STUDENT'S PERFORMANCE MONITORING SYSTEM

A Project Report

Submitted by:

ADNAN ASHRAF (18205135050)

FARIA FARHEEN (18205135086)

SYED DAWAR (18205135094)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER ENGINEERING

at

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

SSM COLLEGE OF ENGINEERING

PARIHASPORA PATTAN, BARAMULLA, KASHMIR

AFFILIATED TO

THE UNIVERSITY OF KASHMIR, HAZRATBAL SRINAGAR

2022

DECLARATION

We hereby declare that the project entitled “Student’s Performance Monitoring System” submitted for the B.E. (CSE) degree is our original work and the project has not formed the basis for the award of any degree, diploma, fellowship or any other similar titles.

Date: **Signature of the students:**

CERTIFICATE

This is to certify that the project entitled

STUDENT'S PERFORMANCE MONITORING SYSTEM

Submitted by:

ADNAN ASHRAF **Enroll: 18205135050**

FARIA FARHEEN **Enroll: 18205135086**

SYED DAWAR **Enroll: 18205135094**

is the Bonafide work carried out by them under supervision of **Mrs. YASMEEN VIQAR**, and is approved for the partial fulfillment of the requirement for the award of the degree of Bachelor of engineering **(Computer Engineering)** of SSM College of Engineering and Technology, Kashmir affiliated to University of Kashmir, Srinagar during the academic year 2022.

This Project Report has not been earlier submitted to any other institute or university for the award of any Degree.

Mrs. YASMEEN VIQAR **Mrs. YASMEEN VIQAR**

Project Guide **Head of Department**

Department of CSE **Department of CSE**

PRINCIPAL

ACKNOWLEGEMENT

It is our proud privilege and duty to acknowledge the kind help and guidance received from several people in preparation for this project. It would not have been possible to prepare this project in this form without their valuable help, cooperation and guidance.

First and foremost, we would like to extend our deepest gratitude to **Mrs. YASMEEN VIQAR**, head of the department CSE and internal guide, for her constant support and encouragement in the preparation of the project and for making data and other facilities available which were required for successful preparation of this project. We extend our heartfelt gratitude to her for providing us with the right guidance and advice at crucial junctures and for showing us the right course of action. We are also grateful to her for her immense support and supervision during the course of the completion of the project.

We are also thankful to the **PRINCIPAL**, of SSM College of Engineering, for expressing his confidence in us by letting us work on a project of this magnitude and providing his support, help, and encouragement in implementing this project.

Finally, we would like to thank our family and friends for their encouragement and support during the course of our work.

ADNAN ASHRAF

FARIA FARHEEN

SYED DAWAR

ABSTRACT

Monitoring is defined as a supervising activity in progress to ensure that observing entities are on track and schedule, to meet the objectives and performance targets. In many institutional organizations, especially colleges, monitoring the performance of students is a crucial factor in determining the growth and progress of a student as an individual and further enhancing the overall result of the institution. There are some guidelines that every institution follows to analyze the performance of every individual in a class and then categorize them based on their learning capabilities. Mostly this course of action follows a manual procedure wherein learning facilitators first collect the academic records of students manually which tends to be inconvenient, then they categorize the students based on the collected data. This approach proves to be time consuming, requires conscious exertion of power and is prone to human errors. The most effective way of resolving this concern is by providing an IT- based solution which will automate the process of performance monitoring. For serving this purpose we have undertaken the project "Student's Performance Monitoring System" in response to the need for an effective and efficient system for tracking and monitoring student's academic performance. This system promises to provide the institutional facilitators techniques to further undertake proper measures for improving the capabilities of the students. It will allow faculty to electronically collect and manage academic records of students and ease out their categorizations in two sets viz slow learners and fast learners. This system aims at providing an efficient solution which not only works on the academic records of the students but also encompasses their behavior and attentiveness in the class. This system assists the faculty to closely monitor and analyze the students and provide subject wise assessment which furnishes a detailed report of the student. The outcome of this project will be a methodological segregation of students and systematic, logical and well-planned record monitoring of students. The technologies used are java enterprise edition for the backend development, bootstrap for frontend development and MySQL for the database.

LIST OF FIGURES

LIST OF TABLES

GANTT CHART/ TIMELINE CHART

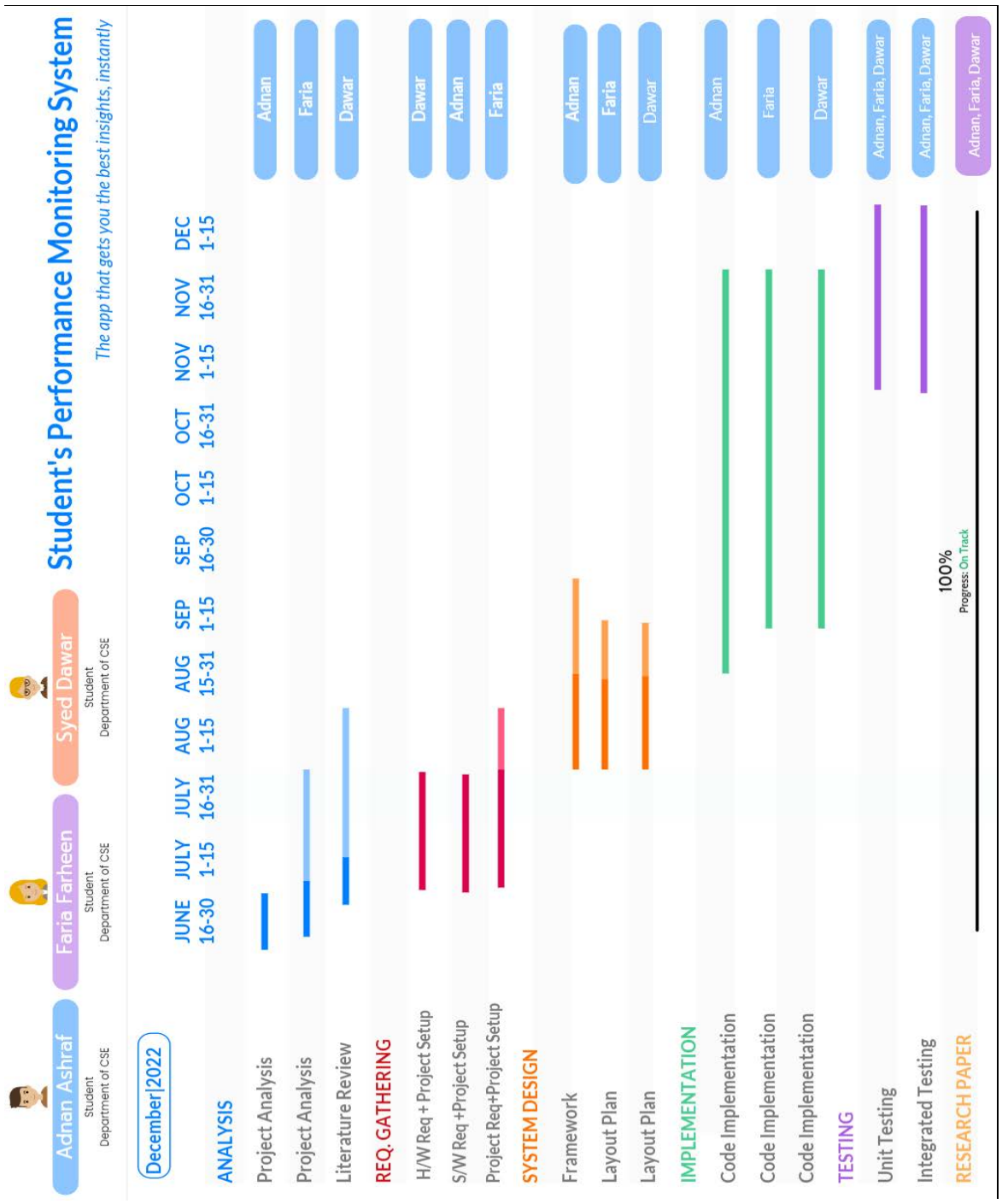


TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The monitoring of performance is an essential factor in promoting the growth of an individual as well as the overall enhancement of the performance of the institution. This monitoring may be done manually or automatically based on the user's choice. It is a widely accepted fact that student evaluation of the institution provides an insight into the effectiveness of teaching in the higher education system. According to the paper published in 2012 by Hsiu-Ping, Tzy- Ling Chen, many universities still believe that student evaluation helps an institute to perform better and improve loopholes [1]. Similarly, evaluation of students is also a major factor that contributes to the overall academic performance of the university or college. Performance can be determined based on various factors like participation of an individual in activities, academic results, behavior in class, etc. The evaluation of a student's performance should be an inclusive part of the institution. In today's world the competition in acquiring a good opportunity for job is a fierce one, thus, preparing the graduating students to face the real challenges becomes a necessary task to make them aware of their capabilities and areas of improvement [2].

For this reason, performance evaluation awakens a sense of accountability in the students. The manual process of classification of students based on their performance takes a lot of time in collecting the required information which serves as the basis for the classification and is prone to human errors. The details are sometimes not available on time which leads to the delay in the results generated by the faculty. Also, the manual process of classification takes a lot of effort, first in collecting the details and then evaluating those details to yield accurate results. The classification of students based on their performance as evaluated is also prone to human errors and is time-consuming. Therefore, we present an automated system to collect the details, assessment of the students, and then the classification of the students based on the evaluation results. This system is automatic in the sense that there is only a need to set the criteria based on

which the segregation will take place and the students must upload the required details with proof that the details are correct to their knowledge. To effectively measure the outcomes of a student an area of assessment must be selected. The area of evaluation must provide a link to the student attributes such as skills and abilities and the result should define these abilities in terms of measurable performance indicators [3]. It can be one or more subjects where the faculty can evaluate a student based on certain parameters set by the university. In our paper we adhere to the evaluation layout presented by the college.

A student's performance monitoring system is an automated system implemented in java using the spring framework to classify slow and fast learners in a class and provide them with opportunities to enhance their capabilities. The faculty evaluates the student based on his/her performance in the class and the system generates an overall result of the performance. This allows for an efficient collection of the details and an easy classification of the students into fast and slow learners. Spring MVC I.e., the Model View Controller of the spring framework has been used to efficiently develop the system.

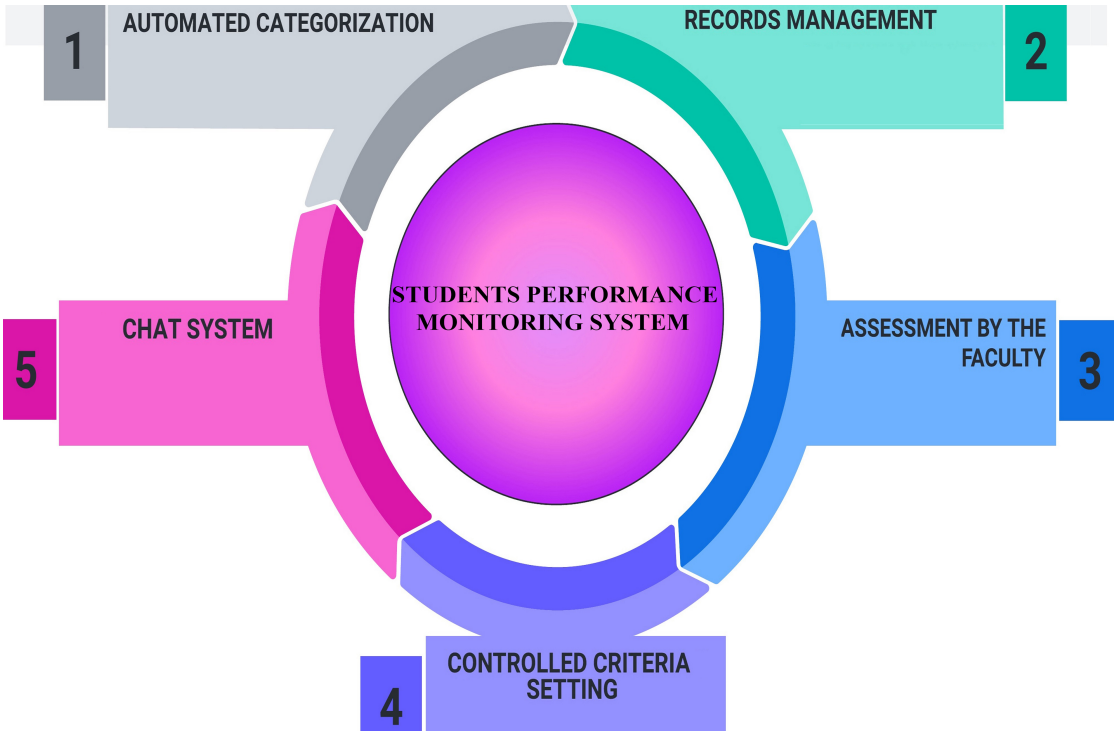


Figure 1.1 Features of Student's Performance Monitoring System

The factors that we choose for the classification of the students are the previous year's results, internal results, and the subject-wise assessment by the faculty. These factors make up a certain portion of a percentage that adds up to make an overall aggregate based on which the students can be classified into two groups. The groups are captioned as the group of slow learners whose obtained aggregate percentage will be less than the required percentage and these students will further get notified by the faculty for counseling sessions, extra remedial classes, or parent-teacher meet. The next group will be that of the fast learners whose obtained percentage will satisfy the required percentage and these students will be provided with opportunities like participation in seminars, career counseling sessions, workshops, etc.

The students' performance system has essential features that will help the faculty to better analyze a student and take proper and necessary actions to enhance their capabilities further. As shown in fig 1.1 the main area of focus of this system will be automatic categorization, records management, a chat system that will enable a faculty member to get connected to the student efficiently, assessment by the faculty to provide better insights of the productivity of the students and a controlled setting of criteria which can be altered by the head of the department at any given point, even after the results are generated by the system.

This individual attention will facilitate the growth of the learning capabilities of both the groups and an individual will also be aware of his performance and try to improve it to come under the category of fast learners. In this way, the overall performance of an institute will be improved. Also, the feature of allotting mentors to the slow learner is of great advantage as mentoring helps to build a positive understanding uniting a counsellor and an individual who is mentored. The mentor provides guidance and transfers his/her knowledge to the student which helps the student to make right decisions and take full benefit of the opportunities coming their way.[4]

Constant monitoring builds quality of wakefulness in a student and he/she strives to be responsive to the events that take place in the future. The subject-wise assessment

helps the educators to provide a detailed review of a student's performance in the class, especially in a particular subject and this assists the educators to have precise information about a student's interest in a particular subject and if the student is weak the educator can take necessary steps to tackle the ineptness of the student.

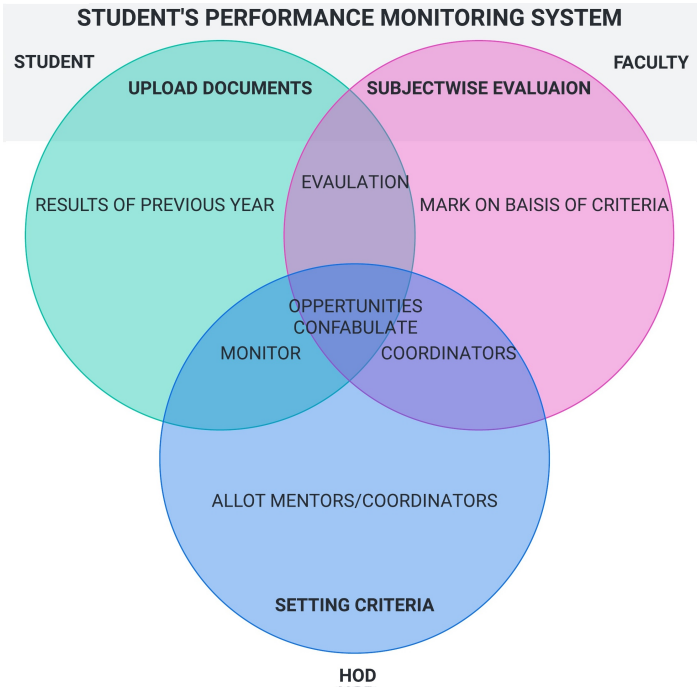


Figure 1.2 Area of use cases

As depicted in fig 1.2, the automated system can be used by the faculty of the institution having any departmental background, the faculty can search the students based on their emails, department, semester and batch. The searched students can be of the same department as the faculty or can be of a different department. After that the faculty can make an evaluation of the students based on their subjects of interest, for example a teacher teaches mathematics to semester 1^{st}, then he/she can make an evaluation of that subject only. Also, a faculty can take the roles of mentor or coordinator and can perform their functions which are already discussed.

Another entity that can make use of this system is the head of the department who can

set criteria of all the three contributors and even set criteria of the overall percentage which will decide the grouping of students. Students are another set of entity that can make an efficient use of this proposed system by uploading their previous results, internal results and can also view their progress as fast/slow learners. Students can also chat with their mentors and grab opportunities through notifications provided by the coordinators.

CHAPTER 2

LITERATURE REVIEW

Several research papers and studies have been conducted to find out students' academic performance monitoring. 29 research papers are taken under consideration for review.

Effective mentorship is necessitated to help students with shortcomings and a lack of a connection between talent and productivity, poor study habits, and ineffective teaching strategies. An inquiry has revealed that several variables, including student learning, age, and gender bias, altered students' performance. Kochhar 2000 [5].

This research examines the effects of distinctive social circle traits on academic attainment using a novel aligned panel data set on students and schools. It tackles the major formulation hurdles that impede our ability to quantify the achievement impacts of social circle makeup directly. Covariates include unexamined or inaccurately quantified student, family, and school characteristics, as well as the bilateral nature of social circle connections. The core technique entails gradually removing the portions of individual student achievement improvement that are most prone to confound family and school pressures with social circle influences. The fixed effects paradigm does not address the ability to collaborate in socialization, which likely causes multivariate regression bias when temporal peer behavior is proxied by current average accomplishment. Extensive research on the premise that social comparison has more sway than family. The pupils' average grade points and peer assistance were substantially correlated. Hanushek et al. 2002[6]

To forecast the students' progress, a matrix factorization and multi-regression approach-based monitor was proposed. It was initially intended to monitor e-commerce apps. However, it may be utilized to track pupils' progress. It employs a degree planner, which forecasts students who will do poorly and may fail the course. It also predicts future paths based on prior success. The sampling strategy to help young with an advising system was developed using Data Mining, Collaborative Filtering. The research made use of ASP.NET and MS SQL database server architecture. In this investigation, the performance standards were ethnicity, cluster number, student division, age, and CGPA. Ganeshan and colleagues (2015) [7].

This paper delves into the consistent protocol called for providing a designed questionnaire to 50 state education department affiliation directors. The views of state school board directors on whether vocational training had a good consequence on student success were investigated abductive. Their remarks were then juxtaposed with the 2009 ranking of state curricula by "Ed Week." The survey discovered that most states do not necessitate skill learning for school board members based on responses from the 26 replying state directors. State board directors acknowledged that school board professional development boosted student success. According to "Education Week" 2009, the states that did mandate school board professional development had an overall rating of B, or C. States that did not necessitate professional development earned a C or D rating. This study explores the attitudes of state directors on professional development for school board members in US public school rhetoric and fills key research gaps. As Per the findings, pupils who actively take part in the learning activity exhibit better CGPAs. Robert and Sampson 2011 [8].

Developed a solution for performance monitoring based on data mining. It keeps track of the student's progress and generates semantic rules that may be used to further evaluate how well the students are doing in the course as a whole. It produces semantic rules that use a decision tree paradigm. To improve the caliber of study material, this system employs semantic web and ontological methodologies. A technique for

measuring the variance correlation matrix of the extrapolated precipitation field is also presented. The proposed approach is then confirmed using an observed daily precipitation dataset from New Zealand. The findings illustrate that the proposed interpolation method may yield precipitation surfaces with high spatial and temporal resolution and lower interpolation errors in both data-sparse and data rich zones. Huang et al. in the year 2014 [9].

Edin Osmanbegovi and Mirza Suljic collected data from first-year student surveys and other information gathered during enrollment at the University of Tuzla in 2012. To make their estimates, they also considered gender, family, distance, high school, GPA, admission test, scholarships, time, supplies, the internet, grade significance, and wages. In this study, three autonomous data mining algorithms were used to pre-frontal brain data to predict course results (passed or failed), and the learning approaches' performance was assessed based on prediction accuracy, ease of learning, and user pleasant aspects. Following the results, the Nave Bayes classifier surpasses prediction decision tree and neural network approaches. It has also been stated that a successful classifier model must be both exact and intuitive to professors. Because the data mining techniques were deployed after the data was captured, this study was based on regular classroom scenarios. [10].

Outlined their approach to tracking the movement of the student's records. The student records of the University Malaysia Sarawak (UNIMAS) are now administered using a student management system, although academics do not have access to the system. The privacy settings, which only allow access to top management, including deans and vice deans of undergrad and student development, are to blame for this. To track students' performance at the Faculty of Computer Science and Information Technology, this project suggests a system called Student Performance Analysis System (SPAS) (FCSIT). The suggested system provides a predictive system that can forecast a student's performance in the course TMC1013 System Analysis and Design. This system helps the lecturers from the information system department identify students by

helping them anticipate their performance [11].

Table 2.1-Recent Work Done in Student's Academic Evaluation

Reference Number	Technique	Focused area	Application	Setup/ Dataset	Results/ Evaluation parameters
[5]	Effective mentorship	To aid students with shortcomings and a connection connecting ability and productivity, poor study habits, and ineffective teaching	To Improve Students' Performance	Role Modeling	learning, age, and gender bias, competence of students in English.
[6]	Mathematical time variant and invariant equations along with idiosyncratic random error	Teacher quality with succeeding grades and measurement of variation in school quality resources.	Improve the quality of teaching and provide better options for parents to select from a good/bad school.	Data derived from data development activity of Harvard and PEIMS	Contribution of teacher quality, variation in grades, correlation in School average math's and reading test scores.
[7]	Data Mining, Collaborative	K-means	Advising system which	ASP.NET and MS SQL	Ethnicity, Cluster number,

	Filtering	algorithm.	provides advice to a group of students.	database server.	Student division, age, CGPA
[8]	Inductive Analyses	Issue of professional evolution of education for institution's members.	Institution's board members must have required skill development to take quality decisions for children.	Data set was collected by questioning 50 directors of state school board associations.	The professional attitude development for an institution's members in US public schools is carried out.
[9]	Kinect sensors	Bayesian minimum-error method, automatic segmentation method.	Automatic evaluation of skills in nurses for the task of transferring patients from bed to wheelchair.	Data set via the colors of the attached markers and calculating their 3D positions by combining color and depth data from two sensors.	The results achieved accuracy exceeding 80%.
[10]	Data Mining	Comparisons with Bayesian classifier, neural networks and decision trees.	Solving problems of prediction, approximation, function, classification, accuracy and pattern	Java, WEKA software package	Chi-Squared, One R, Info Gain, Gain Ratio, AVG Rang

			recognition.		
[11]	Linear Regression Analysis	Fuzzy sets of low, medium, and high grades to analyze significant correlations.	Relationships are evaluated between grades in math and physics courses.	Data was collected by analyzing 53 students who attended the college in semesters of 2008 and 2009.	Prerequisite Course, Subsequent Course, Prereq., Group Values, and mean
[12]	Quantile regression techniques	Estimate educational peer effects	Augment average reading skills in the student population	First wave of the OECD PISA sample with register data	Correlation between peer mean and peer variance
[13]	Scaffolded concept mapping strategy	The whole class should progress in synchronized fashion through this strategy	Meaning Full Learning	54 undergraduates in a database management system course	The students who used the strategy had better learning achievements than those who only experienced traditional lectures

[14]	Data Mining, Decision tree approach	To improve the caliber of education, there is a need to be able to forecast scholastic performance of the students	For predicting academic performance	The IBM Statistical Package for Social Studies (SPSS) is used to apply the Chi- Square Automatic Interaction Detection (CHAID)	Financial status of the students, motivation to learn, gender
[15]	Rubrics and oral feedback	To help students improve performance and meet learning outcomes	To grade students' work in a more consistent, reliable and unbiased manner	Rubrics based assessment	The rubrics assessment led to a significant decrease in the number of complaints and questions regarding grades
[16]	Data Mining Techniques	Predict Students' Future Learning Outcomes	To Analyze Student Performance	350 B.E Students With 24 Criteria's	CGPA, Attendance, SC Marks, The Engineering Cut-Off, The Pedagogical Board, The King Board

[17]	A Web Based Monitoring System	Student Performance Evaluation and Mentoring	Predict Courses Based on Common Elements in the Group of Students	Recommendat io n Systems, Clustering Algorithm K-Means	Elements In Common
[18]	Machine Learning Techniques	Modeling Of Students	Improving Prediction of Students Academic Performance	Records Of 11,027 Students of Babcock University Nigeria	Gender, age, parent's marital status, spouse's education, parent's profession, SSC score, HSC score, as well as the first CGPA
[19]	Data Mining and Use of Different Classifiers	Predicting The Academic Performanc e of The Students, Particularly In Enginering Discipline	Development Of Discission Support System	Complete (1000), Complete (525) and Outliers (960) instances, respectively.	Consistent Behavior and Performance Accuracy
[20]	Data Mining Techniques	Academic Absenteeism (loss of academic	To Monitor Both Academic and Non-Academic Data	Enrollment And Prior Academic	Predicted Attrition in a Student's First Four

		standing)		Records	Enrollments
[21]	Data Mining	The Relevance of Academic Analytics for Educational Institutions and How They Interact to Promote Education	An Intelligent Recommendation Intervention	Artificial Neural Network and Decision Tree	Final grade, Attendance, prerequisite Subject, English and Mathematics Marks
[22]	Data Mining	Student Performance Analysis System (SPAS)	Student Performance Prediction	TMC1013 System Analysis and Design	Outcomes are Student Performance Prediction via Data Mining Techniques like Classification.
[23]	Data Mining Techniques	Improving The Performance of Students	Predict the final grade of students	Decision Tree (ID3)	Student Database
					Gender, Time, Punishment, family, parent live, Parent education,

[24]	Data Mining	To Construct a Classification Model to Predict the Performance of Students	Instance, prediction, classification, association rule mining, and clustering	150 Students taken for the survey based on a questionnaire	Parent financial, environment, Encouragement, Absent, help, Father, mother, Love, accommodation, Work, study, Sleeping, pass
[25]	Educational Data Mining (EDM), Classification, Decision Tree	Traditional Classroom Educational System	Predicting Students Academic Performance	ID3 Decision tree algorithm	Fathers' education, Fathers occupation Mother's Education, Mother's Occupation Category, SSC Board, Admission Type, SSC Medium, SSC Class, First- Sixth Semester Results.

[26]	Un-Supervised learning, Reinforcement learning and Adversarial learning.	FeedBot and LitBots using Social Bot Framework.	The adoption of chatbots in mentoring of students.	Over 700 students used the system and provided feedback. T-REST MITOCAR's API is used in setup.	Students yearned for a real human connection on the other end of the bot and bot's reaction lacked uniqueness.
[27]	Cognitive and Meta-Cognitive and Predictive methods.	Fermat, Math Spring, Auto Tutor, Meta Tutor, SRL processes, PAL3 and MARi.	Importance of incorporating technology in the educational field.	Dataset is captured through web camera for metacognitive and user's data is used in other technologies.	The demands for AI-based knowledge services to support mentoring.

[28]	Fuzzy Systems, Group Performance Model(GPM).	Extracting linguistically interpretable scaled fuzzy weighted rules from student data	Effective Teaching and Learning	Fuzzy LS, GPM,ALL	Fuzzy weighted rules identify useful relationships between student engagement and performance for transparent evaluation.
[29]	Learning Analytics, Computational Ontologies	Relationship between Learning Analytics and Ontologies, and how they have been applied in a coordinated way	Academic performance evaluation of students	1230 Studies	Use of a more adaptive and personalized learning environment, and the better use of pedagogies to enhance teaching/learning.

Probit regressions are employed in this dissertation to describe influences at multiple spots along the probabilistic test score distribution. Weibull regressions offer "snapshots" at different spots of a regression line and are thus a succinct means of explaining the entire distribution. The estimation of training production functions at multiple spots throughout the test score distribution disclosed some noteworthy aspects that would not be evident if only one regression equation, such as the mean, was examined. In accordance with the univariate regression results, there may be diverse peer group impacts at different places in the probabilistic test score distribution. In general, peer group effects are larger in the bottom end of the contingent test score distribution, reinforcing the peer literature's notion that low achievers are dependent

learners. Such disparities in peer influence are most likely attributable to pupils from educationally pampered homes functioning well in most school contexts. In contrast, the performance of pupils from less privileged families may be more complex and influenced by class sizes such as teaching effectiveness, school funds, and their contemporaries. The findings for bright students were found to be adverse, as per Rangvid, B. S. (2003), who demonstrated that mixing skills had an efficacious effect on weak pupils [12].

This study shows a scaffolded concept mapping methodology that is tailored to a student's past knowledge (high or low) and provides dynamic learning aids (scaffolding and fading) for having read and sketched a concept map. The goal of this approach is for the entire class to grow at a timed pace, with each student earning significant learning. The scaffolding is tailored to the student's needs, with varying levels of aid. Considering the escalating need for scaffolding and its perpetual growth, software aid is necessitated. The authors used a rigorous instructional design, together with supporting software, to create an educational atmosphere that encourages strategy execution. The article also examines if this approach boosted student achievement in a database management system curriculum for 54 students. The findings show that students who adopted the method outperformed those who just received prior lectures regarding their learning outcomes. Likewise, the strategy's rollout got a good student response. To evaluate the student's performance in virtual laboratory ideas, a Bayesian network tool is used. In 2013, Chen-Hsuan and colleagues [13].

