**A**

**Project Report**

**On**

**News By Tag**

**(http://newsbytag.herokuapp.com)**

B.Tech (CE) Sem-VI

In fulfillment of all requirements for

**Bachelor of Technology**

**In**

**Computer Engineering**

**SEM VI**

**In the Subject of**

**System Design Practice**

**Rajat N Movaliya (CE-66) (ID:15CEUOS109)**

Under the Guidance of

**Prof. Jigar M. Pandya, Assistant Professor.**

****

**DEPARTMENT OF COMPUTER ENGINEERING**

**FACULTY OF TECHNOLOGY**

**DHARMSINH DESAI UNIVERSITY**

**NADIAD-387001**

**GUJARAT**

****

**CERTIFICATE**

This is to certify that the project carried out in the subject of

System Design Practice entitled “News By Tag” and recorded in

this report is a work of

|  |  |  |
| --- | --- | --- |
| Rajat N. Movaliya | Roll No: CE-66 | ID:15CEUOS109 |

in Department of Computer Engineering, semester VI. He was

involved in project development during the academic session from

Dec-2017 to Apr-2018.

Prof. Jigar M. Pandya Dr. C. K. Bhensdadia

Assistant Professor, Head,

Computer Engineering Dept., Computer Engineering Dept.,

Faculty of Technology, Faculty of Technology,

Dharmsinh Desai University, Dharmsinh Desai University,

Nadiad. Nadiad.

**CONTENTS**

**Abstract** **i**

[**1. Introduction**](#page6) **1**

[**1.1 Project details: Brief Introduction**](#page6) **2**

[**1.2 Technology Used**](#page6) **2**

**2.** [**Software Requirement Specifications**](#page7) **6**

[**2.1 Purpose**](#page7) **7**

[**2.2 Scope**](#page7) **7**

[**2.3 O**](#page7)**verall Description 7**

[**2.3.1 P**](#page8)**roduct Perspective 7**

[**2.3.2 P**](#page8)**roduct Functions 8**

[**2.3.3 User Classes and Characteristics**](#page8) **8**

[**2.3.4 Operating Requirements**](#page8) **9**

[**2.3.5 D**](#page9)**esign and Implementation Constraints 9**

[**2.3.6**](#page9) **Assumptions and Dependencies 10**

[**2.4**](#page9) **External Interface Requirements 10**

[**2.4.1**](#page9) **User Interfaces 10**

[**2.4.2**](#page9) **Hardware Interfaces 10**

[**2.4.3**](#page9) **Software Interfaces 11**

[**2.4.4**](#page9) **Communication Interfaces 11**

[**2.5**](#page9) **Functional Requirements 12**

[**2.6**](#page9) **Non-Functional Requirements 15**

[**3. Design**](#page10) **16**

[**3.1**](#page10) **Activity Network Diagram 17**

[**3.2**](#page10) **Gantt Chart 18**

[**3.3 E**](#page10)**R Diagram 19**

[**3.4 Data Dictionary**](#page10) **20**

[**4. Implementation**](#page13) **22**

[**4.1 Modules Description**](#page13) **23**

[**5. Testing**](#page14) **26**

[**5.1 Testing Plan**](#page14) **27**

[**5.2 Testing**](#page14) **Strategy 27**

[**5.3 Testing**](#page14) **Methods 28**

[**5.4 Test**](#page14) **Cases 29**

[**6.**](#page14) **Screenshots 31**

[**7.**](#page14) **Software Version Deployment Steps 43**

[**8.**](#page14) **Conclusion 47**

[**9.**](#page14) **Limitations and Future Enhancements 50**

[**10.**](#page14) **Bibliography 52**

[**11.**](#page14) **About 54**

**List of Figures**

Link to all original images and docs : https://bit.ly/2JpLgX6

|  |  |  |
| --- | --- | --- |
| **Figure No** | **Figure Name** | **Page No** |
| **3.1** | **Activity Network Diagram** | **17** |
| **3.2** | **Gantt Chart** | **18** |
| **3.3** | **E-R Diagram** | **19** |
| **3.4** | **Data Dictionary (DD)** | **20** |
| **3.4.1** | **Tag Table of DD** | **20** |
| **3.4.2** | **User Table of DD** | **20** |
| **3.4.3** | **User\_Tag Table of DD** | **20** |
| **3.4.4** | **Article Table of DD** | **21** |
| **3.4.5** | **Tag\_Article Table of DD** | **21** |
| **3.4.6** | **Comment Table of DD** | **21** |
| **6.1.1** | **Sign Up Page** | **32** |
| **6.1.2** | **Log In Page** | **33** |
| **6.1.3** | **Changing Password Page** | **34** |
| **6.1.4** | **Feedback Page** | **35** |
| **6.2** | **Home Page before sign in** | **36** |
| **6.3** | **Home Page after sign in** | **37** |
| **6.4** | **Close News View** | **38** |
| **6.5** | **Autocomplete Search Page** | **38** |
| **6.6** | **Profanity word Page** | **39** |
| **6.7** | **Help Page Overview** | **40** |
| **6.8** | **Help Page Closeview** | **41** |
| **6.9** | **Admin Panel** | **42** |

**Abstract**

In the era of digitization, everybody has access to online news via various websites or applications, which shows an overview of all types of breaking news. That’s fine with regular use, but in this type, user irritates with many irrelevant news. So many users every time searches their favorite topics on the internet, which is waste of time.

The solution to this limitation is one can tell their favorite topics and the system will store that topic in that user’s account and give only news relevant to that respective topic. Also that topics will be updated frequently, so when the user comes next time, directly from user’s account, user can select any previously entered topic and see new breaking-news.

This is the aim of this project and by this, it will save the time of the user and increase knowledge in the direction user wants.

**CHAPTER: 1**

**INTRODUCTION**

1. **Introduction**

**1.1** **Project Details: Introduction**

Most of the news publishers give collections of news in the very high level of categories like business, art, politics etc. But most of the users want only news of very specific topics that interest them. Because of this, they end up searching this topic on the internet everywhere, they find possibility. Which is completely time consuming and inefficient process.

So, the aim of this project “News By Tag” is to solve this problem by giving users only news of topics, they want and ease out the process of reading relevant news.

The system will work like following. It will store all favorite topics of the user and update it, every time new news comes respectively to that topic. So the user does not have to go every time everywhere on the internet to just find news of his/her favorite topic.

**1.2** **Technology Used**

**Front End:**

* **HTML (v4.1)**
* **CSS (v4)**
* **JavaScript (jQuery (v3.3.1))**
* **Bootstrap (v3.3.7)**

As the system is a web application, the need for **HTML, CSS, and JavaScript** can be understood.

All the static content like images, custom CSS and JavaScript file of the project will be loaded in requested page directly from the server where it resides.

**JQuery** is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers

**Bootstrap** is an open source toolkit for developing front-end of a web app with HTML, CSS, and JavaScript. It will be loaded every time in the requested page via CDN (Content Delivery Network) service, which loads it in the compressed version from nearest location possible to the requester of the page to faster the process of loading a page.

**Back End:**

* **Python (v3.6.4) ( Flask Framework (v0.12.2))**
* **SqlAlchemy (v1.2) (ORM Module of Python)**
* **News API (v2)**
* **FlaskMigrate (v2.2.1)**

The project is built using **Flask framework** of **python** with **SqlAlchemy** as **ORM module** (Object Relational Mapping) to handle back-end side of database manipulation.

**NewsAPI** is an API service for fetching new breaking news and blog post of particular topics with HTTP requests and getting a response of news in JSON (JavaScript Object Notation) format.

**Flask-Migrate** is an extension that handles ORM SQLAlchemy database migrations for Flask applications using Alembic. To migrate database schema without losing original data.

