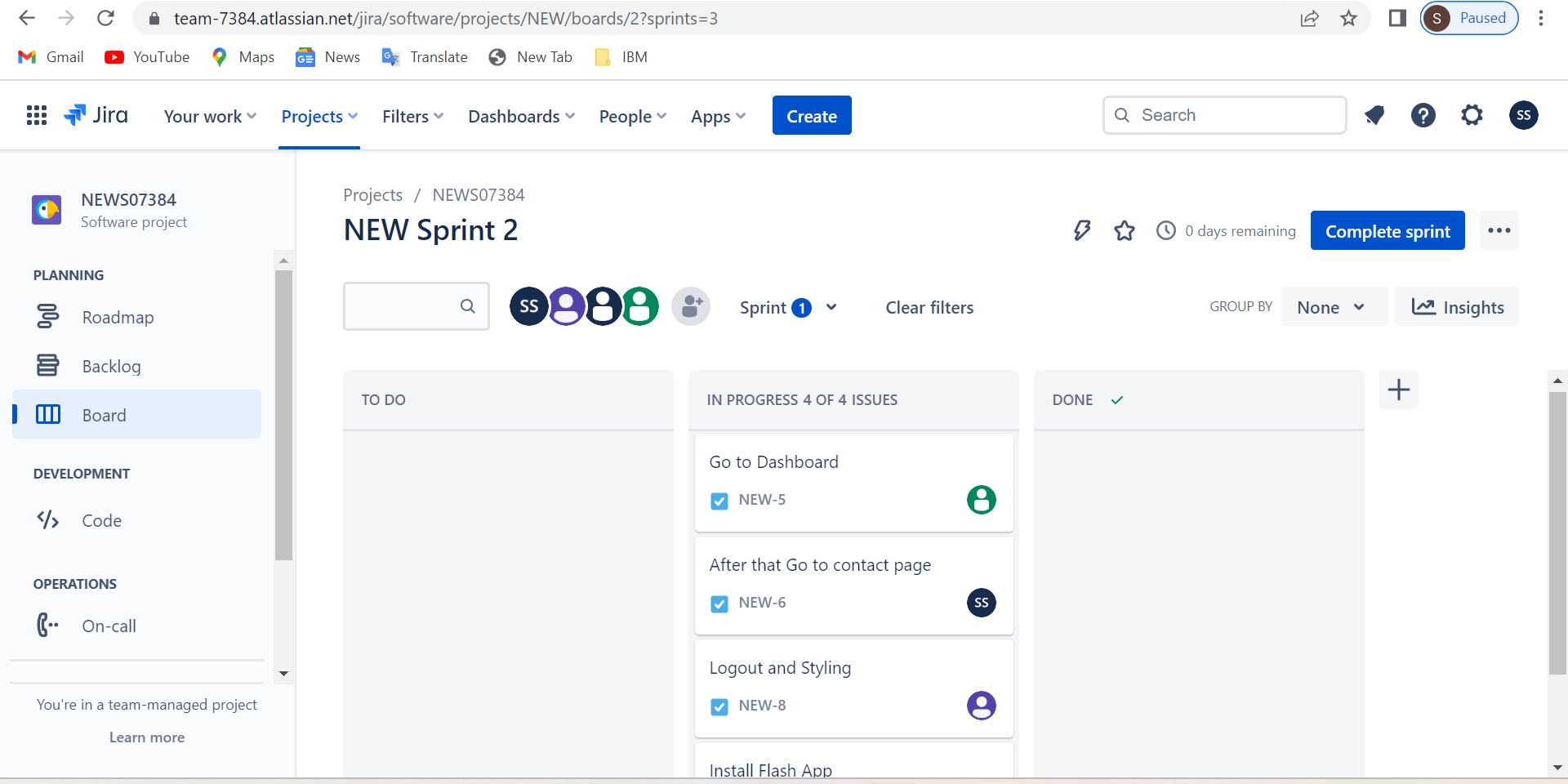


**Sprint - 2**

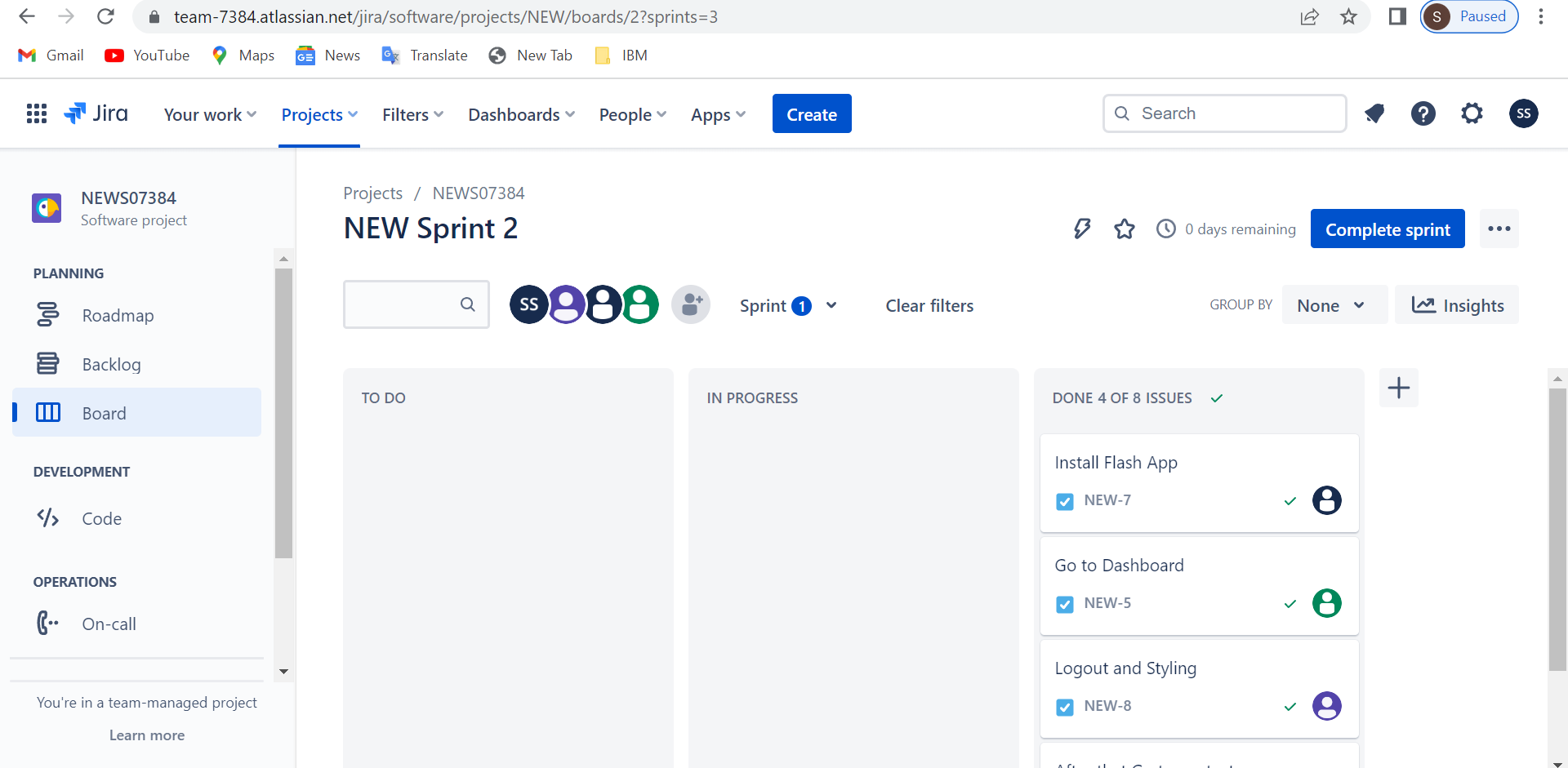
**Sprint in To Do**



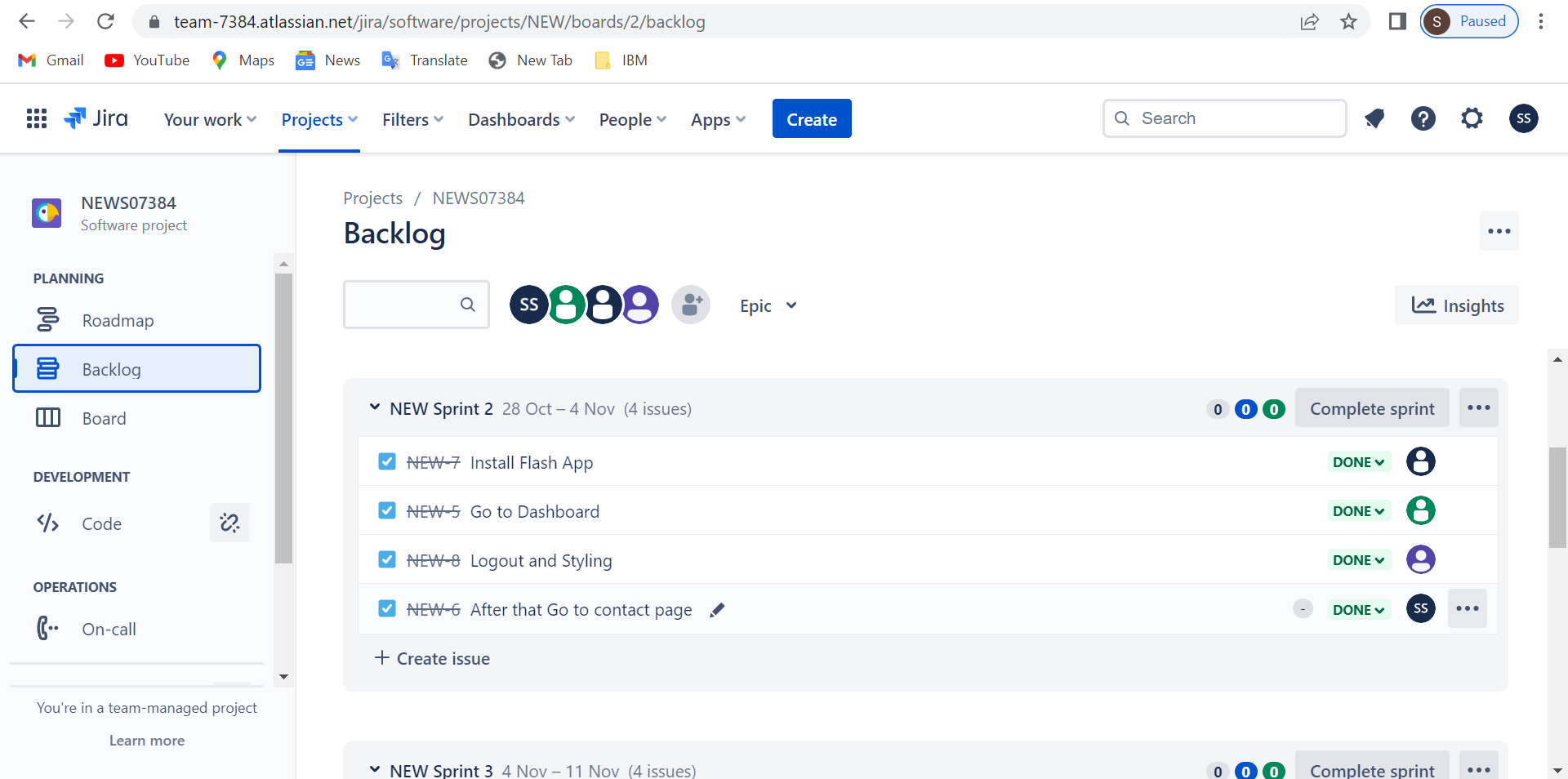
**Sprint in progress**



**Sprint is done**

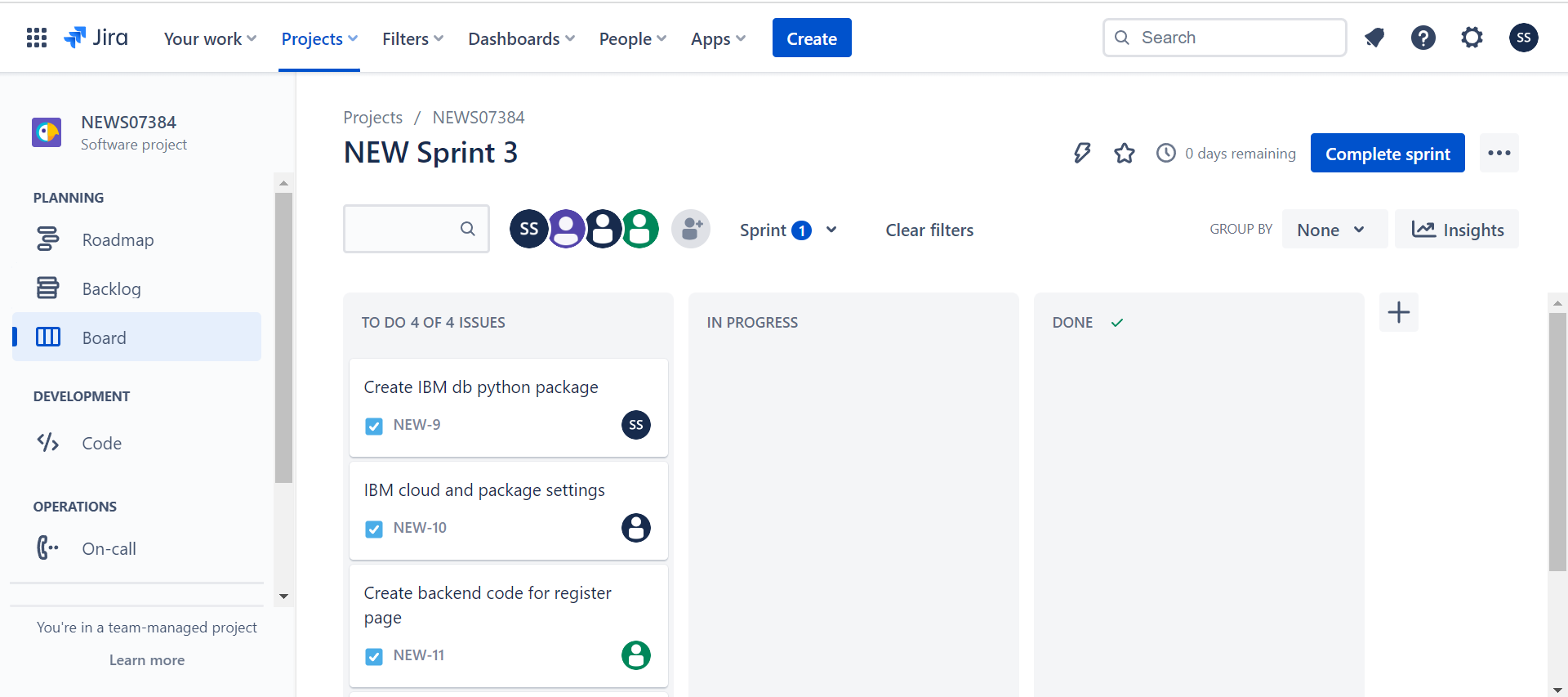


**Final Result - Sprint 2**

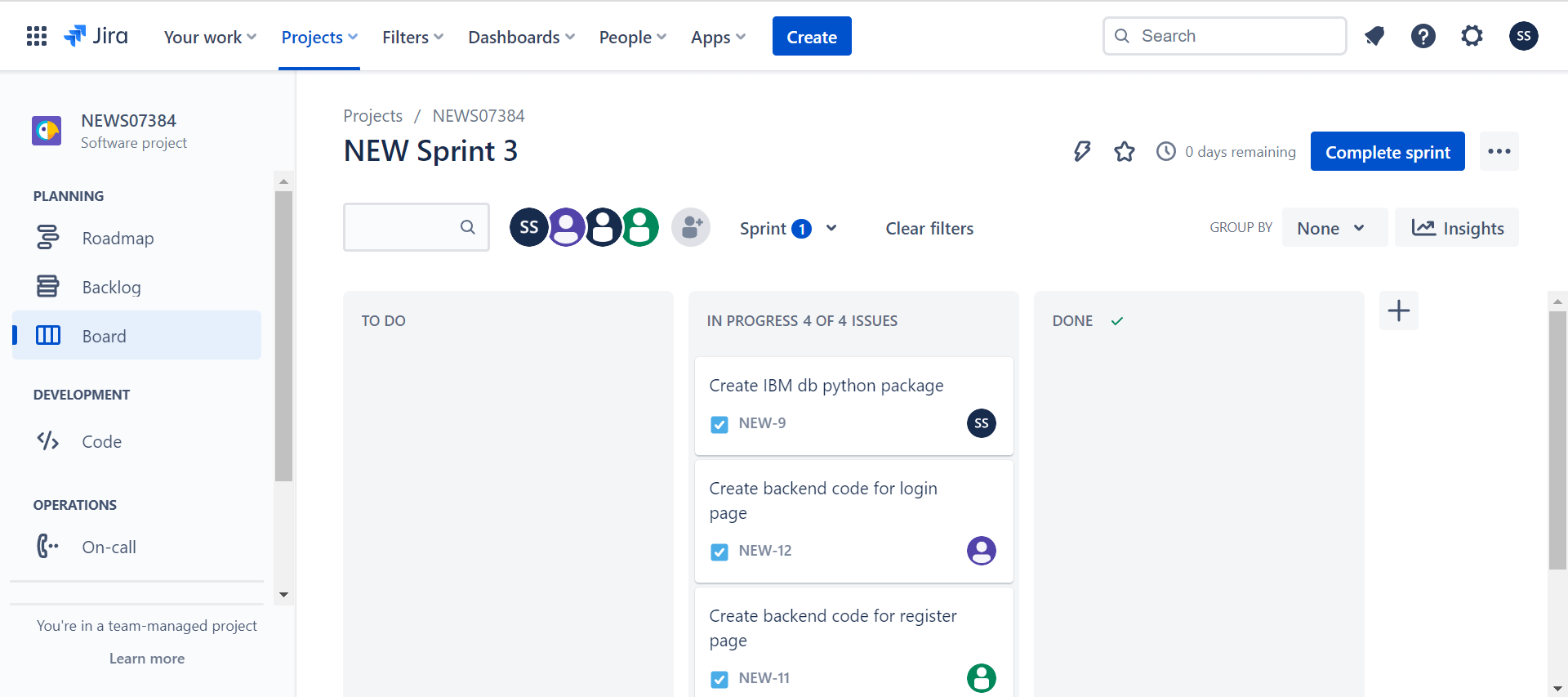


**Sprint 3**

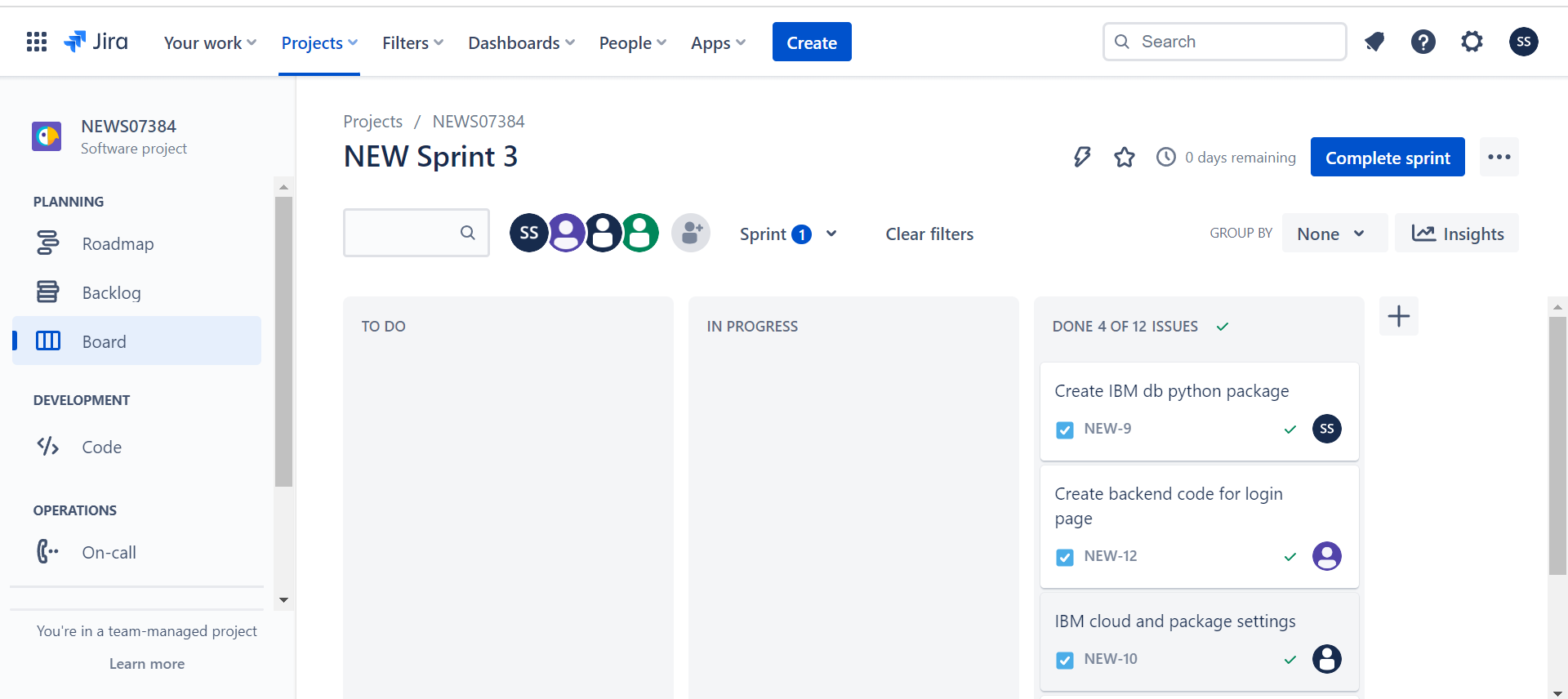
**Sprint in To Do**



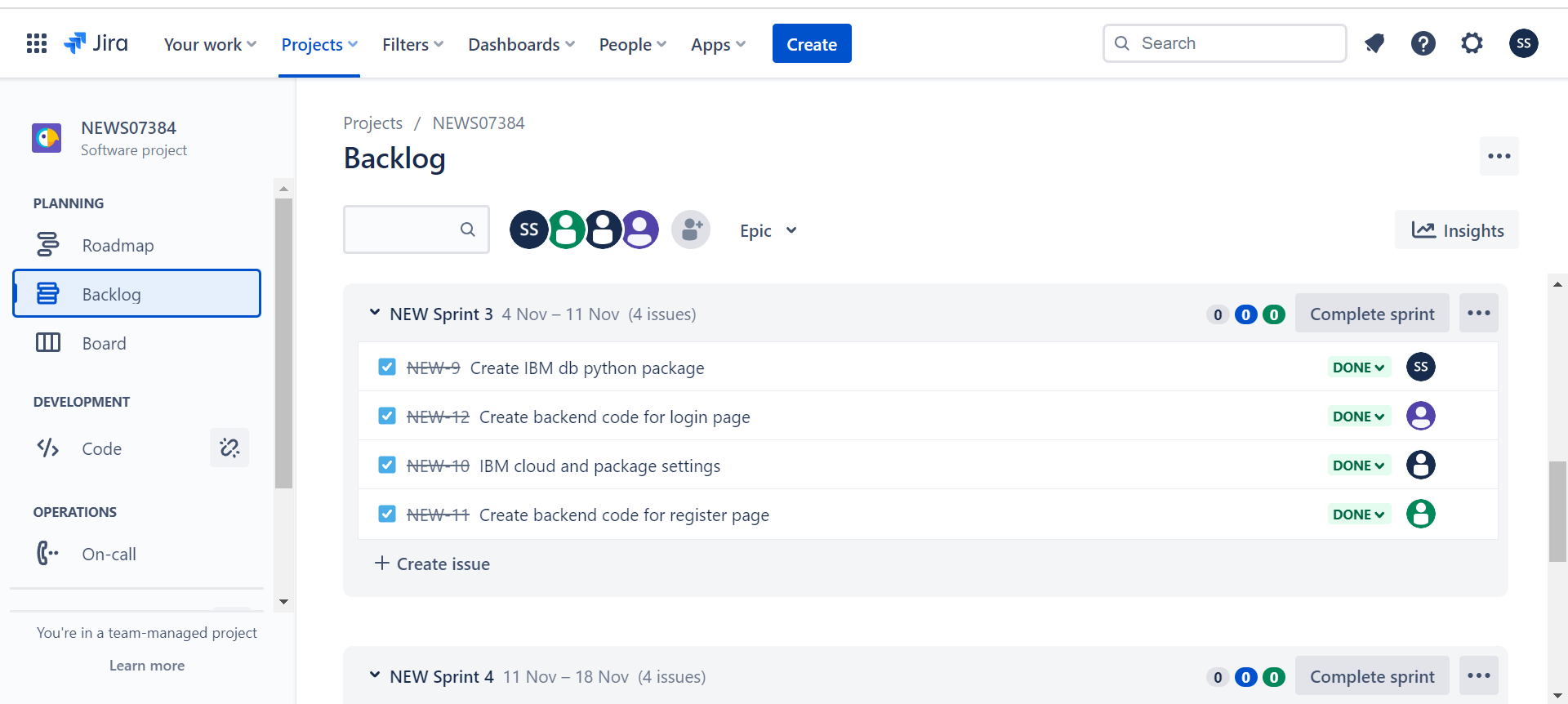
**Sprint in progress**



**Sprint is done**

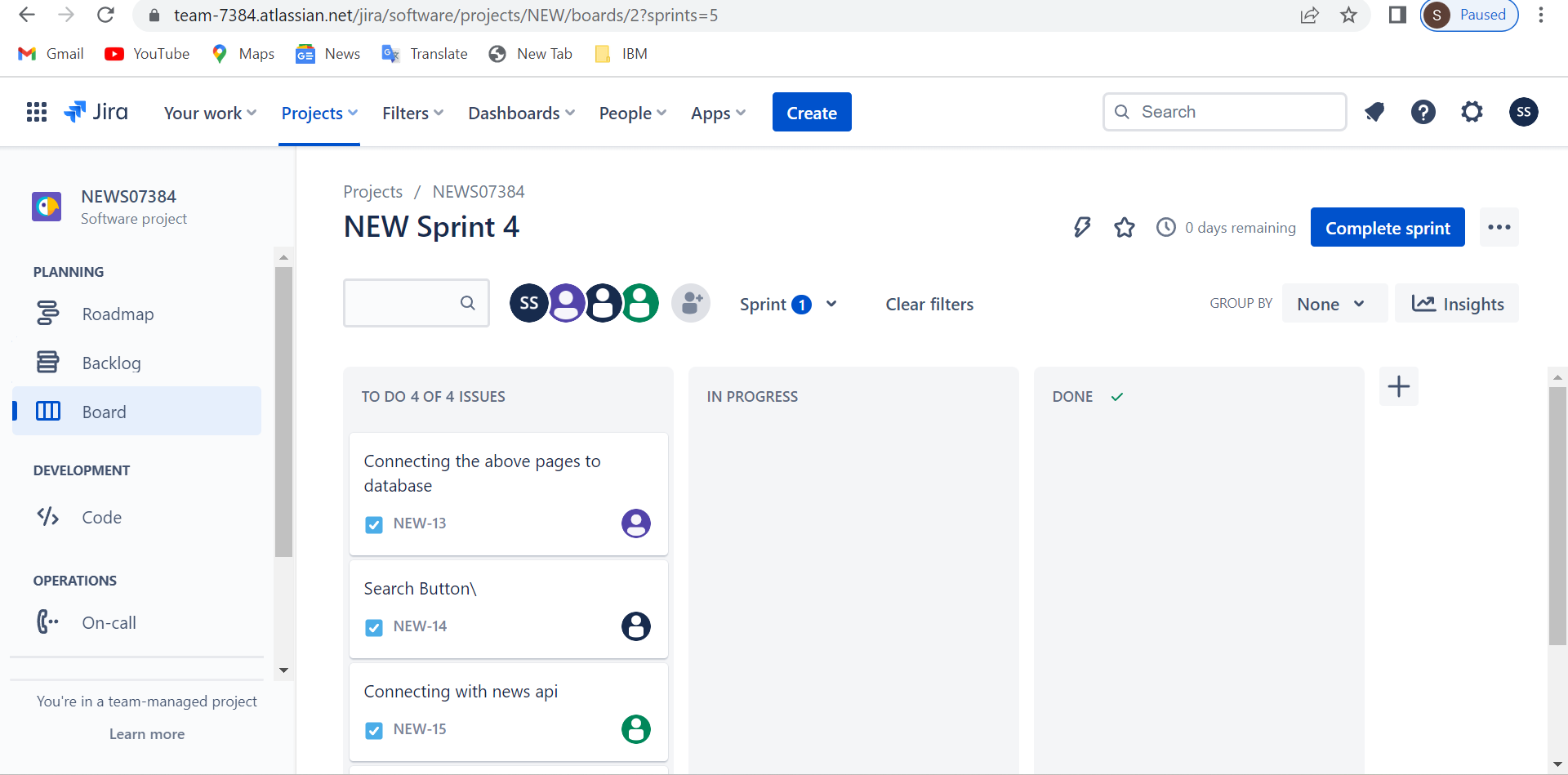


**Final Result - Sprint 3**

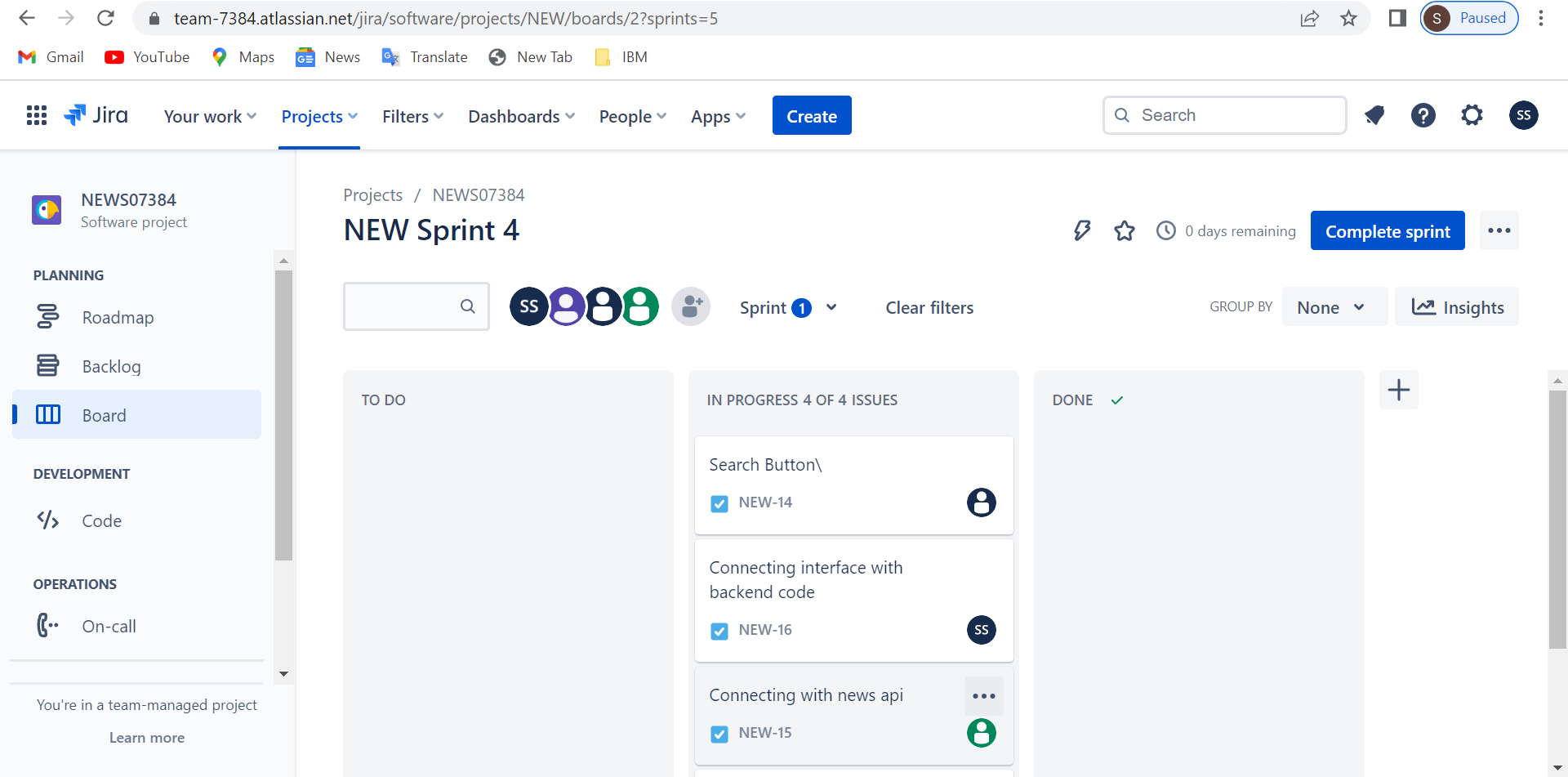


**Sprint - 4**

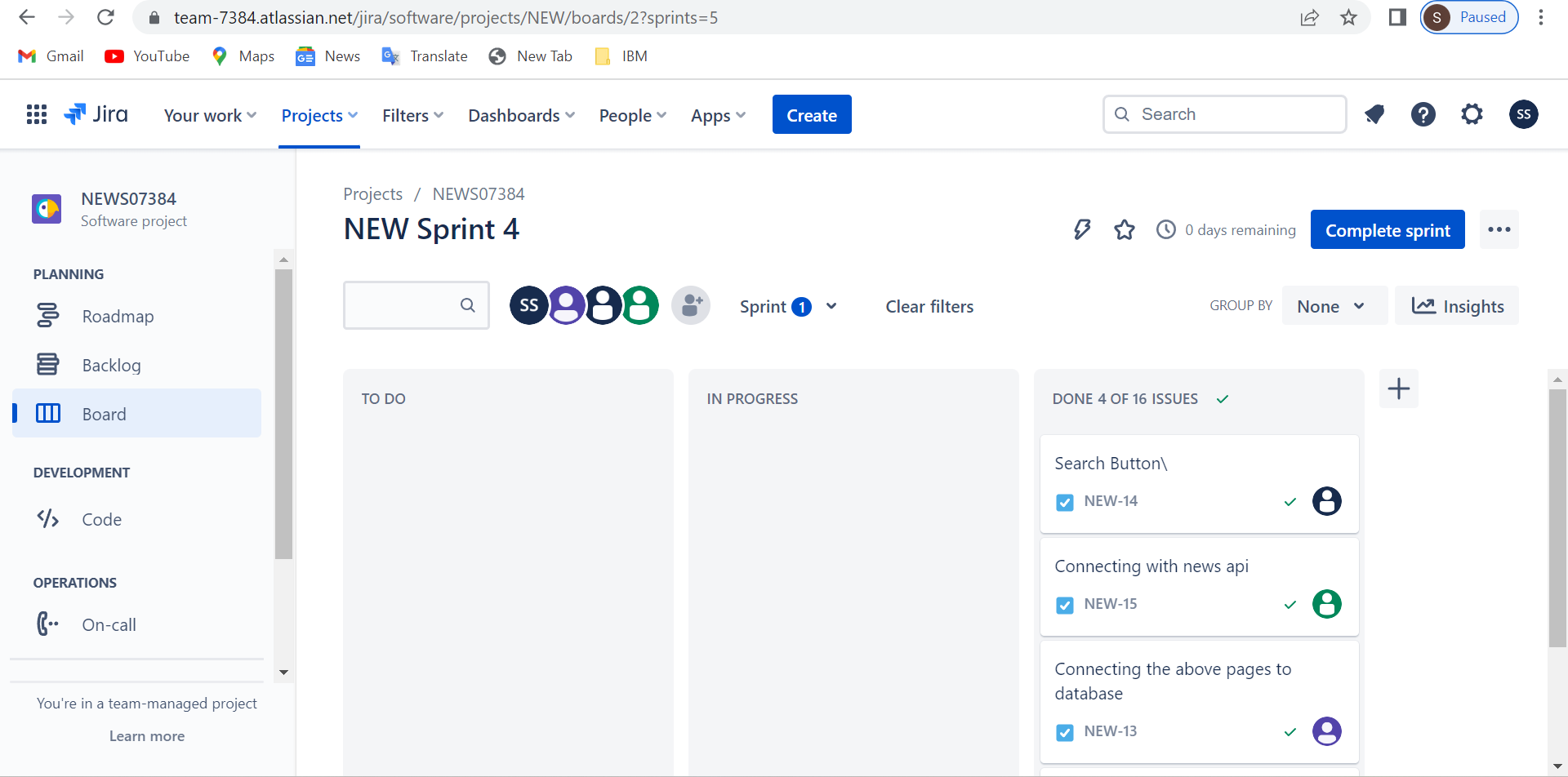
**Sprint in To Do**



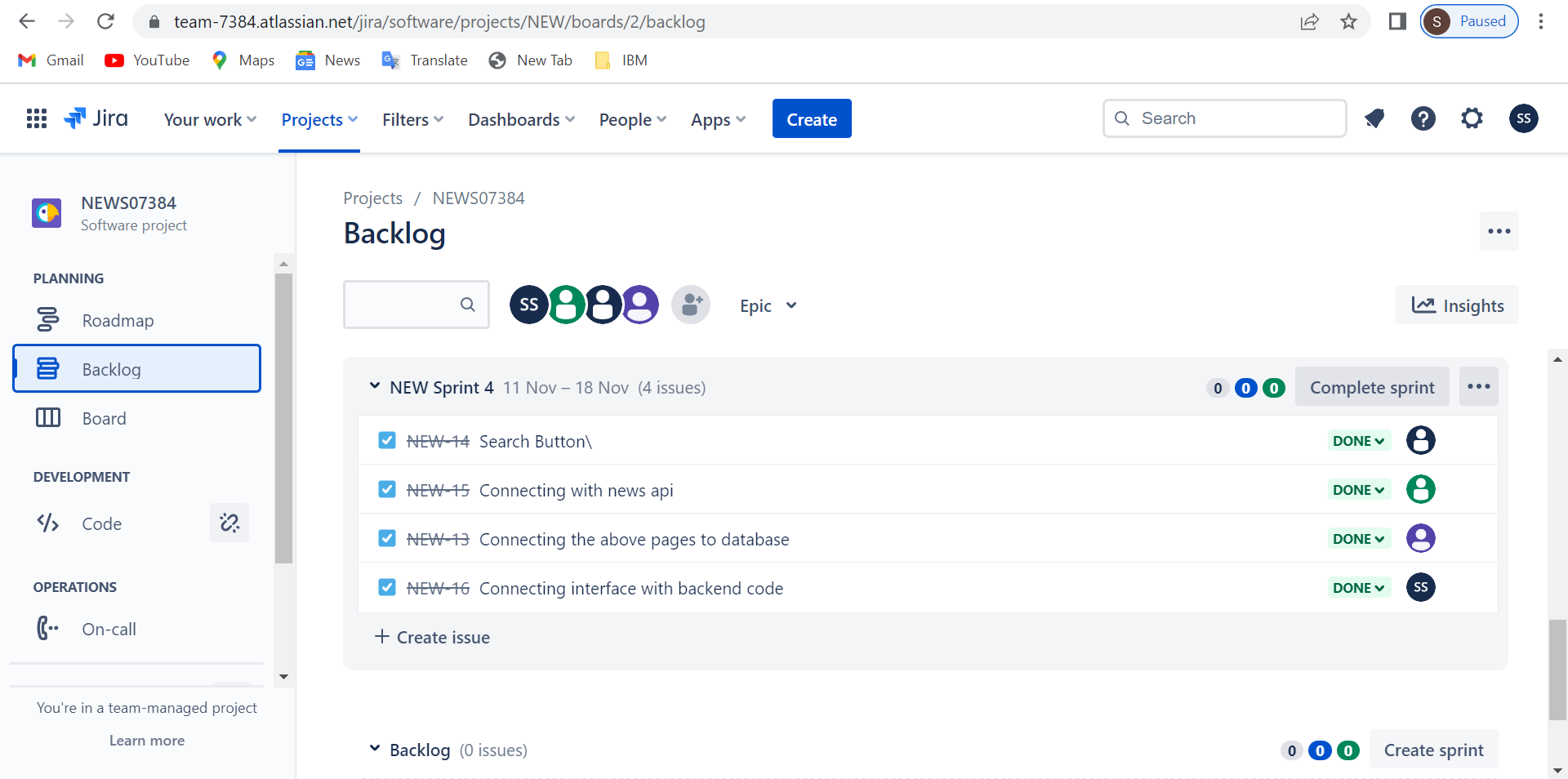
