

News Articles Curator: Project Report

Submitted in partial fulfillment of ISY5001 Reasoning System: Practice Module

Team Members:

- CHEN MINGYI EDMUND/ A0031105E
- VISWANATHAN CHANDRASHEKAR/ A0088591N
- CHENG YUNFENG/ A0215320Y



Contents

News Articles Curator: Project Report	1
1 Executive Summary	3
2 Background.....	4
2.1 Problem Description	4
2.2 Market Research.....	4
3 Objectives and Success Measures	5
4 Overall System Design	5
5 The Rule-based Engine	6
5.1 Knowledge Acquisition & Representation – How to Recommend Articles...	7
5.2 Machine Inference on User Preferences	8
5.3 News Article Retrieval and Labelling	9
5.3.1 Local News Labeler	11
5.3.2 Trending News Labeler	11
6 System Implementation.....	12
7 System Performance & Validation.....	13
7.1 Functional Test Cases.....	14
8 Conclusion	16
8.1 Limitations and Improvements	17
Annex	19
1 User Questionnaire	19
2 Functionalities to Techniques Map	20
3 Installation and User Guide	20
3.1 Running the application on Heroku.....	21
3.2 Running the application on LocalHost	26

1 Executive Summary

The rapid expansion of mobile devices technology resulted in the increase of ownership and usage of mobile devices. One of the many industries greatly impacted by this movement are the news presses. Traditional news presses are compelled by readers to transform hard prints to digital prints. Entry barrier to the news press industry also became lower as the internet made it that much easier to communicate articles to the masses. Readers could now access alternative news sources from social media platforms such as Facebook, Twitter, etc.

The web now contains an endless selection of articles, and a huge number of news providers. Readers are spoilt for choice, but this made selecting what one personally wants to read a daunting task. Major news presses need to cater to the general audience, hence could only categorize their headlines into the typical topics. From social media platforms, one was only limited to what were in their friend's feeds, or tweets.

It struck our team that an app that could cater to the personal interest of a reader and automatically scoured the web for articles most suited to a reader's personal interest would be of great utility. We wanted readers to focus on reading articles, rather than looking for the right article.

To achieve our objective, our team made use of web apis, which could at real-time, scour and build a database of news headlines. Then through interest levels solicited from the readers, we employed a rule-based engine to intelligently craft a list of articles personally suited to the reader. Apart from common topic labels (e.g. business, sports, science, etc) which articles already came in, we employed natural language processing techniques to label whether an article content was about local news, or whether it was about something trending on the net, all in the efforts to capture the user's personal reading habits into our headline recommendations.

The rule-based engine was built with python and integrated to a web backend built using python-flask. The frontend was built using HTML/CSS/JS, and together with the web backend formed our News Curator app.

2 Background

2.1 Problem Description

A typical individual would read daily to acquire knowledge, be it for leisure or for professional reasons. Individuals employed their unique set of heuristic in the selection of articles from the endless articles available on the web, whether it was going to a favorite news provider, or looking at the tweets and feeds of friends on social media.

Our team believed that article selection was a subjective process, as every individual had their own set of preferences, and inclination to a preference differs from person to person. However, we also believed that through asking the right questions and the assignment of certainty factors to interest levels, we could adequately capture the uncertainty in an individual's preferences and utilize it to craft a recommended list of articles.

2.2 Market Research

From our survey of news providers, we realized that major providers typically categorized their headlines into common topics such as business, science, technology, health, etc. It ended there as their objective was to cater to the general masses rather than each individual.

For people who retrieved articles from social media platforms, their choice would be limited to the feeds and tweets of friends, which might not always be relevant to their personal interest. Another con of social media was the legitimacy of the news, as fake news is increasingly becoming prevalent.

We saw an opportunity to introduce an app that could capture the user's personal preference, automatically scour the web for articles, and provide recommendations on relevant articles. We were also cautious in taking articles only from reputable and major news providers to maintain legitimacy of articles.

3 Objectives and Success Measures

Our team aimed to create an intelligent system that could capture the personal preference of the users with respect to news articles selection. Then use an efficient algorithm to scour the web and recommend the most relevant articles based on the preferences captured. Besides the full article, article keywords and summaries were also displayed to give user a quick glance of the recommended articles.

Intuitively our algorithm should capture factors such as

- Topics related to user profession,
- Topics related to leisure reading,
- Preference to certain news providers

To set us apart from typical news providers, we also captured preferences at a more personal level; individual's reading habits so to speak. These includes,

- Preferred amount of time reading a single article,
- User's age (we used results from research papers which correlated age and topic preferences to make an informed assignment of certainty factors)
- Preferences on reading local news and trending news

To give our system ease of use, we added a friendly web-based frontend so that user needed only to specify their preference and wait for a list of recommended articles. Number crunching was left to the backend, hidden from the end users. The final app was then deployed using Heroku: Cloud Application Platform.

Ultimately, we evaluated our system's recommendations through inspection of the outputs; making sanity checks on whether the outputs were in-line with our own expectations.

4 Overall System Design

The overall system design is depicted as a flowchart in the [Figure 1](#). A user-friendly frontend was created to solicit user preferences and display the eventual ranked articles. Web apis were used to scour the web for articles. To enrich the attributes

of the raw articles, they were first preprocessed then put through our article labelers. The News Article objects and User Preferences objects were fed into our rule engine to rank and recommend the top 10 articles to the user.

