A MINOR PROJECT REPORT

on

COURSE RECOMMENDER SYSTEM

Submitted in Partial Fulfillment for the Award of the

Degree of

Bachelor of Technology

in

Information Technology

by

Saurabh Kumar

Prem Kumar

Dileep Bhagat

Brajesh Kumar

Under the supervision of

Anil Kumar Dudyala

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

राष्ट्रीय प्रौद्योगिकी संस्थान पटना
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

CERTIFICATE

This undersigned certify that **Saurabh Kumar** Roll No. **147014, Prem Kumar** Roll No. **1407026, Dileep Bhagat** Roll No. **1407034** and **Brajesh Kumar** Roll No. **1407031** have carried out the project entitled " Course Recommender System " as their 7th semester minor project under my supervision.

…….

Prof. Anil Kumar Dudyala

Assistant Professor (Supervisor)
Department of Computer Science and Engineering
National Institute of Technology Patna

…….

Dr. Prabhat Kumar

Head of Department
Department of Computer Science and Engineering
National Institute of Technology Patna

राष्ट्रीय प्रौद्योगिकी संस्थान पटना
NATIONAL INSTITUTE OF TECHNOLOGY PATNA

DECLARATION

We hereby declare that this project work entitled **"Course Recommender System"** has been carried out by us in the department of **Computer Science & Engineering** of **National Institute of Technology Patna** under the guidance of **Anil Kumar Dudyala**. No part of this work has been submitted for the minor project of degree or diploma to any other institute.

NAME	ROLL NO	SIGNATURE
Saurabh Kumar	1407014	
Prem Kumar	1407026	
Dileep Bhagat	1407034	
Brajesh Kumar	1407031	

Date: __/__/____

ACKNOWLEDGEMENT

We would like to take this opportunity to thank all our sources of aspiration during the course of this project.

First and foremost we are grateful to **Anil Kumar Dudyala**, who gave an opportunity to develop a Course Recommender System and for his continuous support during the project and for his patience, motivation and immense knowledge. He helped us come up with project topic and guided us over almost a semester of development. And during the most difficult time when writing this report, he gave us the moral support and the freedom to move on.

We hereby take the privilege to express my gratitude to all the people who directly or indirectly involved in the execution of this work, without whom this project would not have been a success.

We are also thankful to our senior of M.Tech for his valuable guidance support and cooperation extends by him. Then we would like to thank our project team member for their kind cooperation, help and never ending support.

We are also thankful to NIT Patna for providing us technical skills and facilities which proved to be very useful for our project.

ABSTRACT

Course Recommender System

Recommender Systems are subclass of information filtering systems which predict the rating or preference that user would give to an item. Nowadays, many websites uses recommender systems like- Amazon, Netflix, Quora, StackOverflow etc to recommend different items like books, movies, courses, web-pages etc.

Among the challenges that the students face in day to day life, one of the most unavoidable challenge is lack of awareness about the elective courses as there are huge number of available courses. So, taking decision become more confusing when different seniors/colleagues suggest different courses which lead to selecting wrong courses.

As the range and diversity of different elective courses available for selection have increased, course recommendation systems help students in making choices about courses have become more relevant. In this project, we extend the concept of collaborative filtering approach to develop a course recommendation system. The proposed approach provides student prediction of the top three relevant courses which will be helpful when they decide on selecting elective courses, as these are the important parameters for a student while deciding on an elective course. We experimentally evaluate the collaborative filtering approach on a real life data set and show that the proposed system is effective in terms of accuracy.

CONTENTS

Chapter 1-INTRODUCTION

1.1 PROJECT DESCRIPTION

Course Recommender System increased choices and flexibility and allows students the ability to personalize their studies, challenges have arisen with regard to enabling students to appreciate the range and diversity of courses in each term or semester that are available to them. In particular, the current enrolment system makes it difficult for students to locate course options that might best fit their individual niche interest. Due to the diversity of different electives available, students find it difficult and time consuming to select the courses they will like and at the same time can get relatively better grades.

Students pursuing higher education degrees are faced with two challenges: a myriad of courses from which to choose, and a lack of awareness about which courses to follow and in what order. It is according to their friends and colleagues recommendations that the many of them select their courses and register accordingly. It would be useful to help students in finding courses of interest by the intermediary of a recommender system. Recommender systems implement advance data analysis techniques to help users find the items of their interest by producing a predicted likeliness score or a list of top-N recommended items for a given active user. Item recommendations can be made using different methods where each method is having different results.

Collaborative filtering (CF) based algorithms provides item recommendations or predictions based on the opinions of other like-minded users. In other domains, the benefits of deploying recommendation systems to assist users in finding relevant items is well understood and researcher are finding different usage of recommender systems in generating recommendations for different category of items . More recently, research has been conducted into developing such technology for course recommender systems. We believe collaborative filtering approach provides student an accurate prediction of the elective courses based on their performance in earlier courses.

1.2 MOTIVATION

A Course Recommendation System helps students to select some elective courses from huge number of available courses accurately. Since, selecting the elective courses in huge number of available courses become more confusing when different seniors or colleagues suggest different courses so they generally select wrong courses.

Chapter 2-RELATED WORK

From the last decade, Recommendation System have been widely developed, implemented and accepted for various categories of application like recommendation of products (e.g., books, music, movies) and of services (e.g., restaurants, hotels, websites), likewise research has been conducted into developing such technology for course recommender systems.

Student Course Recommender (SCR), suggests courses by using a strategy based on Bayesian Network Modeling. The SCR network learns from the information stored about the students who have used the system. It requires the presence of enough cases in the student database. Therefore, if a user has not started or completed any courses, and is not pursuing any degree at the university, SCR cannot give him any course recommendation.

The Course Recommender System is based on the several different collaborative filtering algorithms like user-based, item-based. The system can predict the usefulness of courses to a particular student based on other users' course ratings. To get accurate recommendations, one must evaluate as many courses as possible. Based on the evaluation results, it is one of the best algorithms for course recommendation. The system cannot predict recommendations for students who have not taken any courses at the college.

Academic Advisor Course Recommendation Engine (AACORN system), applies a case-based reasoning approach to course recommendation. The AACORN system recommends courses to graduate students at De-Paul CTI. The system uses the experience of past students and their course histories as the basis for course advising. In order to determine the similarities between course histories, the system uses a metric commonly used in bio-informatics called the edit distance. The system requires a partial history of the courses followed by a student before it can provide useful recommendations.

