

Student Resource Recommender

FINAL YEAR PROJECT

Submitted by

SRIKAR AMARA (9916004010)

in partial fulfilment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING



SCHOOL OF COMPUTING

COMPUTER SCIENCE AND ENGINEERING

KALASALINGAM ACADEMY OF RESEARCH AND EDUCATION

KRISHNANKOIL 626 126

Academic Year 2019-2020



KALASALINGAM

ACADEMY OF RESEARCH & EDUCATION

(DEEMED TO BE UNIVERSITY)

Under sec. 3 of UGC Act 1956. Accredited by NAAC with "A" Grade



School of Computing
Department of Computer Science and Engineering
Project Summary

Project Title	Student Resource Recommender	
Project Team Members (NamewithRegisterNo)	SRIKARAMARA	9916004010
Guide Name/Designation	Mr. R. RAJA SUBRAMANINAN , Assistant Professor, Department of Computer Science and Engineering	
Program Concentration Area	RECOMMENDER SYSTEMS	
Technical Requirements	Scrapy, Django framework, Tensorflow, LSTM, python.	
Engineering standards and realistic constraints in these areas		
Area	Codes & Standards / Realistic Constraints	Tick ✓
Economic		
Environmental		
Social	This project is aimed at many communities including students, research scholars and faculties. It establishes a social network platform among the knowledge seekers.	✓
Ethical		
Health and Safety		
Manufacturability		
Sustainability	This project uses open source software including Python Django framework and Scrapy. The core part being recommending users of their interest sources, the project finds its place in every student/ researcher online presence. Hence the project is sustainable.	✓

Realistic constraints:**Sustainability:**

In recent years, many state-of-the-art applications are leveraging recommender systems and its allied sources in application development. The project proposes a novel algorithm to provide useful recommendations pertaining to competitive examinations to students and faculties. As every higher studies program is subjected to a competitive examination clearance, many aspirants are struggling without a mentoring application. Hence an application 'Student Resource Recommender' is developed to counter the problem. In addition, the core part of the application is developed on top of Python frameworks, that are open source. Thus making the application easily available, usable and sustainable.

Social:

Student Resource Recommender provides a platform to unite competitive examination aspirants of similar interests. They are correlated in various aspects with respect to their knowledge, interests, domain of study and web presence. The problem of a community of people, who are not able to afford for the training classes fees, need to be resolved through Student Resource Recommender. Hence the application takes Social entity as one of the major constraint.

Engineering standards:

This project complies to IEEE 610.2-1987 - IEEE Standard Glossary of Computer Applications Terminology

This IEEE standard identifies and provides definitions for computed aided designs, character inference, computer assisted instruction, writer inference tasks, language processing, data processing and analysis among others. The standard typically applies to automation tasks and learning techniques. Language processing and recommendations are the two core modules of Student Resource Recommender. These modules are implemented in the application subject to IEEE 610.2-1987 - IEEE Standard Glossary of Computer Applications Terminology.

DECLARATION

We affirm that the project work titled “**Student Resource Recommender**” being submitted in partial fulfilment for the award of the degree of Bachelor of Technology in Computer Science and Engineering is the original work carried out by us. It has not formed the part of any other project work submitted for award of any degree or diploma, either in this or any other University.

SRIKAR AMARA

(9916004010)



KALASALINGAM
ACADEMY OF RESEARCH & EDUCATION
(DEEMED TO BE UNIVERSITY)

Under sec. 3 of UGC Act 1956. Accredited by NAAC with "A" Grade



BONAFIDE CERTIFICATE

Certified that this project report "STUDENT RESOURCE RECOMMENDER" is the bonafide work of SRIKAR AMARA (9916004010), who carried out the projectwork.

Mr. R.Raja Subramanian,
(Supervisor)
Assistant Professor
Department of Computer Science and
Engineering,
Kalasalingam Academy of Research
and Education
Krishnankoil 626126

Dr. A. FRANCIS SAVIOUR DEVARAJ
Head of the Department
Department of Computer Science and
Engineering,
Kalasalingam Academy of Research
and Education
Krishnankoil 626126

Submitted for the Project Viva-voce examination held on.....

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

First and foremost, I wish to thank the **Almighty God** for his grace and benediction to complete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear **Parents** and affectionate **Family members** for their honest support for the completion of this Project work.

I express deep sense of gratitude to “Kalvivallal” Thiru. **T. Kalasalingam** B.com., Founder Chairman, “Ilayavallal” **Dr.K.Sridharan** Ph.D., Chancellor, **Dr.S.Shasi Anand**, Ph.D., Vice President (Academic) , **Mr.S.Arjun Kalasalingam** M.S., Vice President (Administration) , **Dr.R.Nagaraj**, Vice-Chancellor, **Dr.V.Vasudevan** Ph.D., Registrar , **Dr.P.Deepalakshmi** M.E., Ph.D., Dean (School of Computing) . And also a special thanks to **Dr.A. FRANCIS SAVIOURDEVARAJ**. Head , Department of CSE, Kalasalingam Academy of Research and Education for granting the permission and providing necessary facilities to carry out Project work.

I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor **Mr R. Raja Subramanian**, Assistant Professor/CSE, of Kalasalingam Academy of Research and Education (KARE) for his inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under my Guide, who truly practices and appreciates deep thinking. I will be forever indebted to my Guide for all the time he has spent with me in discussions. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on.

Besides my Project guide, I would like to thank the rest of Class committee members and all faculty members and Non-Teaching staff for their insightful comments and encouragement. Finally, but by no means least, thanks go to all my school and college teachers, well wishers, friends for almost unbelievable support.

ABSTRACT

Recommender systems aim to present the relevant data to the user. The key problem of the recommender systems or any interest prediction algorithms is to maintain the dataset in a specific format. In recommender system-based algorithms, data are obtained on the fly, while the user is using the web application. The user usage logs acts as the preliminary input dataset. Formatting the dataset in an efficient scheme, aids efficient recommendations in time and accuracy perspectives. Finding relevant interest for recommendation is complex and time taking process and wide researches are done in the areas of user profile modeling and data formatting. Many state-of-the-art algorithms is subject to timing constraints. Some algorithms have a trade-off between time and accuracy. To address this problem, we proposed a method to format the data in the dataset using POS-taggers with NLTK framework. We have proposed a user-profile model, that leverages tagging mechanism to provide accurate results. Thus, increasing the effectiveness in recommendation for personalized applications.

Our aim is to create a personal and sophisticated recommender system for students that meets all their academic needs. We have created a web application that consists of modules including News recommender, Quiz Application, Friend requests and Future planner for competitive exams. To achieve this, we have proposed a new approach for the user profile modeling and the data categorization. A hybrid algorithm collaborating content-based and personalized recommender systems is proposed. The empirical evaluation of the proposed algorithm, T-UP-Tree, depicts that it performs better than the state-of-the-art recommender algorithms.

CONTENTS

ABSTRACT	VII
LISTOFFIGURES.....	XI
LISTOFABBREVIATIONS	XIII
1 INTRODUCTION	1
1.1 OVERVIEW	
1.2 STATE OF THE ART APPROACHES	
2 LITERATURE REVIEW.....	3
2.1 PERSONALISED RECOMMENDER SYSTEMS	
2.2 CONTENT BASED RECOMMENDER SYSTEMS	
2.3 HYBRID RECOMMENDER SYSTEMS	
2.4 SCRAPY	
2.5 NLTK	
2.6 DJANGO	
3 PROBLEM DEFINITION AND BACKGROUND	6
3.1 PROBLEM DEFINITION	
3.2 PROBLEM FORMULATION	
4 REQUIREMENT ANALYSIS AND SPECIFICATION	8
4.1 REQUIREMENT DESCRIPTION	
4.2 HARDWARE SPECIFICATION (SERVER SIDE)	
4.3 SOFTWARE SPECIFICATION (SERVER SIDE)	

4.4 SOFTWARE SPECIFICATION (CLIENT SIDE)	
5 SYSTEM DESIGN.....	9
5.1 SYSTEM DESIGN	
5.2 DESIGN CONSTRAINTS AND STANDARDS	
5.3 DESIGN ALTERNATIVES	
5.3.1 T-UP-TREE ALGORITHM	
5.3.2 MATRIX FACTORISATION	
5.3.3 DJANGO AND UI	
5.3.4 RECOMMENDER ALGORITHMS	
6 PROPOSED APPROACH	17
6.1 DYNAMIC MODEL OF USER PROFILE	
6.2 CONSTRUCTION OF TAGGER-USER PROFILE-TREE	
6.3 TAGGER-USER-PROFILE-TREE ALGORITHM	
6.4 FRIEND RECOMMENDATION ALGORITHM	
7 IMPLEMENTATION AND RESULTS.....	22
7.1 IMPLEMENTATION AND RESULTS	
7.2 TESTING	
7.2.1 DATA SET CATEGORISATION	
7.2.2 PERFORMANCE TESTING	

8 USER INTERFACE27

 8.1 DJANGO USER INTERFACE

 8.2 UI FOR LOGIN

 8.3 UI FOR REGISTRATION

 8.4 NEWS RECOMMENDER INTERFACE

 8.5 UI FOR HOME PAGE

 8.6 UI FOR NOTES

 8.7 UI FOR SHOP MODULE

 8.8 UI FOR QUIZ APP MODULE

 8.9 USER PROFILE

 8.10 ADMIN PROFILE

CONCLUSION AND FUTURE SCOPE 35

REFERENCES 36

PUBLICATION 38

LIST OF FIGURES

2.6.1 POS TAGGER.....	4
5.1 USE CASE DIAGRAM FOR SRR.....	10
5.2 ENTITY RELATIONSHIP DIAGRAM FOR SRR.....	11
5.3 SEQUENCE DIAGRAM FOR SRR.....	12
5.4 COMPONENT DIAGRAM FOR SRR.....	13
5.5 WORK-FLOW CHART.....	14
5.6 FLOW CHART DEPICTING THE RECOMMENDATION MODULE FOR THE USER PROFILE.....	15
6.2.1 SYSTEM ARCHITECTURE FOR SRR.....	20
7.1.1 STATISTICS FOR THE USER PROFILE.....	25
7.2.1 T-SNE VISUALISATION OF THE SAMPLE DATASET.....	26
7.2.2 PRECISION GRAPH FOR DATA RETRIEVAL FOR RECOMMENDATIONS.....	27
7.2.3 COMPARISON OF T-UP-TREE AND MUSIC TAGGER ALGORITHM OVER A PERIOD OF TIME.....	28
7.2.4 CPU LOAD DATA ON THE SERVER SIDE.....	28
8.1 LOGIN PAGE.....	31
8.2 SIGN-UP PAGE.....	31
8.3 NEWS-ARTICLES.....	32
8.4 HOME.HTML.....	33
8.5 NOTES.....	33
8.6 SHOP.....	34
8.7 ADDING TO CART.....	34
8.8 SHOPPING CART.....	34
8.9 CHECK OUT.....	34
8.10 LIST OF QUIZES.....	35
8.11 PROGRESS.....	35
8.12 USER PROFILE.....	36
8.13 ADMIN_LAYOUT.....	36

8.14 NOTES_DATA.....	37
8.15 ORDER_DATA.....	37
8.16 DATA SETS OF NEWS	37
8.17 VIEW OF CATEGORISED DATA.....	38

LIST OF ABBREVIATIONS

Abbreviation	Full Form
LDA	LATENT DIRICHLET ALLOCATION
NLTK	NATURAL LANGUAGE TOOL KIT
POS	PARTS OF SPEECH
UP-TREE	USER PROFILE TREE
CNN	CONVOLUTIONAL NEURAL NETWORKS
KNN	K-NEAREST-NEIGHBORS

LIST OF ACADEMIC REFERENCE COURSES

1. CSE318 - Computer Networks.....	5
2. CSE401 - Object Oriented Software Development.....	12
3. CSE303 - Software Engineering.....	13
4. CSE103 - Data Structures.....	20
5. CSE209 - Algorithms & Complexity.....	20
6. MAT222 -Probability And Statistics.....	26
7. CSE402 - Internet Programming.....	27
8. CSE305 - Database Management Systems.....	34

CHAPTER – 1

INTRODUCTION

1.1 Overview

In modern times the recommender systems are playing a major role in people's lives and academic research. Usually, the Recommender system is an information filtering system that filters the relevant information based on the user preferences or the activity of the user. Statistics shown that there has been a drastic change of sales in the multinational corporate companies such as Amazon, Flipkart and other sales Companies, owing to recommender system inclusions. There is a saying called “You do not know what you want, until someone shows it to you”. Recommender System applications takes the statement as base in its applications.

The recommendation system is a tool which provides relevant data to its users based on the usage of other users with similar characteristics. It is often taken from the crude data provided by the user. This crude data might be some sort of the user survey or the category of the user or it could be the tracked from logs of the users in social networks. We use recommender system primarily to filter the articles pertaining to competitive examinations, based on the interests of the user. This will help the user to access more similar topics based on their interests. There are various categories of recommender systems namely: personalized, collaborative, context-aware, hybrid recommender systems.

In this project, we have created a student resource recommendation-based web application that uses these hybrid recommendation algorithms. These recommendation systems help the user better optimize the readability and search processes. There are many modules which helps us achieve better optimization of search results: data categorizaation and data filtering.