The goal of this study is to predict students' academic achievement using a decision tree approach. Education serves as the foundation for a society to raise the effectiveness of its inhabitants. To boost quality education, it is essential to be able to foresee pupils' educational success. The IBM SPSS Version for Social Studies is utilized to make the decision tree structure using the Chi-Square Automatic Interaction Detection (CHAID). Students' socioeconomic situation, inclination to learn, and gender were discovered to influence overall performance. It was anticipated that 66.8% of the pupils would pass, while 33.2% would fail. It was discovered that a far higher percentage of students

seemed capable of passing and that males were more likely to pass than female students. From Nigerian colleges of education, K.D. Kolo and J.K. Alhassan 2015 compiled data on computer science students. They researched information on this topic since they thought the Data Structure course in computer science was one of the most crucial topics to cover in their research. They believed that major determinants of SAP were student demographics including grade, status, gender, financial stability, and attitude toward learning [14].

Rubrics and oral criticism are key techniques for aiding students in improving their ability and fulfilling learning objectives. Their influence on the real benefit attained, however, is dubious. The point of this study is to measure the influence of rubrics and oral criticism on student learning outcomes. An exercise on requirements engineering was executed in a software engineering program, with the two procedures used in course deliverables. These treatments led to statistically significant gains, but no substantive enhancement (a reduction of further than one notch) was attained and the adoption of these interventions led to an increase in student knowledge of teachers' aspirations. There were vastly fewer complaints regarding grades this time than in prior years. The implementation of rubrics led to a significant reduction in the number of tier complaints and queries also given that the gain in educational objectives was less than envisaged, it is dubious if the price ratio of these procedures is significant enough to warrant their usage. Barney, Sebastian and Khurum [15].

The purpose of this study is to convey a comprehensive review of the many data mining tools that have been deployed to forecast student development and performance, as well as how these prediction techniques to aid in the designation of the most significant student characteristic for prognosis. Roughly 350 BE (CSE) students from the KLN College of Information and technology provided the data for R. Sumitha and E.S. VinothKumar's 2016 research. They specifically chose 24 attributes for research, but in the conclusion, they only regarded the criteria with the highest rankings for categorization. The parameters that were identified include the CGPA, the arrears, the attendance, the SSC marks, the engineering cut-off, the pedagogical medium, and the

kind of board [16].

The conclusions of employing an educational data mining method to calculate intellectual absenteeism (loss of academic status) at the Universidad Nacional de Colombia are discussed in this study. Two data mining scenarios were established to scrutinize academic and nonacademic data; the models utilize two classification techniques, nave Bayes and a decision tree classifier, to improve our knowledge of attrition during first enrollments and to assess the quality of the information for the classification task, which can be grasped as the prediction of academic status loss due to poor academic performance. The models are designed to assess disengagement in the pupil's first four enrollments. Examining any of these eras first, and then at a specific enrolment. The models were constructed using historical academic data and records from the enrollment, and they were appraised using tie and previously unseen statistics from a comprehensive academic term. The studies show that adding academic data improves the prediction of academic status erosion. Lopez et al. in the year 2015 [17].

In this study the survey method used was multilayered. To pinpoint relevant students' situational variables, in-depth conversation was used. Parental (educational qualification, occupation) support, household volume, parental marital status, and average household income are the identified background covariates. Following that, 7,500 pupil enrollment records encompassing 68% of Babcock University admissions betwixt 2001 and 2010 were arbitrarily chosen. The study institution was chosen on intention. Based on the student's enrollment records, ten classifiers (Random Forest, random tree, J48, Decision stump, REPTree, JRip, OneR, Xero, PART, and Decision tree) and a multilayer autoencoder (convolutional neural network) learning algorithms were used to garner models from the Waikato Environment for Knowledge Analysis (WEKA). The created models were compared utilizing the following metrics: precision level, perplexity matrices, and model calibration speed. Depending on the outcomes, a framework for an Intelligent Recommender System (IRS) was devised, and the IRS was

implemented utilizing Netbeans IDE and Java programming language, with MySQL as the database server. The analysis revealed that Random Forest, Reptree, J48, JRip, PART, Decision Table, and Multilayer Perceptron fared well, with the minimum accuracy being 96.78%, while Decision stump, OneR, and ZeroR fared marginally worse. Overall, the random tree surpassed other classifiers with an accuracy of 99.908%. In addition, numerous measurements revealed that random trees surpassed other methods. As a result, in the context of this study, a random tree is chosen as the best algorithm. The rules generated by the maximum algorithm serve as the IRS's foundation and help predict students' academic achievement. Maria Goga, Shade Kuyoro, and Nicolae Goga in 2015. [18].

Precise projections of pupils' educational outcomes in the initial phases of the study program aid in the spotting of poor students and permit management to take restorative measures to avoid their flunk. Conventional single-classifier-based forecasting is not highly adaptable from one scenario to the next. Additionally, a testable prediction established for a certain course at a particular college may not be viable for a new trajectory at the same or any other organization. With this in light, this study offers the idea of unified various classifiers for estimating students' intellectual development. The integrated classifier is made up of three supporting algorithms: the K-Nearest Neighbor, the decision tree, and Aggregating One-Dependence Estimators (AODE). To integrate various classifiers for the prediction of the educational performance of engineering students, a commodity of probabilistic merging rule is used. This technique allows a coherent approach to estimate student performance. Using the t-test, the hypothesized technique was implemented and compared to three student performance records. The suggested technique is also compared against the different classifiers as well as the KSTAR, OneR, ZeroR, Naive Bayes, and NB tree classifiers. Mrinal Pandey and S. Taruna 2016. [19].

Outlined a model for academic absenteeism (loss of academic standing) at the

University of Colombia based on an approach to data mining. To monitor both academic and non-academic data, two data mining models were developed. The models use different classifiers naive Bayes and decision tree classifiers to better understand attrition during preliminary enrollments and to assess the quality of the data for the classification task, which can be understood as the forecasting of the academic standing is lost due to mediocre academic performance. The models are developed to predict attrition in a student's first four enrollments. Before examining a specific enrollment, think about any of these times. The utilized models were scrutinized throughout an academic cycle leveraging cross-validation and data that was previously undiscovered, as well as data from the enrollment and prior academic records and records. Perikos and associates 2014 [20].

A framework for defining how to anticipate students' formative assessment should be published in the literature. Different learner variables and traits are used by various writers to evaluate student achievement. The plurality of writers utilized the student's recreational activities, co-circular activities, CGPA, internal and external evaluations, and final exam results as prediction factors. The majority of colleges and universities in India utilize the student's final test grade as their norm for gauging academic success. Any student's final scores are based on a variety of factors, including laboratory file work, viva-voce, external evaluations, and internal and external evaluations. How well a student does is based on how many grades they receive on their final test. The relevance of academic analytics for educational institutions and how they interact to promote education was discussed by Norlida Buniyamin, and PauziahMohd Arsad, et al. in their 2013 study. To raise students' academic performance and success, they also suggested an intelligent enjoinder intercession system. The student grade and student information are the two different student aspects used by this method to assess achievement [21].

In this study, a system is devised to assess the success of students in the FCSIT course

"TMC1013 System Analysis and Design" by monitoring student achievement via data mining classification strategies. Additionally, the Student Performance Analysis System (SPAS) was created to aid instructors to discuss with students by permitting lecturers to view the students' long-term trends within a specific curriculum and semester. Certain goals have been recognized throughout the construction of this system: To cultivate a mechanism for analyzing student outcomes, To facilitate IS lecturers in predicting academic achievement in the course "TMC1013 System Analysis and Design" by using data mining techniques in the current proposal, To examine the factors that influence the pupils' performance in the course "TMC1013 System Analysis and Design," and To assist lecturers in taking note of the students' advancement all through the semester. 2014 saw Chew Li and colleagues [22].

A massive quantity of data is recorded in instructional archives, and these datasets include crucial tools to determine student success. Stratification is one of the most valuable data mining approaches in educational databases. In this research, the decision tree strategy is applied to a pupil's record to forecast the achievement. Here several variables acquired from the pupil's database to forecast the pupil's final grade. This research will assist learners in enhancing their efficiency, identifying students who require additional focus to diminish failing rates, and initiating immediate measures at the opportune moment. The course undertaken by the student, HSD, midterm grades, lab evaluation assessment, seminar staging, assignment, appearance, and learner involvement were all employed by Abeer Badr El Din Ahmed et al. 2014 in their study to predict SAP [23].

The physiological appraisal advocated as a viable solution to the textual documentation process in this study is expensive (in terms of human resources, logistics, time, and space) and difficult to tailor to highly overcrowded classrooms. In light of these considerations, as well as the rising popularity of virtual learning environments in engineering, the LAP model covered was devised for assessing students' lab work in a

VEL environment using the simulated approach. The LAP model was devised to evaluate pupils' laboratory work in a VEL scenario. The model has been extensively defined, and the assessment results have been delivered. Future work will include accumulating huge data sets by throwing the model to use with actual pupils and actual facilities as well as the collaboration of more judges. Azzi and colleagues 2013 [24].

Shah & Anchor Kutchhi Polytechnic, in Chembur, Mumbai, provided data for Jyoti Bansode's 2016 study .The primary goal of Educational Data Mining (EDM) is to enrich instructional strategies. One of the major uses of EDM is anticipating academic achievement. Students' performance may indeed be evaluated using a decision tree. Pupils with bad productivity might be warned. Administration may boost their competence by paying more attention, delivering more lessons, and so on. Student efficiency can be affected as a byproduct of such initiatives. It is conceivable to minimize the number of failures. Inevitably, academic results rise as well.. They deemed the results taken from the very first semester to the sixth semester as the most significant student attributes, along with parent education, parent occupation, category, admission type, and SSC (board, medium, and class) [25].

Artificial intelligence is actively being incorporated into several projects as technology develops in order to increase efficiency. The framework for measuring student achievement is not an exception. Artificially intelligent chatbots were used to adequately tutor students on seminar-related research and make guidance for study material, according to a report published on May 13th, 2021. According to studies done at Leipzig University, many students benefited from the chatbots' assistance with study material after using the program. Using the Social Bot Framework, the researchers developed the bots LitBot and Feedbot. Feedbot availed use of T-REST MITOCAR's API. As for LitBot, data from the learners was acquired and shown using semantic web technologies like RDF and SPARQL. The results documented in the research shows that while many students were happy with the system, a substantial proportion yearned for

a real human connection on the other end of the bot. Furthermore, the bot's reaction lacked the uniqueness that the students wanted [26].

In paper [27] an emphasis is laid on the importance of technology in the educational field for both the providers of knowledge and the receivers as well. Apart from academic marking and grades obtained during the tests and evaluations, it is equally important to analyze the student's emotions and motivations toward a particular task assigned to them. To make the learning process efficient, it is equally important to recognize the status of the participants in the work assigned to them. Milos Kravick and Katharina Schmid concluded that AI-based mentoring of students can benefit students in both academic and personal development after conducting extensive research on the approaches used to mentor students. This paper lays emphasis on how educational concepts should be developed to scale individual mentoring and develop the student's personality. The authors also provide an enlightenment on the topic of mentoring and how it is different from coaching and tutoring. They bring forth the concept that mentoring is done according to the need and interest of the mentee and the mentoring process is more related to emotions rather than academics. This paper proposes the development of utilities that automatically facilitate the working parts of the mentoring process to make the work more efficient and more focused. The ones receiving the mentoring should also benefit from the automatic system and should receive fast and customized responses in real time. The researchers of the said paper have further reviewed the existing methodologies and already developed systems where Cognitive, Meta-Cognitive and Predictive methods are used like Fermat, Math Spring, Auto Tutor, Meta Tutor, SRL processes, PAL3 and MARi. Finally, the results summarize the requirements for AI-based knowledge services to support systems that provide automatic mentoring.

This work describes the use of a fuzzy linguistic summarization strategy for extracting linguistically interpretable fuzzy weighted rules from student data that describe

relevant correlations between student engagement variables and attained performance. The method is based on extracting scaled fuzzy weighted If-Then rules from data that provide a descriptive and easily understandable depiction of the most prominent association patterns between observed student engagement metrics and achieved performance. An intelligent framework was designed for evaluating the individual or group performance of students throughout the activity- and problem-based learning sessions. The method incorporates software and hardware plug-ins to monitor and harvest data on student activity and quality metrics during group-based ALL and CSCL activities. To uncover data clusters having similar learning behaviors and performance attributes, an adjustable unsupervised learning strategy was applied. The fuzzy LS was applied to the clustered and labeled data to produce weighted fuzzy rules. Educators may utilize the easy-to-interpret rules to monitor students' behavior on tasks and create tailored feedback for students based on their recognized learning behavior traits in order to perform better. The fuzzy LS strategy was applied to assess the effectiveness of employing GPM to deploy ALL appropriately in group activity activities completed by students on a doctoral Network Planning and Management curriculum. The fuzzy LS strategy was capable of separating paraphrasing rules, demonstrating that the use of GPM led to a decrease in the time that groups spent on the collaboration process while attaining greater results. The trials demonstrate how the fuzzy LS strategy can be used to successfully monitor students' progress and performance, allowing educators to transparently assess teaching practices and factors affecting student involvement in ALL.[28]

This study carried out a thorough review to evaluate the coordinated use of LA and computational ontologies driven by a taxonomy of educational objectives with main goal of tracking students' academic performance in remote education. The intent was to deepen students' grasp of the application of statistical tools that can aid in the interpretation of educational data and to gain a better understanding of strategies that enable students to monitor their academic performance in order to boost learning. Thirty-one articles were chosen from a total of 1230 studies to give evidence to address

one key research question and five subsidiary questions. The findings demonstrate a considerable rise in the number of studies, indicating a rising interest in this field of study, as well as a pattern in proposing Virtual Learning Tools and anticipating student learning achievement or failure. A related research gap was identified, as significant number of articles were not recovered that used Supercomputing Ontologies and LA in a cohesive way, and more explicitly, no work that used Taxonomies of Educational Objectives to assess student outcomes of educational objectives and performance. Furthermore, the articles obtained in this SLR demonstrate the need for technologies that assist instructors in continuously monitoring the academic development of remote learning pupils.[29]

CHAPTER 3

SYSTEM ANALYSIS AND DESIGN

3.1 AN OVERVIEW OF THE SYSTEM:

A student's performance monitoring system is an automated system implemented in java using the spring framework to classify slow and fast learners in a class and provide them with opportunities to enhance their capabilities. The faculty evaluates the student based on his/her performance in the class and the system generates an overall result of the performance. This allows for an efficient collection of the details and an easy classification of the students into fast and slow learners. Spring MVC i.e., the Model View Controller of the spring framework has been used to efficiently develop the system. The MVC is an architectural framework that aids in the separation of the web application into three main logical components: the model, the view and the controller. Each of the components are designed to have their specific functionalities which helps to design the web application efficiently.

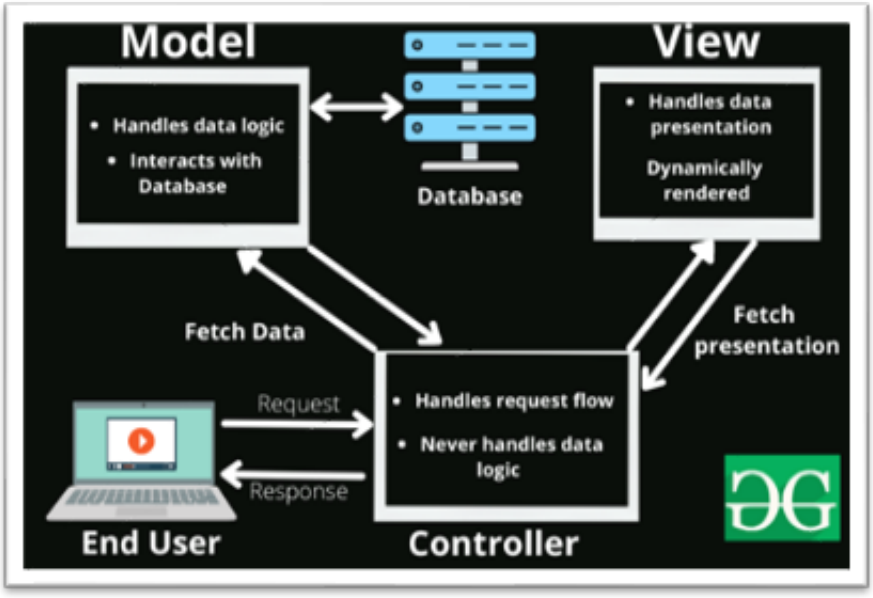


Figure 3.1 MVC Architecture and Design [30]

The model component of the MVC framework was used for the development of the data related logic of the application. Here all the data/objects that the user interacts with are handled by the model component. The addition, retrieval and modification of the data in the database is handled by this component. The objects of students and faculty were instantiated, their data modified and retrieved by using the model component. The view component of the MVC framework was used to handle all the user interface interactions. The data collected by the model component is used in the view component to generate a user interface. The controller component was used to develop the business logic of the web application. The working of the web application is basically dependent on the controller and the controller processes all the incoming requests from the user and directs them to the model which then generates a view. We have used this framework because of its ability to test the components separately since all the component's classes and objects are mostly independent of each other. This framework can be easily extended and is also suitable for TDD (Test Driven Development), where we first put down a small piece of code and then test it for cases. If the code passes the test, then it is finalized, otherwise the code is rewritten.

3.2 METHODOLOGY:

While carrying out the development of an experimental setup, a systematic and structured process is to be followed that allows the fabrication of inexpensive and high valued software in the set duration of time. The SDLC formalizes and provides an elaborate plan with stages, or phases, where each phase encompasses its process and deliverables. In this project, the waterfall model approach is being used. The waterfall model follows a sequence of steps and is suited for smaller projects with modules that are easy to define from the start. This model is very convenient and foreseeable, if the demands of the user are fixed, skillfully written, and easy to understand. Also, if the technology is well understood and the project is brief, then it is a good methodology to start with. It also allows departmentalization and control of modules. It also provides the determination of the end goal early and transfers the functionality of each phase clearly. The phases that will be followed are:

• Requirement Gathering and Analysis: All the possible must-haves of the structure to be set in motion are encapsulated at this juncture. For completion of this phase, an on-campus analysis was done where the group members interacted with the faculty and took note of the requirements for the project.

• System Design: This phase aids in enumerating the hardware and software prerequisites and helps in defining the comprehensive system architecture. The hardware and software requirements are enlisted further in the synopsis.

• Implementation and unit testing: In this phase, the actual implementation of the code is done. For successful implementation a satisfactory comprehension of the programming language and other software requirements is necessary. For this project java technology and spring, a framework is being used.

• Integration and System Testing: This stage is tremendously critical as the caliber of the product is regulated by the efficacy of the evaluating carried out.

• Operation and maintenance: Maintenance of the software is always done after the software has been deployed successfully. The maintenance is performed by every user once the software has been delivered to the customer, installed, and operational.

The student's performance monitoring system has five mod- ules with each module having its functionalities. Each module has control over the web application and serves an important role in the proper working of the application.

The working of the project starts with the login and registration page where the user, whether it is a student or faculty registers themselves on the web application with their email addresses. Then once the registration is completed the users logs in to the system and based on the role their respective profile opens. The functionality depends on the role of the logged in user. If the user is a student, then he/she can update his/her details and upload the previous year's academic results as well as the mid-semester results. These updates will have proof in the form of a photograph attached to the results.

For the faculty there are some roles prescribed. A faculty member can have a role of the HOD where he/she can set the percentage criteria based on which students will be categorized as slow or fast learners. The HOD is also responsible for the allotment of mentors to a group of students and set a coordinator for each semester/year. The HOD keeps the overall track of the number of slow learners and fast learners every semester. The faculty member can also have the responsibility of a coordinator wherein he/she is expected to add subjects in the semester, assign career/personal counseling sessions for the students and notify about the same through notification broadcasting. The coordinator can also organize parent-teacher meetings and make the student aware of the same. The coordinator can also provide a form to the student wherein he/she can ill in the reasons for less attendance, problems being faced by the student personally or academically.

3.3 TECHNOLOGY USED IN THE PROJECT:

The technology used in the said project is core java along with features of advanced java for the backend development, MySQL for the database and bootstrap templates for the frontend. It is a part of Java programming language. It is an advanced technology or advanced version of Java specially designed to develop web-based, network-centric or enterprise applications. We selected this technology because all the group members are satisfactorily versed with the language and have acquired training in the said technology. JEE (advance Java) provides libraries to understand the concept of Client-Server architecture for web- based applications. Another reason for choosing java advance over the core java was that we can also work with web and application servers such as Apache Tomcat.

3.4 REQUIREMENTS FOR THE DEVELOPMENT OF THE PROJECT:

To facilitate implementation of the experimental setup, installation of some software is required, and the hardware or system must satisfy the minimum requirement for the proper installation and working of the required software. The software required for the successful implementation of the project includes an integrated development environment of java which contains basic workspace and an extensible plug-in system for customizing the environment. Java J2EE (enterprise edition), a popular java development IDE is chosen in this regard. Since our project has a requirement to connect to a database, Microsoft SQL server can be used. The minimum operating system required to work with previous versions of SQL Server and J2EE IDE is Windows 7 64-bit. The Eclipse IDE pocket guide recommends a processor of speed 1.5 Ghz which is provided by intel's i3 processor. 1 GB of memory and 256GB of hard disk drive is required for efficient execution of the modules. To summarize it all:

1. MINIMUM SOFTWARE REQUIREMENTS:

• J2EE Enterprise Edition for developing the web application

• Microsoft SQL Server for database connectivity

• Windows 7 Operating System

2. MINIMUM HARDWARE REQUIREMENTS:

• Processor: Intel core i3

• Hard Disk Drive: 256GB

• Memory: 1GB

3.5 PROPOSED FRAMEWORK IN THE METHODOLOGY:

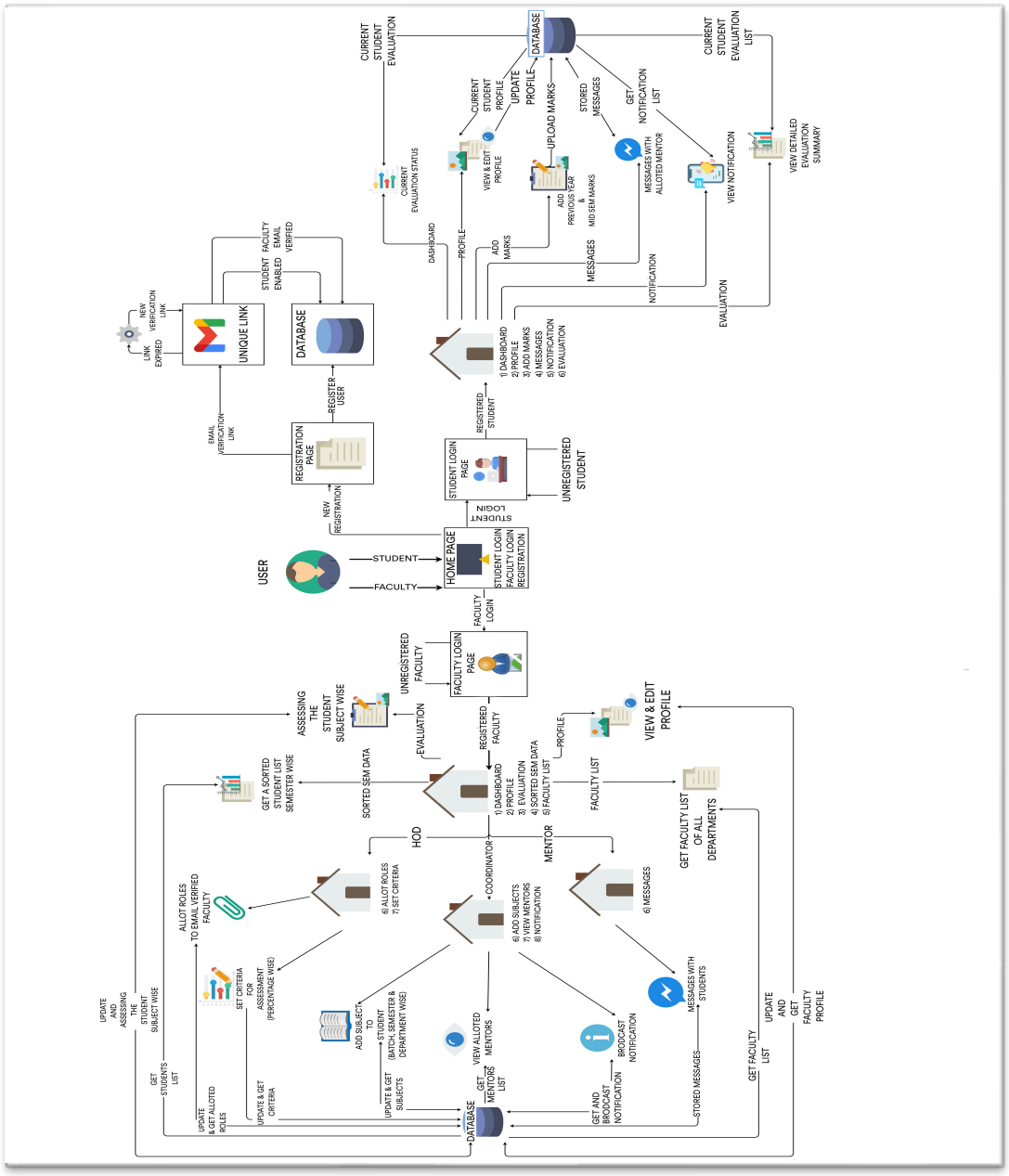


Figure 3.2 Conceptual Framework of SPMS

The conceptual framework of the experimental setup proposed in this paper is shown in fig 3.2 above. The system is a web application that is implemented in java enterprise edition.

Spring MVC framework of java is employed to establish the structure of the system. The model, view, and controller of the MVC framework help to develop the web application in an efficient and organized way. The system initiates with a user dashboard where users can either register themselves for using the system or can directly log in to the system using their email id and password. For logging into the system, one should have registered to the system using the email id and on registering the user must provide the role as to how he/she is going to utilize the system. The roles can be of a faculty or of a student. Once the user provides the email for registration then a link is sent to the specified email of the user and the user must verify the authenticity of the email provided by doing so, he/she validates himself/herself as an authentic user of the system. After the email verification is done the user is now a valid user of the system and can now log on to the system using the same email and password that he/she provided at the time of the registration. If the email and password match with the database entry the user is given access to the homepage. The homepage is slightly different for the student and faculty. As expected, the faculty will have more functionality as compared to the student.

The registered student, after logging in, interacts with the dashboard which shows the current evaluation status of the student. This evaluation result is extracted from the database wherein it is stored after the faculty evaluation and uploading of the results by the said student. The results can be uploaded by the student in the add marks section of the dashboard where he/she can upload mid-semester marks and the previous year's results along with a screenshot as proof. The student also has a feature for profile updating where he/she can update and view his/her profile. The student can also view the detailed evaluation of the semester and can track his/her progress in the semester under the evaluation tab. An important feature that a student can use is the messages and notification section where he/she can view the notifications posted by the coordinator regarding seminars, counseling sessions, parent-teacher meets, etc. The chat section is only enabled for the student who is categorized as a slow learner. With this feature, a slow learner can interact with the mentor personally.

The registered faculty once logged on views the dashboard which comprises the profile updating, where the faculty can update their profiles, evaluation, where the faculty can provide the assessment of the students according to the subject they teach, sorted semester data, where the faculty can view the sorted data of each semester and faculty list, where the faculty can view the registered faculty. Faculty can opt for three roles viz coordinator, mentor, HOD.

The coordinator can add subjects, view mentors according to semester, and broadcast notifications to each batch for slow and fast learners respectively. The mentors can provide one-to-one and one to all guidance to each slow learner through the chat system feature provided. He/she can also add subjects to their respective semesters

The HOD can set the criteria of each assessment as to what percentage of each contributing element holds in the overall evaluation of the student and the HOD can also determine the threshold of the percentage above/below which the system categorizes the students. The HOD also holds the power of allotting roles to the faculty members according to the protocols.

