Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre

A Project Report
on
"Hulaki: A Nepali News Aggregator Powered By Graph Theory and Machine Learning"

[Code No.: COMP307]

(For partial fulfillment of Year III / Semester II in Computer Science)

Submitted by
Mukul Aryal (03)
Anupama Neupane (36)
Yunidh Rawal (48)
Prayash Shakya (51)

Submitted to
Mr. Suman Shrestha
Department of Computer Science and Engineering

January 13, 2025

Bona fide Certificate

This project work on

“Hulaki: A Nepali News Aggregator Powered By Graph Theory and Machine Learning”

is the bona fide work of

“

Mukul Aryal (03),

Anupama Neupane (36),

Yunidh Rawal (48),

Prayash Shakya (51)

”

who carried out the project work under my supervision.

Project Supervisor

Ms. Deni Shahi
Assistant Professor
Department of Computer Science and Engineering

Abstract

The lack of a dedicated news aggregation platform that focuses on delivering localized, credible news while meeting modern digital standards has created a gap in the Nepalese digital media landscape. Existing platforms like Google News and Feedly fail to address the needs of Nepalese audiences due to limited local content and lack of language support, while local apps like Hamro Patro include news as a secondary feature, lacking a focused and user-friendly approach. To address this, we developed Hulaki, a news aggregator app specifically designed for Nepal. The app provides personalized news recommendations by utilizing graph theory and mathematical models, supports both Nepali and English languages in the user interface, and prioritizes verified news sources to reduce misinformation. The system also clusters similar news articles to reduce data duplication and ensure the users do not have to read the same news twice. The mobile application offers a customizable and smooth user experience with multiple news presentation formats to cater to diverse preferences. By filling the gap in Nepal's digital news landscape, this project promotes media literacy and supports local journalism.

Keywords: *News Aggregation, Clustering, Local Journalism, Personalization, Nepali News.*

Contents

1 Introduction **1**
- 1.1 Background . . . 1
- 1.2 Objectives . . . 1
- 1.3 Motivation and Significance . . . 2

2 Related Works **3**
- 2.1 Pre-existing works . . . 3
  - 2.1.1 Google News . . . 3
  - 2.1.2 Feedly . . . 4
  - 2.1.3 Hamro Patro . . . 5

3 Design and Implementation **6**
- 3.1 Design . . . 6
- 3.2 System Requirement Specifications . . . 9
  - 3.2.1 Hardware Specifications . . . 9
  - 3.2.2 Software Requirements . . . 9
- 3.3 Implementation . . . 10
  - 3.3.1 News Scraping . . . 11
  - 3.3.2 News Clustering . . . 12
  - 3.3.3 Graph Recommendation System . . . 14
  - 3.3.4 Mobile Application . . . 16
  - 3.3.5 Firebase Backend . . . 18
  - 3.3.6 Recommendation Server . . . 18

4 Discussion on the achievements **20**
- 4.1 Features . . . 20
  - 4.1.1 Personalized News Recommendations . . . 20
  - 4.1.2 Data Duplication Reduction with Clustering . . . 20
  - 4.1.3 Multilingual Support . . . 21
  - 4.1.4 User-Friendly Interface . . . 21
  - 4.1.5 Verified News Source Prioritization . . . 21

5 Conclusion **22**
- 5.1 Limitations . . . 22
- 5.2 Future Enhancement . . . 22

List of Figures

2.1 Google News 3
2.2 Feedly 4
2.3 Hamro Patro 5

3.1 High Level System Diagram 6
3.2 Use Case Diagram 7
3.3 Entity Relation (ER) Diagram 8
3.4 System Architecture Diagram 10
3.5 Unified Modeling Language(UML) Diagram of Server Computer Diagram 11
3.6 News Clustering Workflow Diagram 12
3.7 Example of the clusters (good cluster) 14
3.8 Example of the clusters (bad cluster) 14
3.9 Visualization of the recommendation graph 15
3.10 Listicle Style 17
3.11 Brainrot Style 17
3.12 Pinterest Style 18

List of Tables

3.1 Number of news according to website 8
3.2 Number of news according to category 8

Chapter 1: Introduction

1.1 Background

A News Aggregator is an application that compiles news from different sources and presents them in one location. We live in an age where most media has become digitized and is available on the internet, including news articles. News publishers worldwide have adapted to our modern digital age where consuming content online has become the standard. In recent years, there has been a rise of news aggregator applications with provide a user-centric service with varying news topics, interactivity, personalized content, and accessibility.

Access to reliable and personalized news is important, and it is the goal of news aggregator apps to provide this service to consumers. While platforms like **Google News** and **Feedly** have established themselves as news aggregators providing content that covers global news, a news aggregator app focused on Nepali news and journalism has yet to establish itself. While the popular app **Hamro Patro** does provide news aggregation features, it's an app that tries to fulfill multiple roles (calendar, foreign exchange rate, horoscope, advertisement etc). Being a "jack of all trades" Nepali app, it does not focus on providing a complete news aggregator experience that is up-to-date with modern UI/UX trends and standards. So as a group, it is our opinion that a platform that fulfills this need for the Nepali audience could be successful.

1.2 Objectives

The objectives of our project include:

- Implement graph theory and mathematical models to understand and adapt to individual user preferences, providing personalized news recommendations.
- Offer multiple UI formats for news presentation, allowing users to customize their news consumption experience.
- Combat misinformation by prioritizing content from verified and credible news sources, thereby promoting trust in the platform.
- Support and promote local journalism by directing traffic to original news sources and highlighting quality content from Nepalese journalists.