**Database: PostgreSQL (v10.3)**

**PostgreSQL** is a powerful, open source object-relational database system. Classified as a semi-structured database program, PostgreSQL (latest version) supports both SQL and NoSQL operation. In the project, only SQL part is used.

**API Testing Tool: Postman (v6.0.1)**

**Postman** tool is used to make API development simple while testing and processing request and response.

**Version Control Tool:**

* **Git (v2.17) (VCS)**
* **GitKraken (v3.5) (GUI Tool for Git)**

**Git** is a [free and open source](https://git-scm.com/about/free-and-open-source) distributed VCS (Version Control System) designed to track modification in project files with versions.

**GitKraken** is GUI (Graphical User Interface) tool to handle all simple Git operation with clicks instead of commands in the terminal in a very fast and effective manner.

**Deployment Platform: Heroku (PaaS)**

**Heroku** is a cloud platform (Platform as a Service) that lets the user build, deliver, monitor and scale apps with the support of CI (Continuous Integration)

**Diagram Environment: ERD-Plus**

**ER-Diagram** of this project is made in online web app ERD-Plus.

Link: https://erdplus.com/

**Operating System: Ubuntu (v17.10)**

**Ubuntu** is an open-source operating system for computers. It is a Linux distribution based on the Debian architecture and is used as the development environment of this project.

**CHAPTER: 2**

**SOFTWARE REQUIREMENTS**

**SPECIFICATIONS (SRS)**

**2.** **Software Requirement Specifications**

**2.1 Purpose**

This is an SRS document refers to News By Tag Release 2018 version 1.0.0. It describes the functionality and specification of how the System will help the user save time by providing only news they want in an efficient way. It also specifies how it will interact with the end users.

**2.2 Scope**

This system will provide the functionality of adding a new tag to account and system with show trending tags based on how many users store that tag.

This system will not provide a general overview of all kind of live breaking news as it is provided by most of the other web application like the-hindu, the-times-of-india etc.



**2.3 Overall Description**

**2.3.1 Product Perspective**

This software is developed as a part of coursework the subject “System Design Practice”. The software aims to provide only news which user wants with ease. The web- app‘s main perspective is towards saving the time of the user and providing a user-flexible system of online news reading.

**2.3.2 Product Functions**

The product offers a breaking news related to tags entered by the user in their account. Once the tag is added to the system, it will be updated every day.

The user can add new tag directly by writing or by searching tag (entered by another user) or also from trending tag section.

The system won’t allow the user to add a tag which is vulgar. Currently support for only three languages English, Gujarati, Hindi in profanity filter. Admin can add any tag to block list if he/she find it vulgar.

**2.3.3** **User Classes and Characteristics**

1. News Reader:-

The system is developed for people of all genre and mainly for those who love to read news and blog post of specific topics without searching it from various sources.

1. Admin:-

Admin of this web app can delete any user, if find it fake and can delete any tag or add to block list if find it vulgar.

**2.3.4 Operating Environment**

Because the system is developed based on RESTful API concept, where host app will provide all back-end task which can be fired by HTTP requests. The system can support all kind of web API consumer interface like web client (Chrome, Firefox, Opera etc.) or Android app or IOS app etc.

Currently, there is support for web client (various browsers) only. Because of the front-end development of web client is responsive, it can adjust UI (User Interface) to any screen size.

**2.3.5 Design and Implementation Constraints**

The main constraint is that there is not every time news exists for rare tags, so when user enter some rare tags, he/she won’t get any news for a long time.

Some vulgar tags may not be identified by the system.

So admin has to manually add it to block list category, which is a time-consuming process.

**2.3.6 Assumptions and Dependencies**

The system may not update new news if the appropriate network is not available because it completely depends on fetching new news from API request.

The system requires the user to be online because web client can’t store that much of data as offline.

**2.4** **External Interface Requirements**

**2.4.1 User Interfaces**

The application can be accessed through any browser interface. The software will be fairly compatible with Microsoft Edge Version 40 and above on which it is tested or other modern web browsers as well.

**2.4.2 Hardware Interfaces**

**Server Side**:

Processor: Pentium 3.0 GHz or higher.

RAM: 2GB or higher.

Hard-Disk: 100GB or more.

**Client Side:**

Processor: Pentium 3.0 GHz or higher.

RAM: 2GB or higher.

Hard-Disk: 100GB or more.

**2.4.3 Software Interfaces**

**Client Side:**

HTML supported Web Browsers, Windows 7, 8, 8.1, 10, MAC OS, Linux (All Flavors).

**Server Side:**

Linux, PostgreSQL Client.

**2.4.4 Communication Interfaces**

The “News By Tag” will use the HTTP protocol for communication over the internet and intranet communication will be through TCP/IP protocol suite.

**2.5** **Functional Requirements**

**2.5.1 R1**: System provides a feature to add new tags to the account and deleting it from the account.

**Input**: Tag Details, Action

**Output**: Stored or deleted tag detail

**Processing**: Given tag name will be added to the database and will be sent for a queue of fetching news related to that tag and user will be redirected to that tag page also delete in case of that provided as action.

**2.5.2 R2**: System will provide functionality of autocompleting tag name while searching.

**Input**: Entered Character

**Output**: All matched tag name result

**Processing**: It will match all the tags with given character and return JSON object with a list of result.

**2.5.3 R3**: System will provide facility to show trending tags in order of number of users, subscribes to given tag.

**Input**: All tags details with user link

**Output**: Ordered list of trending tags

**Processing**: It will order the tags based on a number of users added that tag to their account.

**2.5.4 R4**: System will rank articles based on relevancy to tag name in a best possible way.

**Input**: Tag with corresponding articles

**Output**: Modified link order of article with the tag

**Processing**: Based on indexing algorithm it will sort the articles with relevancy level to that tag and modify that order.

**2.5.5 R5**: System provides a feature that prevent adding new profanity word as the tag name.

**Input**: Tag Details

**Output**: Status of profanity

**Processing**: It matches the list of profanity word and if matched then according to status appropriate message will be given.

**2.5.6 R6**: System provides the basic functionality of sign up, sign in and change the password.

**Input**: Credentials, action

**Output**: Status

**Processing**: It will take log in, sign in or changing password related action based on the input of credential and action.

**2.6 Non-Functional Requirements**

**2.6.1 R1**: GUI of web client should be mobile responsive

**Solution**: In order to provide mobile responsive, its GUI will be created in a mobile responsive framework such as Bootstrap.

