CSE 3243 Web Programming Lab
Mini Project Report on

Automated Web-Based System for Course and Program Outcome Tracking and Analysis

Under the Guidance of:

Prof. Manamohana K, Dr. Roopalakshmi R and Dr. Rajashree Krishna

Department of Computer Science and Engineering
Manipal Institute of Technology, Manipal, Karnataka – 576104

2024-25

Abstract

Traditional academic outcome tracking systems relying on manual Excel-based workflows for Course Outcomes (CO), Program Outcomes (PO), and Program Specific Outcomes (PSO) analysis are prone to inefficiencies, human errors, and lack real-time collaboration. This project addresses these challenges by developing an Automated Web-Based System for Course and Program Outcome Tracking and Analysis, leveraging the MERN (MongoDB, Express.js, React, Node.js) stack. The system streamlines target setting, attainment calculation, and root cause analysis through role-based workflows for administrators, faculty, course coordinators, and HoDs. By digitizing outcome-based education (OBE) workflows, this solution enhances transparency, enables data-driven curriculum improvements, and provides our institution with a scalable framework for academic quality assurance.

Key deliverables include:

- **Automated Attainment Calculation**: Direct (80% CIE/SEE marks) and indirect (20% student feedback) attainment computation with multi-tiered analytics.
- **Excel Integration**: Bulk upload of marks, feedback, and profiles via standardized templates, reducing manual data entry.
- **Analytics Dashboards**: Comparisons of target vs. attainment gaps, action plans, and root cause analysis.
- **Role-Based Access Control**: Secure JWT authentication ensures data integrity, with restricted permissions for target/mapping edits.

Table of Contents

1. Introduction

In the current academic setup, Course Outcomes (CO), Program Outcomes (PO), and Program Specific Outcomes (PSO) are manually tracked and analysed using Excel sheets. This process, while effective, is time-consuming, prone to errors, and lacks real-time accessibility. To address these challenges, this project aims to develop a web-based platform that automates CO-PO-PSO tracking, assessment, and attainment analysis for educational institutions.

The proposed system will transform the manual Excel-based workflow into an interactive and user-friendly web application, enabling faculty members, course coordinators, and administrators to input, manage, and analyse academic data efficiently. The platform will support various functionalities, including:

- **Target Setting**: Faculty members and Heads of Departments (HODs) can set CO, PO, and PSO targets for each semester, ensuring predefined benchmarks for academic performance.
- **CO-PO-PSO Mapping**: A structured interface for entering and displaying CO-PO-PSO mapping, replacing static Excel sheets.
- **Continuous Internal Evaluation (CIE) Assessment**: Faculty can enter and track internal assessment marks, linking them directly to COs for seamless attainment analysis.
- **Semester End Examination (SEE) Assessment**: Faculty will input SEE marks mapped to COs, enabling comprehensive CO-wise evaluation.
- **Attainment Analytics**: Automated calculation of CO and PO attainment based on predefined weightages (e.g., 60% CIE, 40% SEE for direct attainment) and feedback analysis (80% direct, 20% feedback-based attainment).
- **Course Feedback Integration**: Collection and analysis of student feedback through MS Forms or an integrated module, contributing to CO attainment insights.
- **Root Cause Analysis & Action Plan**: Identification of gaps in attainment and provision for corrective actions, including structured action plans for both CO and PO.

The system will ensure seamless data input via an intuitive web interface with functionalities like file upload (drag and drop for assessment marks), real-time visualization of attainment metrics, and automated report generation. Additionally, it will allow multi-user access, ensuring role-based permissions for professors, course coordinators, and administrators.

By transitioning from Excel-based tracking to a dynamic web-based solution, this platform will enhance accuracy, reduce manual effort, and provide data-driven insights for continuous academic improvement. The proposed system aligns with accreditation requirements and institutional quality assurance frameworks, fostering a more structured and transparent academic assessment process.

2. System Analysis

2.1 Product Features

1. Role-Based Access Control
    - Admin: Course and section creation, course coordinator and HoD assignment, and faculty subject allocation.
    - HOD: Department-wide view of faculties and students.
    - Course Coordinator: Set CO-PO, CO-PSO mappings and targets.
    - Faculty: Upload CIE/SEE marks and course feedback.
2. Streamlined Workflows
    - Automated analytics computation.
    - Target vs. attainment comparisons for CO-PO and CO-PSO.
    - Action plan input for underperforming CO's and PO's.
    - Root Cause Analysis input for CO's, PO's and PSO's.
3. Data Management
    - Excel template integration for bulk upload of marks and, student, professor and course profiles.

