Project 2. Adaptive Support on Study Genie.

Abhishek Rao, Varun Gaur, Harsha Illuri, Rohith Vajrala
Arizona State University

Abstract—The volume of information that is currently available in the internet is enormous and to get relevant results becomes very important in every student's academic life. It is very crucial to organize domain knowledge and facilitate better exam preparation. Collaborative learning is an educational approach that involves groups of students working together to solve a problem, complete a task, or create a product. Students learn to relate to their peers and other learners as they work together in group enterprises. This can be especially helpful for students who have difficulty with social skills. Adaptive support is a vital requirement in any such environment. The key focus of our application is to build an application where students can share their notes and form study groups and have a better learning experience. We have developed this application with an underlying recommendation system based on both content based filtering and collaborative filtering.

Keywords— Adaptive feedback, Content based filtering, Collaborative filtering, Adaptive navigation, Python NLTK, TF, TF-IDF.

I. INTRODUCTION

The objective of this assignment is to build a system allowing students to share public/private notes, form groups, assemble cheat sheets, search notes on based of topics, and recommend relevant notes from other users who have subscribed to the same topics. The student can browse through all topics and sub topics in the system to discover new notes related to different topics from other users in the system. Each note can be up voted or down voted by other students and the most popular notes can be shown to the user in his/her recommended page. To avoid over fitting, control is also given to the user to change his recommendations. This system as a whole will serve the purpose of a Study Genie which is a one stop shop for finding all the information about different topics.

II. BACKGROUND

Amidst the rapid growth of technology, user profiling i.e information related to all users that web applications have to collect in order to be effective has to be done in an efficient manner. Techniques such as explicit user information collection and implicit user information collection are employed by the applications. In order to build an adaptive system, implicitly-created profiles are very crucial and should be used more often as there is less load on the user. User profiles can be represented using Keyword Profiles (built using keywords extracted via different methods), Semantic Network Profiles (built using keywords which are added to the network of nodes) and Concept Profiles (built using keywords acting as nodes representing abstract topics).

In any adaptive Collaborative learning environment, Open student modeling is very crucial in evaluating the progress of the individual as compared to other users. In the paper, Open social student modeling: visualizing student models with parallel introspective views [1] by I-Han Hsiao, Fedor Bakalov, Peter Brusilovsk and Birgitta König-Ries, we observe that the Parallel Introspective views interface which are implemented as a part of student modeling gives all students not only their own performance overview but also give them a comparison of their results among their peers. These parallel Introspective views provide a better understanding to the student where does he/she stand in the class and they can explore the areas for improvement. Introduction of these parallel introspective views was found to be beneficial for the students and they were able to achieve better success rate.

There are many approaches in building a user model, in the paper User Models for Adaptive Hypermedia and Adaptive Educational Systems [2] by Brusilovsky, P., & Millán, E., we understand how we should maintain the information and what kind of information should be stored. There are two approaches in building a user model, the first one being the overlay approach model which is to represent a user's knowledge as a subgroup of the domain model that resembles proficient knowledge of the subject. The other model is uncertainty based approach where algorithms are developed to deal with uncertain information. It is also suggested that Overlay user modeling is currently the leading user modeling approach in AES and AHS and there is a lot of potential in the future for uncertainty based approach models.

Adaptive presentation of content to be published on the webpage also plays a vital role in effective management of any website. It is very important to first decide on the relevant data to be published and then apply approaches to effectively adapt the content before publishing it on the web which is nothing but better presentation. Content adaptation can be done using approaches such as page and fragment variants. Content presentation can be done on more relevant data or on context. There is a high potential for research in presentations with other types of adaptions and validation of these techniques.

III. Technology Stack

We have used following technologies in our application.

Front End: BootStrap for CSS and UI elements
Model-View-Controller Framework: Django 1.9
Visualizations: D3.js
Database: SQLite3 database.

IV. System Design

This figure depicts the high level architecture of our application. Whenever a user hits our website, he can login using his userId/Password and can see recommended notes on his homepage. He can create a note and click on save and the particular note will be saved in the database. The recommendation system will fetch the notes from the database, calculate TF and TF-IDF of all the notes and recommends the most relevant notes for a particular user on the basis of his interests, topics he has subscribed, content from similar users.