CourseAgent is a community-based recommendation system that employs a social navigation approach to deliver recommendations for courses based on students assessment of their particular career goals. The main theme of this approach is to obtain students explicit feedback implicitly, as part of their natural interaction with the system. The basic and obvious benefit of the system to the students is as a course management system that keeps information about courses they have taken and facilitates communication with their advisors.

Personalized E-Learning system(PEL-IRT) using item response theory. It recommends appropriate course material to students, taking into account both course material difficulty and student ability. When using PEL-IRT, students can select course categories and units and can use appropriate keywords to search interesting course material. Once the course material has been recommended to students and they have browsed through, the system asks them to answer two questionnaires. This explicit feedback is used by PEL-IRT to re-evaluate the students abilities and adjust the course material difficulty used in the recommendation.

The RARE, a course recommender system based on association rules combines association rules together with user preference data to recommend relevant courses. RARE was used on real data coming from the department of Computer Science at the Universite de Montreal. It analyses the past behavior of students concerning their course choices. More explicitly, it formalizes association rules that were implicit before. These rules enable the system to predict recommendations for new students. A solution to the cold start problem, which is a central question in recommender systems, is also proposed in RARE.

Course Recommender System is based on variation on the widely-used item-based collaborative filtering algorithm. The objective of module recommender system is to facilitate and enhance on-line module selection process by recommending elective modules to students based on the core modules that they have selected. Evaluation using historical enrolment data shows very encouraging performance in terms of both recall and coverage.

Some recent research is focused on using course recommender systems in niche area like for civil engineering professional courses. From the review of the literature, it is evident that recommendation technology applied in education field can facilitate the teaching and learning processes. Considering the significance and seriousness of education, the help of recommendation system can improve efficiency and increase veracity of learners in the actual situation. Comparing to other approaches like SCR based on bayesian network modeling, RARE based on association rules, and AACORN based on case-based reasoning, the proposed approach uses collaborative filtering as in Course Recommender System but using students' grades that is indicator of performance in earlier courses. The other systems like PEL-IRT and CourseAgent are explicit feedback based system but the proposed approach in this paper does not need any feedback from students. Given the challenges and constraints of integrating this technology into an existing live environment, the proposed work is in its initial stages but the vast literature suggests that this domain offers great potential and scope for future research and development.

Chapter 3-REQUIREMENTS ANALYSIS

3.1 REQUIREMENTS GATHERING

As soon as the project idea is confirmed, we have started working on the requirements for the implementation of the project. The idea is to develop a **Web based Recommender System** that can receive the roll no & password of the student and recommends the elective courses to him/her. We did some research on current technologies that are used in industry and decided working on **Machine Learning** approaches to solve above problem. We learn various types of Recommender System algorithms and choose the **User-based Collaborative Filtering** approach for implementation of this project. Also, to provide web-support to python code, we learn the **Python-CGI Programming**.

3.2 REQUIREMENTS SPECIFICATION

These are the technical requirements to develop **Course Recommender System Web Application**:

3.2.1 SOFTWARE REQUIREMENTS

Operating System: Linux

IDE: Jupyter Notebook

Front End: HTML5 and CSS

Backend: Python

Web Server: Apache2

3.2.2 HARDWARE REQUIREMENTS

Processor: Intel Core 2 Duo or Higher

RAM: 2GB or higher

3.2.3 Recommender System Algorithms

3.2.3.1 Collaborative Filtering Algorithms

Collaborative filtering (**CF**) is a technique used by recommender systems. Collaborative filtering has two senses, a narrow one and a more general one.
In the newer, narrower sense, collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person *A* has the same opinion as a person *B* on an issue, A is more likely to have B's opinion on a different issue than that of a randomly chosen person. For example, a collaborative filtering recommendation system for television tastes could make predictions about which television show a user should like given a partial list of that user's tastes (likes or dislikes). Note that these predictions are specific to the user, but use information gleaned from many users. This differs from the simpler approach of giving an average (non-specific) score for each item of interest, for example based on its number of votes.

In the more general sense, collaborative filtering is the process of filtering for information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc. Applications of collaborative filtering typically involve very large data sets. Collaborative filtering methods have been applied to many different kinds of data including: sensing and monitoring data, such as in mineral exploration, environmental sensing over large areas or multiple sensors; financial data, such as financial service institutions that integrate many financial sources; or in electronic commerce and web applications where the focus is on user data, etc. The remainder of this discussion focuses on collaborative filtering for user data, although some of the methods and approaches may apply to the other major applications as well.

Further, there are several types of collaborative filtering algorithms:

1. **User-Based Collaborative Filtering:** Here we find look alike customers (based on similarity) and offer products which first customer's look alike has chosen in past. This algorithm is very effective but takes a lot of time and resources. It requires to compute every customer pair information which takes time. Therefore, for big base platforms, this algorithm is hard to implement without a very strong parallelizable system.

2. **Item-Based Collaborative Filtering:** It is quite similar to previous algorithm, but instead of finding customer look alike, we try finding item look alike. Once we have item look alike matrix, we can easily recommend alike items to customer who have purchased any item from the store. This algorithm is far less resource consuming than user-user collaborative filtering. Hence, for a new customer the algorithm takes far lesser time than user-user collaborate as we don't need all similarity scores between customers

3.2.3.2 User-Based Collaborative Filtering

User-based collaborative filtering was first introduced by GroupLens research systems to provide personalized predictions for Usenet news articles. The basis implementation details of user–based CF remains the same as proposed. CF systems are primarily used to solve the prediction problem or the top-N prediction problem. For an active user *Ua* in the set of users *U*, the prediction problem is to predict the rating active user will give to an item *It* from the set of all items that *Ua* has not yet rated. The steps followed in user-based CF to make a prediction for user *Ua* are as follows:

Step 1: Similarity between the active user *Ua* and every other user is calculated.
Step 2: Based on their similarity value with user *Ua*, set of *k* users, most similar to active user *Ua* is then selected.
Step 3: Finally, prediction for item *It* is generated by taking the weighted average of the ratings given by k-similar users.

In step 1 to calculate the similarity between users Pearson-r correlation coefficient is used. Let the set of items rated by both users *u* and *v* be denoted by *I*, then similarity coefficient (*Simu,v*) between them is calculated as:

$$Sim_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r_u})(r_{v,i} - \bar{r_v})}{\sqrt{\sum_{i \in I}(r_{u,i} - \bar{r_u})^2} \sqrt{\sum_{i \in I}(r_{v,i} - \bar{r_v})^2}}$$

Fig. 3.1

Here *ru,i* denotes the rating of user *u* for item *i*, and ru is the average rating given by user *u* calculated over all items rated by *u*. Similarly, rv denotes the rating of user *v* for item *i*, and rv is the average rating given by user *v* calculated over all items rated by *v*. In some cases to calculate similarity cosine vector similarity is used.