2.2 User Classes and Characteristics

2.2.1 Admin

- Course and section creation, course coordinator and Head of Department assignment, and faculty subject allocation.

2.2.2 HOD

- Monitor department faculty and student profiles.

2.2.3 Course Coordinator

- Set targets and provide CO-PO, CO-PSO maps.

2.2.4 Faculty

- Upload CIE and SEE marks and course feedback.
- Provide Root Cause Analysis for CO's, PO's and PSO's.
- Provide Action Plan for CO's and PO's.

2.3 Operating Environment

2.3.1 Hardware Requirements

- Servers: 4 GB RAM, 2-core CPU (AWS EC2 t3.medium)
- Clients: Devices with modern browsers

2.3.2 Operating Systems:

- Windows 10+
- macOS 10.15+
- Linux (Ubuntu 20.04+)
- Android/iOS

2.3.3 Web Browsers:

- Chrome 85+
- Firefox 80+
- Safari 14+
- Edge 85+

2.4 Use Case Descriptions

2.4.1 UC-1

Title: Creation of faculty, course and section profiles

Description: The admin obtains data from the Department office and uploads it, populating database with faculty, course and section profiles.

Actor: Admin

Preconditions:

1. Admin obtains data from the Department office.
2. Data to be uploaded is in the correct Excel format.

Postconditions:

1. Database is populated with faculty, course and section profiles.

Main Success Scenario:

1. Admin navigates to the webpage for uploading faculty, course and section data.
2. Admin uploads the data.
3. Admin is prompted with an alert on successful upload.

Exceptions:

1. Data uploaded is not in the correct Excel format.

2.4.2 UC-2

Title: Assigning course coordinator and Head of Department roles

Description: The admin assigns role of course coordinator of a course and role of Head of Department of a department to a faculty member.

Actor: Admin

Preconditions:

1. Database contains course and faculty profiles.

Postconditions:

1. Roles of course coordinator and Head of Department assigned.

Main Success Scenario:

1. Admin navigates to the webpage for assigning roles of course coordinator and Head of Department.

2. Selects a faculty member as course coordinator for a course and Head of Department for a department.

3. Admin is prompted with an alert on successful assignment of roles.

Exceptions:

1. Selected faculty member for the roles of course coordinator and HoD do not exist in the database.

2.4.3 UC-3

Title: Allocation of courses to faculty

Description: The admin allots courses to faculty for a particular semester and section.

Actor: Admin

Preconditions:

1. Database contains course and faculty profiles.

Postconditions:

1. Courses assigned to faculty for a particular semester and section.

Main Success Scenario:

1. Admin navigates to the webpage for assigning courses to faculty members.

2. Selects a faculty member and allots courses for a particular semester and section.

3. Admin is prompted with an alert on successful allotment of courses.

Exceptions:

1. Selected faculty member does not exist in the database.
2. Selected course does not exist in the database.
3. Selected semester and section do not match for the selected course.

2.4.4 UC-4

Title: Monitor department profiles

Description: Head of Department (HoD) views the profiles of all faculty and students in their department.

Actor: Head of Department

Preconditions:

1. Department exists in the database.
2. Faculty and students belong to department.

Postconditions:

1. Webpage shows the profiles of all faculty and students in the department.

Main Success Scenario:

1. HoD visits their dashboard and selects “View Faculty” or “View Students” button.
2. Webpage loads with data for all faculty and students in the department.

Exceptions:

1. No faculty and students belong to department.

2.4.5 UC-5

Title: Setting of attainment targets, CO-PO and CO-PSO mapping

Description: The course coordinator sets attainment targets, CO-PO and CO-PSO mapping for a course.

Actor: Course coordinator

Preconditions:

1. Course should exist in the database.

Postconditions:

1. Attainment targets, CO-PO and CO-PSO mapping are persistent in the database.

Main Success Scenario:

1. The course coordinator navigates to the webpage to input attainment targets, CO-PO and CO-PSO mapping.
2. On submit, they are prompted with an alert for successful submission.

Exceptions:

1. Course does not exist in the database.

2.4.6 UC-6

Title: CIE and SEE marks upload

Description: Faculty uploads Excel sheet with marks for assignments and mid-semester examination (CIE) and end-semester examination (SEE), for their assigned section.

