# COURSE FEEDBACK SYSTEM

SE Team – K

Implementation - 1 : FORUM

**Description:**

This is our first proposed solution for our project Course Feedback system. The analysis of the questionnaire and survey with the users revealed that they lack sources to find information and reviews of the course, and desired a public platform with which they can get feedback from students who had already taken the course. This proposed solution focuses on providing feedback and reviews for a particular course under specific Professor purely on the reviews of the users. The users here are the students who have already studied the course and the students who plan to enroll in this course. Every user can login to our Feedback application through their credentials, choose the Forum path, by selecting the courses from the available options and the Professor under whom the course is offered. This selection makes the user enter a unique channel of Course under a definite professor. The users who had already studied the course will post their reviews regarding the course which may include description about the projects offered under the course, Course material, etc. Each review has a like and dislike button to support that particular review, and the review with maximum likes will be highlighted above and so on with decreasing order with decreasing number of likes. The users who are the students willing to take the course can see these reviews in the order of likes and post their doubts in the same forum, so that they get useful feedback regarding their queries, from senior students.

**Done**

We implemented our Course Feedback system by Ruby on Rails. The home page of our application, like any other website as authentication/login information. Each user has his unique profile. Once the user logins he is given three choices which here are our three solutions Forum, Course Evaluation and Course Suggestion. On selecting Forum the user is directed to our first solution which resembles to live-chat application. Our basic implementation is successful where the user can post his reviews. The review of each user is stored in the database with unique user id. Each review has a username associated with it, hence any other user can visit his profile through it and contact him personally as well.

**Doing**

As it is like a public forum we have not restricted any user with any kind of reviews. The main aim of this forum is to provide as accurate as possible information to the students. There may be a case where a review might be irrelevant, to categorize this we are introducing the features likes and dislikes associated with each review. The reviews with maximum likes are displayed above and further displayed with decreasing order of likes. The database table maintains the number of likes and dislikes which helps us high-light the important reviews.

**Road Blocks**

The potential roadblock is we wish to limit the number of reviews and display the reviews that are recent or has maximum likes. It is required as there can be a large number of reviews.

Implementation - 2 : Course evaluation

**Description:**
This approach aims to provide an answer to as many questions a student can get while selecting a course. If a user chooses this approach he gets navigated to a page where he can either take a feedback or provide a feedback through a form. The senior students who had already taken the course and willing to provide useful feedback can fill the form. Our goal is to use this information from form and generate an average result of the answers from the form. So when the user choose to get feedback, he will get a result of answers to possible dilemmas while choosing a course. Results can be in a form a pie-chart or graph. The feedback form will include questions that can cover information as Course material, strength of professor, quality of projects offered, what category of subject the course fall into,etc.

**Done**
We have completed the part of implementing a form consisting of 12 questions. On selecting the course evaluation approach, the user is asked to enter Course code and Professor name, which in-turn display two options, either to give a feedback or get a feedback. The give_feedback part successfully collects all the data, by authenticating the user. We maintained the answers in the database as the aim is to display an average result when the student queries a get_feedback.

**Doing**
We are currently working on how to display the results to the user when he selects 'get_feedback'. We are working on the implementation of the visualizations and designs of the results of the feedback. Once the user receives the feedback we are implementing a way that the user gets redirected to the forum where we can discuss his doubts in the particular course channel.

**Roadblock**
One of the aim of the project was to provide information about existing course evaluation techniques like ratemyprof and course description on NCSU. We are encountering difficulties in gathering data from this websites and thinking a way where we can easily make available this existing data in our feedback application as well.

Implementation - 3 : Course suggestion

**Description:**

When choosing courses for the incoming semester, many students have their specific requirements. Some students want to learn more about JAVA in the incoming semester, while some students want to do more projects in the incoming semester. In this situation, we think that a course suggestion system is very helpful for students who have their own requirements on the courses which they will take in the incoming semester.

In our course suggestion system, we provide multiple filters and each filter have multiple options for users to choose. For example, the filter "programming language/tools/framework" have these options: Java, C, C++, Python, Ruby on rails, Git, language independent. If a student want to know which course have Java programming part, then he/she can choose the option "Java" and our system will provide him/her the results. And if the user chooses options in different filters, our system will provide him/her intersecting results. For example, if a user chooses "C++" for the "programming language/tools/framework" filter and chooses "2" for the "Number of exams" filter, then our system will provide him/her the courses which have 2 exams and have C++ programming part.

All of the data which is needed to support the functions in this solution comes from our 2nd solution -- "course evaluation". We have totally 8 filters, which include "interesting rate" filter, "Job/career opportunities" filter, "workload" filter, "Average grade" filter, "programming language/tools/framework" filter, "Course category" filter, "Number of projects" filter and "Number of exams" filter.

In the home page, clicking the "course suggestion" button will redirect the browser to our course suggestion page. The choosing action for each filter of the user will be record in specific variables, and once the user clicks "search" button, we will use the records of the user's choices to select corresponding courses from the database. And then we show the results to the user.

**Done**

The redirection to Course suggestion solution from the home page and other two implementations is successful. We have implemented the eight filters successfully which gives all the options for each filter in a drop-down menu form.

**Doing**

In our second implementation course evaluation we used questions like workload, number of projects in a course in the feedback form, so that we could use this data in the filter approach and give the accurate intersecting result when a student makes selection of his choices through filters. Current we are working on making and maintaining this database such that it queries the right course with proper intersection of filters.

**Roadblock**

The selection of filters will force the application to query course with very accurate combination of filters, sometimes it may not exist, so we plan to extend it to a range of choices.