3.6 DATA FLOW DIAGRAMS:

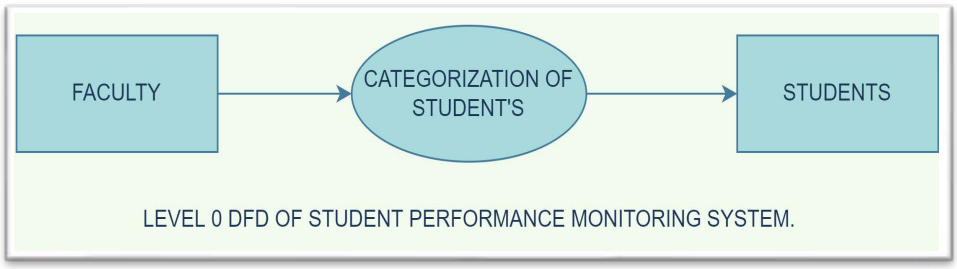


Figure 3.3 Level 0 DFD

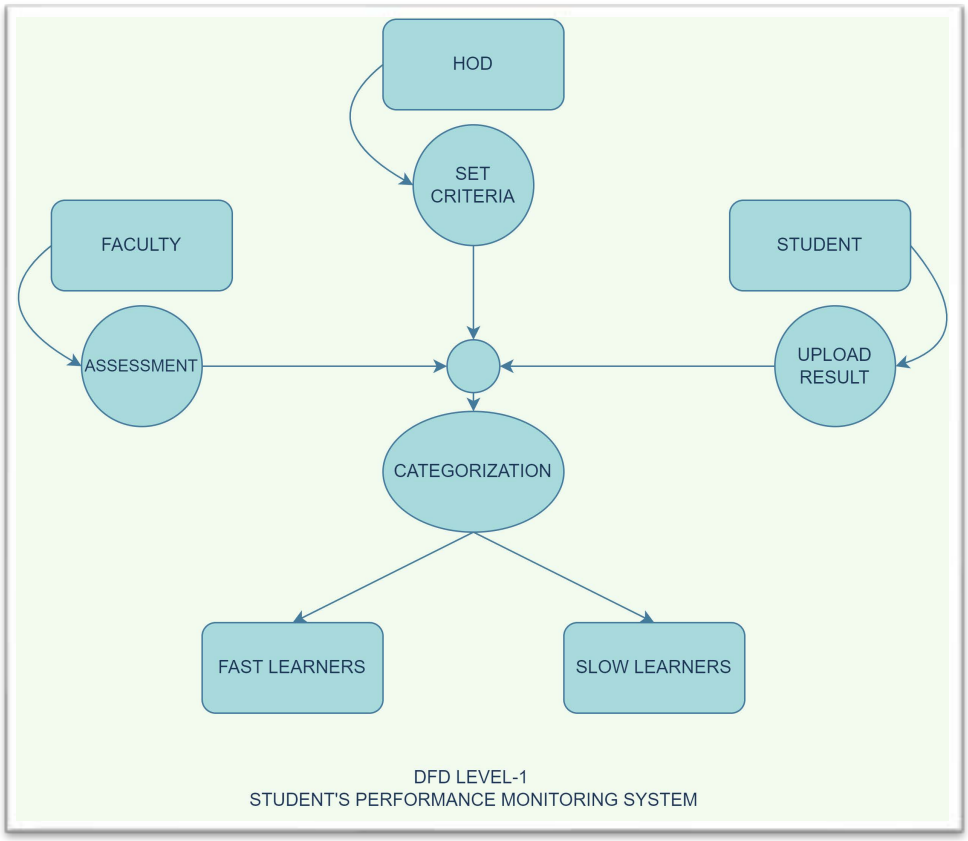


Figure 3.4 Level 1 DFD

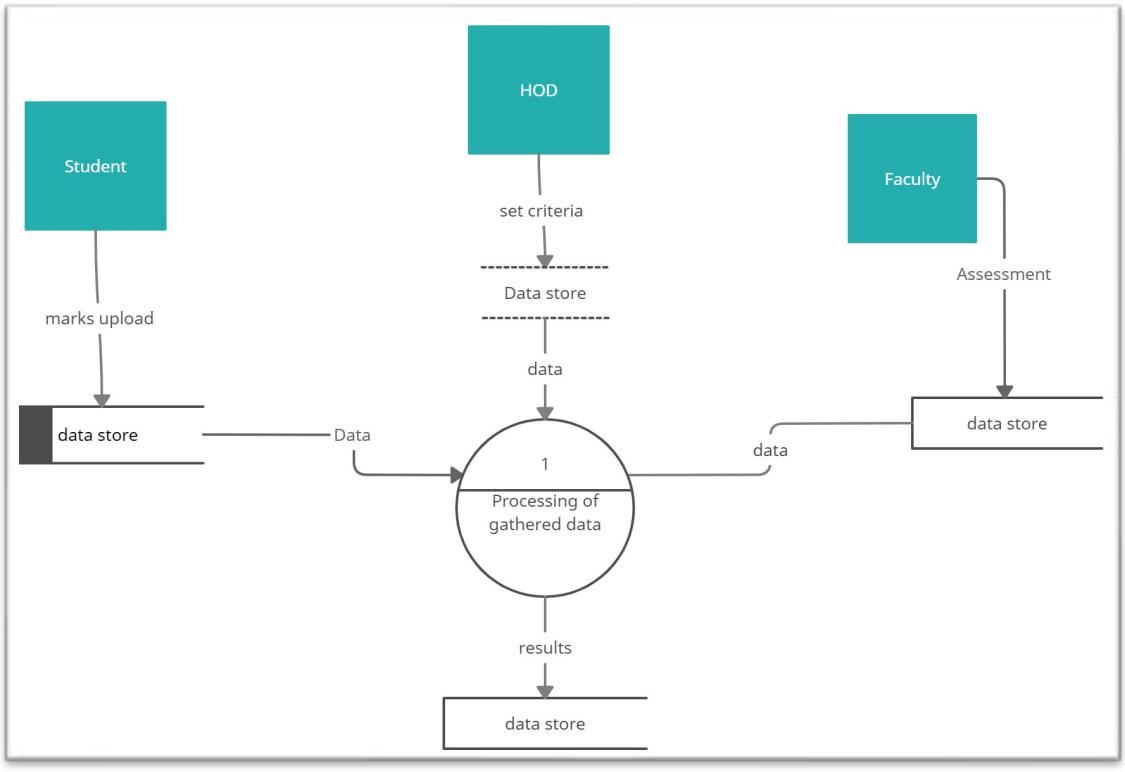


Figure 3.5 Level 2 DFD

The data flow diagrams as shown in fig 3.3 and 3.4 and 3.5 of the proposed system provide the overall view of the system where faculty makes an evaluation which facilitates the categorization of the students. A detailed pictorial representation is provided as to how each module or entity works in making this system a successful one. The entities perform some functions, and each entity is entitled to a required function which is to be performed by the entity to carry out the tasks of the software successfully.

3.7 FLOWCHART:

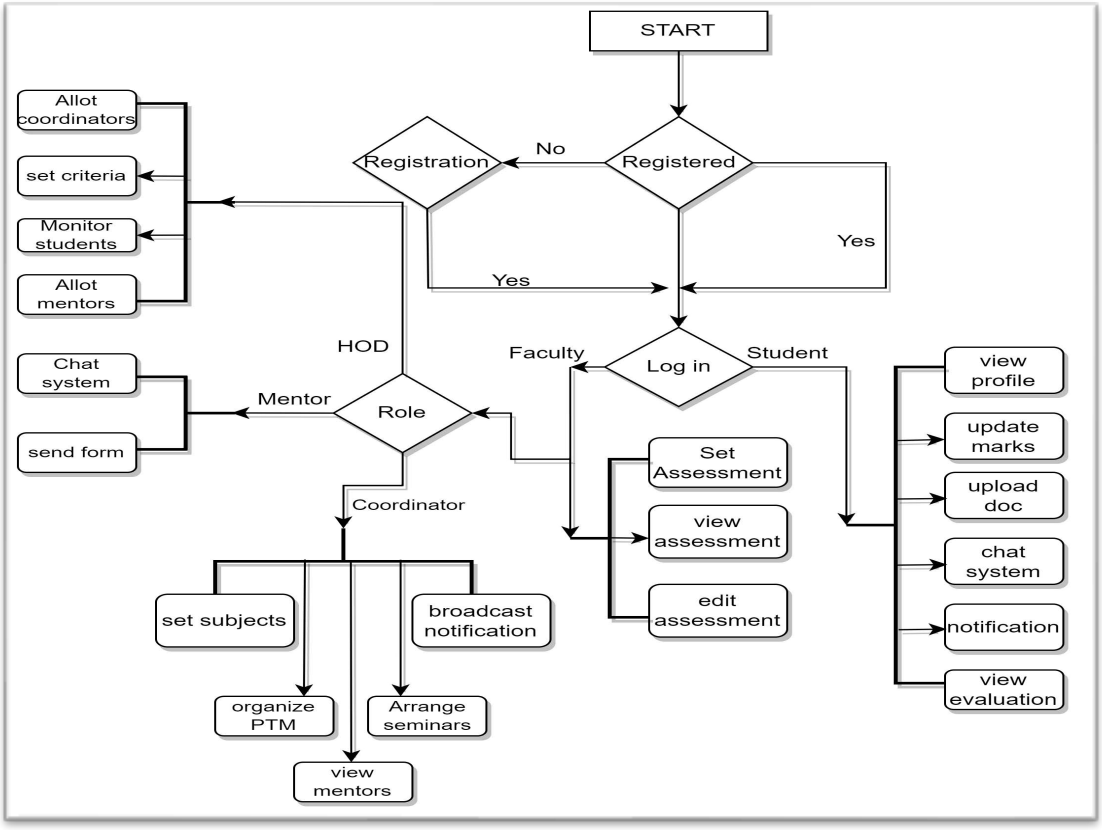


Figure 3.6 Flowchart of Student's Performance Monitoring System

Fig 3.6 provides a flowchart wherein the flow of control is depicted in an efficient manner. The system starts with the registration process which redirects the user to a registration form which is the same for all, the student as well as the faculty. The registration form asks for the e-mails which are then verified by the system. Once the verification is done then the user can login to the system based on its role. The role can be either of a student or of the faculty. After logging in, each module has its own functionality. These functions are already discussed in the methodology of the given research paper. The system is currently in its development phase where 80-85% of the work has been completed and each developed module has been unit tested by providing dummy test datasets. Once the system is completed an integration testing phase will start wherein, we will take the actual dataset from the college.

3.8 ALGORITHM:

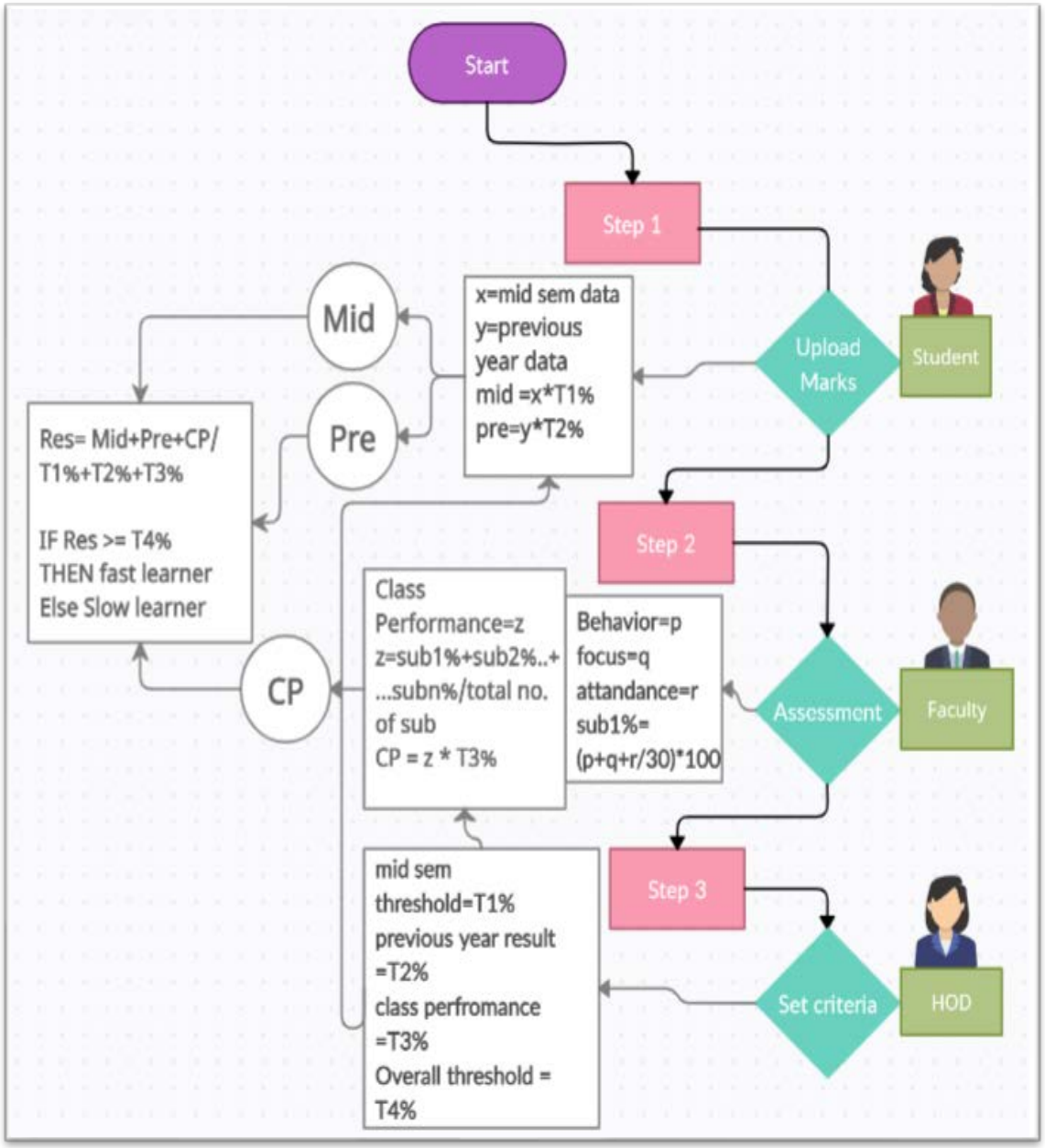


Figure 3.7 Algorithm

The algorithm of the system uses the gathered data and manipulates it using some mathematical equations mentioned below. This computation helps to categorize fast and slow learners.

Step 1:

The student uploads a mid-semester result and previous year result.

Let the mid semester result be x,

Therefore, the contribution of this criteria is computed as:

$$mid = x * T1\% \quad (1)$$

Where T1% is the threshold percentage of mid semester set by the HOD.

Similarly,

Let the previous year's result be y,

Then contribution is computed as:

$$pre = y * T2\% \quad (2)$$

Where T2% is the threshold percentage of previous year set by the HOD.

Step 2:

Subject-wise evaluation is done by assigning a pointer out of 10 to each student based on their behavior, attentiveness in the class and their attendance in the subject. Then the overall subject evaluation percentage of one subject is calculated as:

$$Sub\% = (\text{Behavior value} + \text{attendance value} + \text{attentiveness value} \div 30) * 100 \quad (3)$$

Here 30 is the total weighted value of all the attributes.

The formula to calculate the aggregate percentage of all the subject wise assessments (Class Performance) in percent format is:

CP% = [(sub1% + sub2% +... sub n %) ÷ total no. of subjects filled] * 100 from (4)

Step 3:

The head of the department dynamically sets the criteria threshold as:

T1% = Weightage of Mid semester results

T2% = Weightage of previous year results

T3% = Weightage of overall class performance

T4% = Total threshold of all the criteria

The segregation of the students is based on the calculation of overall evaluation percent based on percentage controller which is set by the head of the department:

Formula used to calculate the overall percentage is given as:

Res = (mid %+pre %+CP) ÷ (T1% + T2% + T3%) from (1) (2) (4)

If Res >= T4%,

Then student is fast learner

Else, student is slow learner

3.9 ENTITY RELATIONSHIP DIAGRAM (ERD):

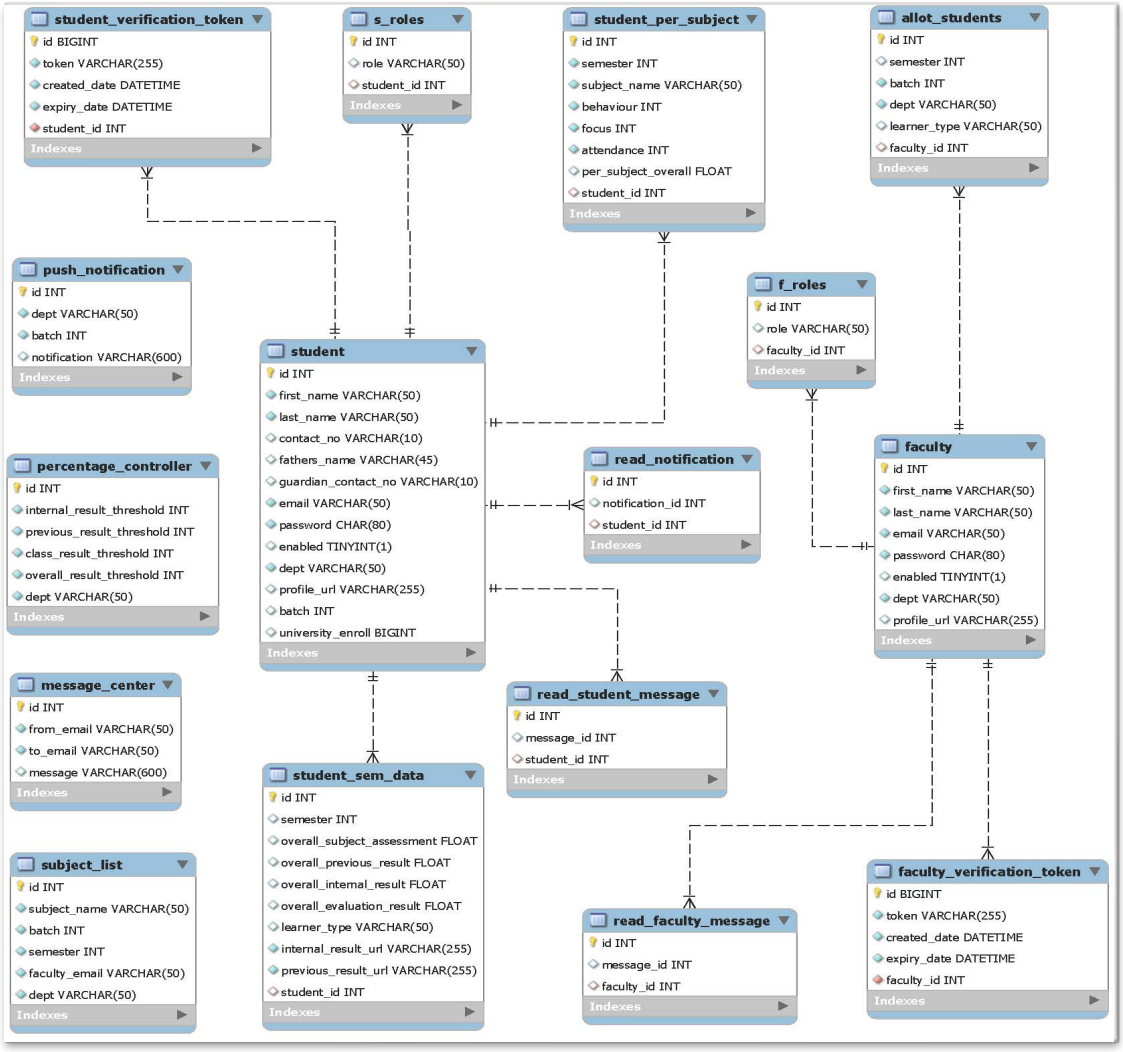


Figure 3.8 ER-Diagram for Student's Performance Monitoring System

The entity relation diagram depicts the relationship between various entities involved in this project/system. This ERD illustrates how "entities" such as students, faculty and subjects relate to each other within the system. This ERD depicts the design and is used to further debug the relational database. This diagram was initially used to determine requirements for the information system project. Later it was used to model the database "Student's Performance Monitoring System". The dataset used in this project was collected

from the college under the supervision of the guide. The main entities involved in this project are the student, faculty and subject the faculty teaches a particular group of students based on the semester. The entity “student” has various attributes like first_name, last_name, contact_no, email, password, and id where id acts as a primary key and links the student to student’s subject, semester data and notification or message sections. The entity “faculty” has attributes like email, password, dept, profile_url, and id where again the id acts as a primary key which associates this entity with allotment of students, message and notification functions and the email links the faculty to the subjects and verification code. Various other tables are included in the ERD, which ensures the system works in a better and efficient way and handles as many queries as possible with ease. The faculty has an important role of providing subject-wise assessment to the students and help in effective monitoring of their behavior in the class and provide an insight of their interest in the subject. The faculty can also view their given assessment and edit it if required. Some of the faculty members can work as mentors for the students. These faculty members have the functionality of providing counseling sessions as scheduled by the coordinator, attending parent-teacher meets, solving grievances of students through chat system, and reviewing the forms filled in by the students. The mentors are allotted to slow learners only as they need guidance.

3.10 TESTING OF THE SYSTEM:

The process of verifying and validation whether a software or application is bug-free, meets the technical requirements as guided by its design and development, and meets the user requirements effectively and efficiently by handling all the exceptional and boundary cases is called testing. This process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy, and usability. It mainly aims at measuring the specification, functionality, and performance of a software program or application.

1. Verification: it refers to the set of tasks that ensure that the software correctly implements a specific function.

2. Validation: it refers to a different set of tasks that ensure that the software that has been built is traceable to customer requirements.

In our testing phase, the modules developed were initially tested just after the completion of each function. This implies that unit testing was done when the modules were developed. The verification of the project was also accomplished during the development phase. The test data was provided to the system to check the system's functionalities and those were recorded to be satisfactory. Once all the modules were unit tested then the integration of the modules was done and integration testing, verification and validation were again performed to double check the working of the system. Some errors were encountered and corrected. Errors related to the databases were also encountered during both the unit testing and integration testing where some of the queries failed to execute due to either missing fields or data. Those errors were also resolved during the testing phase. The user's input data is also validated against the rules, example a user cannot enter a number or numerical where alphabet is required, or a user cannot miss a required field.

CHAPTER 4

RESULTS AND OUTPUTS

4.1 RESULTS:

The student's performance monitoring system yields a satisfactory result where the segregation of fast learners and slow learners can be easily identified. The results obtained were at par with the data set collected from the college where 30 students from two different batches were evaluated for three semesters. These results will be proven to be beneficial to faculty, especially the head of the departments where he/she can identify the fast learners and slow learners of any semester and batch and then can take necessary steps to further enhance their capabilities. For fast learners the faculty can take measures such as organize seminars, workshops and career counselling sessions whereas for slow learners the faculty can reach on conclusions whether to reach out to the parents of the student or to organize remedial classes for the student. By automatic result generation where the calculations are performed by the system itself, it becomes very convenient for the institutional members to improve the result of the class and it also plays a major role in improving the overall result of the institute. This automatic system's result generation proves to be beneficial as it does not take much time to register with the system and the evaluation of each student is done by a particular faculty member teaching the student, therefore making the evaluation precise and less time consuming. The only condition being that every teacher teaching the student must provide the evaluation and student must have uploaded the results. This system also provides a dynamic setting of criteria with which the head of the department can easily change the criteria whenever needed and the outputs/ results are updated accordingly without any modification in the subject-wise assessment. A striking feature of this web application is that it is both suited to be used on a pc or laptop and can also be used with the same ease on the mobile phone without downloading any application. The feature of scaling itself or the pages of the application to adjust to any screen size is a beneficial function which saves the extra step of installing an android application and utilizing extra memory for the same.

4.2 OUTPUTS/SCREENSHOTS:

The following pages will depict the working of the system through the screenshots taken at the time of adding the data into the system. The user is presented with the homepage whenever he/she is accessing the web application. It is presented with a student log-in and a faculty log-in. Also, a registration tab is provided so that the students and faculty can first register themselves to use the system.

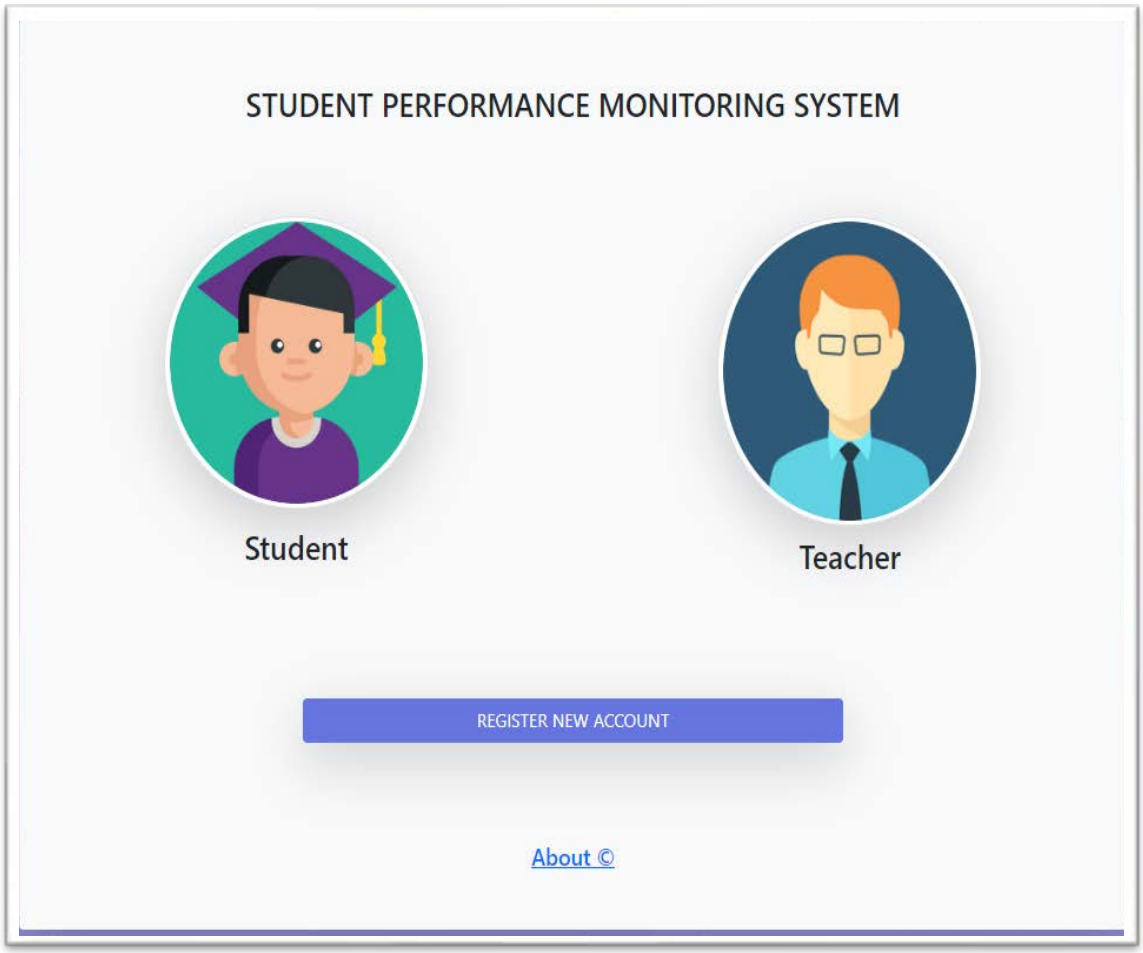


Figure 4.1 Homepage of Student's Performance Monitoring System

For the registration process for both students and faculty a form is displayed wherein the user must enter his/her required details and specify the department and the registration type whether the user is faculty or student. The rest of the details like enrollment and semester can be updated in the profile updating section of the web application.

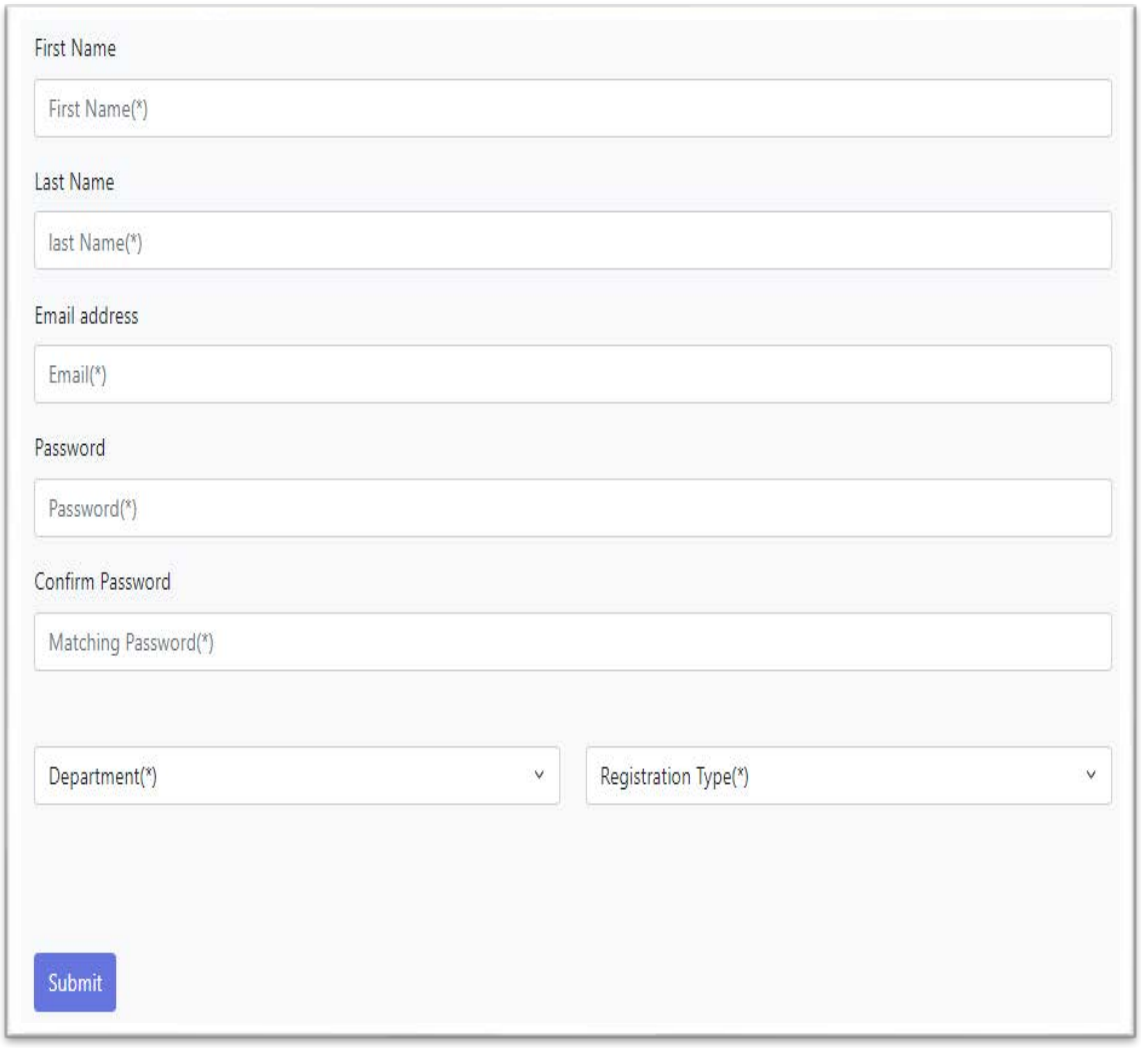


Figure 4.2 Registration Page of the Student's Performance Monitoring System

After clicking on the submit tab a page is displayed mentioning that a verification link has been sent to the email id provided by the user. This is done to increase the security of the system and that only valid users use the system.