1.3 Motivation and Significance

We chose to develop a news aggregator app to fulfill the lack of a modern, centralized and user-centric platform in the current media landscape of Nepal. We believe media literacy and knowledge about current affairs is very important for the general public.Popularity of traditional medium for news like newspapers or news TV channels have declined over the years, with the simultaneous increase in popularity of social media and online platforms. While news websites and apps like **HamroPatro** are already providing news and some aggregation features, our project specifically caters to modern users and provides an alternative to social media platforms by prioritizing credible local news sources. Unlike existing international news aggregators, our application focuses specifically on preserving and supporting local Nepalese journalism, which is struggling in the digital age. Our solution stands out by offering multilingual support in both English and Nepali, ensuring accessibility for all users. Additionally, we're innovating by providing a modern application with customizable UI preferences, fulfilling diverse user needs. This approach not only promotes media literacy but also creates a unique, localized news consumption experience that existing international platforms cannot offer to the Nepalese audience.

Chapter 2: Related Works

2.1 Pre-existing works

2.1.1 Google News

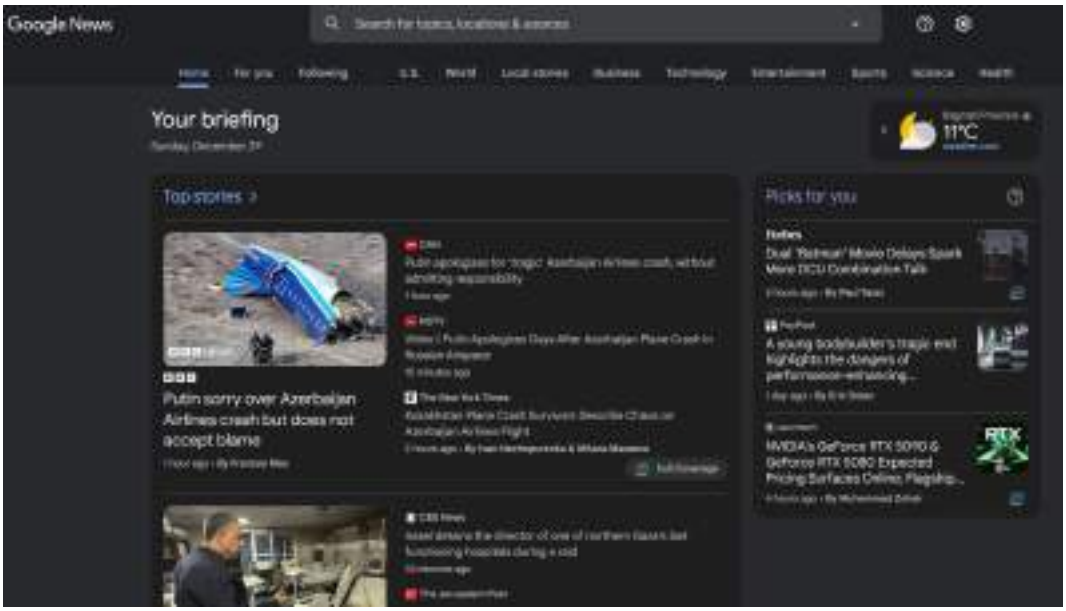


Figure 2.1: Google News

Google News is a news aggregator developed by Google to provide users with a comprehensive and real-time collection of news stories form various sources. It organizes and displays news from around the world, covering a wide range of topics, including politics, technology, entertainment and sports. The platform uses algorithm to present news that is relevant and personalized for each user, based on their preferences and browsing habits. It has a user-friendly interface that allows users to stay updated on the latest news from a variety of perspectives.

Limitation: While Google News is a great app to stay up-to date with current affairs happening globally and even has the option to view news from local sources, it does not have a feature for Nepali language. Due to this limitation, it is not accessible for Nepalese people who wish to read Nepali news.