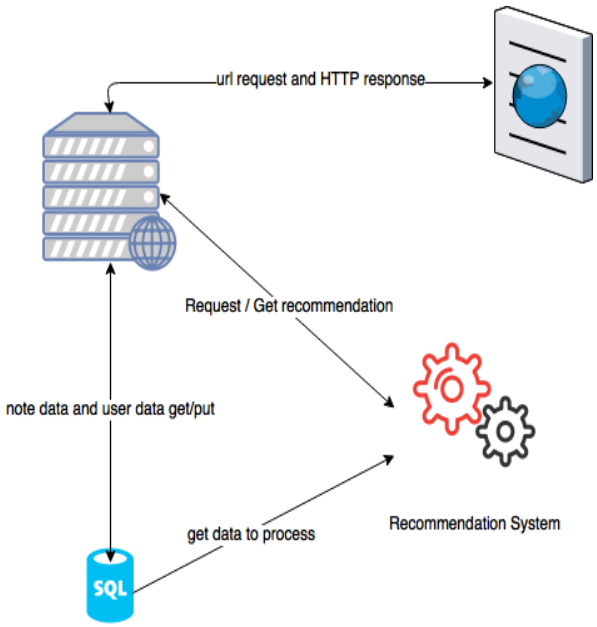

Fig 1. Higher Level Architecture

Database Design:

Figure 2 shoes the hig level architecture of our database schema. When the user registers the auth_user gets populated and the permissions are setup for the user. The data model is optimally designed to eliminate and redundancy. The title and content of all the notes is stored in notes_notes.

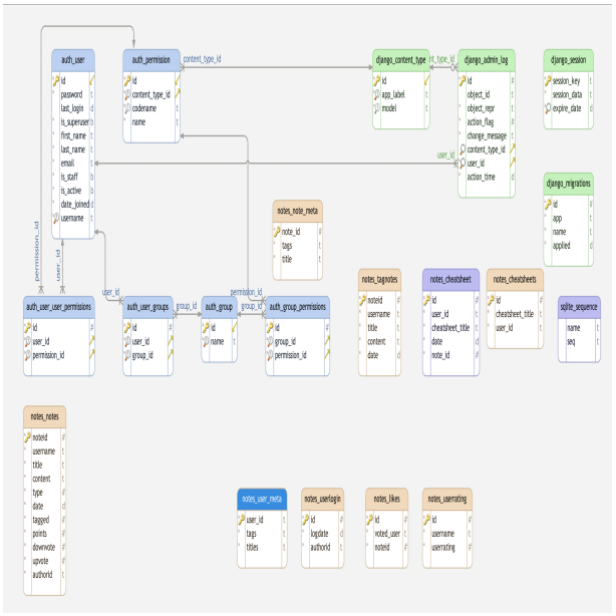

Fig 2. Database Schema

V. Methodology

We have adopted the following models as a part of our application.

User Model:

The user model depicted below is a mix of Implicit and Explicit models. Differential weights are assigned based on interactions of all the components in the system hence capturing more implicit data and also user can have access to his profile and he can add/remove recommended tags which are developed by the system based on his interactions with different modules. This ensures avoiding of over fitting of data and the user can choose to have the recommendations which he wants. Weighted keyboard based user profile has been adopted and different weights has been assigned which will help in efficient capturing of data thus facilitating more effective user model.

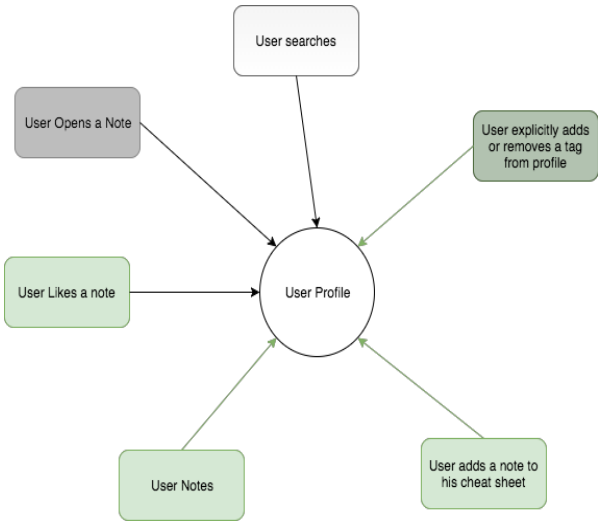

Fig 3. User Model

Whenever a user likes a note or click on up-vote for a note, the content of the note and tag associated with the note

is captured and the information is used in building a user model. Similarly, whenever a user opens a note, searches for a particular note, adds a note to his cheat sheets, all the information is captured in his user model.