Once similarities are calculated, a set of users most similar to the active user *Ua* are selected in step 2.

In step 3 to compute the prediction for an item *i* for target user *u*, an adjusted weighted formula is used to take into account the fact that different users have different rating distributions.

3.2.3.3 APACHE2 WEB SERVER

Apache is the most widely used web server software. Developed and maintained by Apache Software Foundation, Apache is open source software available for free. It runs on 67% of all web servers in the world. It is fast, reliable, and secure. It can be highly customized to meet the needs of many different environments by using extensions and modules. Most Word Press hosting providers use Apache as their web server software. However, Word Press can run on other web server software as well.

It is a modular, process-based web server application that creates a new thread with each simultaneous connection. It supports a number of features; many of them are compiled as separate modules and extend its core functionality, and can provide everything from server side programming language support to authentication mechanism. Virtual hosting is one such feature that allows a single Apache Web Server to serve a number of different websites.

What is Web Server?

Well, a web server is just like a restaurant host. When you arrive in a restaurant, the host greets you, checks your booking information and takes you to your table. Similar to the restaurant host, the web server checks for the web page you have requested and fetches it for your viewing pleasure. However, A web server is not just your host but also your server. Once it has found the web page you requested, it also serves you the web page. A web server like Apache, is also the Maitre D' of the restaurant. It handles your communications with the website (the kitchen), handles your requests, makes sure that other staff (modules) are ready to serve you. It is also the bus boy, as it cleans the tables (memory, cache, modules) and clears them for new customers.

So basically a web server is the software that receives your request to access a web page. It runs a few security checks on your HTTP request and takes you to the web page. Depending on the page you have requested, the page may ask the server to run a few extra modules while generating the document to serve you. It then serves you the document you requested. Pretty awesome isn't it.

3.2.3.4 PYTHON-CGI PROGRAMMING

What is CGI

- The Common Gateway Interface, or CGI, is a standard for external gateway programs to interface with information servers such as HTTP servers.

Web Browsing

To understand the concept of CGI, let us see what happens when we click a hyper link to browse a particular web page or URL.

- Your browser contacts the HTTP web server and demands for the URL, i.e., filename.
- Web Server parses the URL and looks for the filename. If it finds that file then sends it back to the browser, otherwise sends an error message indicating that you requested a wrong file.
- Web browser takes response from web server and displays either the received file or error message.

CGI Architecture Diagram

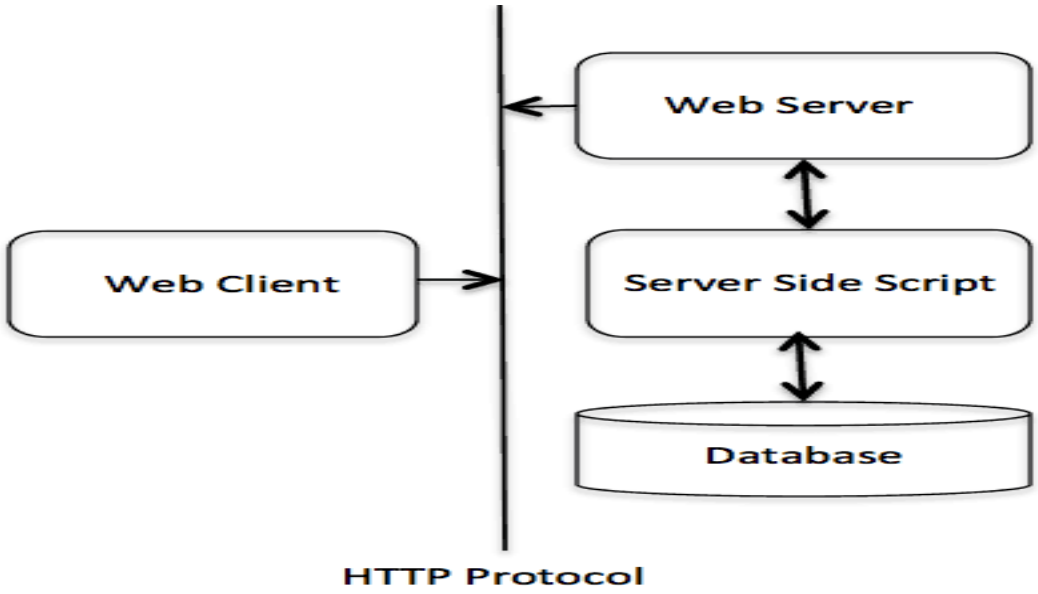


Fig. 3.2

Web Server Support and Configuration

Before you proceed with CGI Programming, make sure that your Web Server supports CGI and it is configured to handle CGI Programs. All the CGI Programs to be executed by the HTTP server are kept in a pre-configured directory. This directory is called CGI Directory and by convention it is named as /var/www/cgi-bin. By convention, CGI files have extension as. **Cgi,** but you can keep your files with python extension **.py** as well.

By default, the Linux server is configured to run only the scripts in the cgi-bin directory in /var/www. If you want to specify any other directory to run your CGI scripts, comment the following lines in the httpd.conf file –

```
<Directory "/var/www/cgi-bin">
   AllowOverride None
   Options ExecCGI
   Order allow,deny
   Allow from all
</Directory>

<Directory "/var/www/cgi-bin">
Options All
</Directory>
```

Fig. 3.3

3.2.3.5 DATA COLLECTION

Dataset is the most important part of any artificial intelligence model , in fact accuracy of any predicting model or any recommender system is totally depending upon the data we trained to the model. More the dataset accurate more predicting accuracy of model would be high. Keeping this fact in mind we collected our dataset accurately from each user. We visited to each student and asked many relevant questions like their grades, their point of view about the subject, how much relevant it would be for job purpose, how much it would be relevant for future research field, how much the particular subject would be scoring if a junior would select that subject and recorded their answer. Sample data is shown in **table 1.**

There are 10 columns for each record which describe each subject which is taken by the each user. Description of each column is given below-

STUDENT_ID- Institute roll number of student.

COURSE_CODE- Code of the course set by the institute.

COURSE_TITLE- Name of the course.