Actor: Faculty

Preconditions:

1. Course (taught to a section, for a batch, belonging to a department) allotted to the faculty exists in the database.
2. Attainment targets, CO-PO and CO-PSO mapping for the course have been set by the coordinator.

Postconditions:

1. Marks for each CIE assessment and SEE are uploaded to the database and are persistent.
2. Direct target attainment analytics are calculated for the uploaded marks.
3. Upload history displays the assessments for which marks have been uploaded.

Main Success Scenario:

1. Faculty navigates to the webpage for uploading CIE assessment marks and SEE marks.
2. Faculty uploads Excel sheet with marks onto the portal and submits.
3. The webpage displays an alert for successful submission and upload history is updated.

Exceptions

1. Course does not exist in the database.
2. Attainment targets, CO-PO and CO-PSO mapping for the course have not been set by the coordinator.

3. The uploaded Excel sheet is not in the correct format.

4. The uploaded Excel sheet is for a different course, section, batch or has a different student list.

2.4.7 UC-7

Title: Course feedback is uploaded to portal

Description: Faculty uploads Excel sheet with CO-wise course feedback for a course, for their assigned section.

Actor: Faculty

Preconditions:

1. Course (taught to a section, for a batch, belonging to a department) allotted to the faculty exists in the database.

2. Attainment targets, CO-PO and CO-PSO mapping for the course have been set by the coordinator.

Postconditions:

1. Indirect target attainment analytics are calculated for the uploaded feedback.

2. Upload history shows feedback has been uploaded.

Main Success Scenario:

1. Faculty navigates to the webpage for course feedback data.

2. Faculty uploads Excel sheet with course feedback onto the portal and submits.

3. The webpage displays an alert for successful submission and upload history is updated.

Exceptions:

1. Course does not exist in the database.

2. The uploaded Excel sheet is not in the correct format.

3. Attainment targets, CO-PO and CO-PSO mapping for the course have not been set by the coordinator.

4. The uploaded Excel sheet is for a different course, section, batch or has a different student list.

2.4.8 UC-8

Title: Root Cause Analysis for CO, PO and PSO

Description: After review of overall attainment levels, faculty provides root cause analysis.

Actor: Faculty

Preconditions:

1. Course (taught to a section, for a batch, belonging to a department) allotted to the faculty exists in the database.

2. Attainment targets, CO-PO and CO-PSO mapping for the course have been set by the coordinator.

3. All CIE and SEE marks have been uploaded by the faculty.

Postconditions:

1. Webpage display the root cause analysis provided by the faculty.

Main Success Scenario:

1. Faculty navigates to the webpage for root cause analysis for CO, PO and PSO.

2. On submit, they are prompted with an alert for successful submission.

Exceptions:

1. Course does not exist in the database.

2. Attainment targets, CO-PO and CO-PSO mapping for the course have not been set by the coordinator.

3. Any CIE or SEE marks have not been uploaded.

2.4.9 UC-9

Title: Providing Action Plan for CO, PO and PSO

Description: After reviewing CO/PO/PSO-wise target attainment, faculty provides action plan for improving performance.

Actor: Faculty

Preconditions:

1. Course (taught to a section, for a batch, belonging to a department) allotted to the faculty exists in the database.

2. Attainment targets, CO-PO and CO-PSO mapping for the course have been set by the coordinator.

3. All CIE and SEE marks have been uploaded by the faculty.

Postconditions:

1. Webpage displays all action plans provided by the faculty.

Main Success Scenario:

1. Faculty navigates to the webpage for action plan for CO, PO and PSO.
2. On submit, they are prompted with an alert for successful submission.

Exceptions:

1. Course does not exist in the database.
2. Attainment targets, CO-PO and CO-PSO mapping for the course have not been set by the coordinator.
3. Any CIE or SEE marks have not been uploaded.

2.5 External Interface Requirements

2.5.1 User Interfaces

The user interface (UI) will provide a clean, responsive, and intuitive experience for users.

Components:

- Login Screen
- Sidebar: User details and list of sections (faculty) and available actions (other profiles).
- Main Dashboard: Provide link to pages to perform various functions.
- Data Grids: Editable tables for target and CO-PO and CO-PSO mapping adjustments.
- Data Upload: Buttons for choosing upload data type, choose file and submitting. Text input fields for entering Root Cause Analysis and Action Plans.
- Upload History: History of assignments and examination marks uploaded.
- Analytics: Display for computed analytics based on uploaded marks and feedback.
- Alerts: Color-coded warnings for different attainment level thresholds.
- Help: Exportable Excel templates for assignments, examinations, professor, course and section data.
- Logout Button

2.5.2 Hardware Interfaces

No specific hardware interface required for this project.