1.2 State-of-the-Art Techniques:

PRemiSE [12] is a state-of-the-art and personalized news recommendation framework by implicit Social Experts. It is a framework that leverages implicit feedback data and provide recommendations in the field of news. **LOGO** [12] is a framework, in which the long-term and short-term reading preferences of users are seamlessly integrated, for recommending news items to individual online users. **Content-based** [6] method uses user profiles or item descriptions to make recommendations. In **Hybrid-based** [7] methods, the goal is to combine the best of content-based recommenders with respect to a context, thus providing robust recommender systems. Our work is essentially a hybrid recommendation approach. Research on recommendation systems [16] proposed a

community-based user recommendation model, namely CB-MF, that uses LDA for clustering users into communities to enhance the existing MF-based user recommendations.

Marco and Siva [17, 45] adopted topic models at the user level where documents are replaced by users' streams and recommend users who have a distribution highly similar to a target user. **Interest-Based Recommendation (IBR)** [13] is a technique that uses interactive agent-based argumentation to generate interesting recommendations. State-of-the-art researches [3,4,8] supports the claim that argumentation is useful in various kinds of user support systems like expert systems, systems for automated negotiation, recommender systems etc.

Music tagger algorithm [14,15] is used to associate tags to the music playlist and the recommendation is done based on user invocations, to provide similar music tags. Recommending videos without visibility into the accounts or groups is challenging [1, 2]. Wang et al. [3] tackle the problem by introducing virtual users who represent activities from a single account in a sub-period. Each account is split into virtual users and then similar virtual users are merged to identify persona. They assume that different persona in an account have different viewing patterns according to time of the day. In [4], Yang et al. also identify persona based on the temporal viewing pattern of different genre. Similar viewing patterns in a single time interval are attributed to the same user.

CHAPTER – 2

LITERATURE REVIEW:

2.1 PERSONALISED RECOMMENDER SYSTEMS:

Personalized recommender systems are a type of recommender systems which focuses on the user profile modeling, In this recommender system, the recommendations usually happen by the user's activity. Since it's a personalized recommender, the recommendations differ from user to user. The news modules of our application consist of personalized recommender framework. It usually recommends the articles based on the number of hits on the articles that the user took or the users of similar interests took.

2.2 CONTENT BASED RECOMMENDER SYSTEMS:

Unlike the personalized recommender systems, the content-based recommender systems usually use the crude data given to the user profile while signing up (or) by default user selection such as Pinterest. Hence there is the faster retrieval of data in the recommendation systems but cannot be further processed. The content-based recommender system is used in the exam module in our project.

2.3 HYBRID RECOMMENDER SYSTEMS:

It is a modular approach in recommender systems, which includes two or more recommender systems. Common hybrid recommender systems include content + personalised recommenders for user profiling, which can be visualized in many real-time shopping websites such as Amazon.

2.4 SCRAPY: Scrapy is an open-source python framework which is used to scrape data from the web pages in a fast, simple and yet extensible way. This framework is used in the retrieval of the data. This framework is used to scrape the visible content of the web pages such as articles, headings, URLs. Scrapy is fast, versatile and efficient. There are other data scrapers available in the market such as selenium, import.io, beautifulsoup, kimono, apifier. The reason that the scrapy is used in this application is because of its reliability. It has many features that help scrape the data and store the scraped data in the database simultaneously. It is robust compared to other scrapers. Scrapy is a complete framework which has the tools for managing every stage of a web crawl such as :

- Request manager
- Selector

- Pipelines

Request manager is basically in-charge of downloading pages and the great bit about it is, it does all concurrently behind the scenes. Selector is used to parse the html document to find the bits required. Pipelines are used pass the retrieved data through various functions that acts on it. The usage of the scrapy has a key role in this application. As it integrates with Django very well it is used to scrape the data from web pages of news articles.

2.5 NLTK: Natural language toolkit (NLTK) is a platform that works with the aid of human language data providing features like sentence tokenizer and POS tagger [5]. Python has an initial tokenizer but NLTK is more reliable. It is a platform used for building python programs that will work for human language data, applying in statistical natural language processing (NLP). It will tokenize the sentences as {'this', 'is', 'a', 'token'}. The list of taggers is as follows CC: coordinating, FW: Foreign word, NN: Noun.

So, this NLTK is used to format the data using POS tagger and that formatted data is used as the source of input to recommendation algorithms. The example for the formatting of the sentence is shown in the Fig-2.6.1.

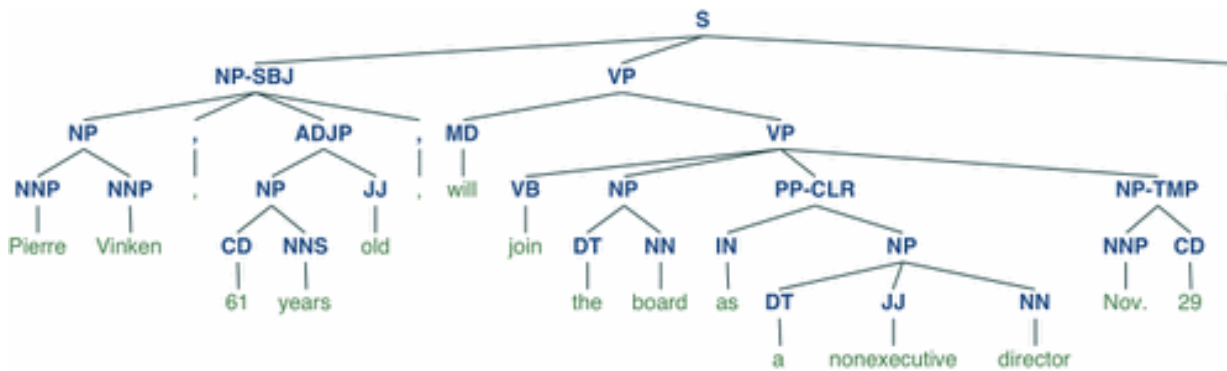


Figure -2.6.1 POS Tagger

This NLTK is used in the initial stage of data categorization and tokenization in this project. When the data is scraped, it is stored in the database. The semi-structured data is subjected to the NLTK framework and the data is categorized and tokenized accordingly. So, the final dataset is sent for processing in recommendation engine with the Django framework.

2.6 DJANGO:

Django is a high-level python web framework that encourages rapid development and clean, pragmatic design. This framework consists of many elements which helps the user not to worry about the basic parts and focus on writing the application without any hassle. There are many alternatives for Django framework in python but what makes more unique in Django, is that, it is more secure and it speeds up the development process. **Security:** Django includes prevention of common attacks like CSRF (cross-site request forgery) and SQL injections. It is very useful for building large web applications. Many software companies use Django as their go-to application. One of such companies is Instagram.

Note: Security is essential for any application. Django provides the inbuilt security that includes prevention of common attacks like CSRF (cross-site request forgery) and SQL injections. We've learnt the security topics in Computer Networks (CSE-318)

CHAPTER-3

PROBLEM DEFINITION AND FORMULATION

3.1 PROBLEM DEFINITION:

From the beginning of the recommendation systems the data retrieval and categorisation has been hard. Since most of the approaches mainly concentrates on the recommendation mainly based on the user's performance or activity, we tried to solve the following set of problems in the basic recommenders.

- Data Categorisation
- Data Retrieval
- User-profile modelling

3.2 PROBLEM FORMULATION:

Given a set of data which is unorganized and uncategorized, the first approach is to model the data set by categorising them. And to do that, we have used NLTK Framework. It is a leading platform for building python programs to work with human language data. It has been called as the best tool for computational linguistics in python. By using this toolkit we categorize the data using POS taggers and verb taggers. Hence the data can be easily and efficiently observed. The data categorization is taken place by the custom NLTK program which we have designed to generate tags and allocate them to the text data (News data) retrieved by the scrapy.

After categorization and allocation of tags for the text data, a unique user profile model was developed which makes use of these tags. By the usage of content and personal recommendation strategies the recommendation is done. For instance we have users $U = u_1, u_2, u_3, u_4, \dots$ and the handful of data $C = c_1, c_2, c_3, \dots$ based on the user activity, the tags of the data which is viewed by the user is allocated to the user in the user matrix u_1 . (Say $u_1[T_1, T_5, T_7]$, where T_1, T_5, T_7 are the allocated tags to the user profile u_1). The matrix size varies with respect to the usage statistics. Initially the data tags for the user is null and when the user logs into the application the data retrieval is done based on the department and the year of the user (crude data) provided during the user registration.

After the series of hits, the matrix is updated and the tags are allocated to the user profile. During the next login the data is retrieved based on the updated tags of the user profile. This process done through matrix factorization technique. The model of the same is depicted in eqn-3.1. It is done for every user individually, since it is a personalized recommender system. The below equation is a user-item rating matrix used in matrix factorization.

$$\tilde{r}_{ui} = \sum_{f=0}^{n\text{factors}} H_{u,f} W_{f,i} \quad \dots\dots\dots 3.1$$

Matrix factorization is used to assign the tags to the user profile. The purpose of the matrix factorization is to optimize the search results to the user. Here we can say that if matrix factorization is not implemented, then there's no room for personal recommendation. Singular Valued Decomposition is an alternative for matrix factorization, but SVD is not efficient over missing values in the dataset. Hence matrix factorization technique better suits our application.

CHAPTER 4

REQUIREMENT ANALYSIS AND SPECIFICATION

4.1 REQUIREMENT DESCRIPTION

The custom NLTK program will run whenever the data is being retrieved from the unknown sources . It is categorized and updated as the dataset in the datastore. The framework called Scrapy is used in retrieval of data from the given sources and sent to the database. Once the data is retrieved and tokenized, the recommendation algorithms works on to identify interest points. To do these processes, the server system is used. As light-weight frameworks are used with efficient algorithms, the system requirements are economic. The hardware and software specifications are given below.

4.2 HARDWARE SPECIFICATION (Server side):

Minimum hardware requirement for using the student resource recommender system is provided below

- Processor: intel i5 (4-cores)
- Hard Drive: 500 GB
- Disk Space: 8 GB
- RAM: 4 GB or More

4.3 SOFTWARE SPECIFICATION (Server Side)

Minimum software requirement for using the student resource recommender system is provided below

- **Operating system:** Linux, windows, mac
- **Frontend:** Any web browser with JavaScript support
- **Backend:** Python, NLTK, Mongodb, Tensorflow, django framework.

4.4 SOFTWARE SPECIFICATIONS (CLIENT SIDE)

Minimum software requirement for using the student resource recommender system is provided below

- **Operating system:** Linux, windows, mac
- **Frontend:** Any web browser with JavaScript support

CHAPTER 5

5.1 SYSTEM DESIGN:

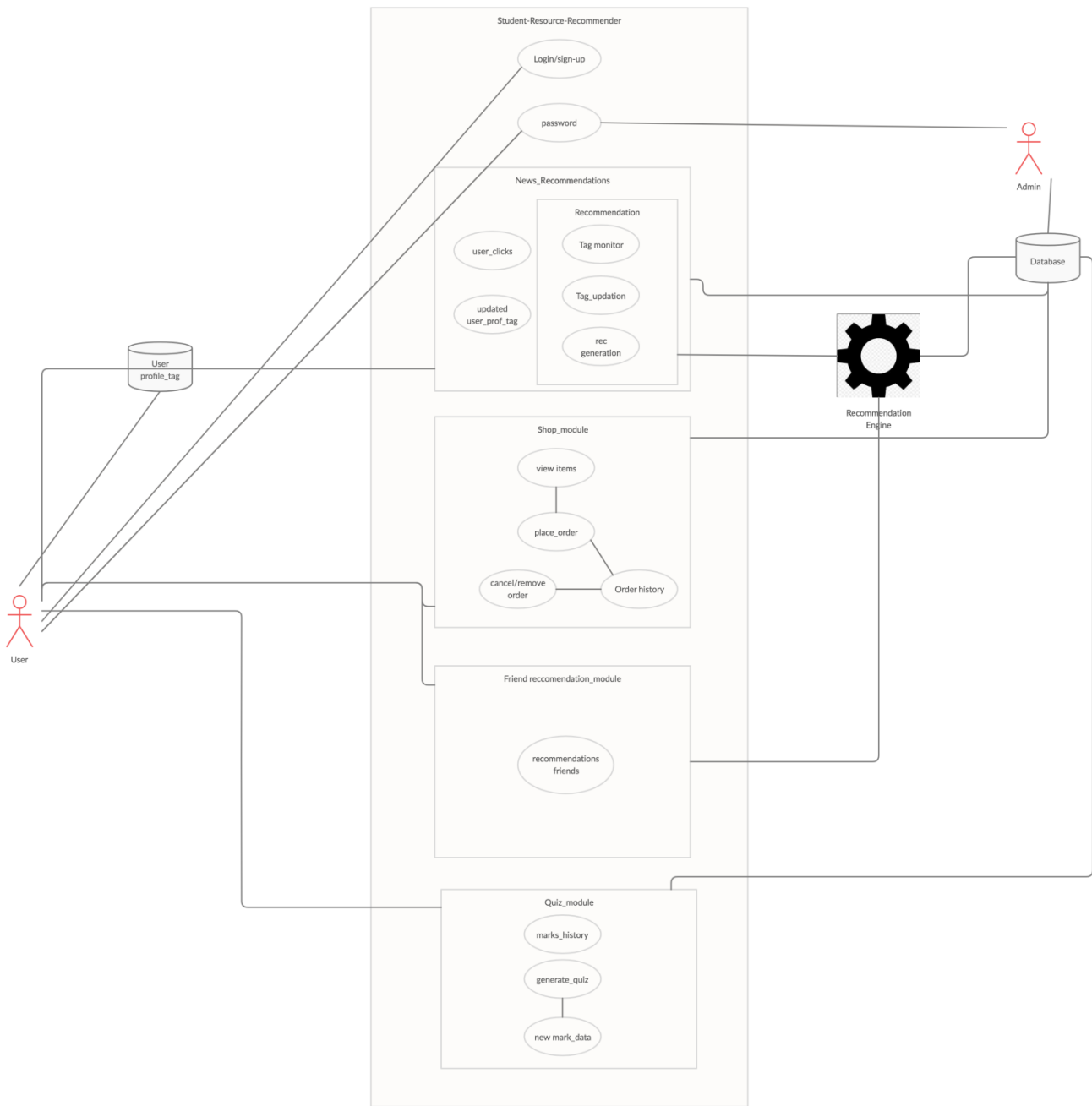


FIGURE-5.1 : USE-CASE DIAGRAM FOR SRR

Figure-5.1 represents the USE-CASE-Diagram for the Student Resource Recommender. A use case diagram at its simplest is a representation of a user's interaction with the system. It shows the relationship between the user and the different use cases in which the user is involved.