CREDIT- Credit of the subject (actually credit measure the importance of the subject, 4 is high 3 is slightly lesser)

FACULTY_RATING- faculty rating is rating given by the student to the faculty assigned to that subject on a scale of 0 to 10.

JOB_ORIENTED- job oriented describe how much subject is relevant for the job purpose i.e. how much subject would be helpful for the future career.

GRADE- Grade column is the grade which is get by the student in that subject.

SCORING- It describe how much subject is scoring in his point of view**.** i.e if anyone would choose the subject weather he would able to score or not. It is on a scale of 0 to 2.

PLACEMENT_TYPES- It is the users experience how much the subject is important for the placement purpose. It is the rating on the scale of 0 to 2.

RESEARCH_PURPOSE- This column describes how much subject is important for research field. It is a rating on a scale of 0 to 2.

Student_ID	*Course_Code*	*Course_title*	*Credit*	*Faculty_rating*	*Job_oriented*	*Grade*	*Scoring*	*Placement_types*	*Research_Purpose*
1407001	1EC101	Elements Of Electronics Engg	4	9	5	5	0	1	2
1407001	1MA101	Engineering Mathematics-I	4	9	8	10	2	2	0
1407001	2PH101	Engineering Physics	4	9	4	9	2	0	2
1407001	2CS101	Introduction to Computing	3	9	5	7	1	1	0
1407001	3CS105	Object Oriented Methodology	4	6	3	8	2	0	0
1407001	3MA106	Discrete Mathematical Structure	3	8	10	9	2	2	0
1407001	3CS103	Data Structures and Algorithms	4	8	2	5	0	0	1
1407001	4CS107	Computer Organization & Architecture	4	10	4	9	2	0	2
1407001	4CH104A	Green Technologies(Environmental Science)	3	6	6	6	1	1	1
1407001	4CS109	Database Management Systems	4	5	8	8	2	2	1
1407001	5CS115	Operating Systems	4	7	5	9	2	1	1

Fig. 3.4

Chapter 4-SYSTEM DESIGN

4.1 UML DIAGRAMS

UML diagrams are used to explain the design of the system. Once the requirement gathering is completed, system design is done using Unified Modeling Language (UML). UML plays an important role in designing software by using graphical notations to depict the design of the system.

4.1.1 Use Case Diagram

In its simplest form, a use case can be described as a specific way of using the system from users (actors) perspective. A more detailed description might characterize a use case as:

- A pattern of behavior the system exhibits
- A sequence of related transactions performed by actor and system
- Delivering something of value to the actor

Use Cases provide a mean to:

- Capture system requirements
- Communicate with end users and domain experts
- Test the system

The User of the system is a Student, who provide the roll and password and view all the recommended courses and take any two elective courses for that particular semester. Use Case diagram is shown as below:

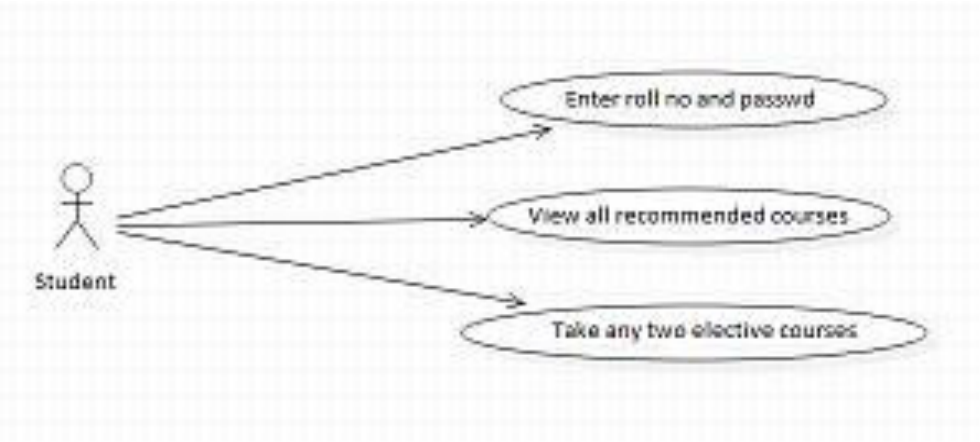


Fig. 4.1

4.2 PROPOSED MODEL

System Design must be based on below shown model:

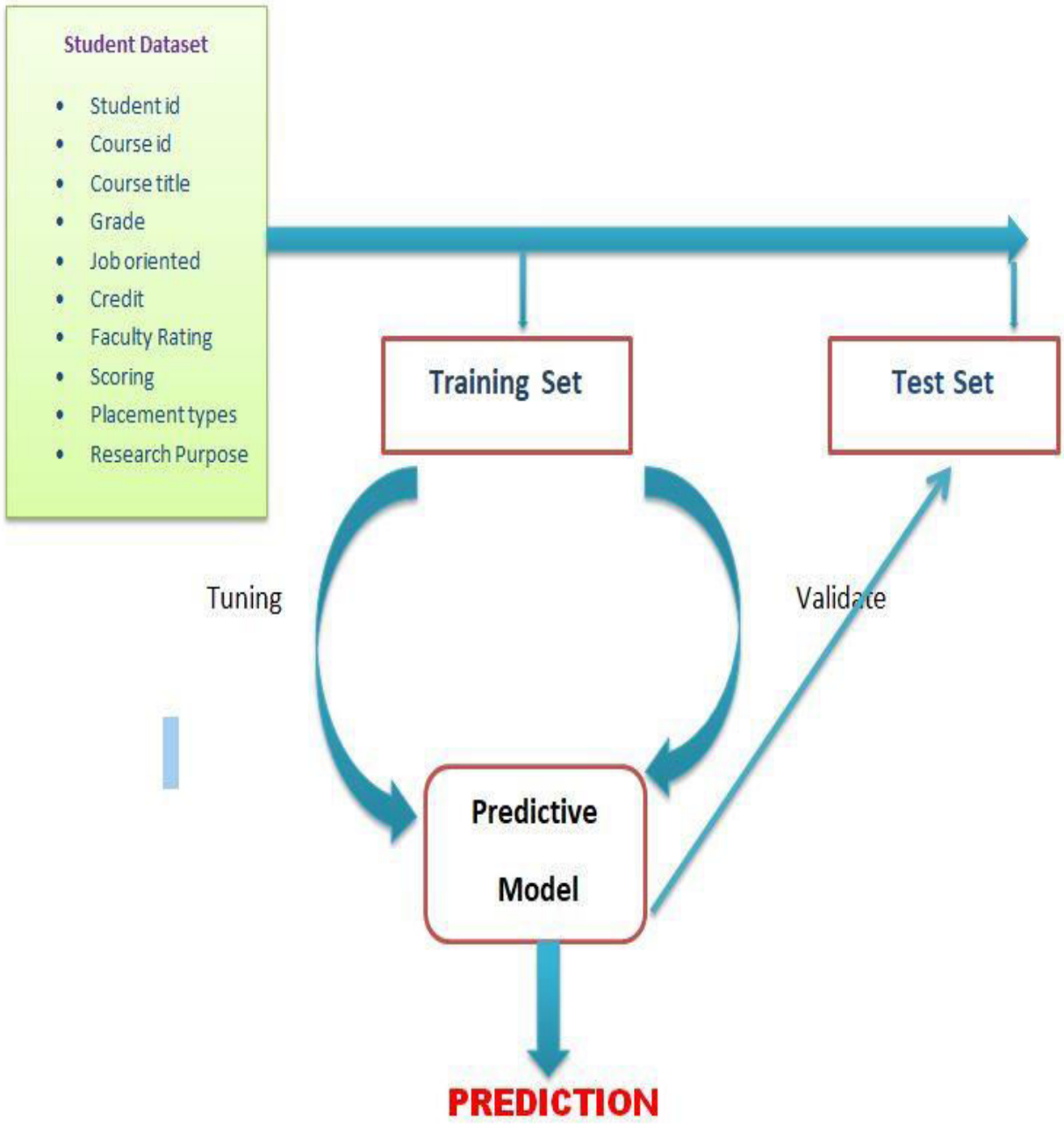


Fig. 4.2

4.3 FLOW CHART

Flow Chart of working model should be worked as below (user's perspective):

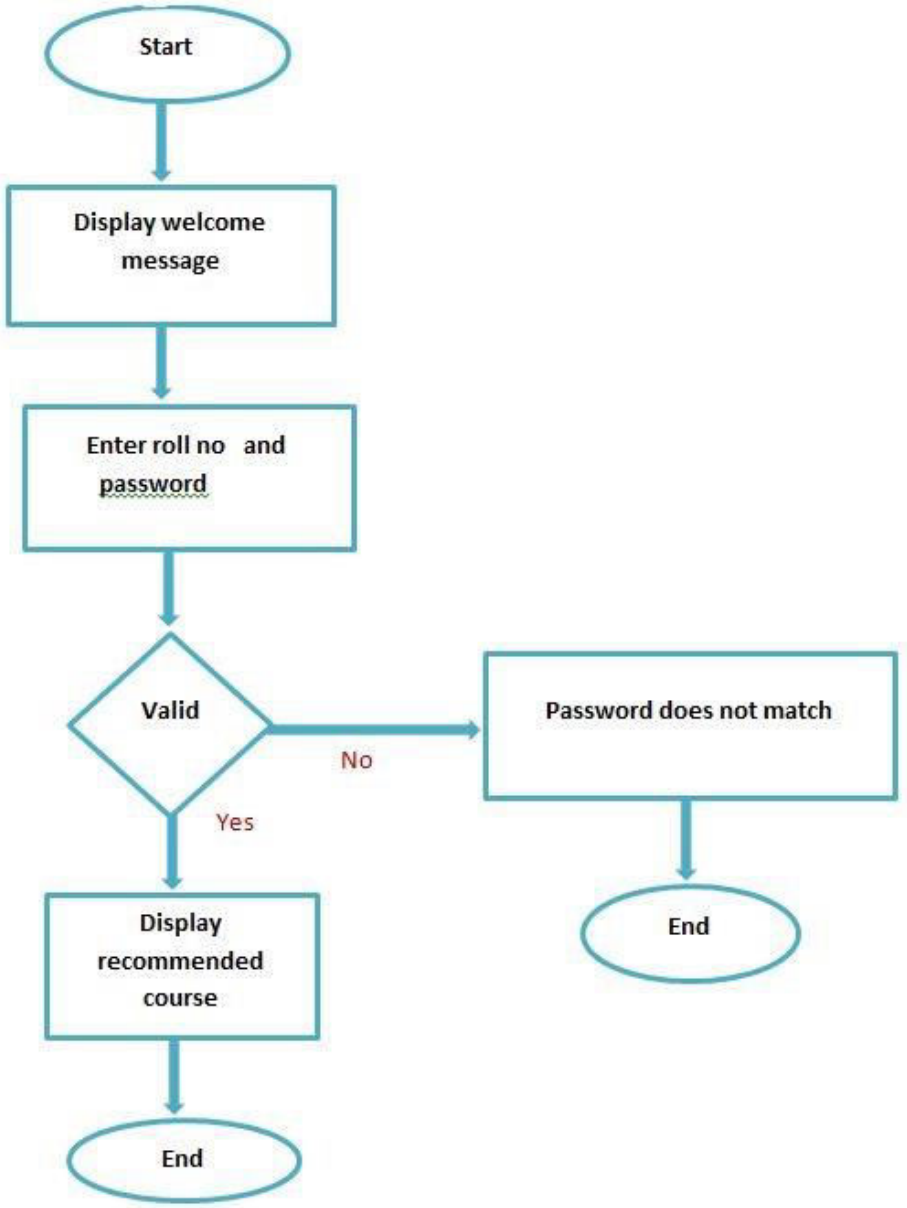


Fig. 4.3

Chapter 5-IMPLEMENTATION

Course Recommender System is a web application that is developed to help students to take elective courses. The main objective of this project to recommend courses to new students based on the previous data set of students.

5.1 CODING

5.1.1 CLIENT-SIDE CODE

Below shown codes are written on index.html which is the homepage of Apache2 web server.

```html
<html>
<body>
<style>
body{background-color: #93B874;}</style>
<centre>
<h1 align="center" style="color:blue;">Course Recommender System </h1>
</centre>
</br></br></br>
<form action="cgi-enabled/index.py" method="GET">
<h3 align="center" style="color:yellow;">Enter Roll:

<input type="text" name="rollno">
<h3 align="center" style="color:yellow;">Enter Pin:
<input type="password" name="passwd" id="passwd">
<label for="password"></label><br><br><br>
<input type ="submit" value="login"></h3>
</form>




</br></br></br></br></br></br></br></br></br></br></br></br></br></br></br></br></br></br></br>

 <div class='grid_20'>
            <hr>
            <p id='web'>National Institute of Technology,Patna-800005,Bihar,INDIA<br/>
             2017 NIT Patna</p>
        </div>
</body>
</html>
```