Weights Calculation for all Notes:

All the notes are saved in the database with a particular weight associated with it. Whenever a note is created, using Python's Natural Language ToolKit and Scikit-learn library, we calculate TF(term Frequency) and assign tag weights based on their frequency within 66% of total weight. Also to relate the different tags together and perform a better indexing of the data, we perform TF-IDF operation (Frequency–inverse document frequency, a numerical statistic that is intended to reflect how important a word is to a document in a collection) on the note along with all the notes in the database and assign 33% of the total weight to it. This distribution of weightage was found to be more accurate in getting the results for our application.

The below formula depicts how we are calculating the TF and TF-IDF of a particular note:

$$\mathrm{tf}(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

Fig 4. tf and tf-idf calculation

The below figure indicates how we are storing a note in the database and calculating its TF and TF-IDF. Weights are assigned to all the notes before getting stored in the database by calculating TF of the note and TF-IDF of the note against all the already existing notes.

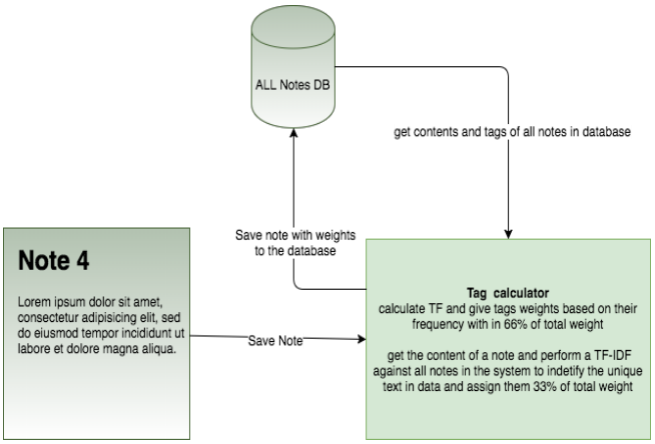

Fig 5.. Weighted Tag Calculation for notes

VI. RECCOMENDATION TECHNIQUES

We have used hybrid of content based filtering and collaborative based filtering in our application for recommending the different notes to the users.

Content based recommendation is used to suggest user's relevant public notes from other users based on the content written in their own notes. The content from the user's notes is captured and it is compared with all the notes in the system and the matching notes are stored in each user's recommendation list.

Collaborative based recommendation is used for recommending notes from similar users. Similar users are found by comparing user profile and weights and topics with all the other users in the system. This kind of recommendation is very important because the user needs to know what are the topics followed by the other users and he has a better understanding of the whole subject. The notes from similar users are shown in user's home page in his recommendation list. We have also used collaborative filtering for visualizing user model while comparing it with similar user.

VII. FEAUTURES OF THE APPLICATION

The user gets redirected to Home once he is signed in to the system. Here his three recent notes are visible along with the recommended notes generated dynamically based on the notes viewed, tagged, liked or deleted by the user using content based filtering and collaborative filtering techniques.

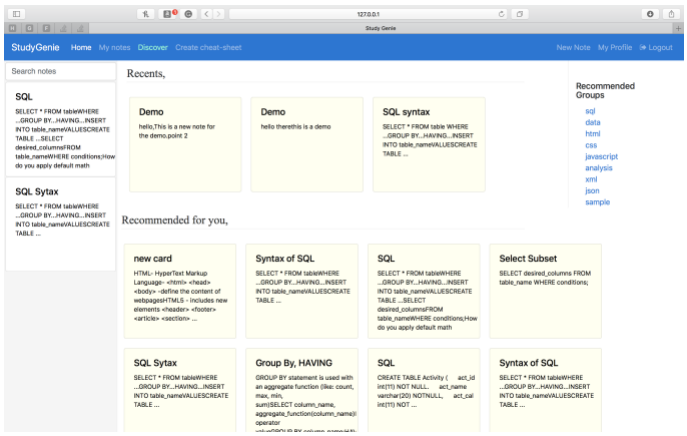

Fig 6. Home page

This is the My notes tab where the user can review his notes and create new notes.