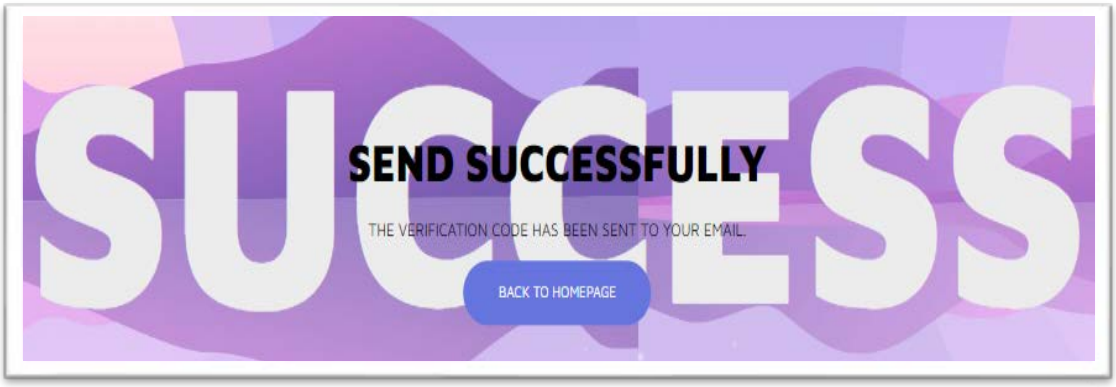


Figure 4.3 Verification code sent to email

The email verification link is then received by the user and this link is active for 30 minutes for the user to verify their email id and validate herself/himself as a valid user.

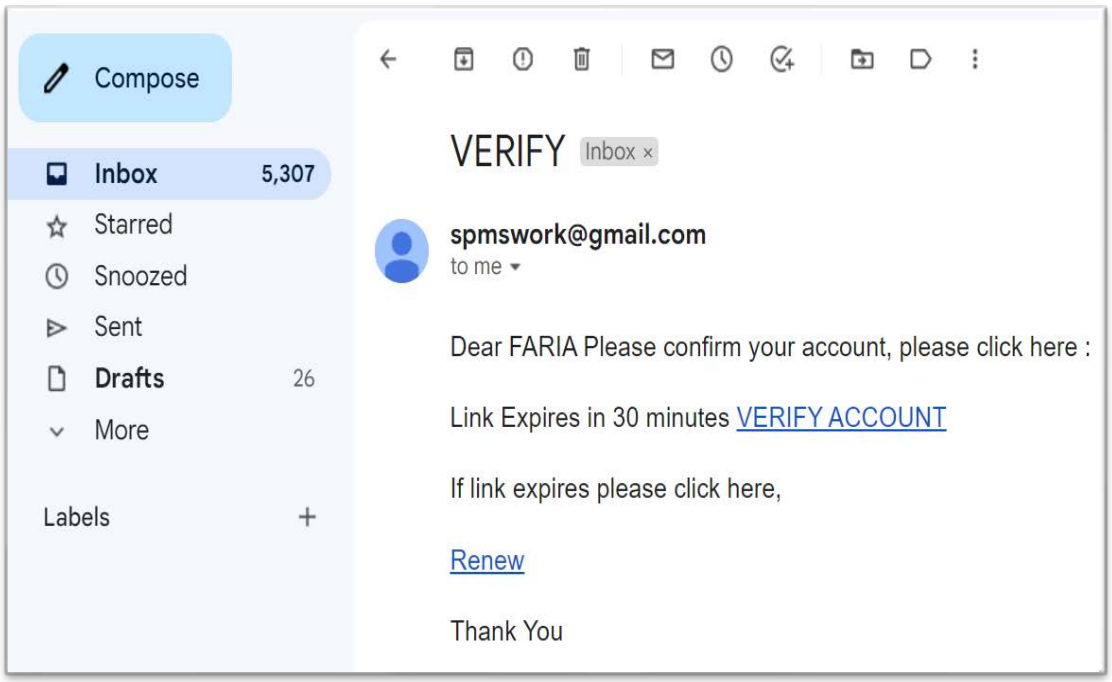


Figure 4.4 Validation of user through verification link

Once the user verifies his/her email id and validates herself/himself as a valid user then his/her account becomes active and now he/she can log on to the system and proceed with the next step of profile updating.

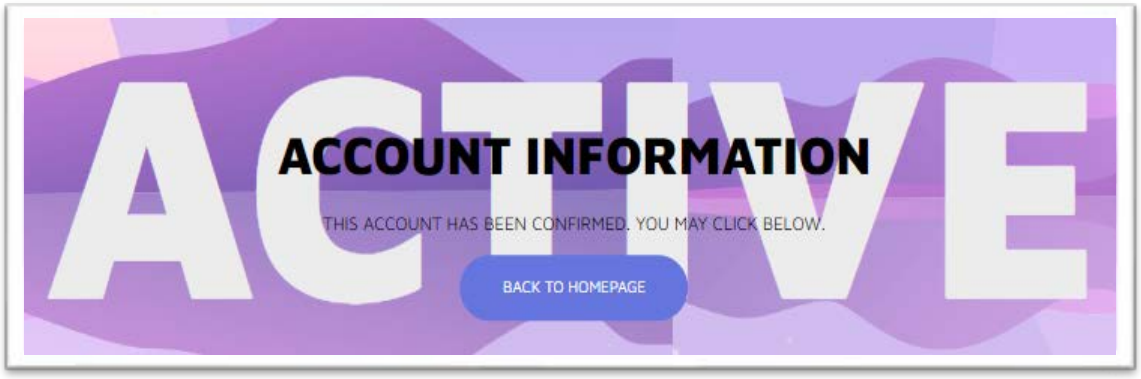


Figure 4.5 Activation of the user's account

The log in page of the student and the faculty consists of two input fields where the user provides his/her verified email id along with the password. If the email and password are correct and match with the entries in the database, then the user is given access to the system otherwise a message of "invalid user/id" is displayed.

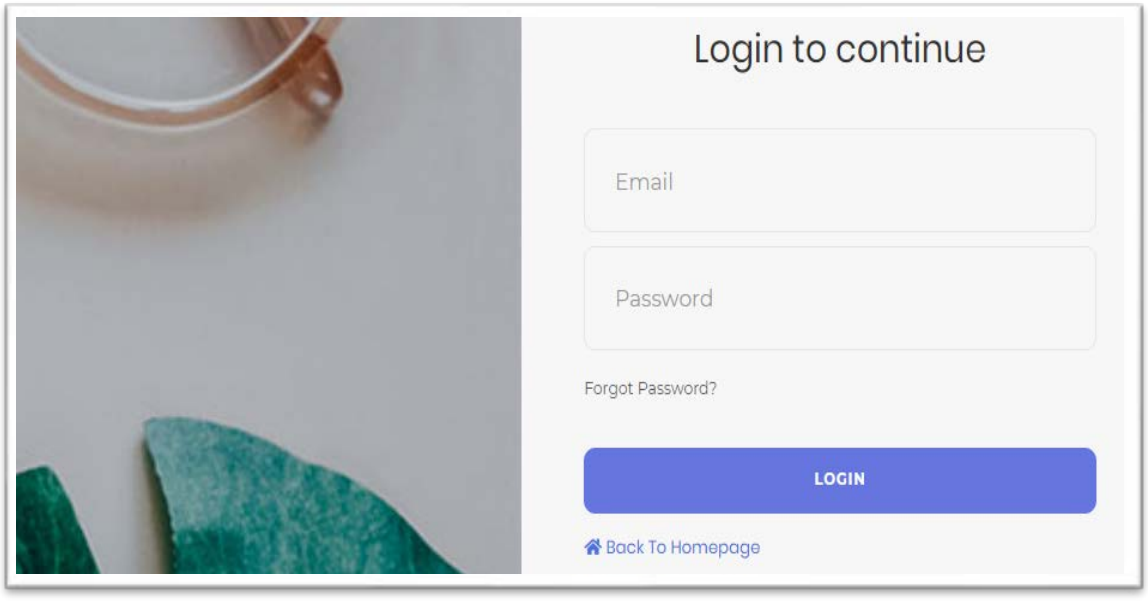


Figure 4.6 Log In page for the user

4.3 STUDENT FUNCTIONALITY OF THE SYSTEM:

The student's dashboard presents a view of the profile of the student along with the current semester overall performance which is already evaluated by the faculty. It also depicts whether the student has uploaded their results and shows the mid semester result and internal result updating status. On the left-hand side, a list of functions is presented.

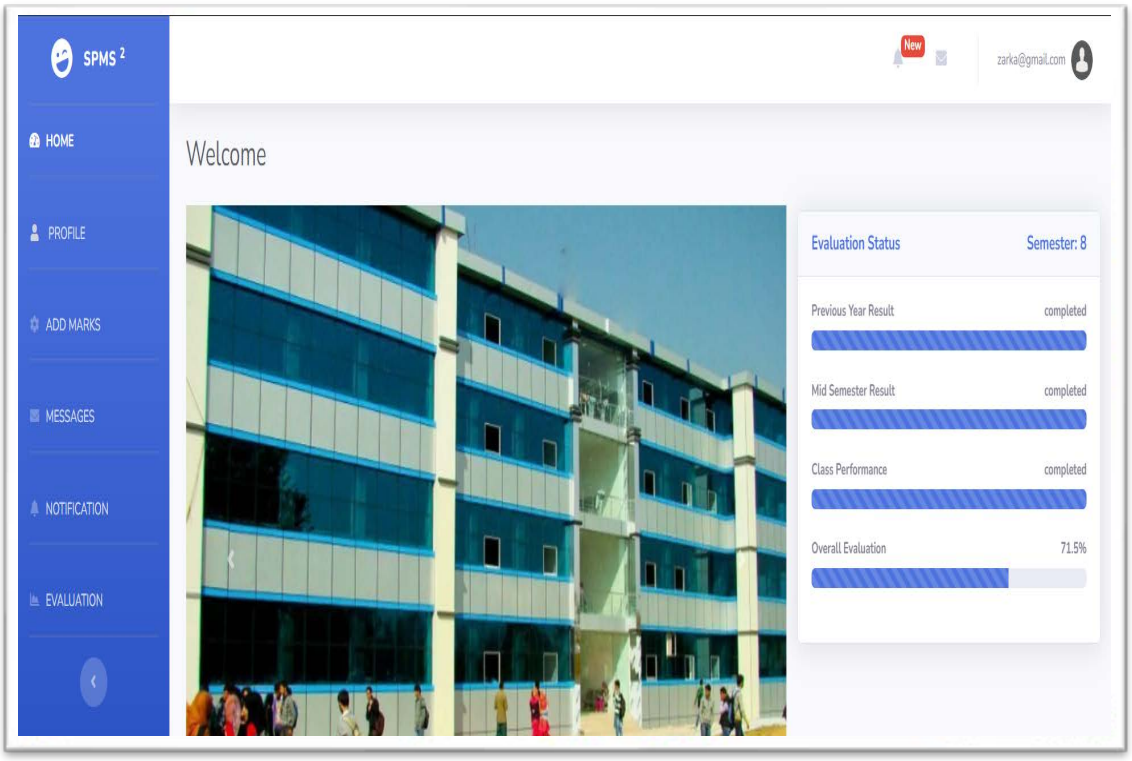


Figure 4.7 Student's Dashboard

In the profile function of the system the user is prompted to update his details like contact details, semester, enrollment, profile picture uploading and even the student is prompted to fill in the guardian details along with their contact number. This enables the faculty to contact the guardian whenever the need arises.

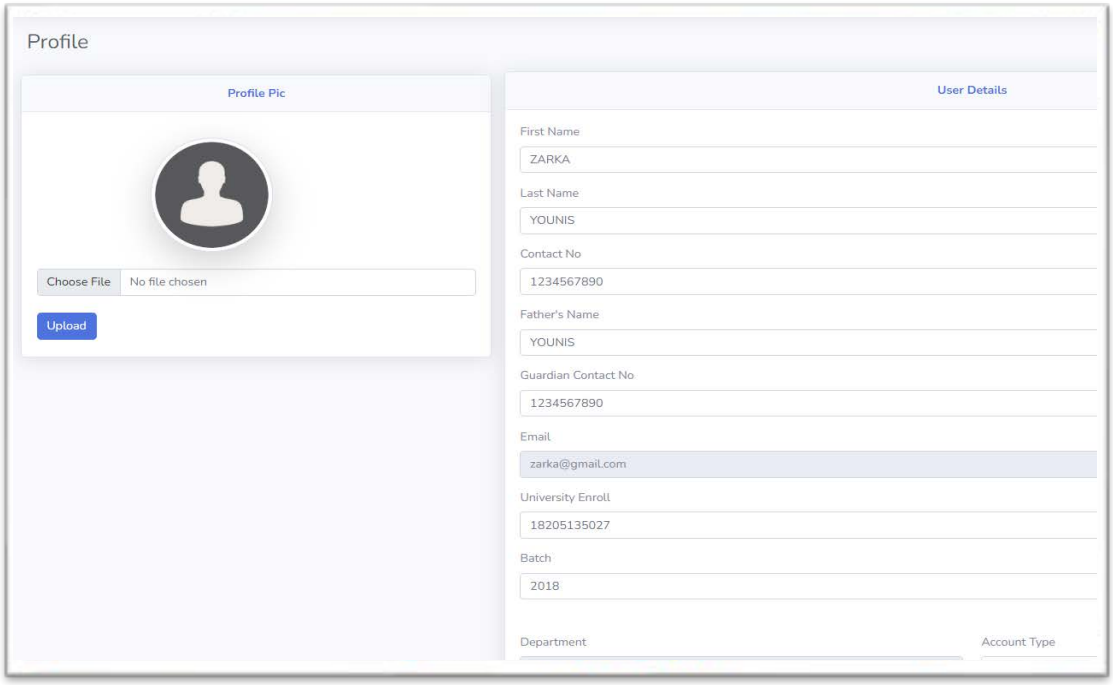


Figure 4.8 Student's Profile Updating Feature

After updating the profile of the student, then the student is prompted to upload the marks/ result of the current mid semester examination and the marks of the previous year. The marks uploaded are in the form of percentage and not CGPA.

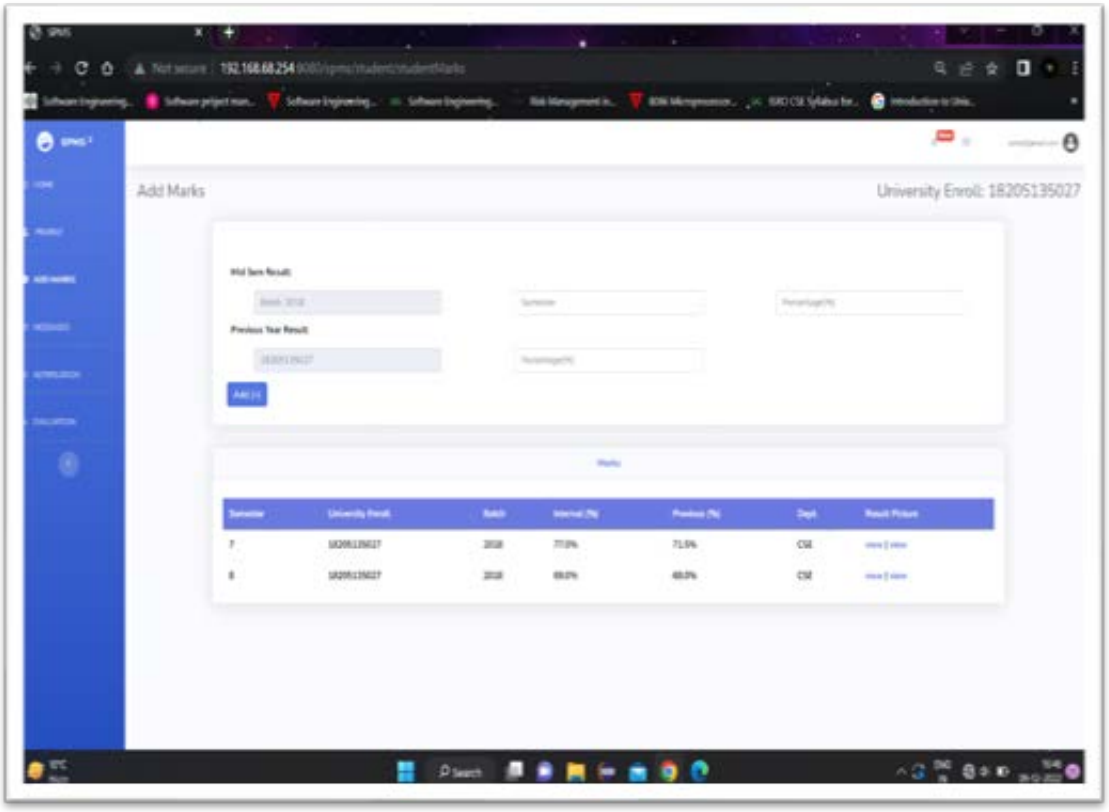


Figure 4.9 Marks Uploading

After the addition of the marks along with semester, the user is prompted to upload a file that will act as a proof to support their claim of the results. This report can be a screenshot of the result which is already available on the student portal or university website. After successful completion of the format the marks are added which can be viewed below.

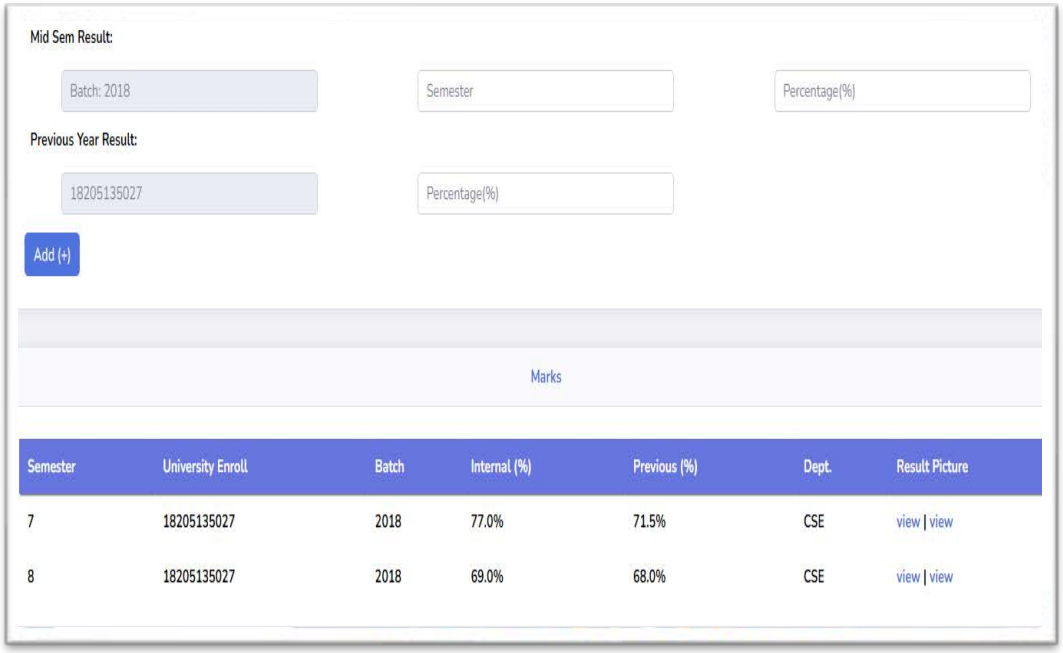


Figure 4.10 Proof Uploading in the form of picture.

There is also a notification viewing section where every notification posted by the coordinator for the batch in which the student is, is posted. The notification alert is also generated at the bell icon of the profile whenever a new notification is posted. Previously posted notifications are also enlisted and are not removed while posting a new one.

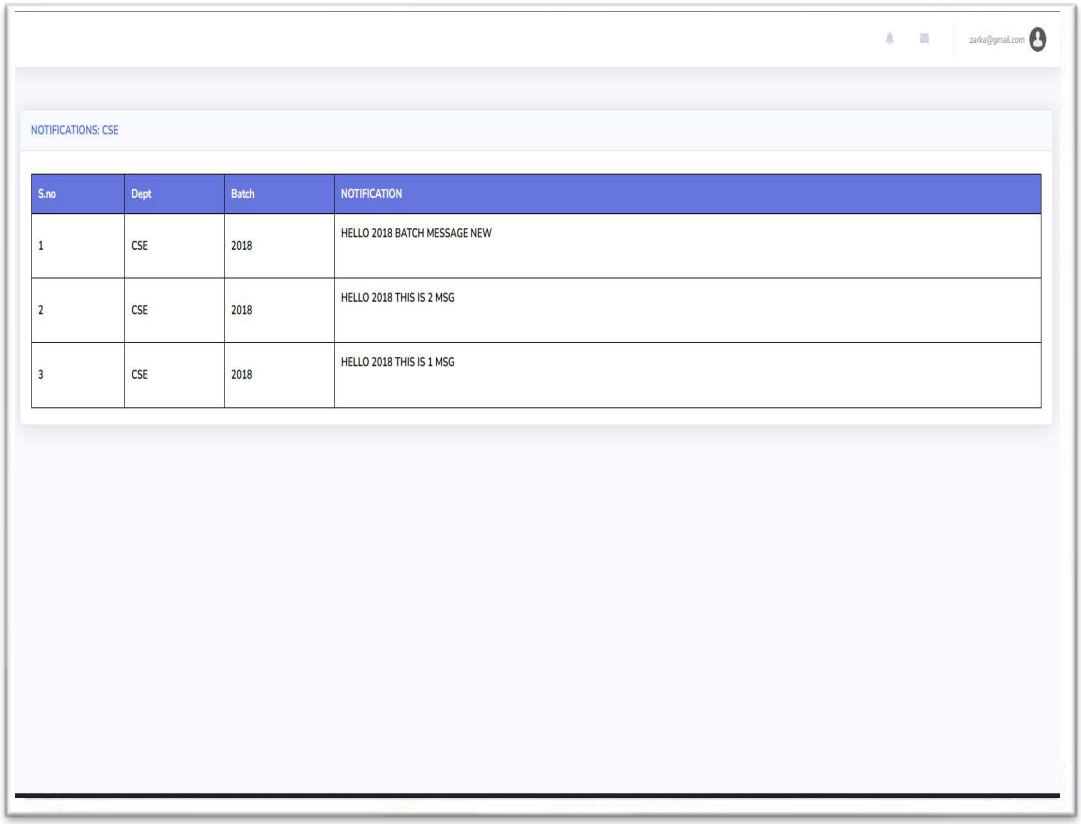


Figure 4.11 Notification Area for students

The student can also view the evaluation report where he/she is categorized into fast/ slow learners along with the percentage obtained. In this example the student is evaluated for 7th and 8th semesters and in both semesters the student has been categorized as a fast learner. This gives the student a sense of responsibility and accountability and it motivates the student to perform even better in the upcoming semesters.

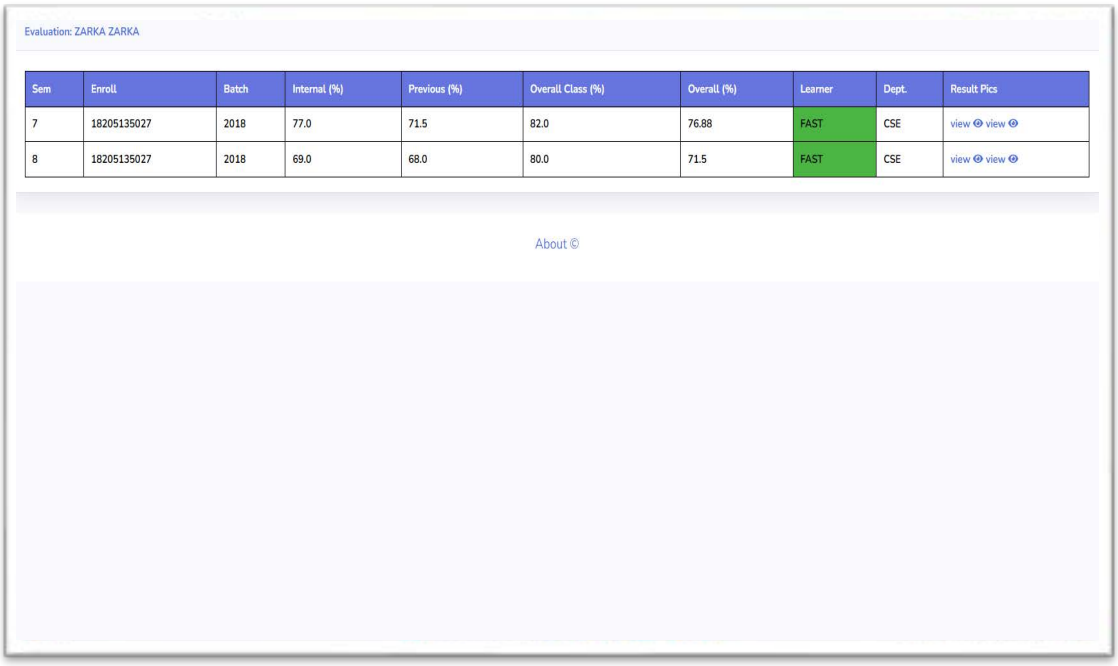


Figure 4.12 Evaluation Report of Student

A slow/fast learner who has been assigned under a mentor has a message section also where he/she can contact and connect to the mentor regarding any personal counselling or any personal meeting. The messaging is one to one and is initiated by the mentor at the first place. The student can reply to the message once the message has been received and like the notification alert a message alert is also visible at the messaging icon if any new message is received by the student. This message system works on the emails of the sender and the receiver.

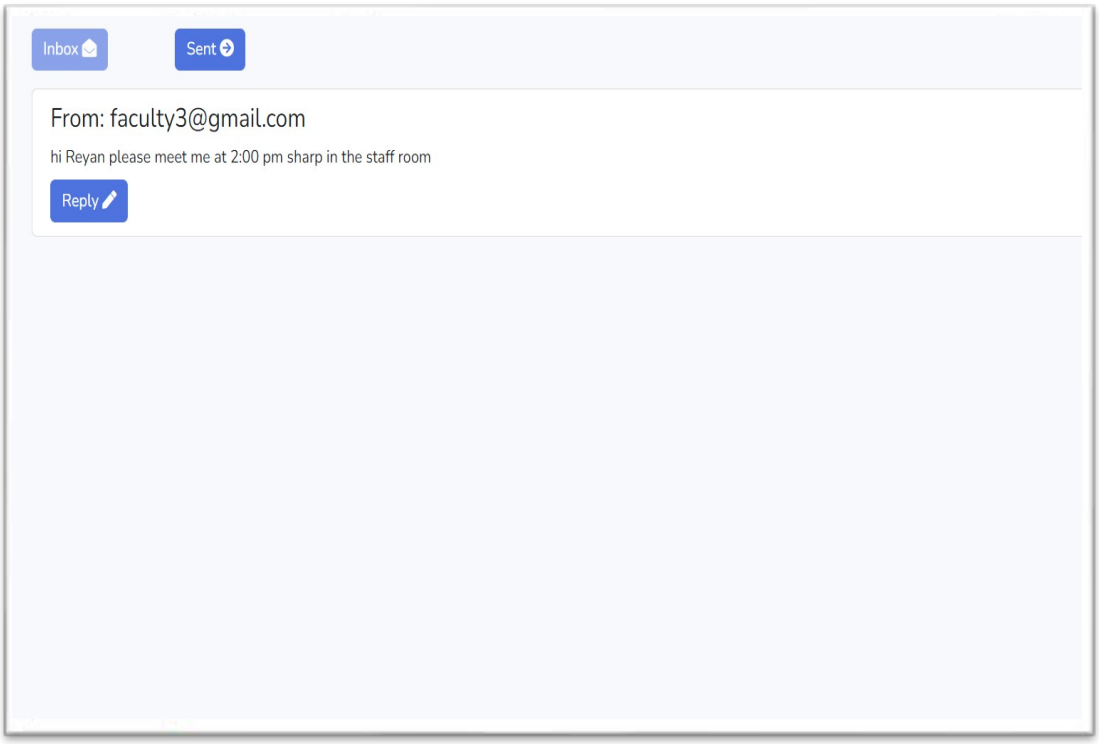


Figure 4.13 Messaging

4.4 FACULTY FEATURES OF THE SYSTEM:

There are various functions a faculty can perform, and these are specifically divided based on the role of the faculty. The basic registration and log in process of the faculty is same as the student, a difference being that once the student verifies the email link his/her account is activated. But for the faculty, the account is not activated unless and until the head of the department validates the faculty as a verified person.