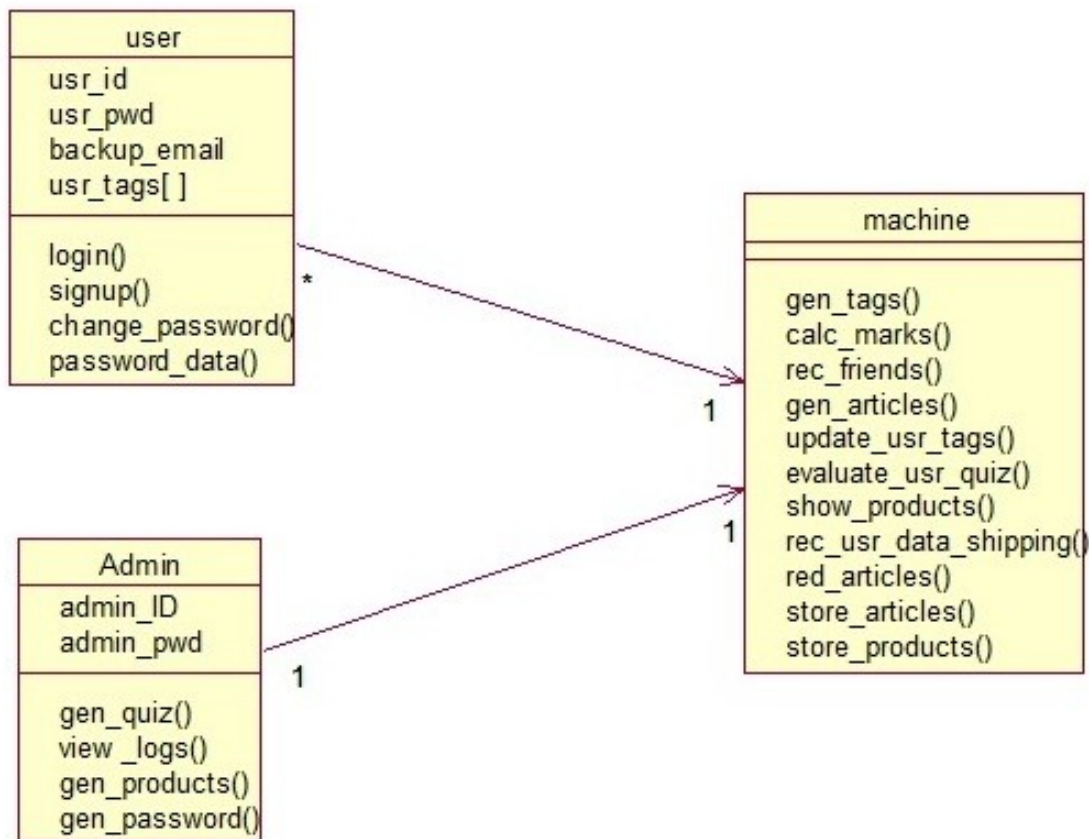


Figure -5.2 Entity relationship diagram for Student Resource Recommender

Figure-5.2 represents the ER diagram for Student Resource Recommender. An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities.

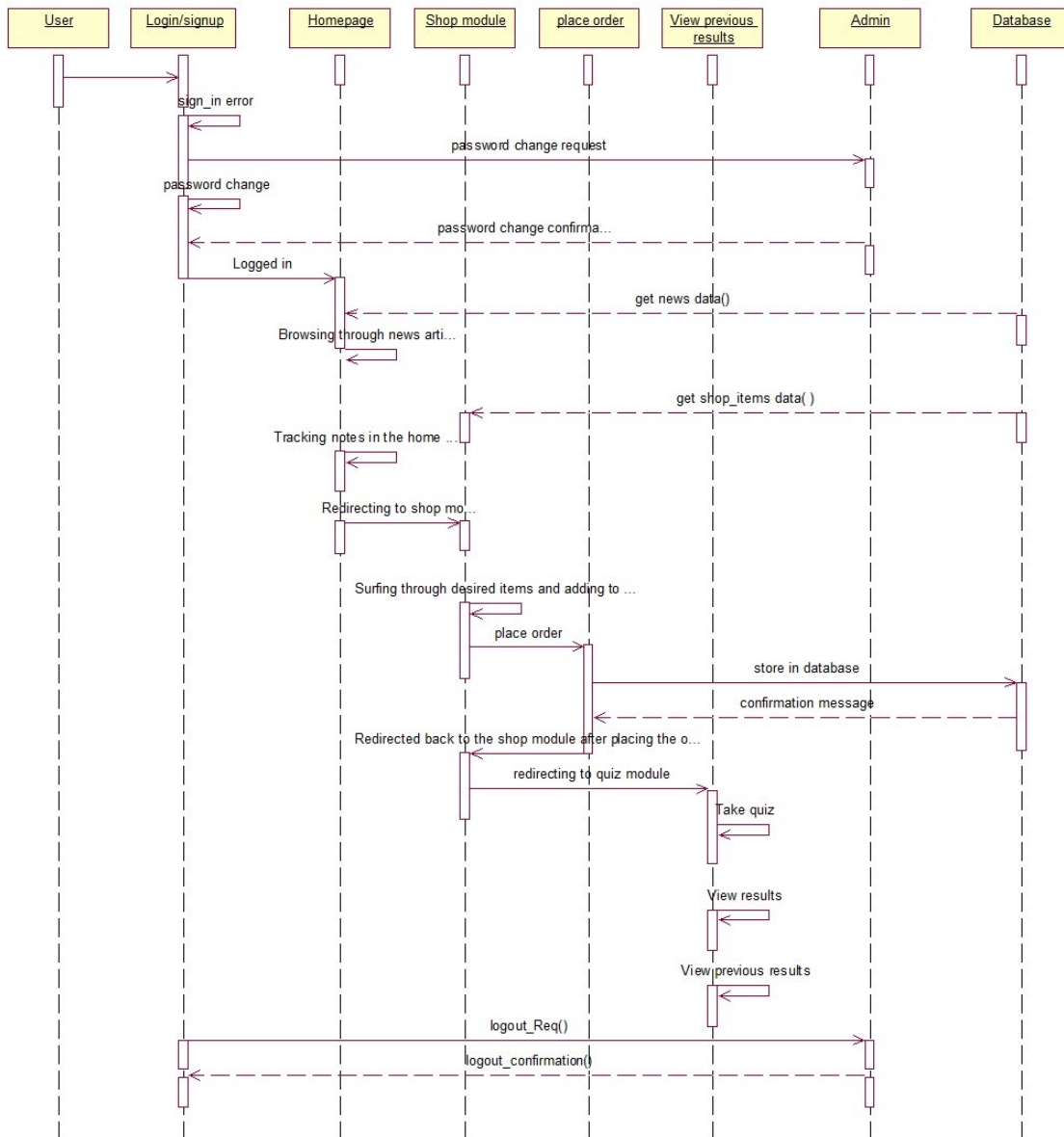


Figure -5.3 Sequence diagram for Student Resource Recommender

Figure-5.3 represents the Sequence diagram for Student Resource Recommender. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

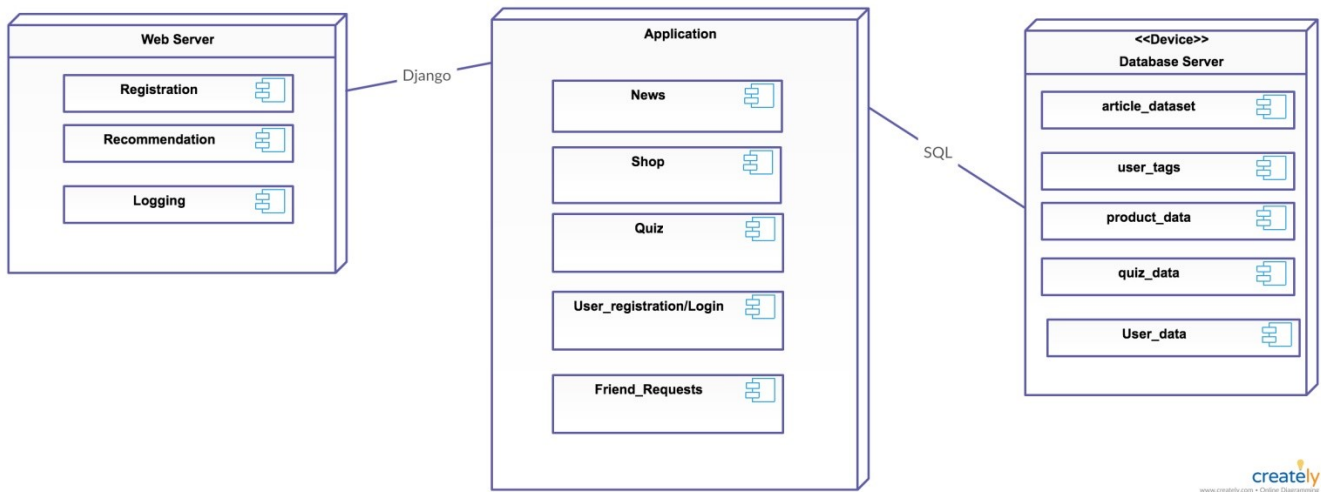


Figure 5.4 Component Diagram for Student Resource Recommender System

Figure 5.4 represents the component diagram for the Student Resource Recommender. In Unified Modelling Language, a component diagram depicts how components are wired together to form larger components or software systems. They are used to illustrate the structure of arbitrarily complex systems

Note: We've used the use-case, ER, component, system architecture , sequence diagrams for comprehending detailed system design. We've learnt this concept in Object Oriented Software Development (CSE-401).

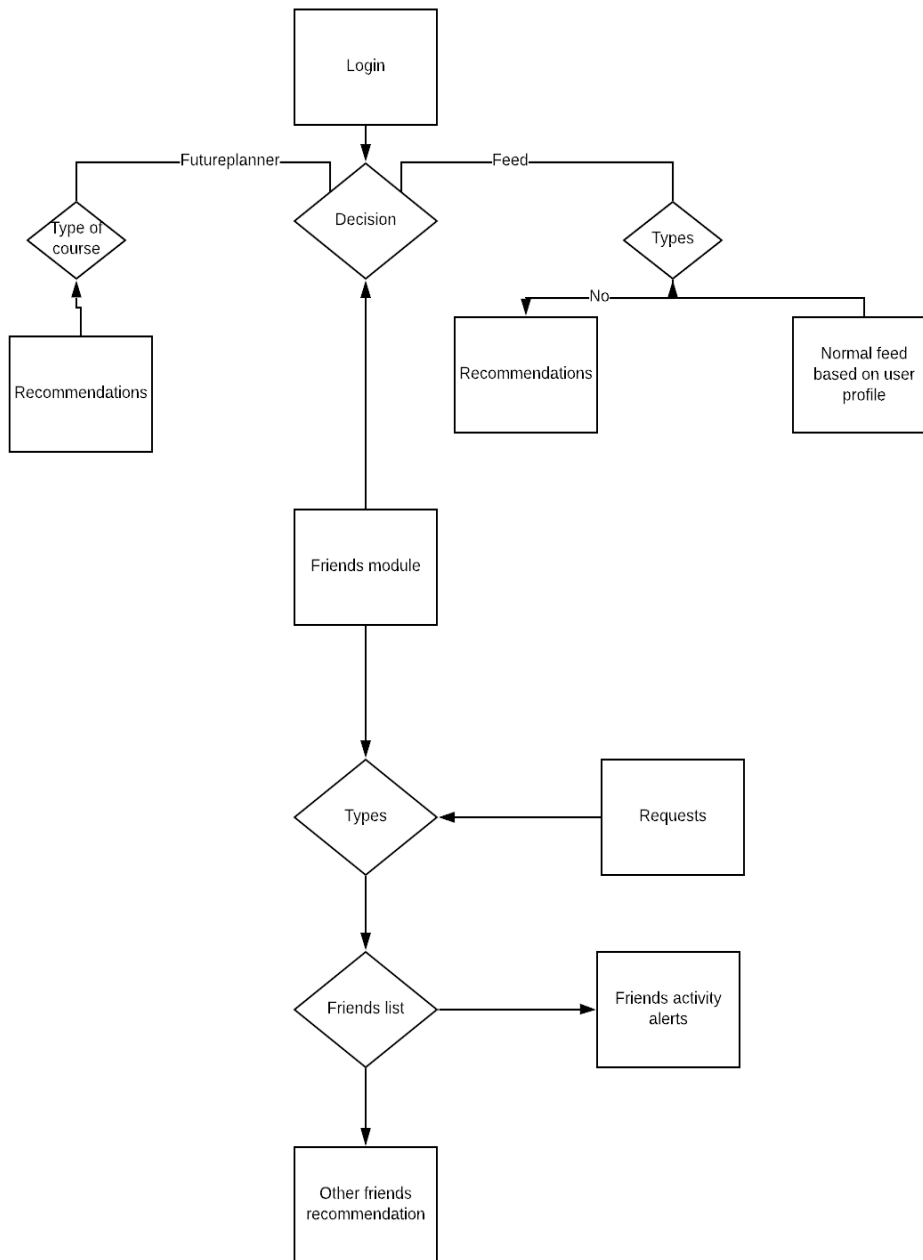


Figure -5.5 Work-flow chart

Note: We've used modular approach in building this project. Modular approach subdivides the project into many modules/parts which can be independently created, replaced, modified. We have learnt this concept in software engineering in 5th semester (CSE-303)