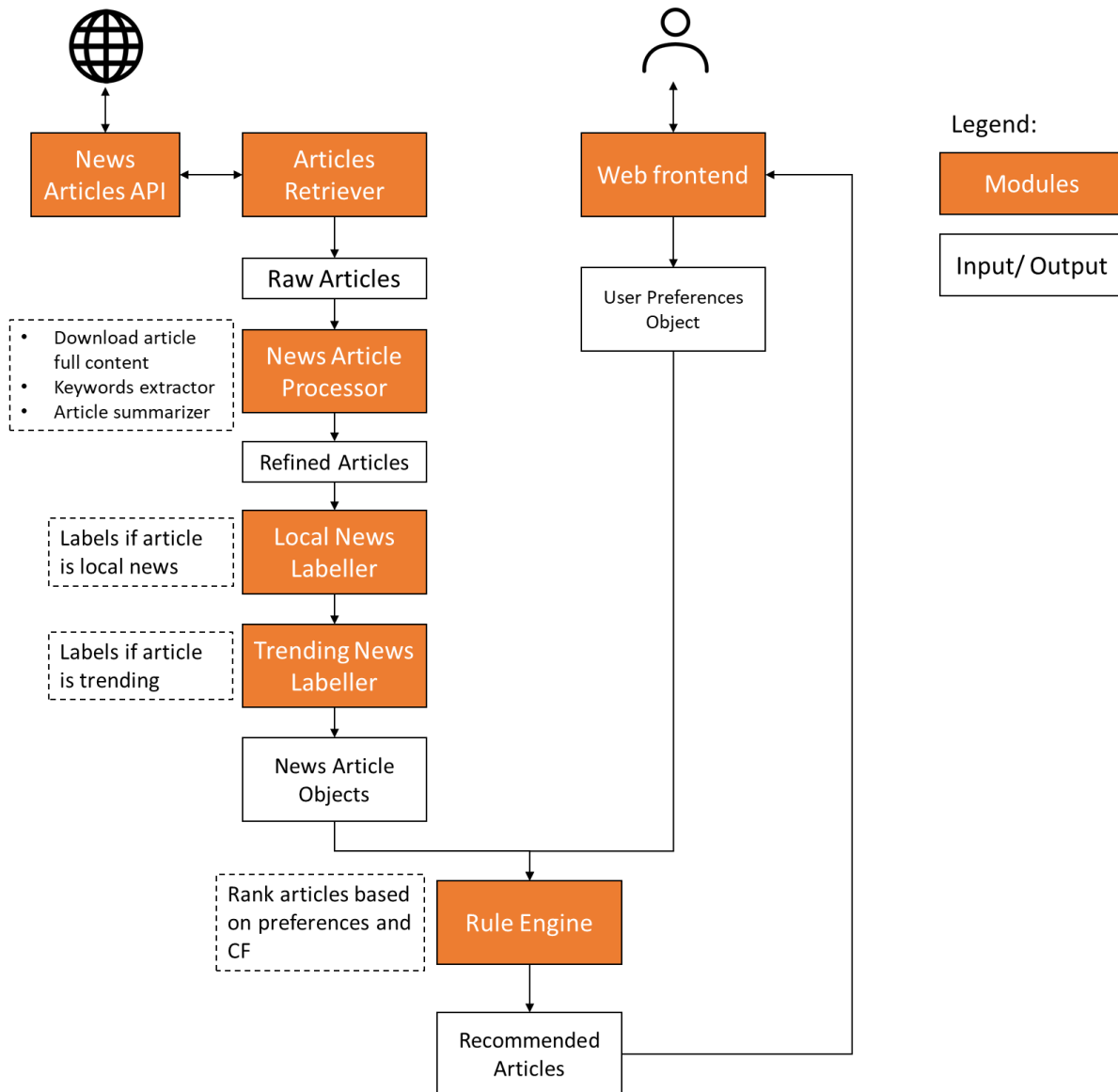


Figure 1: Overall System Design

5 The Rule-based Engine

This section will elaborate the underlying workings of our rule-based engine.

5.1 Knowledge Acquisition & Representation - How to Recommend Articles

First and foremost was the knowledge acquisition process, our team brainstormed for the most relevant factors that affected an individual's selection of news articles. These were represented as the inference diagram in [Figure 2](#). The top-level goal is how we should recommend relevant news articles. We break the top-level goal into sub-goals; which are factors affecting the selection of news articles. The observables under each sub-goal formed the preferences of an individual.

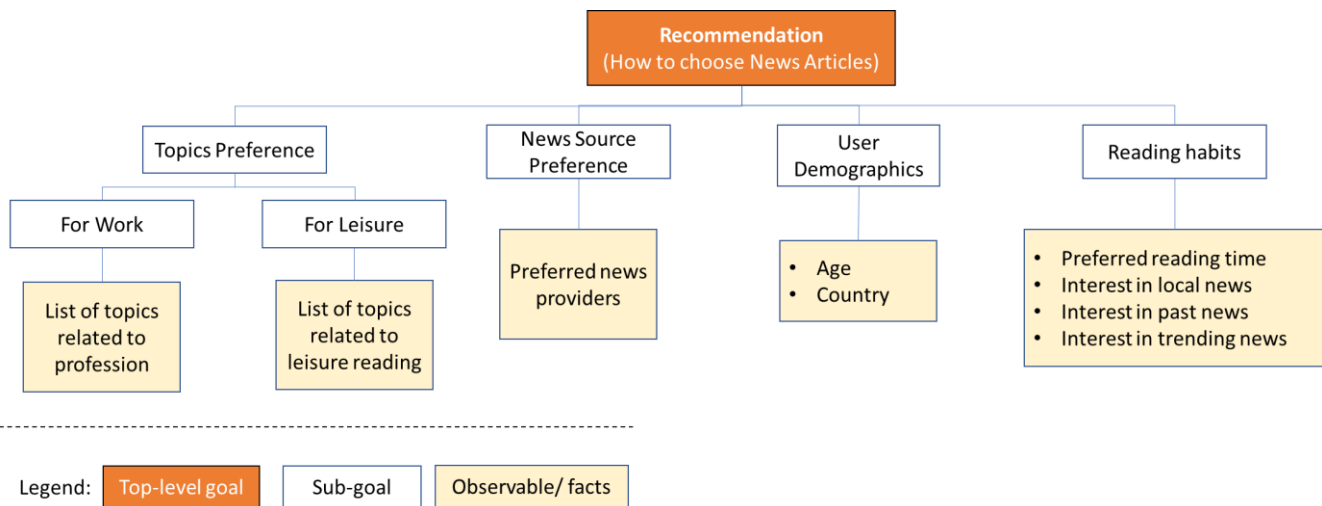


Figure 2: Inference Diagram to Recommend News Articles

With the inference diagram established, it became clear what questions to ask and preferences to solicit from the end-users. It also provided guidance on what attributes should a news article possess. We constructed the News Article Attribute Table as shown in [Table 1](#).

Data Object	Object Field	Data Type	Range
News Article	ID	Integer	≥ 0
	Content	String	NA
	Length (no. of words)	Integer	> 0
	Reading Time (mins) (inferable from Length)	Integer	> 0
	Source	Enum	Guardian, Bloomberg, BBC, CNN, CNA, etc

	Topic	Enum	Business, Technology, Sports, Science, Health, Entertainment, Current Affairs
	Date published	Date	NA
	?TrendingNews	Boolean	T/F
	?LocalNews	Boolean	T/F
	Country of origin	Enum	Countries

Table 1: News Article Attribute Table

5.2 Machine Inference on User Preferences

User preferences were subjective, a clear-cut “yes/no” response might not represent them adequately. To do inference on user preferences under uncertainty, we employed Certainty Factor (CF) in our rule engine. These CFs for each interest level was derived collectively through the group’s brainstorming sessions and through survey results from research articles. The complete questionnaire to solicit user’s preferences, along with the corresponding CFs, can be found it [Table 2](#) of Annex.