2.5.3 Software Interfaces

Frontend:

- React 18+

Backend:

- Node.js 18+
- Express.js
- MongoDB 5+ with Mongoose ODM
- REST APIs with JWT authentication

2.5.4 Communications Interfaces

- Protocols: All communications between the client and the server will be done via HTTP/HTTPS. HTTPS will be mandatory to ensure data security.
- Authentication Tokens: Communication between clients and servers will be secured

with tokens (JWT) to verify users' identities and protect their data.

2.6 Non-Functional Requirements

2.6.1 Performance Requirements

- API response time ≤2s for concurrent users.
- Excel upload processing ≤10s for ≤100 students.

3. System Design

3.1 UML Diagram

UML diagrams model software systems by visualizing structure and interactions.

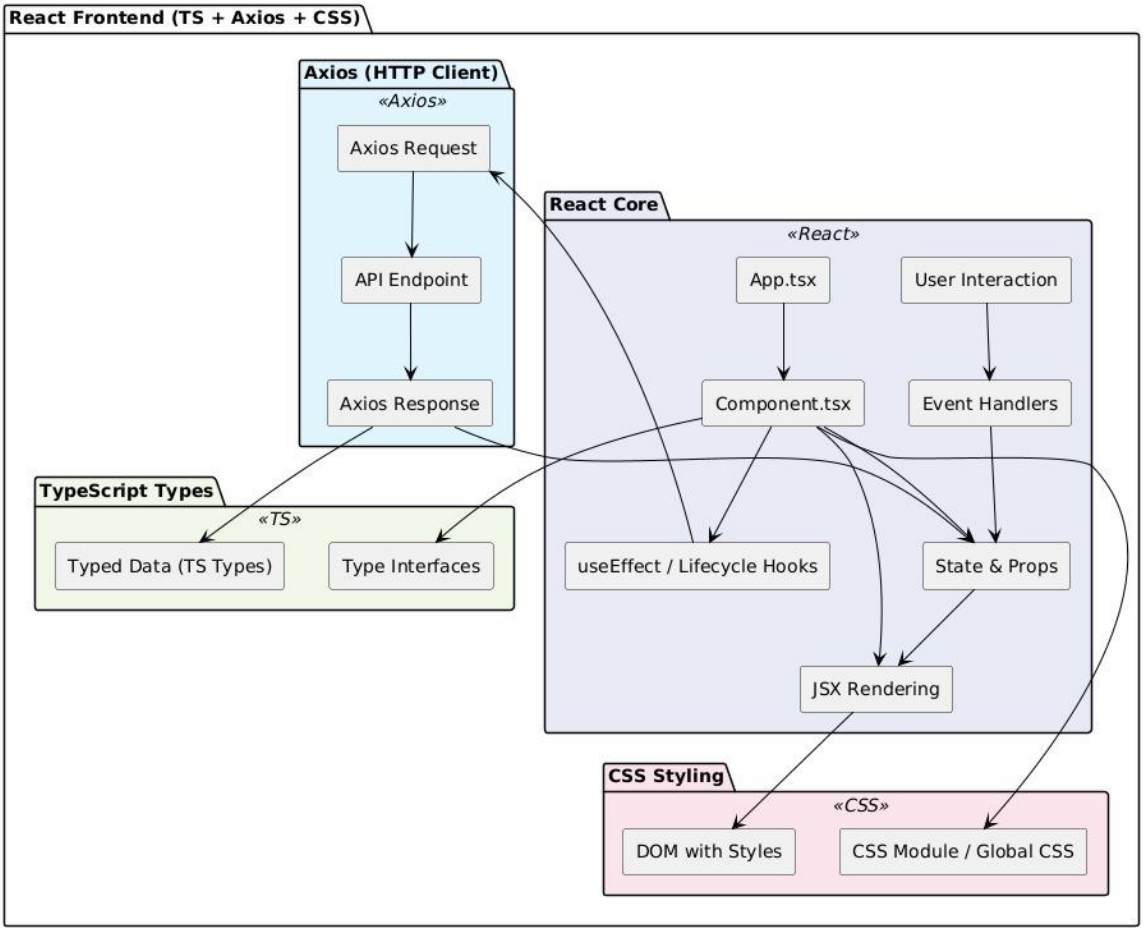

Figure 3.1 UML diagram explaining the front-end architecture for out project.