**Sprint in Progress**



**Sprint is Done**

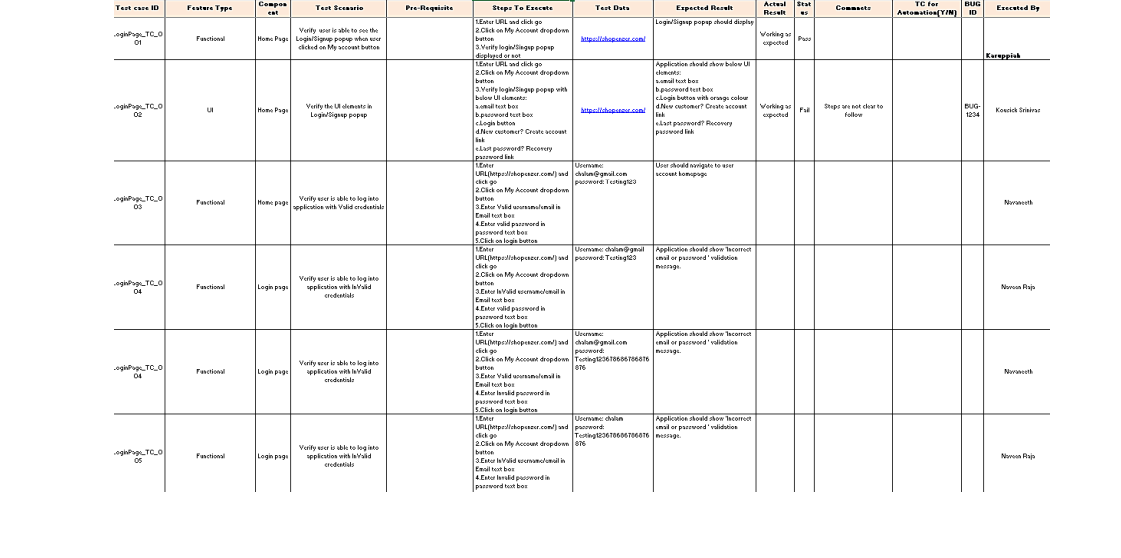


**Final Result - Sprint 4**



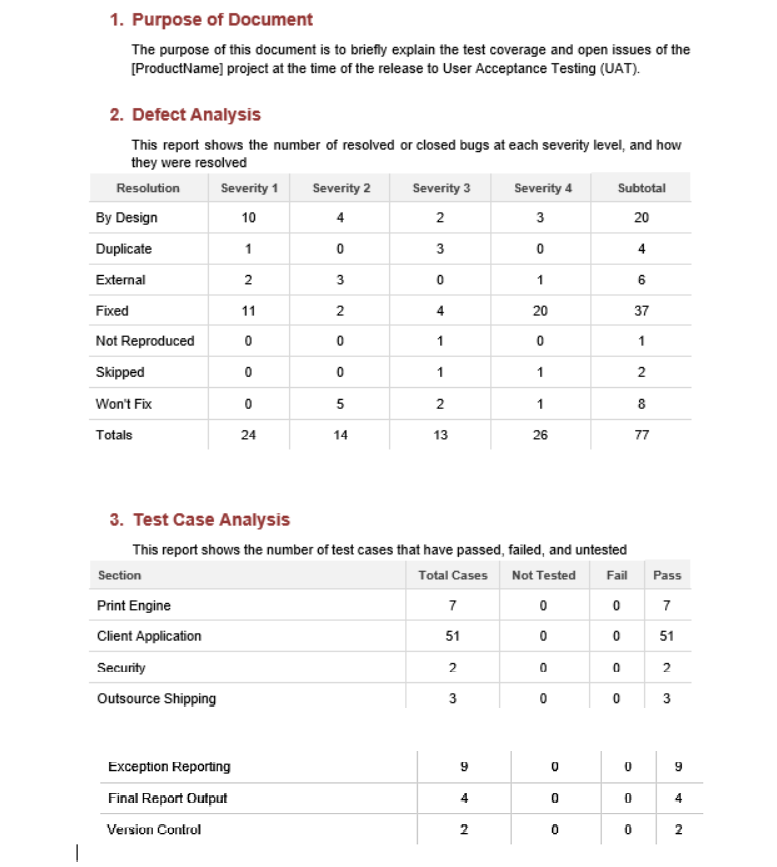
**TESTING**

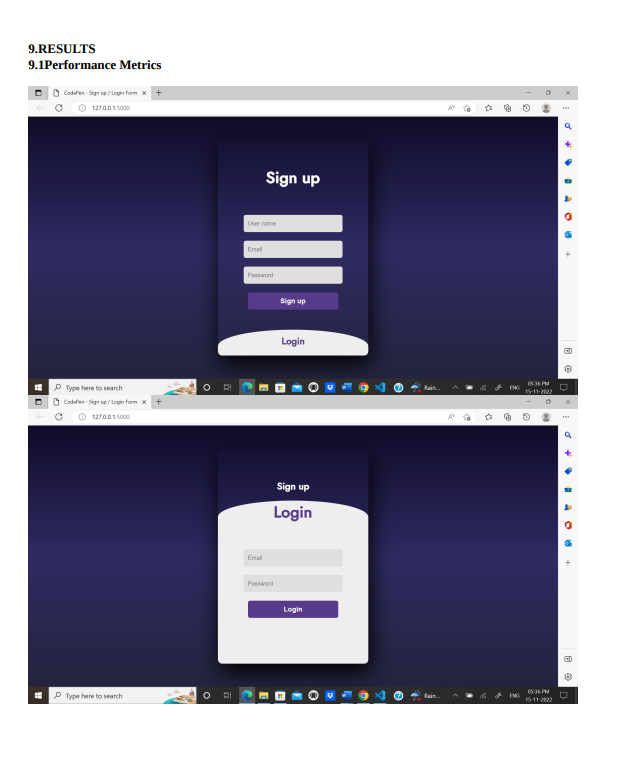
**USER ACCEPTANCE TESTING**

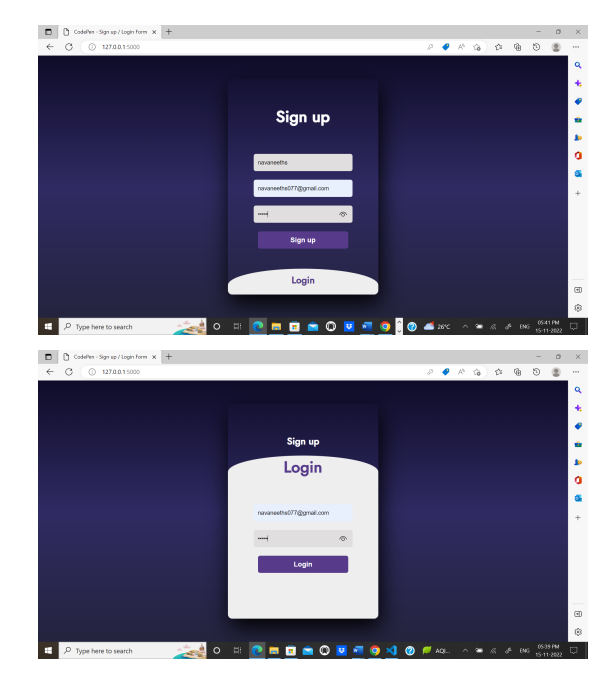
 **Acceptance Testing**

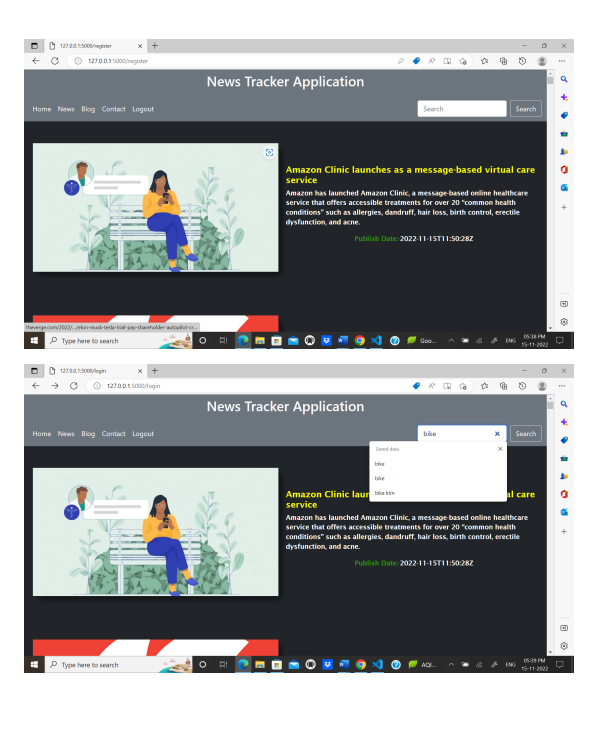
**UAT Execution & Report Submission**

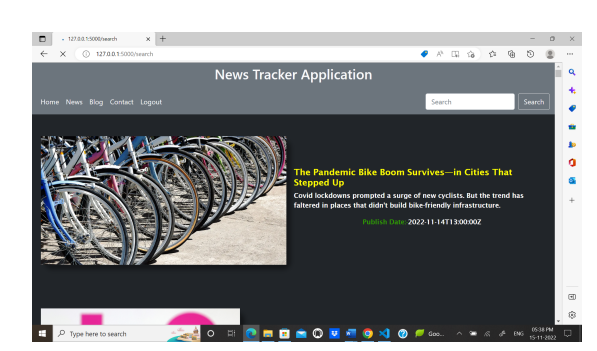
|  |  |
| --- | --- |
| **Team ID** | **PNT2022TMID07300** |
| **Project Name** | **New paper Tracking** |
| **Maximum Mark** | **4 Marks** |











**ADVANTAGES**

1. All trending news will be displayed in a orderly manner.

2. We can costomize the topic which we are interested and make them display.