2.1.2 Feedly

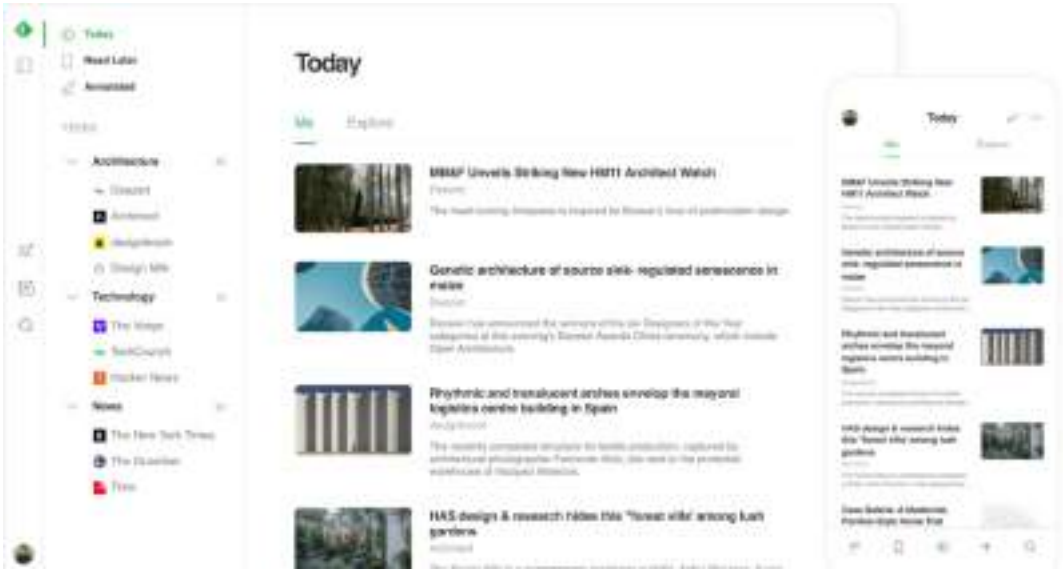


Figure 2.2: Feedly

Feedly is a feed aggregator and news reader that allows users to organize and consume content from their favorite websites, blogs, and news sources in one centralized location. It enables users to curate their own personalized news feeds by subscribing to specific publications, topics, or keywords. The platform offers powerful organization tools, letting users categorize content into custom collections and boards. Users can save articles for later reading, share content across social media, and integrate with other productivity tools. Feedly's clean, customizable interface makes it easy to scan headlines and stay on top of relevant industry news, research, and topics of interest.

Limitation: While Feedly has great personalization features, it only has access to a few western news sources for aggregation. There are no options for Nepalese news, or Nepali language.

2.1.3 Hamro Patro

Figure 2.3: Hamro Patro

Hamro Patro is a popular web and mobile that was initially launched as a calendar app but has since has evolved to provide a range of services, including a Nepali news. Today it is a comprehensive platform that provides essential tools for daily life like weather forecasts, financial tools, and religious and cultural information relevant to Nepali community. The app aggregates news form various sources about current events, politics, entertainment, and sports both in Nepal and globally.

Limitation: As mentioned previously, Hamro Patro has established itself as a ”jack of all trades” app which provides many services relavant to Nepalese users, but when it comes to news aggregation its features are lacking compared to options like Google News or Feedly.

Chapter 3: Design and Implementation

Creating a robust and well-performing system requires a fair bit of planning and clear implementation. We spent a lot of time designing the entire system and even more time implementing it. This chapter is split into two sections: design and implementation where each topic is discussed in detail.

3.1 Design

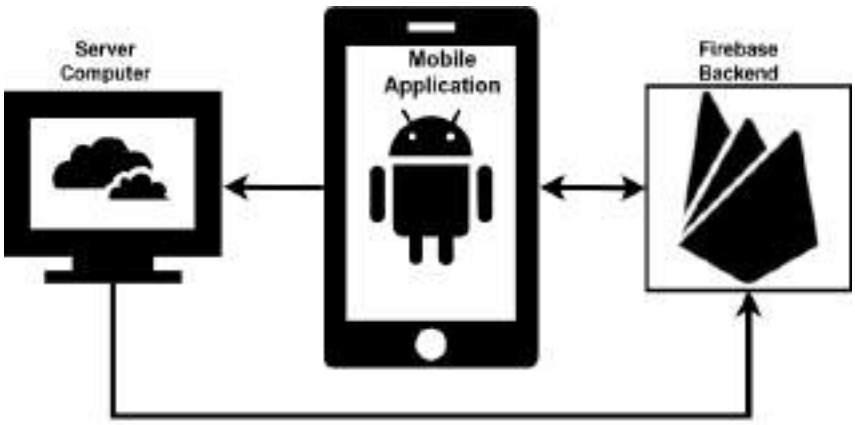


Figure 3.1: High Level System Diagram

The entire system of Hulaki consists of three distinct parts; the mobile application, the Firebase backend, and finally the self-hosted backend and database. The self-hosted backend is responsible for the scraping, clustering, and recommendation of news to the users. The Firebase backend is used for data persistence as well as to handle authentication and authorization of the users. The mobile application is the main point of entry for the users, where they can view and interact with the news.

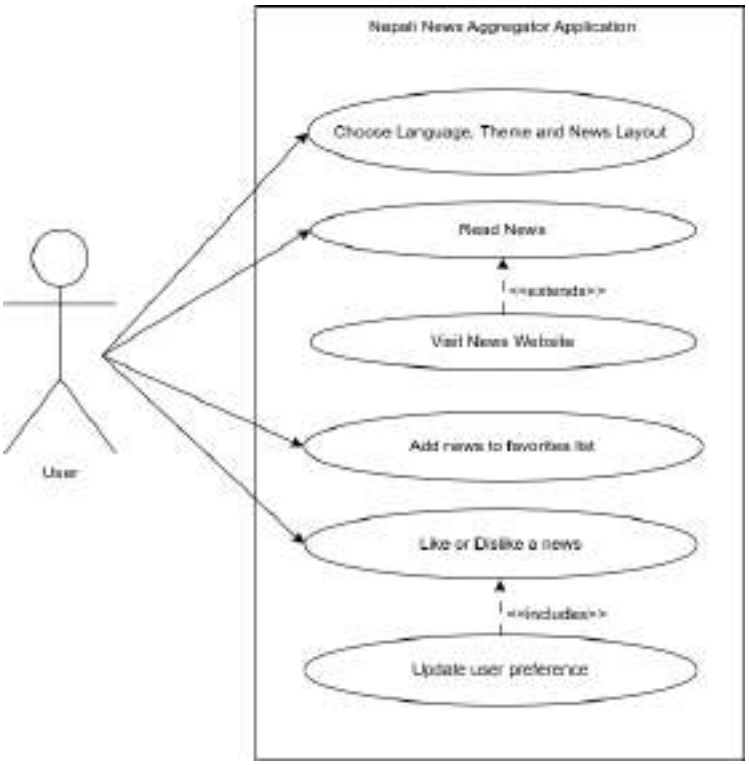


Figure 3.2: Use Case Diagram

The Hulaki application is intended to be used by a single type of user. They can customize the application experience to some degree by selecting the desired theme, language, and news layout. The user can read snippets of the latest news as well as visit the respective news website to read the full news article. The user can also save the news to a favorite news if they desire to read the news later. They can also like or dislike a particular news article which in turn updates the users' preferences.