Upon obtaining the user’s preference, we further degraded the CFs of certain preferences as we felt some ought to be more important than others. For example, a user’s topic preference was the center piece in our rule-engine, and hence suffered no degradation. On the other hand, topic preferences based on age were inferred from population surveys, this reflected the general population’s preference and not each individual. We still wanted to account for such CFs but had degraded them by a factor of 0.25. These degradation factors were determined through the team’s collective intuition.

Preference	Topics	Age	Reading Time	Past news, local news, trending news	Article Source
Degradation Factor	1.0	0.25	0.25	0.75	0.5

Table 2: Degradation factor for each CF. E.g. a user aged 30 – 39 would have a CF of 0.62 for business topic, after degradation the CF becomes $0.62 * 0.25 = 0.155$

The CFs were then used as follows. Each rule in our inference engine had the same structure below:

If <article attribute> then <select article> (cf)

For example, suppose a user indicates his interest level of local news as *strongly agree* and business news as *likely to be related*, this would generate the following rules with corresponding cfs:

If <article is local news> then <select article> (0.2)¹

If <article topic is business> then <select article> (0.4)

Two cfs were then combined using the formula below:

$$cf(cf_1, cf_2) = \begin{cases} cf_1 + cf_2 \times (1 - cf_1) & \text{if } cf_1 > 0 \text{ and } cf_2 > 0 \\ \frac{cf_1 + cf_2}{1 - \min[|cf_1|, |cf_2|]} & \text{if } cf_1 < 0 \text{ or } cf_2 < 0 \\ cf_1 + cf_2 \times (1 + cf_1) & \text{if } cf_1 < 0 \text{ and } cf_2 < 0 \end{cases}$$

With this approach, an overall cf was derived of each article, and it forms the basis to rank the articles. The top ten articles were then recommended to the user.

5.3 News Article Retrieval and Labelling

Raw articles were obtained using a third-party Python package called News API. This package provided options for us to filter news articles by specific categories, countries, keywords, languages, preferred news sources and desired timeframes. For every user request, we retrieved a total of 20 news articles across all categories supported by News API – Business, Sports, Health, Technology, Science, Entertainment and General – for both US (global news) as well as the user's home country (local news). Further, we fetched 5 more *trending* articles based on our

¹ *Strongly Agree* is assigned a cf = 0.8, however for local news preference it was degraded by 0.25, resulting in a cf = 0.2.

proprietary logic (more details in section 5.3.2). The News API returned a JSON response comprising of details such as news source, author, title, published date, original article URL, content (truncated at 500 characters), etc.

These articles were first put through our News Articles Processor module which uses a Python Natural Language Processing (NLP) package called Newspaper3k that scraped the news provider's webpage to download the full article content. Subsequently, the same package was used to summarize and extract keywords from the full content.

At this stage, while the articles comprised of critical information that could be used for comparing with the user preferences and subsequent ranking, they still lacked the `local_news` and `trending_news` attribute. In order to identify if a news article was local news or trending news, we designed two algorithms to label the articles. Their system designs are shown in the flowchart in [Figure 3](#).

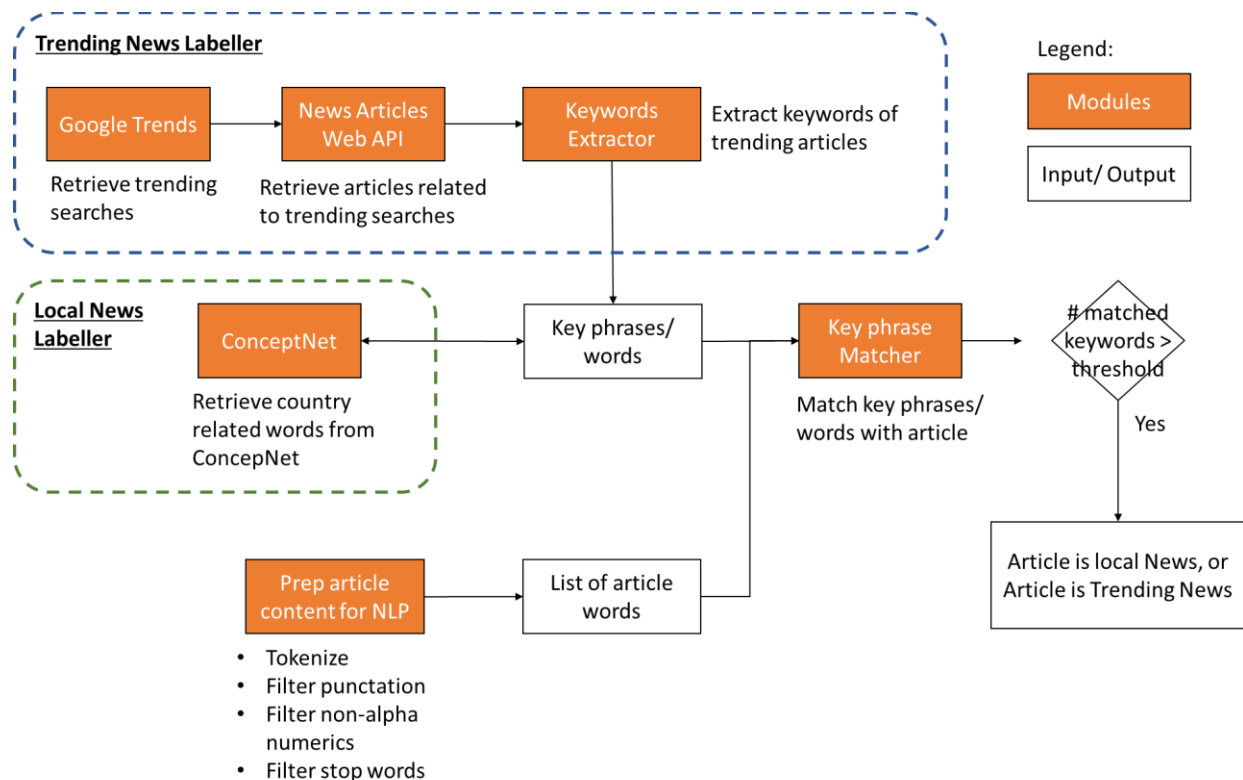


Figure 3: Article Labeler System Design

The core of the article labeler was a matcher that matched country related words, or trending related words to the news article content.

5.3.1 Local News Labeler

The reason we developed a separate local news labeler despite fetching news articles specific to the user's country was because the News API uses a naïve logic of fetching local news; headline/news source name which merely contained the country name was returned. For example, a Yahoo Singapore news article which was unrelated to Singapore would be returned when we search for Singapore articles because the news source name (Yahoo Singapore) contained "Singapore". In order to avoid such false positives, we came up with our own algorithm to label local news.