He/she does so by allotting the role of faculty in front of their email addresses which then in turn activates their accounts and they are deemed as valid users of the system. The email address of the head of the department is directly inserted into the database using SQL queries so that his/her account is activated automatically without any verification. This can be done by a database administrator. The dashboard of the head of the department looks like:

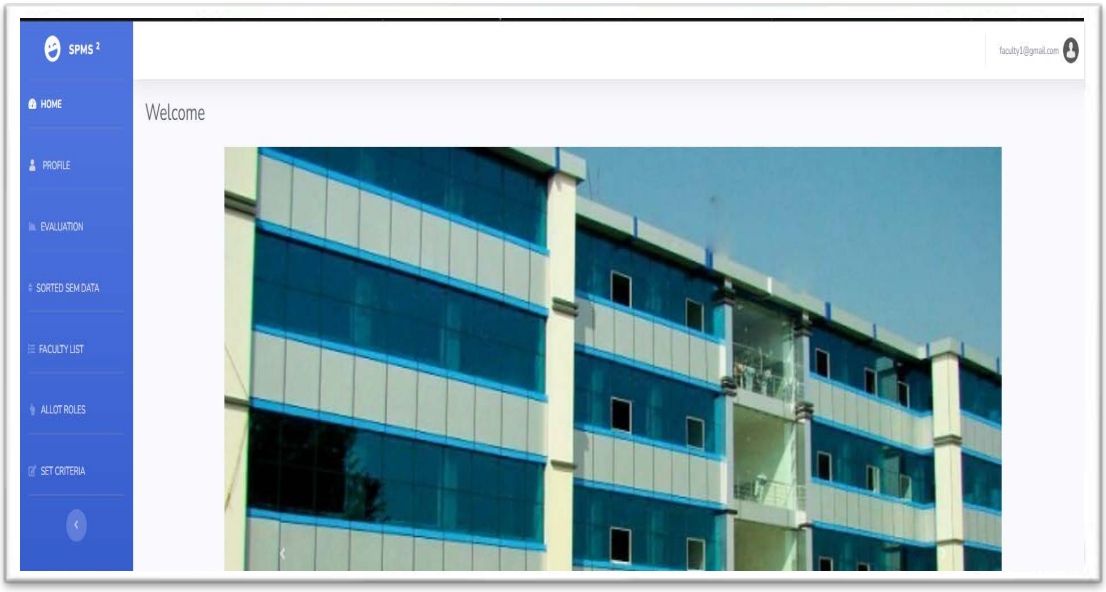


Figure 4.14 Dashboard of the HOD

The profile window of the Head of the department consists only of profile picture updation and name editing/ role editing.

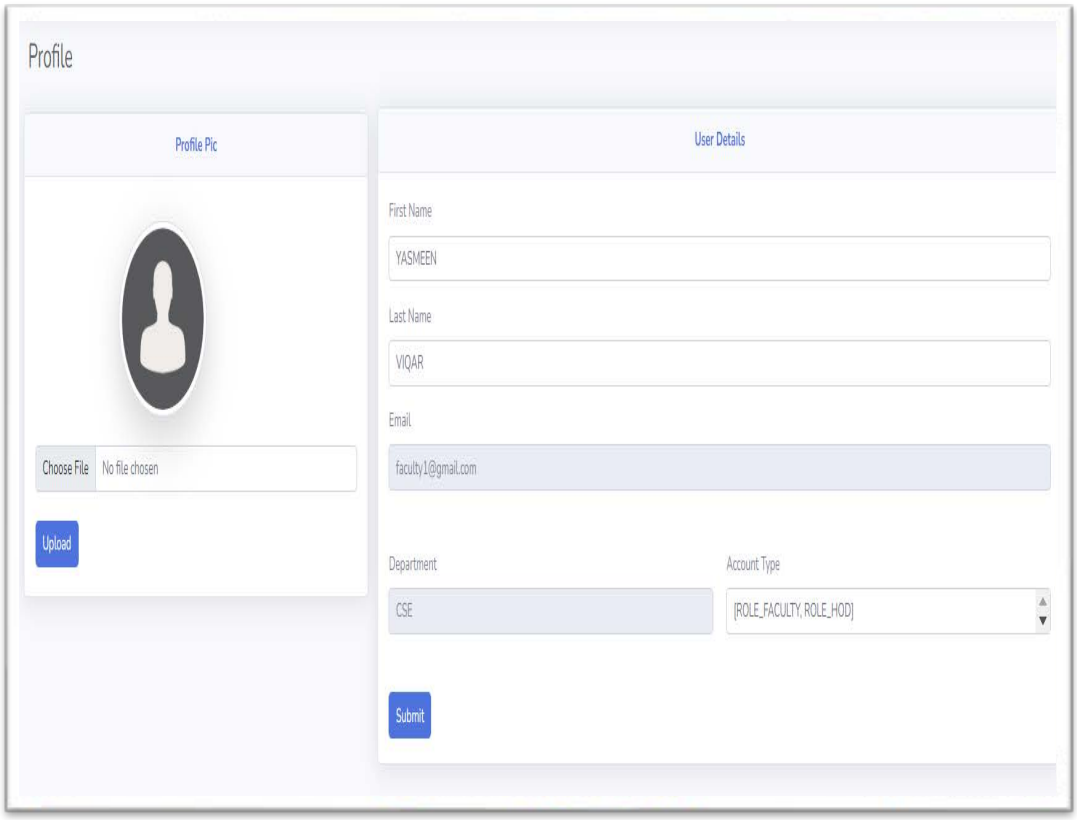
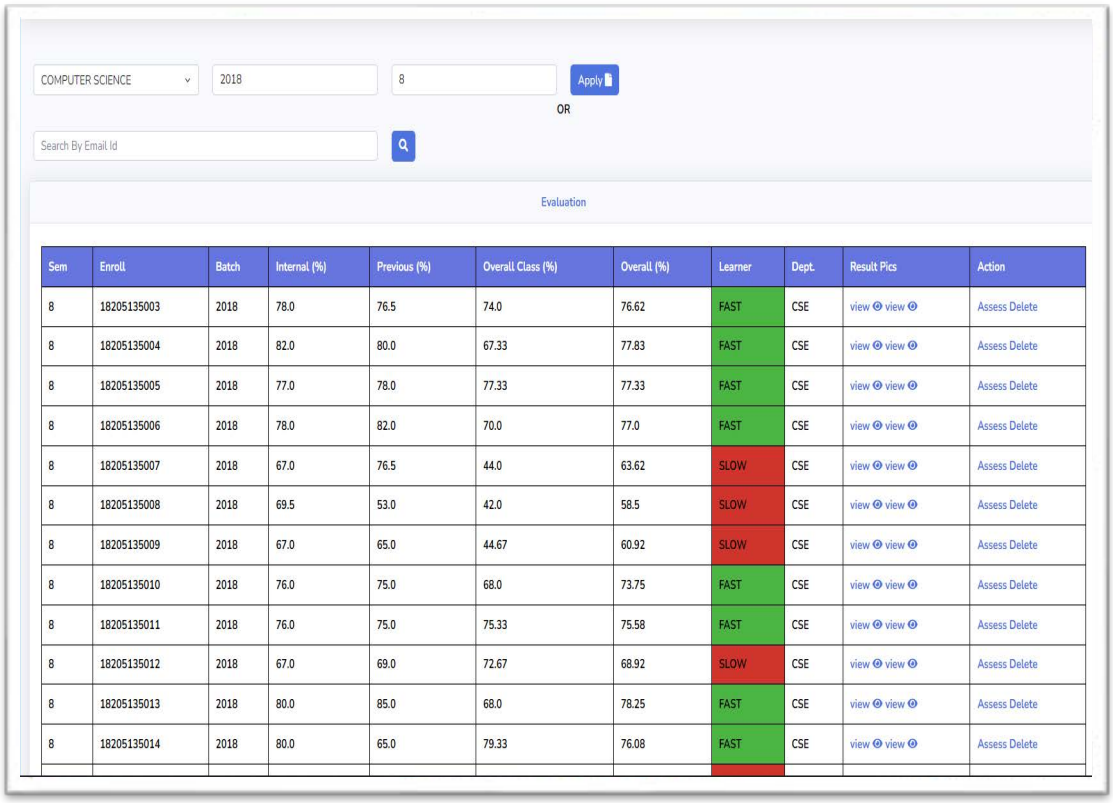


Figure 4.15 HOD Dashboard

The HOD can also be teaching a subject to any semester and hence is expected to evaluate the students based on their class performance. The faculty is prompted to select the department and batch to get the list of students to be evaluated. The faculty can select the assess tab or even delete/edit the assessment previously made.



Sem	Enroll	Batch	Internal (%)	Previous (%)	Overall Class (%)	Overall (%)	Learner	Dept.	Result Pics	Action
8	18205135003	2018	78.0	76.5	74.0	76.62	FAST	CSE	view view	Assess Delete
8	18205135004	2018	82.0	80.0	67.33	77.83	FAST	CSE	view view	Assess Delete
8	18205135005	2018	77.0	78.0	77.33	77.33	FAST	CSE	view view	Assess Delete
8	18205135006	2018	78.0	82.0	70.0	77.0	FAST	CSE	view view	Assess Delete
8	18205135007	2018	67.0	76.5	44.0	63.62	SLOW	CSE	view view	Assess Delete
8	18205135008	2018	69.5	53.0	42.0	58.5	SLOW	CSE	view view	Assess Delete
8	18205135009	2018	67.0	65.0	44.67	60.92	SLOW	CSE	view view	Assess Delete
8	18205135010	2018	76.0	75.0	68.0	73.75	FAST	CSE	view view	Assess Delete
8	18205135011	2018	76.0	75.0	75.33	75.58	FAST	CSE	view view	Assess Delete
8	18205135012	2018	67.0	69.0	72.67	68.92	SLOW	CSE	view view	Assess Delete
8	18205135013	2018	80.0	85.0	68.0	78.25	FAST	CSE	view view	Assess Delete
8	18205135014	2018	80.0	65.0	79.33	76.08	FAST	CSE	view view	Assess Delete

Figure 4.16 List of students to be evaluated

One the student is selected for the assessment the name along with parent's name and contact is displayed on the dashboard and the subject tab appears where the faculty chooses the subject to be evaluated.

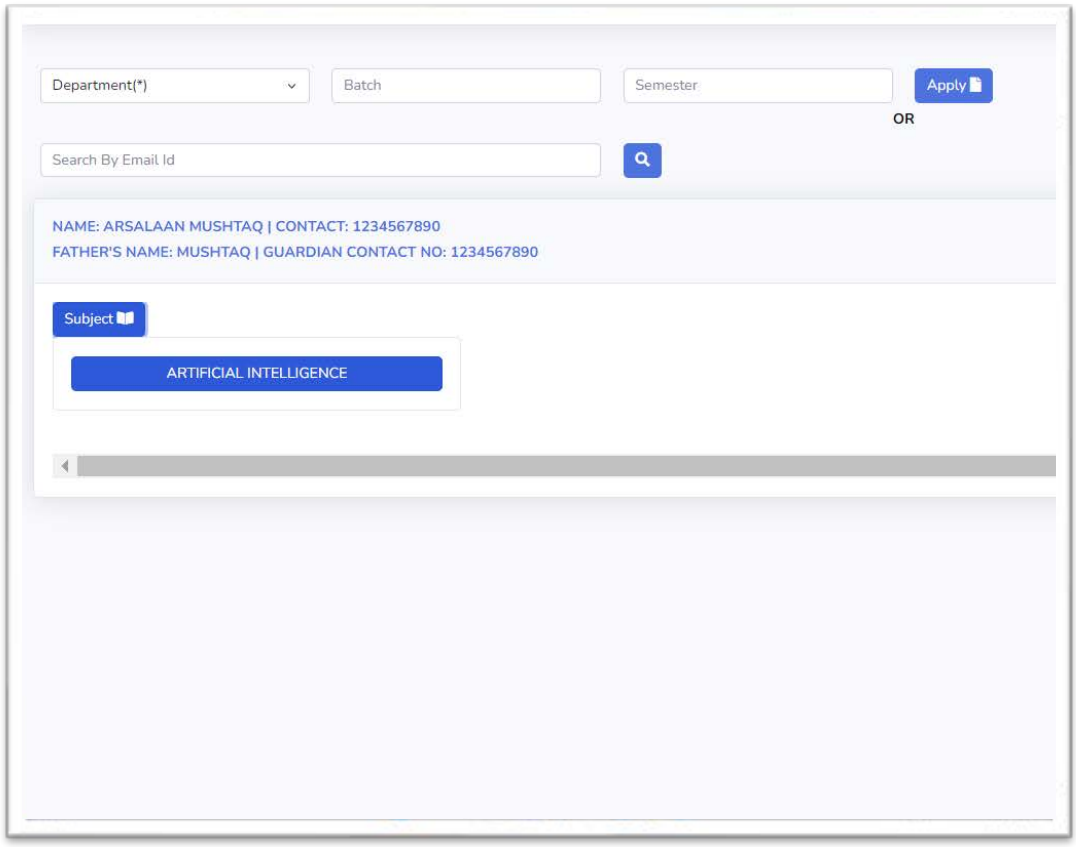


Figure 4.17 Selection of subject to be evaluated

Now the evaluation is done, and it is done by assigning a pointer out of 10 to each student based on their behavior, attentiveness in the class and their attendance in the subject.

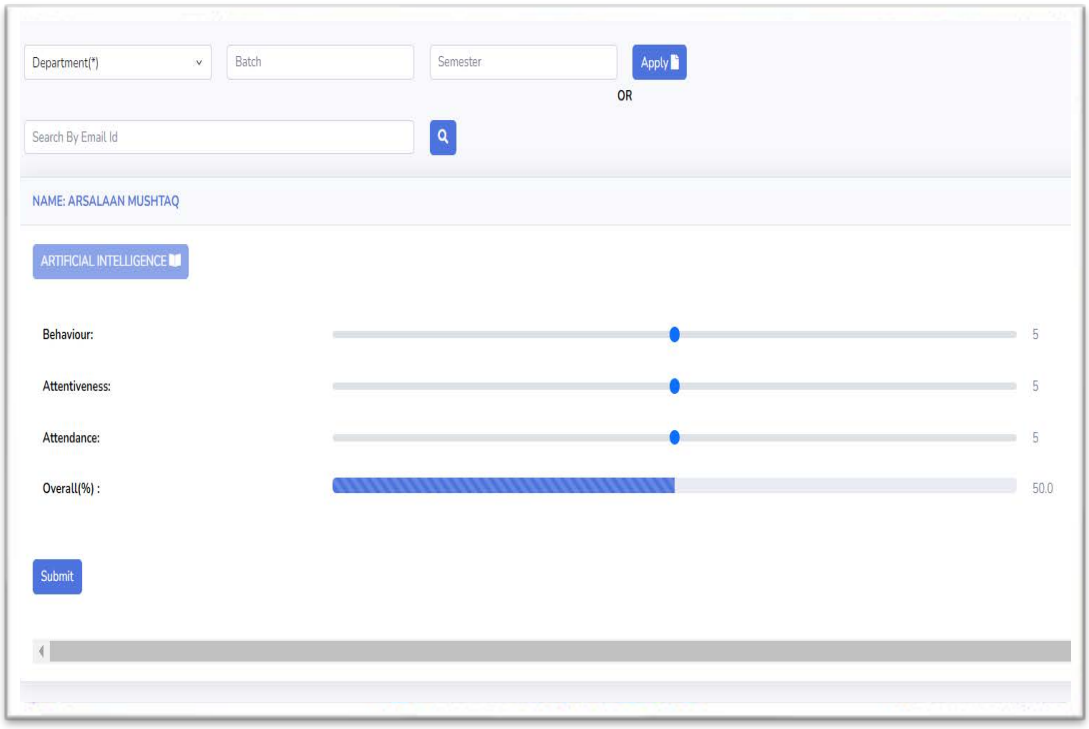


Figure 4.18 Assessment of a subject

The faculty can also view the semester data in the sorted sem data tab where he/she can select the department, batch, semester and the type of students he/she wishes to see. The list can be of slow learners, fast learners or the students whose evaluation of subjects is pending and the students who have not uploaded their marks yet. This gives the faculty precise control over the students.

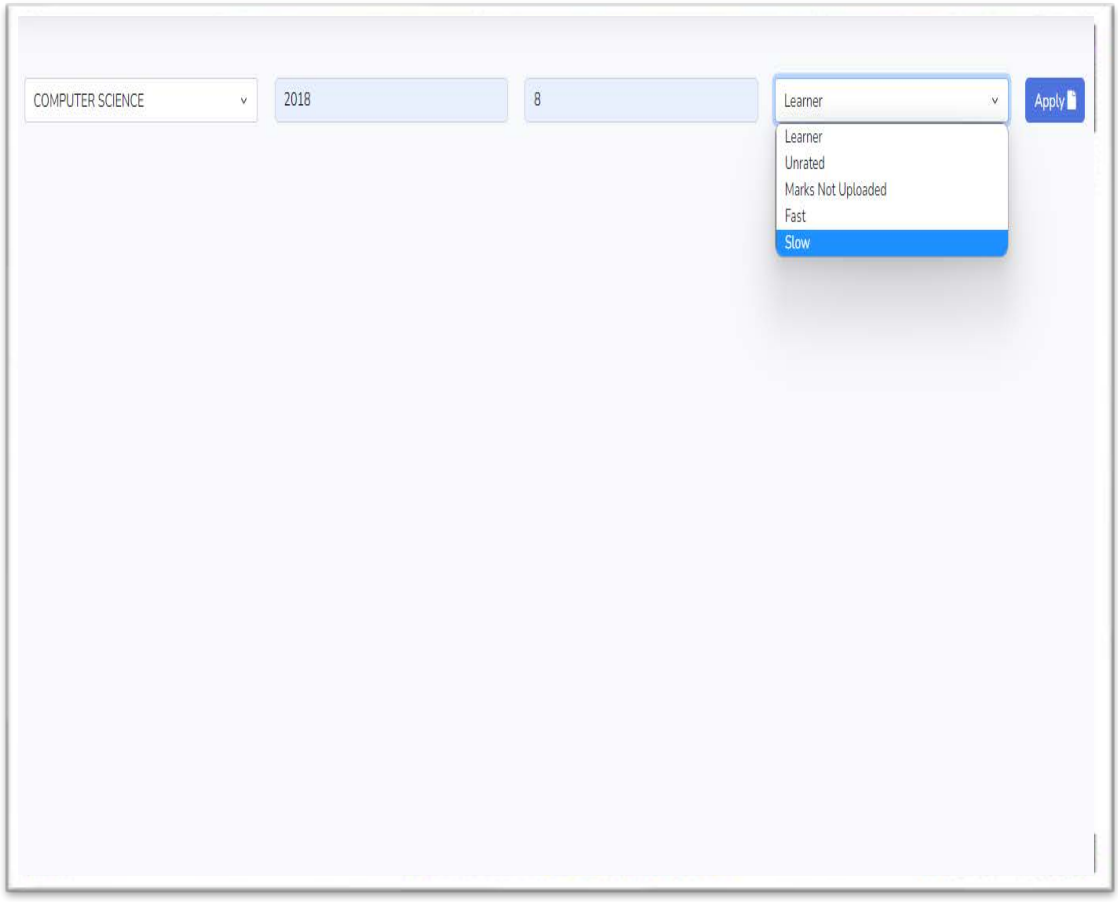
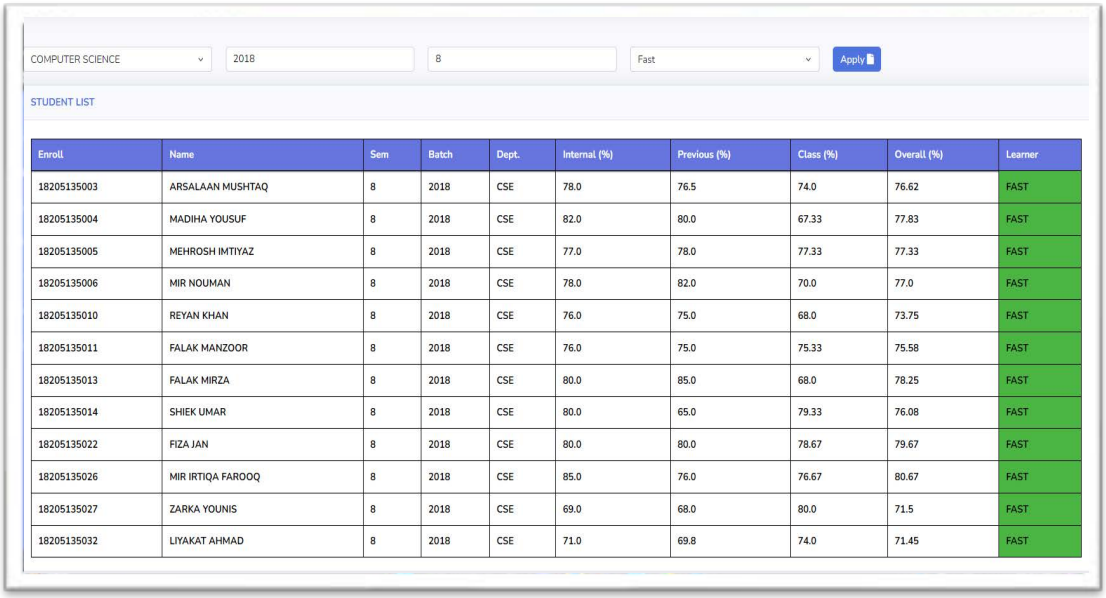


Figure 4.19 Sorted Semester Data

The list of fast learners from the CSE department batch 2018, semester 8th is shown as:

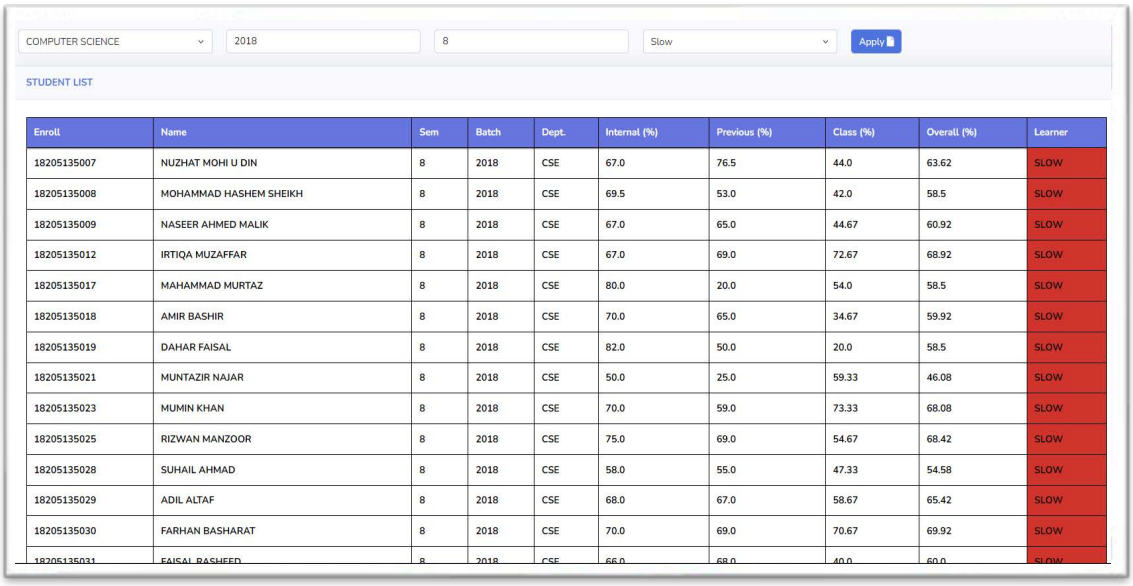


COMPUTER SCIENCE | 2018 | 8 | Fast | Apply

STUDENT LIST

Enroll	Name	Sem	Batch	Dept.	Internal (%)	Previous (%)	Class (%)	Overall (%)	Learner
18205135003	ARSALAAN MUSHTAQ	8	2018	CSE	78.0	76.5	74.0	76.62	FAST
18205135004	MADIHA YOUSUF	8	2018	CSE	82.0	80.0	67.33	77.83	FAST
18205135005	MEHROSH IMTIYAZ	8	2018	CSE	77.0	78.0	77.33	77.33	FAST
18205135006	MIR NOUMAN	8	2018	CSE	78.0	82.0	70.0	77.0	FAST
18205135010	REYAN KHAN	8	2018	CSE	76.0	75.0	68.0	73.75	FAST
18205135011	FALAK MANZOOR	8	2018	CSE	76.0	75.0	75.33	75.58	FAST
18205135013	FALAK MIRZA	8	2018	CSE	80.0	85.0	68.0	78.25	FAST
18205135014	SHIEK UMAR	8	2018	CSE	80.0	65.0	79.33	76.08	FAST
18205135022	FIZA JAN	8	2018	CSE	80.0	80.0	78.67	79.67	FAST
18205135026	MIR IRTIQA FAROOQ	8	2018	CSE	85.0	76.0	76.67	80.67	FAST
18205135027	ZARKA YOUNIS	8	2018	CSE	69.0	68.0	80.0	71.5	FAST
18205135032	LIYAKAT AHMAD	8	2018	CSE	71.0	69.8	74.0	71.45	FAST

Figure 4.20 List of Fast Learners



COMPUTER SCIENCE | 2018 | 8 | Slow | Apply

STUDENT LIST

Enroll	Name	Sem	Batch	Dept.	Internal (%)	Previous (%)	Class (%)	Overall (%)	Learner
18205135007	NUZHAT MOHI U DIN	8	2018	CSE	67.0	76.5	44.0	63.62	SLOW
18205135008	MOHAMMAD HASHEM SHEIKH	8	2018	CSE	69.5	53.0	42.0	58.5	SLOW
18205135009	NASEER AHMED MALIK	8	2018	CSE	67.0	65.0	44.67	60.92	SLOW
18205135012	IRTIQA MUZAFFAR	8	2018	CSE	67.0	69.0	72.67	68.92	SLOW
18205135017	MAHAMMAD MURTAZ	8	2018	CSE	80.0	20.0	54.0	58.5	SLOW
18205135018	AMIR BASHIR	8	2018	CSE	70.0	65.0	34.67	59.92	SLOW
18205135019	DAHAR FAISAL	8	2018	CSE	82.0	50.0	20.0	58.5	SLOW
18205135021	MUNTAZIR NAJAR	8	2018	CSE	50.0	25.0	59.33	46.08	SLOW
18205135023	MUMIN KHAN	8	2018	CSE	70.0	59.0	73.33	68.08	SLOW
18205135025	RIZWAN MANZOOR	8	2018	CSE	75.0	69.0	54.67	68.42	SLOW
18205135028	SUHAIL AHMAD	8	2018	CSE	58.0	55.0	47.33	54.58	SLOW
18205135029	ADIL ALTAF	8	2018	CSE	68.0	67.0	58.67	65.42	SLOW
18205135030	FARHAN BASHARAT	8	2018	CSE	70.0	69.0	70.67	69.92	SLOW
18205135031	FAISAL RASHEED	8	2018	CSE	66.0	68.0	40.0	60.0	SLOW

Figure 4.21 List of Slow learners

The list of students who are not evaluated by the faculty is given as “Unrated”:

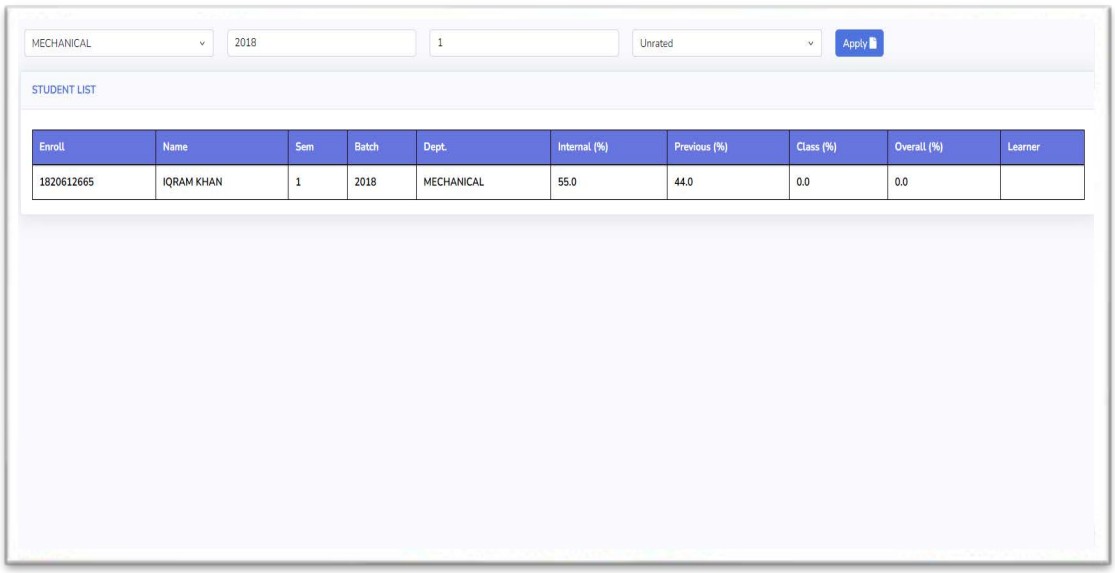


Figure 4.22 Unrated students list

The list of students who have not uploaded their results is given as:

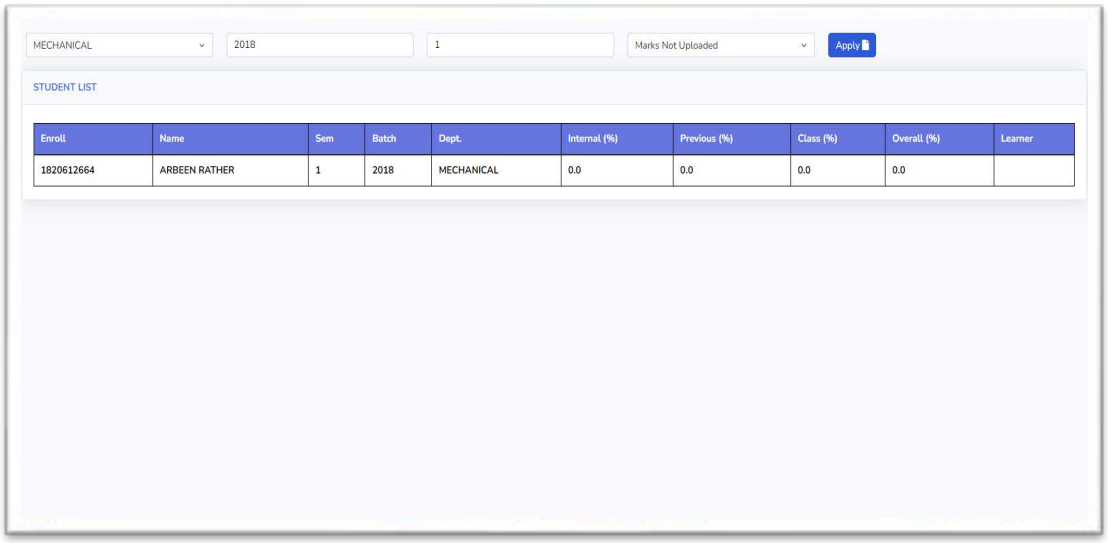


Figure 4.23 List of students who have not uploaded their marks

The head of the department can also view the faculty list of the department by selecting the department. Along with the name of the faculty their email id is displayed.