During the development of the system, the team followed the Iterative Waterfall Model of Software Development Lifecycle. It allows for improvements and changes to be made at each stage of the development process, instead of waiting until the end of the project (GeeksforGeeks, 2024). Being able to get feedback and resolve issues in a quick time made the development process smooth.

The news dataset was collected through the means of web scraping. We chose 3 reputed Nepali news websites with varying political views to collect the data for this project. The news websites chosen are Onlinekhabar, Ratopati and Setopati. Since news article belongs to one of 6 categories according to what page they were scraped from. The categories are: breaking, national, international, finance, sports, and entertainment. As of January 12, 2025, we have collected 5,011 distinct news items. The distribution of data can be seen in Table 3.1 and Table 3.2.

Table 3.1: Number of news according to website

S.N.	Website	No. of news scraped
1	Onlinekhabar	1,863
2	Ratopati	1,980
3	Setopati	1,168
	Total	**5,011**

Table 3.2: Number of news according to category

S.N.	Category	No. of news
1	Breaking	1,517
2	National	1,360
3	International	357
4	Finance	725
5	Sports	560
6	Entertainment	492
	Total	**5,011**

The collected data perfectly illustrates the major focus of Nepali media outlets, which is national and political news (Acharya, 2018).

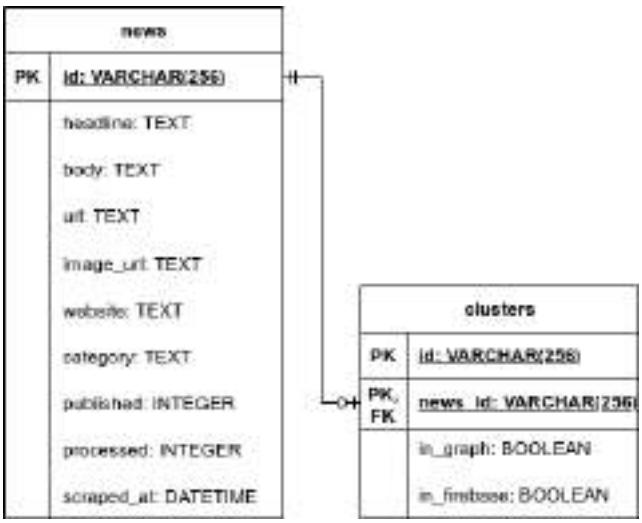


Figure 3.3: Entity Relation (ER) Diagram

We have used two different databases for data persistence in the project. The Firestore database in the Firebase backend is simply the local database implemented in a NoSQL database. The Firestore dataset is the main database the mobile application interacts with and retrieves news articles from. The local SQLite database

consists of two tables, “news” and “clusters” as illustrated in figure 3.3. The “news” table is responsible for holding all relevant details about a single news article which are headline, body, URL, image URL, website, category, date of publishing, and scraping time. After the news goes through the clustering algorithm, they are assigned a unique cluster and the “clusters” table is used to preserve this relationship between a single news article and the assigned cluster. Furthermore, there are also some boolean flags to keep track of the status of data in the system.

3.2 System Requirement Specifications

3.2.1 Hardware Specifications

1. Any computer with the following capabilities:
   (a) CPU equivalent of i5 or above
   (b) At least 8 GB RAM
   (c) Internet access
2. Any Android mobile device with the following capabilities:
   (a) Android Marshmallow (6.0) or above.
   (b) At least 200 MB of free space
   (c) Internet Access

3.2.2 Software Requirements

1. Python 3.11 or above
2. SQLite
3. BeautifulSoup
4. SciPy
5. NetworkX
6. FirebaseCLI and related packages
7. Android SDK
8. Flutter SDK

3.3 Implementation

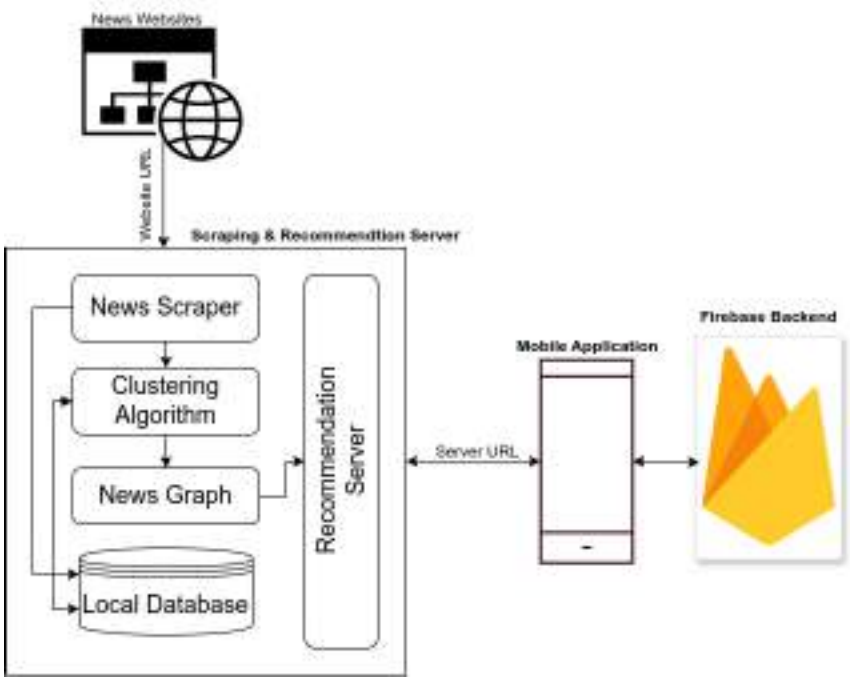


Figure 3.4: System Architecture Diagram

The system for Hulaki consists of a server computer, a Firebase backend, and the user's mobile device. The news scraper script uses website URLs to gather data and the mobile application is connected to the server computer using the server URL. The server computer is responsible for scraping and clustering the news articles as well as providing news recommendations according to the user's preference using a graph data structure.