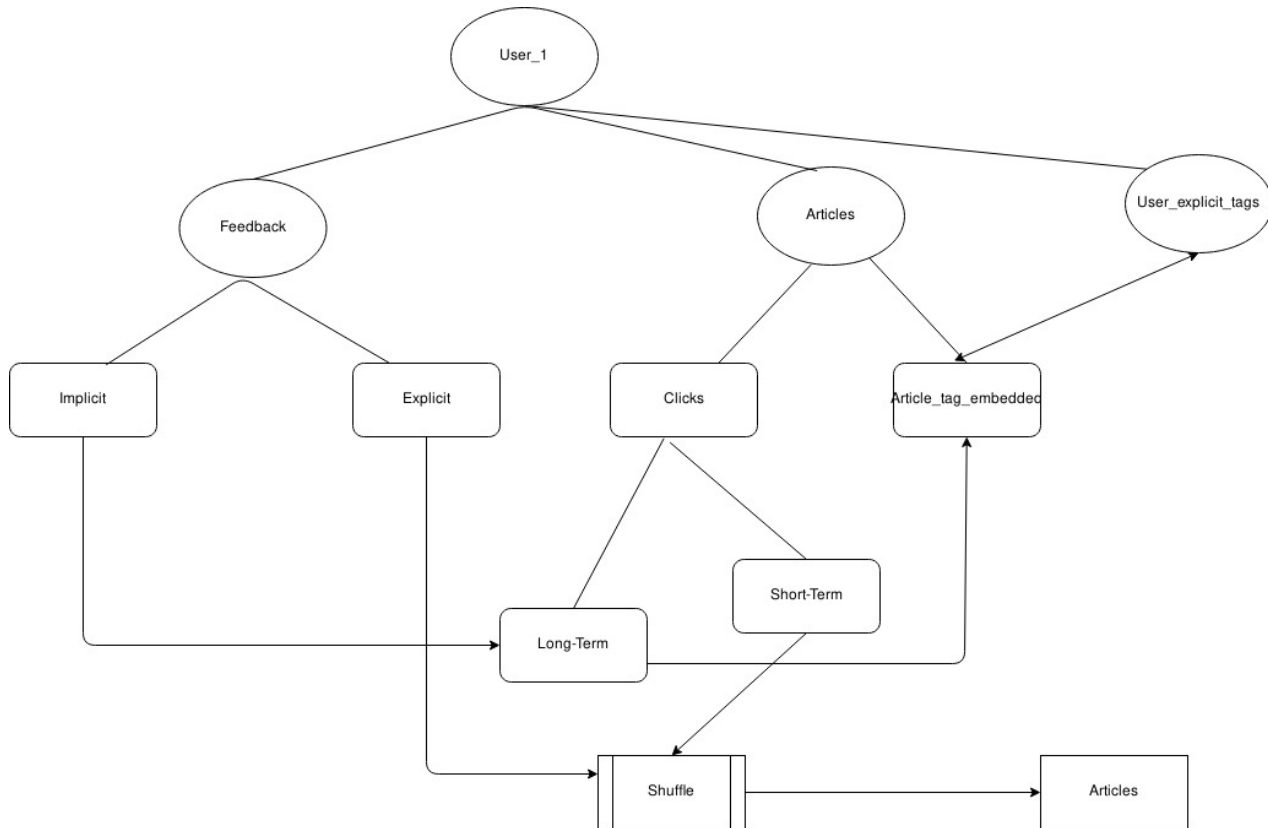


Figure 5.6 : Flow chart depicting the recommendation module for user profile

The Work flow of Student Resource Recommender application is depicted in figure -5.6. we have the user-profiling model that is designed to formulate the article tags corresponding to the user tags, with respect to the number of hits on similar articles by the user/ similar users. The article recommendation depends on the user's behavior as well as the tags allocated to the articles in the dataset.

Figure-5.5 depicts that initially the scraper works and gets the data of articles. After scraping the data using the pipelines, the data is stored in the database. The NLTK framework uses this data to generate the tags to it such as {'science', 'computer', 'security', 'GATE', 'GRE'}. These tags are allocated to the text data and stored in the database as a final table. The Django framework will get the data and presents the processed interest articles to the user, as recommended articles. The initial recommendation is done based on the user's crude data such as department in which he is studying. Later on the algorithm starts to work, leveraging the matrix factorization technique. Thus the user-tags

corresponding to the article-tags are correlated. Based on the updated tags, the recommendation is done, when the users logs on to the application in future.

The friend recommendation module is based on the user's usage of the articles and marks module. If two users have similar tags(user tags) then they are recommended to each other. After the request and acceptance of the friend requests, the articles of the other friends are recommended to the former users, based on the marks obtained in the tests module.

5.2 Design Constraints and Standards:

The application considers two constraints, namely, social and sustainability. Social constraint is satisfied by providing plausible friend requests and chat communications. The main operation of the application pertains to recommendation algorithms, providing personalised suggestions to users with respect to their profile and usage statistics. The entire application is built on top of open-source frameworks leveraging python programming. Hence the application is sustainable. The recommendation process is modelled using user-tags generated through language processing tool, viz. NLTK. Using the tags and appropriate recommendation algorithms, various suggestions/ instructions can be given to assist the user. The language processing and computer assisted instruction complies to IEEE 610.2-1987 - IEEE Standard Glossary of Computer Applications Terminology.

5.3 Design Alternatives:

5.3.1 T-UP-TREE ALGORITHM:

There are many state-of-art approaches rather than T-UP-tree algorithms such as UP-tree algorithm. Although the UP- tree algorithm relies on the user's prolonged usage it uses native recommendation, i.e., monitoring user and updating tags preferences. But we are using tagging systems in this application. Initially the recommendation for the user is not reliable. but as the number of users increase the accuracy of the result increase in T-UP-Tree algorithm.

5.3.2 MATRIX FACTORIZATION:

Matrix factorization is a class of collaborative filtering algorithms used in recommender systems. It works by decomposing user-item interaction. The idea behind the matrix factorization is to represent the users and items in a Lower dimensional latex space. Here we have modified this a little bit to changed it into a user-item-tag interaction; As there would be a matrix for tags and the user-items.

As the user clicks on the items (articles) the tag-matrix is updated and the highest viewed tags are allocated to the user and updated consistently.

5.3.3 DJANGO & UI :

The usage of the Django framework has a unique meaning; as the framework itself has many inbuilt features and security provisions. So the developer doesn't have to worry about the trivial items and concentrate only onto the development part. Since the Django is a python framework, it is compatible with rest of the applications, since we use scrapy and NLTK. Thus framework embedding will not be tedious.

5.3.4 RECOMMENDER ALGORITHMS:

We proposed a tagger-based hybrid recommender algorithm leveraging NLTK framework. The recommendation process can also be done using alternative methods including content-based and personalized settings. The performance of those along with the state-of-the-art Music tagger algorithm is tested with our news recommendation problem. The empirical evaluations of the proposed approach compared to the alternative designs is depicted graphically in section 8.2.2. It is evident that the proposed algorithm outperforms the existing state-of-the-art algorithms for news recommendation.

CHAPTER - 6:

PROPOSED APPROACH:

6.1 Dynamic model of the User Profile:

To learn user's interest, generally a personalized recommender system needs to construct the user profile. Traditionally user's interests can be tracked by the user's history [1,2]. Our goal in this paper is to dynamically build user interest features and add tags to the user profile. After applying Tagger-UI-LDA model, a special user profile is designed to construct the model from user's implicit feedback and use the model to add tags to the user profile, so based on the tags and usage of the user, the model will dynamically adapt to the user interest for the profile. The tags are generated through the language processing process which comply to IEEE 610.2-1987 - IEEE Standard Glossary of Computer Applications Terminology.

6.2 Construction of Tagger-User Profile-Tree:

To capture user's reading interests the user profile is given the basic tags during the sign-up process. To build a the user profile, we use the decision tree-based approach for modeling user interest based on his/her choosing behavior of the articles. Figure 6.2.1 diagrammatically explains the proposed T-UP-Tree algorithm. The architecture works in a bottom-up way. Initially the data pre-processing module starts working. The scrapy component crawls the data and stores it in an initial data storage. The language processing component, NLTK, processes the articles and assigns tags to it. The data storage component comprises of various datasets including user_data, article_data, tagged_article_data, user_profile_tag_data, quiz_data, shop_item_data. The Recommendation Engine is another layer which connects with the datasets and the Django layer. It processes the user's data and generates the recommendations for every module. The UI is the Django layer. The user interface contains shop, quiz, friends and news modules in which the user interacts. Final layer is User types. The recommendations are being generated based on the user. the amount of user's activity is proportional to the recommendation accuracy.

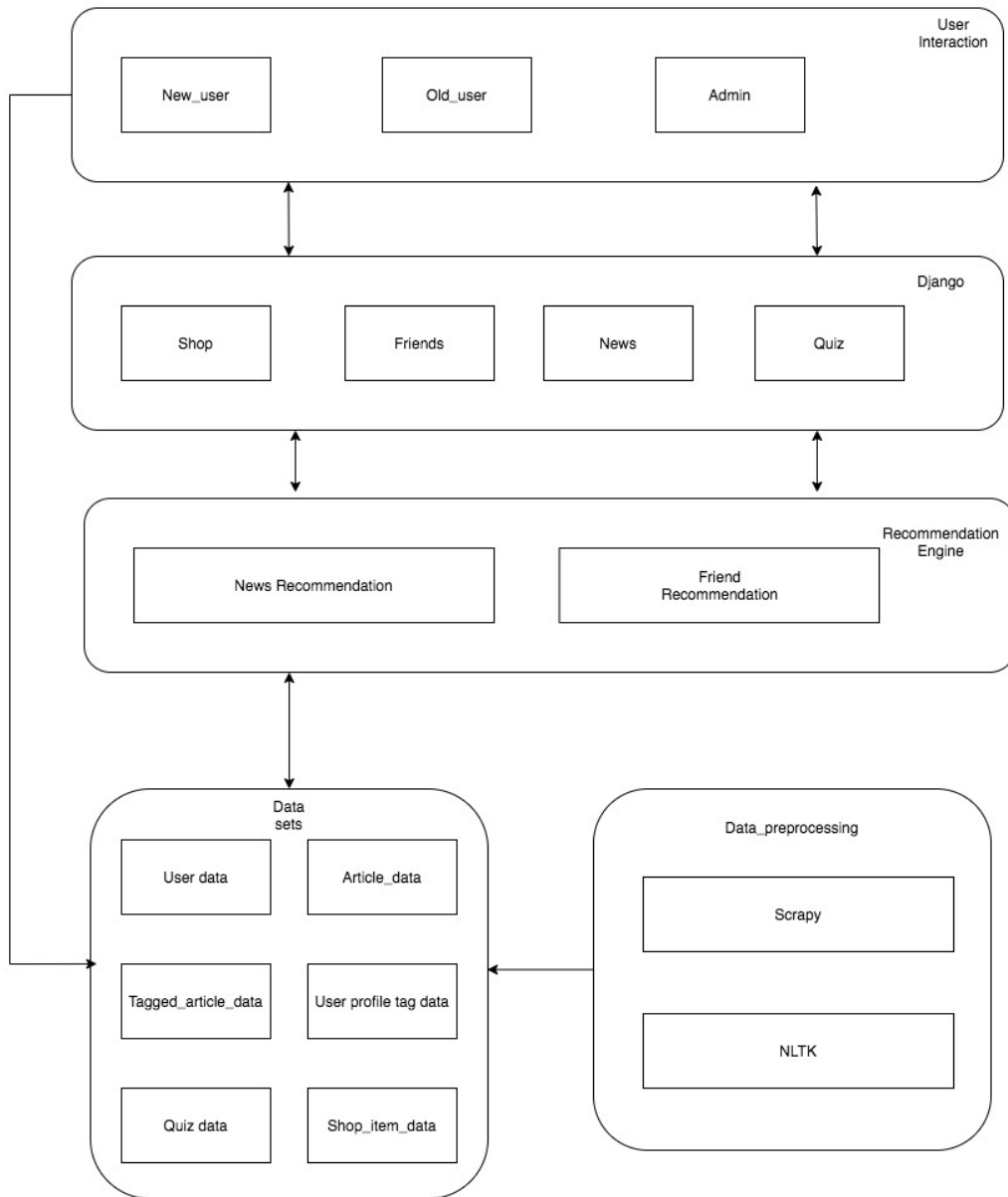


Figure 6.2.1 : System architecture for T-UP-Tree algorithm

Algorithm-1:Tagger User Profile Tree Algorithm

Input: Word vectors $\{w\}$, Dataset, User profile,UP- tags, Article tags.