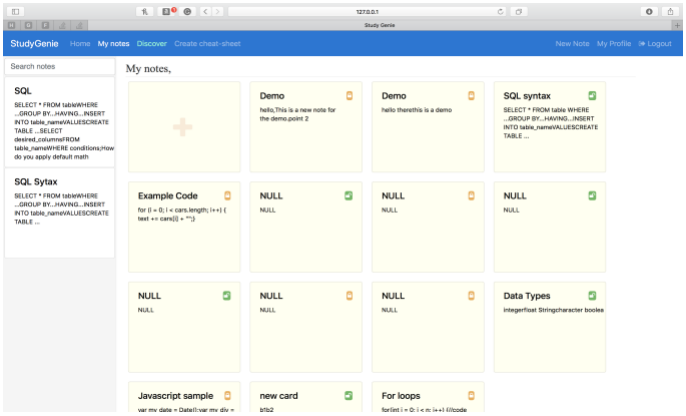

Fig 7. My notes page

When the user clicks on the create note tab, he is presented with a fully features rich text format text editor along with tagged notes on the left. The tagged notes are aimed at helping him create new notes while referring them.

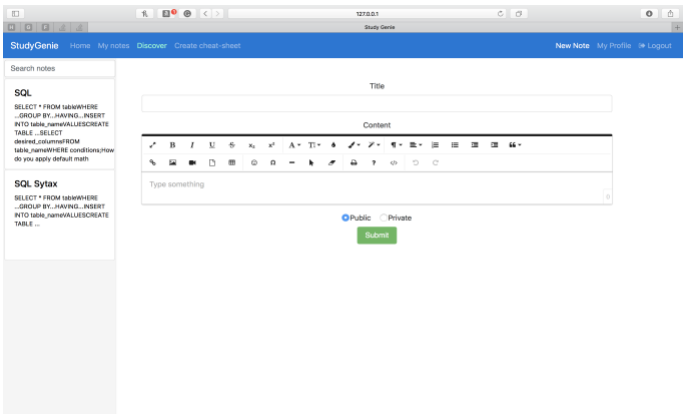

Fig 8. Create Note template

The Discover tab is aimed at helping users explore various publicly available notes using visual adaptive navigation. The user can search the main topics such as SQL, Java or even look for subtopics under it. Once the user searches for the topic the results appear under search results.

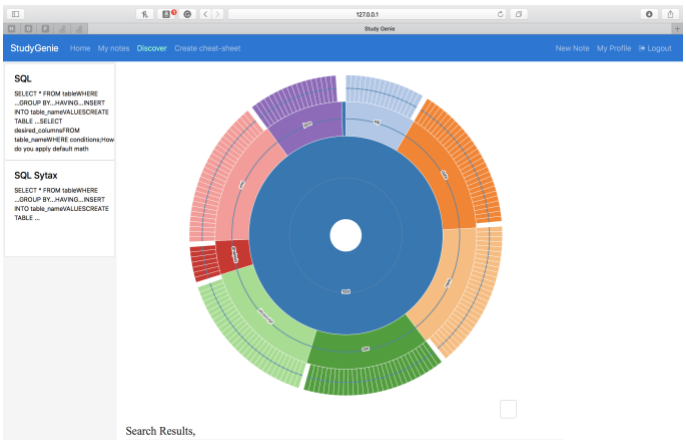

Fig 9. Discovery Page

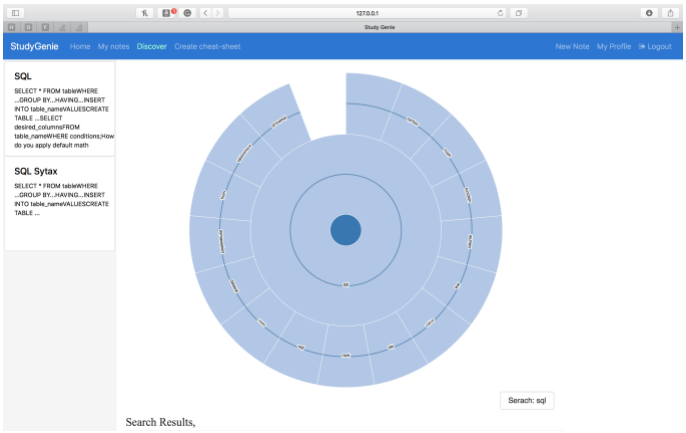

Fig 10. Discovery sub-category on click

Figure 11 shows the sub topic under the sub-category sql and user can search the topic or go back by clicking on the center circle to revert to the previous category view.