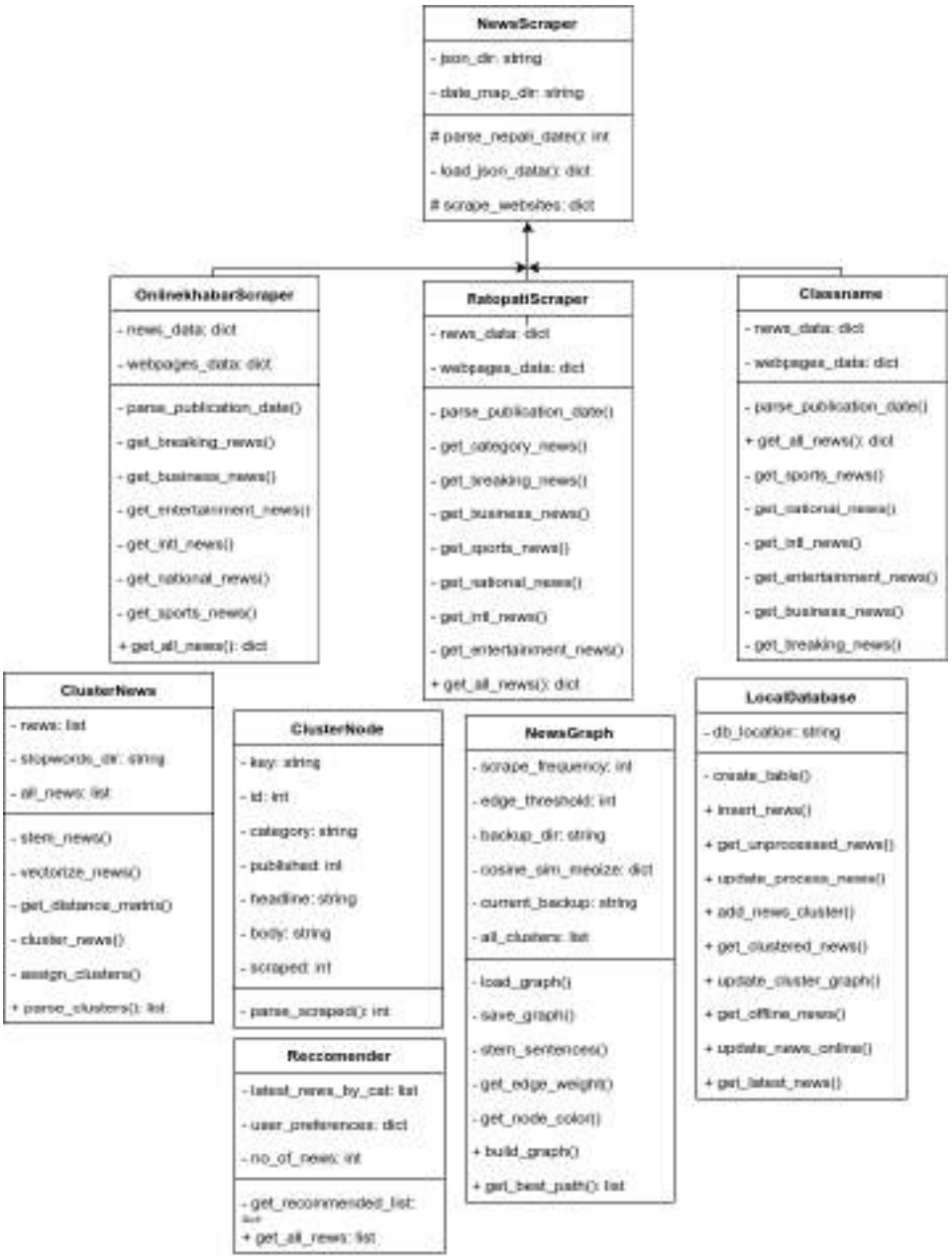


Figure 3.5: Unified Modeling Language(UML) Diagram of Server Computer Diagram

The server computer hosts a combination of Python programs that are responsible for their tasks. All relevant details will be further explored in the report.

3.3.1 News Scraping

We made a conscious choice to only select the news websites that are Server-Side (SSR) rendered since SSR websites are generally more straightforward than scraping Client-Side Rendered (CSR) websites (Pradeep, 2024)which makes the scraping

process easier and faster. We made use of the Python "requests" library to get webpage data and used the "BeautifulSoup4" library to parse the HTML into meaningful data. To make the scraping process as fast as possible, we also ran each website's scraping process parallelly on separate threads.
We first curated a list of all relevant web pages separated according to the website and the news categories and collected them in a JSON file. The scraping script makes use of the data in this file and performs the following actions on each iteration:

1. Scrape all web pages contained in the JSON file.
2. Collect all published articles and their respective URLs.
3. Scrape individual articles from their respective URLs along with all relevant details.
4. Insert new data into the local database.