**2.6.2 R2**: Page response should be fast because people want to read news quickly

**Solution** : To achieve this, I will keep back-end database as PostgreSQL, which is known for its faster processing.

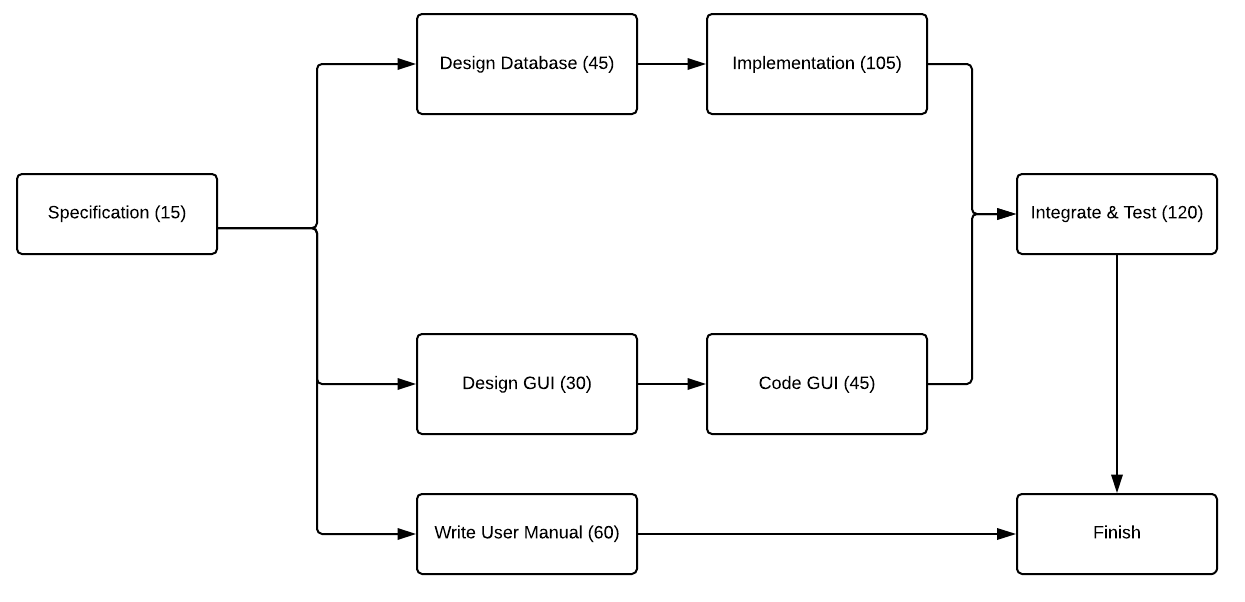
**2.6.3 R3**: Web app should handle a minimum of 1000 users simultaneously in the first release.

**Solution** : To achieve this, I will deploy it to heroku platform which provides this facility as free tier which can be extended in case of more traffic.