3.2 ER Diagram

ER diagrams help designs and visualize database structures by mapping entities, attributes, and relationships.

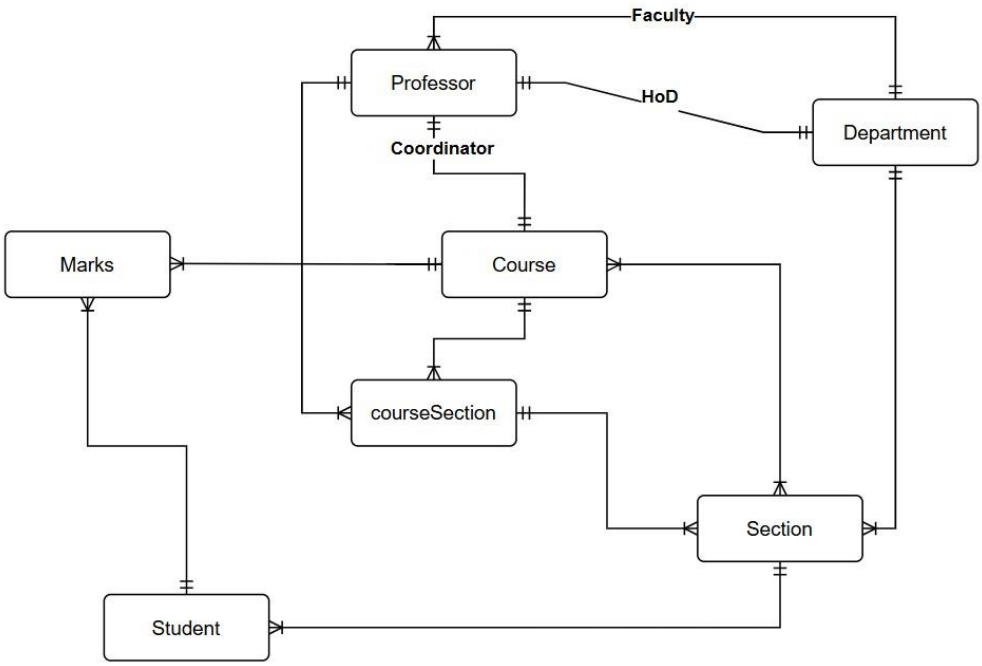

Figure 3.2 ER diagram shows the database structure for our project.

Class Attributes in DBML format:
courseSection

- _id (ObjectId, PK)
- courseId (ObjectId, FK)
- sectionId (ObjectId, FK)
- coAttainment [JSON]
- poAttainment [JSON]
- psoAttainment [JSON]
- coActionPlans [JSON]
- poActionPlans [JSON]
- psoActionPlans [JSON]
- rootCauseCo (varchar)
- rootCausePo (varchar)
- rootCausePso (varchar)

Course

- _id (ObjectId, PK)
- courseID (varchar, unique)
- name (varchar)
- type (varchar)
- coordinator (ObjectId, FK)
- program (varchar)
- sem (varchar)
- year (integer)
- oddEven (varchar)
- coStatements [JSON]
- coPoMapping [JSON]
- coPsoMapping [JSON]

- dept (ObjectId, FK)
- ass1 [JSON]
- ass2 [JSON]
- ass3 [JSON]
- ass4 [JSON]
- midSem [JSON]
- endSem [JSON]

Professor
- _id (ObjectId, PK)
- facultyID (varchar, unique)
- name (varchar)
- email (varchar, unique)
- phoneNo (varchar)
- dept (ObjectId, FK)
- password (varchar)
- hasTaught [JSON]
- section [JSON]

Department
- _id (ObjectId, PK)
- name (varchar, unique)
- hod (ObjectId, FK)
- courses [JSON]
- poStatements [JSON]

Section
- _id (ObjectId, PK)
- name (varchar)
- dept (ObjectId, FK)
- program (varchar)
- batch (varchar)
- sem (varchar)

Student
- _id (ObjectId, PK)
- regNo (integer, unique)
- name (varchar)
- section (ObjectId, FK)
- prevCourses [JSON]
- currentCourses [JSON]
- reRegisteredCourses [JSON]

Marks
- _id (ObjectId, PK)
- student (ObjectId, FK)
- course (ObjectId, FK)
- status [JSON]
- ass1 [JSON]
- ass2 [JSON]

- ass3 [JSON]
- ass4 [JSON]
- midSem [JSON]
- endSem [JSON]
- feedback [JSON]
- grade (varchar)

3.3 DFD Diagrams

DFD diagrams illustrate how data moves through a system, including inputs, processes, and storage.

