The Fake News Predictor

Background

Fake news is false information spread via print or broadcasted by news media or social media. Fake news is written and published usually with the intent to mislead in order to damage an agency, entity, or person and/or to gain financially or politically. The issue has become rather critical and various organizations (such as WhatsApp) are actively campaigning to stop it.

Our project is a cure for this fake news disease. The product helps people decide the validity of a given article.

Overall description

The Fake News Predictor is a platform which helps eradicate fake news. The people can validate any suspicious news they come across. People can also put up various articles for review and review articles put up by other people. The entire system is based on merit based voting.

Environmental characteristic

People

The target group for the application is people active on various social media platforms. These people are the ones that interact the most with such articles and are the perfect sample space for deciding the validity of given news.

Hardware and Software

The product is a web application and the app should be deployed on a server that is capable of hosting node.js based web applications. The server should also support connection to a database service, preferably MySQL. The basic hardware is a machine that can listen to incoming requests and the OS on the machine should support node.js apps to serve over the network. The OS should also support database integration.

Functional Requirements

Functional Partitioning

On the basis of functioning, the product is to be divided into various partitions. They are:

- Login & Registration Module
- Dashboard

These partitions are further divided into sub modules.

a.) Login & Registration Module

This partition deals with connecting the users to platform. This modules consists of the following submodules:

1. **Registration-** Here, the user registers to our platform. This submodule collects the following information: Name, Email, Phone Number, a username and a password. User will be represented by the username.

2. **Login-** Here, the registered users can access their dashboard (See the following section). The authentication is based on username and password***.*** An additional provision for resetting the password in case the user forgets it is also required here.

b.) *Dashboard*

This partition helps user to explore, analyze, vote and submit their review all over the pages for different articles. Dashboard of our project is divided in 3 parts.

1. ***Perceiving and Searching of articles-*** User can view, read and search for variety of articles.

2. ***Review Articles-*** Our voting system allows user to vote on a variety of articles. This voting is merit based that is the more correct the user is over time, the more weight their vote carries. The user can also report inappropriate articles.

3. ***Submit your articles-*** Registered user can submit their articles for other users to vote on them. The article is declared valid if there is at least a 70:30 ratio of upvotes vs downvotes among at least 50 votes. In case the votes are less than 50, the article is in the review state i.e., an undecisive state. Once enough vote and the required ratio are acquired, the article goes into valid or invalid state, depending on the result of the voting.

Apart from that, the dashboard should also provide an ‘About User’ section where the user can see and edit their personal information. They should also be able to view their performance stats.

Control Description

The flow of control would be as follows:

1. The user starts with the app. The module rendered in that case will be login module. The user can log into the system, sign up, or continue without registering, as anonymous user.

2. In case the user wants to register, the user is taken to a form. On filling the form, the user gets a mail in order to validate their account creating request. The mail then redirects the user back to the beginning of the flow.

3. In case the user logs in, the user is taken to the dashboard module. The dashboard the leads to the above mentioned submodules. The user should be able to get back to dashboard from any of those submodules.

4. In case the user does not log in, they are only allowed to search and view articles.

5. If the user searches for an article, they can use keywords to find an article along with its state: valid, fake or under review. A registered user can report any inappropriate articles at

this point.

6. If the user votes on an article, they can see the vote in the about user section of the dashboard. The vote is immediately reflected on the article, deciding its future state based on the current state. Hence, the user can review various articles.

7. If the user decides to put up an article for review, it will be accessible to various people of the platform. The registered users can review the article. In case the article is inappropriate, the user loses a part of merit of their vote.

