UNIVERSITÉ
Concordia
UNIVERSITY

Gina Cody School, Department of electrical and computer engineering
Team: Red_raven
Application: Trends of Today

Submitted by:

Arshdeep Singh (40230807)
Maria Dona Thomas (40228451)
Gurleen Kaur (40206674)
Shivendra Arulalan (40197455)
Ragul Nivash Rangasamy Sekar (40169564)

Product Report

1. Introduction:

1.1. Project Description:

This project is focused on creating a trending news application which also works like a social media application. “Trends of Today” is a website that will show all the trending news of his/her area and interest, the website will retrieve user information and his/her preferences by the user activity such as location, recent searched news and his area of interest.

As soon as the user logs in trends of today's website, all the trending news would be displayed on his news feed. Along with viewing the news the website would allow the user to rate the news or share the news. If auser wants to read some news later then an option to read it later is also available, in this the user must add that news to his reading library.

The best feature that makes this project interesting is the news feed would show news according to the userstaste which would increase the user interaction of the app. This website enables the user to discover their friends on the app and keep a track on their news activity along with messaging features.

1.2 Proposed Solution:

Users can access the application via a unique user id and create a user profile. The application we build is a web-based application which fetches trending news, which is thenparsed according to the user's interest, and location. The application consists of a login dashboard, then a customized user news feed panel with search option. The following features are provided for the users.

Login and Registration

Web Service prompt for single login on first access to the application, and eventually support browsing andutilizing the website without re-logging in. The same login name and password generates a temporary certificate for the user to surf through the website whenever necessary. To log out, the user must hover the point over his profile avatar and press the log out icon to log out.

Dashboard

News in the homepage will be fed with trending news along with a recommendation list. Each article will have its image, headline and a brief description, and redirects to the URL of the news to see the full article.Bookmarking features will be implemented which will help the user to read later or to save the article.

Search Bar

To view more news, a dropdown of trending news is suggested along with an option to do more specificsearch by particular words, location or categories will be provided.

Other Features

Rate news: Allows users to rate the news from scale of 1 to 5. Notification:
Fetches and updates the scoreboard at regular intervals.Finding friends,
instant messages, and following their feeds.

1.3. Process

Epic and user stories are defined for the project and each sprint would pick a set of user stories or epics todeliver the project by the due date. Technologies used for frontend are HTML, CSS and JavaScript, for backend is Flask and Git is used to maintain the repositories. The main epics are:

- Registration Module
- Dashboard 1.1
- Dashboard 1.2
- User Involvement Module
- Improving User experience Module
- User Interaction and Profile Management ModuleThe whole process however have four main stages:

a. Software specification: The application is a trending news application and is supposed to show the trending news for a scalable user base.

b. Design and Implementation: The UI/UX flow of the software should be easy to understand. Number of clicks required to perform a particular use case will be kept minimum. Wireframes aredesigned before the implementation.

c. Software Validation: Application code will be compiled and executed multiple times to check theaccuracy of its functioning and verified with the system requirements**.**

d. Software Evolution: The project can be updated based on its usage and future changes canbe easily accommodated within the application.

1.4. PRODUCT PERSPECTIVE

For this project, we are going to build a web-based application which fetches trending news, which is thenparsed according to user's interest, and location. The application consists of a login dashboard, then a customized user news feed panel with search option.

1.5. PROJECT GOAL

To enable users to have access to trending news, as well as connect with the world through a user and environment-friendly web-based trending news feed application

1.6. PRODUCT USERS AND STAKEHOLDERS

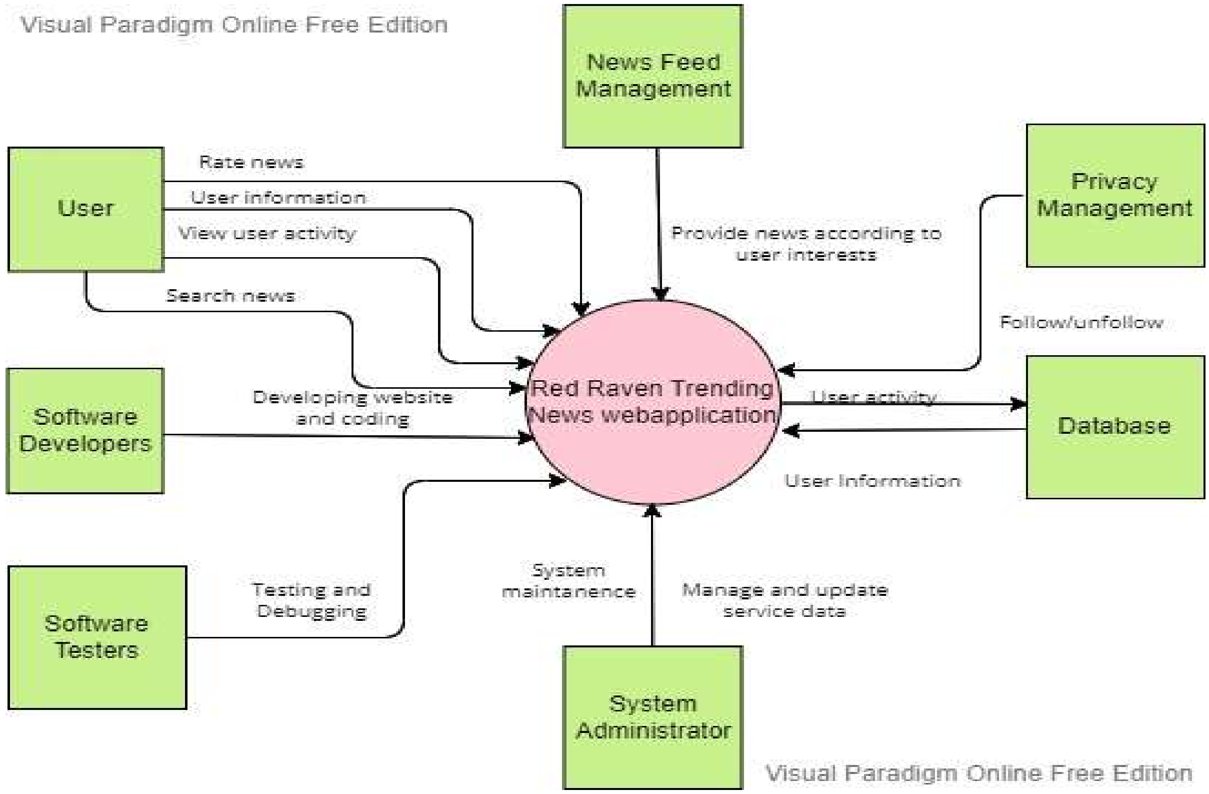
The following are the users and stakeholders for our product:

- **Engineers**: These are the primary stakeholders of the product as they are brain behind the designand implementation of the product
- **Board of directors**: They hold a higher importance in the product succeeding as they are the primary investors in the product. They are also vital in making key decisions as concerned with theproduct. In our case, our professor Tariq Daradekh and POD Salah Harb are board of directors.
- **Miscellaneous holders**: We can classify them as companies, brands, small businesses etc who standto make profits from advertisements and other marketing strategies through our application.
- **General Public**: As this is a social news application anyone and anything stands the chance ofbeing noticed and in any corner of the world. Hence the stakes are high for everybody.

2. System Context:

In our system the user can perform various activities via the pages provided in our page like rating the news, viewing user activity, searching news. We have other systems involved in our project like newsfeed management to provide news according to user preference, database system which stores various data related to user and user activity. We also have software developers who has developed the code for the application, software testers who generate various test cases to verify the operation of the code and system administrator to maintain, manage and update service data.

The below diagram represents the system context diagram:

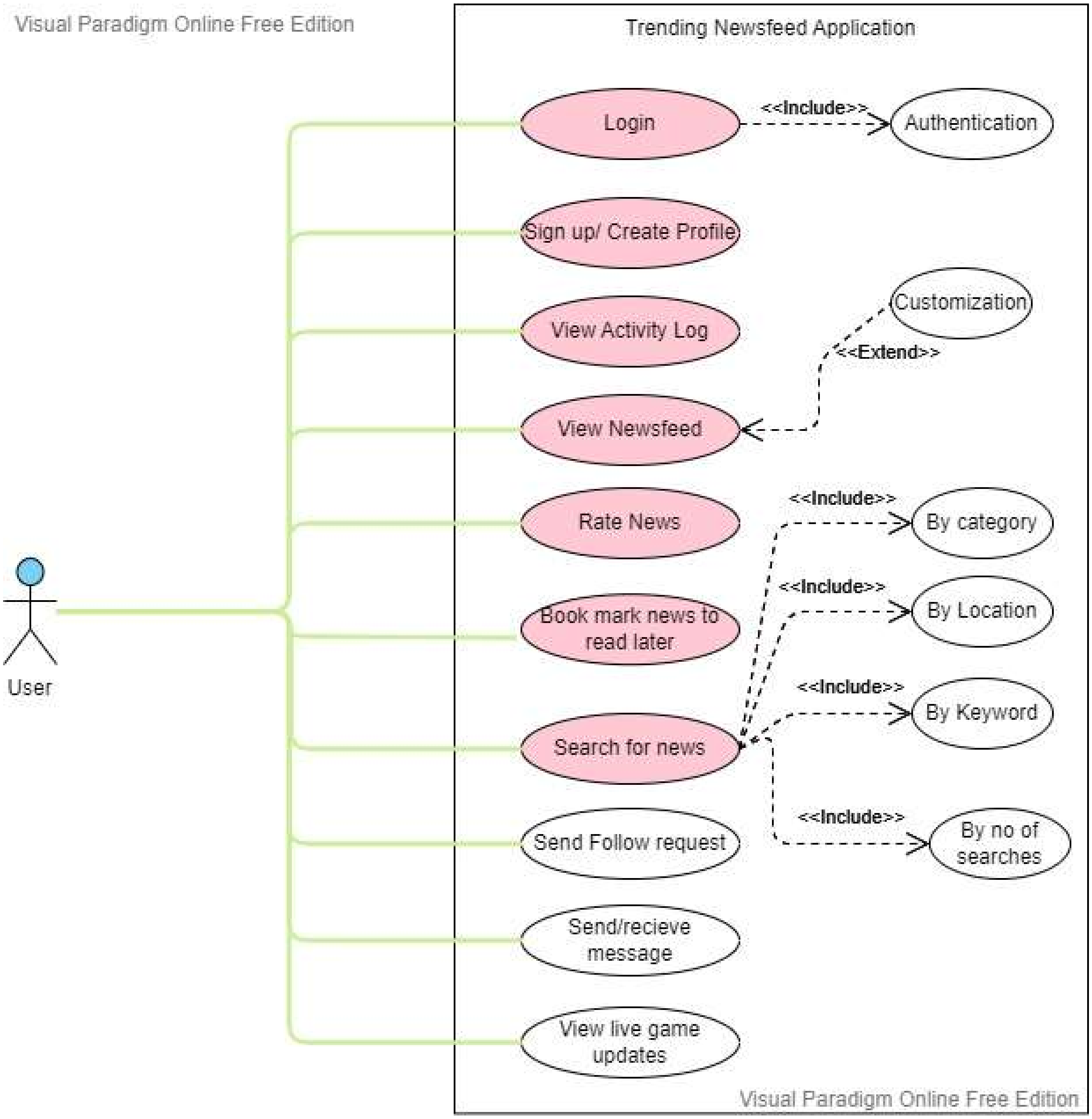


3. REQUIREMENTS

3.a. Our sprint goal was to complete as many user stories as possible maintaining the coding standards.

3.b. Use case diagram of the functionality to be implemented and an activity diagram

The following diagram represents our trending news web application for sprint 3



Note: The pink colored use cases have been implemented

By the end of the final sprint, we were successfully able to implement the functionalities shown below:

Function 1: Login

This function caters to 'Users'. It is a text input field i.e., Username and Password. The user is expected toprovide their correct registered credentials to the system. If there is any error, the system will prompt an error message denying access to their respective profiles.

Function 2: Sign-up/ Create Profile

This function allows the user to create their profile in the system for ease of access. The user is prompted tofill certain mandatory text fields i.e., name, email address, password. Once created, the user can access his/her profile via the login functionality.

Function 3: View News-Feed

This function displays the trending news around the user's location

Function 4: Search

This function lets the user search for news according to categories, keyword, number of searches andlocation.

Function 5: View Activity Log

This function provides the user the timestamp and the activity performed by the user like what news has been viewed or which news has been saved to read later etc.

Function 6: Rate News

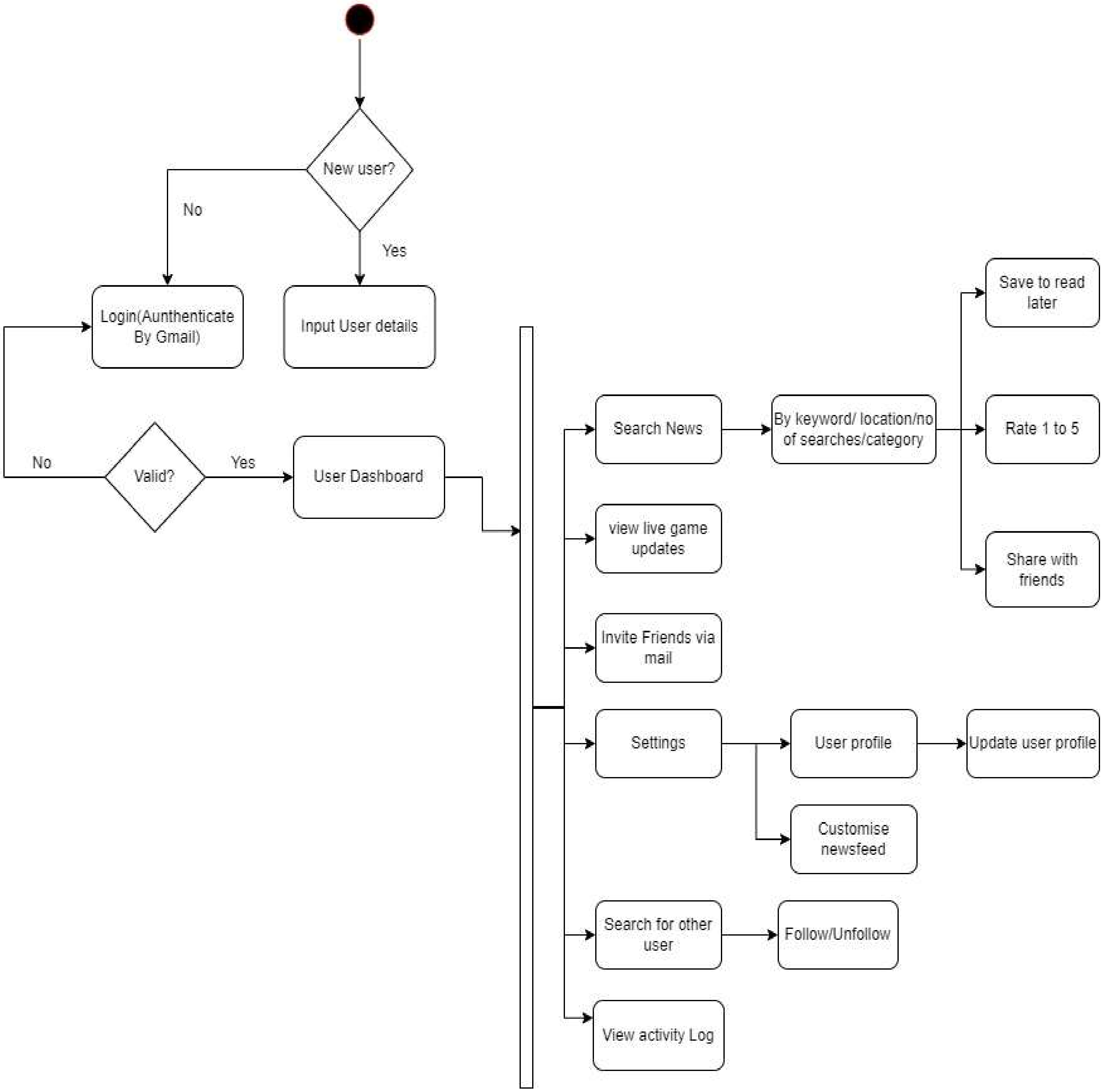
This function allows the user to rate the news viewed by the user from 1 to 5, 1 being the least and five being the highest.