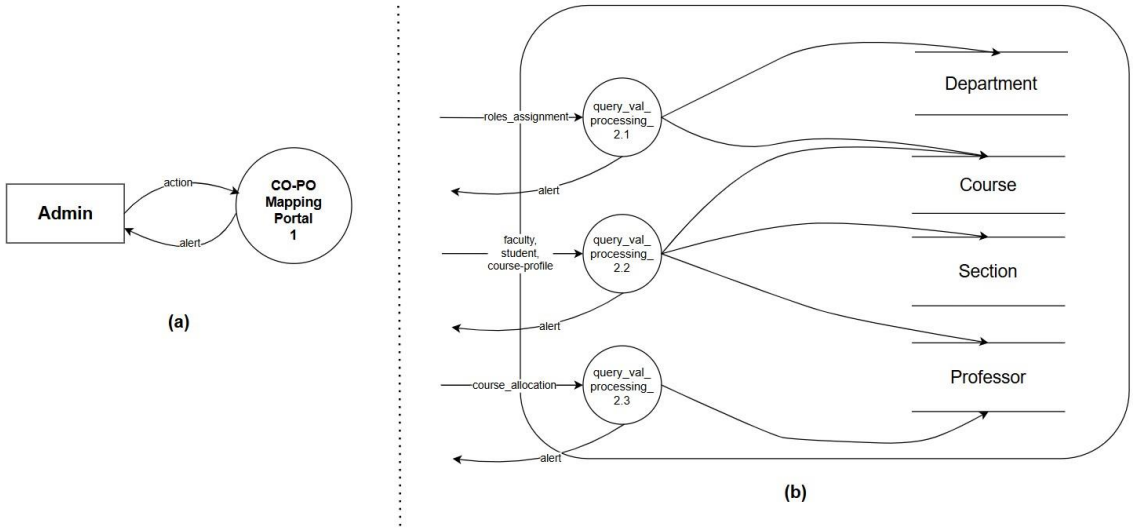

Figure 3.3.1 DFD for Admin: (a) Level 0 and (b) Level 1

Data dictionary:
action: [roles_assignment, faculty_student_course_profile, course_allocation]
roles_assignment: professor + [hod, coordinator]
faculty_student_course_profile: {details}+
course_allocation: professor + course
professor: ObjectId
course: ObjectId
hod: String
coordinator: String
details: String
alert: String

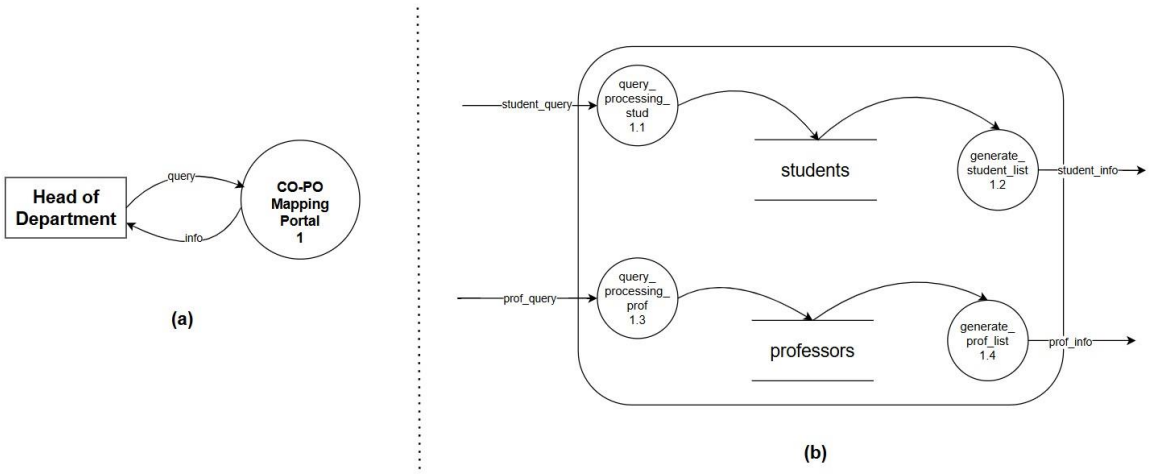

Figure 3.3.2 DFD for Head of Department: (a) Level 0 and (b) Level 1

Data Dictionary:
query: [student_query, prof_query]
info: [student_info, prof_info]
student_query: String
prof_query: String
student_info: {student}+
prof_info: {prof}+
student: ObjectId
prof: ObjectId