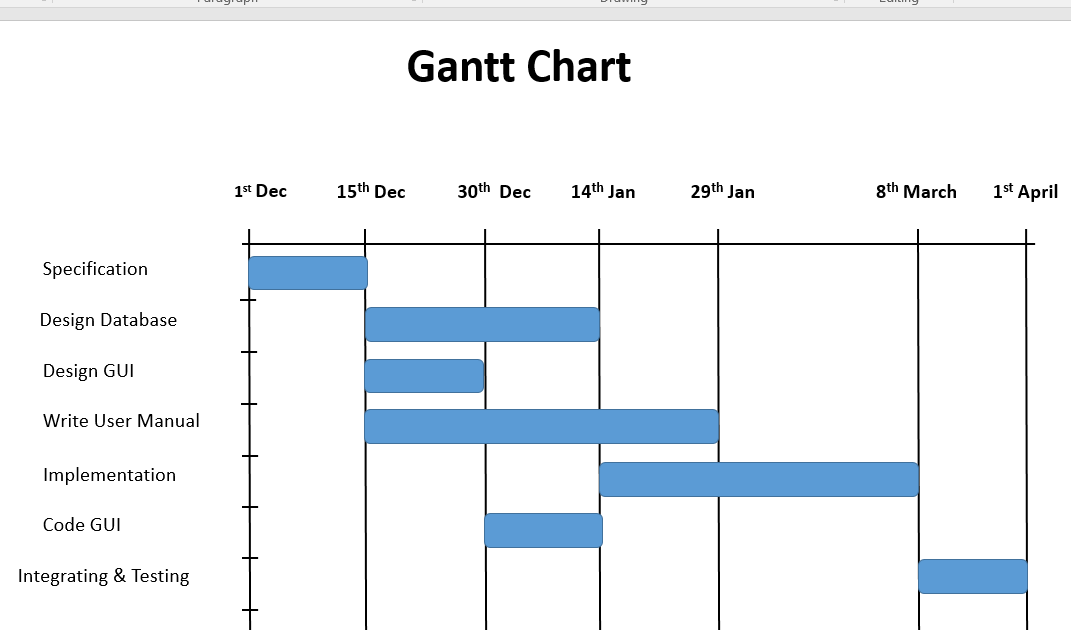
**CHAPTER: 3**

**DESIGN**

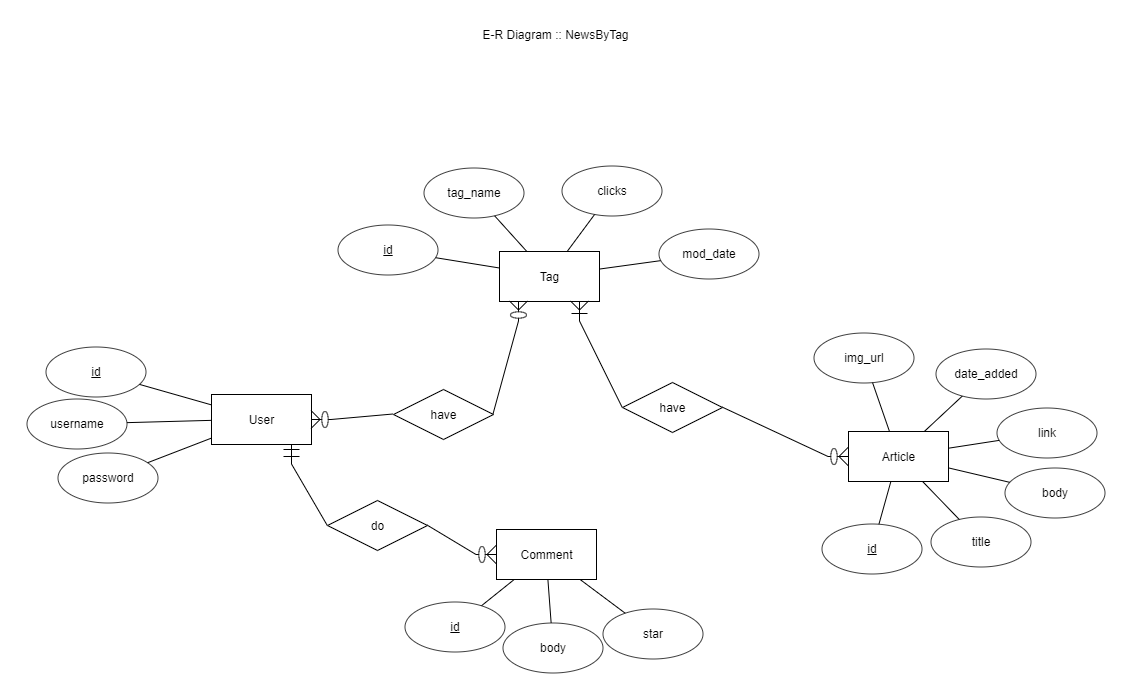
**3.1 Activity Network Diagram**

****

**3.2 Gantt Chart**

****

**3.3 E-R Diagram**

****

**3.4 Data Dictionary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User** | | | | |
| **Sr.no** | **Field\_name** | **Data\_type** | **Width** | **Constraint** |
| 1 | Id | Int |  | PRIMARY KEY  UNIQUE |
| 2 | Username | Varchar | 20 | NOT NULL |
| 3 | Password | Varchar | 20 | NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tag** | | | | |
| **Sr.no** | **Field\_name** | **Data\_type** | **Width** | **Constraint** |
| 1 | Id | Int |  | PRIMARY KEY  UNIQUE |
| 2 | Tag\_name | Varchar | 20 | NOT NULL |
| 3 | Clicks | Int |  | NOT NULL |
| 4 | Mod\_date | DateTime |  | NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User\_Tag** | | | | |
| **Sr.no** | **Field\_name** | **Data\_type** | **Width** | **Constraint** |
| 1 | User\_id | Int |  | NOT NULL  FOREIGN KEY |
| 2 | Tag\_id | Int |  | NOT NULL  FOREIGN KEY |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Article** | | | | |
| **Sr.no** | **Field\_name** | **Data\_type** | **Width** | **Constraint** |
| 1 | Id | Int |  | PRIMARY KEY  UNIQUE |
| 2 | Title | Varchar | 200 | NOT NULL |
| 3 | Body | Varchar | 1000 | NOT NULL |
| 4 | Img\_url | Varchar | 1000 | NOT NULL |
| 5 | Link | Varchar | 1000 | NOT NULL |
| 6 | Date\_added | DateTime |  | NOT NULL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tag\_Article** | | | | |
| **Sr.no** | **Field\_name** | **Data\_type** | **Width** | **Constraint** |
| 1 | Article\_id | Int |  | NOT NULL  FOREIGN KEY |
| 2 | Tag\_id | Int |  | NOT NULL  FOREIGN KEY |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Comment** | | | | |
| **Sr.no** | **Field\_name** | **Data\_type** | **Width** | **Constraint** |
| 1 | Id | Int |  | PRIMARY KEY  UNIQUE |
| 2 | Body | Varchar | 200 | NOT NULL |
| 3 | Star | Int |  | NOT NULL |
| 4 | User\_id | Int |  | NOT NULL  FOREIGN KEY |

**CHAPTER: 4**

**IMPLEMENTATION**

**4. Implementation**

**4.1 Modules Description**

* **Sign-Up / Log - In Module**

It is used to store information of users who are new to the system and accessing the system for the first time. It also authenticate the users before they can explore various auctions listed in the system.

**Input**: User’s information or credentials

**Output**: Stored or verified successfully

**Processing**: Check user’s credentials in the database while logging in or store them in the database while registering a new user.

* **Tag Module**

This module comes into action when a user wants to access or store tag related information.

**Input**: Username, Tag name, Action

**Output**: Status of Action in JSON

**Processing**: It will add, delete or modify data about tag as given in action parameter.

* **News Module**

It will fetch news from the database and return back all result articles of given tag.

**Input**: Tag name

**Output**: Status of Result and News in JSON

**Processing**: It will return all the articles related to given tag name and return back to JSON format.

* **Updater Module**

It will update new entered tags as well as renew already entered tags with new news by fetching news from API request and make the link between the tag and that article.

**Input**: Tag name

**Output**: Status of Result

**Processing**: It will link tag with the corresponding article fetched from API request and store it to the database.

* **Admin Control Module**

This module is used by the administrator of the system in order to maintain control of the present tag, user, and comment. By this admin can delete any tag, user and also add a tag to block list category to avoid next time addition of that tag to the system, if an admin finds it vulgar.

**CHAPTER: 5**

**TESTING**

1. **Testing**

**5.1 Testing Plan**

The testing is a technique that is going to be used in the project is black box testing, the expected inputs to the system are applied and only the outputs are checked.

**5.2 Testing Strategy**

The development process repeats this testing subprocess a number of the lines for the following phases.

* Unit Testing
* Integration Testing

Unit Testing tests a unit of code after coding of that unit is completed. Integration Testing tests whether the previous programs that make up a system, interface with each other as desired. System testing ensures that the system meets its stated design specifications. Acceptance testing is testing by users to ascertain whether the system developed is a correct implementation of the software requirements specification.

Testing is carried out in such a hierarchical manner to that each component is correct and the assembly/combination of the component is correct. Merely testing a whole system at the end would most likely throw up errors in a component that would be very costly to trace and fix. I have performed both Unit Testing and System Testing to detect and fix errors.

**5.3 Testing Methods**

I have performed Black-box testing for the testing purpose. A brief description is given below:

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied to virtually every level of software testing: unit, integration, system, and acceptance. It typically comprises most if not all higher level testing, but can also dominate unit testing as well.

**5.4 Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test,**  **Case ID** | **Test**  **Scenario** | **Test**  **Steps** | **Test**  **Data** | **Expected**  **Results** | **Pass**  **or**  **Fail** |
| T01 (2.5.6) | Sign Up  with proper data | 1.Provide right  Information on the signup page | User  Information | Stored  Successfully | pass |
| T02 (2.5.6) | Sign Up  with empty as well as wrong data | 1. In signup page, enter wrong details | User  information | Display a respective error message | pass |
| T03  (2.5.6) | Log In  with proper data | 1.Enter proper credential | Login Credentials | Display Homepage as logged in | pass |
| T04  (2.5.6) | Log In  with improper data | 1. In the login page, enter wrong details | Login Credentials | Display a respective error message | pass |
| T05 | Adding proper New Tag | 1. Enter proper tag name with keeping the limit in character | Tag Name | Adding a tag to the database and fetching new news | pass |
| T06  (2.5.5) | Adding Profanity word | 1.Add Improper profanity word | Profanity word as a tag name | Showing error message accordingly | pass |
| T07 | Tag Limit Per Account | 1.Add tags to limit in one account  2.Enter one more tag | Tag Limit per Account | Showing error message accordingly | pass |
| T08  (2.5.6) | Changing password with the wrong old password | 1. Enter the wrong old password | Password matching capability | Showing according to msg | pass |
| T09  (2.5.6) | Changing password with the right old password | 1.Enter the right old password | Password matching capability | The message of the changed password on the homepage | pass |
| T10 | Deleting a tag from the account | 1.Click Delete the icon from my tag section | Deleting tag | The tag should not appear that in my tag section | pass |
| T11 | Adding the tag to the block list | 1.Open Admin Panel  2.Click add icon near any tag | Adding a tag to block list | Tag name should be shown in the blacklist and should not be added next time in the system | pass |

**CHAPTER: 6**

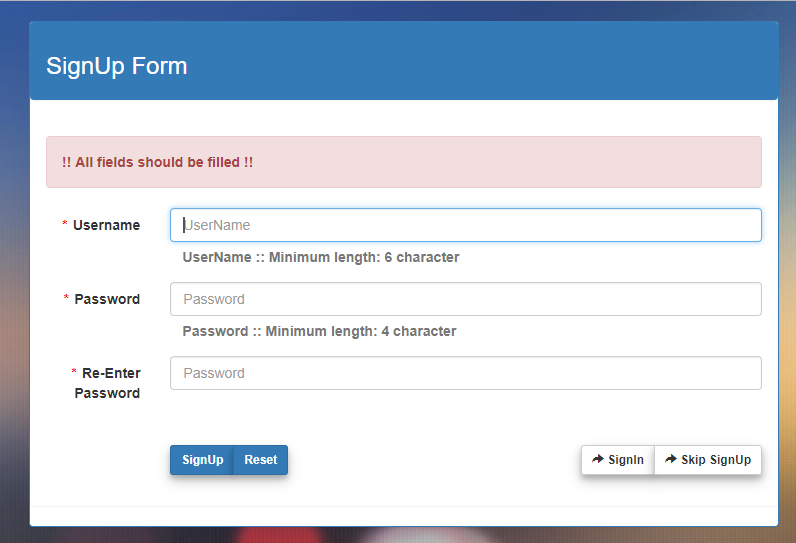
**SCREENSHOTS**

**6. Screen-shots**

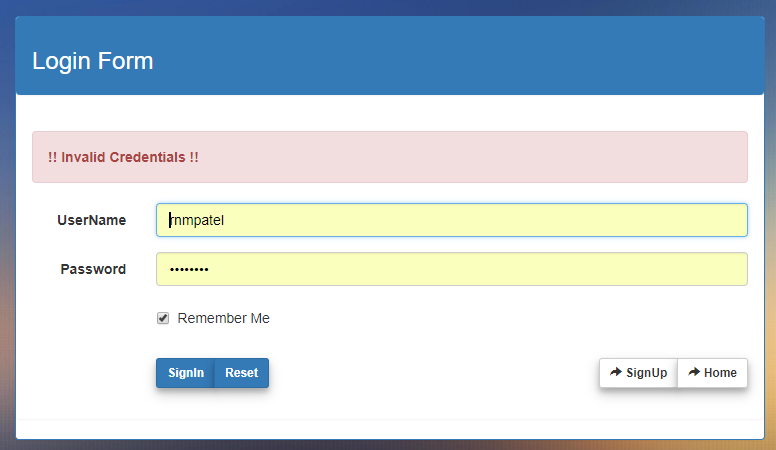
**6.1.1 - 6.1.4 Sign Up, Log In, Changing Password And**