Fig. 5.1

5.1.2 SERVER SIDE CODE

This code is written in python language which is present in /var/www/html/cgi-enabled/index.py.

```
#!/usr/bin/python

import cgi
import numpy as np
import pandas as pd

print "content-type:text/html"
print""

val=cgi.FieldStorage();

# Getting the roll_no of student
r=val.getvalue('rollno')
p=val.getvalue('passwd')

# Changing roll_no in int data type
roll=int(r)
password=int(p)

df3=pd.read_excel('/home/bhagat/Downloads/user_password.xlsx')
get_pass=df3.set_index('Student_ID').loc[roll].tolist()[0]
#print(get_pass)


# In[91]:

# Extracting the Dataset_new.xlsx file and take it into df data frame
df=pd.read_excel('/home/bhagat/Downloads/Dataset_new.xlsx')
# df.head()

# In[92]:

# Extracting the Elective_DataSet.xlsx file and take it into df2 data frame
df2=pd.read_excel('/home/bhagat/Downloads/Elective_DataSet.xlsx').set_index('Student_ID')
# df2.head()

# In[93]:

# Applying pivot_table() method on Grade coloumn and getting grade_crosstab data frame
grade_crosstab=pd.pivot_table(data=df,values='Grade',index='Course_Code',columns='Student_ID')

# In[94]:

# Selecting grade_crosstab only for roll coloumn
roll_grade = grade_crosstab[roll]

# In[95]:
```

Fig. 5.2

```
# Finding the correlation using corrwith method

similar_to_roll = grade_crosstab.corrwith(roll_grade)

corr_roll = pd.DataFrame(similar_to_roll, columns=['sim_grade'])

# In[96]:

# Filling the nan value with 0.0
corr_roll.fillna(value=0.0)

# In[97]

faculty_crosstab=pd.pivot_table(data=df,values='Faculty_rating',index='Course_Code',columns='Student_ID')

# In[98]

roll_faculty = faculty_crosstab[roll]

# In[99]

similar_to_roll_f = faculty_crosstab.corrwith(roll_faculty)

# In[100]

corr_roll_f = pd.DataFrame(similar_to_roll_f, columns=['sim_faculty'])

# In[101]

corr_roll_f.fillna(value=0.0)

# In[102]

# Adding sim_faculty coloumn into sim_faculty
corr_roll['sim_faculty']=corr_roll_f['sim_faculty']

# In[103]
job_crosstab=pd.pivot_table(data=df,values='Job_oriented',index='Course_Code',columns='Student_ID')

# In[104]
roll_job = job_crosstab[roll]

# In[105]
similar_to_roll_j = job_crosstab.corrwith(roll_job)

# In[106]
corr_roll_j = pd.DataFrame(similar_to_roll_j, columns=['sim_job'])
```

Fig. 5.3

```
# In[107]
corr_roll_j.fillna(value=0.0)

# In[108]
corr_roll['sim_job']=corr_roll_j['sim_job']

# In[109]
scoring_crosstab=pd.pivot_table(data=df,values='Scoring',index='Course_Code',columns='Student_ID')

# In[110]
roll_scoring = scoring_crosstab[roll]

# In[111]
similar_to_roll_s = scoring_crosstab.corrwith(roll_scoring)

# In[112]
corr_roll_s = pd.DataFrame(similar_to_roll_s, columns=['sim_score'])

# In[113]
corr_roll_s.fillna(value=0.0)

# In[114]
corr_roll['sim_score']=corr_roll_s['sim_score']

# In[115]
place_crosstab=pd.pivot_table(data=df,values='Placement_types',index='Course_Code',columns='Student_ID')

# In[116]
roll_place = place_crosstab[roll]

# In[117]
similar_to_roll_p = place_crosstab.corrwith(roll_place)

# In[118]
corr_roll_p = pd.DataFrame(similar_to_roll_p, columns=['sim_place'])

# In[119]
corr_roll_p.fillna(value=0.0)

# In[120]
corr_roll['sim_place']=corr_roll_p['sim_place']

# In[121]
research_crosstab=pd.pivot_table(data=df,values='Research_Purpose',index='Course_Code',columns='Student_ID')

# In[122]
roll_research = research_crosstab[roll]
```