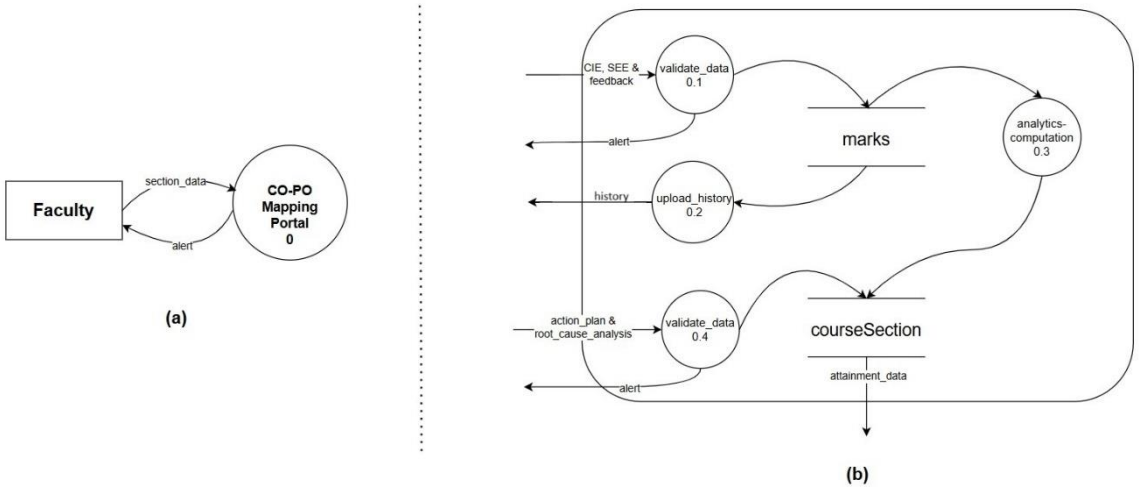

Figure 3.3.3 DFD for Faculty: (a) Level 0 and (b) Level 1

Data Dictionary:
section_data: [CIE_SEE_feedback, action_plan&root_cause_analysis]
alert: [alert, history, attainment_data]
CIE_SEE_feedback: Excel sheet
action_plan&root_cause_analysis: String
alert: String
history: String
attainment_data: Array

4. Implementation

The technologies used to build this project are Node.js (for backend and middleware), MongoDB (for database), and React with TypeScript and Vite (for frontend).

4.1 Backend and API Files

4.1.1 server/controllers/

- cieController.js – Handles internal assessment (CIE) logic.
- courseController.js – Manages course-related operations.
- feedbackController.js – Manages feedback processing logic.
- finalController.js – Final result aggregation or generation logic.
- indexController.js – Entry point for combined controller operations.

4.1.2 server/middlewares/

- isAuthUser.js – Middleware to verify user authentication.
- multer.js – Handles file uploads (e.g., .xlsx files).

4.1.3 server/models/

- courseModel.js – Defines schema for courses.
- departmentModel.js – Defines schema for departments.
- marksModel.js – Defines schema for storing marks.
- professorModel.js – Schema for storing professor information.
- sectionModel.js – Schema for sections.
- studentModel.js – Schema for storing student information.
- courseSection.js – Schema for section, semester and course-wise data.

4.1.4 server/routes/

- cieRouter.js – Routes for CIE-related endpoints.
- courseRouter.js – Routes for course-related endpoints.
- feedbackRouter.js – Routes for feedback endpoints.
- finalRouter.js – Final results and mappings related routes.
- indexRouter.js – Master router to include and export all others.

4.1.5 server/utils/

- generateToken.js – Generates authentication tokens.
- getCourseNameById.js – Fetches course name given its ID.
- testModel.js – Test utility for model debugging.
- server/server.js - Main server entry file to initialize middleware, DB, and routes.

4.1.6 server/config/mongoose-connection.js

- Handles MongoDB database connection logic.

4.2 Frontend Files

- client/src/App.tsx - Main app wrapper for React routing and context providers.
- client/src/context/UserContext.tsx - Implements and exports user authentication context.
- client/src/components/ - Likely contains reusable UI components (not fully visible in screenshot).
- client/src/LandingPage.tsx - Frontend landing page shown upon entry.
- client/src/main.tsx - Application bootstrap file using React and Vite.
- client/public/ - Assets like logos and images for the interface.