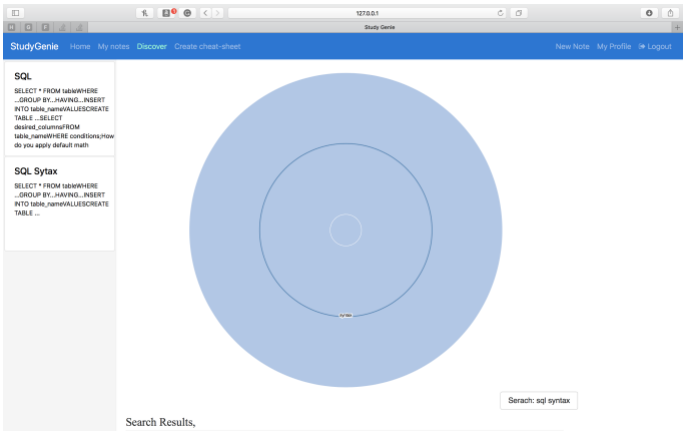

Fig 11. Discovery sub-topic

Figures 9,10 and 11 shows how the visualization changes on clicking the topic of interest.

The note detail view displays the note with its inherent formatting and here the user can like, unlike, tag the note and add it to a cheat sheet. This view also recommends notes similar to it.

The up votes and down votes add credibility to the note as well as increments the user rating on the whole. These factors are factored in while recommending notes to other users. Ones the user adds the notes the cheat sheet it is visualized to show that it is selected. The author name date and time are displayed below the title.

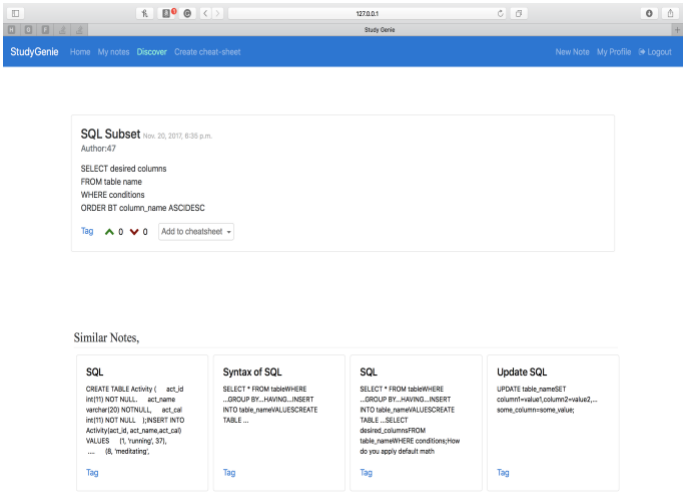

Fig 12. Note view page

In figure 12 we can see the similar notes recommndation based on the content and the title of the displayed note.

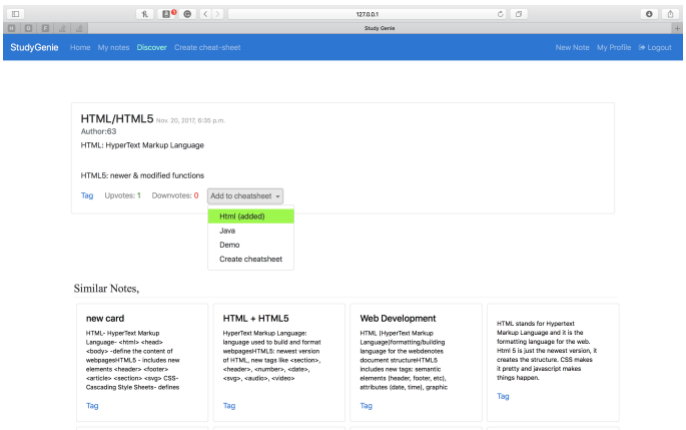

Fig 13. Adding to cheat sheet page

The cheat sheet interface helps the user prepare for his exam by displaying the notes in the cheat sheet in one place. Once the cheat sheet created user can add noted to it.