Fig 5.4

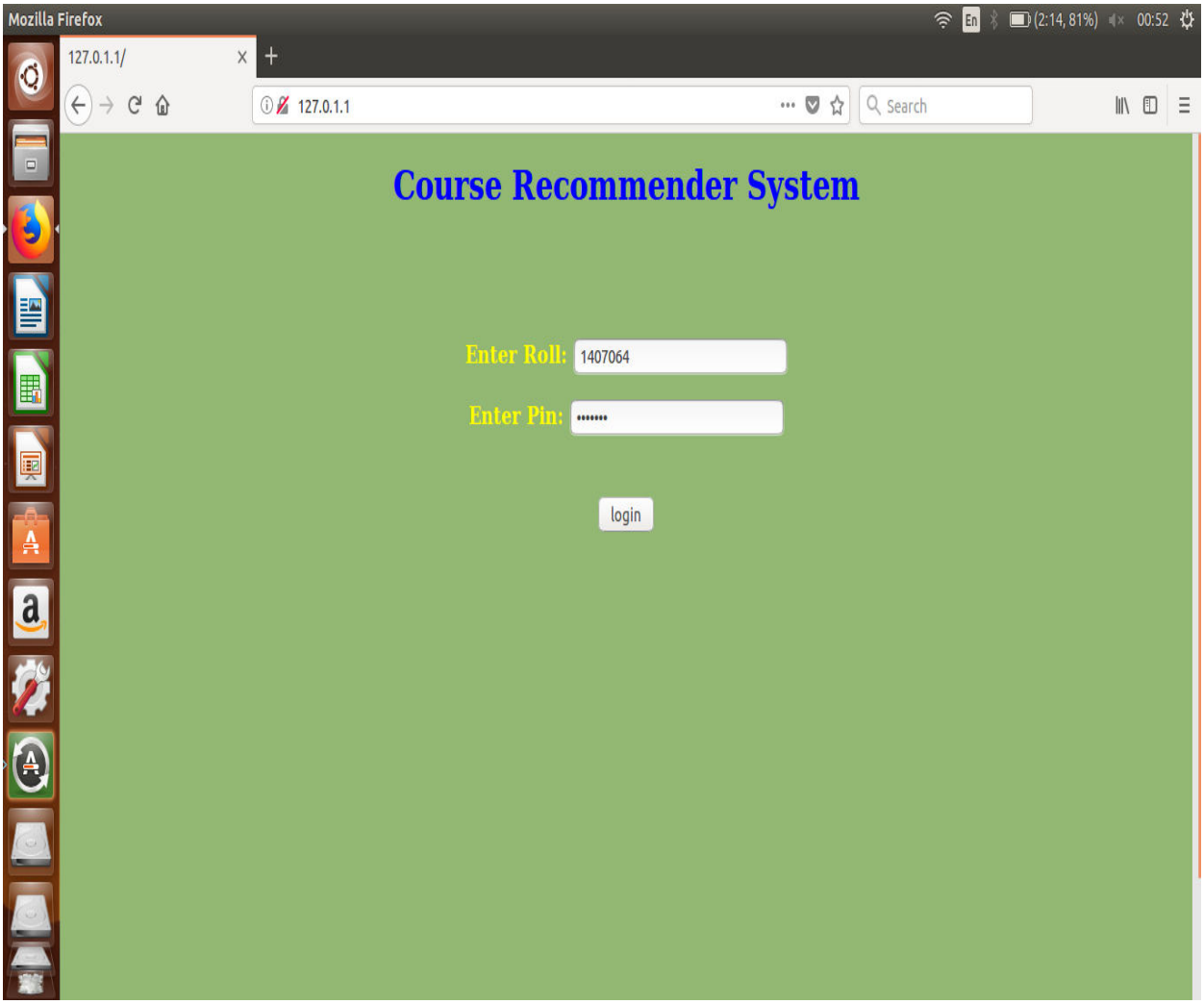
```
# In[123]
similar_to_roll_r = research_crosstab.corrwith(roll_research)

# In[124]
corr_roll_r = pd.DataFrame(similar_to_roll_r, columns=['sim_research'])

# In[125]
corr_roll_r.fillna(value=0.0)

# In[126]
corr_roll['sim_research']=corr_roll_r['sim_research']

# In[127]
corr_roll['pearsonR']=corr_roll.mean(axis=1)

# corr_roll.head()

# In[128]
summary=corr_roll.reset_index()

# In[129]
summary=summary.sort_values('pearsonR',ascending=False)

# In[130]
result=summary[(summary['Student_ID']<1407056) & (summary['Student_ID']!=roll)].head(3)

# In[131]
result_list=result['Student_ID'].tolist()

#print(result_list)

# In[132]
final_A=[]
final_B=[]
for i in range(0,3):
        temp_list = df2.loc[result_list[i]].tolist()
        list_a = [temp_list[0],temp_list[1]]
        list_b = [temp_list[2],temp_list[3]]
        final_A.append(list_a)
        final_B.append(list_b)



# In[133]

print "<html>"
```

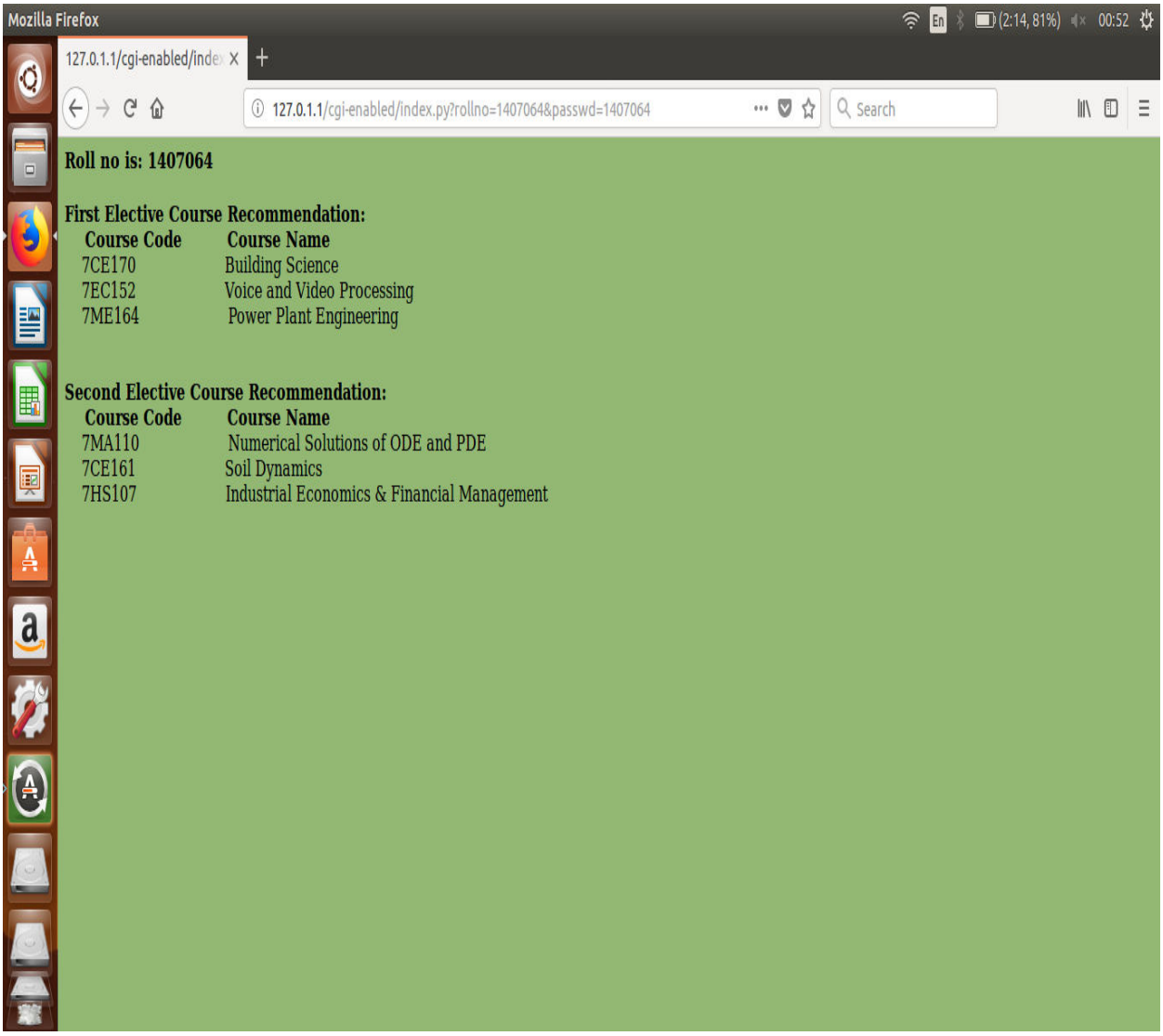
Fig. 5.5

```
print "<head>"

print "<style>"
print "body{ background-color: #93B874; }"
print "</style>"

print "</head>"
print "<body>"

redun = set()
cancel=0

if get_pass!=password:
   cancel=1
   print('Password does not match')
else:
   print "<h4> Roll no is: %s </br></h2>" % (r)

#print "<b>First Elective Course Recommendation:</br></b>"
#print "<b>    Course Code          Course Name </br></b>"

for i in range(0,3):
    if final_A[i][0] not in redun and cancel==0:
        if i==0:
                print "<b>First Elective Course Recommendation:</br></b>"
                print "<b>    Course Code          Course Name </br></b>"
        print('     {}                      {} </br>'.format(final_A[i][0],final_A[i][1]))
        redun.add(final_A[i][0])

redun.clear()

#print "<b></br> </br> Second Elective Course Recommendation: </br></b>"
#print "<b>    Course Code         Course Name </br></b>"
for i in range(0,3):
    if final_B[i][0] not in redun and cancel==0:
        if i==0:
                print "<b></br> </br> Second Elective Course Recommendation: </br></b>"
                print "<b>    Course Code         Course Name </br></b>"
        print('     {}                   {} </br>'.format(final_B[i][0],final_B[i][1]))
        redun.add(final_B[i][0])


redun.clear()

# print "<h2> Roll no is %s </h2>" % (roll)
print "</body>"
print "</html>"
```