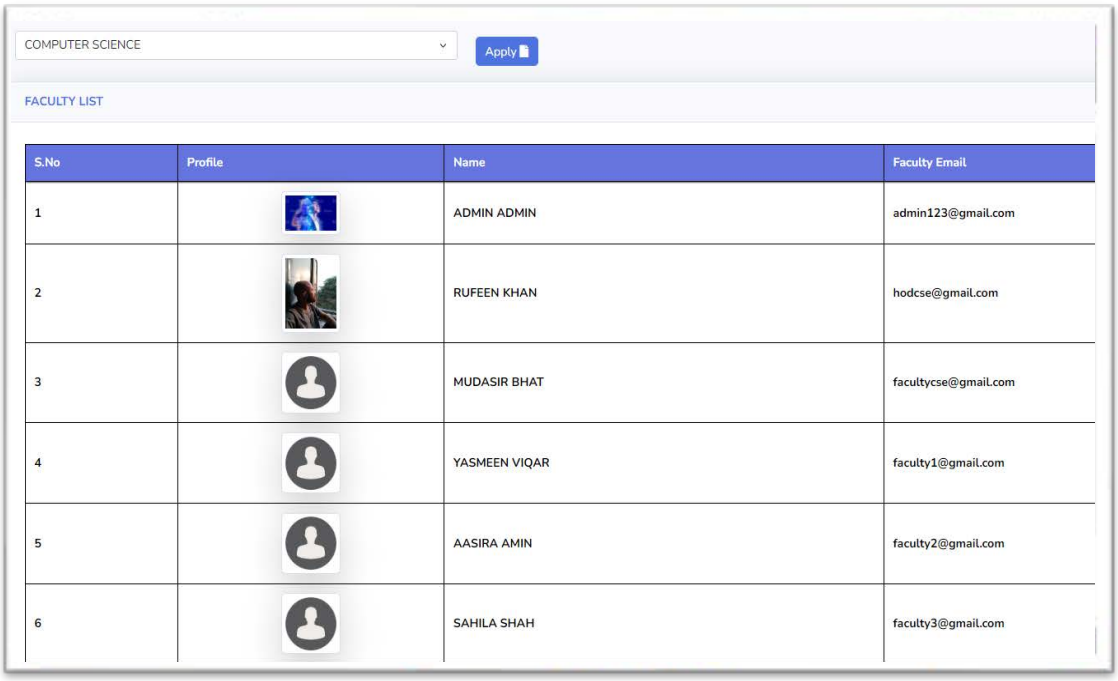


Figure 4.24 Faculty List

The head of the department has an important role in allotting the faculty their roles. The HOD can validate a faculty by allotting them the role of faculty member and can also make the faculty a mentor, coordinator or HOD.

Profile	Name	Email	is HEAD OF DEPARTMENT	is COORDINATOR	is MENTOR	is FACULTY
	ADMIN ADMIN	admin123@gmail.com	YES	YES	YES	YES
	RUFEEN KHAN	hodcse@gmail.com	YES	NO	NO	YES
	MUDASIR BHAT	facultycse@gmail.com	NO	YES	NO	YES
	YASMEEN VIQAR	faculty1@gmail.com	YES	NO	NO	YES
	AASIRA AMIN	faculty2@gmail.com	NO	YES	NO	YES
	SAHILA SHAH	faculty3@gmail.com	NO	NO	YES	YES
	MUDASIR MATTOO	faculty4@gmail.com	NO	NO	NO	YES
	BEHJA RIYAZ	faculty5@gmail.com	NO	NO	NO	YES

Figure 4.25 Allotment of role to faculty

Another important function of a HOD is to dynamically set the criteria based on which the students are further classified as slow/ fast learners. The three criteria are class performance (already calculated), internal results and previous year's result. The overall threshold is also set by the HOD

Figure 4.26 Dynamic Setting of Criteria

4.5 FACULTY AS A COORDINATOR:

The faculty as a coordinator has functionalities of allotting subjects, viewing mentors and allotting students to the mentors and posting notifications.

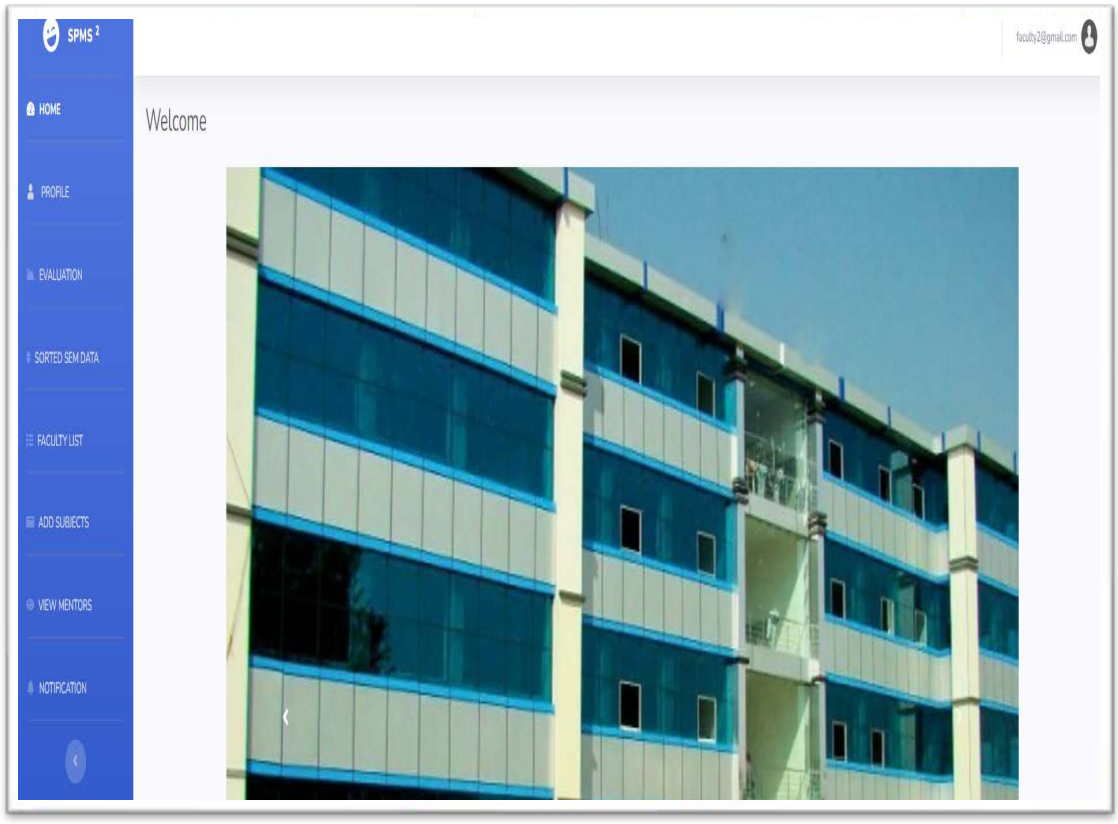


Figure 4.27 Dashboard of coordinator

The subjects can be added by the coordinator by inserting the name of the subject, batch, semester and email address of the faculty teaching that subject.

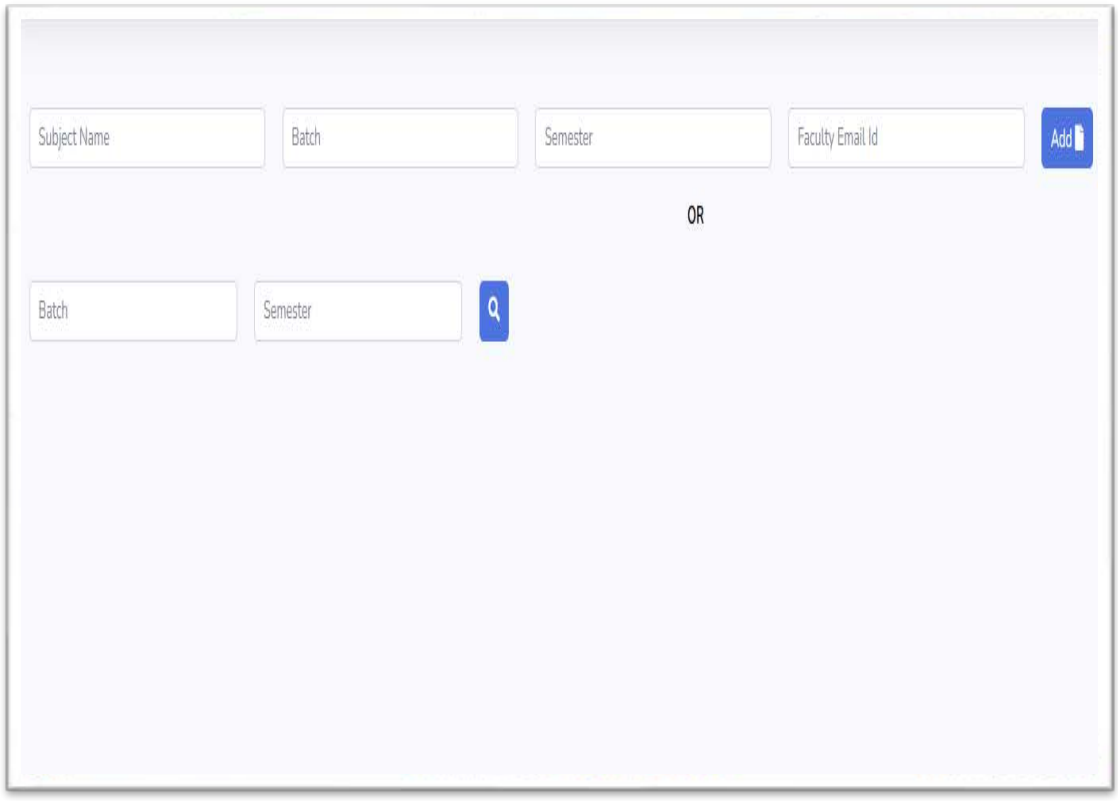


Figure 4.28 Add Subjects to semesters

The coordinator can also view the allotted subjects by providing the batch and semester.

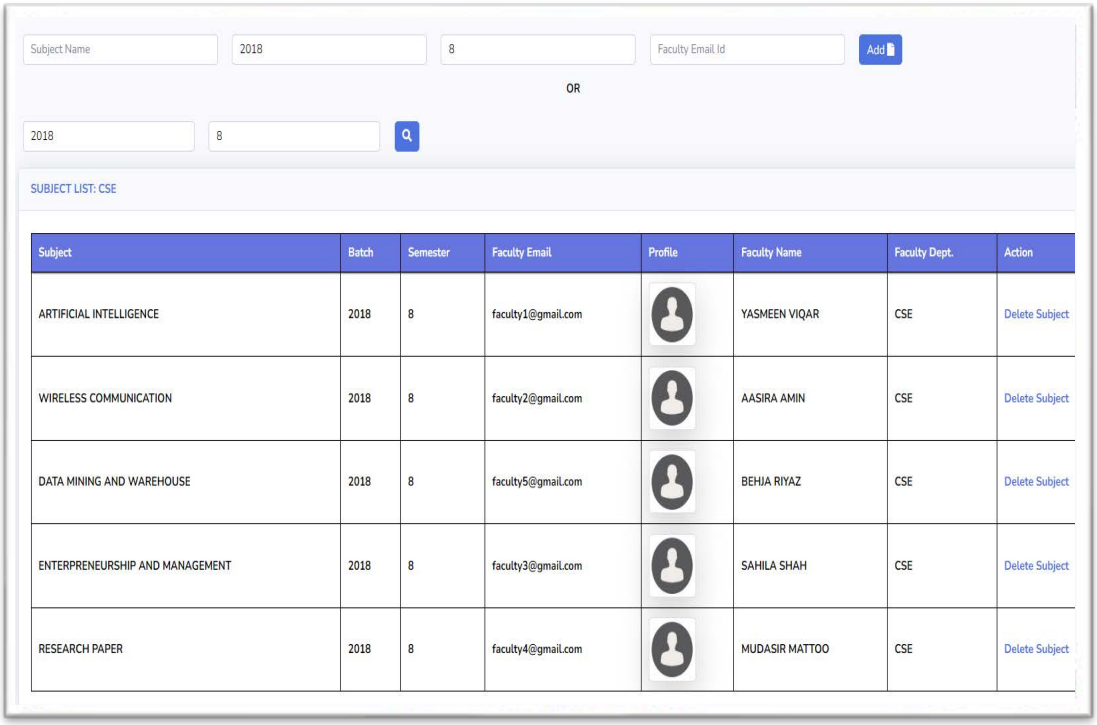


Figure 4.29 List of subjects

The coordinator has an option to view the mentors that were allotted by the head of the department and then the coordinator can allot students to the specified mentor.

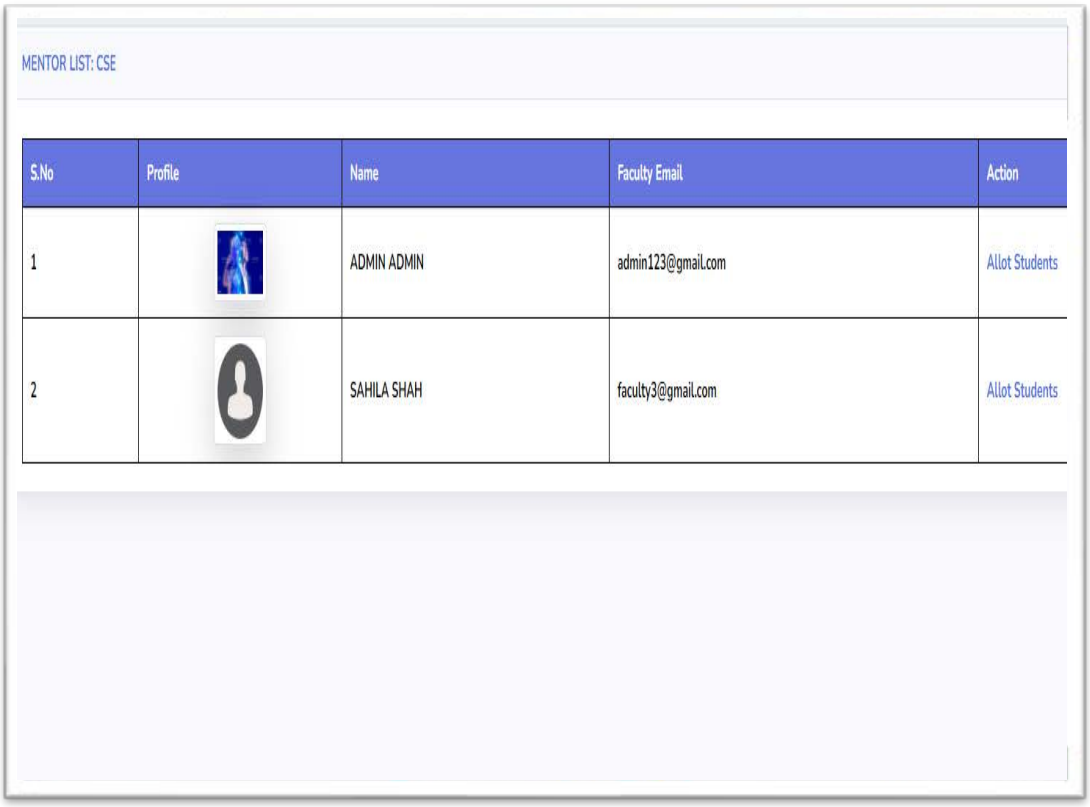


Figure 4.30 List of mentors

The coordinator can allot the students to the mentor by clicking on the mentor and then allotting a selected batch, semester and type of learner to the mentor. Usually, the slow learners are allotted to the mentors.

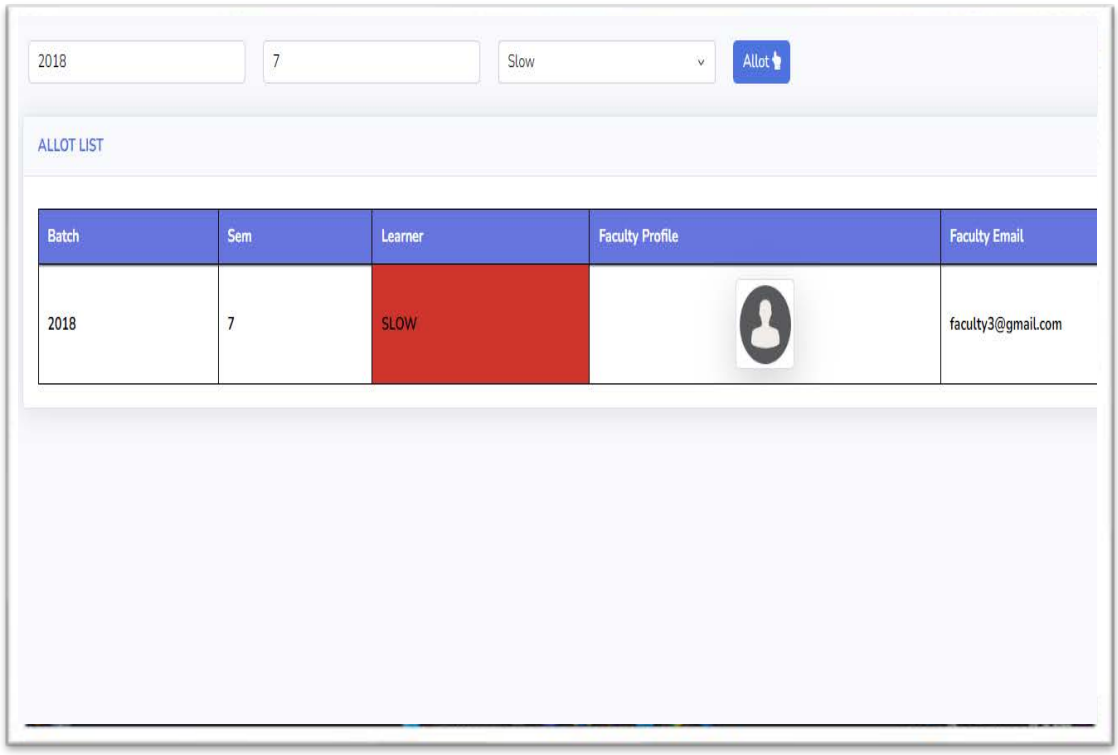


Figure 4.31 Allotting Students to mentors

The coordinator can also post the notifications for each semester by providing the semester and batch as input. This notification can then be viewed by the respected students of that batch and semester. The notifications posted are saved and a list is displayed simultaneously.

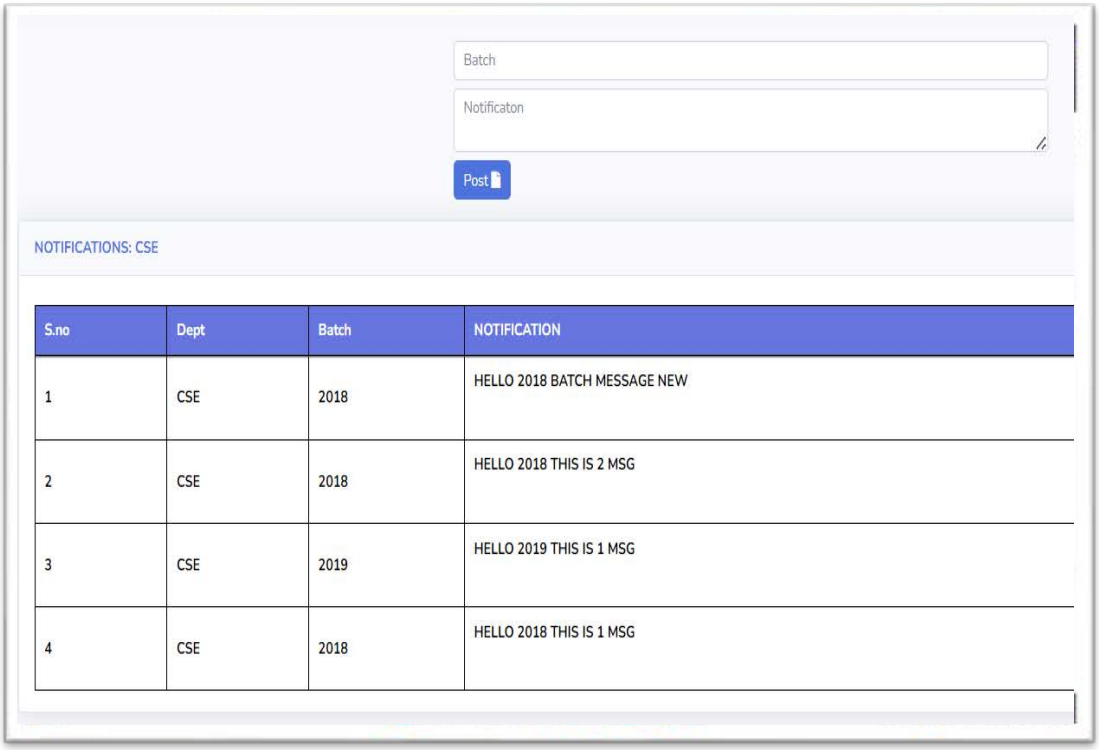


Figure 4.32 Notification Posting

4.6 FACULTY AS MENTOR

The faculty which has been allotted a role of mentor have all the roles same as the faculty, only one additional function is provided to the mentor that is used to contact with the allotted students. The messaging feature of the mentor allows him/her to connect to the students on a personal level and provide information regarding PTM, personal counselling schedules etc. The mentor can select the allotted group of students from drop down list.

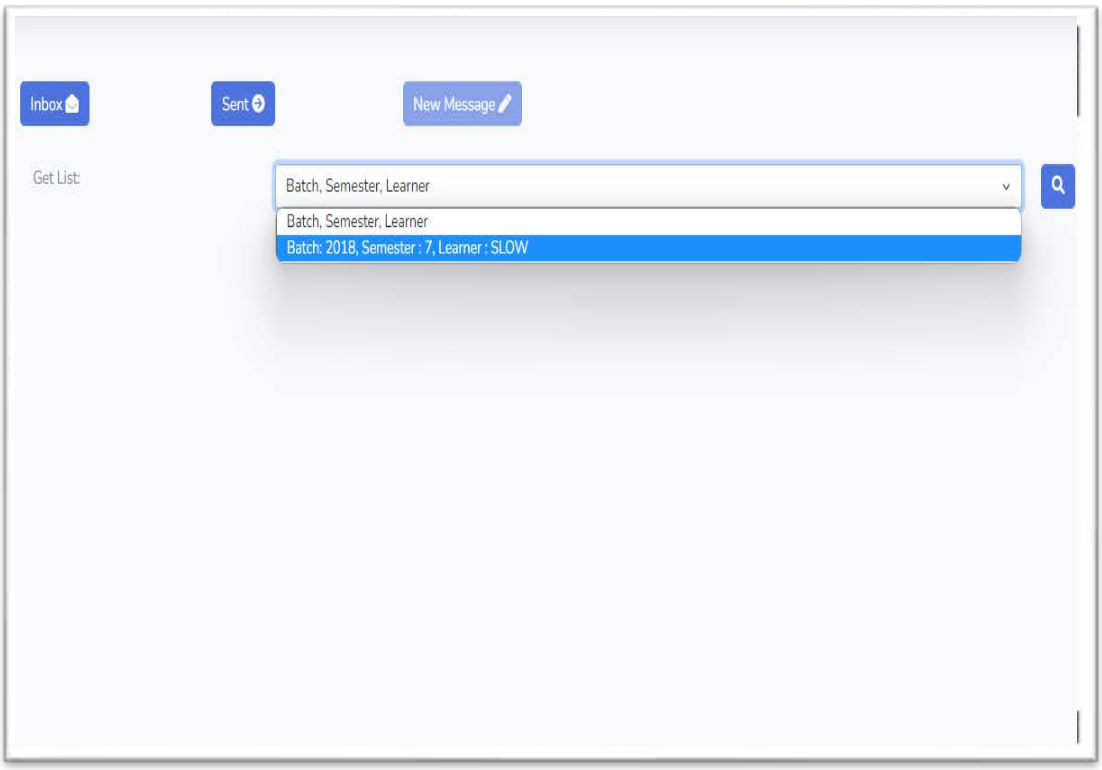


Figure 4.33 Messaging system of mentor

The list of students under the guidance of the mentor is displayed where he/she can communicate one to one with the student. The only condition is that the mentor must initiate the messaging.

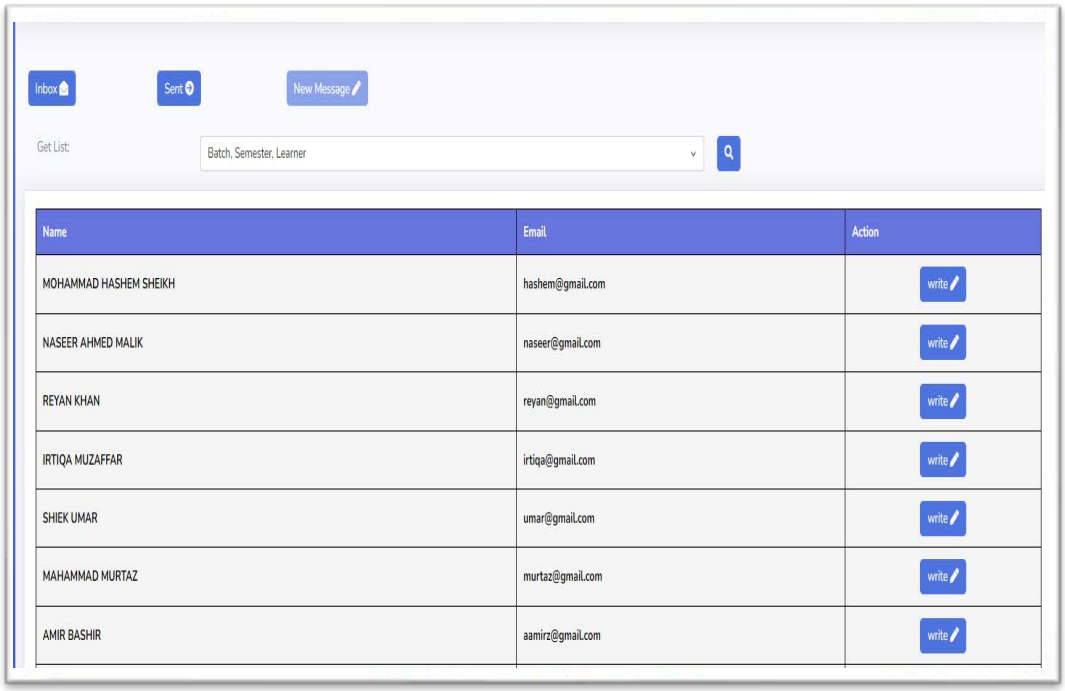


Figure 4.34 List of allotted students

The mentor writes the message and sends it to the student whose email address is already mentioned as a receiver. The student can reply to the message which will be notified to the mentor as an alert on the messages icon and then the mentor can reply to the message.

Figure 4.35 Writing Message

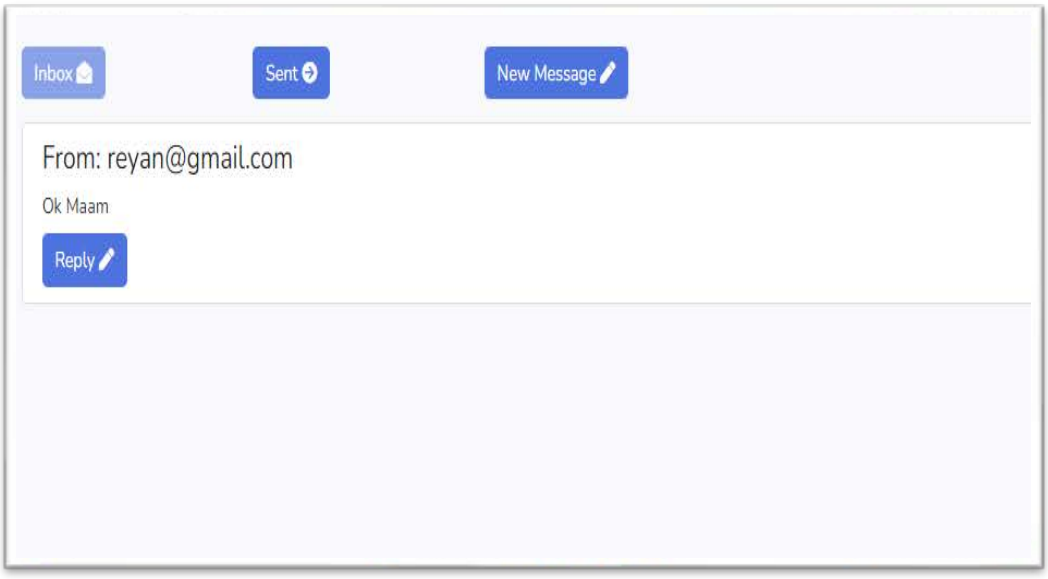


Figure 4.36 Viewing and Replying to the message

4.7 FORGETTING THE PASSWORD:

If the faculty or the student forgets the password, then they can recover their account by clicking on the forgot password tab and it redirects them to enter their verified email and proceed to acknowledge their account. Once they do so, then a verification link is sent to their email address, and they can then change their password.