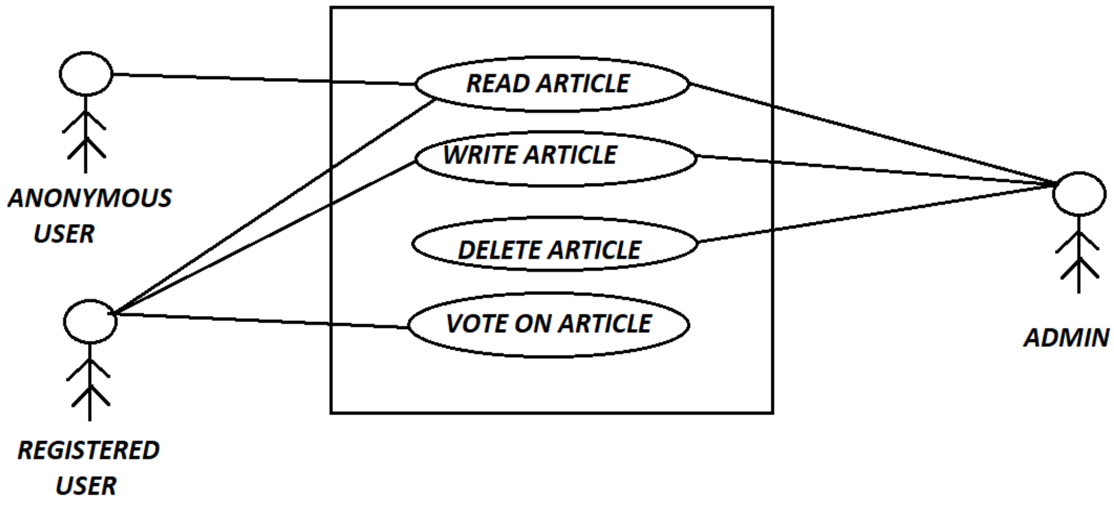
The entire business logic is described in the above flow.

Non-Functional requirements

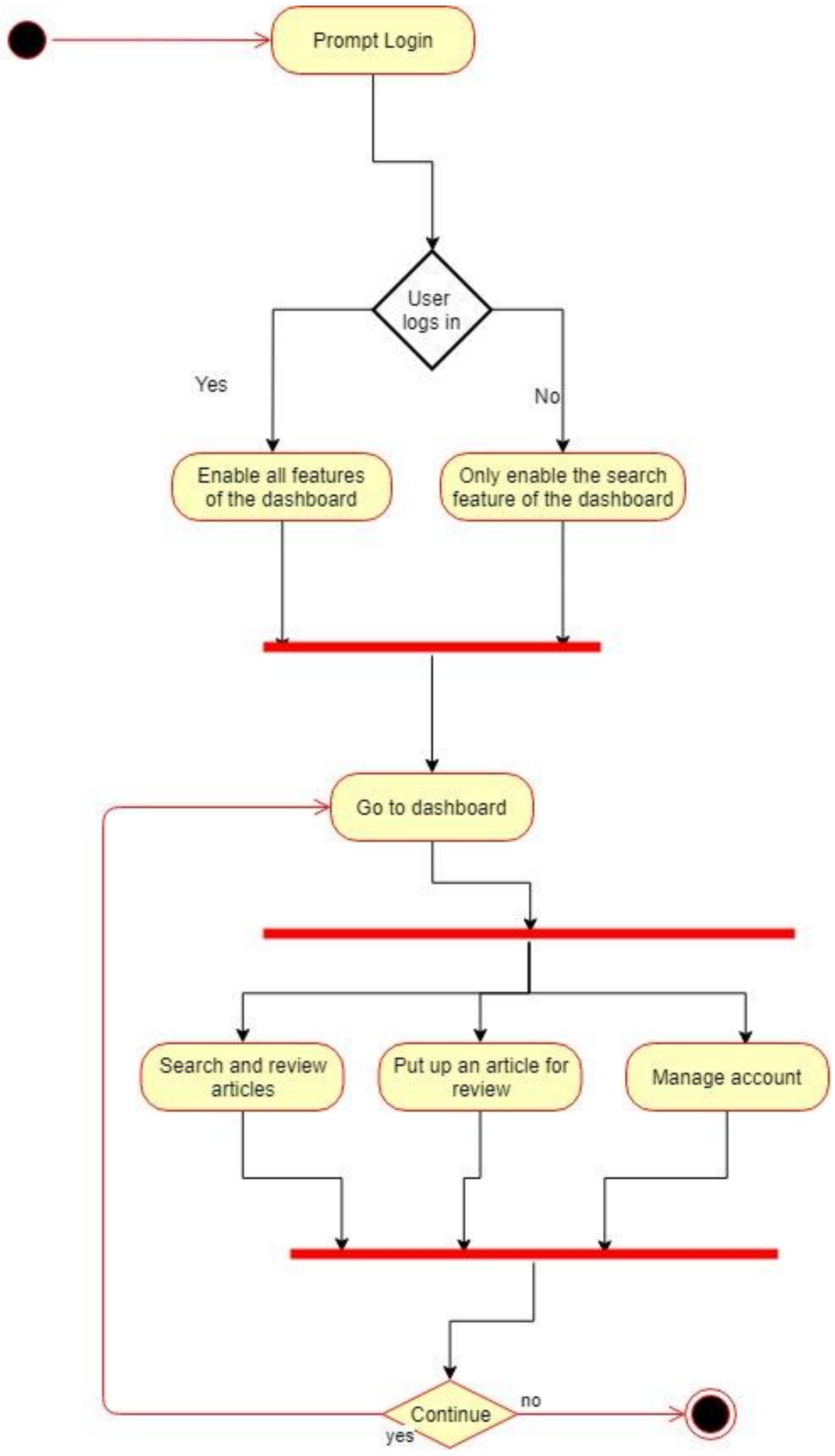
Our non-functional requirements are user and majorly active voters, who will vote on submitted articles. These users can also review and report articles. Due to these voters only our project will be on successful prediction of articles.