Function 7: Bookmark news to read later

This feature allows the user to mark a news as read later so that when the user logs in the second time the news marked as read later would appear in the bookmark section of the application.

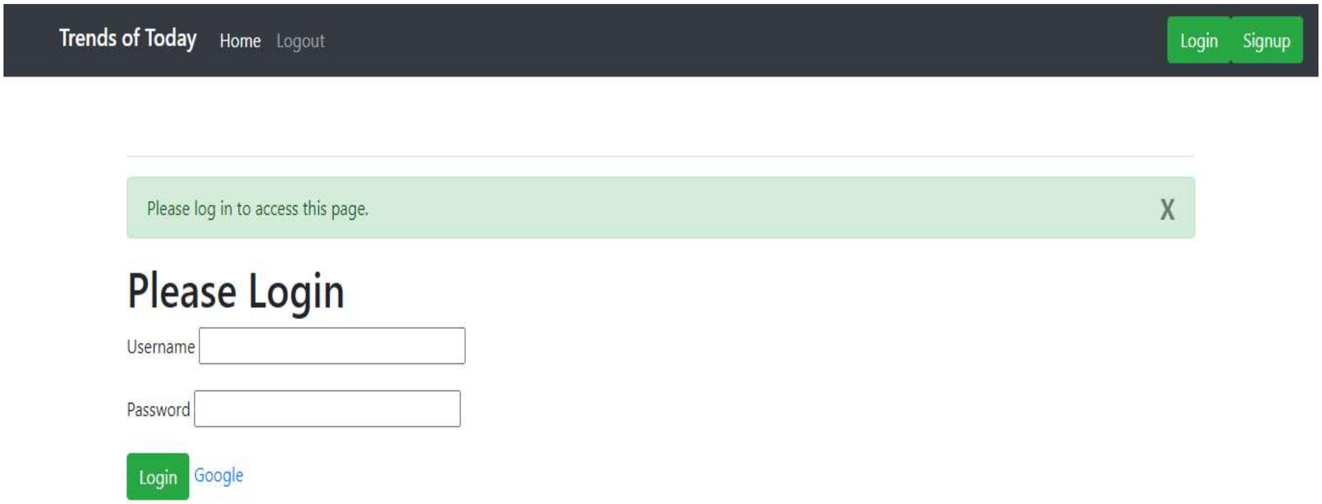
Activity Diagram:

The below diagram represents the main processes and the total functionality functionalities of the project.



3.c. Wireframes of the functionality to be implemented:

Login page design:



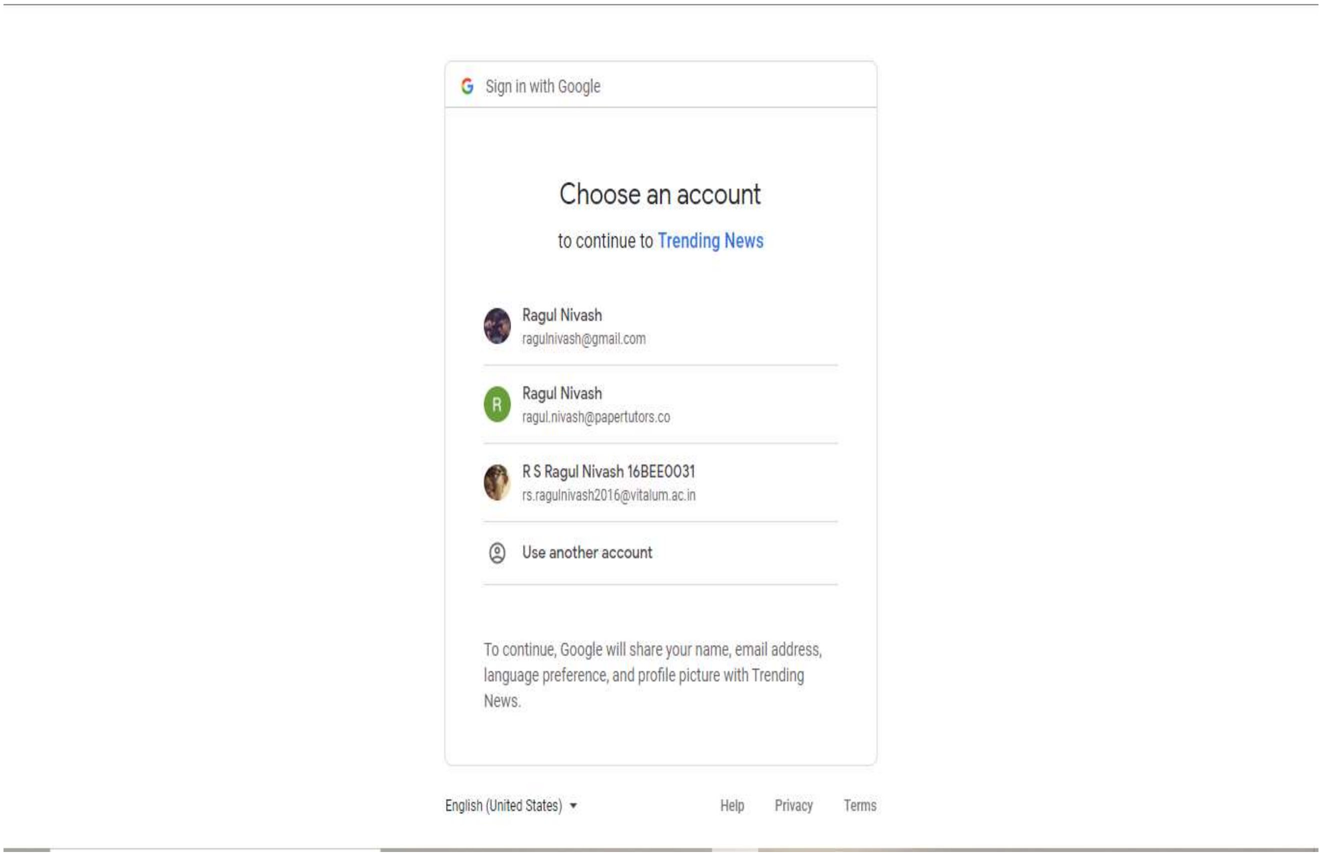
Please log in to access this page.

Please Login

Username

Password

Google O-Auth page design:



Sign in with Google

Choose an account

to continue to Trending News

Ragul Nivash
ragulnivash@gmail.com

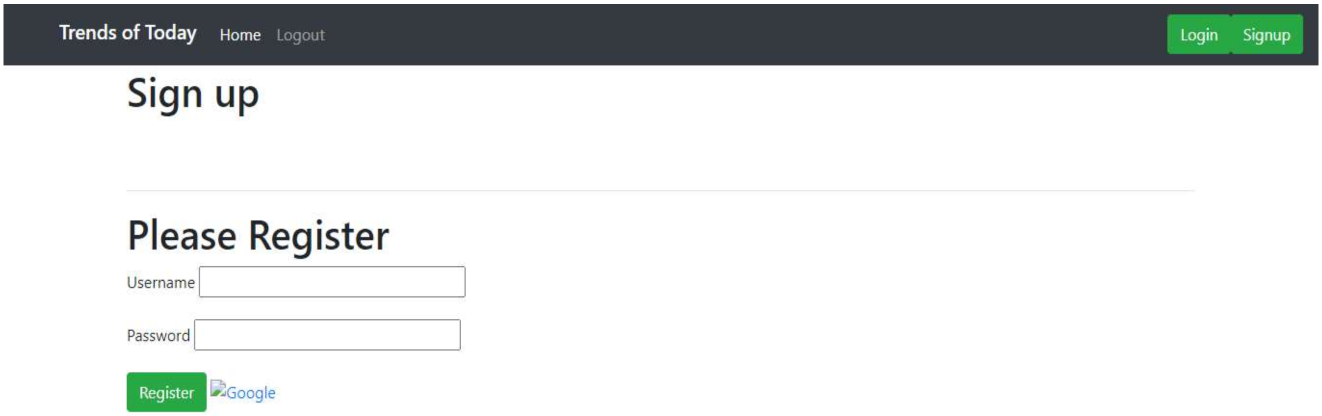
Ragul Nivash
ragul.nivash@papertutors.co

R S Ragul Nivash 16BEE0031
rs.ragulnivash2016@vitalum.ac.in

Use another account

To continue, Google will share your name, email address, language preference, and profile picture with Trending News.

Registration page:



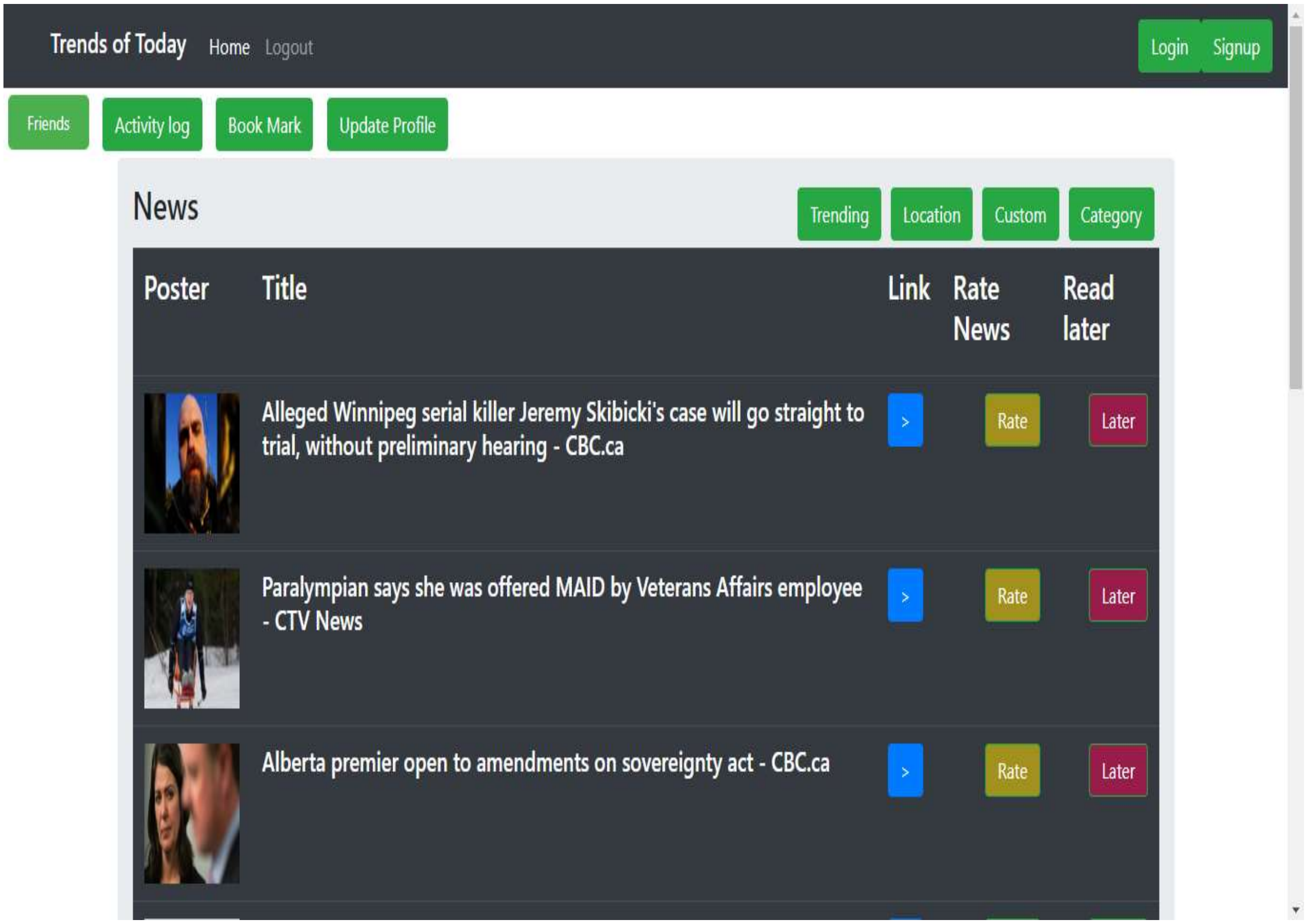
Sign up

Please Register

Username

Password

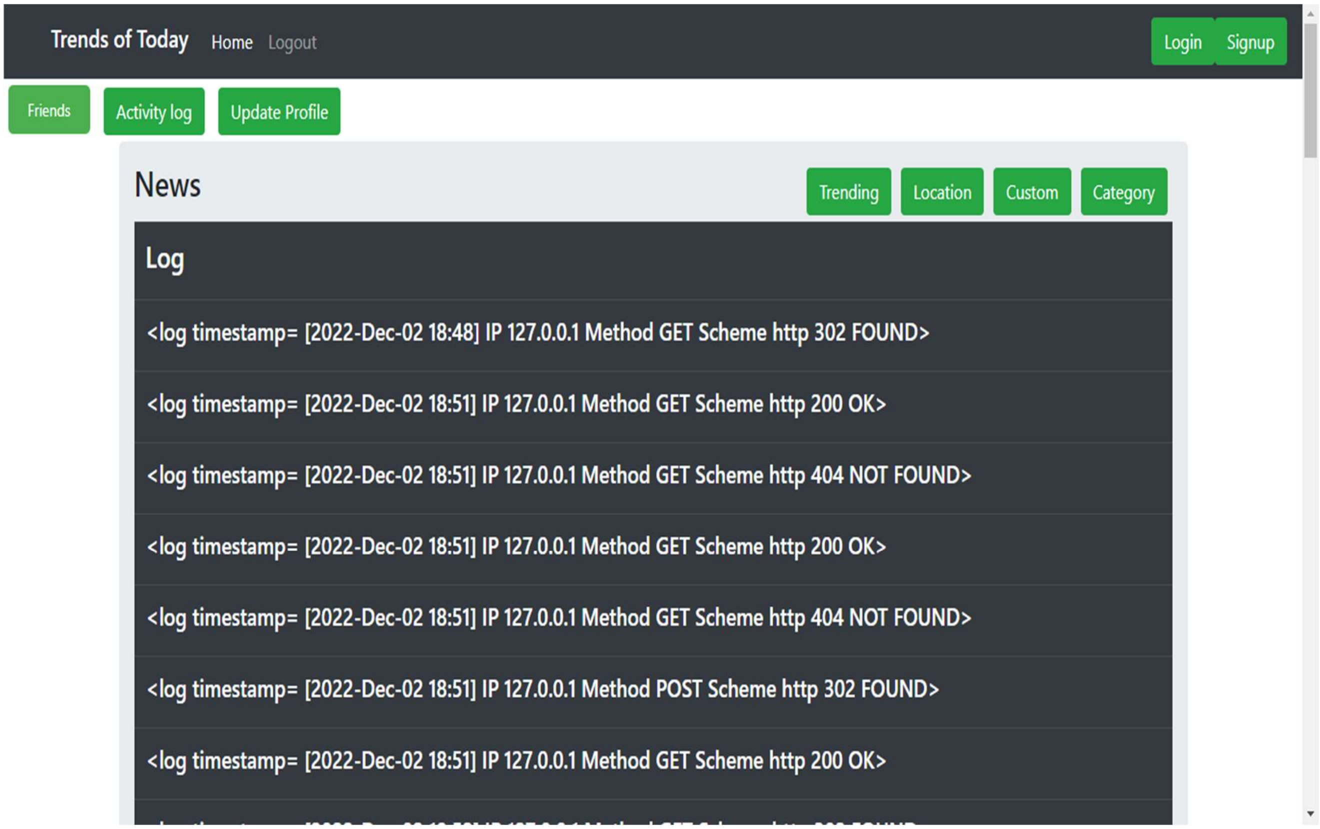
Landing page:



News

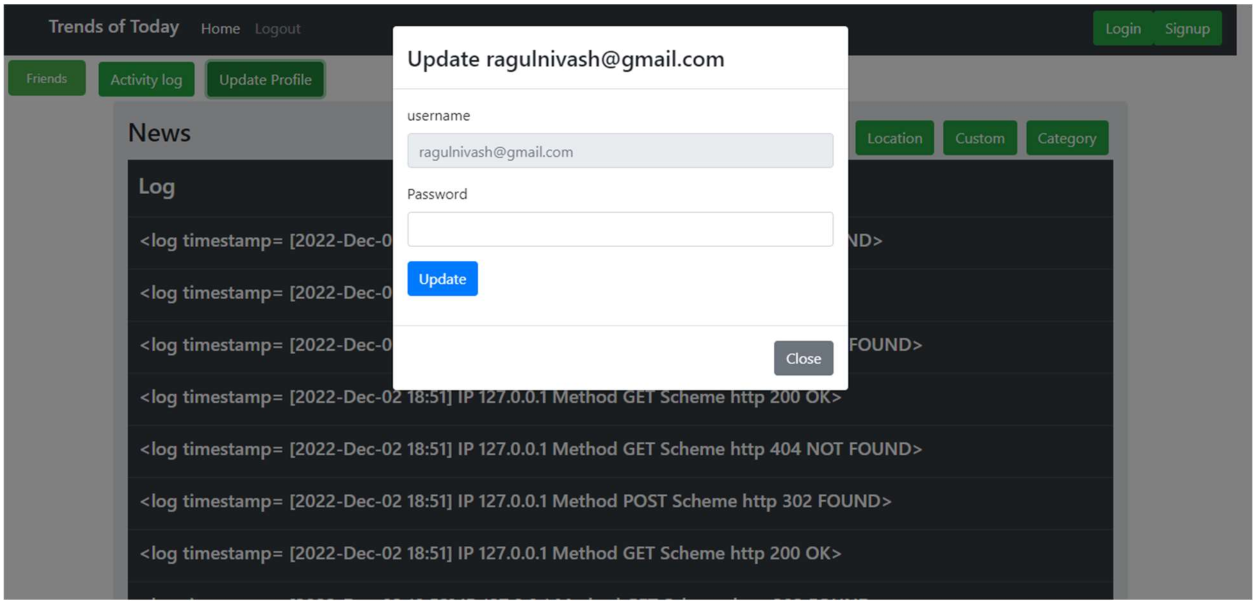
Poster	Title	Link	Rate News	Read later
	Alleged Winnipeg serial killer Jeremy Skibicki's case will go straight to trial, without preliminary hearing - CBC.ca	>	Rate	Later
	Paralympian says she was offered MAID by Veterans Affairs employee - CTV News	>	Rate	Later
	Alberta premier open to amendments on sovereignty act - CBC.ca	>	Rate	Later

Activity Log page:



```
Log
<log timestamp= [2022-Dec-02 18:48] IP 127.0.0.1 Method GET Scheme http 302 FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 200 OK>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 404 NOT FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 200 OK>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 404 NOT FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method POST Scheme http 302 FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 200 OK>
```

Update user profile password and location page design:



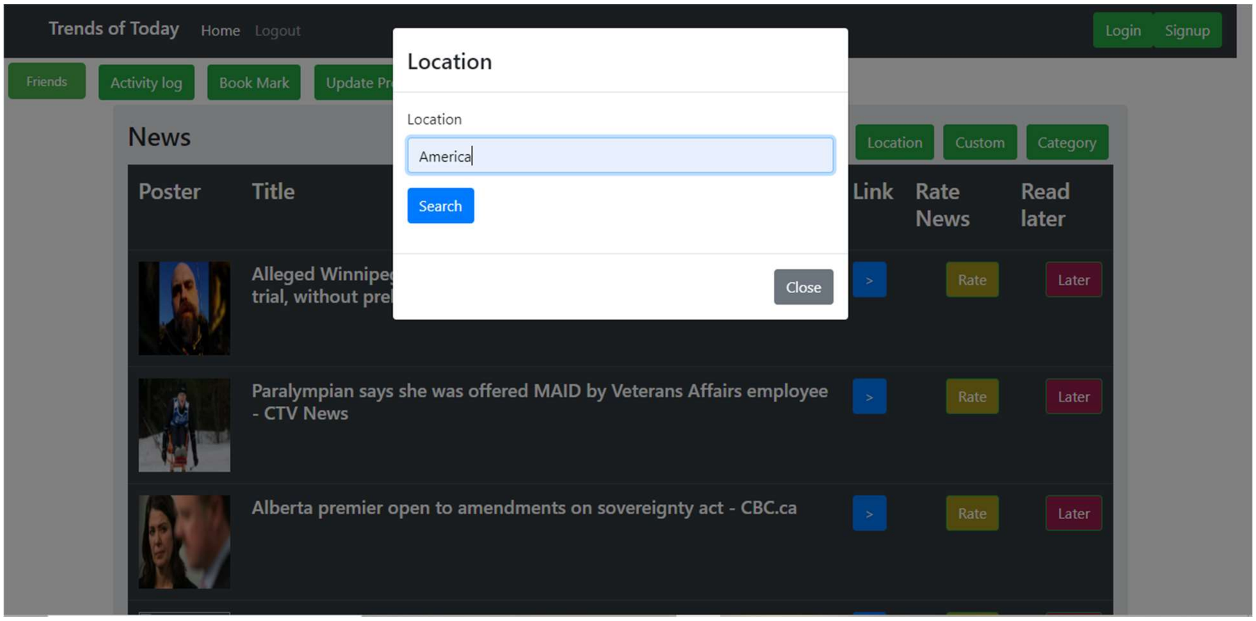
Update ragulnivash@gmail.com

username

ragulnivash@gmail.com

Password

```
Log
<log timestamp= [2022-Dec-0                                ND>
<log timestamp= [2022-Dec-0
<log timestamp= [2022-Dec-0                                FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 200 OK>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 404 NOT FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method POST Scheme http 302 FOUND>
<log timestamp= [2022-Dec-02 18:51] IP 127.0.0.1 Method GET Scheme http 200 OK>
```

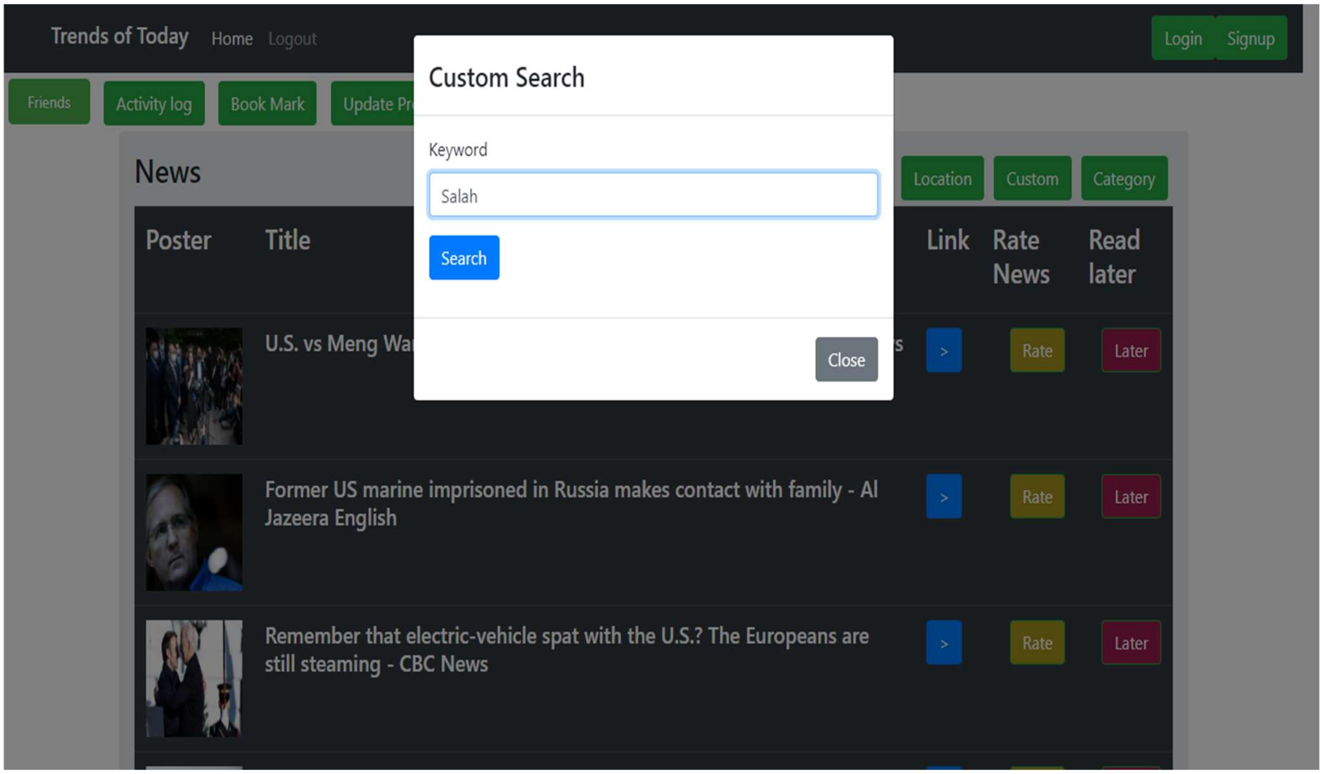


Location

Location: America

Poster	Title	Link	Rate News	Read later
	Alleged Winnipeg trial, without pre	>	Rate	Later
	Paralympian says she was offered MAID by Veterans Affairs employee - CTV News	>	Rate	Later
	Alberta premier open to amendments on sovereignty act - CBC.ca	>	Rate	Later

Custom search page design:

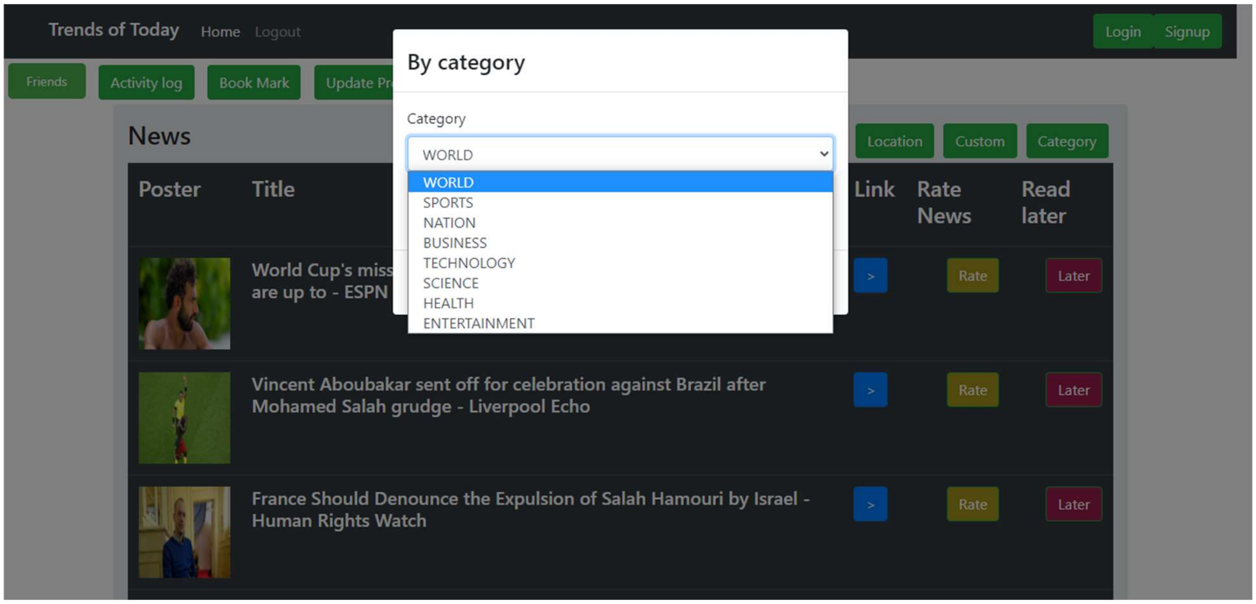


Custom Search

Keyword: Salah

Poster	Title	Link	Rate News	Read later
	U.S. vs Meng Wa	>	Rate	Later
	Former US marine imprisoned in Russia makes contact with family - Al Jazeera English	>	Rate	Later
	Remember that electric-vehicle spat with the U.S.? The Europeans are still steaming - CBC News	>	Rate	Later