To do so, we obtained a set of country-related words using the web API's of ConceptNet². Subsequently, we compare the words of the news article content against the country-related words. Articles with high enough word matches between its content and the country-related words would be labelled as local news.

5.3.2 Trending News Labeler

While the approach was similar for trending news labeling, the retrieval of trending keywords was more complicated. The top 5 trending Google search words were first retrieved using Python Pytrends package. Subsequently, we utilized News API to retrieve one article related to each of the trending keywords as illustrated in Figure 4. Similar to the approach which was used for processing the original list of news articles, these trending news articles were also subjected to full content download and keyword extraction. The keywords of these 5 trending news articles (one article for each trending Google search word) were then compared to the

² ConceptNet is a semantic network based on the information in the OMCS database. It is expressed as a directed graph whose nodes are concepts, and whose edges are assertions of common sense about these concepts. Concepts represent sets of closely related natural language phrases, which could be noun phrases, verb phrases, adjective phrases, or clauses.

keywords of the original list of 20 news articles. Articles with a high match with the keywords of the trending articles were marked as trending news.

1	Thursday Night Football 2020 NFL schedule update: Why there is no Thursday Night ... CBSSports.com • 20h ago	1M+ searches
2	Steve Scully C-SPAN political editor Steve Scully suspended after admitting he ... CNN • 14h ago	200K+ searches
3	Rudy Giuliani Washington Post: White House was warned that Giuliani was being ... CNN • 9h ago	100K+ searches
4	Twitter down Twitter Down: Social Network Suffers Widespread Technical Problems Variety • 10h ago	100K+ searches
5	Vaughn McClure	100K+

Figure 4: In addition to the 20 news articles, one article is retrieved for each of the above trending keywords

6 System Implementation

The application backend was implemented using Flask, a web framework used to develop web applications in Python. The web frontend was implemented using HTML, CSS, and Javascript. User preferences were stored in remotely in a SQL database. The whole application was hosted via a Cloud Service provider called Heroku so that it was publicly accessible. The end-to-end system architecture can be seen from Figure 5.

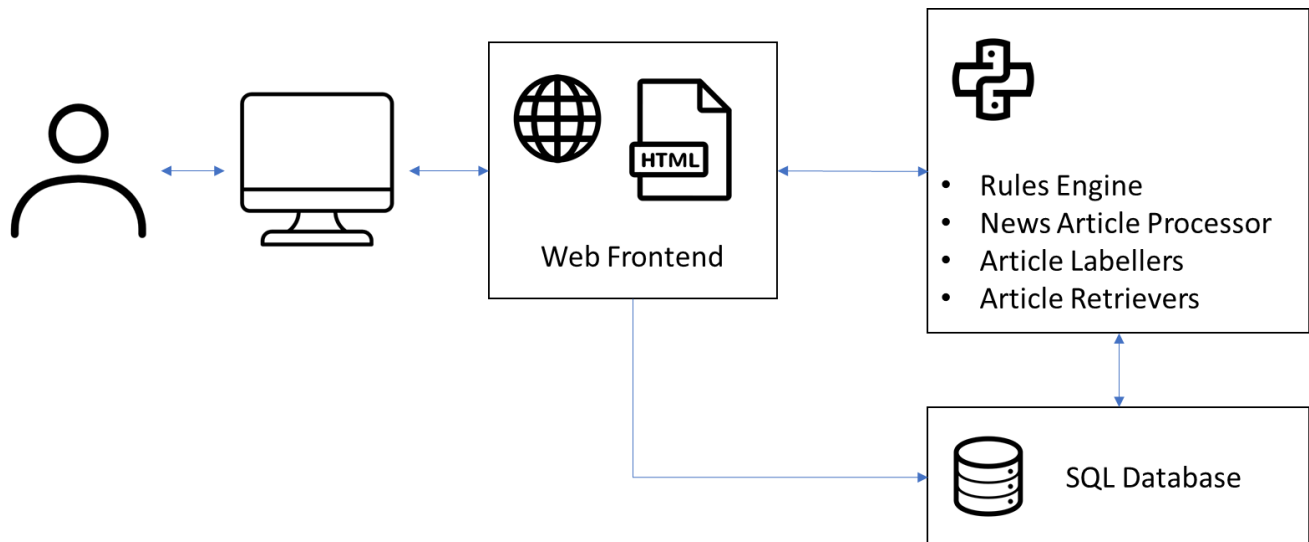


Figure 5: End-to-end System Architecture

The following table lists the API's that were developed as part of the backend:

API Name	HTTP Request	API URL	API Definition
Login	GET/POST	/login	<ul style="list-style-type: none"> The GET API returns the landing page of the application. The POST API accepts the user email and redirects the user to the User Profile page.
Show User Profile	GET	/userProfile	<ul style="list-style-type: none"> The Show User Profile API displays the form that needs to be filled up by the user indicating user preferences.
Create User Preferences	POST	/userPreferences	<ul style="list-style-type: none"> The Create User Preferences API is used to submit the user preferences from the form shown on the UI and subsequently persist the user details as well as the user preferences in the database. A unique id is created and returned to the frontend.
Show News Articles	GET	/newsarticles?id={}&articletype={}	<ul style="list-style-type: none"> The Show News Articles API is used to fetch Professional and Leisure news articles based on the user preferences stored in the database and returns the list of ranked news articles to the UI. This API takes in query parameters, id and articleType to retrieve the user preferences stored in the database. The id refers to the unique user id created in the previous API and the articleType can be either Profession or Leisure.

Table 3: List of API's developed as part of the News Curator backend

7 System Performance & Validation

As part of the system validation testing, we followed best practices of quality assurance in software testing. Once we defined the requirements and designed the application, we also prepared a test plan. This plan consisted of a detailed list of functional test cases that were performed after the development was completed. The test scope included validation of the various user flows in the application, the look and feel of the user interface and performance of the application. In addition

to functional testing, we also chose the most salient test cases and included them as part of our regression test cases which were carried out every time we made any updates to the frontend or backend of our application. The below table lists subset of the various happy path as well as negative functional test cases along with their expected and actual results.