Output:Updated user profile tags, Multinomial parameters

1. //Initialisation
 2. Initialise the $nm(k),n(m),nk(t),nk$ to zero
 3. //Get the initial data of the user profile tag data and the articles dataset
 4. Get the initial tag data $\{t\}$ from the user profile
 5. Get the articles related to $\{t\}$ and show the data on top of the list
 6. //User Interaction
 7. **for** all articles $a \in [1, A]$ do
 8. **for** all words $n \in [1, N_m]$ in article m do
 9. Sample topic index $Z_{m,n} = k \sim \text{Mult}[1/k]$
 10. Increment article-topic count: $nm(k) += 1$.
 11. Increment article-topic sum: $nm += 1$.
 12. Increment article-tag count: $c_{m(k)} += 1$
 13. **end for**
 14. **end for**
 15. //Gibbs sampling over a period of time
 16. //Assigning the articles maximum tag count to the user profile
 17. **while** unfinished do
 18. **for** all articles $m \in [1, M]$ do
 19. //Allocate the maximum article topic count to the user profile tags(New User preferences)
 20. **If** article-topic count($c_{a(K)}$) is max
 21. article-tag \rightarrow user profile tag[]
 22. **end for**
 23. **end while**
-

Note:This proposed algorithm is a dynamic programming algorithm. A dynamic programming algorithm (also known as dynamic optimization algorithm) remembers the past result and uses them to find new result means it solve complex problems by breaking it down into a collection of simpler modules, then solving each of those modules only once, and storing their solution for future use instead of re-computing their solutions again. We've learnt about this dynamic programming algorithmic concept in Algorithms and complexity (CSE-209).

6.3 Tagger-User Profile-Tree algorithm:

The algorithm that we have proposed in this paper is Tagger-User Profile-Tree algorithm. It is a novel approach for the user-profiling algorithm, which enhances the recommendation for the user in a significant manner. Initially, the recommendations are done by the crude data of the user such as his/her department, designation, domain, among others. The initial tag list of the user is set to zero. If the user clicks on the article the article-topic count increases. The sum of the article-topic is $(nm(K))$. The tag count of the above articles is $cm(k)$.

//Gibbs sampling over a period of time

Check the article-topic count and assign the highest article tags to the user profile tags.

// Recommendation after a period of time.

Get the user's profile tags.

Output:Show the relevant articles which match with the user's tag.

This algorithm is not limited to this application. It can be implemented in any applications that can use the concept of tagging mechanism.

Note:T-UP-Tree(Tagger-User Profile-Tree) proposed algorithm is a construction of Directed Acyclic Graph which was learnt from Data structures (CSE-103)

6.4 FRIEND RECOMMENDATION ALGORITHM:

The friend recommendation algorithm is similar to the normal recommendation algorithm used. It uses the user-profile tags to recommend the users. Some of the factors included for recommending the friends to the user are:

- Based on the similarity of the profiles tags
- Based on the quiz scores
- Based on similar department

The friend recommendation algorithm collects the user’s data such as user_profile_tags, quiz results and shopping history. With the available data from various users, Interest analysis is done to associate users in content and context perspectives. The interest analysis is done primarily by calculating the similarity index between the users.

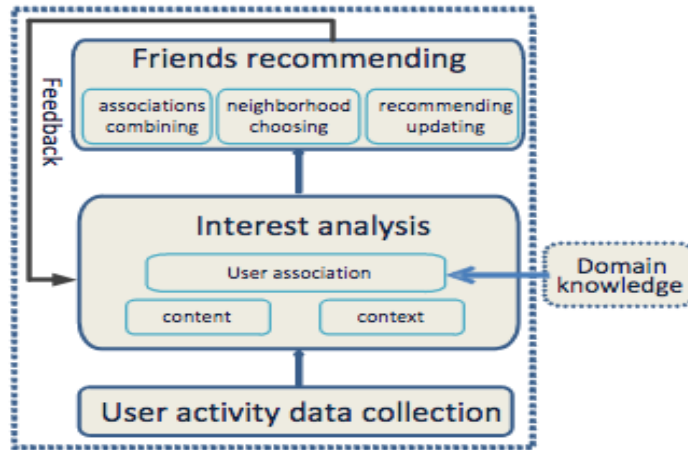


Figure -6.4.1 : Friend recommendation model

$$Similarity(I_U, I_V) = \frac{|I_U \cap I_V|}{|I_U \cup I_V|} \quad (6.1)$$

Figure 6.4.1 represents the Friend Recommendation Model. The Friend recommendations takes place by the calculating similarity of the user-profile-tags and the shopping list for content based analysis and time for context based recommendation. The Similarity index measure between two user interests (I_U, I_V) is depicted in equation 6.1.

CHAPTER 7 :

7.1 IMPLEMENTATION AND RESULTS:

In this section, we have shown the results of the several experiments that we have conducted on the datasets. The grouping of data is done by the NLTK framework which is a language processing tool. And the tree for grouping of sample data in the data set is shown in figure-8.2.1. It is evident in the literature that the data retrieval is done efficiently in the UP-tree model. The proposed algorithm extends UP-tree model for faster retrieval of data and the same is achieved by the profile taggers and data (article) taggers. In our experiments we used the following methods as baselines: Since the existing works in news article recommendation systems considered only the contents, to provide recommendations to the users, the proposed algorithm provides recommendations dynamically. The data in the data set is categorized and the tags are allocated to the user profile which made it significantly easier to update user-profile and aids faster data retrieval for analysis.

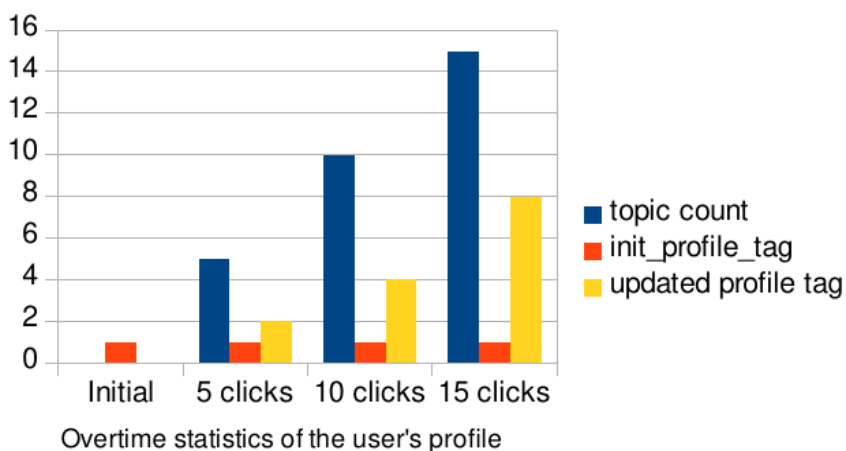


Figure -7.1.1 Statistics of the user's profile tags

The graph [figure 7.1.1] represents the transformation of the user profile tag overtime based on the user's behavior. As the number of clicks of the article increases, the tag of the similar articles which have been visited more will be added to the user's profile and the recommendations are done leveraging the user profile.

7.2 TESTING:

7.2.1 Data set categorisation:

One of the problems that many recommenders systems face is the data sets formatting. We are using NLTK for the tag allocation to the articles. The figure-7.2.1 represents the simulated dataset. It is evident that the data are of various categorized with high variance between the categories and high bias within the categories. Hence the data is subject to noise. Thus matrix factorization techniques help contour such noise data with bias.

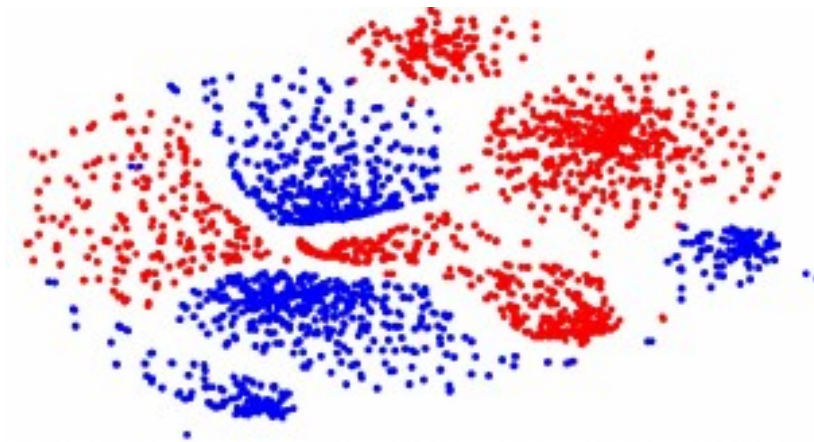


Figure - 7.2.1: T-SNE visualization of the sample dataset

7.2.2 PERFORMANCE TESTING:

We have used Mean average precision to calculate the precision rate of the recommender system. We have compared it with the existing personalized and content-based recommender algorithms. The precision of retrieval of the data is significantly improved in the Tagger-User Profile-Tree when compared to content-based or the personalized recommendation algorithms. The graph shows the precision level between the three recommender systems.

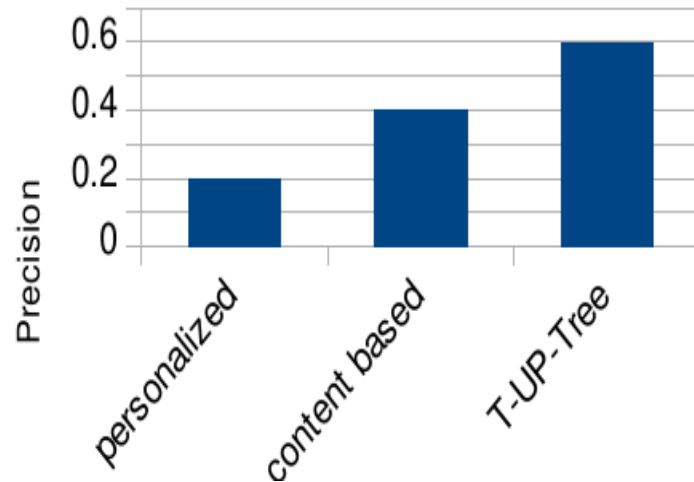


Figure 7.2.2: Precision graph for data retrieval for recommendations

The figure 7.2.3 shows the precision of the Music tagger algorithm and the T-UP-Tree algorithm in real-time. Experiments are conducted with 5 different categories viz {GATE, GMAT, GRE, TOEFL, IELTS} with uniform random distribution on 30 users. The existing music-tagger algorithm is applied in our application and found that as a number of users is proportional to the accuracy level. It is evident from the figure 7.2.3 that music tagger yields an accuracy level of 77.824%. In the similar case the proposed T-UP-Tree algorithm obtained an accuracy rate of 79.965%. The comparison of proposed T-UP-Tree algorithm with various state of the art alternative solutions is depicted in Table-1.

