Team Readly Members

1. Aviral Mansingka
2. Nihanshu Purohit
3. Sidhant Chadda
4. Sidhant Chitkara
5. **Sayali Aniket Kate (Project Co-ordinator)**

1 Problem Statement

Existing news applications either display all news, with the option of filtering by category, or only display news based on pre-decided categories of interest by the user. We plan to make the application more user centric by providing a voting system on the articles and learning more about the user's preferences. So users can be given news of relevance and interest to them rather than needing to look for it, with more accuracy over time.

2 Background Information

Throughout most of history, we have been given our news in the form of newspapers. Due to the Internet, writers have been able to put their articles on the web for everyone to access. However, despite having the power of the web, we are still unable to personalize the news content we desire to read. This is a problem because individuals want to learn about the world around them, but sometimes have trouble doing so because of the noise that pollutes news sources. Our team wants solve this problem because we want to connect people with the news they care about.

3 Environment

Front-end

Our front-end will predominantly be written in Riot.js. We made this choice of framework since riot.js is the most lightweight micro-framework out there. It simply requires knowledge of HTML and CSS, and integrates very well with existing jQuery libraries and is more extensible when it comes to using typescript, ES6 etc.

Back-end

Our back-end will be using a microservices architecture and have components written in Javascript and Python. The Javascript component will be largely responsible for all of the server side rendering and content delivery, and the python will be responsible for the recommendation system. Python is the language of choice for machine learning, and Javascript is a better tool for a backend.

Databases

Our database of choice is MySQL. The decision was primarily made since the nature of our data is very relational. Furthermore, relations are more suited for recommender systems. Most importantly, MySQL is known for having the largest collection of documentation. We would have chosen PostgreSQL but it doesn't have any features over MySQL that we really need.

News API

For actually getting our data, the APIs we have put into consideration are 'News API' and 'NYTimes API'. This will give us enough information to be able to give information to our users, and run the appropriate recommendations.

4 Requirements:

Functional:

As a User:

ID	Functional Requirement	Hours	Sprint No.
1	I would like to have facebook login and signup	4	1
2	I would like to have google login and signup	4	1
3	I would to see the trending news	10	1
4	I would like to upvote articles	5	2
5	I would like to downvote articles	5	2
6	I'd like news sources recommended to me initially	15	1
7	I'd like an estimate on reading time	5	2
8	I'd like recommendations based on past preferences	10	1
9	I'd like to filter my recommendations based on strength	5	2
10	I'd like to filter my recommendations based on length	5	2
11	I'd like multiple sources for a single topic	7	2
12	I'd like to mark articles relevant or not	8	1
13	I'd like to share articles on facebook	5	2
14	I'd like to be able to comment on an article	12	2
15	I'd like to blacklist certain news sources (if time permits)	15	2
16	I'd like to subscribe to a particular topic (if time permits)	20	2
17	I'd like to favorite an article	5	1
18	I'd like to click to read more about an article	10	1
19	I'd like to read headlines on the platform	10	1
20	I'd like to be able to contact developers for feedback	5	2
21	I'd like to report users for inappropriate content	6	2
22	I'd like to save articles for later reading	20	1
23	I'd like 'saved news' to be organized by categories	7	1
24	I'd like each category to be represented by images and text	6	2
25	I'd like to see my own settings on the platform	15	2
26	I'd like to see how preferences are stored (if time permits)	10	2

Non-Functional:

As a user:

ID	Non-functional Requirement	Hours	Sprint No.
27	I expect my preferences to be secure from other users	7	1
28	I expect a responsive and smooth experience (if time permits)	15	2

As a developer:

ID	Non-functional Requirement	Hours	Sprint No.
29	I expect my preferences to be secure from other users	7	1
30	I expect a responsive and smooth experience (if time permits)	15	2
31	I'd like the recommendation module to be reusable and extensible	10	1
32	I'd like recommendations and content delivery to be cheap to run	5	2