7.1 Functional Test Cases

Test ID	Test Case Description	Expected Result	Actual Result
1	Verify that the email address is mandatory in the landing login page	UI needs to display "Please fill in this field" if user tries to login without entering email address	PASSED
2	Verify that user is redirected to user preferences page after login with email address	User Preferences page comprising of options for user to indicate preferences is displayed	PASSED
3	Choose preferences for any of the topics under "Select Topics Relevance To Your Profession". E.g.: Business – "Strongly Related", Sports – "Likely to be related" and the rest of the topics are "Not Sure". Click on Get Work Reads button.	User is displayed 10 news articles with the top 5 articles appearing with keywords and an article summary. Make sure the top articles with highest cf values are all Business Articles followed by Sports articles.	PASSED
4	Choose preferences for any of the topics under "Select Topics Relevance To Your Leisure Activities". E.g: Technology – "Likely to be Related", Entertainment – "Strongly related" and the rest of the topics are "Not Sure". Click on Get Work Reads button.	The work reads tab must still provide only Business and Sports articles as per choices in test case #3. It SHOULD NOT show Technology and Entertainment Articles as these are Leisure Reads.	PASSED
5	For the same choices selected in test case #3 and test case #4, click on Get Leisure Reads button.	The leisure reads must display Entertainment articles with maximum cf followed by Technology articles and should not show any of the work reads.	PASSED
6	Reset preferences to "Not Sure" for all topics. Now choose "Strongly Related" for both Entertainment and General under Professional Topics.	The work reads tab must show General Articles on top with a higher cf value than Entertainment Articles. This is because readers above the age of 45 tend to have more interest in General topics	PASSED

	Secondly, enter age as 45. Click on Get Work Reads.	than Entertainment Topics. For further details, please check Annex 1.	
7	Reset preferences to "Not Sure" for all topics. Now enter "How much time do you spend to read an article" as 5. Click on Get Work Reads.	Now we will observe the impact of the time to read an article on the cf values. We observe some articles with a higher cf than other articles. These articles may have a lesser overall world count and therefore fit the user's preference of being within 5 min. If all articles displayed on the page fit the user criteria of being within 5 min to read, the cf values could still differ based on other factors such as preferred topics based on age etc.	PASSED
8	Reset preferences to "Not sure" for all topics. Select "Definitely" for "Want to catch up on past news also". Click on Get Work Reads or Get Leisure Reads.	The Work reads tab would show articles which are more than 1 day old (UTC time) with a higher cf than the recently published articles. However, this is dependent on the time the query is made. By the time we make the query, if there are several recently published news articles, then we would never fetch any old news articles. This is expected behavior since the priority of our application is to provide most relevant and recent news for consumers.	PASSED
9	Reset preferences for all topics as "Not sure". Select any of the topics, for example Business as "Strongly Related". Choose "Like to know more about Local News" as Definitely. Click on Get Work Reads button.	The Work Reads tab shows the top articles and the articles with the highest cf will be labelled as "Local" as well. In the event that no articles are labelled as local, it is possible that none of the articles retrieved from the News API had any local country content.	PASSED
10	Reset preferences for all topics as "Not Sure". Choose "Like to see trending news" as Definitely. Click on Work Reads button.	The application would try to apply the trending news labeler to the different news articles and the work reads tab should show different topic articles tagged as Trending. However, if none of the articles retrieved from News API are labeled as Trending, then the Work Reads tab should directly display articles related to Google Trending Searches and not from	PASSED

		News API. These articles will not have a topic displayed.	
11	Reset preferences for all topics as "Not Sure". For any of the topics, for ex: Business, choose preference as "Strongly Related". Choose "Like to know more about Local News" as Definitely. Change country to "Philippines". Click on Get Work Reads button.	The highest ranked articles in the Work Reads tab must be Business and also Local news. This local news must belong to Philippines.	PASSED
12	Reset preferences for all topics and other general questions as "Not Sure" or "No preference". For any of the topics, for ex: Business, choose preference as "Strongly Related". Change country from "Philippines" to "Singapore" or vice-versa.	Under the "Which news sources appeal to you the most" section, we must see the news sources automatically changing based on the country top news sources.	PASSED
13	Reset preferences for all topics and other general questions as "Not Sure" or "No preference". For any of the topics, for ex: Business, choose preference as "Strongly Related". Select country as "Singapore". Under News Sources section, choose Bloomberg as "Include". Click on Get Work Reads button.	The Work Reads tab must return Bloomberg article with the highest cf. Once again there is a possibility that a Bloomberg article may not be displayed because the News API did not return any Bloomberg article. Further a Bloomberg article may also have a lower cf as other user preferences may have reduced its rank.	PASSED
14	The same test cases can be repeated for Leisure Reads as well. Make sure to change the topics preference under the Leisure section when testing for Leisure Reads. Further once the user is navigated to the Work Reads or Leisure Reads page, they can navigate to other tabs by clicking on the buttons on top as well.	Same as above cases (REPEATED FOR LEISURE)	PASSED

Table 4: Application Test Plan

8 Conclusion

The practice module allowed each of us to delve into unfamiliar domains while also enriching our experiences in familiar territories.

Through this exercise, we honed our skills in drawing inference diagrams and system design flowcharts by designing a fairly complex application. Designing this application also required us to create innovative algorithms to solve challenges posed by the limitations of third party packages used in this project. In addition to this, acquiring reliable data, designing the database, defining rules and measuring system performance are all valuable skillsets that we gained and we believe are critical for an AI engineer to get a clear picture of the problem, and understand the relationships between each moving component.

We have also experienced firsthand the benefits of harnessing the rich pool of packages available in the open-source community, and the challenges in integrating them together to serve a common goal. Lastly, we also believe that marrying the strengths of AI along with the advantages of software engineering certainly ensured that the application was robust, stable, and maintainable. In that regard, we spent significant effort in putting together the frontend and backend and deploying it via a cloud service provider to develop a complete web application that we believe enhances the user experience significantly. This opportunity to build an end-to-end system and to work with fellow students from different backgrounds and skillsets was an enriching and rewarding learning experience.

8.1 Limitations and Improvements

We understood that this MVP created was by no means the perfect product, we listed below some items which we felt could be improved further.

1. Support for non-English articles - As NLP techniques was something very new of all of us, we had faced many challenges in integrating the NLP packages for English language, let alone applying it for non-English language. With more experience, we will certainly like to extend the app to cater for non-English articles too.
2. Incomplete news information - As specified earlier, we used a package called Newspaper3k to extract the complete HTML content of the news articles and to perform NLP tasks such as keyword extraction and article summarization. Due to differently formatted HTML pages across different websites, the

Newspaper3k package sometimes extracted image captions or advertisement text too. In order to tackle these challenges, we had provided the original URL to the complete news article thereby allowing users to read the article from the original news source.