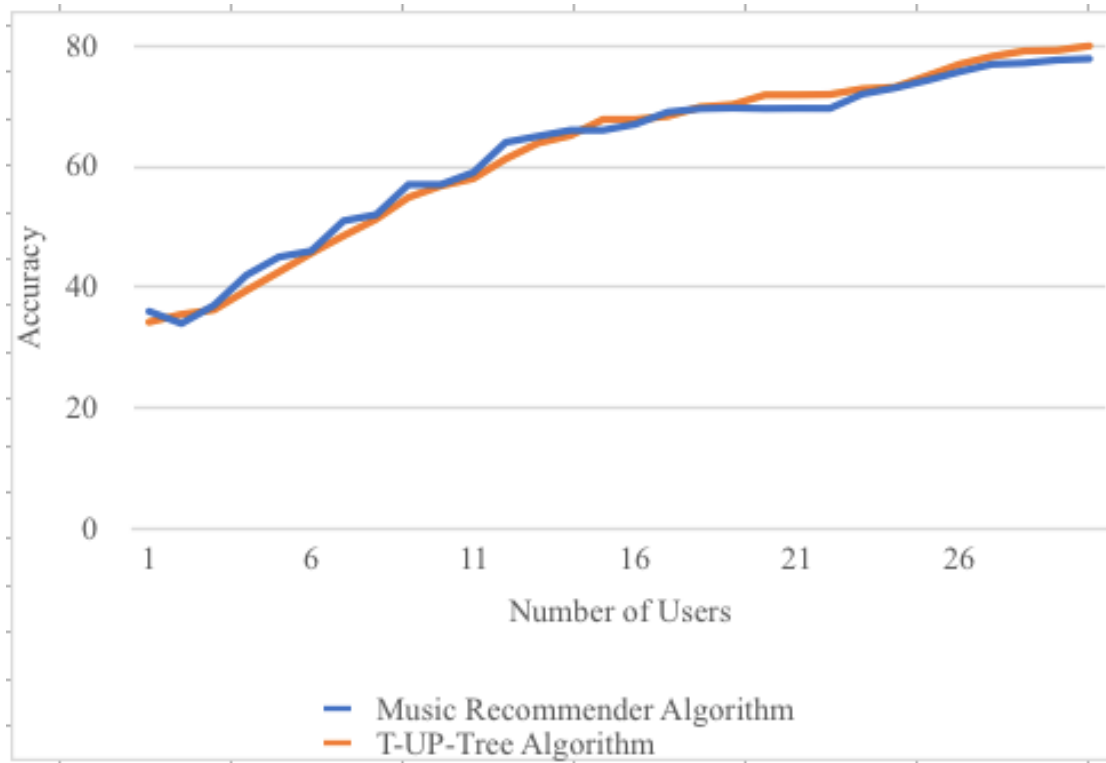


Figure 7.2.3: Comparison of T-UP-tree and music tagger algorithm over a period of time

Table-1 : Comparison of recommender systems with the proposed work

Inference Techniques	Accuracy
Music Tagger	77.824%
Content based filtering	66.670%
Personalised filtering	50.570%
T-UP-Tree(proposed work)	79.965%

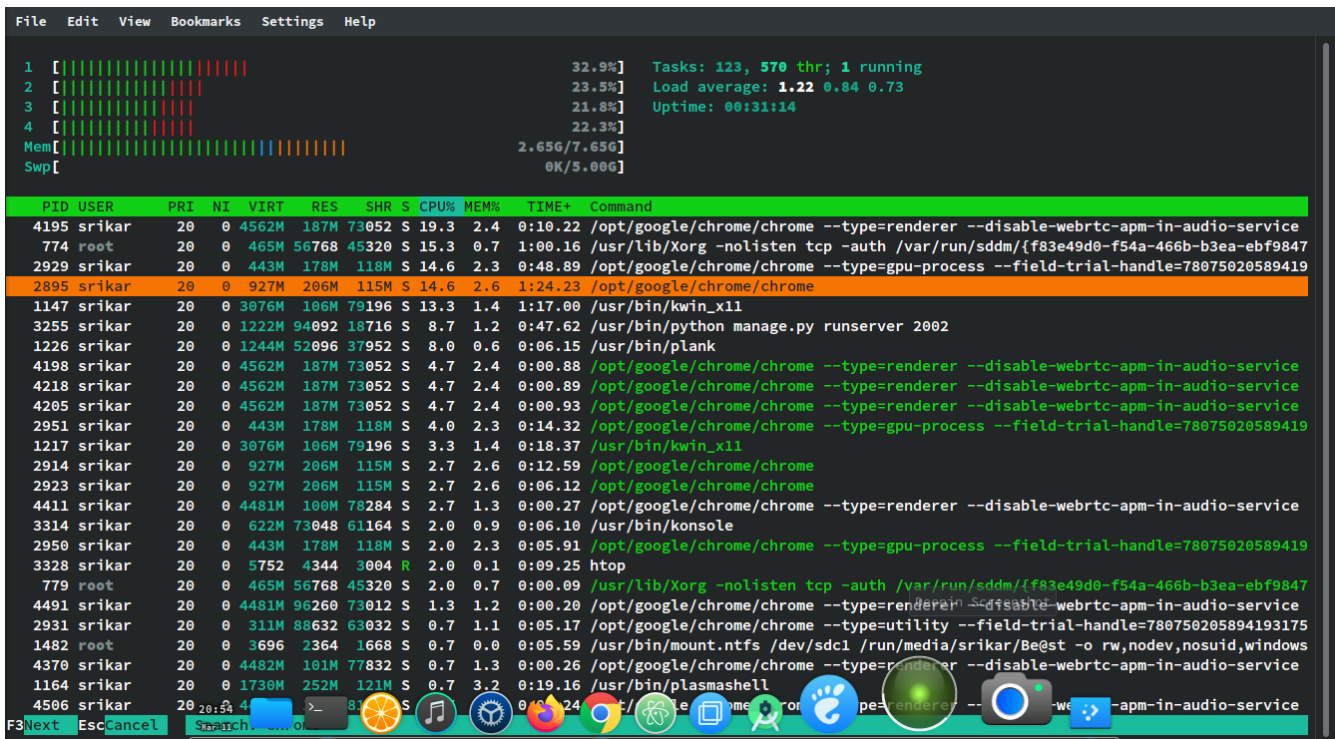


Figure 7.2.4 CPU load data on the server side

The figure 7.2.4 states the load data in the server side for 30 users. The memory consumed in the server side is 2.65GB out of 8 GB in Intel i5 processor server. The CPU load for the 4 cores is around [32.9, 23.5, 21.8, 22.3] respectively.

Note: In probability theory and statistics, the continuous uniform distribution or rectangular distribution is a family of symmetric probability distributions. The distribution describes an experiment where there is an arbitrary outcome that lies between certain bounds. Uniform random distribution is a concept which we've learned from Probability and statistics (MAT-222).

CHAPTER 8

USER INTERFACE

8.1 DJANGO USER INTERFACE:

The goal of the user interface design is to produce an environment that is easier to use, efficient, enjoyable (user-friendly) and stable to operate machine in a way to produce a desired experience. Users usually expect a user-friendly interface which is simple. Django framework gives us the interface which is effective and robust and makes easier for the creators to create and sustain the data in the database. We have several modules that are used in this student resource recommender system which are be discussed below.

Note: For Frontend: The user interface is clean and elegant. The programming concepts used in the user interface in this application are HTML, CSS and Java script. HTML and CSS are used for beautification and the content. Java script is used for many modules to get the data from the database and to maintain and store the form data in the database such as login data, shopping data, notes and other modules. We've learnt these programming concepts in internet programming (CSE-402)

For Backend : The Django is used as the backend for this application. The Django framework is sophisticated and robust framework for creating the web applications. This framework is based on python language which is learnt from the supervisor who has completed "Joy of Computing using Python" NPTEL course with 95%.

8.2 UI for Login:

The user login interface is a simple model with the textbox for the username and password to sign-in as shown in the figure -8.1.



Figure 8.1: Login_page

The user interface for the login is used for a user to login and access the user profile. The username and the password are the unique attributes which defines the user. By using those credentials, the user can login and access the data.

8.3 UI for Registration:

The user registration interface is used to register the user into the database and thereby giving him a membership as a user. The sample layout of the user registration interface is shown in the figure -

8.2:

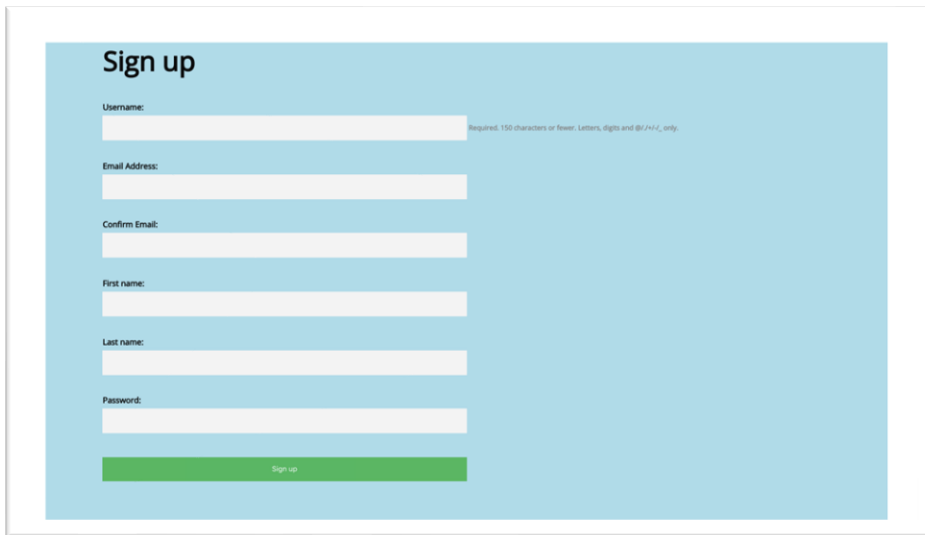


Figure 8.2 : Sign-up page

8.4 News Recommender Interface:

The news recommender interface is the index page of the user in which the news are recommended to the user. The news articles are shown in this page. When the user clicks it they will be redirected to the webpage where that article belongs to, and the data is recorded and the recommendations are shown during the next visit as shown in the figure 8.3.

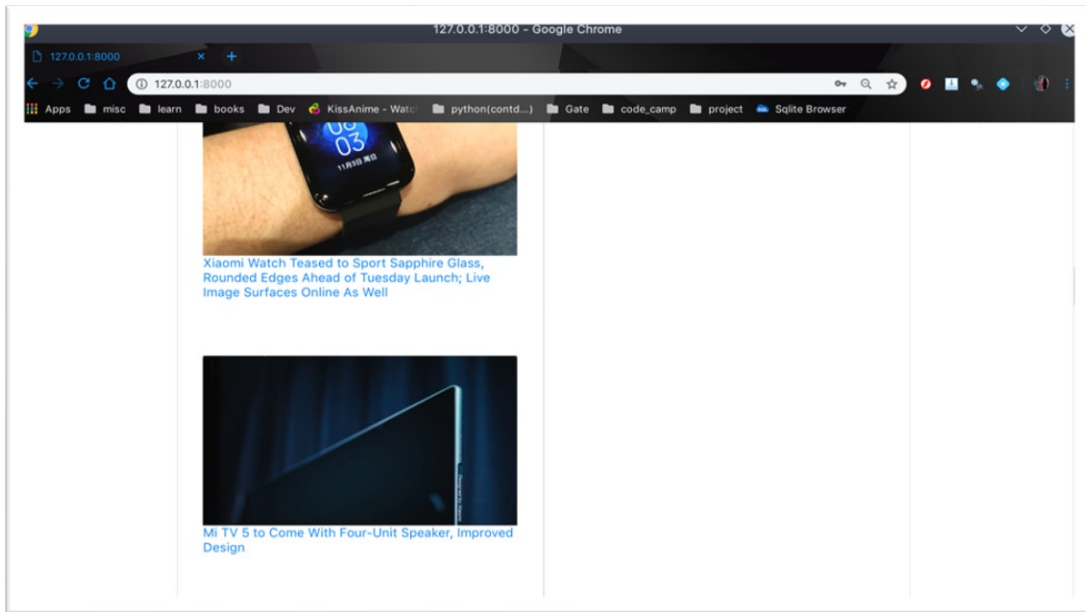


Figure 8.3: News-articles module

8.5 UI for Home page:

The home page is a default page for the user after logging into the webpage. After logging into the webpage, the user will be directed to the home page where he can access the other modules in the web application. The sample of the homepage is shown in figure 8.4.

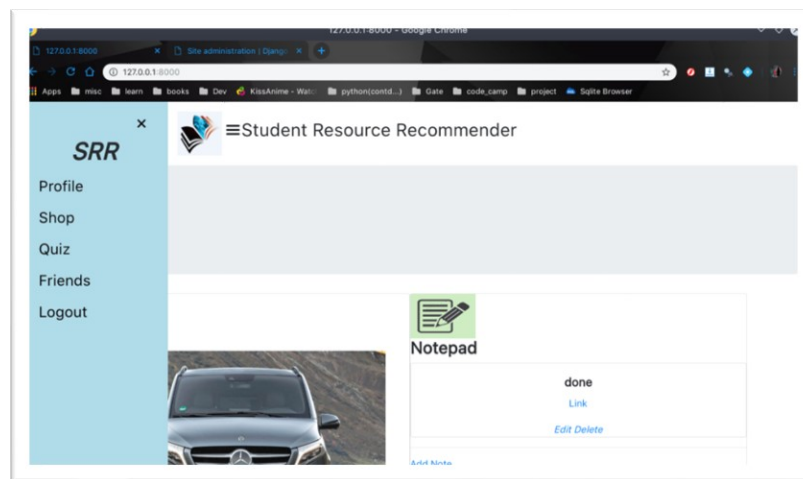


Figure 8.4 : Home.html

This is the home page in the SRR which has the drawer icon. It has links through which, the user can access the other modules.

8.6 UI for Notes:

This is an additional module which acts as an accessory for the users while using the news module or the user can use this for their personal usage, to take notes.