3. The user experience are quit simple and friendly.

**DISADVANTAGES**

1. Same news which is published in different websites can be displayed.

2. Delay response of news api can make the news collapse.

**CONCLUSION**

Thus the projected using IBM CLOUD is tested,verified and executed successfully.

**FUTURE SCOPE**

In future, we planned to implement this project in large scale which will be helpful and used by all the people.

**CODING AND SOLUTIONING**

#.py file

#login page

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

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},app.app.config['SECRET\_KEY']

)

response.set\_cookie("access\_token",str(access\_token),httponly=True)

response.set\_cookie("email",str(email),httponly=True)

return response

return{"status":"Successfully Logged in"},200

#register page

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

#getnews

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

#verify

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

#apiFetch

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)

#cookie checker

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'])

except:

resp = {"status":"not logged in"}

@after\_this\_request

def deleter(response):

response.delete\_cookie("access\_token", domain="127.0.0.1",

path="/")

response.delete\_cookie("email",domain="127.0.0.1",path="/")

return response

return resp, 401

return f(data['email'],\*args, \*\*kwargs)

return decorated

#dbconfig

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\_port,d

sn\_protocol,dsn\_uid,dsn\_pwd)

return dsn

#dbQuery

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

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

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

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

#.css file

#index.css

@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@100;300&display=swap');  
\* {  
 padding: 0;  
 margin: 0;  
 box-sizing: border-box;  
 font-family: 'Roboto', sans-serif;  
}  
body {  
 display: flex;  
 flex-direction: column;  
 width: 100vw;  
 /\* background-image: url("../assets/pexels-cottonbro-3944418\ 1.png");  
 background-repeat: none;  