3. Trending and Local news labeler - While we created our own algorithms to label news as trending or local, our algorithms are not foolproof and users may observe discrepancies. With regards to local news, while ConceptNet provided a comprehensive list of terms related to a country, no list could be exhaustive, therefore there could be scenarios where an article was not correctly labeled as local news. With regards to Trending News, often real-time trending Google searches are significantly different, in nature, from news articles. This would result in none of the news articles being labeled as trending. However, in such scenarios, we included the news articles related to the trending searches along with the originally retrieved list of news articles and let the ranking system decide if these trending articles should be included as part of the top results returned to the user.
4. Performance – Since we are using an external package to download and process nearly 25 news articles, the latency was quite high and this was something we would like to improve. Caching the news articles' summary and keywords would improve the efficiency significantly since we would process the articles only once. However, since this was not the prime objective of the practice module, we are listing down performance of the application as a known limitation.
5. More user testing/ feedback - Given more resources, we would certainly like to field our app into the market and gather valuable real user feedback. CF numbers are subjective, thus real user feedback would give us valuable data to finetune them.

1 User Questionnaire

Q1) Which topics are related to your profession?									
a) Business	Let CF = -0.8 to 0.8 represent the strength of relationship. <ul style="list-style-type: none"> -0.8 = not related, -0.4 = unlikely to be related, 0.0 = not sure, 0.4 = likely to be related, 0.8 = related. 								
b) Technology									
c) Sports									
d) Science									
e) Health									
f) Entertainment									
g) General									
Q2) Which topics are related to your leisure activities?									
a) Business	Let CF = -0.8 to 0.8 represent the strength of relationship. <ul style="list-style-type: none"> -0.8 = not related, -0.4 = unlikely to be related, 0.0 = not sure, 0.4 = likely to be related, 0.8 = related. 								
b) Technology									
c) Sports									
d) Science									
e) Health									
f) Entertainment									
g) General									
Q3) How much time are you willing to spend on reading one news article?									
a) <= 5min	Articles with reading time < 5min have CF = 0.8 Articles with reading time 5 – 10 min have CF = 0.6 Articles with reading time >10 min have CF = 0.2								
b) >5 min, <10min	Reading time < 5min have CF = 0.6 Reading time 5 – 10 min have CF = 0.8 Reading time >10 min have CF = 0.6								
c) >10min	All articles have CF = 0.0								
Q4) What is your age?									
a) 18 - 29	CF assigned based on survey results in the link. https://www.americanpressinstitute.org/publications/reports/survey-research/social-demographic-differences-news-habits-attitudes/								
b) 30 - 39	Age	Biz	Tech	Sport	Sci	Health	Entmt	General	
	18-29	0.62	0.59	0.41	0.59	0.62	0.58	0.57	
c) 40 - 50	30-39	0.67	0.69	0.65	0.69	0.57	0.46	0.79	
	40-59	0.69	0.59	0.41	0.59	0.68	0.28	0.73	
d) >= 60	>=60	0.80	0.58	0.50	0.58	0.69	0.31	0.79	
Q5) Want to catch up on past days happenings also?									
a) Strongly disagree	Articles >1 day old have CF = -0.8								
b) Disagree	Articles >1 day old have CF = -0.4								
c) Not sure	Articles >1 day old have CF = 0.0								
d) Agree	Articles >1 day old have CF = 0.4								
e) Strongly Agree	Articles >1 day old have CF = 0.8								
Q6) Would you like to know more about local news?									
a) Strongly disagree	Local articles have CF = -0.8								
b) Disagree	Local articles have CF = -0.4								
c) Not sure	Local articles have CF = 0.0								
d) Agree	Local articles have CF = 0.4								
e) Strongly Agree	Local articles have CF = 0.8								
Q7) Would you like to see trending news?									
a) Strongly disagree	Trending news have CF = -0.8								
b) Disagree	Trending news have CF = -0.4								

c) Not sure	Trending news have CF = 0.0
d) Agree	Trending news have CF = 0.4
e) Strongly Agree	Trending news have CF = 0.8
Q8) Which news article source do you prefer?	
a) Guardian	Let CF = -0.8 to 0.8 represent the strength of preference <ul style="list-style-type: none"> -0.8 = Exclude -0.4 = Prefer to exclude 0.0 = No preference 0.4 = Prefer to include 0.8 = Include
b) Bloomberg	
c) BCC	
d) CNN	
e) CNA	
f) ... others	

Table 3: User Questionnaire

2 Functionalities to Techniques Map

Primary App Features/ Design Concepts	Courses/ Concept
System Design	<ul style="list-style-type: none"> Knowledge Acquisition – domain expert experience Knowledge representation (inference diagram, attribute table)
Rule-based Engine	<ul style="list-style-type: none"> Machine Inference Rule inference Inference under uncertainty (certainty factor)
Webapp frontend	<ul style="list-style-type: none"> System-user dialogue
News Article Processor	<ul style="list-style-type: none"> Knowledge Discovery (using NLP techniques) <ul style="list-style-type: none"> Article keywords extractor Article summarizer Discovering deeper attributes (news age, reading time, local_news, trending_news) from raw article data
News Article Labeler (local_news, trending_news)	<ul style="list-style-type: none"> Cognitive System techniques (NLP) Tokenization, case normalization, lemmatization, punctuation/stopwords removal Ontology - ConceptNet

Table 4: Functionalities to Techniques Map

3 Installation and User Guide

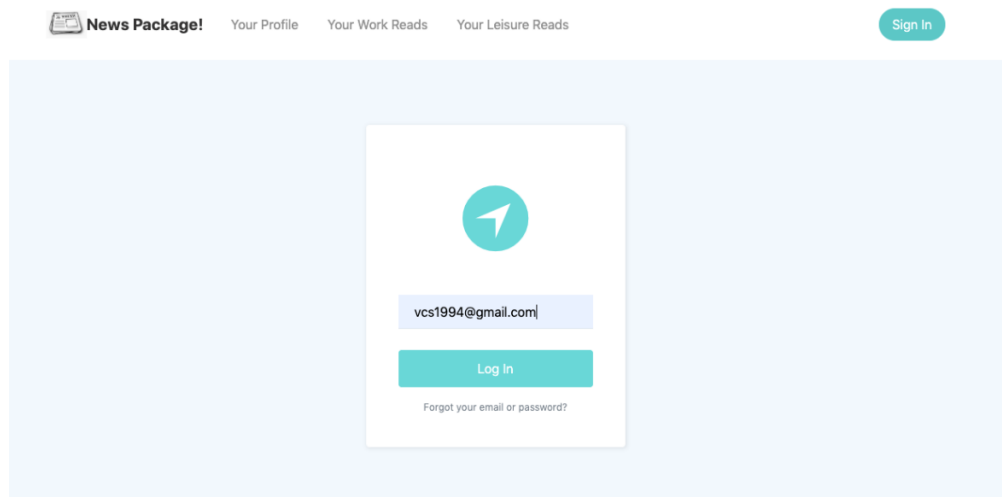
The preferred mode of using the News Curator application was to access the public URL hosted via Heroku, a cloud service provider. However, Heroku had a hard time limit of 30 seconds for every request and occasionally, we observed that downloading and processing ~25 news articles using the Newspaper3k package

could have a high latency thereby causing Heroku to return *Application Error* message. Therefore, we had also added instructions on how to run the application in the user's localhost.

