Academic course outcome analytics

SABARISH M, PERCY JONE N, RAM KUMAR J, PRANAV KR
Department of Computer Science and Engineering, PSNA College of Engineering and Technology Dindigul, TamilNadu, India

***Abstract*— This paper presents Academic Course Outcome Analytics, a web-based platform designed to provide detailed analysis of student performance across individual Course Outcomes (COs). Faculty members can upload question papers for exams, assignments, and case studies, with the system automatically mapping questions to specific syllabus units. Faculty enter student marks on a per-question basis, allowing precise tracking of performance. The system compiles a searchable question bank, enabling students to access past questions by topic for practice. Interactive visual reports support data-driven decision-making for faculty, administrators, and students. The platform enhances academic assessment, facilitates personalized learning, and improves educational outcomes through secure data entry, automated tracking, and an intuitive interface.**

***Keywords*— Course Outcome, Question Bank, Performance Analytics, Educational Assessment, CO Mapping.**

I. INTRODUCTION

Academic Course Outcome Analytics is a web-based application that allows for CO-wise (Course Outcome-wise) analysis of student performance through examinations, assignments, case studies, and other learning assessments used within the curriculum. Faculty can upload examination and assignment question papers into the program, which will map every question automatically as the system will allocate each question to a specific syllabus unit (CO). It provides a more accurate measurement of student marks at the unit level. Also, the application collates all the questions from the exams into a searchable question bank so that students can access past questions by topic and practice them. Finally, it generates a comprehensive performance report for faculty, administrators, and students to help identify trends so that all concerned parties can make data-informed decisions. It is a more efficient way to view student performance, while also providing for secure data entry for the tracking of academic assessment, focused learning, and ultimately improved educational outcomes.

II. LITERATURE REVIEW

With the increasing focus on data-informed education, many systems have been created to assess and improve student learning outcomes. However, traditional assessments often provide only summative scores, which prevents a deep examination of student outcomes. A particularly useful approach in this regard is the unit-wise (i.e. Course Outcome-wise) analysis of results. Previous research has shown that examining unit-wise performance is a means by which faculty can identify units that students struggle with, and differentiate learning outcomes to enable targeted teaching approaches leading to improved learning results.

Generally existing educational platforms (e.g., Moodle, Blackboard) offer basic assessment capabilities, but they fail to provide granular unit-level detail as well as dynamic question banks that enable student self-practice; even if they did, the combination of a limited course outcome framework and knowledge level approach is seldom implemented. There are tools which help manage OBE elements, e.g., create rubrics to map questions to course outcomes, or which enable outcome monitoring using rubrics and statistics but rarely both. In addition, common education management tools do a poor job of providing a qualitative user interface which faculty and students can intuitively access to review individual performance at the level of individual questions.

The Academic Course Outcome Analytics system has been developed to address three critical functions to increase the depth of analytics – CO wise performance tracking; a automated question bank, and analytics which offer actionable insights through an interactive visualization approach. Each function works together to improve the teaching/learning experience by providing data to generate informed academic planning and planning for continuous improvements of learning outcomes.

III. SYSTEM ARCHITECTURE

A. Faculty Portal
- Upload question papers
- Automatic question-to-unit mapping
- Student Mark entry per Question basis
- Generate Report

B. Student Interface
- Check performance reports
- View CO-wise past questions

C. Admin Dashboard
- Manage user roles
- CRUD operation for students, faculty and department
- Review analytics
- Generate reports

IV. AUTOMATED QUESTION MAPPING MECHANISM

To ensure precise alignment between assessment items and Course Outcomes (COs), our system implements an automated question mapping mechanism that parses structured question papers and systematically extracts relevant metadata. The end-to-end process is outlined as follows:

- Input Format and Parsing

The system accepts question papers in .docx format to ensure consistency and compatibility with automated parsing routines. We utilize the Mammoth library to convert .docx documents into clean, semantic HTML by removing extraneous styling and focusing on core content. This HTML is then processed using Cheerio, a fast and lightweight server-side DOM manipulation tool, to extract structured data elements.

- Identifying the Question Table

The HTML content is scanned to detect tables that may contain question metadata. A table is classified as a valid question table if its header row includes any combination of the following keywords: Q.No, Questions, Marks, CO, PI, and BL. This keyword-based filtering allows the system to isolate only the tables relevant to academic assessment.

- Extracting Questions and Metadata

Once a valid table is identified, the subsequent rows are interpreted as individual question entries. The system dynamically maps each column header to its respective attribute, allowing for flexibility in the column arrangement. For example, the Marks and CO columns may appear in varying positions across different documents, yet the system accurately associates each value with the correct field through intelligent header tracking.

- Mapping to Syllabus Units

Each question is assigned to a specific CO based on the uploaded metadata. These COs are pre-associated with syllabus units as configured by faculty during the course setup phase. This mapping enables fine-grained performance analysis across individual syllabus units and promotes consistency in evaluation.

- Error Handling and Validation

To maintain data quality and integrity, the system performs validation checks during parsing. If mandatory columns (e.g., CO, Marks) are missing or if the table structure is inconsistent, the system halts processing and notifies the user, thereby preventing the entry of invalid data into the database.