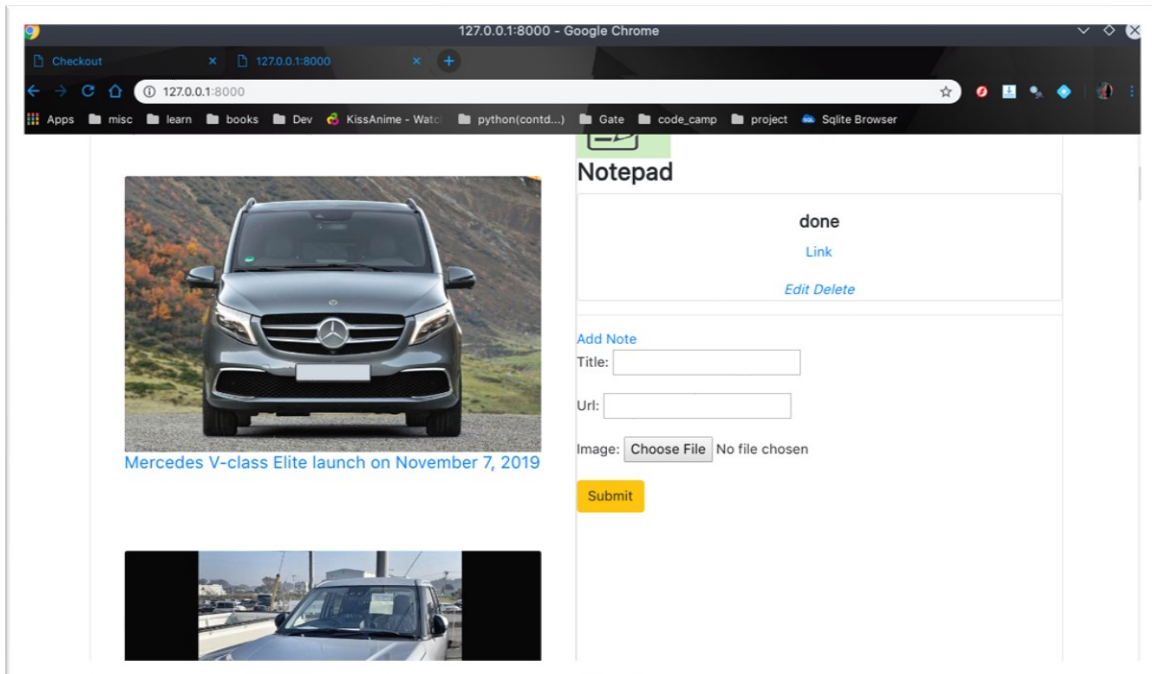


Figure 8.5: Notes

As shown in the figure 8.5, the notes module is used to take sudden notes. The user can also add images, URLs and text-notes.

8.7 UI for Shop module:

The shop module is used to buy books or other accessories from the store provided by the admin (In case, any university or an organization is using this application for personalized usages). The shop module consists of the items[figure 8.6] which are uploaded by the admin and they are categorized [figure 8.7] based on interest and added to cart.

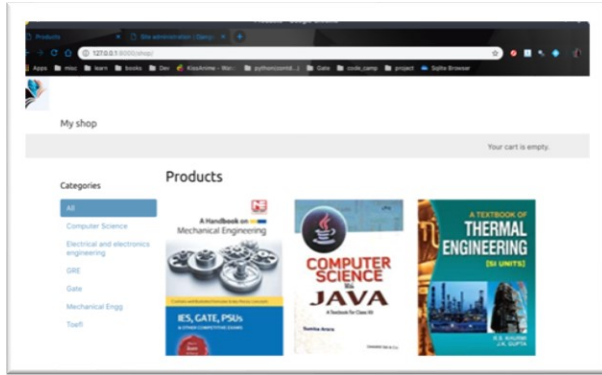


Figure 8.6: Shop

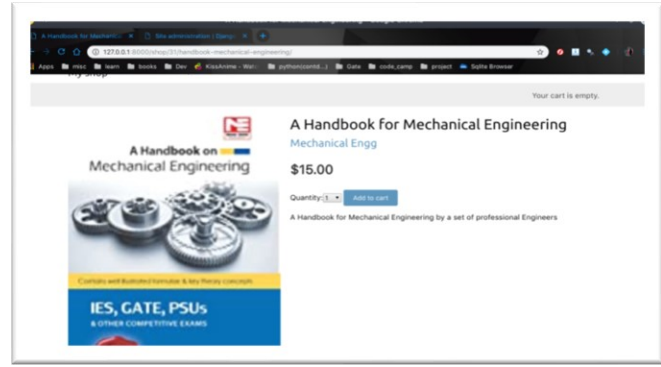


Figure 8.7: Adding to cart

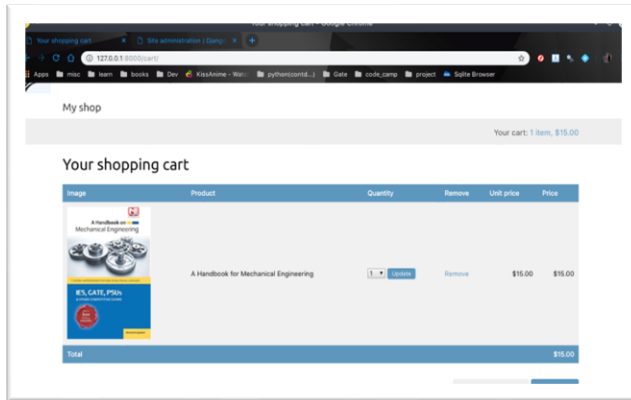


Figure 8.8 :Shopping cart

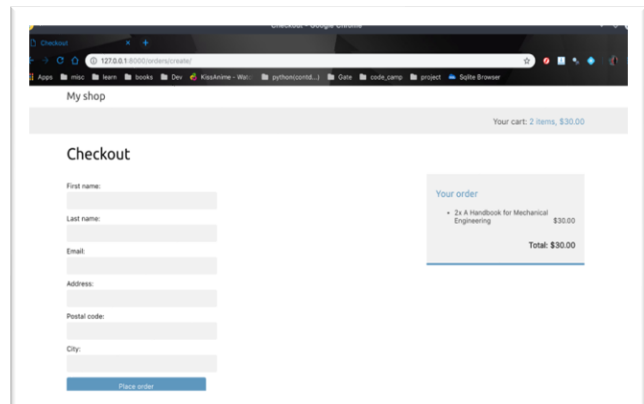


Figure 8.9: Check out

As shown in the figure 8.8 and figure 8.9, we have the shopping cart for gathering the item data, that the user wants to buy. The check out section is used to manage the process of sending the item to the specific address. All this data is stored and is used for recommendation.

8.8 UI for Quiz app module:

The quiz application module is used to take the cognitive tests of the user. The results will be stored in the database and is used to correlate with other friends in the friend recommendations module.

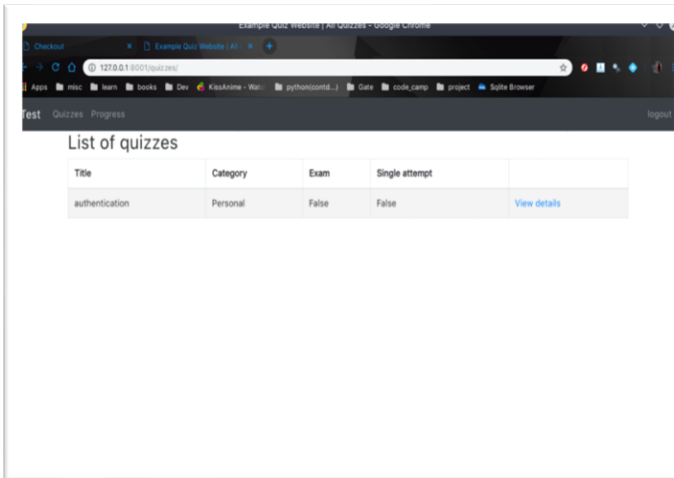


Figure 8.10: list of quizzes

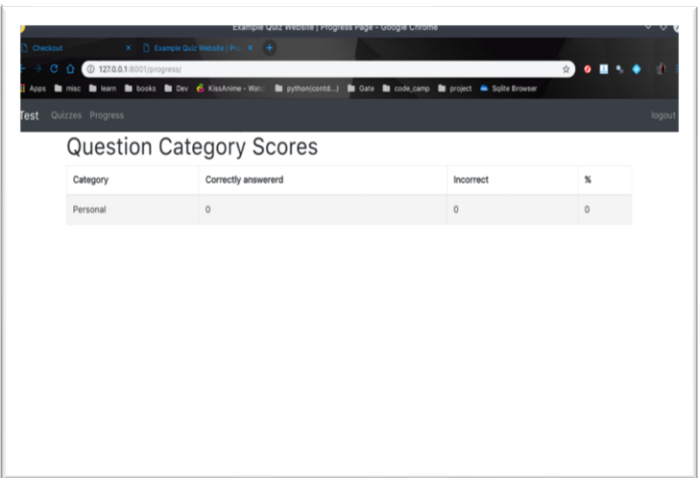


Figure 8.11 : Progress

This quiz module has two sections namely quizzes [figure 8.10] and progress of the user [figure 8.11], with this simple user interface the quizzes are shown to the user and the data such as marks in various categories, is stored in the database.

8.9 User Profile:

This is a simple module that shows about the profile of the user as depicted in the figure 8.12.

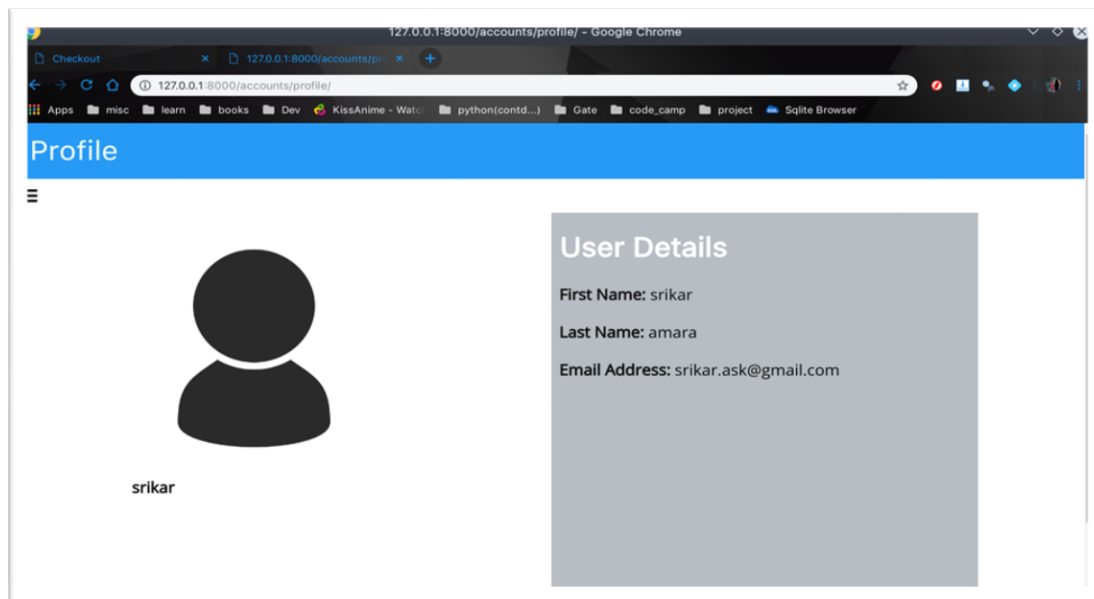


Figure 8.12: User Profile

8.10 Admin Profile:

Admin profiling is given in django by default. It is a simple interface which contains all the models which are registered in the admin.py file of the application. The sample view of the admin page is shown in figure 8.13.

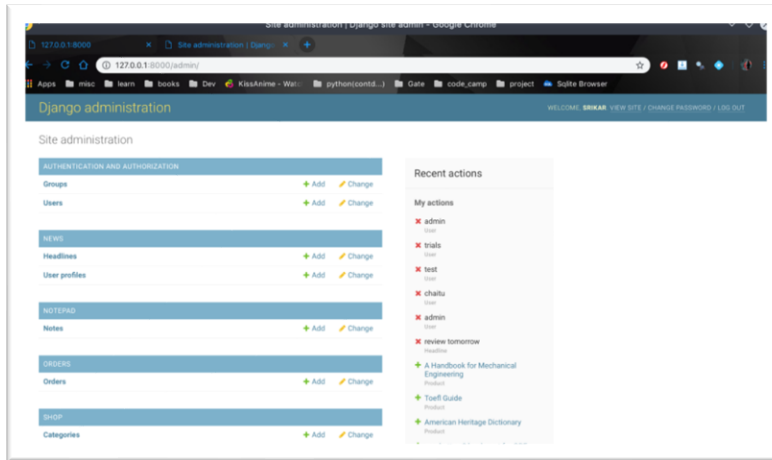


Figure 8.13 Admin_layout

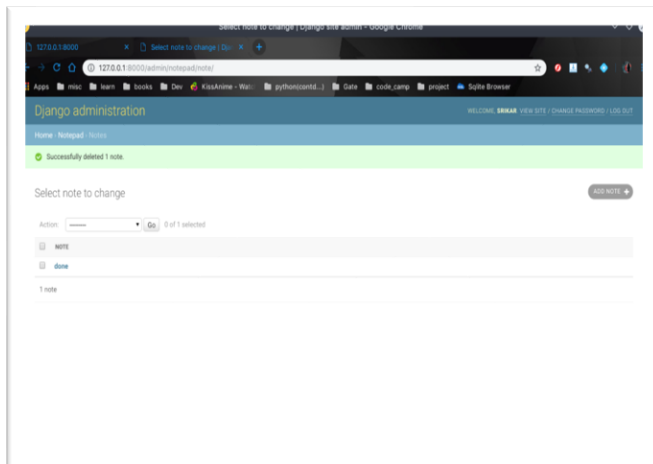


Figure 8.14 Notes_data

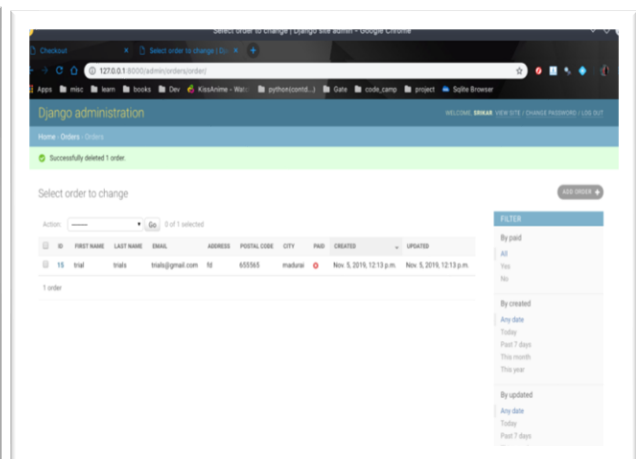


Figure 8.15 Order_data

Figure 8.15 shows the order details of the user with all the details of the user who placed the order. Figure 8.16 shows the scrapy data relevant to the items shopped. And figure 8.17 shows the categorized data by NLTK. The tags are allocated to data and is stored in the database.