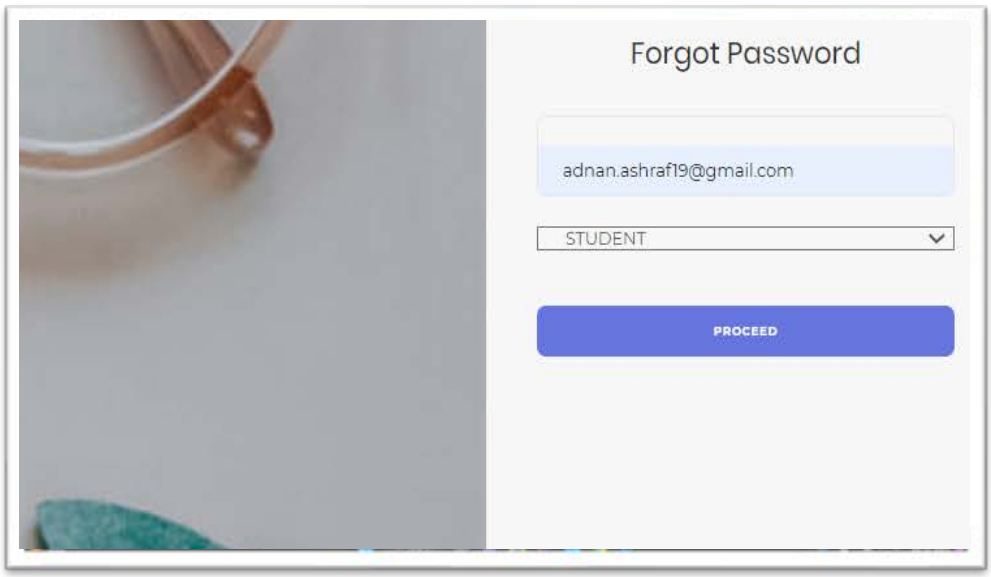


Figure 4.37 Forgot Password

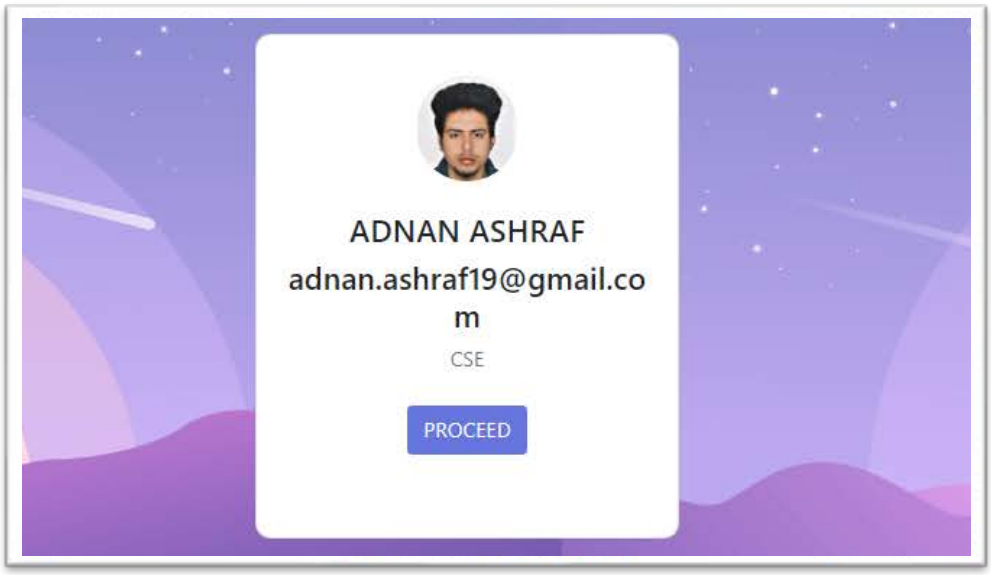


Figure 4.38 Acknowledge Account

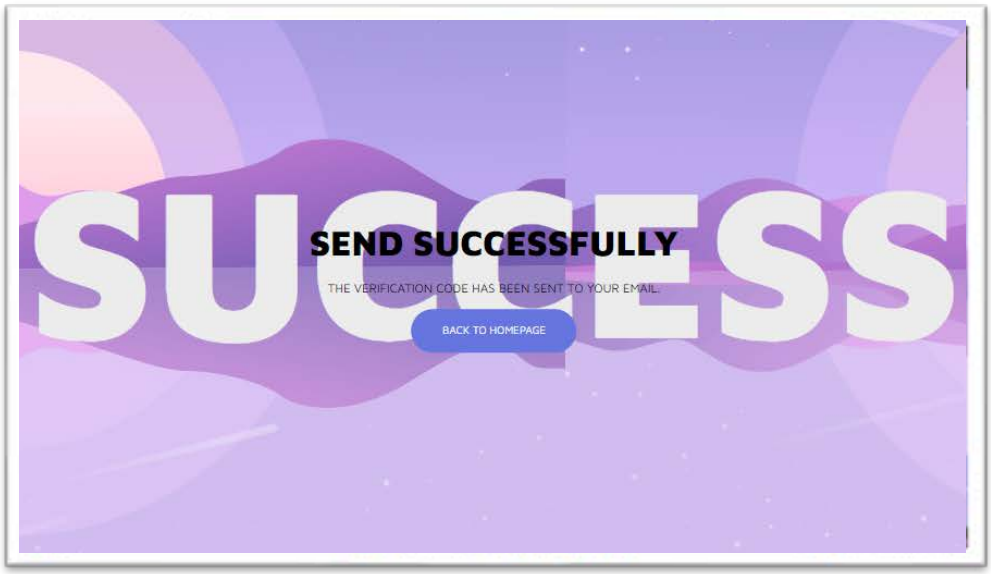


Figure 4.39 Verification code sent

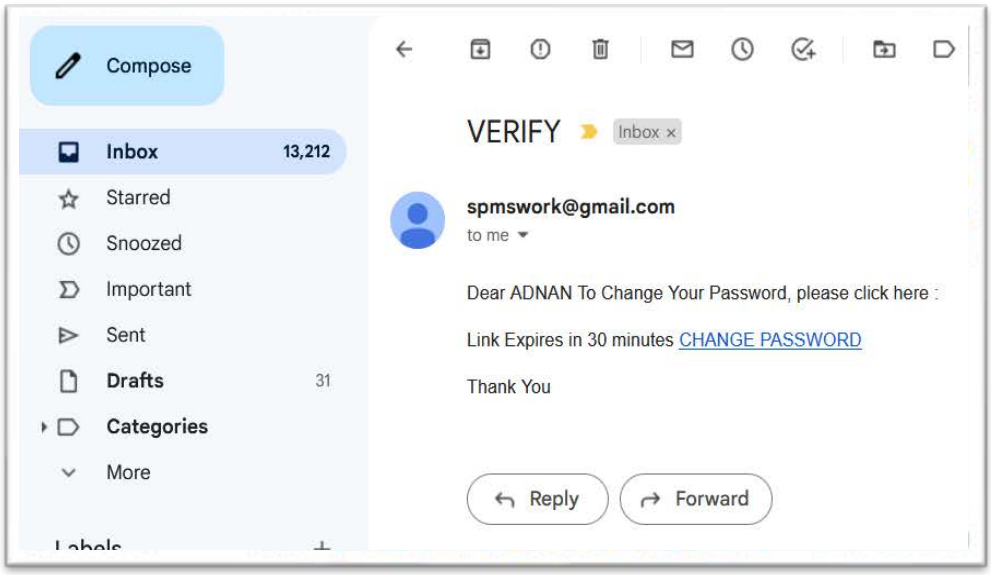


Figure 4.40 Password Changing link sent

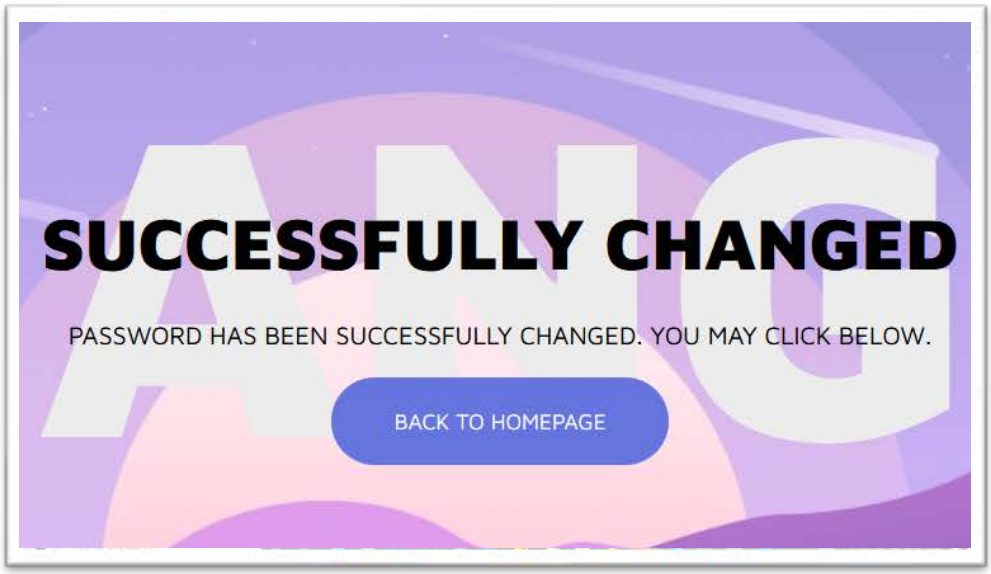


Figure 4.41 Password Changed Successfully

4.8 MAIN PROGRAM PSEUDO CODE:

Web Configuration:

SpmsConfiguration.java :-

```
package com.adnan.icode.fun.spms.config;
@Configuration
@EnableWebMvc
@EnableTransactionManagement
@ComponentScan("com.adnan.icode.fun.spms")
@PropertySource({"classpath:persistence-mysql.properties"})
public class SpmsConfiguration implements WebMvcConfigurer {
@Autowired
private Environment env;
@Bean
public InternalResourceViewResolver viewResolver() {
InternalResourceViewResolver viewResolver = new InternalResourceViewResolver();
viewResolver.setPrefix("/WEB-INF/view/");
viewResolver.setSuffix(".jsp");
return viewResolver;
}
@Bean
public DataSource myDataSource() {
ComboPooledDataSource myDataSource = new ComboPooledDataSource();
try {
myDataSource.setDriverClass(env.getProperty("jdbc.driver"));
} catch (PropertyVetoException e) {
e.printStackTrace();
}
```

```
myDataSource.setJdbcUrl(env.getProperty("jdbc.url"));
myDataSource.setUser(env.getProperty("jdbc.user"));
myDataSource.setPassword(env.getProperty("jdbc.password"));
myDataSource.setInitialPoolSize(Integer.parseInt(env.getProperty("connection.pool.initialPoolSize")));
myDataSource.setMinPoolSize(Integer.parseInt(env.getProperty("connection.pool.minPoolSize")));
myDataSource.setMaxPoolSize(Integer.parseInt(env.getProperty("connection.pool.maxPoolSize")));
myDataSource.setMaxIdleTime(Integer.parseInt(env.getProperty("connection.pool.maxIdleTime")));
return myDataSource;
}
private Properties getHibernateProperties() {
Properties props = new Properties();
props.setProperty("hibernate.dialect", env.getProperty("hibernate.dialect"));
props.setProperty("hibernate.show_sql", env.getProperty("hibernate.show_sql"));
return props;
}
@Bean
public  LocalSessionFactoryBean sessionFactory() {
LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();
sessionFactory.setDataSource(myDataSource());
sessionFactory.setPackagesToScan(env.getProperty("hibernate.packagesToScan"));
sessionFactory.setHibernateProperties(getHibernateProperties());
return sessionFactory;
}
@Bean
@Autowired
```

```
public HibernateTransactionManager transactionManager(SessionFactory
sessionFactory) {
HibernateTransactionManager txManager = new HibernateTransactionManager();
txManager.setSessionFactory(sessionFactory);
return txManager;
}
@Override
public void addResourceHandlers(ResourceHandlerRegistry registry) {
registry
.addResourceHandler("/resources/**")
.addResourceLocations("/resources/");
}
@Bean
public CommonsMultipartResolver multipartResolver() {
CommonsMultipartResolver multipartResolver = new CommonsMultipartResolver();
return multipartResolver;
}
}
```

SpmsDispatcherServlet.java :-

```
package com.adnan.icode.fun.spms.config;
public class SpmsDispatcherServlet extends
AbstractAnnotationConfigDispatcherServletInitializer {
@Override
protected Class<?>[] getRootConfigClasses() {
return null;
}
@Override
protected Class<?>[] getServletConfigClasses() {
return new Class[] {SpmsConfiguration.class};
```

```
}
@Override
protected String[] getServletMappings() {
return new String [] {"/"};
}
@Override
protected void customizeRegistration(Dynamic registration) {
boolean done = registration.setInitParameter("throwExceptionIfNoHandlerFound",
"true");
}}
```

Security Configuration:

SpmsSecurityStudentConfiguration.java :-

```
package com.adnan.icode.fun.spms.config;
@Configuration
@EnableWebSecurity
@Order(1)
public class SpmsSecurityStudentConfiguration extends WebSecurityConfigurerAdapter
{
@Autowired
private StudentLoginService studentLoginService;
@Autowired
private CustomStudentSuccessHandler customStudentSuccessHandler;
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
auth.authenticationProvider(studentAuthenticationProvider());
}
@Bean
public BCryptPasswordEncoder bCryptPasswordEncoder() {
```

```
return new BCryptPasswordEncoder();
}
@Bean
public DaoAuthenticationProvider studentAuthenticationProvider() {
DaoAuthenticationProvider auth = new DaoAuthenticationProvider();
auth.setUserDetailsService((UserDetailsService)studentLoginService);
auth.setPasswordEncoder(bCryptPasswordEncoder());
return auth;
}
@Override
protected void configure(HttpSecurity http) throws Exception {
http.csrf().disable();
http.antMatcher("/student/**").authorizeRequests()
.antMatchers("/student/**").hasRole("STUDENT")
.and()
.formLogin().loginPage("/student/studentLogin")
.loginProcessingUrl("/student/authenticateTheStudent")
.successHandler(customStudentSuccessHandler)
.usernameParameter("email").passwordParameter("password")
.permitAll()
.and()
.rememberMe().userDetailsService(studentLoginService)
.alwaysRemember(true).rememberMeCookieName("s_user")
.and()
.logout().logoutUrl("/student/logout").permitAll()
.and().exceptionHandling()
.accessDeniedPage("/faculty/facultyAccessDenied");
}}
```

SpmsSecurityFacultyConfiguration.java :-

```
package com.adnan.icode.fun.spms.config;
@Configuration
@EnableWebSecurity
@Order(2)
public class SpmsSecurityFacultyConfiguration extends WebSecurityConfigurerAdapter
{
@Autowired
private FacultyLoginService facultyLoginService;
@Autowired
private CustomFacultySuccessHandler customFacultySuccessHandler;
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
auth.authenticationProvider(facultyAuthenticationProvider());
}
@Bean
public BCryptPasswordEncoder bCryptPasswordEncoder() {
return new BCryptPasswordEncoder();
}
@Bean
public DaoAuthenticationProvider facultyAuthenticationProvider() {
DaoAuthenticationProvider auth = new DaoAuthenticationProvider();
auth.setUserDetailsService(facultyLoginService);
auth.setPasswordEncoder(bCryptPasswordEncoder());
return auth;
}
@Override
protected void configure(HttpSecurity http) throws Exception {
http.csrf().disable();
http.antMatcher("/faculty/**").authorizeRequests()
```

```
.antMatchers("/faculty/**").hasRole("FACULTY")
.and()
.formLogin().loginPage("/faculty/facultyLogin")
.loginProcessingUrl("/faculty/authenticateTheFaculty")
.successHandler(customFacultySuccessHandler)
.usernameParameter("email").passwordParameter("password")
.permitAll()
.and()
.rememberMe().userDetailsService(facultyLoginService)
.alwaysRemember(true).rememberMeCookieName("f_user")
.and()
.logout().logoutUrl("/faculty/logout").permitAll()
.and()
.exceptionHandling()
.accessDeniedPage("/student/studentAccessDenied");
}
}
```

SpmsSecurityWebApplicationInitializer.java :-

```
package com.adnan.icode.fun.spms.config;
public class SpmsSecurityWebApplicationInitializer extends
AbstractSecurityWebApplicationInitializer {                    }
```

Main Controllers:

StartController.java :-

```
package com.adnan.icode.fun.spms.controllers;
@Controller
@RequestMapping("/start")
public class StartController {
```

```
@Autowired
private CookieCheckerRedirector checkerRedirector;
@Autowired
private SpmsService spmsService;
@GetMapping("/homePage")
public  String homePage(HttpServletRequest request, HttpServletResponse response) {
String redirect = checkerRedirector.redirect(request, response);
if (redirect != null) {
return redirect;
}
return "general/welcome";
}
@GetMapping("/about")
public  String about(HttpServletRequest request, HttpServletResponse response,
Model theModel) {
String redirect = checkerRedirector.redirect(request, response);
if (redirect != null) {
return redirect;
}
List<Integer> picNumber = new ArrayList<Integer>();
for(int i = 1; i<= 17; i++) {
picNumber.add(i);
}
theModel.addAttribute("picNumber", picNumber);
return "general/about";
}
@GetMapping("/studentConfirmation")
public  String studentConfirmation(@RequestParam("token") String token,
Model theModel) {
Date currentDate = null;
```

```
Date expiryDate = null;
StudentVerificationToken studentToken = spmsService.findStudentToken(token);
if(studentToken != null ) {
currentDate = new Date();
expiryDate = studentToken.getExpiryDate();
if(expiryDate.getTime() - currentDate.getTime() <= 0) {
return "information/invalid-verification";
}else if(expiryDate.getTime() - currentDate.getTime() > 0) {
System.out.println("not expired");
String studentEmail = studentToken.getStudent().getEmail();
Student tempStudent = spmsService.loadStudentByEmail(studentEmail);
tempStudent.setEnabled(true);
spmsService.updateStudent(tempStudent);
spmsService.deleteAllStudentTokens(tempStudent);
theModel.addAttribute("email", studentEmail);
return "information/account-verified";
}
}
return "information/invalid-verification";
}
@GetMapping("/facultyConfirmation")
public  String facultyConfirmation(@RequestParam("token") String token,
Model theModel) {
Date currentDate = null;
Date expiryDate = null;
FacultyVerificationToken facultyToken = spmsService.findFacultyToken(token);
if(facultyToken != null ) {
currentDate = new Date();
expiryDate = facultyToken.getExpiryDate();
if(expiryDate.getTime() - currentDate.getTime() <= 0) {
```

```
return "information/invalid-verification";
}else if(expiryDate.getTime() - currentDate.getTime() > 0) {
String facultyEmail = facultyToken.getFaculty().getEmail();
Faculty tempFaculty = spmsService.loadFacultyByEmail(facultyEmail);
spmsService.deleteAllFacultyTokens(tempFaculty);
theModel.addAttribute("email", facultyEmail);
return "information/account-verified";
}
}
return "information/invalid-verification";
}
@GetMapping("/renewEmail")
public  String renewStudentEmailLink(
@RequestParam("email") String email,
@RequestParam("account") String account,
HttpServletRequest request) {
if(account.equals("student")) {
Student theStudent = spmsService.loadStudentByEmail(email);
if(theStudent != null  && !theStudent.getEnabled()){
spmsService.newEmailToken(email, account, request);
return "information/email-link-renewed";
}
}
if(account.equals("faculty")) {
Faculty theFaculty = spmsService.loadFacultyByEmail(email);
if(theFaculty != null  && !theFaculty.getEnabled()){
spmsService.newEmailToken(email, account, request);
return "information/email-link-renewed";
}
}
```

```
return "information/invalid-verification";
}
}
```

StudentController.java :-

```
package com.adnan.icode.fun.spms.controllers;
@Controller
@RequestMapping("/student")
public class StudentController {
@Autowired
private CookieCheckerRedirector checkerRedirector;
@Autowired
private SpmsService spmsService;
@InitBinder
public void initBinder(WebDataBinder dataBinder) {
StringTrimmerEditor trimmerEditor = new StringTrimmerEditor(true);
dataBinder.registerCustomEditor(String.class, trimmerEditor);
}
@GetMapping("/studentLogin")
public String studentLogin(HttpServletRequest request, HttpServletResponse response) {
String redirect = checkerRedirector.redirect(request, response);
if (redirect != null) {
return redirect;
}
return "student/student-login";
}
@GetMapping("/studentAccessDenied")
public String accessDenied() {
return "student/student-access";
}
```

```
@GetMapping("/studentPage")
public  String studentPage(Model theModel) {
Student currentStudent = new Student();
List<StudentSemData> semData = new  ArrayList<StudentSemData>();
PercentageController pController = new PercentageController();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
semData = spmsService.loadStudentSemData(currentEmail);
pController = spmsService.loadPercentageControllerByDept(currentStudent.getDept());
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
BasicEvalStatModel stat = new BasicEvalStatModel();
int internalStat, previousStat, classPerStat, evalSem;
float overallStat;
String internalStatMsg, previousStatMsg, classPerStatMsg;
if(semData != null) {
StudentSemData tempSemData = semData.get(semData.size()-1);
evalSem = tempSemData.getSemester();
if(tempSemData.getOverallInternalAssessment() == 0.0 ) {
internalStat = 0;
internalStatMsg = "incomplete";
```

```
}else {
internalStat = 100;
internalStatMsg = "completed";
}
if(tempSemData.getOverallPreviousAssessment() == 0.0 ) {
previousStat = 0;
previousStatMsg = "incomplete";
}else {
previousStat = 100;
previousStatMsg = "completed";
}
if(tempSemData.getOverallSubjectAssessment() == 0.0 ) {
classPerStat = 0;
classPerStatMsg = "incomplete";
}else {
classPerStat = 100;
classPerStatMsg = "completed";
}
OverallAssessmentCal overallCal = new
OverallAssessmentCal(tempSemData,pController);
overallStat = overallCal.overallPercent();
stat.setSemester(evalSem);
stat.setInternalStat(internalStat);
stat.setPreviousStat(previousStat);
stat.setClassPerStat(classPerStat);
stat.setOverallStat(overallStat);
stat.setiMsg(internalStatMsg);
stat.setpMsg(previousStatMsg);
stat.setcMsg(classPerStatMsg);
theModel.addAttribute("stat", stat);
```

```
}
theModel.addAttribute("currentStudent", currentStudent);
return "student/student-home";
}
@GetMapping("/studentProfile")
public  String studentProfile(Model theModel,
@ModelAttribute("message") String message) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
TempStudentProfileModel profileModel = new TempStudentProfileModel();
profileModel.setFirstName(currentStudent.getFirstName());
profileModel.setLastName(currentStudent.getLastName());
profileModel.setUniversityEnroll(String.valueOf(currentStudent.getUniversityEnroll()));
profileModel.setBatch(String.valueOf(currentStudent.getBatch()));
profileModel.setContactNo(currentStudent.getContactNo());
profileModel.setFatherName(currentStudent.getFatherName());
profileModel.setGuardianContactNo(currentStudent.getGuardianContactNo());
theModel.addAttribute("currentStudent", currentStudent);
```

```
theModel.addAttribute("profileModel", profileModel);
return "student/student-profile";
}
@GetMapping("/profileUpdating")
public  String profileUpdating(@Valid
@ModelAttribute("profileModel") TempStudentProfileModel profileModel,
BindingResult theBindingResult,
Model theModel) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("currentStudent", currentStudent);
if(theBindingResult.hasErrors()) {
return "student/student-profile";
}else {
Student tempStudent = spmsService.
loadStudentByEnroll(
Long.parseLong( profileModel.getUniversityEnroll() )
);
```

```
if( ( tempStudent != null ) && ( ( tempStudent.getUniversityEnroll() ) != (
currentStudent.getUniversityEnroll() ) ) ) {
theModel.addAttribute("enrollExists", "enroll already exists");
return "student/student-profile";
}
currentStudent.setFirstName(profileModel.getFirstName().toUpperCase());
currentStudent.setLastName(profileModel.getLastName().toUpperCase());
currentStudent.setUniversityEnroll(Long.parseLong(profileModel.getUniversityEnroll())
);
currentStudent.setBatch(Integer.parseInt(profileModel.getBatch()));
currentStudent.setContactNo(profileModel.getContactNo());
currentStudent.setFatherName(profileModel.getFatherName().toUpperCase());
currentStudent.setGuardianContactNo(profileModel.getGuardianContactNo());
spmsService.updateStudent(currentStudent);
}
return "redirect:/student/studentProfile";
}
@PostMapping("/imageUpdating")
public  String imageUpdating(@RequestParam("image") CommonsMultipartFile file,
HttpServletRequest request,
Model theModel) throws FileOutException {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
if(file.getSize() <= 0 || file.getSize() > 500000 ) {
theModel.addAttribute("message", "must be less than 500kb");
return "redirect:/student/studentProfile";
```

```
}
if(!(file.getContentType().equals("image/jpeg"))){
theModel.addAttribute("message", "file must be jpg");
return "redirect:/student/studentProfile";
}
RandomAlphanumericGenerator generator = new RandomAlphanumericGenerator();
String uniqueString = generator.generate();
String newFileName = uniqueString+file.getOriginalFilename();
String path =
request.getRealPath("/")+"resources"+File.separator+"profile"+File.separator+newFileNa
me;
System.out.println(path);
byte[] image = file.getBytes();
try {
FileOutputStream fos = new FileOutputStream(path);
fos.write(image);
fos.close();
} catch (IOException e) {
throw new FileOutException("Error in uploading File");
}
currentStudent.setProfilePic(newFileName);
spmsService.updateStudent(currentStudent);
return "redirect:/student/studentProfile";
}
@GetMapping("/studentMarks")
public  String studentMarks(Model theModel) {
Student currentStudent = new Student();
List<StudentSemData> semData = new ArrayList<StudentSemData>();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
```

```
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
semData = spmsService.loadStudentSemData(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("assessment", semData);
theModel.addAttribute("currentStudent", currentStudent);
theModel.addAttribute("studentMarksModel", new StudentAddMarksModel());
return "student/student-marks";
}
@GetMapping("/marksUploading")
public  String marksUploading(@Valid
@ModelAttribute("studentMarksModel") StudentAddMarksModel studentMarksModel,
BindingResult theBindingResult,
Model theModel) {
Student currentStudent = new Student();
List<StudentSemData> semData = new  ArrayList<StudentSemData>();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
semData = spmsService.loadStudentSemData(currentEmail);
}
```

```
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("assessment", semData);
theModel.addAttribute("currentStudent", currentStudent);
if(theBindingResult.hasErrors()) {
return "student/student-marks";
}else {
if(semData != null) {
for(StudentSemData tempData: semData) {
if(tempData.getSemester() == Integer.parseInt(studentMarksModel.getSemester())) {
theModel.addAttribute("semExist", "Current Semester Data Already Uploaded");
return "student/student-marks";
}}}
theModel.addAttribute("semester", studentMarksModel.getSemester());
theModel.addAttribute("internalPercent",
studentMarksModel.getOverallInternalAssessment());
theModel.addAttribute("previousPercent",
studentMarksModel.getOverallPreviousAssessment());
return "student/student-marks-images";
}}
@PostMapping("/marksPicsUploading")
public  String marksPicsUploading(
@RequestParam("semester") String semester,
@RequestParam("internalPercent") String internalPercent,
```

```
@RequestParam("previousPercent") String previousPercent,
@RequestParam("internalPic") CommonsMultipartFile internal,
@RequestParam("previousPic") CommonsMultipartFile previous,
Model theModel,
HttpServletRequest request) throws FileOutException {
Student currentStudent = new Student();
List<StudentSemData> semData = new ArrayList<StudentSemData>();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
semData = spmsService.loadStudentSemData(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("assessment", semData);
theModel.addAttribute("currentStudent", currentStudent);
if((internal.getSize() <= 0 || internal.getSize() > 500000) || (previous.getSize() <= 0 ||
previous.getSize() > 500000)
|| (!(internal.getContentType().equals("image/jpeg"))) ||
(!(previous.getContentType().equals("image/jpeg"))) ) {
if( (!(internal.getContentType().equals("image/jpeg"))) ) {
theModel.addAttribute("internalMessage", "file must be jpg");
}
```

```
if( (!(previous.getContentType().equals("image/jpeg"))) ) {
theModel.addAttribute("previousMessage", "file must be jpg");
}
if(internal.getSize() <= 0 || internal.getSize() > 500000) {
theModel.addAttribute("internalMessage", "must be less than 500kb");
}
if(previous.getSize() <= 0 || previous.getSize() > 500000) {
theModel.addAttribute("previousMessage", "must be less than 500kb");
}
theModel.addAttribute("semester", semester);
theModel.addAttribute("internalPercent",internalPercent);
theModel.addAttribute("previousPercent", previousPercent);
return "student/student-marks-images";
}
RandomAlphanumericGenerator generator = new RandomAlphanumericGenerator();
String uniqueStringOne = generator.generate();
String uniqueStringTwo = generator.generate();
String newInternalFileName = uniqueStringOne+internal.getOriginalFilename();
String newPreviousFileName = uniqueStringTwo+previous.getOriginalFilename();
String pathOne =
request.getRealPath("/")+"resources"+File.separator+"uploads"+File.separator+newInter
nalFileName;
String pathTwo =
request.getRealPath("/")+"resources"+File.separator+"uploads"+File.separator+newPrevi
ousFileName;
byte [] imageOne = internal.getBytes();
byte[] imageTwo = previous.getBytes();
try {
FileOutputStream fosOne = new FileOutputStream(pathOne);
fosOne.write(imageOne);
```

```
fosOne.close();
}catch (IOException e) {
throw new FileOutException("Error in uploading File");
}
try {
FileOutputStream fosTwo = new FileOutputStream(pathTwo);
fosTwo.write(imageTwo);
fosTwo.close();
}catch (IOException e) {
throw new FileOutException("Error in uploading File");
}
DecimalFormat df = new DecimalFormat("0.00");
StudentSemData tempSemData = new StudentSemData();
tempSemData.setSemester(Integer.parseInt(semester));
tempSemData.setOverallInternalAssessment( Float.parseFloat( df.format(
Float.parseFloat(internalPercent)  ) ) );
tempSemData.setOverallPreviousAssessment(Float.parseFloat( df.format(
Float.parseFloat(previousPercent)  ) ) );
tempSemData.setInternalPic(newInternalFileName);
tempSemData.setExternalPic(newPreviousFileName);
spmsService.uploadMarks(tempSemData,currentStudent.getEmail());
return "redirect:/student/studentMarks";
}
@GetMapping("/studentNotification")
public  String studentNotification(Model theModel) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
```

```
}
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
List<Notification> notificationList =
spmsService.loadNotificationByBatchAndDept(currentStudent.getBatch(),
currentStudent.getDept());
if( !(notificationList.isEmpty()) ) {
int notificationId = notificationList.get(0).getId();
ReadNotification currentlyReadNotification = new ReadNotification();
if(currentStudent.getReadNotification() != null) {
currentlyReadNotification = currentStudent.getReadNotification();
}
currentlyReadNotification.setNotification_id(notificationId);
currentStudent.addReadNotification(currentlyReadNotification);
spmsService.updateStudent(currentStudent);
}
theModel.addAttribute("currentStudent", currentStudent);
theModel.addAttribute("notificationList", notificationList);
return "student/student-notification";
}
@GetMapping("/studentEvaluation")
public  String studentEvaluation(Model theModel) {
Student currentStudent = new Student();
List<StudentSemData> semData = new  ArrayList<StudentSemData>();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
```

```
semData = spmsService.loadStudentSemData(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("semData", semData);
theModel.addAttribute("currentStudent", currentStudent);
return "student/student-evaluation";
}
@GetMapping("/inbox")
public  String inbox(Model theModel) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
List<MessageCenter> studentInbox = new ArrayList<MessageCenter>();
studentInbox = spmsService.loadInboxByEmail(currentStudent.getEmail());
if( !(studentInbox.isEmpty()) ) {
int messageId = studentInbox.get(0).getId();
```

```
ReadStudentMessage currentlyReadMessage = new ReadStudentMessage();
if(currentStudent.getReadStudentMessage() != null) {
currentlyReadMessage = currentStudent.getReadStudentMessage();
}
currentlyReadMessage.setMessageId(messageId);
currentStudent.addReadStudentMessage(currentlyReadMessage);
spmsService.updateStudent(currentStudent);
}
theModel.addAttribute("currentStudent", currentStudent);
theModel.addAttribute("studentInbox", studentInbox);
return "student/inbox";
}
@GetMapping("/outbox")
public  String outbox(Model theModel) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("currentStudent", currentStudent);
List<MessageCenter> studentOutbox = new ArrayList<MessageCenter>();
```

```
studentOutbox = spmsService.loadOutboxByEmail(currentStudent.getEmail());
theModel.addAttribute("studentOutbox", studentOutbox);
return "student/outbox";
}
@GetMapping("/writeMessage")
public  String writeMessage(@RequestParam("toEmail") String toEmail,
Model theModel) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
NotificationChecker notificationChecker = new NotificationChecker(currentStudent,
spmsService);
boolean newNotification = notificationChecker.check();
theModel.addAttribute("newNotification", newNotification);
StudentMessageChecker messageChecker = new
StudentMessageChecker(currentStudent, spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
theModel.addAttribute("currentStudent", currentStudent);
theModel.addAttribute("toEmail", toEmail);
return "/student/write-message";
}
@PostMapping("/sendMessage")
public  String sendMessage(@RequestParam("toEmail") String toEmail,
@RequestParam("message") String message) {
Student currentStudent = new Student();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
```

```
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentStudent = spmsService.loadStudentByEmail(currentEmail);
}
MessageCenter newMessage = new MessageCenter();
newMessage.setFromEmail(currentStudent.getEmail());
newMessage.setToEmail(toEmail);
newMessage.setMessage(message);
spmsService.saveOrUpdateMessage(newMessage);
return "redirect:/student/inbox";
}
}
```