**Feedback Page**

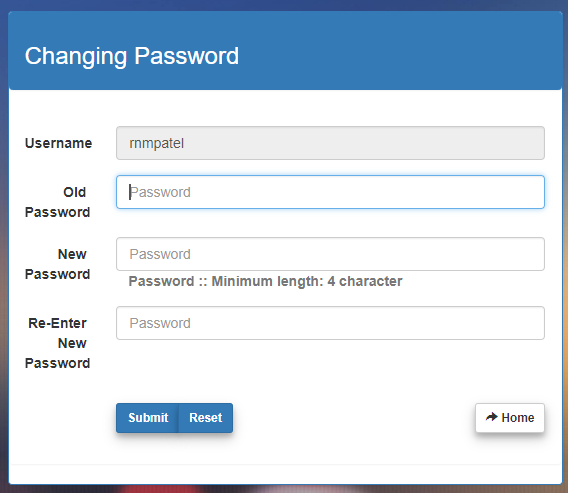
* **Sign-Up Page**
  + This sign up page will be starting page when some new user comes to the website.
  + The user can skip sign up page with button **Skip SignUp** or can go to sign in page by **SignIn** button.
  + All fields are necessary to be filled for completing the sign-up process, otherwise, an appropriate error message will be shown as in the following screenshot.
  + After successful sign-up, the user will be redirected to home page of the site.



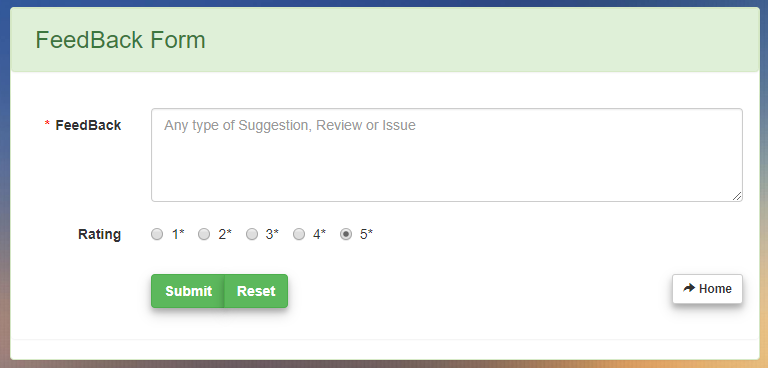
* **Login Page**
  + This login page will come after clicking on sign in button anywhere from the site.
  + All fields are necessary to be filled for logging process, otherwise, an appropriate error message will be shown as in the following screenshot.
  + After successful sign-up, the user will be redirected to home page of the site.



* **Changing Password Page**
  + This page will come after clicking on account button in home page after login.
  + All fields are necessary to be filled in correct way as shown in form with a right old password for changing the password, otherwise, an appropriate error message will be shown.
  + After successful sign-up, the user will be redirected to home page of the site.

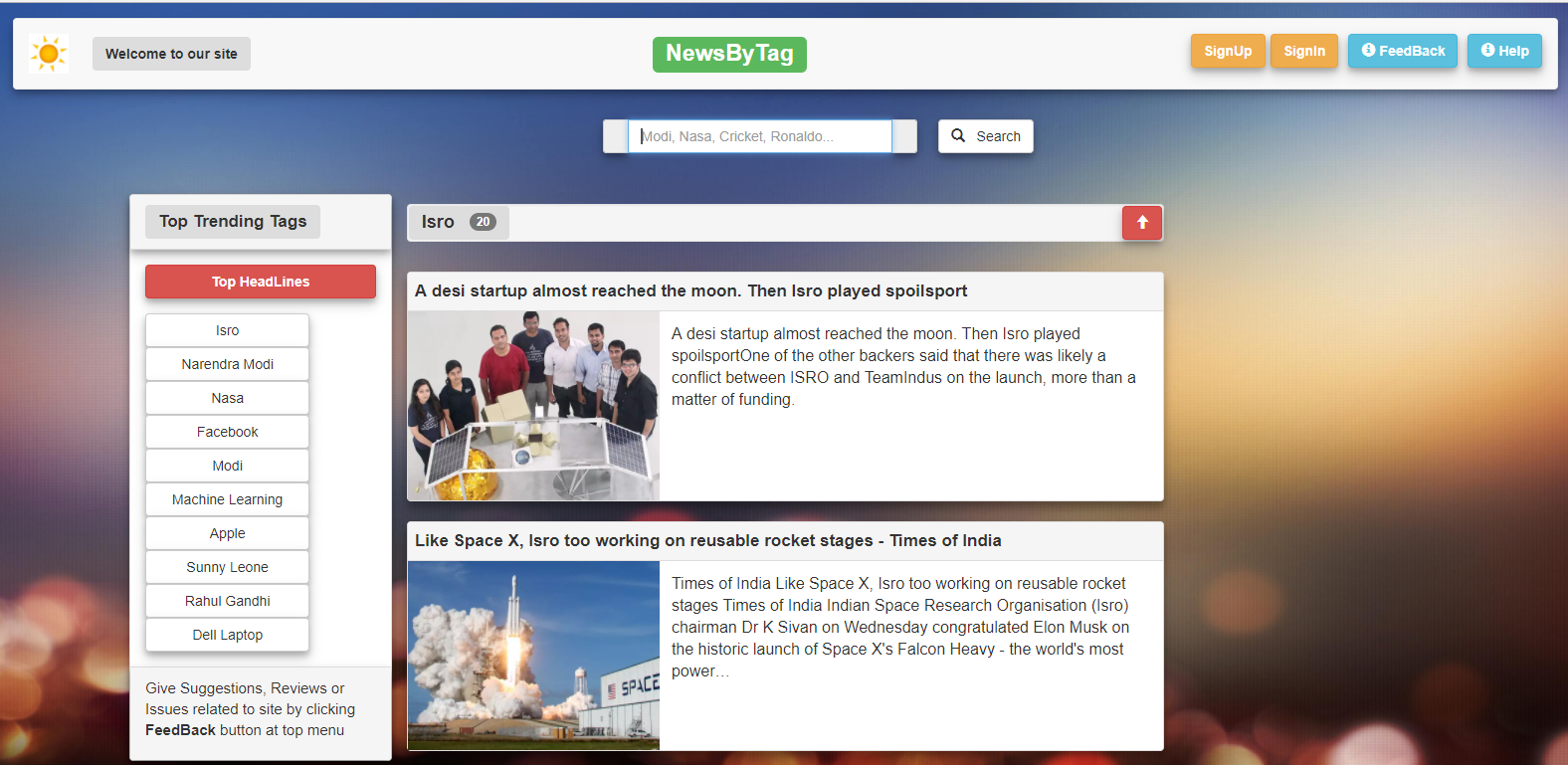
****

* **Feedback Page**
  + This page will come after clicking on **Feedback** button.
  + All fields are necessary to be filled in correct way as shown in form, otherwise, an appropriate error message will be shown.
  + After successful submission, the user will be redirected to home page of the site.