Note: Currently, our application supported only countries with English language news articles: Australia, Canada, India, Ireland, Malaysia, New Zealand, Nigeria, Philippines, Saudi Arabia, Singapore, South Africa, United Kingdom and United States of America.

3.1 Running the application on Heroku

1. Navigate to the following public URL to launch the News Curator app: <https://mrpm-ay2021-newscurator.herokuapp.com/>. As we are using the free version of Heroku, users may sometimes notice that it takes 10-15s for the webapp to load.
2. Enter your email address as shown in the below screenshot and click on the Login button.



3. User is redirected to the user profile page where the user can enter their preferred choices.
4. All fields except email address on the user profile have an impact on the final list of news articles displayed to the user. Details on how each of these fields impacts the final rank can be found under "User Questionnaire" in the Annex.

- User is presented with 7 different topics to choose from and can indicate their degree of interest out of 5 choices in each of these topics as shown below.

SELECT TOPIC'S RELEVANCE TO YOUR PROFESSION.

Business Not sure	Technology Not sure
Science Not sure	Entertainment Not sure
General Not sure	

SELECT TOPIC'S RELEVANCE TO YOUR LEISURE ACTIVITIES.

Business Not sure	Sports Not sure	Technology Not sure
Science Not sure	Health Not sure	Entertainment Not sure
General Not sure		

- User can indicate their in interest in topics in 2 different areas: Professional and Leisure. The "Profession" choices would impact only the Work Reads whereas the "Leisure" choices would impact only the Leisure Reads.
- In the next step, user needs to indicate a few general choices which would apply to both Leisure as well as Work Reads as shown below.

HOW MUCH TIME (IN MINUTES) DO YOU SPEND TO READ AN ARTICLE?

5

WANT TO CATCH UP ON PAST NEWS ALSO?

No preference

LIKE TO KNOW MORE ABOUT LOCAL NEWS?

No preference

LIKE TO SEE TRENDING NEWS?

No preference

PLEASE INDICATE YOUR COUNTRY.

Singapore

8. Please note that the time to read an article must have a minimum value of 5 minutes. If the user enters a smaller value, an error will be thrown when the user attempts to navigate to the next page.
9. These general choices also have 5 different degrees of interest for the user to choose from as shown below.

WANT TO CATCH UP ON PAST NEWS ALSO?

No preference

Definitely Not
Not really
✓ No preference
I suppose
Definitely

LIKE TO SEE TRENDING NEWS?

No preference

10. User can choose their local country from the dropdown which comprises of the supported countries listed earlier.
11. In the next step, the user can choose their preferred news sources from the 5 different degrees of interest as shown in the below screenshot.

WHICH NEWS SOURCES APPEALS TO YOU MOST?

9to5Mac allkpop
No preference No preference

Business Times CNA
No preference No preference

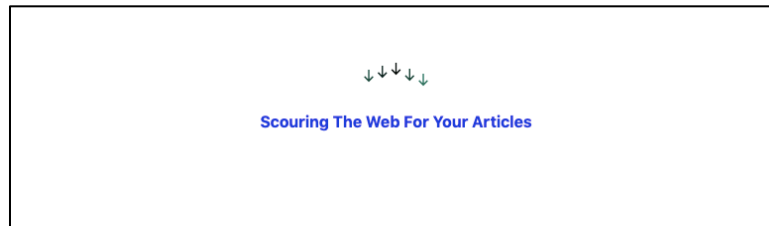
Daily Mail ESPN EurekaAlert
No preference No preference No preference

Exclude
Prefer to exclude
✓ No preference
Prefer to include
Include

12. Kindly note the preferred choice for news sources is only a secondary filter because our application is completely reliant on the news articles returned by the third party package, News API. If the articles returned by News API contain the news sources preferred by the user, it will be ranked highly. If not, the ranking will take place based on other user preferences.
13. Once the user has filled all their preferences, they can click on the Get Work Reads or Get Leisure Reads button as shown below. Once the user clicks the button, the application will retrieve, process and rank the news articles. Meanwhile, the user will be displayed a loading page as shown in the following page. This step could take ~20-30 seconds.

The Guardian No preference	The Straits Times No preference	Theguardian.comsociety No preference
Wired No preference	Yahoo Entertainment No preference	ZDNet No preference

Get Work Reads Get Leisure Reads



14. The user will be displayed the work/leisure reads as per their indicated preferences as shown below. The user has the option of clicking on the see more button to get a short summary and keywords of the article or alternately, the user can navigate to the original website of the article which is also displayed.

Man who didn't isolate after return from holiday led to at least 56 infections - Independent.ie (cf: 0.3240)

<https://www.independent.ie/world-news/coronavir...>
[Health] (Published: 2020/10/19)
[see more ...](#)

Coronavirus survives on skin five times longer than flu: Study - TODAYonline (cf: 0.3240)

<https://www.todayonline.com/world/coronavirus-s...>
[Health] (Published: 2020/10/19)
[see more ...](#)


China's economy continues to bounce back from virus slump - BBC News (cf: 0.3240)

<https://www.bbc.com/news/business-54594877...>
[Business] (Published: 2020/10/19)
[see more ...](#)


15. While the top 5 articles along with their certainty factor indicating their ranking will be displayed with the summary and the keywords, the next 5 articles can be seen when the user clicks on the "Other Articles" button. For these 5 articles, no keywords and summary will be present but a hyperlink to the original article is provided as shown below.

Other Articles
[Technology] Rainbow Six Siege Game Night - The Escapist (cf: 0.3180)
[Science] Scientists Measure The Shortest Length of Time Ever: Zeptoseconds - ScienceAlert (cf: 0.3180)
[Science] How to watch the Orionid meteor shower, expected to peak early Wednesday morning - Boston 25 News (cf: 0.3180)
[Entertainment] 'Teen Mom': One Kid Made \$50k On the Show - Showbiz Cheat Sheet (cf: 0.3160)
[Entertainment] Princess Diana almost did not use sexy dress because of 14-year-old Prince William - The Independent (cf: 0.3160)

16. After viewing the Work Reads, the user has the option to navigate to their user profile to modify their preferences or to view Leisure Reads. Both of these can be done via the tabs on the top as shown below.


News Package!