FacultyController.java :-

```
package com.adnan.icode.fun.spms.controllers;
@Controller
@RequestMapping("/faculty")
public class FacultyController {
@Autowired
private CookieCheckerRedirector checkerRedirector;
@Autowired
private SpmsService spmsService;
@InitBinder
public void initBinder(WebDataBinder dataBinder) {
StringTrimmerEditor trimmerEditor = new StringTrimmerEditor(true);
dataBinder.registerCustomEditor(String.class, trimmerEditor);
}
@GetMapping("/facultyLogin")
public String facultyLogin(HttpServletRequest request, HttpServletResponse response) {
String redirect = checkerRedirector.redirect(request, response);
```

```
if (redirect != null) {
return redirect;
}
return "faculty/faculty-login";
}
@GetMapping("/facultyAccessDenied")
public String accessDenied() {
return "faculty/faculty-access";
}
@GetMapping("/facultyPage")
public String facultyPage(Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if(!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
theModel.addAttribute("currentFaculty", currentFaculty);
return "faculty/faculty-home";
}
@GetMapping("/facultyProfile")
public String facultyProfile(Model theModel,
```

```
@ModelAttribute("message") String message) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
TempFacultyProfileModel profileModel = new TempFacultyProfileModel();
profileModel.setFirstName(currentFaculty.getFirstName());
profileModel.setLastName(currentFaculty.getLastName());
theModel.addAttribute("currentFaculty", currentFaculty);
theModel.addAttribute("profileModel", profileModel);
return "faculty/faculty-profile";
}
@GetMapping("/profileUpdating")
public String profileUpdating(@Valid
@ModelAttribute("profileModel") TempFacultyProfileModel profileModel,
BindingResult theBindingResult,
Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
```

```
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
if(theBindingResult.hasErrors()) {
theModel.addAttribute("currentFaculty", currentFaculty);
return "faculty/faculty-profile";
}else {
currentFaculty.setFirstName(profileModel.getFirstName().toUpperCase());
currentFaculty.setLastName(profileModel.getLastName().toUpperCase());
spmsService.updateFaculty(currentFaculty);
}
return "redirect:/faculty/facultyProfile";
}
@PostMapping("/imageUpdating")
public String imageUpdating(@RequestParam("image") CommonsMultipartFile file,
HttpServletRequest request,
Model theModel) throws FileOutException {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
```

```
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
if(file.getSize() <= 0 || file.getSize() > 500000 ) {
theModel.addAttribute("message", "must be less than 500kb");
return "redirect:/faculty/facultyProfile";
}
if(!(file.getContentType().equals("image/jpeg"))){
theModel.addAttribute("message", "file must be jpg");
return "redirect:/faculty/facultyProfile";
}
RandomAlphanumericGenerator generator = new RandomAlphanumericGenerator();
String uniqueString = generator.generate();
String newFileName = uniqueString+file.getOriginalFilename();
String path =
request.getRealPath("/")+"resources"+File.separator+"profile"+File.separator+newFileNa
me;
System.out.println(path);
byte[] image = file.getBytes();
try {
FileOutputStream fos = new FileOutputStream(path);
fos.write(image);
fos.close();
} catch (IOException e) {
```

```
throw new FileOutException("Error in uploading File");
}
currentFaculty.setProfilePic(newFileName);
spmsService.updateFaculty(currentFaculty);
return "redirect:/faculty/facultyProfile";
}
@GetMapping("/evaluation")
public  String evaluation(Model theModel,
@ModelAttribute("emailExist") String emailExist
) {
Faculty currentFaculty = new Faculty();
StudentFilterModel filterModel = new StudentFilterModel();
EmailModel emailModel = new EmailModel();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if (  mentorAuth != null  &&   ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR"))  )   ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
theModel.addAttribute("filterModel", filterModel);
theModel.addAttribute("emailModel", emailModel);
theModel.addAttribute("currentFaculty", currentFaculty);
return "faculty/faculty-evaluation";
```

```
}
@GetMapping("/studentFilterList")
public String studentFilterList(@Valid
@ModelAttribute("filterModel") StudentFilterModel filterModel,
BindingResult theBindingResult,
@ModelAttribute("emailModel") EmailModel tempEmail,
Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
theModel.addAttribute("currentFaculty", currentFaculty);
if(theBindingResult.hasErrors()) {
return "faculty/faculty-evaluation";
}
List<StudentSemData> semData = new ArrayList<StudentSemData>();
semData = spmsService.loadStudentSemDataByFilter(filterModel);
theModel.addAttribute("semData", semData);
theModel.addAttribute("searchType", "filter");
return "faculty/faculty-evaluation";
```

```
}
@GetMapping("/studentByEmail")
public  String studentByEmail(@Valid
@ModelAttribute("emailModel") EmailModel tempEmail,
BindingResult theBindingResult,
@ModelAttribute("filterModel") StudentFilterModel filterModel,
Model theModel) {
Faculty currentFaculty = new Faculty();
List<StudentSemData> semData = new  ArrayList<StudentSemData>();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if (  mentorAuth != null  &&   ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR"))  )   ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
theModel.addAttribute("currentFaculty", currentFaculty);
if(theBindingResult.hasErrors()) {
return "faculty/faculty-evaluation";
}
Student tempStudent = spmsService.loadStudentByEmail(tempEmail.getEmail());
if(tempStudent == null)  {
theModel.addAttribute("emailExist", "email does not exist");
return "redirect:/faculty/evaluation";
```

```
}
semData = spmsService.loadStudentSemData(tempEmail.getEmail());
theModel.addAttribute("semData", semData);
theModel.addAttribute("searchType", "email")
return "faculty/faculty-evaluation";
}
@GetMapping("/assessStudent")
public String assessStudent(@RequestParam("email") String studentEmail,
@RequestParam("semester") int studentSemester,
@RequestParam("searchType") String searchType,
Model theModel) {
Faculty currentFaculty = new Faculty();
Student assessStudent = new Student();
StudentFilterModel filterModel = new StudentFilterModel();
EmailModel emailModel = new EmailModel();
List<SubjectsList> subjects = new ArrayList<SubjectsList>();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
assessStudent = spmsService.loadStudentByEmail(studentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
```

```
String studentName = assessStudent.getFirstName()+" "+assessStudent.getLastName();
String studentContact = assessStudent.getContactNo();
String studentFather = assessStudent.getFatherName();
String studentGuardianContact = assessStudent.getGuardianContactNo();
subjects = spmsService.loadSubjectList(assessStudent.getBatch(), studentSemester,
assessStudent.getDept(), currentFaculty.getEmail());
theModel.addAttribute("filterModel", filterModel);
theModel.addAttribute("emailModel", emailModel);
theModel.addAttribute("currentFaculty", currentFaculty);
theModel.addAttribute("subjects", subjects);
theModel.addAttribute("semester", studentSemester);
theModel.addAttribute("email", studentEmail);
theModel.addAttribute("studentName", studentName);
theModel.addAttribute("studentContact", studentContact);
theModel.addAttribute("studentFather", studentFather);
theModel.addAttribute("studentGuardianContact", studentGuardianContact);
theModel.addAttribute("searchType", searchType);
return "faculty/faculty-subject";
}
@GetMapping("/assessStudentSubSelected")
public  String assessStudentSubSelected(
@RequestParam("semester") int studentSemester,
@RequestParam("subject") String subjectName,
@RequestParam("email") String studentEmail,
@RequestParam("searchType") String searchType,
Model theModel
) {
Faculty currentFaculty = new Faculty();
Student assessStudent = new Student();
StudentFilterModel filterModel = new StudentFilterModel();
```

```
EmailModel emailModel = new EmailModel();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if (!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
assessStudent = spmsService.loadStudentByEmail(studentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
String studentName = assessStudent.getFirstName()+" "+assessStudent.getLastName();
theModel.addAttribute("filterModel", filterModel);
theModel.addAttribute("emailModel", emailModel);
theModel.addAttribute("currentFaculty", currentFaculty);
theModel.addAttribute("studentName", studentName);
theModel.addAttribute("semester", studentSemester);
theModel.addAttribute("subject", subjectName);
theModel.addAttribute("email", studentEmail);
theModel.addAttribute("searchType", searchType);
StudentPerSubject perSubject = new StudentPerSubject();
PerSubjectModel perSubjectModel = null;
perSubject = spmsService.loadPerSubjectData(studentEmail, studentSemester,
subjectName);
if(perSubject != null) {
perSubjectModel = new PerSubjectModel();
```

```
perSubjectModel.setId( perSubject.getId() );
perSubjectModel.setBehaviour(String.valueOf( perSubject.getBehaviour() ) );
perSubjectModel.setFocus(String.valueOf( perSubject.getFocus() ) );
perSubjectModel.setAttendance(String.valueOf( perSubject.getAttendance() ) );
perSubjectModel.setOverallSubject(String.valueOf( perSubject.getSubjectOverall() ) );
}else {
perSubjectModel = new PerSubjectModel();
}
theModel.addAttribute("perSubject", perSubjectModel);
return "faculty/faculty-assessment";
}
@GetMapping("/assessmentProcessing")
public  String assessmentProcessing(@Valid
@ModelAttribute("perSubject") PerSubjectModel perSubjectModel,
BindingResult theBindingResult,
@ModelAttribute("filterModel") StudentFilterModel filterModel,
@ModelAttribute("emailModel") EmailModel tempEmail,
RedirectAttributes redirectAttributes,
Model theModel
) {
if(theBindingResult.hasErrors()) {
return "redirect:/faculty/evaluation";
}
float overallSubjectPercent;
overallSubjectPercent =
(float)( Integer.parseInt(perSubjectModel.getBehaviour()) +
Integer.parseInt(perSubjectModel.getFocus()) +
Integer.parseInt(perSubjectModel.getAttendance()))/30;
overallSubjectPercent = overallSubjectPercent * 100;
DecimalFormat df  = new DecimalFormat("0.00");
```

```
overallSubjectPercent = Float.parseFloat( df.format(overallSubjectPercent) );
StudentPerSubject perSubject = new  StudentPerSubject();
perSubject.setId( perSubjectModel.getId() );
perSubject.setSemester( perSubjectModel.getSemester() );
perSubject.setSubjectName( perSubjectModel.getSubject() );
perSubject.setBehaviour( Integer.parseInt(perSubjectModel.getBehaviour() ) );
perSubject.setFocus( Integer.parseInt( perSubjectModel.getFocus() ) );
perSubject.setAttendance( Integer.parseInt( perSubjectModel.getAttendance() ) );
perSubject.setSubjectOverall(overallSubjectPercent);
spmsService.saveOrUpdateSubjectData(perSubject, perSubjectModel.getEmail());
Student currentStudent =
spmsService.loadStudentByEmail(perSubjectModel.getEmail());
List<SubjectsList> allSemSubjects = new ArrayList<SubjectsList>();
allSemSubjects = spmsService.loadSubjectList(currentStudent.getBatch(),
perSubject.getSemester(), currentStudent.getDept());
List<StudentPerSubject> allFilledSubjects = new ArrayList<StudentPerSubject>();
allFilledSubjects = spmsService.loadPerSubjectDataList( currentStudent.getEmail(),
perSubject.getSemester() );
if( (allSemSubjects.size()) == (allFilledSubjects.size()) ) {
float overallSubjectPercentage;
OverallSubjectsPercentageCal subjectsPercentageCal = new
OverallSubjectsPercentageCal();
overallSubjectPercentage = subjectsPercentageCal.calculate(allFilledSubjects);
StudentSemData overallSemData = new StudentSemData();
PercentageController pController = new PercentageController();
overallSemData = spmsService.loadStudentOneSemData(currentStudent.getEmail(),
perSubject.getSemester() );
pController = spmsService.loadPercentageControllerByDept(currentStudent.getDept());
overallSemData.setOverallSubjectAssessment(overallSubjectPercentage);
float overallevaluationPercentage;
```

```
OverallAssessmentCal overallAssessmentCal = new
OverallAssessmentCal(overallSemData, pController);
overallevaluationPercentage = overallAssessmentCal.overallPercent();
String learnerType=null;
if(pController != null ){
if(pController.getOverallThreshold() > overallevaluationPercentage) {
learnerType = "SLOW";
}else if(pController.getOverallThreshold() <= overallevaluationPercentage) {
learnerType = "FAST";
}
}else {
if(60 > overallevaluationPercentage) {
learnerType = "SLOW";
}else if(60 <= overallevaluationPercentage) {
learnerType = "FAST";
}
}
overallSemData.setOverallEvaluation(overallevaluationPercentage);
overallSemData.setLearnerType(learnerType);
spmsService.uploadMarks(overallSemData, currentStudent.getEmail());
}
if(perSubjectModel.getSearchType().equals("filter")) {
filterModel.setDept(currentStudent.getDept());
filterModel.setBatch(String.valueOf(currentStudent.getBatch()));
filterModel.setSemester(String.valueOf(perSubject.getSemester()));
redirectAttributes.addFlashAttribute("filterModel", filterModel);
return "redirect:/faculty/studentFilterList";
}
tempEmail.setEmail(currentStudent.getEmail());
redirectAttributes.addFlashAttribute("emailModel", tempEmail);
```

```
return "redirect:/faculty/studentByEmail";
}
@GetMapping("/semData")
public String semData(Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if(!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
SortedSemDataModel semData = new SortedSemDataModel();
theModel.addAttribute("currentFaculty", currentFaculty);
theModel.addAttribute("semData", semData);
return "faculty/summary-students";
}
@GetMapping("/sortSemData")
public String sortSemData(@Valid
@ModelAttribute("semData") SortedSemDataModel semData,
BindingResult theBindingResult,
Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
```

```
if(!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
theModel.addAttribute("currentFaculty", currentFaculty);
if(theBindingResult.hasErrors()) {
return "faculty/summary-students";
}else {
List<StudentSemData> semDataList =
spmsService.loadStudentSemDataByLearner(semData);
theModel.addAttribute("semDataList", semDataList);
return "faculty/summary-students";
}
}
@GetMapping("/facultyList")
public  String facultyList(Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if(!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
```

```
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
}
theModel.addAttribute("currentFaculty", currentFaculty);
theModel.addAttribute("deptModel", new DepartmentOnlyModel());
return "faculty/view-faculty";
}
@GetMapping("/getList")
public String getList(@Valid
@ModelAttribute("deptModel") DepartmentOnlyModel deptModel,
BindingResult theBindingResult,
Model theModel) {
Faculty currentFaculty = new Faculty();
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
if(!(authentication instanceof AnonymousAuthenticationToken)) {
String currentEmail = authentication.getName();
currentFaculty = spmsService.loadFacultyByEmail(currentEmail);
}
Authentication mentorAuth = SecurityContextHolder.getContext().getAuthentication();
if ( mentorAuth != null && ( mentorAuth.getAuthorities().stream().anyMatch(a ->
a.getAuthority().equals("ROLE_MENTOR")) ) ) {
MentorMessageChecker messageChecker = new MentorMessageChecker(currentFaculty,
spmsService);
boolean newMessage = messageChecker.check();
theModel.addAttribute("newMessage", newMessage);
```

```
}
theModel.addAttribute("currentFaculty", currentFaculty);
if(theBindingResult.hasErrors()) {
return "faculty/view-faculty";
}else {
List<Faculty> facultyList =  spmsService.loadFacultyByDept(deptModel.getDept());
List<Faculty> enabledFacultyList = new ArrayList<Faculty>();
if( !(facultyList.isEmpty()) ) {
for(Faculty tempFaculty : facultyList) {
if(tempFaculty.getEnabled() == true) {
enabledFacultyList.add(tempFaculty);
}}}
theModel.addAttribute("facultyList", enabledFacultyList);
return "faculty/view-faculty";
}}}
```

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

According to the conclusions of the study and survey, a student's performance evaluation serves as the most important aspect of determining the overall growth of a student's professional career and personality traits. Scholars have advocated a variety of approaches for carrying out this procedure. Most analysts employed data mining techniques and artificial intelligence procedures to appraise an individual based on predefined criteria that fit their assessment and study, and they obtained results that supported an individual's performance. Our research focuses on one such deployment, which will give an automatic student performance evaluation to assist the institution's staff in effectively monitoring an individual student's academic and professional advancement, as well as individual mentoring of a student.

This system provides an effective and efficient way of analysis of performance of students, and it also provides a way to efficiently gather all the required data and assess the students effectively and in a fast manner. The results concluded at the end were proven to be satisfactory as they were at par with the results obtained by the manual procedure conducted at the college by the faculty. This system, if deployed, would be easily scalable and every department head would be able to enter the details of their respective departments such as faculty and students. Also, the database we have used is MySQL, which is compatible with most of the servers, but if the user wishes to use any other database, he/she can do so easily with minimal changes in the program.

5.2 FUTURE SCOPE

The proposed project can be further extended in the future by incorporating a technique of machine learning. The machine learning algorithms can be infused into the said project to make it more precisely automatic, predictive, and accurate. Various machine learning algorithms help with predictive analysis and classification, which are the basic features of

our project. Machine learning is an application of artificial intelligence which provides a system with comprehensibility of its own and gives power to the system to think and act on its own without being explicitly programmed. The input data/old data provided by the student or previous assessments of the students can be analyzed by the system using a specific machine learning algorithm which can then try to find the patterns in the data and based on the type of pattern the system will try to make predictive analysis and segregate the students into slow and fast learners even before a new set of data is available to it.

Researchers can choose the algorithm according to their feasibility and the system they want to develop but an unsupervised learning methodology can be implemented to train the system to learn from unlabeled datasets which will be provided at the initial state of the working of the system. In this case, we provide a mixed dataset which consists of a collection of data about the student like the previous year's results, midsemester results, and subject-wise assessment and it can even consist of the previous category of the student. This dataset is fed into the model and the model analyses the data to figure out the patterns in it in the end, it categorizes the students based on the patterns in their dataset into slow and fast learners. Herein we provide the data to the system and the system does the rest of the work.

Artificial Intelligence can also be used in the project with the assistance of machine learning where the growing technology of bots can be used for personal as well as academic-related mentoring.

A chatbot can be incorporated into the project wherein the software will aim in collecting data from students and then aid in academic research related to the seminar or provide personal motivation during self-study.

Artificial intelligence can also be used in the project by incorporating an emotion detection sensor that will detect the emotions of students exhibited during a course of the work such as joy or anger.

It can also be used to estimate the time spent on a particular work and estimate the performance of a student by visually presenting the outcomes.

CHAPTER 6

REFRENCES

[1]. Hsiu-Ping, Tzy-Ling Chen, Li-An Chiu, San-Liang Lee, An-Bang Wang (2012) "Student Evaluation of Teaching Effectiveness of a Nationwide Innovative Education Program on Image Display Technology".

[2]. Gao Min (2018) School of Foreign Studies, Northwestern Polytechnical University, Xi'an Shaanxi, China" Information Management System for Comprehensive Quality Evaluation of Students Based on the Fusion of Information Exchange Model".

[3]. Seifedine Kardy (2015) American University of the Middle East EGAILA,Kuwait, "Systematic Assessment of Student Outcomes in Mathematics for Engineering Students".

[4]. Javeriya Farheen, Dr. Sunanda Dixit (2018), Dayananda Sagar College of Engineering Bangalore, India,"E- Mentoring System Application".

[5]. Kochhar, S.K. (2000). Educational and Vocational Guidance in Secondary Schools. New Delhi: Sterling Publishers Private Limited.

[6]. Hanushek, E. A., Kain, J. F., Markman, J. M., & Rivkin, S. G. (2003). Does peer ability affect student achievement?. Journal of applied econo-met-rics,18(5), 527-544

[7]. Ganeshan, Kathiravelu and Li, Xiaosong. " An intelligent student advising system using collaborative filtering", 2015 IEEE Frontiers in Education Conference (FIE), pp. 1-8, Oct. 2015.

[8]. Roberts, K. L., & Sampson, P. M. (2011). School board member professional development and effects on student achievement. International Journal of Educational

Management, 25(7), 701-713

[9]. Huang, Zhifeng and Nagata, Ayanori and Kanai-Pak, Masako and Maeda, Jukai and Kitajima. ” Self-Help Training System for Nursing Students to Learn Patient Transfer Skills” IEEE, vol. 7, pp. 319-332, Oct 2014.

[10]. Edin Osmanbegović and Mirza Suljic, DATA MINING APPROACH FOR PREDICTING STUDENT PERFORMANCE, Economic Review Journal of Economics and Business, Vol. X, Issue 1, May 2012.

[11]. Simpson, Jane and Fernandez, Eugenia.”Student per- formance in first year, mathematics, and physics courses: Implications for success in the study of electrical and computer engineering”, 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, pp. 14, Oct 2014.

[12]. Rangvid, B. S. (2003). Educational peer effects quantile regression evidence from Denmark with PISA2000 data. Do Schools Matter, 45.

[13]. Chen, Hsuan-Hung and Chen, Yau-Jane and Chen, Kim-Joan. ” The Design and Effect of a Scaffolded Concept Mapping Strategy on Learning Performance in an Undergrad- uate Database Course” IEEE Transactions on Education, vol. 56, pp. 300-307, Aug 2013.

[14]. Kolo David Kolo, Solomon A. Adepoju, John Kolo Alhassan, A Decision Tree Approach for PredictingStudents Academic Performance, I.J. Education and Management Engineering, 2015, 5, 12-19Published Online October 2015 in MECS(http://www.mecs-press.net) DOI: 10.5815/ijeme.2015.05.02.

[15]. Barney, Sebastian and Khurum. ” Improving Students WithRubricBased Self-Assessment and Oral Feedback”, IEEE Transactions on Education, vol. 55, pp. 319-325,

Aug 2012.

[16]. R. Sumitha and E.S. Vinoth kumar, Prediction of Students Outcome Using Data Mining Techniques,International Journal of Scientific Engineering and Applied Science (IJSEAS) Volume-2, Issue- 6,June2016 ISSN: 2395-3470.

[17]. Lopez Guarin, Camilo Ernesto. ” A Model to Predict Low Academic Performance at a Specific Enrollment Using Data Mining”, IEEE Revista Iberoamericana de Tecnologias del Aprendizaje, vol. 10, pp. 119-125, Aug 2015.

[18]. Maria Goga, Shade Kuyoro, Nicolae Goga, A recommender for improving the student academicperformance, Social and Behavioural Sciences 180 (2015) 1481-1488.

[19]. Mrinal Pandey and S. Taruna, Towards the integration of multiple classifiers pertaining to the student's performance prediction,http://dx.doi.org/10.1016/j.pisc.2016.04.0762213-0209/© 2016 Published by Elsevier GmbH. This is an open access article under the CC BY-NC-ND license(http://creativecommons.org/licenses/by- nc-nd/4.0/).

[20]. Grivokostopoulou, Foteini. ” Utilizing semantic web technologies and data mining techniques to analyze students learning and predict final performance” 2014 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE), pp. 488-494, Dec 2014.

[21]. U.bin Mat, N. Buniyamin, P. M. Arsad, R. Kassim, An overview of using academic analytics to predict and improve students’ achievement: A proposed proactive intelligent intervention, in: Engineering Education (ICEED), 2013 IEEE 5th Conference on, IEEE, 2013, pp. 126–130.

[22]. Sa, Chew Li and bt. Abang Ibrahim, Dayang Hanani. ” Student performance

analysis system (SPAS)", The 5th International Conference on Information and Communication Technology for The Muslim World (ICT4M), pp. 1-6, Nov. 2014.

[23]. Abeer Badr El Din Ahmed and Ibrahim Sayed Elaraby, Data Mining: A prediction for Student'sPerformance Using Classification Method, World Journal of Computer Application and Technology2(2): 43-47, 2014.

[24]. Achumba, I. E. and Azzi, D. and Dunn, V. L. and Chukwudebe, G.A."Intelligent Performance Assessment of Students' Laboratory Work in a Virtual Electronic Laboratory Environment." IEEE. Transactions on Learning Technologies, vol. 6, pp. 103116, Apr 2013.

[25]. Jyoti Bansode, Mining Educational Data to Predict Student's Academic Performance, InternationalJournal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169, Volume:4 Issue: 1, 2016.

[26]. Alexander Tobias Neumann, Tamar Arndt , Laura Köbis, Roy Meissner , Anne Martin,Peter de Lange, Norbert Pengel, Ralf Klamma and Heinz-Werner Wollersheim," Chatbots as a Tool to Scale Mentoring Processes: Individually Supporting Self-Study in Higher Education" ,13th may 2021.

[27]. Milos Kravcik ,Katharina Schmid, Christoph Igel,(2019),"Towards Requirements for Intelligent Mentoring Systems."

[28]. Faiyaz Doctor Member IEEE, Rahat Iqbal." An Intelligent Framework for Monitoring Student Performance Using Fuzzy Rule-Based Linguistic Summarisation", 10-15, June .2012.

[29]. Laecio Araujo COSTA, Leandro Manuel PEREIRA SANCHES, Ricardo José ROCHA AMORIM," Monitoring Academic Performance Based on Learning Analytics

and Ontology",December,2019.

[30]. GeeksForGeeks, www.geeksforgeeks.org/