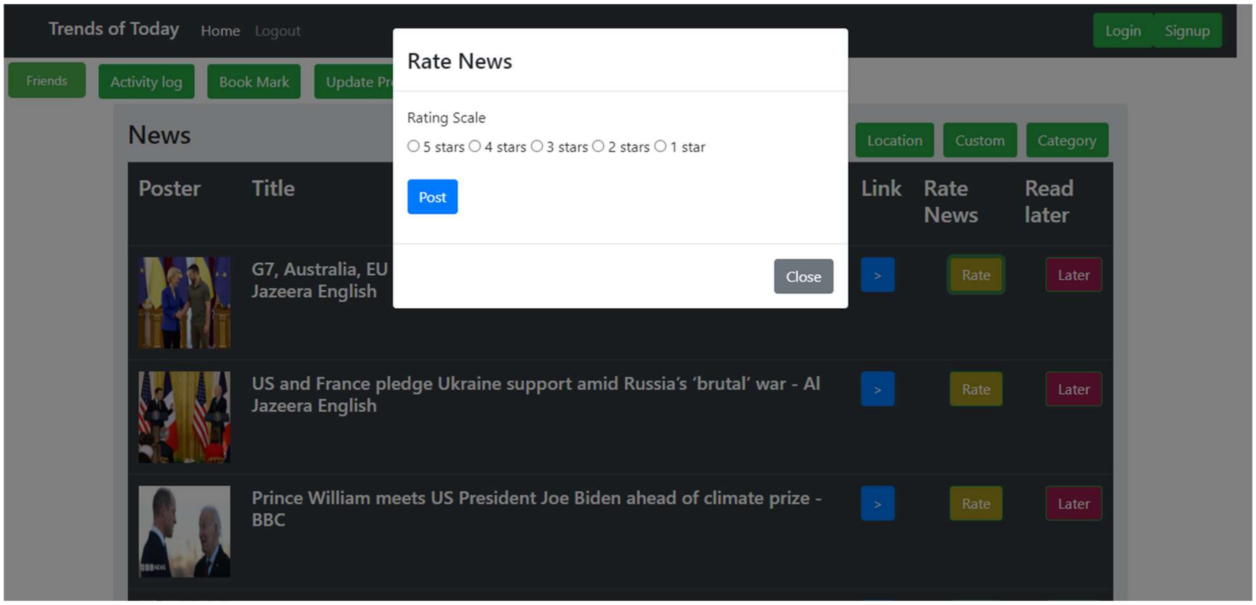
Search by category page design:


By category

Category: WORLD

Poster	Title	Link	Rate News	Read later
	World Cup's miss… are up to - ESPN	>	Rate	Later
	Vincent Aboubakar sent off for celebration against Brazil after Mohamed Salah grudge - Liverpool Echo	>	Rate	Later
	France Should Denounce the Expulsion of Salah Hamouri by Israel - Human Rights Watch	>	Rate	Later

Rate news page design:

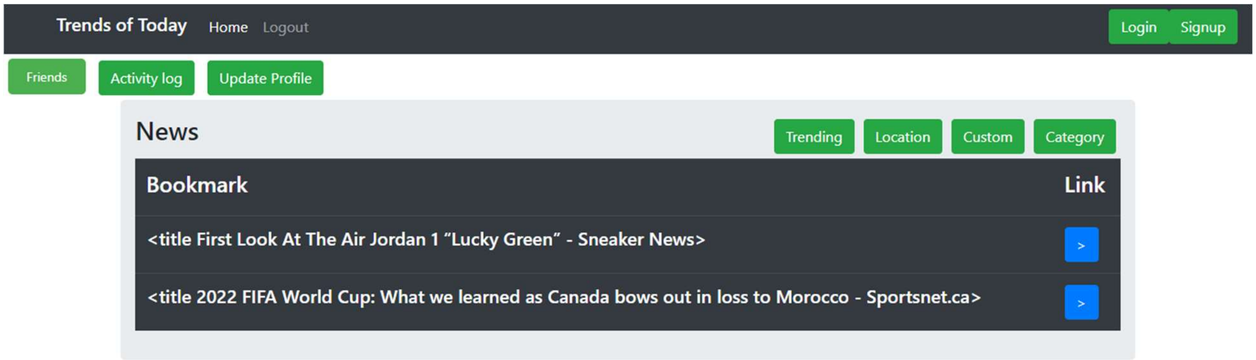

Rate News

Rating Scale

○ 5 stars ○ 4 stars ○ 3 stars ○ 2 stars ○ 1 star

Poster	Title	Link	Rate News	Read later
	G7, Australia, EU… Jazeera English	>	Rate	Later
	US and France pledge Ukraine support amid Russia's 'brutal' war - Al Jazeera English	>	Rate	Later
	Prince William meets US President Joe Biden ahead of climate prize - BBC	>	Rate	Later

Bookmark news to read later page:



Bookmark	Link
<title First Look At The Air Jordan 1 “Lucky Green” - Sneaker News>	>
<title 2022 FIFA World Cup: What we learned as Canada bows out in loss to Morocco - Sportsnet.ca>	>

3.2. NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements in order:

1. **Availability**

   The system must be online/available for use all the time with very small downtime for maintenance, updates etc. This is because it is designed to host multiple core functionalities simultaneously and any disruption can cause a major setback. Users can access the web-basedsystem at any time to view the trending news, thus this non-functional requirement is paramount

2. **Scalability**

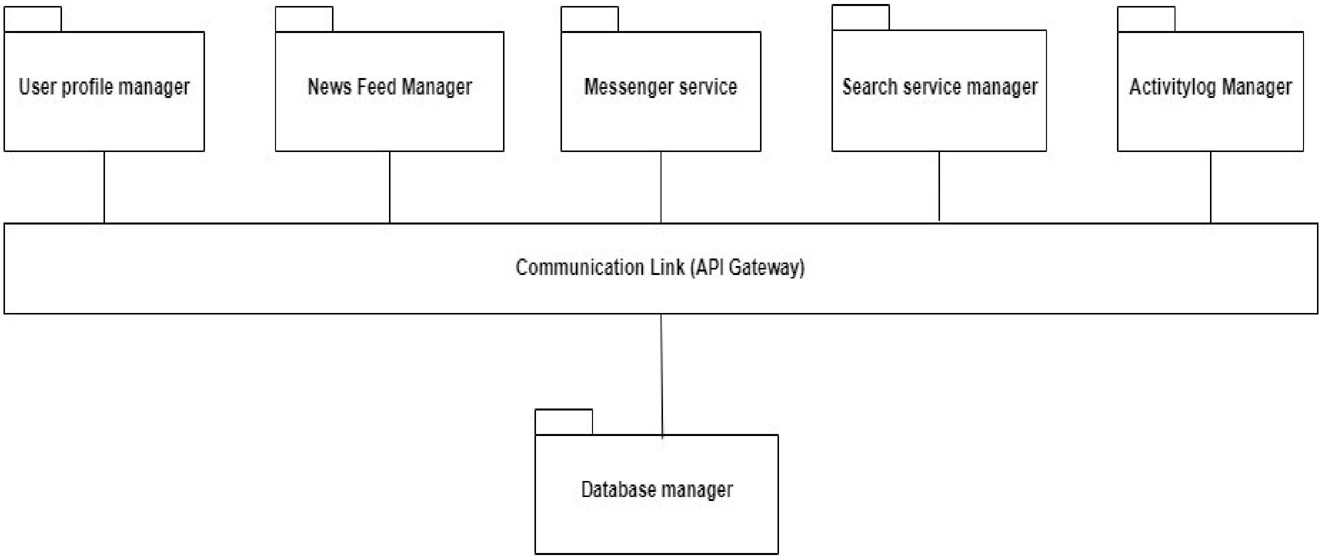
   At present the product is being developed for a smaller audience, however the goal is to expandfor a larger crowd.

3. **Performance**

   The response time and other performance metrics are also relevant. This is since it caters to multiple event sources. The web-application need to perform at a high level with as little delayas realistically possible

4. Architecture

4a. Functional model with description of each component's description is given below



Component ID	**001**
Component Name	User profile Manager
Component description	1. The user can register 2. The user can login 3. The user can have access to the trending news of the world

Component ID	**002**
Component Name	Newsfeed Manager
Component description	The news feed manager helps the user to customize his/her newsfeed based on his/her interests like technology, trending, location etc.

Component ID	**003**
Component Name	Messenger Service
Component description	With the help of messenger service, the user can communicate with his/her friends, share news with friends.

Component ID	**004**
Component Name	Messenger Service
Componnet description	With the help of messenger service, the user can communicate with his/her friends, share news with friends.

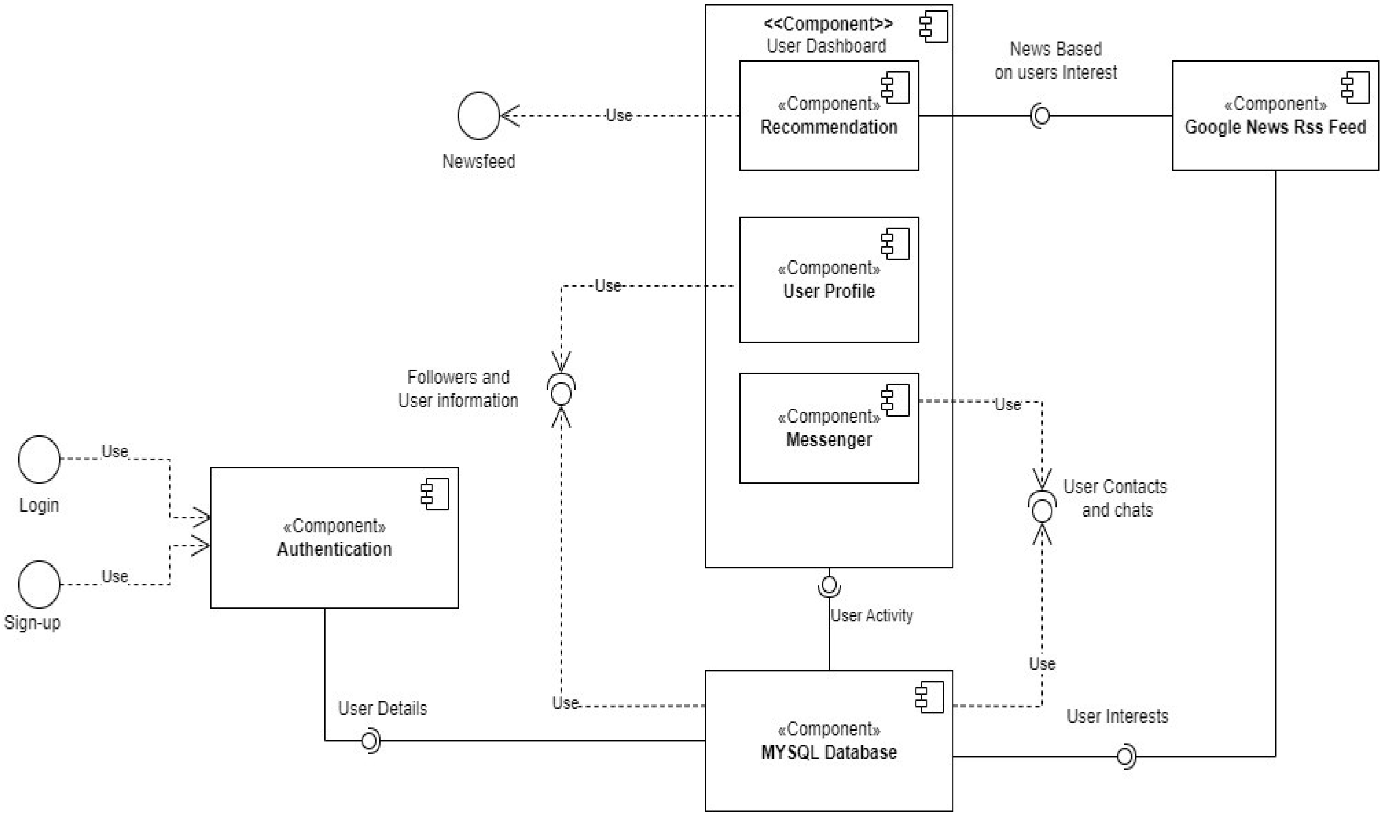
Component ID	**005**
Component Name	Search Service
Componnet description	The search service manager helps the user to search the news by category, location, number of searches etc.

Component ID	**006**
Component Name	Activity log manager
Componnet description	The activity log manager lets the user track his/her activity

	in the website like which news has been liked by the user at what time. This functionality basically provides activity of the user along with timestamp.

Component ID	**007**
Component Name	Database Manager
Component description	The data feed manager basically stores the user information, user activity, user preferences and with the information in the data base we help generate the customize newsfeed feature

4.1. COMPONENT DIAGRAM

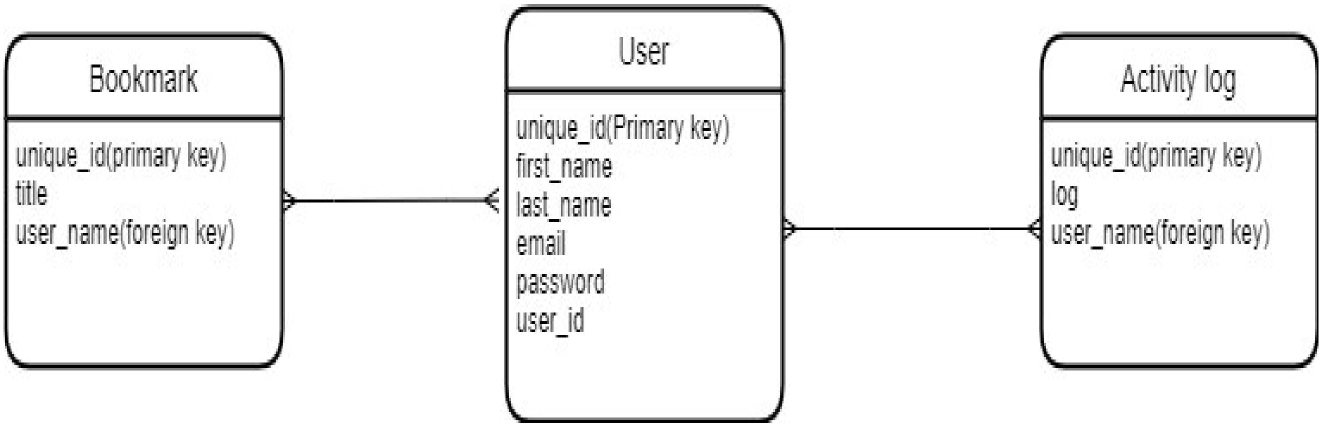


The description for each component is as follows:

- **Authentication**: For the authentication component we have used google O-Authentication protocol.In more technical terms, OAuth is an open standard for secure access delegation, which means it is aservice that allows web giants like Google or Microsoft to permit its users to share their own selected pieces of information with third-party websites or applications, while protecting the confidential info of users at the same time.
- **User dashboard**: This component contains user profile, messenger and recommendation components.
  a. **Recommendations:** Provides recommendations to the customer regarding the newsinterest that they would like to see on their newsfeed
  b. **User Profile**: Provides the user to update his profile information as well allows the followers to view the user's interests and followers
  c. **Messenger**: Allows the user to communicate with fellow users and share the news
- **Google News API**: Google News API is a popular news aggregating service that providesup-to-date news information, collected from numerous sources around the world. It returns searchresults for current and historic news articles. It provides all the data in JSON format.