Since the local database has a unique headline and unique URL constraint for each news article, it also makes sure that no duplicate news articles are inserted into the database and hence are not sent out for other processes like clustering, insertion in the graph, and insertion in the Firestore database. That being said, the scraping and subsequent steps are repeated every 20 minutes. Through experiments, we found this time interval to be the perfect sweet spot to get new news articles without wasting time scraping the ones we have already scraped.

3.3.2 News Clustering

One of the major hurdles in news aggregator systems is the existence of redundant data. Detecting duplicates is a rather difficult task which is very common when dealing with content aggregators (Aleksandrovich, 2022). Given the high possibility of the news websites posting the same news on their respective platforms, we decided to cluster the news to reduce data duplication. This not only makes the user experience better but also gives them more time to read other news as well.

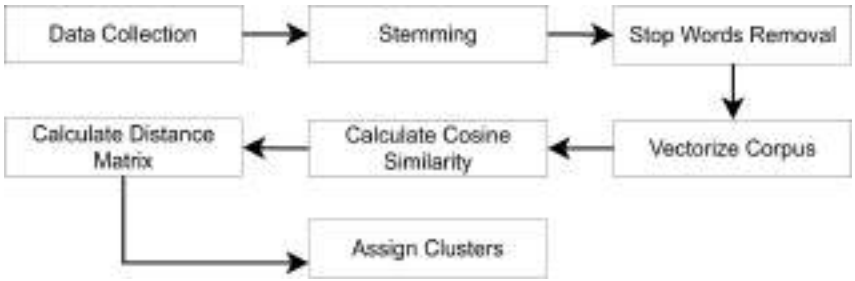


Figure 3.6: News Clustering Workflow Diagram

The system implements a news clustering system specifically designed for Nepali language text, using hierarchical agglomerative clustering based on TF-IDF vectorization and cosine similarity. The system processes unprocessed news articles from a database, clusters similar articles together, and stores the clustered results back in the database.

The process begins with data collection, where news articles are retrieved from a local database. Once collected, these articles undergo a preprocessing phase that starts with stemming - a crucial step particularly important for Nepali language text processing. The stemming process reduces words to their root or base form, which helps in standardizing different variations of the same word. After stemming, the system removes stop words that don't contribute significantly to the meaning of the text. This cleanup step helps in reducing noise and focusing on the meaningful content of each article.

The cleaned text is then vectorized using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization, which converts the text data into numerical vectors that computers can process. This vectorization creates a mathematical representation of the text corpus where each article is represented as a vector in a high-dimensional space.

Once the text is vectorized, we need to measure the similarity of the texts. Many methods exist to calculate the similarity of vectors but Cosine similarity is the better measure of text similarity when compared to Euclidean, Jaccard, and Pearson similarity (Popat et al., 2017). Cosine similarity measures how similar two articles are based on their vector representations. This similarity score is then converted into a distance matrix by subtracting the similarity values from 1. The distance matrix represents how different each article is from every other article in the collection.

Finally, using this distance matrix, the system performs hierarchical agglomerative clustering to group similar articles together. The clustering algorithm looks at the distances between articles and progressively groups them into clusters based on their similarity, with a threshold determining how similar articles need to be to be grouped. These final clusters are then stored back in the database, completing the workflow.

Figure 3.7: Example of the clusters (good cluster)

For a large chunk of the clusters, we found that the algorithm performs extremely well. The clusters contain new articles that share some grammatical as well as syntactical meaning. This greatly helps in the reduction of the number of duplicate news articles presented to the users. We can also find some bad clusters

Figure 3.8: Example of the clusters (bad cluster)

generated by the algorithm. This mostly occurs when the headlines and the body of the scraped text are shorter than other texts in the corpus. We believe that it is an issue with the TF-IDF method for vectorization as it has been found that this method performs poorly on texts shorter than 100 words due to sparsity issues (Kenter and De Rijke, 2015).

3.3.3 Graph Recommendation System

In this project, we have utilized a novel way of creating recommendation systems. Rather than opting for machine learning or deep learning applications, we have created a graph-based recommendation system. The recommendation system consists of two steps: first building a graph of connected news articles and then using this

graph to make recommendations.

In the graph-building stage, each news cluster becomes a node in the graph. The system then creates connections (edges) between these nodes based on how related they are to each other. This relationship is measured by looking at four main factors: how close together the articles were published, how similar their content is, whether they're in the same category (like sports or national), and how close in time they were collected from news sources. If two articles are related enough, scoring above a certain threshold, they get connected in the graph, with the strength of their connection reflected in the edge weight.

The recommendation process then uses this graph structure along with user preferences to suggest news articles. It starts with news articles that the user has previously shown interest in and finds paths through the graph to reach the newest articles. Think of it like finding a way through a map where each stop is a news article, and the paths represent how related these articles are to each other. The system prefers paths that go through more closely related articles with stronger connections. Since all the edge weights are integers, there is a high likelihood that there exist multiple paths with weights equal to the shortest part. With this characteristic of the graph, we take the path with most nodes but with minimum weight and these nodes in the path are the recommendations.