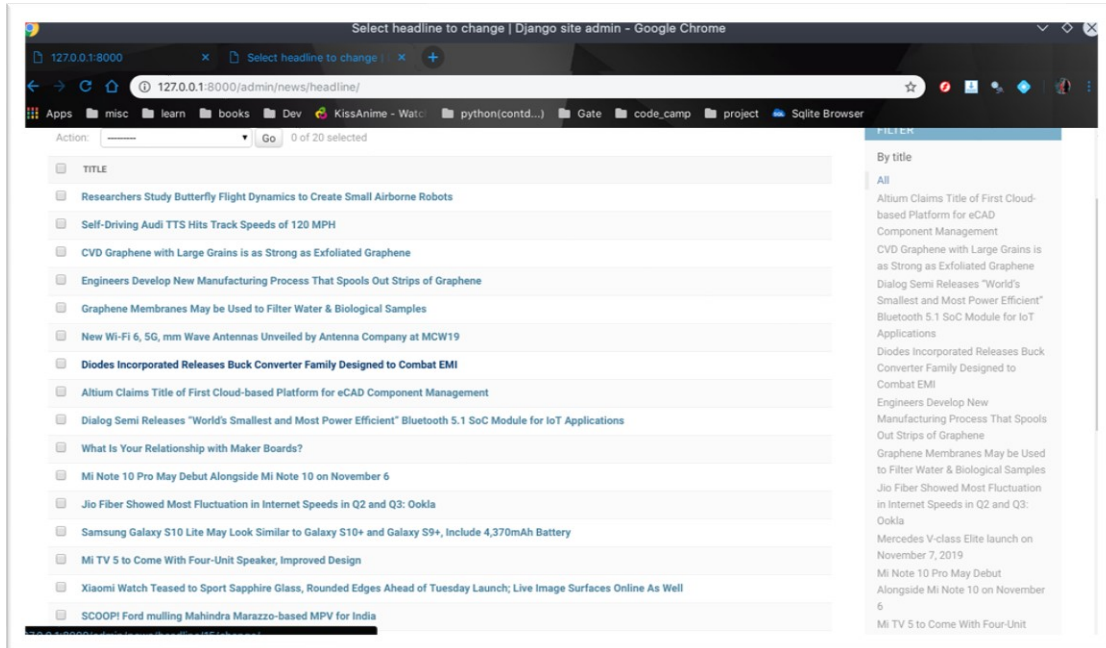


Figure 8.16 News Dataset

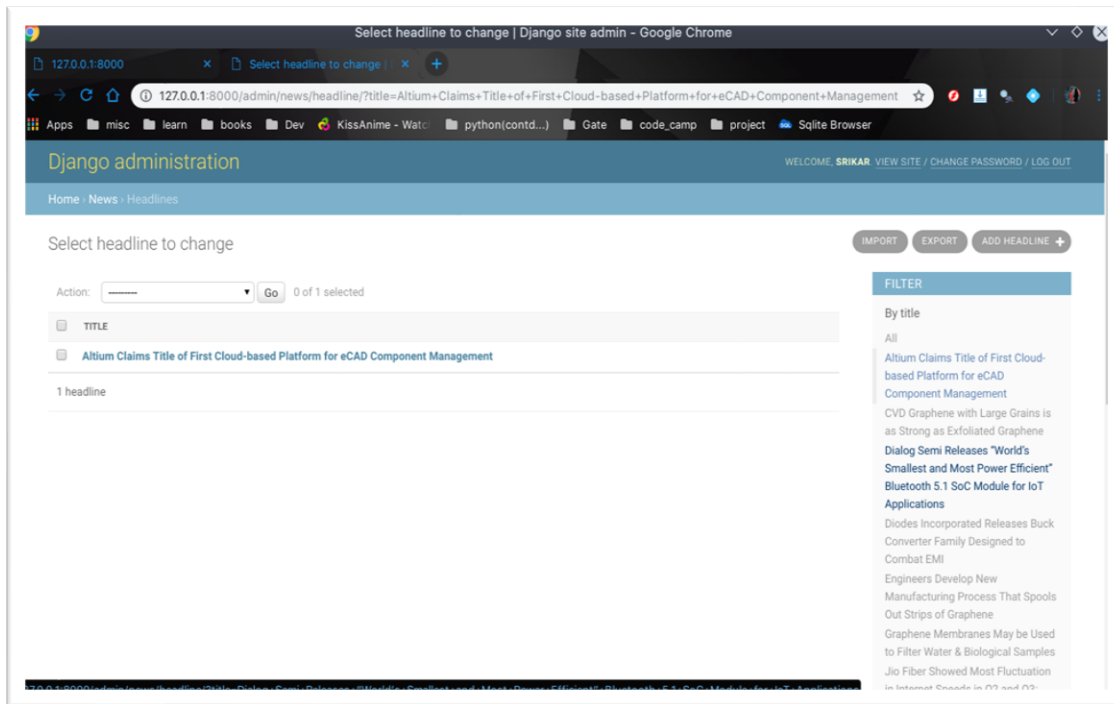


Figure 8.17 View of categorized data

Note: The data storage and retrieval is done using mysql database. The data set is stored in the database using scrapy. By using the pipelines the scrapy stores the dataset into the database. Many database concepts including key concepts, creation, updation, manipulation, referential integrity are used in this application. The concept of the database management systems was learnt in Database Management Systems(CSE-305)

CHAPTER 9

CONCLUSION AND FUTURE SCOPE

The vast resources available on the web and the overwhelming information availability blurs the specific information we sought for. In this project we have presented an effective approach to retrieve interest data from the abundant information. In this project, we have proposed a tagging system for categorizing the data in the dataset, leveraging the novel user-profiling algorithm “Tagger-User Profile-Tree”. The tags of the user will be constantly updated and the relevant information is visible to the user. Experimental results demonstrate especially that T-UP-Tree achieves better accuracy and retrieval performance than normal content and personalized recommender systems. We are also attempting to further improving the algorithm on time and space bounds.

Initially this recommender system is not reliable, although it has better accuracy overtime. Whilst we compare the recommendation process of the other types of systems with T-UP-Tree, and proved that T-UP-Tree provides better accuracy for the underlying news recommendation context. This tagger schema can be used in many systems other than news recommendation, because the tags could be allocated to any items including apparels, accessories, electronics, movies, games, music, etc. Hence the algorithm is generic.

This project Student Resource Recommender follows the sustainability constraints, as the project uses the open source applications such as python, Django, scrapy, mongodb. It is intended to run in most of the platforms. This project is aimed at many communities such as student, faculty, research. Due to the usage of new and emerging technology in this application it is beneficial to those communities and are not limited to those. Hence this application can be viewed as a community development or social project.

REFERENCES:

- [1] Hongwei Wang, Fuzheng zhang, Xing Xie, and Minyi Guo. “DKN: deep knowledge-Aware network for news recommendation.” In Proceedings of the 27th international conference on World Wide Web. ACM, 2018.
- [2] Shumpei Okura, Yukihiro Tagami, Shingo Ono, and Akira Tajima. “Embedding-based news recommendation for millions of users.” In Proceedings of the 23th international conference on Knowledge Discovery and Data Mining. ACM, 2017, 1933–1942.
- [3] Gabriella Kazai, Iskander Yusof, and Daoud Clarke. “Personalised news and blog recommendations based on user location, facebook and twitter user profiling.” In Proceedings of the 39th international ACM SIGIR conference on Research and development in information retrieval. ACM, 2016, 1129–1132.
- [4] Jiahui Liu, Peter Dolan, and Elin Rønby Pedersen. “Personalized news recommendation based on click behavior.” In Proceedings of the 15th international conference on Intelligent user interfaces. ACM, 2010, 31–40.
- [5] Emmanuele Coviello, Riccardo Miotto, Gert R. G. Lanckriet Combining content-based auto-taggers with decision-fusion. In 12th International Society for Music Information Retrieval Conference (ISMIR 2011)
- [6] T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: a model for redicting social tags from acoustic features on large music databases. *Journal of ew Music Research*, 37(2):115–135, June 2008.
- [7] M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In Proc. ISMIR, pages 369–374, 2009.
- [8] J. Kittler. Combining classifiers: A theoretical framework. *Pattern Analysis and Applications*, 1(1):18–27, 1998.

- [9] R. Miotto, L. Barrington, and G. Lanckriet. Improving auto-tagging by modeling semantic co-occurrences. In Proc. ISMIR, pages 297–302, 2010.
- [10] S.R. Ness, A. Theocharis, G. Tzanetakis, and L.G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic svm outputs. In Proc. ACM MULTIMEDIA, pages 705–708, 2009.
- [11] Li, L., Zheng, L., & Li, T. “LOGO:a long-short user interest integration in personalized news recommendation.” ACM Conference on Recommender Systems, Recsys 2011, Chicago, Il, Usa, October, 2011, Vol.16, pp.317-320).
- [12] Ke Xu, Yi Cai, Huaqing Ming, Xushen Zheng, Haoran Xie, and Tak-Lam Wong. “UIS-LDA: A user Recommendation based on social connections and interests of users in uni-directional social networks.” In: Proceedings of the International Conference on Web Intelligence, ACM,2017, 260-265.
- [13] Marco Pennacchiotti, Siva Gurumurthy. “Investigating topic models for social media user recommendation.” In Proceedings of International Conference on World Wide Web, ACM, 2011,101-102.
- [14] Hannes OlivierMarc, Waselewsky, Niels Pinkwart “MusicTagger: Exploiting User Generated Game Data for Music Recommendation”, In international conference on human-computer interaction
- [15] “Location-Aware Music Recommendation Using Auto-Tagging and Hybrid Matching” In proceedings of the 7th ACM conference on recommender systems.

PUBLICATION:

Srikar Amara, R. Raja Subramanian, “Collaborating personalised recommender system and content-based recommender systems using TextCorpus”, 6th International Conference on Advanced Computing and Communication Systems (ICACCS), India, 2020, pp, 105-109.



SPONSORED BY



Event by & at :



Sri Eshwar
College of Engineering
An Autonomous Institution
Coimbatore, India



6th

2020

International Conference on
**Advanced Computing &
Communication Systems**

TECHNICAL SPONSORS



Certificate of Presentation

Certify that

Mr./Ms./Dr. Srikar Amara

Kalasalingam Academy of Research and Education, Virudhunagar, India

has presented a paper in the International Conference on
Advanced Computing & Communication Systems - ICACCS 2020
on 6th & 7th March 2020 at Sri Eshwar College of Engineering,
Coimbatore, TamilNadu, India.

Paper Title :

**Collaborating Personalized Recommender System and
Content-Based Recommender System Using Text Corpus**

Dr. H. Anandakumar
Conference Chair

Dr. R. Subha
Convener

Dr. Sudha Mohanram
Patron



SPONSORED BY



Event by & at :



Sri Eshwar
College of Engineering
An Autonomous Institution

Coimbatore, India



6th

2020

International Conference on

**Advanced Computing &
Communication Systems**

TECHNICAL SPONSORS



Certificate of Presentation

Certify that

Mr./Ms./Dr. R. Raja Subramanian

Kalasalingam Academy of Research and Education, Virudhunagar, India

has presented a paper in the International Conference on
Advanced Computing & Communication Systems - ICACCS 2020
on 6th & 7th March 2020 at Sri Eshwar College of Engineering,
Coimbatore, TamilNadu, India.

Paper Title :

**Collaborating Personalized Recommender System and
Content-Based Recommender System Using Text Corpus**

Dr. H. Anandakumar
Conference Chair

Dr. R. Subha
Convener

Dr. Sudha Mohanram
Patron