****

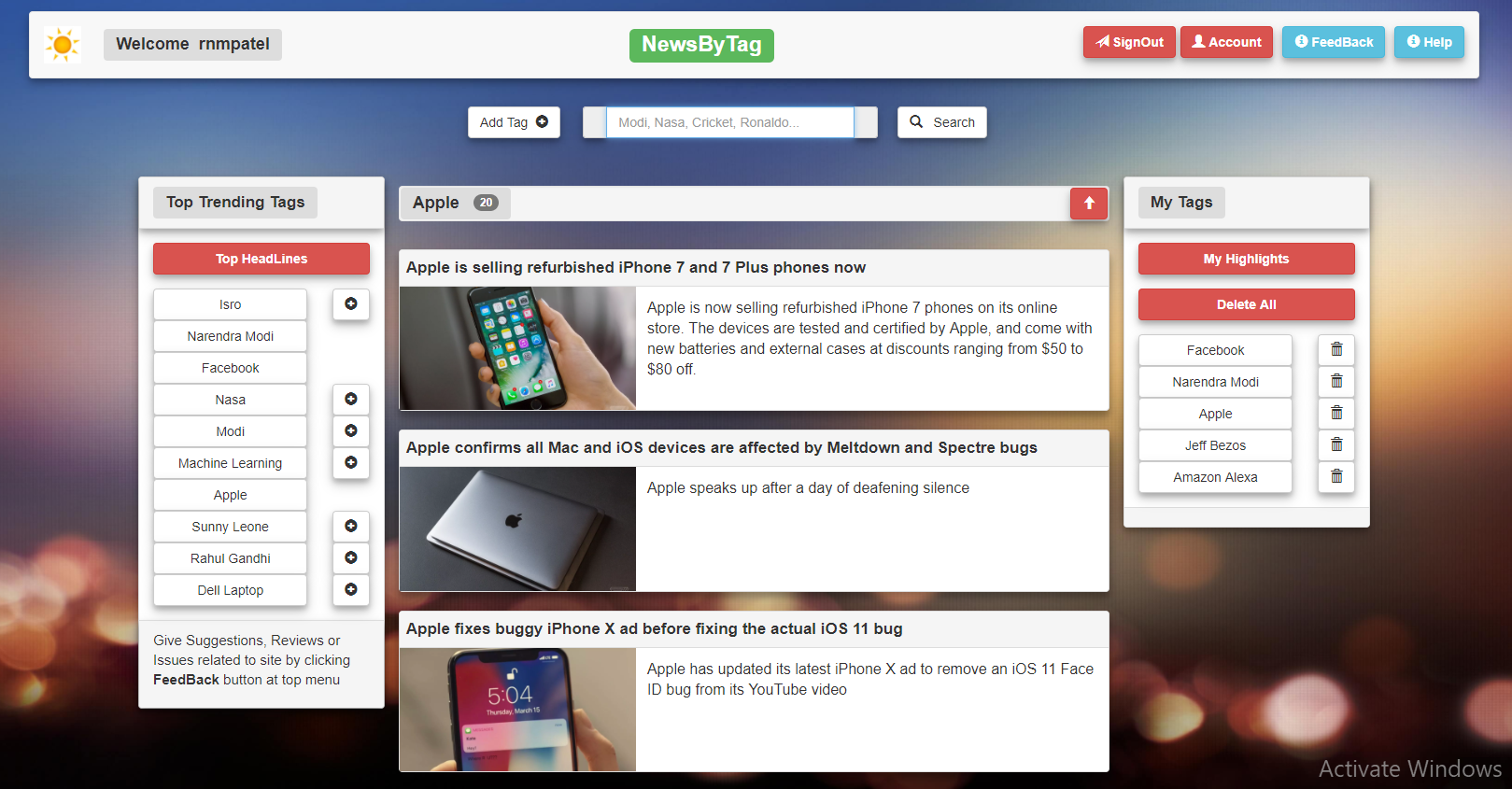
**6.2 Home Page before Sign-In**

* + This page will come after clicking on **Skip Sign up** or **Home** button anywhere from the site without login or signup.
  + The most trending tag will be shown by default and user can search tags (entered by another user) and can read the news of it.
  + The user can also see headlines overview from top trending tags by clicking on button **Top Headlines.**
  + The user can select any of tag from given **Top Trending Tags** section.
  + The user can go to feedback page, help page, sign up a page or sign in page any time by clicking appropriate buttons given in menu bar of this page.

****

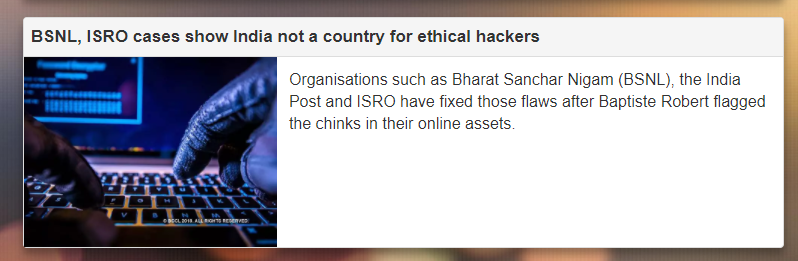
**6.3 Home Page After Sign-In or Sign-Up**

* + This page will come after successful **Sign-Up** or **Sign-In.**
  + Extra from non-signed-in or non-signed-up users, one can,
  + Add new tags in the system and account.
  + Delete tags from their account by clicking delete icon near to that tag and also can delete all tags by clicking on **Delete All** button.
  + See highlights of news respective to the tags in their account by clicking on **My Highlights** button.

****

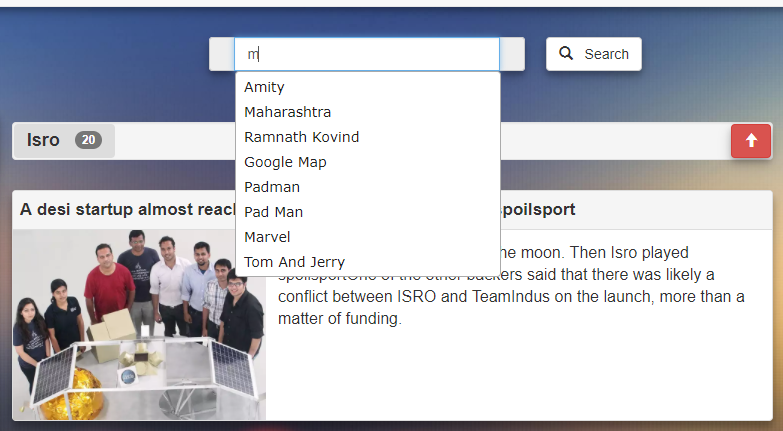
**6.4 Close News View**

* + This is a close view of how each news will be shown to the user.

****

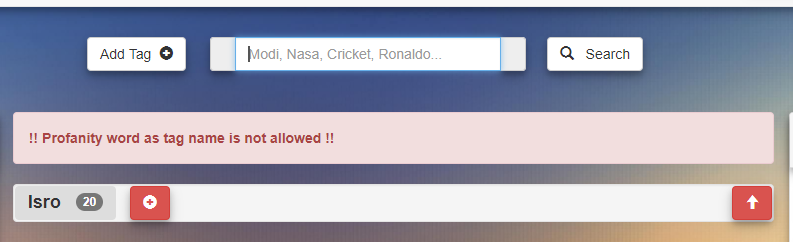
**6.5 Autocomplete Search**

* + This is a close view of how to autocomplete search will show matching tag names from database to given input.