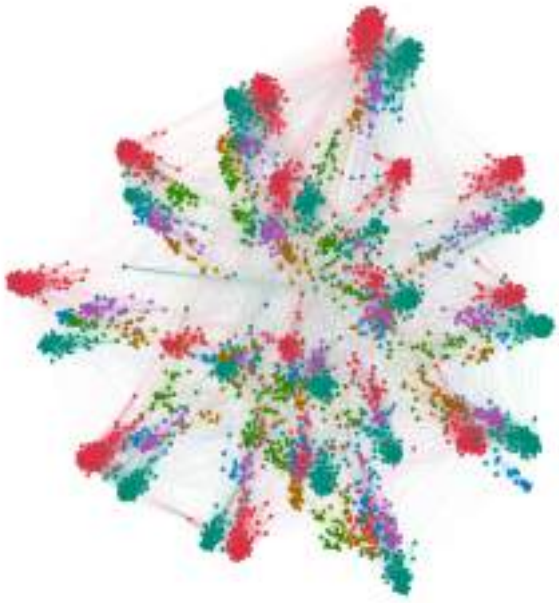


Figure 3.9: Visualization of the recommendation graph

The visualization of the graph can be seen in figure 3.9. This is a subset of the entire news graph with just 3,347 nodes and 104,681 edges visualized using the ForeAtlas layout. Each node color represents the news category of the particular news article. As can be seen, the news of the same category are very closely related and form strong connections. Also, you can see the connection of various nodes of several categories. Upon further inspection, it was found that the news articles that were published or scraped close to one another also have strong connections despite being of different categories.

Finally, the system compiles these recommended articles, sorts them by how recently they were collected, removes any duplicates, and if needed, fills up the remaining spots with the latest news articles to ensure the user gets enough recommendations. This way, users get news suggestions that are not only related to their interests but also fresh and diverse.

3.3.4 Mobile Application

The Android application for Hulaki was developed using the Flutter framework. The major aim of the mobile application was user satisfaction. Keeping this aim in mind, we have worked on 4 major features for the mobile application which are:

- **Theme Customization**: The users can choose what theme (light or dark) they want the application to be in.
- **Localization:** The application has all text available in Nepali as well as English language. This ensures that we can cater to users who can speak both languages.
- **User data stored locally:** Keeping the privacy of users in mind, we made the decision not to store user's data like preferences and favorites list online. All this data is stored offline which ensures data safety. The only information about the users we store is their email address.
- **Multiple layouts:** In the Hulaki app, the user can choose between 3 available news layouts to their liking. This customizability ensures the users have a smooth experience and can read the news in their layout of choice.

The different layouts we have offered are codenamed Listicle, Brainrot, and Pintrest. The descriptions of each layout are as follows:

1. **Listicle:** The Listicle news layout is a simple layout inspired by websites like Ycombinator Hacker News. This layout strips away any images and only offers news articles in simple text. This layout is aimed at people who do not care about fancy animations and interactivity like older people.

Figure 3.10: Listicle Style

2. **Brainrot:** The Brainrot news layout mimics TikTok's scrolling layout with lots of interactivity. One unique feature of this layout is that the users can share news articles with their friends on other apps. This layout is aimed at the younger generation who are familiar with such layout and care about images and interactivity.

Figure 3.11: Brainrot Style

3. **Pinterest:** This news layout is inspired by the popular app, Pinterest. This layout is the perfect blend between the previous two layouts. The layout

arranges various news articles in a masonry grid configuration along with the image associated with each image.

Figure 3.12: Pinterest Style

3.3.5 Firebase Backend

Firebase is a backend-as-a-service provided by Google. It provides lots of features and robust libraries to be used with various programming languages. In this project, we have used Firebase to handle the registration as well as authentication and authorization of the users. We have also utilized the Firestore NoSQL database to store the news clusters such that the mobile application can retrieve them. In the server computer, after each new news is inserted in the graph, it is push into the online Firestore database.

3.3.6 Recommendation Server

Each time the user opens the app, the app needs to get a list of recommendations based on the user's preferences. To serve this data from the graph recommendation system, we have created a small REST API using Python's FastAPI framework. The server is extremely simple with just two routes which are:

- /health: This is a simple GET route to check if the server is online. It is hit by the mobile application on each startup or refresh. The route returns true if the

server is online.

- /get-news: This is the main route of the server. It is a POST route that takes in a list of news liked by the user for each category. It then curates a list of 100 news articles consisting of both recommended news as well as latest news. It returns a list of news ids.

Chapter 4: Discussion on the achievements

Throughout the development of Hulaki, we faced challenges, particularly in reliable data for Nepali news. We overcame this by scraping content from respected Nepali news websites like Onlinekhabar, Ratopati, and Setopati, gathering over 5,000 articles across various categories. A significant achievement was implementing a clustering algorithm, which reduced data duplication by 14.2%, improving content relevance by decreasing repetitive news. We also used graph theory and mathematical models to provide personalized recommendations, enhancing the user experience by tailoring the app to individual preferences.

To combat misinformation, we prioritized verified news sources ensuring trustworthy content. The app's intuitive and customizable interface, built with Flutter, supports both Nepali and English, making it accessible to a wider audience. Additionally, by promoting local journalism, Hulaki contributes to the growth of Nepali media. Features like saving articles for later reading also increase user engagement. Overall, Hulaki fills a significant gap in Nepal's digital news landscape, offering a reliable, user-focused platform with room for future improvements such as AI-based personalization and offline capabilities.

4.1 Features

Keeping all our needs and major realm of focus for this project, we were able to implement the following features in the system.

4.1.1 Personalized News Recommendations

By utilizing graph theory and mathematical models, the app provides personalized news recommendations based on user preferences. As users interact with the app, it continuously learns and adapts to their reading habits, offering relevant and engaging news content.

4.1.2 Data Duplication Reduction with Clustering

A clustering algorithm was implemented to reduce data duplication by 14.2%. This ensures users do not see the same news article multiple times, improving content delivery and making the news feed more efficient.