4.3. Data Model:

Below represents the data model for the data base and the implemented data base:



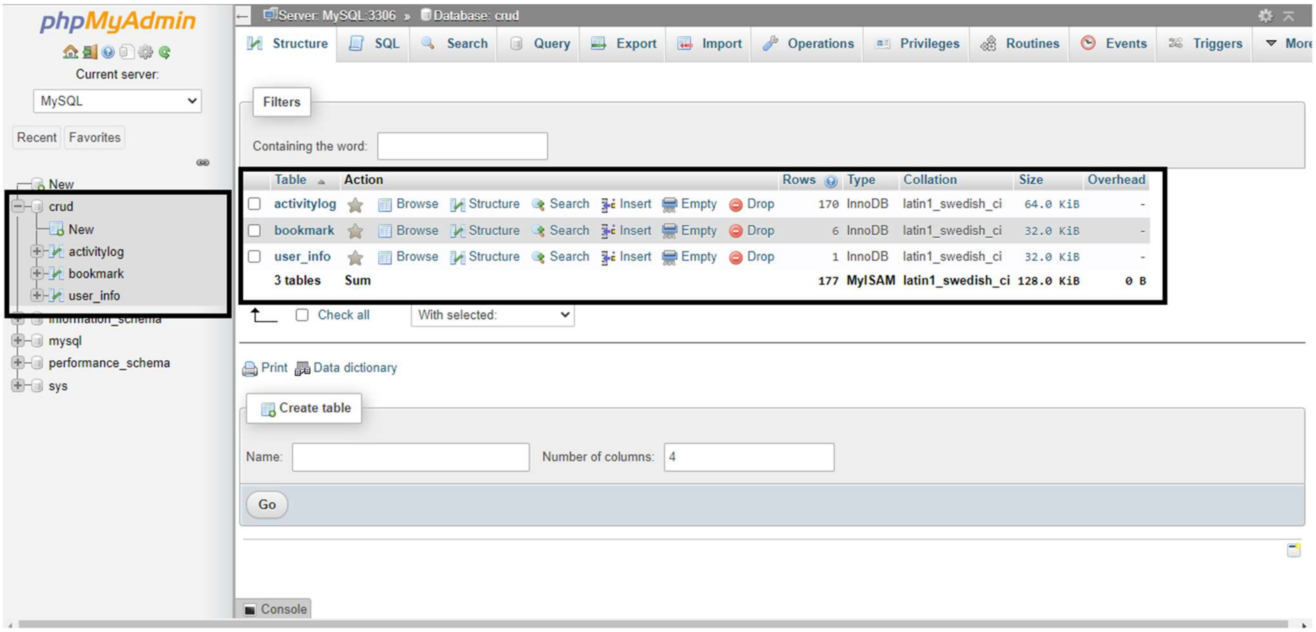
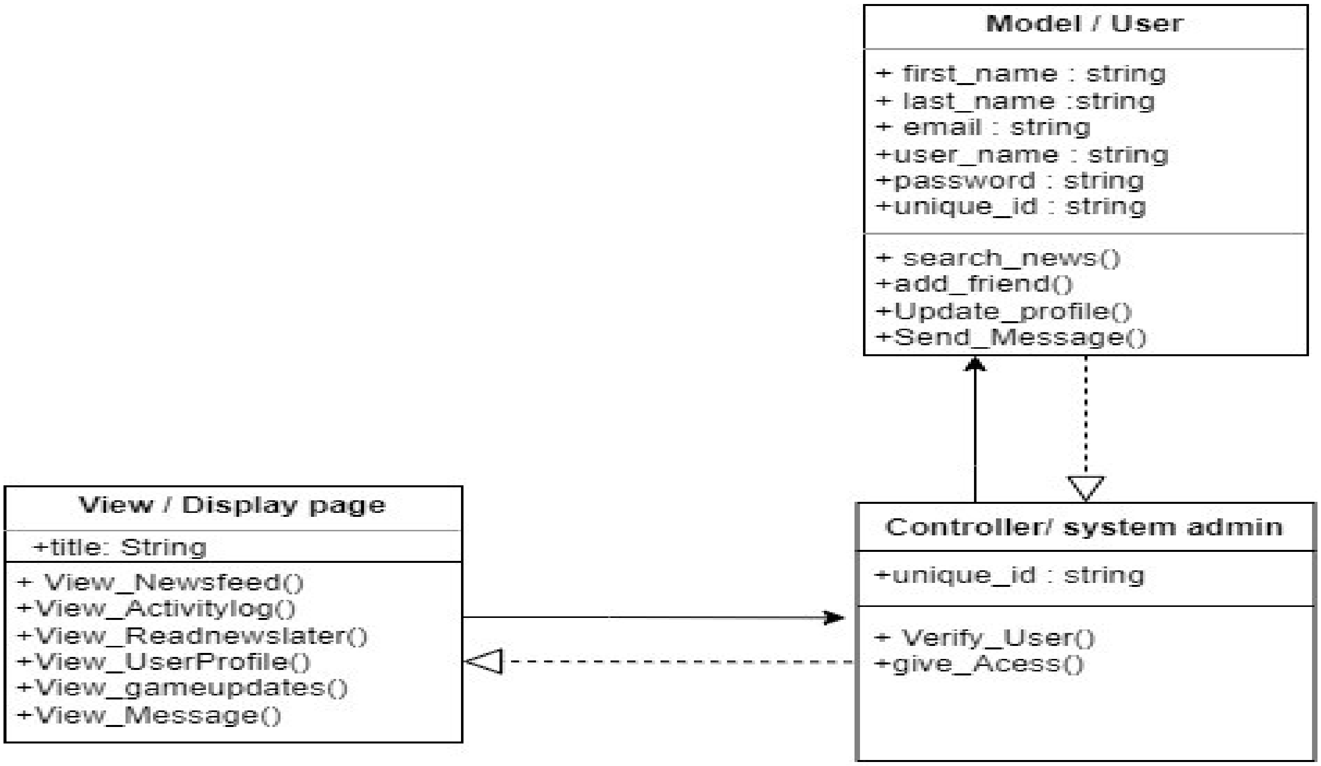


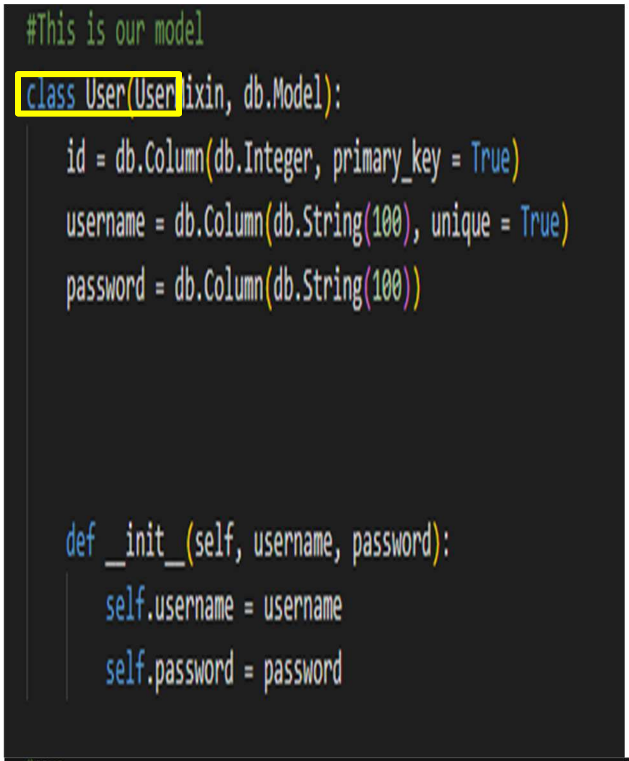
	Table		Action						Rows	Type	Collation	Size	Overhead
□	activitylog	★	Browse	Structure	Search	Insert	Empty	Drop	170	InnoDB	latin1_swedish_ci	64.0 KiB	-
□	bookmark	★	Browse	Structure	Search	Insert	Empty	Drop	6	InnoDB	latin1_swedish_ci	32.0 KiB	-
□	user_info	★	Browse	Structure	Search	Insert	Empty	Drop	1	InnoDB	latin1_swedish_ci	32.0 KiB	-
	3 tables		Sum						177	MyISAM	latin1_swedish_ci	128.0 KiB	0 B

5. Design

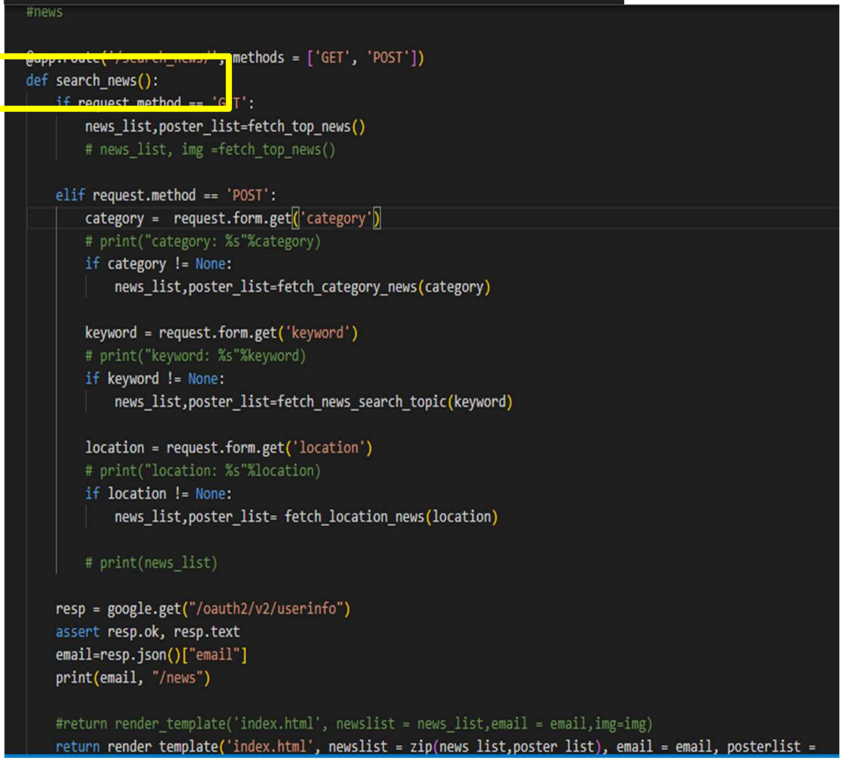
5.1. Class Diagram

The class diagram of our system along with the snippets showing that our code reflects the class diagram has been provided below

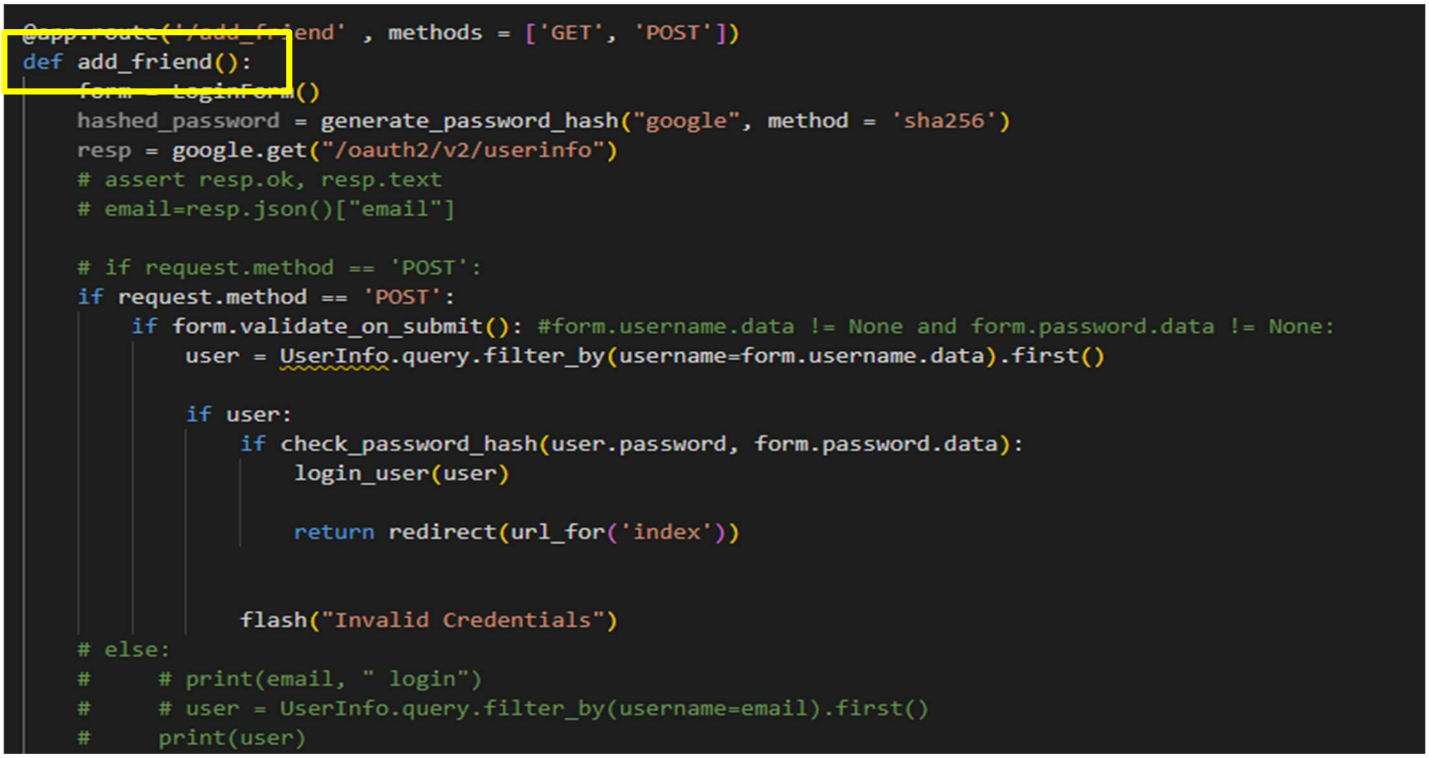




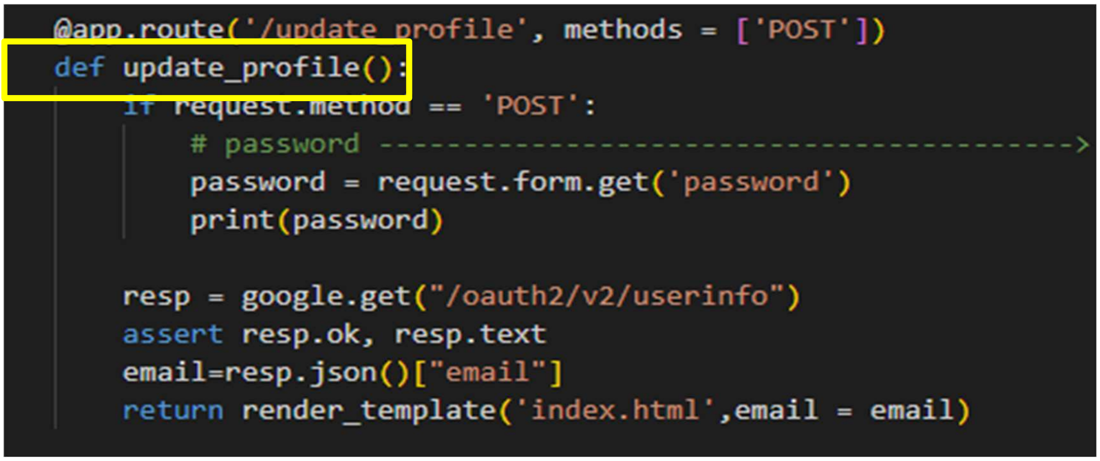
```
#This is our model
class User(UserMixin, db.Model):
    id = db.Column(db.Integer, primary_key = True)
    username = db.Column(db.String(100), unique = True)
    password = db.Column(db.String(100))




    def __init__(self, username, password):
        self.username = username
        self.password = password
```