Fig. 5.6

5.2 OUTPUTS SCREEEN




Mozilla Firefox
127.0.1.1/cgi-enabled/index
127.0.1.1/cgi-enabled/index.py?rollno=1407064&passwd=1407064
Search
Roll no is: 1407064
First Elective Course Recommendation:
Course Code Course Name
7CE170 Building Science
7EC152 Voice and Video Processing
7ME164 Power Plant Engineering
Second Elective Course Recommendation:
Course Code Course Name
7MA110 Numerical Solutions of ODE and PDE
7CE161 Soil Dynamics
7HS107 Industrial Economics & Financial Management

5.3 RESULTS

The actual elective course code and predicted elective course code are shown in following table –

Student_ID	Course_A (Predicted)	Course_B (Predicted)	Course_A (Actual)	Course_B (Actual)
1407056	7HS109 7MA107	7EC148 7CE161 7EE163	7HS109	7EE163
1407059	7CE170 7EC152	7MA110 7HS110	7CE170	7HS110
1407061	7CE170 7HS109	7MA110 7EC163	7CE170	7MA110
1407063	7HS109 7EC152	7CE161 7ME167 7HS110	7EC152	7HS107
1407064	7CE170 7EC152 7ME164	7MA110 7CE161 7HS107	7ME164	7CE161
1407065	7CE170 7MA107 7AR253	7EC148 7EE163 7HS107	7AR253	7EC148
1407067	7ME164 7MA107	7HS107 7EE163 7CE161	7MA107	7CE161
1407071	7CE170 7AR253	7HS110 7CE161 7MA110	7CE170	7HS110
1407074	7MA107 7EE618 7HS109	7EE163 7HS107 7MA107	7EE618	7MA110
1407075	7CE170 7AR253 7MA107	7MA110 7HS107 7EE163	7CE170	7HS107
1407076	7ME164 7AR253	7HS110 7HS107	7HS109	7MA110
1407077	7CE170 7EC152	7HS107 7CE161 7HS110	7EC152	7HS110
1407081	7MA107 7CE170	7HS107 7MA110	7MA107	7HS107
1407083	7EC152 7ME164	7CE161 7HS110 7HS107	7ME164	7CE161
1407084	7ME164 7HS109 7CE170	7HS110 7EC148 7MA110	7CE170	7EC148

Table 5.1

CHAPTER 6-CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

In this paper, we have used user-based collaborative algorithms to recommend elective courses for new students based on their previous data set. . We can see that all the initial requirements of the project are achieved. The web application is very friendly; any student can easily check their recommended courses using it. Starting from coming up with project idea, understanding the requirements and choosing the best technologies for the implementation, all this give us very good experience and exposure in development of a full stack web application. Overall, our results on real-life experimentation were very encouraging. We believe, collaborative filtering approach provide more or less exact matching of courses students will take. The course recommender system will be helpful for students to establish College Student Relationship Management strategy as well.

6.2 FUTURE ENHANCEMENT

For future work, research can done in developing integration strategies for approaches that can accurately predict student performance in courses and approaches that help to predict grades in particular course or subjects. These approaches can be used to provide valuable advice to students during career guidance advice and courses selection process.

CHAPTER 7-REFERENCES

[1] Linden, G., Smith, B., York, J.: Amazon.com Recommendations: Item-to-Item Collaborative Filtering. In: IEEE Internet Computing, Volume 7, Issue 1, pp.76-80

[2] Breese, J., Heckerman, D. Kadie, C: Empirical Analysis of predictive Algorithms for collaborative filtering

[3] Ekdahl M, Lindstrom S, Svensson C: A student course recommender. Master of Science Programme Thesis, Lulea University of Technology

[4] Sandvig J, Burke R : AACORN: A CBR Recommender for Academic Advising. Technical Report TR05- 015, Depaul University, Chicago, USA

[5] Farzan R, Brusilovsky P: Social Navigation support in a Course Recommendation System.

[6] Bendakir N and Aimeur E: Using Association Rules for Course Recommendation

[7] ShivamBansal ,Understanding Basics of Recommender Systems with case study, Analytics Vidhya.

[8] Shuvayan Das, Beginners Guide to learn about collaborative filtering based Recommender System, Analytics Vidhya

[9] Jose Portilla: Python for Data Science and Machine Learning Boot camp, Department of Mechanical Engineering, Santa Clara University

[10]Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An Algorithm Framework for Performing Collaborative Filtering. In: Proceedings of SIGIR, ACM, pp. 77--87,(1999)

[11] http://students.cse.tamu.edu/meghz17/Final_Report.pdf

[12]https://www.uwb.edu/mscsse/curriculum/thesis-project/capstone-projects/papali-susan

[13] Deshpande, M., Karypis, G.: Item-based Top-N Recommendation Algorithms. ACM Transactions on Information Systems 22(1): 35. Vol. 22, No. 1, pp. 142--177 (2004)

[14] https://www.learnpython.org

[15] https://www.tutorialspoint.com/python/python_cgi_programming.htm