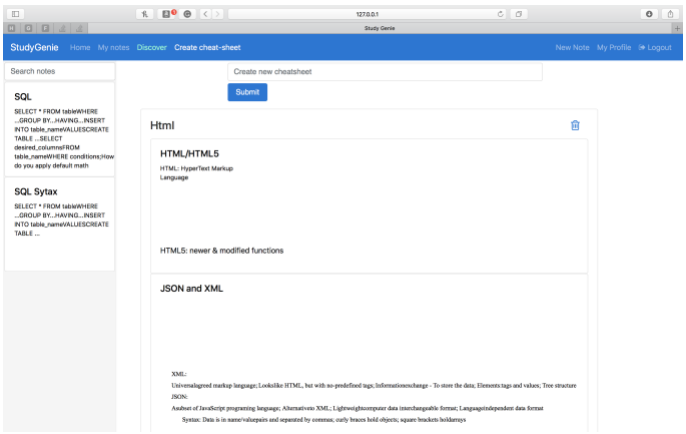

Fig 14. Cheat sheet view

VIII. VISUALIZATIONS

Interactive visualizations are very important in building any stable adaptive web application. We have provided two types of visualization in our system.

Topic Visual Adaptive Navigation.

Users can browse through all the topics and sub topics in the system to discover and read new notes from different users in the system. This visualization is very important for the user to discover topics which he is not familiar with and is present in the system. He can also discover how many users have subscribed to a particular topic and if he finds a hot topic which many users have subscribed to, he can add it to his recommendation lists.

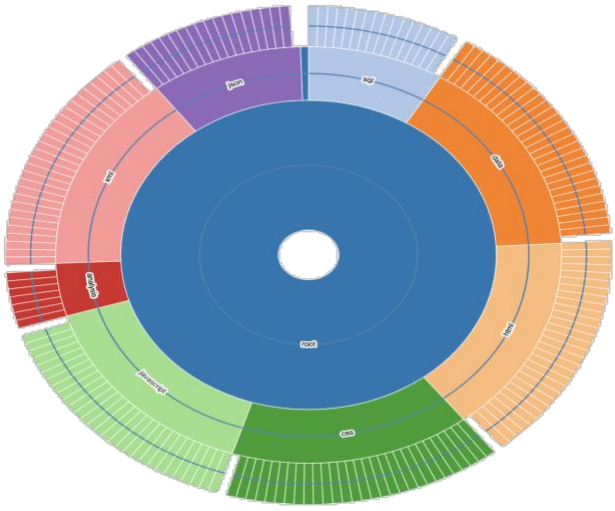

Fig 15. Adaptivel Navigation

User Profile Visual Navigation.

In this visualization, similar users corresponding to a particular user are found and he can see the topics each of them have subscribed to.

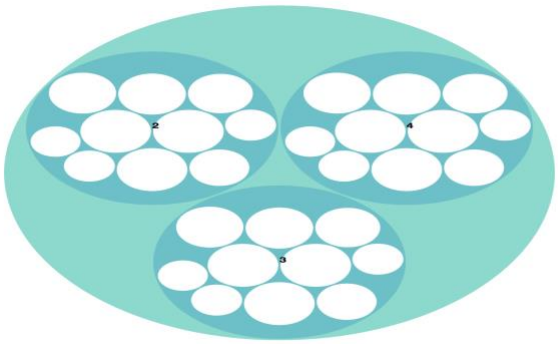

Fig 16. User Profile view

This visualization gives the user an idea of what are different topics his classmates/peers have listed in their profile and he can add that particular topic to his profile. Using this visualization, the user is more informed about the different topics he needs to prepare for in his exam.

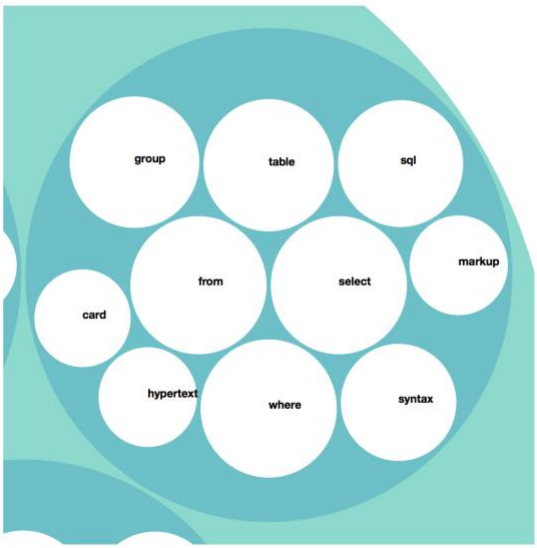

Fig 17. User tags

IX. EVALUATION

All the features of the application were tested before deployment by creating temporary users and having each user subscribe to the particular topic.