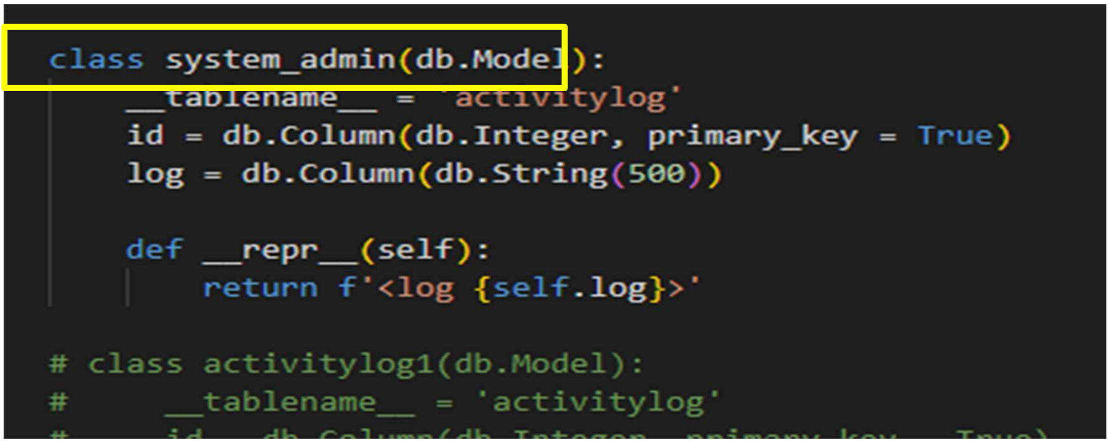
```
#news

@app.route('/search_news/', methods = ['GET', 'POST'])
def search_news():
    if request.method == 'GET':
        news_list,poster_list=fetch_top_news()
        # news_list, img =fetch_top_news()

    elif request.method == 'POST':
        category =  request.form.get('category')
        # print("category: %s"%category)
        if category != None:
            news_list,poster_list=fetch_category_news(category)

        keyword = request.form.get('keyword')
        # print("keyword: %s"%keyword)
        if keyword != None:
            news_list,poster_list=fetch_news_search_topic(keyword)

        location = request.form.get('location')
        # print("location: %s"%location)
        if location != None:
            news_list,poster_list= fetch_location_news(location)

        # print(news_list)

    resp = google.get("/oauth2/v2/userinfo")
    assert resp.ok, resp.text
    email=resp.json()["email"]
    print(email, "/news")

    #return render_template('index.html', newslist = news_list,email = email,img=img)
    return render_template('index.html', newslist = zip(news_list,poster_list), email = email, posterlist =
```



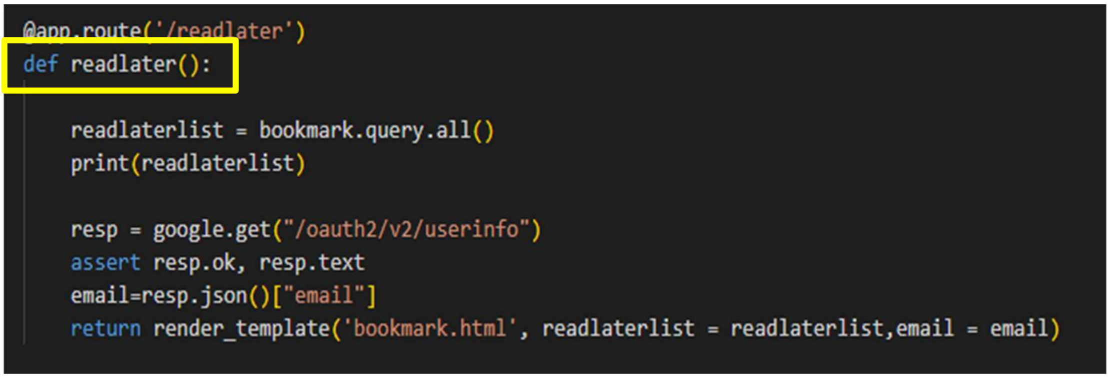
```
@app.route('/add_friend' , methods = ['GET', 'POST'])
def add_friend():
    form = LoginForm()
    hashed_password = generate_password_hash("google", method = 'sha256')
    resp = google.get("/oauth2/v2/userinfo")
    # assert resp.ok, resp.text
    # email=resp.json()["email"]

    # if request.method == 'POST':
    if request.method == 'POST':
        if form.validate_on_submit(): #form.username.data != None and form.password.data != None:
            user = UserInfo.query.filter_by(username=form.username.data).first()

            if user:
                if check_password_hash(user.password, form.password.data):
                    login_user(user)

                    return redirect(url_for('index'))


                flash("Invalid Credentials")
    # else:
    #     # print(email, " login")
    #     # user = UserInfo.query.filter_by(username=email).first()
    #     print(user)
```



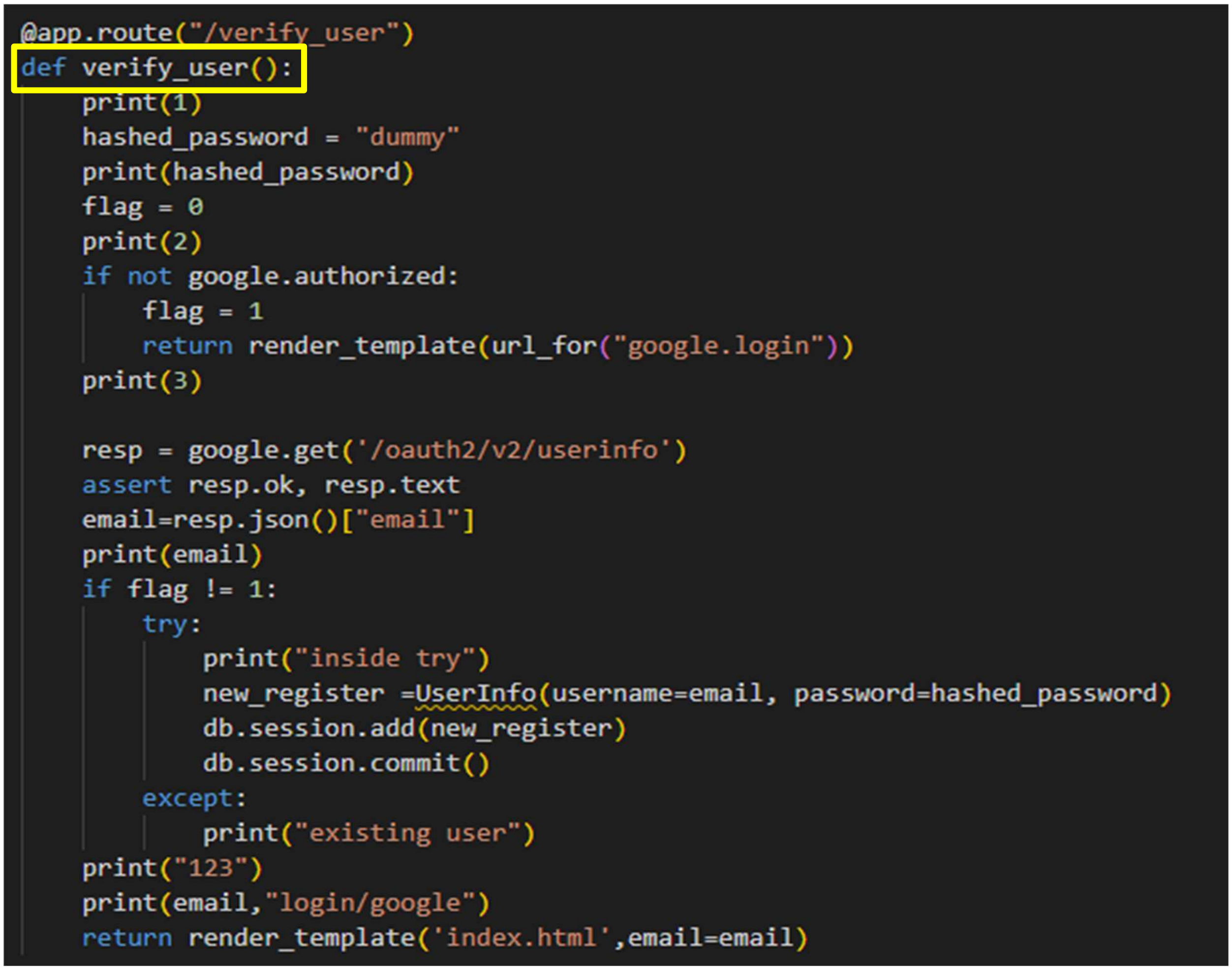
```
@app.route('/update_profile', methods = ['POST'])
def update_profile():
    if request.method == 'POST':
        # password ---------------------------------------->
        password = request.form.get('password')
        print(password)

    resp = google.get("/oauth2/v2/userinfo")
    assert resp.ok, resp.text
    email=resp.json()["email"]
    return render_template('index.html',email = email)
```



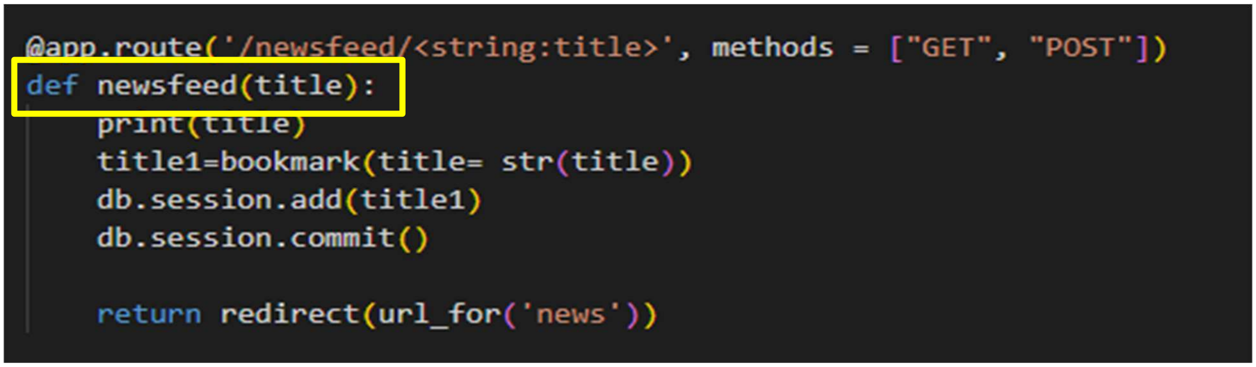
```
class system_admin(db.Model):
    __tablename__ = 'activitylog'
    id = db.Column(db.Integer, primary_key = True)
    log = db.Column(db.String(500))

    def __repr__(self):
        return f'<log {self.log}>'

# class activitylog1(db.Model):
#     __tablename__ = 'activitylog'
#     id = db.Column(db.Integer, primary_key = True)
```



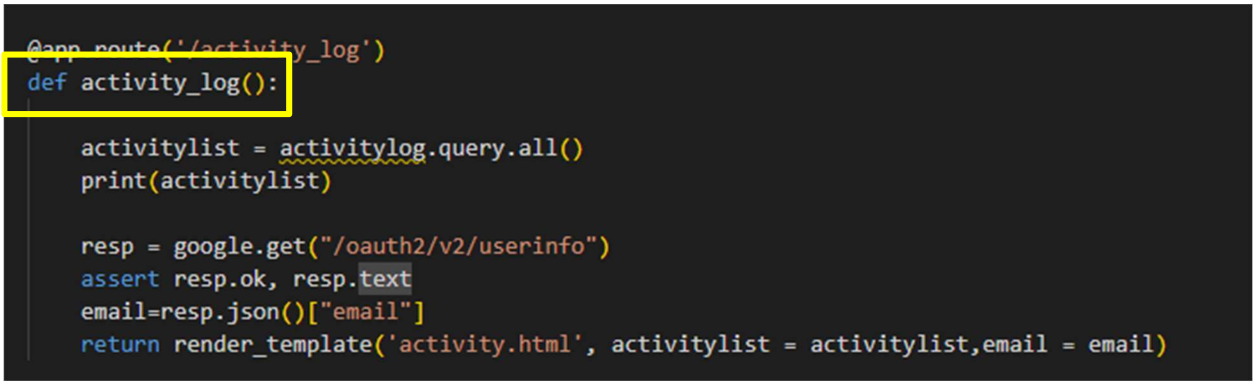
```python
@app.route('/readlater')
def readlater():

    readlaterlist = bookmark.query.all()
    print(readlaterlist)

    resp = google.get("/oauth2/v2/userinfo")
    assert resp.ok, resp.text
    email=resp.json()["email"]
    return render_template('bookmark.html', readlaterlist = readlaterlist,email = email)
```



```python
@app.route("/verify_user")
def verify_user():
    print(1)
    hashed_password = "dummy"
    print(hashed_password)
    flag = 0
    print(2)
    if not google.authorized:
        flag = 1
        return render_template(url_for("google.login"))
    print(3)

    resp = google.get('/oauth2/v2/userinfo')
    assert resp.ok, resp.text
    email=resp.json()["email"]
    print(email)
    if flag != 1:
        try:
            print("inside try")
            new_register =UserInfo(username=email, password=hashed_password)
            db.session.add(new_register)
            db.session.commit()
        except:
            print("existing user")
    print("123")
    print(email,"login/google")
    return render_template('index.html',email=email)
```