 background-size: cover; \*/  
}  
nav {  
 display: flex;  
 flex-direction: column;  
 justify-content: space-between;  
 align-items: center;  
 margin: auto;  
 min-width: 100%;  
 box-shadow: 0 0 5px rgba(35, 35, 35, 0.584);  
 /\* padding: 1rem; \*/  
 background-color: #d9d9d9be;  
}  
.title\_container {  
 display: flex;  
 justify-content: space-between;  
 align-items: center;  
 width: 100%;  
 margin: auto;  
}  
.search\_container {  
 display: flex;  
 justify-content: space-between;  
 padding: 5px;  
}  
.menu\_container {  
 display: flex;  
}  
.menu\_container h3 {  
 padding: 1rem 1rem 0.1rem 1rem;  
}  
.menu\_container h3:first-child {  
 border-bottom: solid 3px black;  
}  
#ham\_icon {  
 padding: 1rem;  
}  
#user\_icon {  
 padding: 1rem;  
}  
#home {  
 display: flex;  
 flex-direction: column;  
 justify-content: flex-start;  
 align-items: center;  
 width: 100%;  
 padding: 2rem 2rem 0 2rem;  
}  
.news\_cont {  
 display: flex;  
 box-shadow: rgba(149, 157, 165, 0.2) 0px 8px 24px;  
 transition: 0.5s ease;  
 border-radius: 10px;  
 padding: 1rem;  
 margin-bottom: 1rem;  
}  
.news\_cont:hover {  
 box-shadow: rgba(17, 17, 26, 0.1) 0px 1px 0px, rgba(17, 17, 26, 0.1) 0px 8px 24px, rgba(17, 17, 26, 0.1) 0px   
16px 48px;  
}  
.news\_thumbnail {  
 border-radius: 5px;  
 width: 300px;  
 height: auto;  
}  
.news\_content {  
 padding-left: 1.5rem;  
}  
.news\_content h2 {  
 padding-bottom: 0.5rem;  
}