Behavioural Description

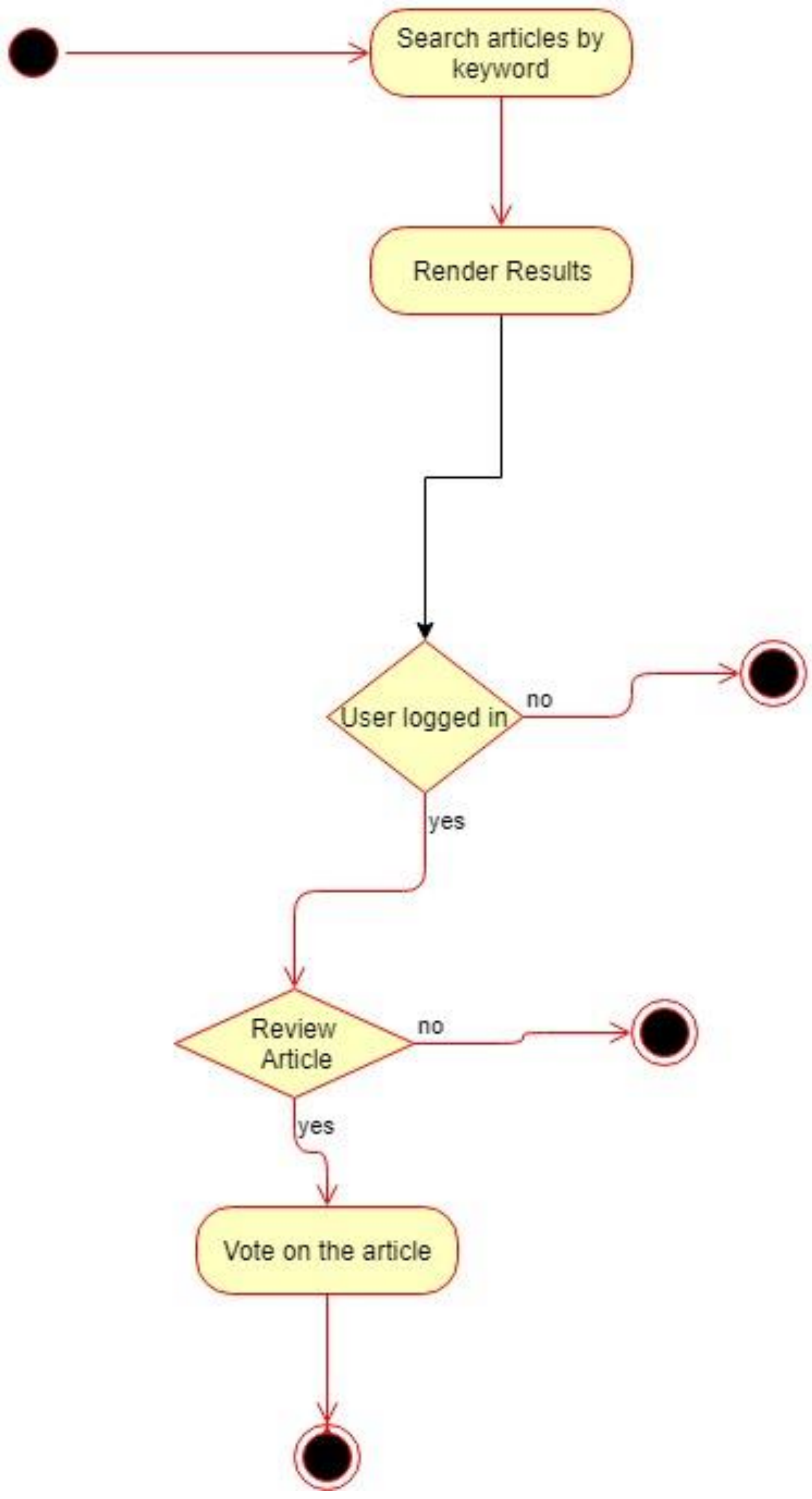
Use Case Diagram



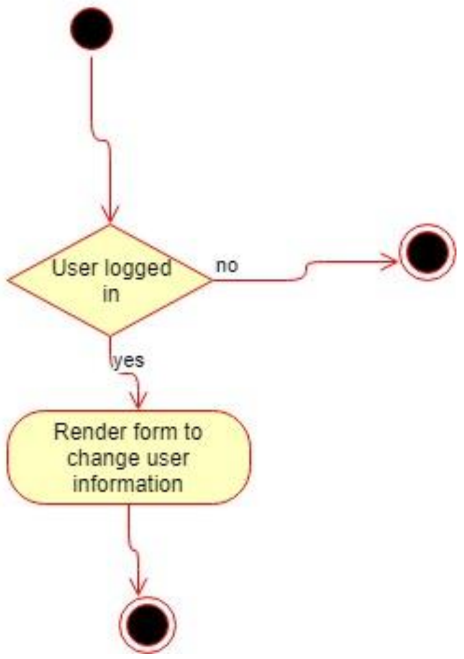
Activity Diagram


Prompt Login
User logs in
Yes
No
Enable all features of the dashboard
Only enable the search feature of the dashboard
Go to dashboard
Search and review articles
Put up an article for review
Manage account
Continue
no
yes