5 Use Cases:

5.1 Facebook Login

Action	System Response
1 Select "login with facebook" option	2 Provide pop up window to collect facebook login
3 Enter facebook login details	4 Verify information to allow or deny access

5.2 Google Login

Action	System Response
1 Select "login with google" option	2 Provide pop up window to collect google login
3 Enter google login details	4 Verify information to allow or deny access

5.3 Trending Articles

Action	System Response
	1 System will recommend trending news articles to user
2 User will select article of interest	3 System will remember interest for future reference

5.4 Up-vote

Action	System Response
1 Select "up-vote" option to vote for an article	2 Increment and display number of total votes for that content 3 Note user interest in this source or category of content

5.5 Down-vote

Action	System Response
1 Select "down-vote" option to vote against an article option	2 Decrement and display number of total votes for that content 3 Note user's dislike in this source or category of content

5.6 Initially Recommended Articles

Action	System Response
	1 Collect wide range of news categories for user to view
2 View options of recommended content	
3 Read and vote on articles	4 Note user interests in categories

5.7 Estimated Reading Time

Action	System Response
1 Select time icon on page of article	2 Calculate and provide an estimate reading time of the article

5.8 Recommended articles based on usage

Action	System Response
2 Vote on articles to further improve accuracy	1 Use user data collected over time to filter through available news content

5.9 Filter recommendations based on strength

Action	System Response
1 Click on "filter by strength" checkbox	2 Filter and display recommended news according to how strongly the system recommends it

5.10 Filter recommendations based on length

Action	System Response
1 Click on "filter by length" checkbox	2 Filter and display recommended from shortest to longest in length

5.11 Multiple sources for one topic

Action	System Response
	1 Determine news for user based on filters, recommendations etc 2 Show news of choice from all available sources, sorted in preference

5.12 Mark articles as relevant or not

Action	System Response
1 Click on "relevance rating" option to provide a rating for the system's future use	2 Use relevance rating to discard or increase similar articles

5.13 Facebook sharing

Action	System Response
1 Click on "share on facebook" button on the article screen	2 post to user's facebook as a shared article

5.14 Comment on articles

Action	System Response
1 Click on comments section to view comments 3 use an inline editor text box	2 Display comments of other users for selected article 4 Allow user to write comment, update comment screen with new comment

5.15 Blacklist news source

Action	System Response
1 Select a "blacklist" option on a source	2 Place a filter to not allow the particular source to be displayed or recommended to the user

5.16 Subscribe to topic

Action	System Response
1 select a subscribe checkbox	2 Alert user of all news related to subscribed topics

5.17 Favorite an article

Action	System Response
1 Select a heart icon by the article's preview to favorite it	2 Add article information to favorites and 3 Favorited articles to be considered by recommendation system,

5.18 Click to read more

Action	System Response
1 Click on a suggested article to read more	2 Redirect user to external source of article on a new tab 3 Note user interest in articles and sources that are opened

5.19 View article headlines

Action	System Response
	1 Display headlines of reccomended articles as links to the source articles

5.20 Contact developers for feedback

Action	System Response
1 Click on a "contact us" button 3 Type internal message to developers from the site itself	2 Redirect user to a template displaying contact information of development team 4 Send user's message to back end as an alert for immediate assistance

5.21 Report users for innapropriate content

Action	System Response
1 Click on a flag icon next to comments to alert system of innapropriate language, content etc	2 Alert development team of flagged content, remove if inappropriate

5.22 Save articles for later

Action	System Response
1 Click on a save for later button	2 Add desired article to a saved page user can access later and vote etc as usual Provide confirmation alert when article is saved for later

5.23 Organize saved articles by category

Action	System Response
	1 Once the user has saved multiple articles, group them by category for convenience 2 Inside each category, save articles in queue

5.24 Represent saved news by image and text

Action	System Response
1 Click on a "saved articles" button	2Direct user to view saved articles sorted by category 3 Each article should have a headline, preview text and a suitable image

5.25 View personal settings

Action	System Response
1 Click on a "personal settings" button	2 Redirect user to a settings form that displays all settings and filters he or she has added over time