****

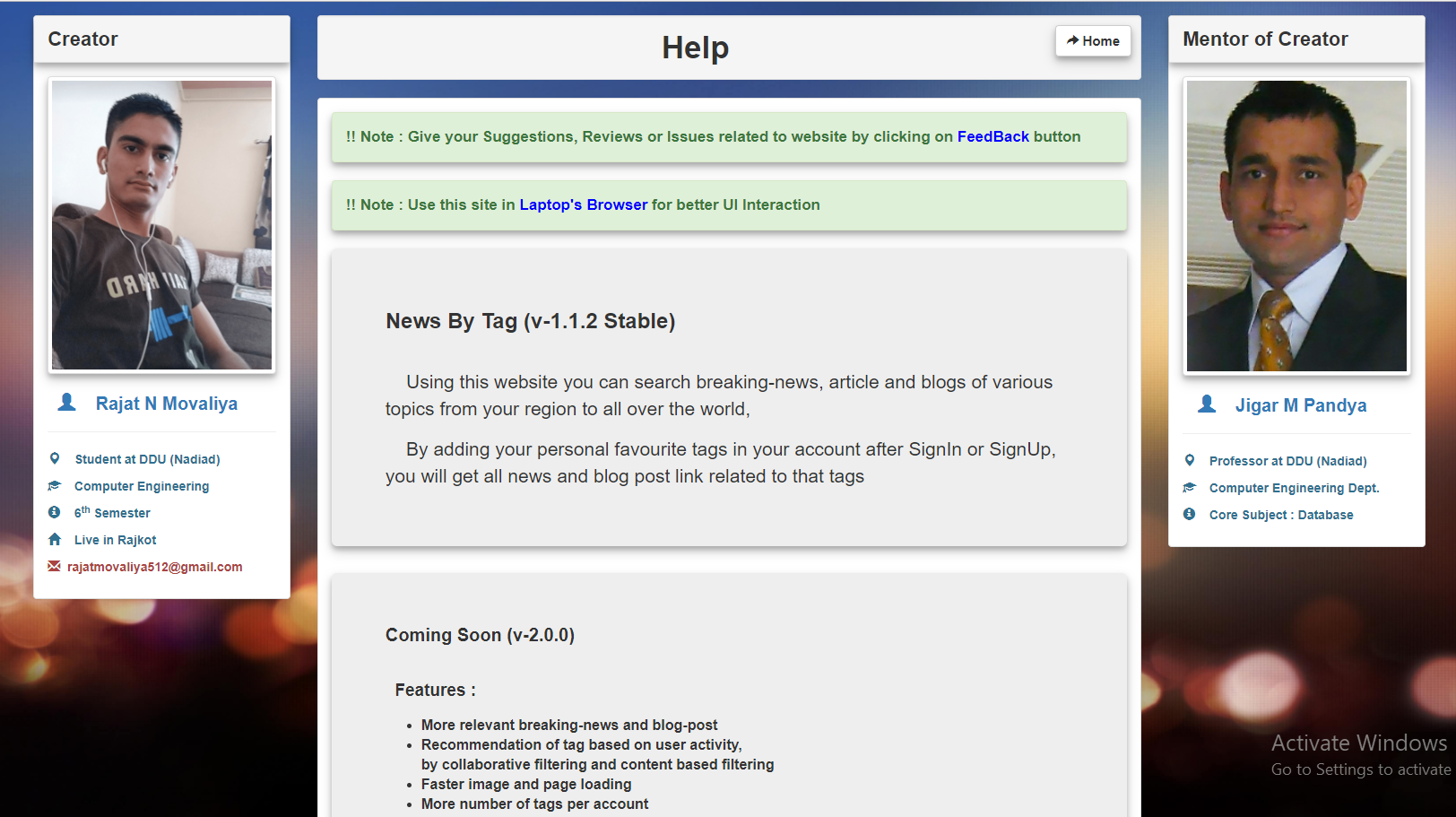
**6.6 Profanity word as a tag name**

* + This is a close view of how error message will be shown in case user try to enter the profanity word as a tag name.

****

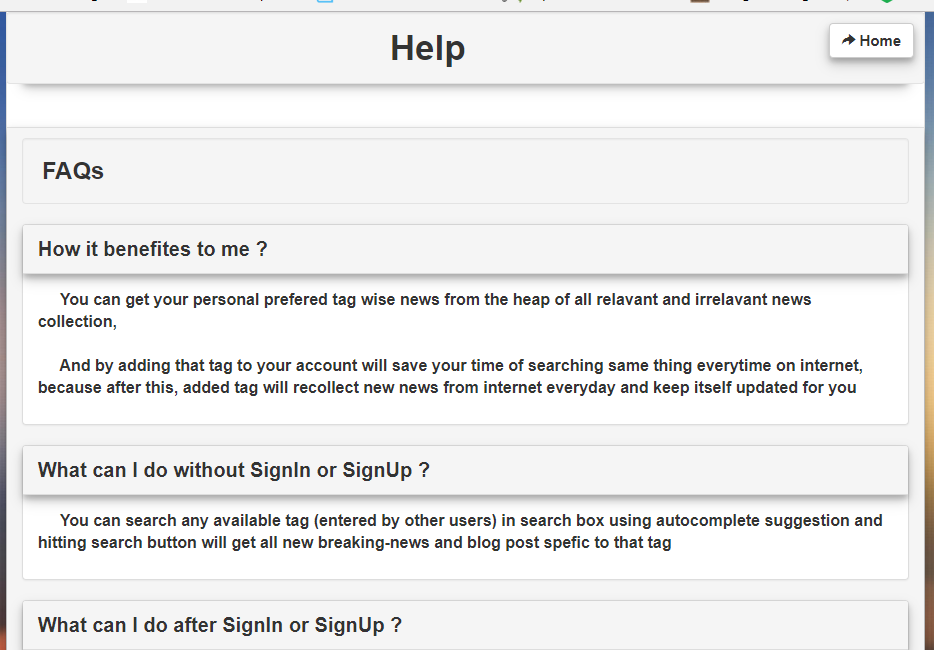
**6.7 Help Page Overview**

* + This is help page of the site, comes when the user clicks on **Help** button from home page.
  + It will show details of the creator of the site, mentor of the creator, site purpose, overview, current release version, future version, features, notes and FAQ (Frequently Asked Questions).