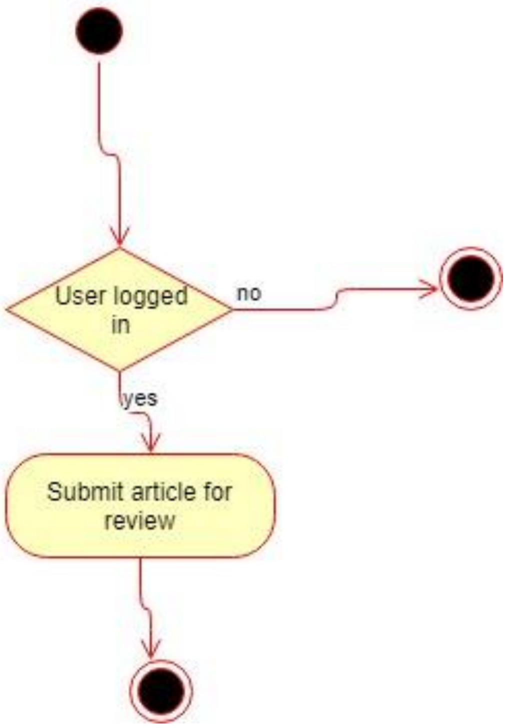
Search and review articles



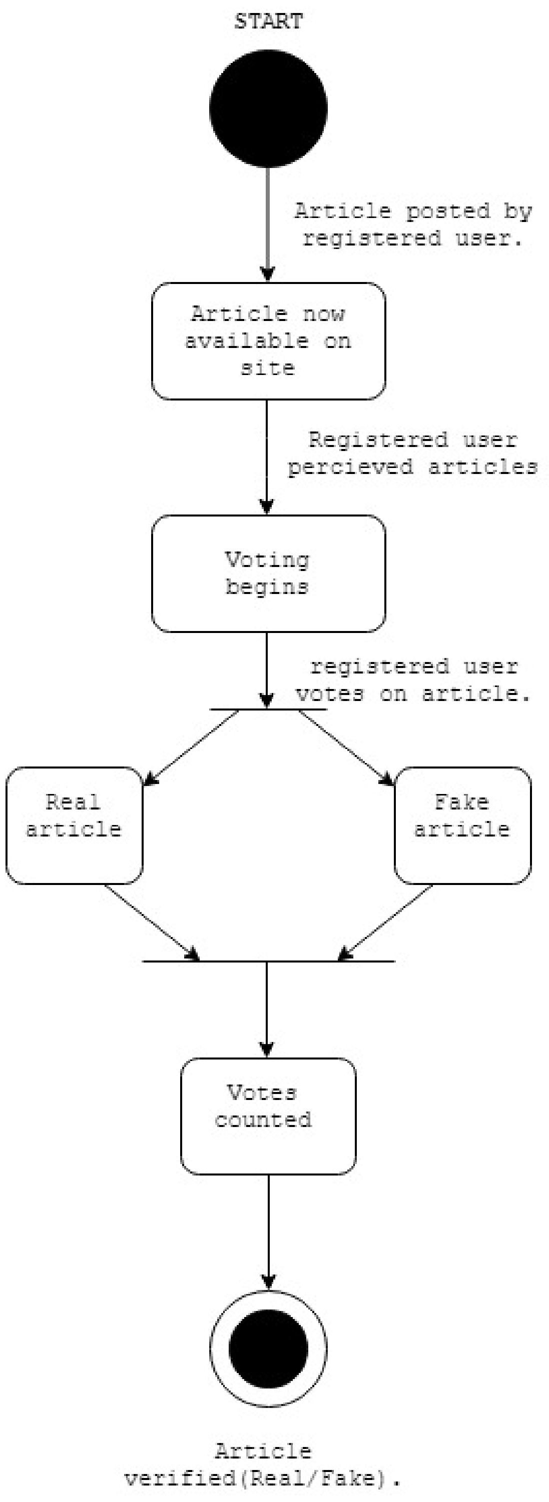
Manage Account



Put up article



State diagram for article review process


START
Article posted by registered user.
Article now available on site
Registered user percieved articles
Voting begins
registered user votes on article.
Real article
Fake article
Votes counted
Article verified(Real/Fake).

Deployment

On local machine for testing and development purposes and on an online web application hosting platform (preferably Heroku) for production code.

Interfaces

This section describes various interfaces used to interact with different components of the system.

Interface with database

MySQL Server – Community Edition

Interface with user

React based progressive web application. This application can be accessed using any modern browser. Even on relatively older browsers, the core functionality should be established.

Interface with the web server

Node.js based APIs and routing

Constraints

For successful prediction of articles, size of the target group should be considerably large.

Validation

1.Performance Bound: The system serves a large number of users at a time. The operation is CPU intensive and not interface intensive. The server should be able to serve the users with minimum delay. Hence, the application should serve users asynchronously.

2.Classes of tests: There are multiple classes of test required. The first class is the module validation. Each and every module should be checked for proper interfaces. That can be done by providing various test inputs and checking the corresponding output. The second class is simple flow validation wherein the flow of data between various submodules is checked. The final test is the logic check wherein the logic for voting and other fundamental operations that form the base of the application are tested.

3.Response to an undesired event: The app should go to an error state in such event. Depending on the severity of the error, the app may continue its flow or gracefully end, logging the user out for the current session. The error state should provide information about the event that occurred.

Group : 7

Arjun Chouksey : 171112257

Prabhat Kumar : 171112272

Yogesh Poonia : 171112258

Yash Mahalwal : 171112275