- Flexibility in Table Design

The platform supports flexible table designs. Column order is not fixed, and additional attributes—such as Bloom's Taxonomy Level (BL) or Program Indicators (PI)—can be included as needed. The dynamic header recognition logic ensures accurate extraction of each question's metadata, regardless of layout variations.

- Support for Multiple Tables

For documents containing multiple tables (e.g., different sections or types of assessments), the system iterates through each table independently, evaluating them for structural validity. Only tables containing recognized headers are parsed, while others are ignored to avoid false data capture.

- Scalability and Performance

The combined use of Mammoth and Cheerio ensures efficient, high-performance parsing with minimal memory overhead. This makes the system well-suited for institutional use, capable of handling large documents or batch uploads without performance degradation.

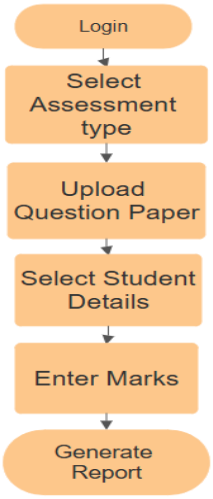
- CO-to-Unit Association Logic

During course configuration, each Course Outcome is mapped to one or more syllabus units. When a question is linked to a CO, the system automatically

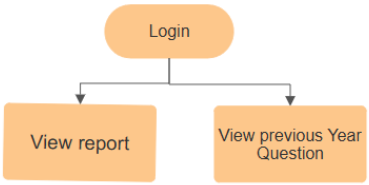
determines the relevant unit(s) using these predefined mappings. This relationship enables downstream modules—such as report generation—to organize and display performance data unit-wise.

V. USER FLOW DIAGRAM

1. Faculty Login



2. Student Login



VI. USES CASES

- Faculty uploads question paper

The system automates the extraction and mapping of questions to the appropriate Course Outcomes (COs) and syllabus units, while providing the ability to input student grades for CO analysis.

- Faculty enters student marks

This feature allows faculty to input the marks of each

- Students for every question they have attended.

Students view unit-wise questions across subjects
Students can search for past questions from specific units, without needing to sift through an entire exam bank, for focused practice.

- Admin checks class-level trends

Administrator can analyse CO-level performance and student assessment data for academic planning and data-driven decision-making.

VII. IMPLEMENTATION STRATEGY

Phase 1: Database Schema and Question Parsing
Design and develop the database schema to store questions, marks and CO mappings. Create and implement the parsing functionalities to extract and store question information from uploaded documents.

Phase 2: Question Bank and CO Mapping
Create a question bank that is searchable, and dynamic in how questions are referenced automatically to its course outcomes (COs) as well as course syllabus units.

Phase 3: Report Generation and UI
Create CO-wise performance reports, and provide a user interface that can be used by faculty, students and administration.

Phase 4: Deployment & Feedback
Deploy application to a cloud platform. Obtain user feedback that will assist in improving the performance, usability, and user experience of the system as a whole.

VIII. TECHNOLOGY STACK

- Frontend: React
- Backend: Node.js, Express
- Database: PostgreSQL
- ORM: PRISMA

IX. LIMITATIONS AND FUTURE WORK

LIMITATIONS:

- Auto-Mapping Accuracy

→Different document formats may introduce variation in question-to-CO mapping.
→ If the mapping was inaccurate, this may take some manual effort to fix.

- Structured Data Dependency

→ The system relies on structured formats (.docx)
→ Unstructured data may lead to parsing errors.

- Scalability of Reporting

→ With larger datasets, real-time report generation may be slow.
→ Will have to optimize the backend for performance.

FUTURE WORK:

- AI for Intelligent Mapping

→ Utilize NLP to improve the accuracy of question-to-CO mapping.
→ Enhance mapping accuracy for complex or unstructured documents.

- Personalized Learning Suggestions

→ Introduce AI-supported and AI-driven recommendations for learning.
→ Curate study resources based on performance and provide personalized recommendations.

- Mobile App Integration

→ Develop a mobile version for easy access on a mobile device.
→ Improve experience for faculty and students where increased mobility is essential.

- LMS Integration

→ Allow integration and use existing systems in the LMS.
→ Sync said performance data and reports between systems

X. CONCLUSION

The Analytical System for Academic Course Outcomes helps both instructors and students with data-driven information to understand student achievement at the CO level. Assuring clarity in the assessments administered, the system supports question mapping to the syllabus units, contributing to consistent and accurate evaluation of assessment tasks. With the reporting capabilities and actionable reports, instructors can analyze and make decisions to improve teaching practice. And for students, it offers specific practice with a searchable question bank, and students can determine where their practice is targeted, delivering a more personalized way a student can study. Provided the amount of assessment data available, the system supports continuous improvement in student learning and achievement by providing assessors with a more accessible way to use data and make actionable decisions.

XI. REFERENCES

[1] Prisma, "Prisma ORM Documentation," [Online]. Available: https://www.prisma.io/docs/.
[2] Facebook, "React: A JavaScript library for building user interfaces," [Online]. Available: https://reactjs.org/.
[3] Mamoth.js, "Mamoth: Convert .docx documents into HTML," [Online].Available: https://github.com/mwilliamson/mamoth.

APPENDIX