We simulated classroom environment by creating a group of similar users who have subscribed to the same topics or almost same topics. For our Recommendation system, TF(term Frequency) is calculated and tag weights are assigned based on their frequency within 66% of total weight, we also calculate TF-IDF operation on the note along with all the notes in the database and assign 33% of the total weight to it. This weightage assignment was found to be close to ideal distribution and we were able to get around 80% of accurate recommendations. For example: For all the notes in the SQL section, we are getting almost all relevant notes. Below is the distribution of the recommended data which we were able to get for SQL topic from different users.

The recommendation system also verifies the upvotes/downvotes of a particular note before adding it to the recommendation lists for any user.

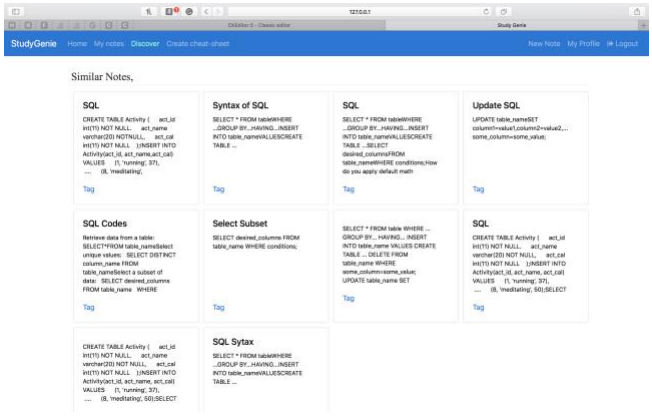

Fig 18. Similar notes recommendation

X. CONCLUSION

An effective study genie having all the functionalities such as adding a note, making a note Public/Private, up voting a note, down voting a note, assembling cheat sheets, form study groups, providing recommendation to the users has been implemented. This system as a whole will serve the purpose of a portal which is a one stop shop for finding all the information about different topics that a user has to prepare for his exam.

Also, we have built a user model which uses implicit information by giving different weights to all the notes depending on the internal implementation and also which uses explicit information to avoid over fitting of data. To calculate weightage of Notes, we are using Python's Natural Language ToolKit and Scikit-learn library. TF(term Frequency) is calculated and tag weights are assigned based on their frequency within 66% of total weight, we also perform TF-IDF operation on the note along with all the notes in the database and assign 33% of the total weight to it. This weightage assignment was found to be close to ideal distribution and we were able to index the data accurately. The main feature of the system is collaborative filtering which we have incorporated to recommend different notes to the user. Effective user modeling coupled with collaborative recommendation techniques makes the system more precise and helps the student in better preparation for the exams. The system also includes interactive and adaptive visualizations which will help the student to monitor the different topics in the system and have a better idea of different topics.

XI. DISCUSSIONS AND FUTURE WORK

Our system has a lot of scope for improvements as well, we can have handwritten notes also as a part of notes and can have a image capturing feature where an image of that handwritten note can be added to the notes db. We can also extend the system to have chat discussion forums for all the study groups where every member can chat with any member if he has any questions on the other user's notes. We can have website recommendations associated with each

note which has more information if the student wants to dive deep and understand more about the topic. In the cheat sheets section, we can have a print option where a user once he has assembled the cheat sheets can print them and take it directly to the exam hall. Also, all sorts of formatting options while printing the document can also be implemented.

XII. ACKNOWLEDGMENT

We would like to thank our professor to Dr. Sharon Hsiao for her guidance and support throughout the project. She always provided necessary information regarding the project and encouraged us to incorporate adaptive features in our project.

XIII. REFERENCES

[1] Hsiao, I. H., Bakalov, F., Brusilovsky, P., & König-Ries, B. (2011). Open social student modeling: visualizing student models with parallel introspectiveviews. In User Modeling, Adaption and Personalization (pp. 171-182). Springer Berlin Heidelberg.

[2] Brusilovsky, P., & Millán, E. (2007, January). User models for adaptive hypermedia and adaptive educational systems. In The adaptive web (pp. 3-53).Springer-Verlag.

[3] Bunt, A., Carenini, G., & Conati, C. (2007). Adaptive content presentation for the web. In The adaptive web (pp. 409-432). Springer Berlin Heidelberg.

[4]Python NLTK documenetation - http://www.nltk.org/

[5]Language processing and python NLTK- www.nltk.org/book/ch01.html

[6] NLP libraries - https://elitedatascience.com/python-nlp-libraries