4.1.3 Multilingual Support

The app supports both Nepali and English languages, ensuring that a broader audience can use the app and access news in their preferred language.

4.1.4 User-Friendly Interface

The app has a modern and customizable user interface that allows users to choose their preferred news layout, theme, and language. Additionally, users can save news articles to read later, providing a personalized experience.

4.1.5 Verified News Source Prioritization

To reduce misinformation, the app prioritizes news from verified and trusted sources. This helps users access accurate and reliable news, promoting trust in the platform.

Chapter 5: Conclusion

Hulaki successfully addresses the gap in Nepal's digital news ecosystem by providing a localized, personalized, and user-friendly news aggregation platform. By combining technologies such as web scraping, clustering, graph theory, and a modern mobile application, Hulaki offers an innovative solution tailored to the needs of Nepali users. The system effectively reduces data duplication by clustering similar news articles and ensures users receive personalized recommendations while promoting credible news sources. With multilingual support and an intuitive interface, Hulaki enhances user engagement and accessibility, fostering a richer and more efficient news-reading experience.

5.1 Limitations

- **Clustering Accuracy for Short Articles:** The clustering algorithm struggles with shorter news articles due to sparsity issues in TF-IDF vectorization.
- **Lack of Automated Fact-Checking:** The system does not include mechanisms for verifying the authenticity of news articles.
- **Dependence on Website Structures:** The scraping system is vulnerable to changes in the structure of the selected news websites, potentially disrupting data collection.
- **Internet Dependency:** The app requires a consistent internet connection, making it inaccessible for users in areas with unreliable connectivity.
- **Limited Personalization Techniques:** The recommendation system relies on graph-based approaches and could benefit from more advanced machine learning models for deeper personalization.
- **Scalability:** The current architecture may face challenges in scaling efficiently with a significant increase in the number of users or news sources.

5.2 Future Enhancement

- **Improved Clustering Algorithm:** Implementing deep learning-based NLP models like BERT or GPT for clustering and short-text similarity analysis.

- **AI-Powered Fact-Checking:** Integrating automated fact-checking tools to combat misinformation and improve content reliability.
- **Adaptive Scraping System:** Developing a more robust scraping mechanism that can adapt to dynamic changes in website structures.
- **Offline Access:** Adding offline reading capabilities to allow users to access news without an active internet connection.
- **Advanced Recommendation System:** Incorporating machine learning methods such as collaborative filtering, reinforcement learning, or hybrid recommendation systems for more accurate and adaptive personalization.
- **Scalability Enhancements:** Optimizing the architecture to handle a larger user base and a broader range of news sources seamlessly.
- **Real-Time Notifications:** Introducing push notifications for breaking news or personalized updates based on user preferences.
- **User Analytics:** Adding user behavior analytics to continuously improve the app experience and better understand user needs.
- **Multimedia Content Support:** Expanding the app to include multimedia content like videos and podcasts to enhance user engagement.
- **Regional News Coverage:** Increasing coverage of regional and community news to provide a more localized experience for users.

References

Acharya, B. B. (2018). An overview of nepali news media: Challenges and prospects. *The Journal of University Grants Commission*, 7(1):75–92.

Aleksandrovich, K. D. (2022). Research of the methods of creating content aggregation systems. , (1):9–31.

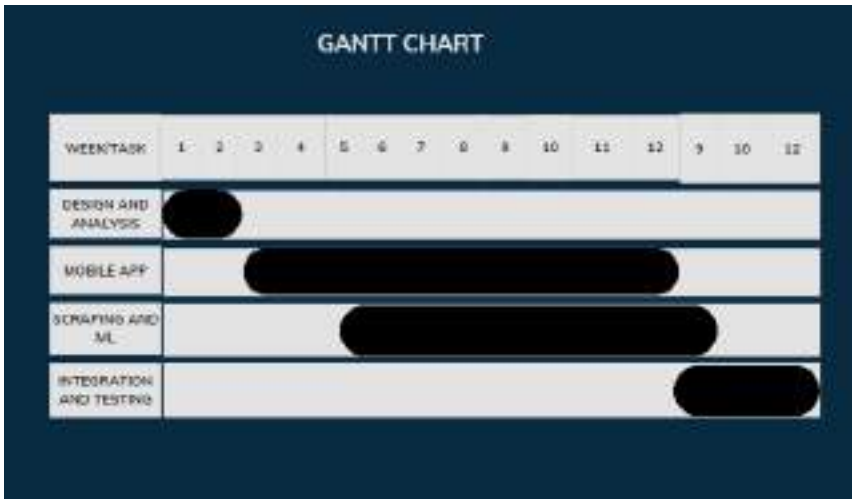
GeeksforGeeks (2024). Software engineering — iterative waterfall model. `https://www.geeksforgeeks.org/software-engineering-iterative-waterfall-model`.

Kenter, T. and De Rijke, M. (2015). Short text similarity with word embeddings. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1411–1420.

Popat, S. K., Deshmukh, P. B., and Metre, V. A. (2017). Hierarchical document clustering based on cosine similarity measure. In *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM)*, pages 153–159. IEEE.

Pradeep, A. (2024). Scraping dynamic websites with python: Ssr vs csr with selenium and puppeteer.

Appendix 1: Gantt Chart



Gantt Chart

Appendix 2: Mobile Application Screenshots

Screenshot of the login screen

Screenshot of the register screen

Screenshot of the listicle layout

Screenshot of the brainrot layout

Screenshot of the Pinterest layout

Screenshot of the profile screen

Screenshot of the news picker screen (when a cluster has multiple news)