#login.css

@import url('https://fonts.googleapis.com/css2?family=Khand:wght@300&display=swap');  
:root{  
 --blue:#2A85F8;  
 --blue2:#010B2E;  
 --gradient:linear-gradient(360deg, hsla(213, 94%, 57%, 1) 0%, hsla(213, 94%, 57%, 1) 14%, hsla(227, 96%,   
9%, 1) 81%);  
 --gradient2:linear-gradient(90deg,#009ffd, #2a2a72);  
 --gradient3:linear-gradient(to bottom left, #414141 -19%, #000000 100%);  
}  
\*{  
 font-family:'Khand', sans-serif;  
 margin:0;  
 padding:0;  
 border:none;  
 outline: none;  
 transition:all .2s linear;  
 box-sizing: border-box;  
 text-decoration: none;  
}  
\*::selection{  
 background:white;  
 color:var(rgb(145, 145, 207));  
}  
/\*html header font size\*/  
html{  
 font-size: 62.5%;  
 overflow-x: hidden;  
}  
body{  
 background-color:#f6f5f7;  
 display:flex;  
 justify-content:center;  
 align-items: center;  
 height:100vh;  
 margin:0 auto;  
 background:url(../assets/sample2.jpg)no-repeat;  
 background-size:cover;  
 background-position:center;  
 background-attachment:fixed;  
 position:relative;  
}  
.container{  
 background-color: white;  
 border-radius: 10px;  
 width: 768px;  
 max-width: 100%;  
 min-height: 480px;  
 position: relative;  
 overflow: hidden;  
 box-shadow: 0px 14px 28px rgba(0,0,0,0.25),  
 0px 14px 10px rgba(0,0,0,0.22);  
}  
.form-container{  
 position:absolute;  
 top:0%;  
 height:100%;  
 transition:all 0.8s ease-in-out;  
}  
.sign-up-container{  
 left:0;  
 width:50%;  
 z-index:1;  
 opacity:0;  
}  
form{  
 background-color: white;  
 display:flex;  
 align-items: center;  
 justify-content: center;  
 flex-direction: column;  
 padding:0 50px;  
 height:100%;  
 text-align: center;  
}  
h1{  
 font-weight: bold;  
 font-size:2.8rem;  
}  
h2  
{  
 color:red;  
}  
a{  
 color:#333;  
 font-size:2.5rem;  
 margin:15px;  
}  
span{  
 font-size:2rem;  
}  
.form-container form .box{  
 width:100%;  
 border-radius: 10px;  
 border:.2rem solid #666;  
 font-size:1.6rem;  
 color: #333;  
 margin:0.5rem;  
 padding:0.5rem;  
}  
/\*input box text adjustment\*/  
.form-container form .box::placeholder{  
 font-style:normal;  
}  
/\*underline text adjustment\*/  
.form-container form .box:focus{  
 border: 1px solid;  
 border-radius:10px;  
 border-color:rgba(135, 140, 169, 0.822);  
}  
.sign-in-container{  
 left:0;  
 width:50%;  
 z-index:2;  
}  
.overlay-container{  
 position:absolute;  
 top:0;  
 left:50%;  
 width:50%;  
 height:100%;  
 overflow:hidden;  
 z-index:100;  
 transition:all 0.6s ease-in-out;  
}  
.overlay{  
 background-color:var(--blue);  
 background: linear-gradient(to bottom,var(--blue),var(--blue2));  
 background-repeat: no-repeat;  
 background-size:cover;  
 background-position: 0 0;  
 color:white;  
 position:relative;  
 left:-100%;  
 width:200%;  
 height:100%;  
 transform: translateX(0);  
 transition:all 0.6s ease-in-out;  
}  
.overlay-panel{  
 position: absolute;  
 display: flex;  
 flex-direction: column;  
 width:50%;  
 height:100%;  
 top:0;  
 align-items: center;  
 justify-content: center;  
 transform:translateX(0);  
 padding:0rem 2.5rem;  
}  
.overlay-left{  
 transform: translateX(-20%);  
}  
.overlay-right{  
 right:0;  
 transform: translateX(0);  
}  
.container.right-panel-active .sign-in-container{  
 transform: translateX(100%);  
}  
.container.right-panel-active .sign-up-container{  
 transform: translateX(100%);  
 opacity: 1;  
 z-index: 5;  
 animation :slide 0.6s;  
}  
@keyframes slide{  
 0%,49.99%{  
 opacity:0;  
 z-index:1;   
 }  
 50%,100%{  
 opacity:1;  
 z-index:5;  
 }  
}  
.btn{  
 font-weight:bold;  
 font-size:1.5rem;  
 border-radius: 2rem;  
 text-transform: capitalize;  
 letter-spacing:1px;  
 color:white;  
 background:var(--gradient2);  
 padding:0.8rem 2rem 0.8rem 2rem; /\*top right bottom left\*/  
 margin-top:1.5rem;  
 margin-bottom: 5rem;  
 position: relative;  
 z-index: 0;  
 overflow: hidden;  
 box-shadow: 0 .1rem .3rem rgba(255, 0, 0, 0.822);  
 cursor:pointer;  
}  
.btn:active{  
 transform: scale(.85);  
}  
.btn::before{  
 content:'';  
 position:absolute;  
 top:-1rem;  
 left:-100%;  
 background: #fff9;  
 height:150%;  
 width:20%;  
 transform:rotate(25deg);  
 z-index:-1;  
}  
.btn:hover::before {  
 transition:.5s linear;  
 left:120%;  
}  
.container.right-panel-active .overlay-container{  
 transform: translateX(-100%);  
}  
.container.right-panel-active .overlay{  
 transform:translateX(50%);  
}  
.container.right-panel-active .overlay-left{  
 transform: translateX(0);  
}  
.container.right-panel-active .overlay-right{  
 transform: translateX(20%);  
}  
.container-checkbox {  
 max-width: 640px;  
 font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;  
 font-size: 13px;  
}  
ul.ks-cboxtags {  
 list-style: none;  
 padding: 20px;  
}  
ul.ks-cboxtags li{  
 display: inline;  
}  
ul.ks-cboxtags li label{  
 display: inline-block;  
 background-color: rgba(255, 255, 255, .9);  
 border: 2px solid rgba(139, 139, 139, .3);  
 color: #000;  
 border-radius: 25px;  
 white-space: nowrap;  
 margin: 3px 0px;  
 -webkit-touch-callout: none;  
 -webkit-user-select: none;  
 -moz-user-select: none;  
 -ms-user-select: none;  
 user-select: none;  
 -webkit-tap-highlight-color: transparent;  
 transition: all .2s;  
}  
ul.ks-cboxtags li label {  
 padding: 8px 12px;  
 cursor: pointer;  
}  
ul.ks-cboxtags li label::before {  
 display: inline-block;  
 font-style: normal;  
 font-variant: normal;  
 text-rendering: auto;  
 -webkit-font-smoothing: antialiased;  
 font-family: "Font Awesome 5 Free";  
 font-weight: 900;  
 font-size: 12px;  
 padding: 2px 6px 2px 2px;  
 content: "\f067";  
 transition: transform .3s ease-in-out;  
}  
ul.ks-cboxtags li input[type="checkbox"]:checked + label::before {  
 content: "\f00c";  
 transform: rotate(-360deg);  
 transition: transform .3s ease-in-out;  
}  
ul.ks-cboxtags li input[type="checkbox"]:checked + label {  
 border: 2px solid #1bdbf8;  
 background-color: #12bbd4;  
 color: #fff;  
 transition: all .2s;  
}  
ul.ks-cboxtags li input[type="checkbox"] {  
 display: absolute;  
}  
ul.ks-cboxtags li input[type="checkbox"] {  
 position: absolute;  
 opacity: 0;  
}  
ul.ks-cboxtags li input[type="checkbox"]:focus + label {  
 border: 2px solid #e9a1ff  
}

#verify.css

@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@100;300&display=swap');  
\*,\*::before,\*::after{  
 margin: 0;  
 padding: 0;  
 box-sizing: border-box;  
 font-family: 'Roboto', sans-serif;  
}  
body{  
 background-color: aliceblue;  
 min-height: 100vh;  
}  
nav {  
 display: flex;  
 flex-direction: column;  
 justify-content: space-between;  
 align-items: center;  
 margin: auto;  
 min-width: 100%;  
 box-shadow: 0 0 5px rgba(35, 35, 35, 0.584);  
 background-color: #d9d9d9be;  
 padding: 1rem;  
}  
.status-cont{  
 display: flex;  
 min-height: 70vh;  
 flex-direction: column;  
 align-items: center;  
 justify-content: center;  
 font-weight: medium;  
 text-align: center;  
 padding: 1rem;  
}  
.welcome-text{  
 font-size: 2.5rem;  
}  
.sub-text{  
 font-size: 1.5rem;  
}

#.html file

#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>  
 <div class="menu\_container">  
 <h3>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>  
 <section id="home">  
 <div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div>  
 <div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div>  
 <div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div>  
 <div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>

</ div><div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div><div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div><div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div><div class="news\_cont">  
 <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. Fugiat   
dolores eligendi ex vel harum nobis velit qui expedita rerum maiores possimus   
voluptates similique, doloremque quisquam accusantium impedit atque animi   
est.</p>  
 </div>  
 </div>  
 </section>  
</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 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>  
 </div>  
 <script src="../js/login.js" type="module"></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>

#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.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(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>')