[Your Profile](#)
[Your Work Reads](#)
[Your Leisure Reads](#)

[vcs1994@gmail.com](#)


YOUR PROFESSION READS

Low take-up of electronics jobs and training opportunities; 'great pity' if positions not filled by S'poreans: Josephine Teo - TODAYonline (cf: 0.5268)

<https://www.todayonline.com/singapore/low-take-...>

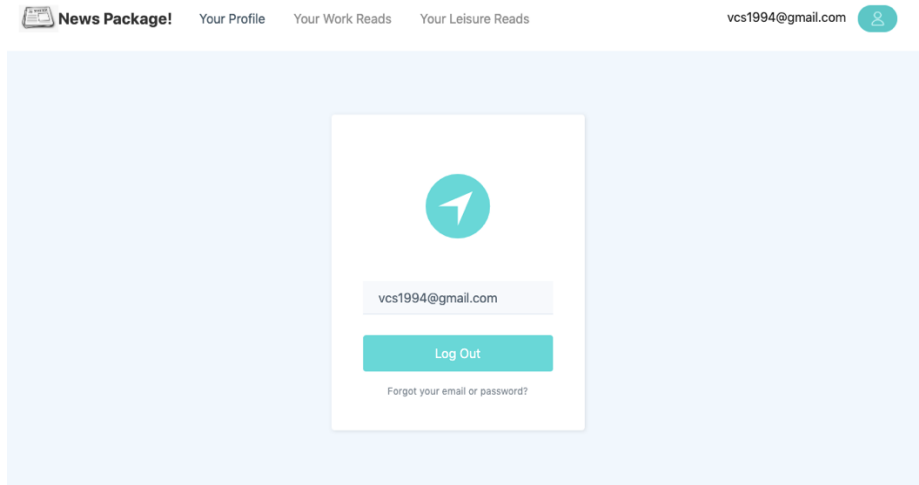
[Business | Local] (Published: 2020/10/19)

[see more ...](#)

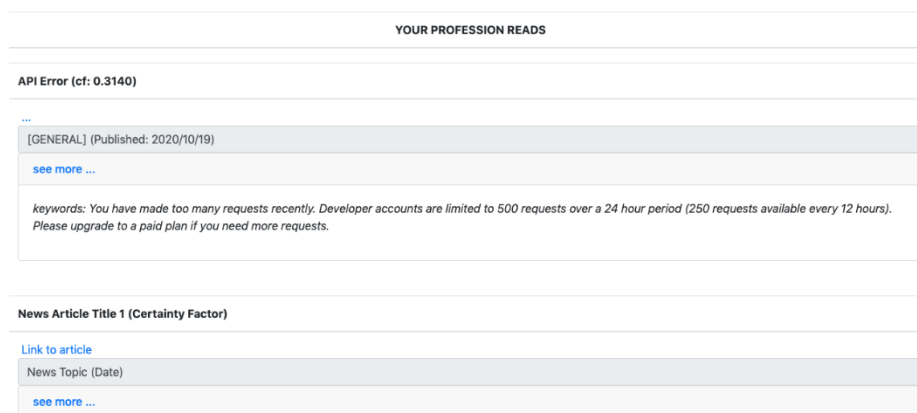
keywords: training, teo, jobs, opportunities, sporeans, takeup, positions, josephine, pity, low

SINGAPORE — Between April and September, there were more than 2,800 job positions and training opportunities offered in the electronics industry, but only about 220 — or around 8 per cent — of these have been taken up.

17. The navigation using these tabs can be done only from the Your Work Reads and the Your Leisure Reads tabs. If the user goes back to the "Your Profile" page, they can retrieve work reads or leisure reads only by clicking on button at the bottom of the page as shown earlier. Any changes made to the user preferences will be automatically saved in the database when the user clicks on this button and retrieved later when required.
18. If the user decides to close the window at any point, they can once again navigate to the URL and need not login again as the user email would be saved in the cookies at the first step itself. If the user is logged in automatically, they can directly click on the "Your Profile" tab to go to their user profile as shown below.



19. If the user receives the following page when checking the Work Reads or the Leisure Reads, this could mean that the third party package, News API is no longer accepting requests from our application. This is because we are using the free version of News API and they have limits of 250 API requests per 12 hours. This roughly translates to ~12 user requests to either Work Reads or Leisure Reads tab since every user request involves ~25 API calls to News API.



20. In order to bypass this issue, we advise the user to run the application on their localhost for which the installation steps are provided below.

3.2 Running the application on LocalHost

While the steps for using and testing the News Curator application are the same as above, the following steps will provide a guideline for the user to download, install

and execute the application on their local system. We have provided the required installation commands and the user can directly copy paste these commands into their Terminal/Command Prompt.

Note: The system needs to have Python3 version installed. The following steps assume that the user has installed Python3 in their local system. If the user would like to only check out the commands, the commands have been highlighted in bright red in the points below.

1. Clone the source code from the following GitHub link:
<https://github.com/nuschandra/NewsCurator.git>
2. Once the code is cloned and Python3 is installed, install Flask using the following command: **pip3 install flask**. Once Flask is installed, you can verify the installation by opening the Python interpreter (type python3 and press Enter) and type the command: **import flask**. If there are no errors, flask is successfully installed.
3. Now, run the following command – **pip3 install Flask-SQLAlchemy**. This is a Flask extension for SQLAlchemy, a Python SQL toolkit.
4. Run the following command to install the News API client which helps retrieve news articles – **pip3 install newsapi-python**
5. Next, install the Newspaper3k library which helps with downloading the HTML content of the news article and performing NLP tasks by running the following command – **pip3 install newspaper3k**
6. Run the following command to install the PyTrends package which fetches real-time trending Google searches – **pip3 install pytrends**
7. Next, the user must run the following command to enable NLP tasks such as word tokenization, stop words removal and word stemming - **pip3 install nltk**.
8. Run the following command to install marshmallow package – **pip3 install marshmallow**.
9. Lastly, once all the installations are successful, execute the following command – **export FLASK_APP=newscurator.py**. If using Microsoft Windows replace export with set in the given command. The application can then be executed

by typing `flask run` and the user can navigate to <http://127.0.0.1:5000/> or <http://localhost:5000/>

NOTE: If the user faces the same error as mentioned in point 19 under “Running the application on Heroku”, the user has the option of changing the API key variable `self.__newsapiKey` directly in the codebase in the following file: `process_news_articles.py`. We have commented out a few valid API keys that the user can choose from as shown in the screenshot below.

```
# VALID API KEYS: '1c190720995145afb28e31ee299cb526' # '679debc5c5dc4cb8b4ab551bcfbf35ce' # '7580ffe71bec47f7acfe7ea'
class ProcessNewsArticles:
    def __init__(self):
        self.__userProfilesDB = {}
        self.__keywordMatcher = None
        self.__newsapiKey = '1c190720995145afb28e31ee299cb526'
        self.__trendingArticles = []
```

To run and test the application, kindly follow the same steps from point 2 as shown under ‘Running the application on Heroku’.