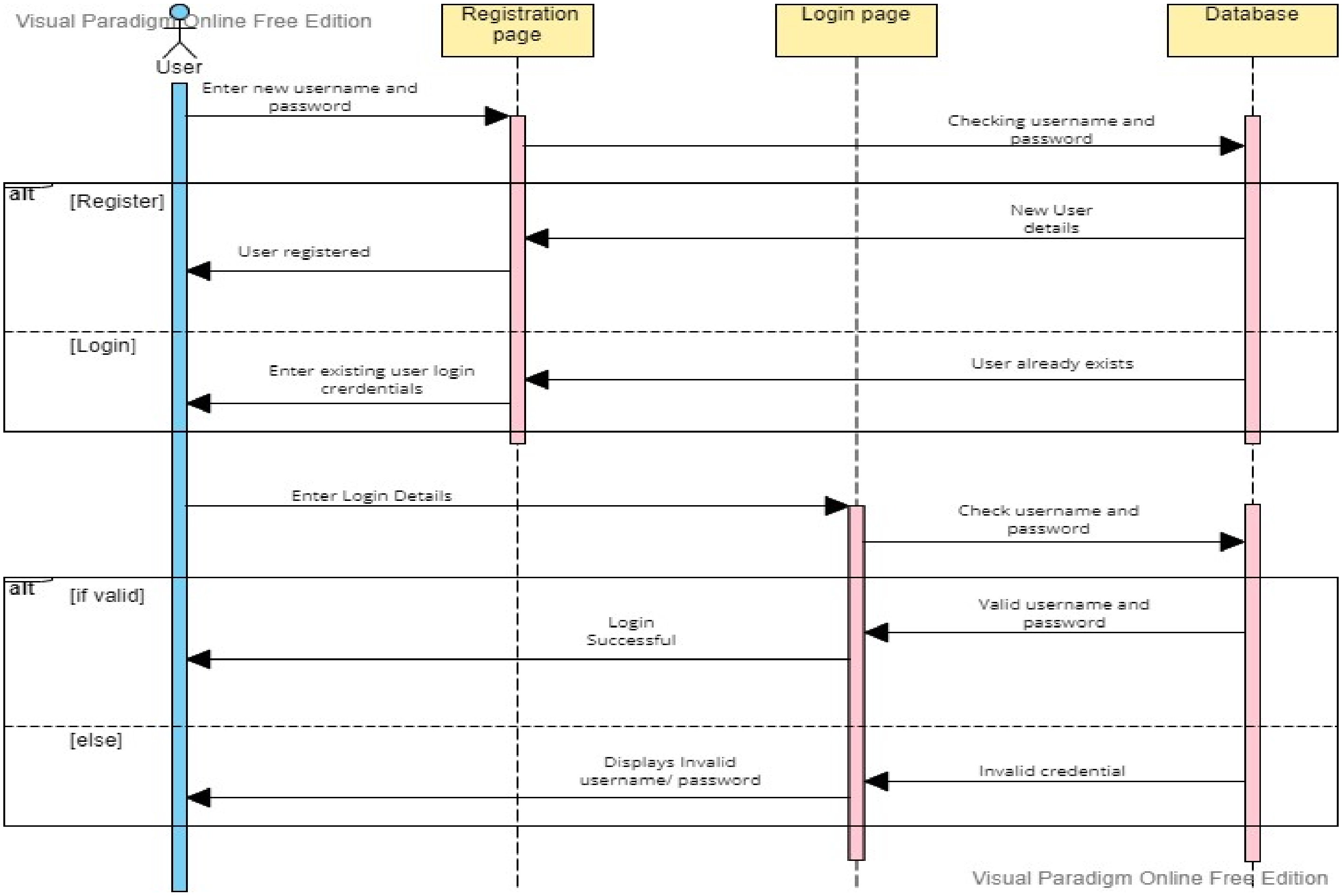
```python
@app.route('/newsfeed/<string:title>', methods = ["GET", "POST"])
def newsfeed(title):
    print(title)
    title1=bookmark(title= str(title))
    db.session.add(title1)
    db.session.commit()

    return redirect(url_for('news'))
```



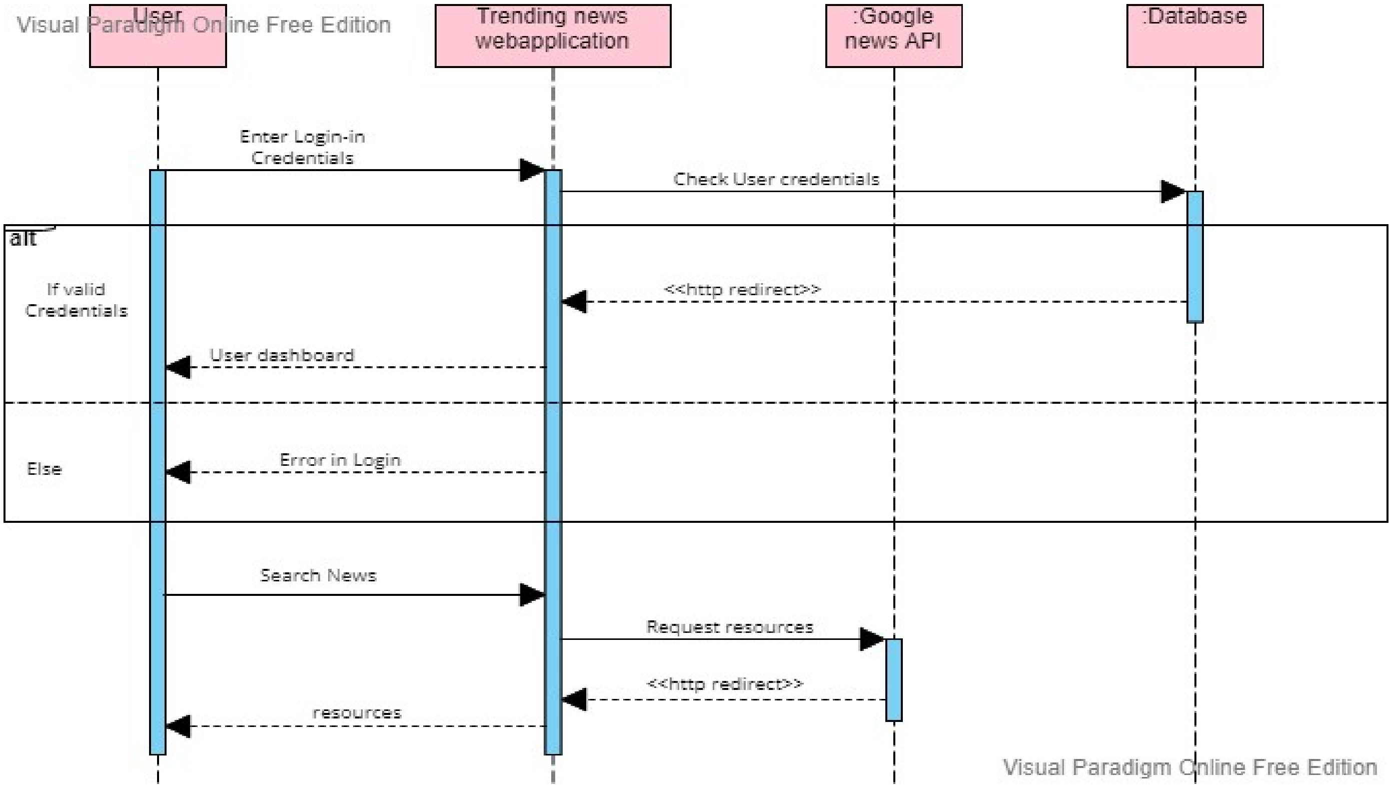
```
@app.route('/activity_log')
def activity_log():

    activitylist = activitylog.query.all()
    print(activitylist)

    resp = google.get("/oauth2/v2/userinfo")
    assert resp.ok, resp.text
    email=resp.json()["email"]
    return render_template('activity.html', activitylist = activitylist,email = email)
```

5.2. SEQUENCE DIAGRAM

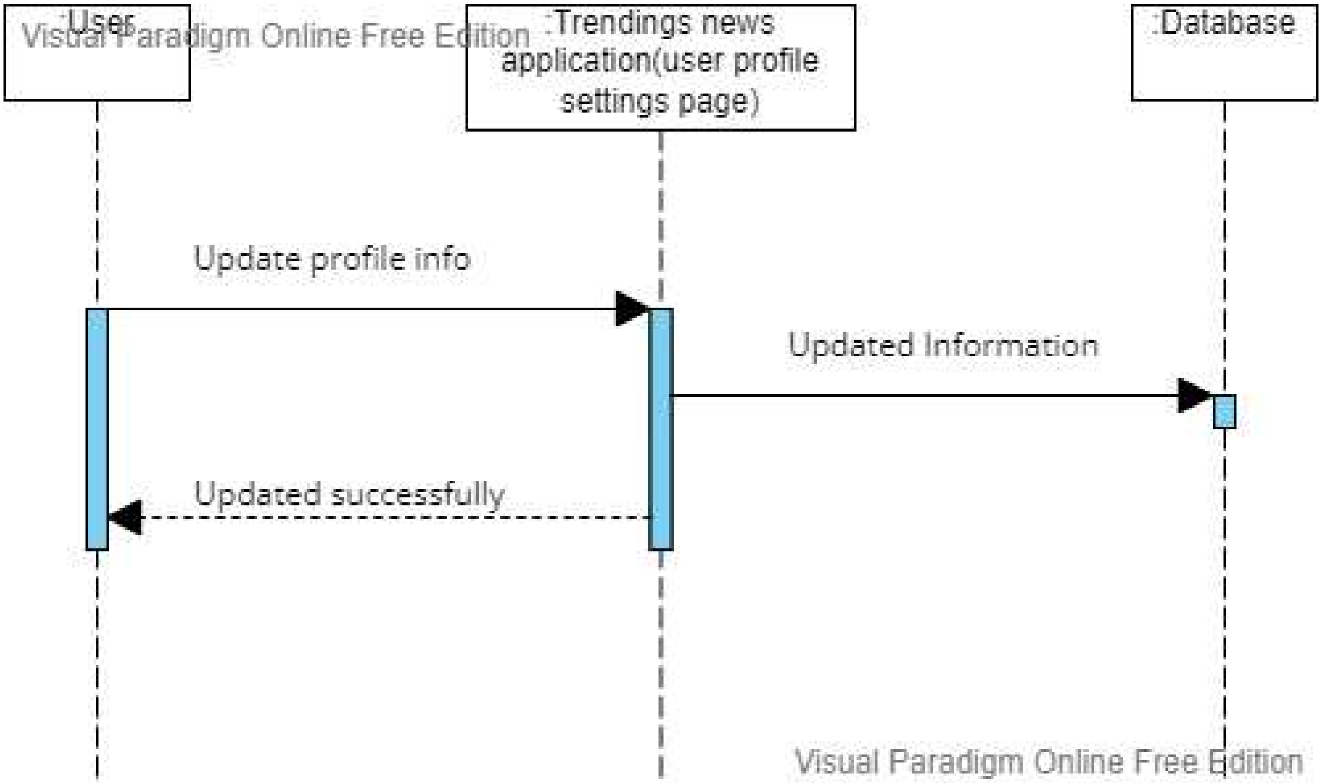
5.a. sequence diagram for login and registration page


Visual Paradigm Online Free Edition
User
Registration page
Login page
Database
Enter new username and password
Checking username and password
alt
[Register]
New User details
User registered
[Login]
User already exists
Enter existing user login crerdentials
Enter Login Details
Check username and password
alt
[if valid]
Valid username and password
Login Successful
[else]
Displays Invalid username/ password
Invalid credential
Visual Paradigm Online Free Edition

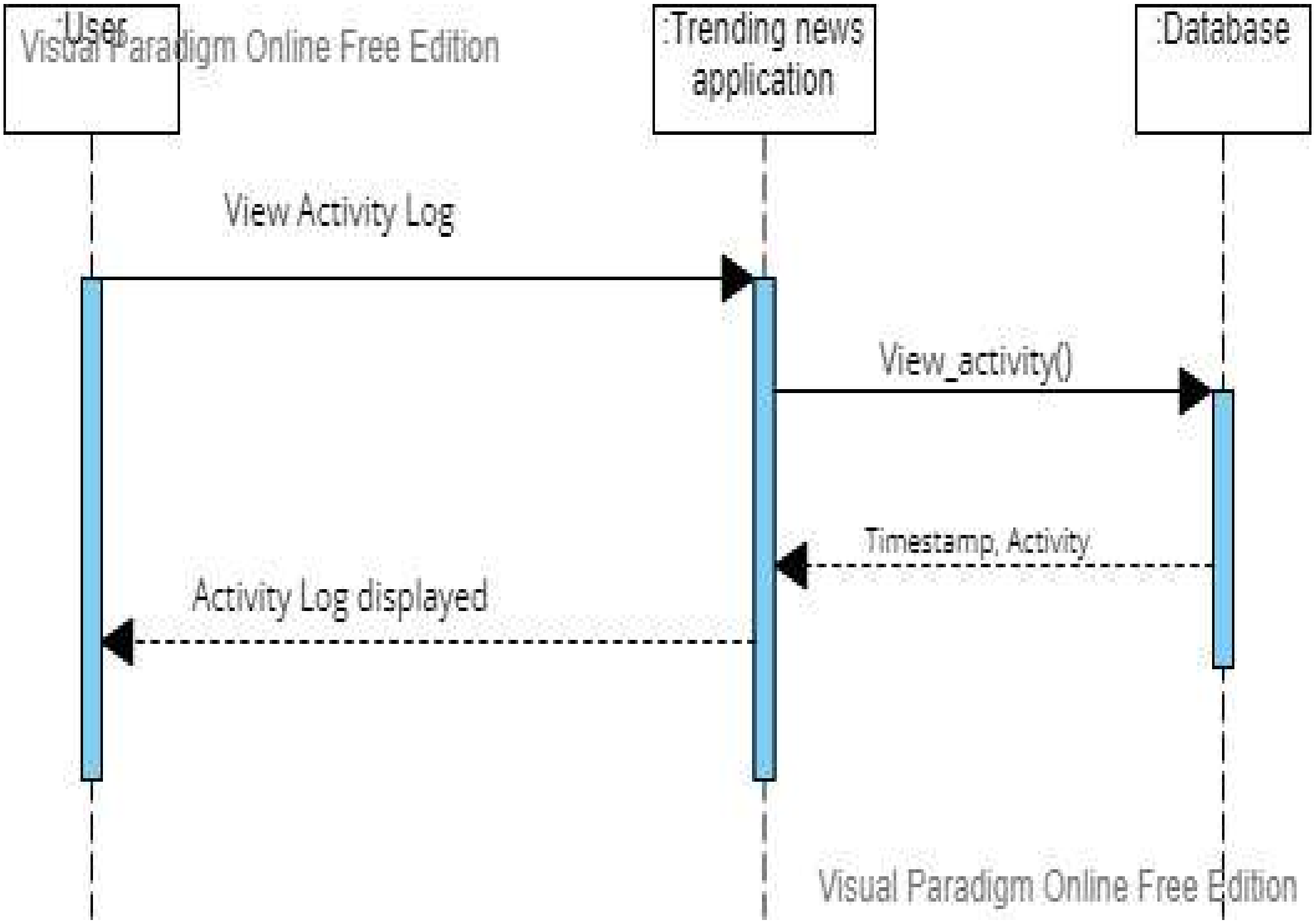
5.b Sequence diagram for searching news



5.c. Sequence diagram for updating profile



5.d. Sequence diagram for viewing activity log



6. Link to our GitHub Repository is:

If you have any difficulty in accessing our repository, please do let us know we will sort it out as soon as possible

https://github.com/RagulNivash/RagulNivash.github.io

6.1. Software Used:

- IDE: Visual Studio
- Front end: HTML/CSS/JavaScript
- Backend: Python (Flask) Framework
- MySQL
- Management Tool: Jira and Confluence
- Communication Tools: Microsoft teams, Google meet and WhatsApp.