****

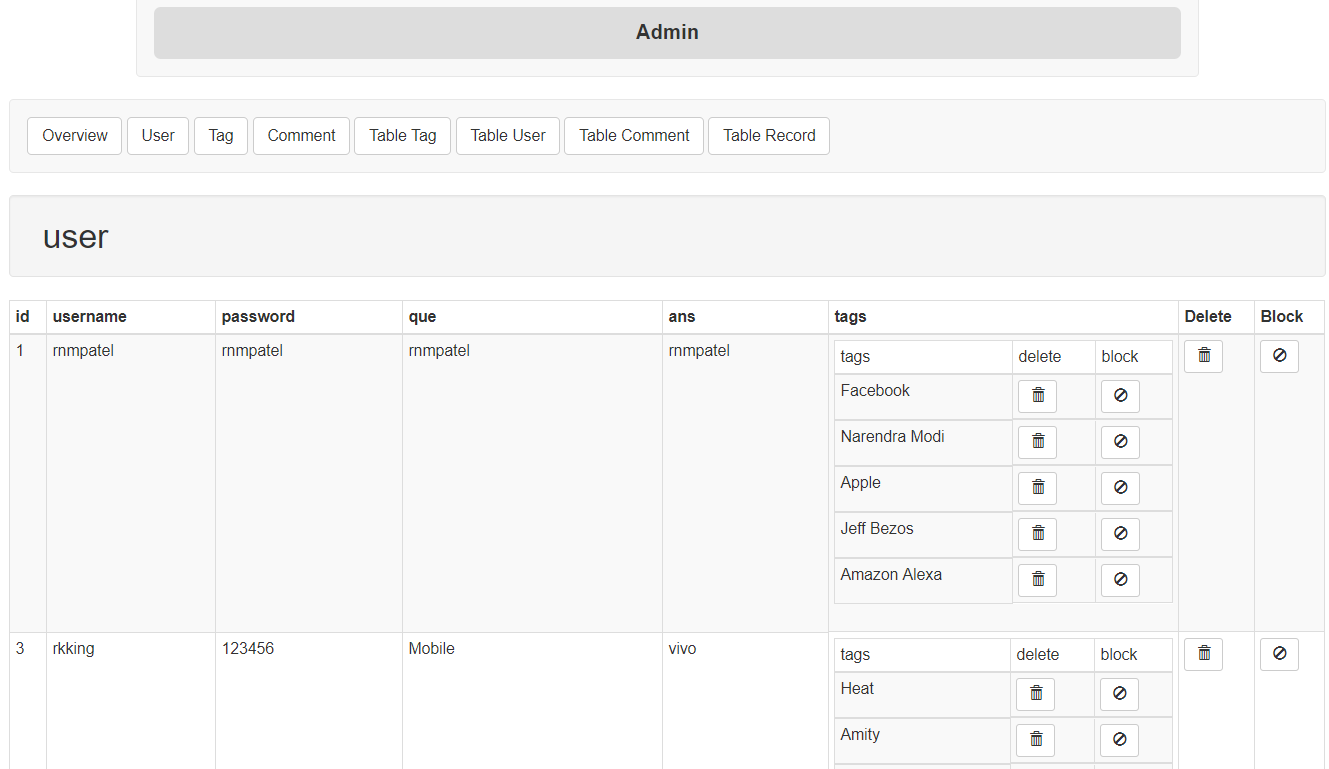
**6.8 Help Page Close view**

* + This is a close view of FAQs in help page of the site.

****

**6.9 Admin Panel**

* + After login to admin panel, admin can do many things like,
  + Can see how many users have registered with all details of user entered while sign up.
  + Can block any tag, which will be then added to block list and can’t be entered again in the database.
  + Can delete any tag from the system.
  + Can see all comments.
  + Can see all the tables currently in the database.

****

**CHAPTER: 7**

**Software Version Deployment Steps**

**7. Software Version Deployment Steps**

* **Steps :**

**Note:**

* Pre-Requirements : Linux Based OS (Recommended - Ubuntu)
* Run all the following command in Terminal, which will be opened by pressing (Ctrl + Alt + T).
* **Pre-Requirement :**
  + Install “pgadmin” from software center directly by searching its name.
  + Create “p8” database from GUI directly with option “create new database”.

1. Most of the Linux based OS comes python installed.

You can check this by typing

**python –V**

This will give you output as “python <version>” if getting this then fine otherwise run command

**sudo apt-get update**

**sudo apt-get –y upgrade**

This will upgrade your system with latest python version.

1. Go to the root of “Phost” project directory where app.py is resided and enter the following command which will change your python environment from your local computer to virtual environment given in project directory from where all required modules will be fetched in the current project.

**source ./venv/bin/activate**

In order to reverse the process, you can just type the following command.

**deactivate**

This will reset your python environment to local machine environment as was earlier.

1. After changing this virtual environment, run following command to start API hosting program.

**python3 app.py**

This will host web API hosting program in 5000 port with IP: 127.0.0.1 (localhost) with its own built-in development web server.

1. Do all same steps (2) and (3) from the root of the “Pconsume” project directory by opening new terminal window, because previous will be used for hosting program.

From whole repeated process only change will be that this consumer program will run on port number 5001 instead of 5000 and will be on same IP address that is 127.0.0.1 (localhost) with its own built-in development web server.

1. Now open any web browser available (Recommended: Google’s Chrome Browser)

and enter the following URL

**127.0.0.1:5001**

This will bring you to the home page of the project.

You can continue using it as flow direct you in pages.

**CHAPTER: 8**

**CONCLUSION**

**8.** **Conclusion**

All functionalities in the SRS are successfully implemented by me which are as follow.

* Showing breaking news of tags provided by the user.
* The user can add a tag and delete tag to the account.
* Autocomplete of tag name while searching new tags.
* Updating tags every day with new breaking news.
* User account management like sign-up, log-in etc.
* Preventing user from adding profanity word as a tag name.
* Showing trending tag based on the number of users subscribes to respective tag.
* Admin can add any vulgar tag to block-list, from which system will prevent next time from adding that tag to the database.
* System’s GUI is responsive to any screen size.
* The system will automatically remove a long time unused tags from the database, where time will be set by the admin.
* The system won’t update tags which are not used so long, where time will be set by the admin.

Hereby, I declare that the functionality implemented in this system was performed by understanding all the modules.

This project’s main aim is to provide breaking news by tag names provided by the users which were fulfilled to a greater extent. All the diagrams were prepared before implementation and the coding was done accordingly. After the coding was completed, comprehensive testing was performed and the results were provided in the report. Unit testing of all modules was done and later, Integration Testing was also performed.

**CHAPTER: 9**

**LIMITATIONS**

**AND**

**FUTURE ENHANCEMENT**

**9. Limitations and Future Enhancements**

* **Limitations**

1. Admin needs to manually add vulgar tags entered by the user to block list, in case system does not recognize it.
2. Many rare tags do not have, that frequent new breaking news, so the user has to wait for a long time to news updates for that tag.

* **Future Enhancements**

1. Recommendations of tags based on user activity and also bycollaborative filtering and content-based filtering.
2. More and more, tag relevant news and blog post.
3. Android App can be made because it makes a lot easier for the user to use with offline news storage facility and it will reduce the number of steps to open news.
4. Separate favorite news storage facility.
5. News in other than the English language as well.
6. Filtration of news based on post dates, the word in body and title, and much more.
7. Some fastest process of fetching new tag news in API request.
8. Loading images and pages faster by CDN (Content Delivery Network) and by providing the various size of thumbnail versions of images.
9. Adding SSL (Secure Socket Layer) based communication to add more security to credentials.

**CHAPTER: 10**

**BIBLIOGRAPHY**

**10. Bibliography**

**Web Tutorials:**

* <http://docs.sqlalchemy.org>
* <http://flask.pocoo.org/docs/0.12/tutorial>
* https://flask-migrate.readthedocs.io/en/latest/
* <https://www.tutorialspoint.com/python/index.htm>
* <https://www.tutorialspoint.com/postgresql/index.htm>
* <http://www.vogella.com/tutorials/Git/article.html>
* <https://devcenter.heroku.com/categories/reference>
* <http://jinja.pocoo.org/>

**CHAPTER: 11**

**ABOUT**

**11. About**

* **Project Name : NewsByTag**
* **Released Version : 1.1.2 (stable)**
* **Released Date : 1st April 2018**
* **Deployed Link :** [**http://newsbytag.herokuapp.com**](http://newsbytag.herokuapp.com)
* **-----------------------------------------------------------------------**
* **Note:** Site has been bounded with Google Analytics.

After sharing link within DDU students of Computer Branch.

Following is report after two days.

* + It got around **6500 page requests**.
  + Around **300 users**.
  + Almost **600 session**.
* **Creator of web app :**

**Name :** Rajat Nathabhai Movaliya (Student)

**Branch :** Computer Engineering

**Sem. :** 6th

**Email :** rajatmovaliya512@gmail.com

* **Mentor of creator :**

**Name :** Jigar M. Pandya (Assistant Professor)

**Department :** Computer Engineering