PROCESS REPORT

Notes:

1. Sprint meeting report:
https://gurleenkaur.atlassian.net/wiki/spaces/REDRAVEN/pages/10780673/Meeting+Report+Sprint+3

2. Meeting notes: daily scrum :
https://gurleenkaur.atlassian.net/wiki/spaces/REDRAVEN/pages/12419108/Daily+Scrum+Meeting+Sprint+3

3. Retrospective:
https://gurleenkaur.atlassian.net/wiki/spaces/REDRAVEN/pages/11632654/Retrospective+Meeting+Sprint+3

4. Sprint Review meeting:
https://gurleenkaur.atlassian.net/wiki/spaces/REDRAVEN/pages/11567105/Review+Meeting+Sprint+3

5. Link to the code repository: https://github.com/RagulNivash/RagulNivash.github.io

6. Link to the confluence page: https://gurleenkaur.atlassian.net/wiki/spaces/REDRAVEN/overview

7. Link to Jira page: https://gurleenkaur.atlassian.net/jira/software/projects/SE/boards/1

8. Link to application demo video: https://drive.google.com/file/d/1JvzIIkbprwvYOM2yzDtSdUb5Si2vpjcG/view

9. Link to the website: https://trendsoftoday-b792a.web.app/

1. SPRINT 3 REVIEW MEETING REPORT:

A. SPRINT GOAL

To implement front-end and backend development of three epics namely 1. Improving user experience 2. User Involvement and 3. User Interaction and Profile Management.

- SE-14 - A user wants to rate trending news on a scale of 1 to 5. Done
- SE-18 - A user wants to be reminded of matches which are scheduled and updated with live scores of ongoing games. Done

- SE-24 - User wants to visit profile of his friends to see his friend's activity. Done
- SE-25 - User wants to share news and send messages to their friends In Progress
- SE-48 - A user wants to share a news to a follower or friend In Progress
- SE-67 - Activity Log Done
- SE-66 - Bookmark Done
- SE-68 - Displaying Image Done

B. SPRINT REVIEW

Sprint review meeting of one hour was conducted on 23rd November 2022 to analyse the progress and backlogs of the sprint. Plan to complete the tasks were re-analysed along with creating a testing plan and document was decided.

- **Challenges Faced:**
  - Working on new technologies with no prior experience is challenging, it is longer time to understand and implement things than expected.
  - Need to prioritize few features over the other in order to meet actual timeline.
  - One of our team mates was sick and hence, might take longer to implement than expected timeline.
- **Improvements:**
  - With two sprints done, developers are finding coding relatively easier and they are understanding things in a better way.
  - Ownership could be seen in this sprint.
  - UI/UX interface is more user friendly and easy to use as compared to the previous sprints.
  - As of now, rating feature and profile feature with updating user profile features are complete.
- **Expected Time per Person V/s actual time:**

Member	Week 1 expected(hrs)	Week 1 actual (hrs)	Week 2 expected(hrs)	Week 2 actual (hrs)
Gurleen (Front end Developer and Product Owner)	15	17	11	12
Dona (Front end Developer)	10	13	16	16
Arshdeep (Backend Developer)	10	9	14	16
Shiv (Backend Developer)	14	7	12	17
Ragul (Backend Developer and Scrum master)	20	18	13	19

- **Risk Analyzation:**
  - Detailed risk analyzation is done in the meeting report, one of our team members got sick in this sprint.
  - Other team members took over and shared the work load and most of the things are on track with a slight delay that can be covered.
  - A lot of time is being consumed in understanding techniques and figuring out which might be best for our website. As of now, nothing much can be done about this but a guidance from POD, seniors and batch mates is being taken to work in expected timeline.
- **Communication:**
  - Methods used: 1. Microsoft teams (for scheduled meetings) 2. WhatsApp group (for reminding, updating and acknowledging) 3. Phone call (for urgent matters)
  - Improvements: We initially started with having daily scrums every alternate day but later decided on having them every day to streamline the process and for improving progress.
  - Key aspect: Everyone was available mostly every time whenever someone needed help, texts were responded on time and every one attended every meeting (if someone was unable to due to some reason, they called the scrum master to inform them and later to know what happened in the meeting)
  - There was no cultural barrier in our group as we share a common background and hence, blending in and creating a working atmosphere was relatively very easy.
- **Team Work:**

- Every member was assigned tasks based on their expertise but if anyone was having hard time doing a task, he/she were helped by other team members.
  - Helping each other and taking ownership of the work was remarkable during this sprint.
  - As beginners, at some point some of us used to lose motivation due to the difficulty but other team members inspired them and helped them with their work and keeping them motivated.
  - In our team, three people are working on backend and two on frontend. The report is shared contribution of everyone.
- **What could be improved:**
  - Skill set: We need in depth knowledge of all the software. We are now familiar with all the technologies now, so it'll be easy to experiment and explore. This would help us in upcoming projects.
- **What could be repeated:**
  - This team: as it was easy to manage, compatible with one another and always there to help each other.
  - Communication methods: Methods used for communication were sufficient and can be used again.
- **SUGGESTIONS FOR FUTURE STUDENTS:**
  - At least one member in the group should have a full stack developer experience so that he/she can guide everyone in the group.
  - Take ownership of your work.
  - Help your team mates, it's a group project not an individual project. If someone is facing a problem then tackle it as a team.
  - Plan everything, regular update everything and change your plans if they don't seem to work.

Go with the flow, even

TESTING REPORT

Link to testing report:

https://gurleenkaur.atlassian.net/wiki/spaces/REDRAVEN/pages/13205505/Testing+Document

Created test cases of all functionalities.
Manual testing of UI and functionalities were performed.
Total functional and UI test case count.
Total test cases= 54, Pass=48, Fail=6

Login/Signup page test cases.

S.No	Test Case	Pre requisite	Execution Steps	Expected Result	Test Pass/Fail
1	Login with blank email and password.	Server should be up and running.	1. Keep email and password field blank. 2. Click on login.	Login unsuccessful.	Test Pass
2	Login with blank email and filled password.	Server should be up and running.	1. Keep email blank and fill password field. 2. Click on login.	Login unsuccessful.	Test Pass
3	Login with filled email and blank password.	Server should be up and running.	1. Keep password blank and fill login field. 2. Click on login.	Login unsuccessful.	Test Pass
4	Login with incorrect email and incorrect	Server should be up and	1. Fill incorrect email and incorrect	Login unsuccessful.	Test Pass

	password.	running.	password. 2. Click on login.		
5	Login with correct email and incorrect password.	Server should be up and running.	1. Fill correct email and incorrect password. 2. Click on login.	Login unsuccessful.	Test Pass
6	Login with incorrect email and correct password.	Server should be up and running.	1. Fill incorrect email and correct password. 2. Click on login.	Login unsuccessful.	Test Pass
7	Login with correct email and correct password.	Server should be up and running.	1. Fill correct email and correct password. 2. Click on login.	Login successful.	Test Pass
8	Verify if page gets redirected after successful login.	Server should be up and running.	1. Fill correct email and correct password. 2. Click on login.	Page redirects to dashboard.	Test Pass
9	Verify if page does not gets redirected after unsuccessful login.	Server should be up and running.	1. Fill incorrect credentials. 2. Click on login.	Page does not redirect to dashboard.	Test Pass
10	Verify error message on unsuccessful login.	Server should be up and running.	1. Fill incorrect credentials. 2. Click on login.	Error message displayed on wrong credentials.	Test Pass
11	Verify password field is hidden.	Server should be up and running.	1. Fill password in password field.	Password field is hidden by default.	Test Pass
12	Verify password displays on clicking on show password.	Server should be up and running.	1. Fill password in password field. 2. Click on show password.	Password displays.	Test Pass
13	Verify remember me button checks.	Server should be up and running.	1. Fill correct email and correct password. 2. Click on remember me. 3. Click on login.	Button checks.	Test Pass
14	Verify remember me button unchecks.	Server should be up and running.	1. Fill correct email and correct password. 2. Click on remember me. 3. Uncheck the remember me button. 4. Click on login.	Button unchecks.	Test Pass

15	Verify remember me account shows up during next login.	Server should be up and running.	1. Fill correct email and correct password. 2. Click on remember me. 3. Click on login. 4. Logout. 5. Enter with same credentials.	User able to login with these credentials.	Test Pass

Dashboard User Interface test cases:

S.No	Test Case	Pre-Requisite	Execution Steps	Expected Result	Test Pass/Fail
1	Dashboard displaying News	Server should be up and running.	1. Login to dashboard.	News displayed.	Test Pass
2	Dashboard displaying pictures of News	Server should be up and running.	1. Login to dashboard.	Image displaying.	Test Pass
3	Dashboard displaying home button.	Server should be up and running.	1. Login to dashboard.	Home button displaying.	Test Pass
4	Dashboard displaying Logout button.	Server should be up and running.	1. Login to dashboard.	Logout button displaying.	Test Pass
5	Dashboard displaying Profile button.	Server should be up and running.	1. Login to dashboard.	Update profile button displaying.	Test Pass
6	Dashboard displaying Trending Button.	Server should be up and running.	1. Login to dashboard.	Trending button displaying.	Test Pass
7	Dashboard displaying Location Button.	Server should be up and running.	1. Login to dashboard.	Location button displaying.	Test Pass
8	Dashboard displaying Custom button.	Server should be up and running.	1. Login to dashboard.	Custom button displaying.	Test Pass
9	Dashboard displaying Category button.	Server should be up and running.	1. Login to dashboard.	Category button displaying.	Test Pass
10	Dashboard displaying Link of News.	Server should be up and running.	1. Login to dashboard.	Link displaying.	Test Pass
11	Dashboard displaying Rate news button.	Server should be up and running.	1. Login to dashboard.	Rate news button displaying.	Test Pass
12	Dashboard displaying Later button.	Server should be up and running.	1. Login to dashboard.	Later button displaying.	Test Pass
13	Dashboard displaying Bookmark news button.	Server should be up and running.	1. Login to dashboard.	Bookmark news button displaying.	Test Pass
14	Dashboard displaying Logs button.	Server should be up and running.	1. Login to dashboard.	Application logs button displaying.	Test Pass
15	Dashboard displaying friend’s button.	Server should be up and running.	1. Login to dashboard.	Friends button displaying.	Test Pass

Dashboard Functional Test cases.

S.No	Test Case	Pre-Requisite	Execution Steps	Expected Result	Test Pass/Fail
1	To verify trending news is displayed.	Server should be up and running.	1. Login to dashboard.	Trending news is displayed.	Test Pass

2	Clicking on link page redirects to the news.	Server should be up and running.	1. Login to dashboard. 2. Click on link of a news.	Detailed news opens.	Test Pass
3	News displayed after clicking on link.	Server should be up and running.	1. Login to dashboard. 2. Click on link of a news.	Detailed news opens.	Test Pass
4	Back button takes back to dashboard.	Server should be up and running.	1. Login to dashboard. 2. Click on link of a news. 3. Press back button	Dashboard displays after back button.	Test Pass
5	Clicking on location displays different regions.	Server should be up and running.	1. Login to dashboard. 2. Click on location. 3. Select a location.	Different regions displayed.	Test Pass
6	Selecting location redirects to news of the location.	Server should be up and running.	1. Login to dashboard. 2. Click on location. 3. Select a location.	News of location displays.	Test Pass
7	Clicking on custom allows to search news by keyword.	Server should be up and running.	1. Login to dashboard. 2. Click on custom. 3. Type a keyword.	Able to search news with keyword.	Test Pass
8	Selected custom word redirects to the related news.	Server should be up and running.	1. Login to dashboard. 2. Click on custom. 3. Type a keyword.	Able to search news with keyword.	Test Pass
9	Clicking on category displays categories.	Server should be up and running.	1. Login to dashboard. 2. Click on category. 3. Select a category.	Categories being displayed.	Test Pass
10	Selected category displays related news.	Server should be up and running.	1. Login to dashboard. 2. Click on category. 3. Select a category.	News of selected category displays.	Test Pass
11	Logout button logs out of dashboard.	Server should be up and running.	1. Login to dashboard. 2. Click on logout.	Logs out of the page.	Test Pass
12	Update profile displays profile.	Server should be up and running.	1. Login to dashboard. 2. Click on update profile.	Profile displayed with username and password.	Test Pass

13	To verify password is changed from update profile.	Server should be up and running.	1. Login to dashboard. 2. Click on update profile. 3. Change your password.	Password changed.	Test Pass
14	To verify rate news open rating.	Server should be up and running.	1. Login to dashboard. 2. Click on rate news.	Rating opens.	Test Pass
15	To verify rating of rated news is saved.	Server should be up and running.	1. Login to dashboard. 2. Click on rate news. 3. Rate a news.	Rating is saved.	Test Pass
16	To verify clicking on later button saves news.	Server should be up and running.	1. Login to dashboard. 2. Click on read later.	News saved for later.	Test Pass
17	Bookmark tab shows read later news.	Server should be up and running.	1. Login to dashboard. 2. Click on read later. 3. Open Bookmark tab.	Bookmark tab displays saved news.	Test Pass
18	Clicking on friends button displays list of friends.	Server should be up and running.	1. Login to dashboard. 2. Click on friends.	List of friends displayed.	Test Fail
19	Clicking on friends button displays messenger.	Server should be up and running.	1. Login to dashboard. 2. Click on friends. 3. Click on view friends.	Messenger displayed.	Test Fail
20	To verify to be able to send friend request.	Server should be up and running.	1. Login to dashboard. 2. Click on friends. 3. Search a friend. 4. Send friend request.	Friend request sent.	Test Fail
21	To verify to accept a friend request.	Server should be up and running.	1. Login to dashboard. 2. Click on friends. 3. See friend’s request.	Friend request is received.	Test Fail
22	To verify accepted friends is displayed.	Server should be up and running.	1. Login to dashboard. 2. Click on friends. 3. See friend’s request. 4. Accept a friend’s request.	Friends displayed.	Test Fail

23	To verify message is sent to friends.	Server should be up and running.	1. Login to dashboard. 2. Click on friends. 3. See friends list. 4. Send message to a friend.	Message is sent.	Test Fail
24	To verify logs display logs.	Server should be up and running.	1. Login to dashboard. 2. Click on Application Logs.	Logs are displayed.	Test Pass

Bugs Fixed:
1. Not able to login without authentication.
2. Pictures not displaying on news.
3. UI of dashboard is not aligned.
4. Read later not displaying news on bookmark.
5. Category button not displaying all categories.

Bugs Open:
1. Friends feature not working.
2. Not able to send friend request.
3. Messenger not working.
4. Application logs display logs upside down.
5. Login button displaying twice on dashboard.
6. Rate and Read later button not aligned properly.

Conclusion

Overall, we were able to work well as a group. Group members complimented each other’s ability, knowledge, skill and efficiency. We were able to achieve most of what we planned to complete in sprint3.