4.3 Pseudo Code – Analytics for lab performance evaluation.

4.3.1 Assessment Sheet

FOR each student:

```
    Total_IA_Marks = SUM(All CO marks)
    Grade = Fetch from grade range

4.3.2 Direct Attainment
FOR each CO:
    Scaled_CO = CO_Mark / Sum(Max_CO_Marks) * 60

4.3.3 Attainment Level Calculation (CIE)
FOR each CO:
    IF Scaled_CO >= 45: Level = 3
    ELSE IF 45 > Scaled_CO >= 40: Level = 2
    ELSE: Level = 1

4.3.4 Lab CO Attainment
FOR each CO:
    CO_Attainment = Level (from 4.3.3)

4.3.5 CES Feedback Attainment
FOR each CO:
    Total_Students_Responded = count(CES entries for (CO)i)
    Weighted_Score = Σ (Rating × Count of Students who gave that rating)
    CES_Average = Weighted_Score / Total_Students_Responded
    IF CES_Average >= 3.5: CES_Level = 3
    ELSE IF 3.5 > CES_Average >= 2.5: CES_Level = 2
    ELSE: CES_Level = 1

4.3.6 Overall Lab CO Attainment
FOR each CO:
    Direct = CO_Attainment (from 4.3.4)
    Indirect = CES_Level (from 4.3.5)
    Overall_CO_Attainment = (Direct * 0.8) + (Indirect * 0.2)

4.3.7 CO Target Achievement Check
FOR each CO:
    Target_Value = Fetch from Target Table
    IF Overall_CO_Attainment >= Target_Value:
        Target_Achieved = Yes
    ELSE:
        Target_Achieved = No

4.3.8 Action Plan for CO Attainment
FOR each CO:
    IF Target_Achieved == No:
        Action_Plan = "Conduct remedial or additional lab work"
    ELSE:
        Action_Plan = "Continue same process"

4.3.9 PO/PSO Attainment for Lab
FOR each PO/PSO:
    Direct_PO/PSO = Σ (CO_Attainment × CO-to-PO/PSO weight)
    Indirect_PO/PSO = Σ (Overall_CO_Attainment × CO-to-PO/PSO weight)
    IF Direct_PO/PSO >= Target_Value:
        Target_Achieved = Yes
    ELSE:
        Target_Achieved = No
```

4.3.10 Action Plan for PO/PSO Attainment (Lab)
FOR each PO/PSO:
  IF Target_Achieved == No:
    Action_Plan = "Refine experiments, clarify mapping"
  ELSE:
    Action_Plan = "Maintain lab methodology"

4.4 Pseudo Code – Analytics for theory subject performance.

4.4.1 Assessment Sheet
FOR each student:
  Total_IA_Marks = SUM(All CO marks for ISA)
  Grade = Fetch from grade range

4.4.2 Direct Attainment (ISA)
FOR each CO:
  Scaled_CO = (CO_Mark / Sum(Max_CO_Marks)) * 60
4.4.3 Direct Attainment (SEE)
FOR each CO:
  Scaled_CO = (CO_Mark / Max_Marks_per_CO) * 40

4.4.4 Attainment Level Achieved (Theory)
FOR each CO:
  Attainment_Level =
    if Scaled_CO >= 75%: level = 3
    if 75 > Scaled_CO >= 50%: level = 2
    else: level = 1

4.4.5 Overall CO Attainment (Theory)
FOR each CO:
  Overall_CO_Attainment = (ISA_Attainment * 60 + SEE_Attainment * 40) / 100
4.4.6 CO to PO/PSO Mapping (Theory)
FOR each CO:
  Map CO to (PO, PSO) from matrix
  Calculate PO/PSO attainment = SUM(CO_Attainment * CO_PO_Mapping_Weight)

4.4.7 CES Feedback Attainment
FOR each CO:
  Total_Students_Responded = count(CES entries for (CO)i)
  Weighted_Score = Σ (Rating × Count of Students who gave that rating)
  CES_Average = Weighted_Score / Total_Students_Responded
  IF CES_Average >= 3.5: CES_Level = 3
  ELSE IF 3.5 > CES_Average >= 2.5: CES_Level = 2
  ELSE: CES_Level = 1

4.4.8 CO Attainment (Direct, Indirect, and Overall)
FOR each CO:
  Direct_Attainment = Average of CIE and SEE Attainment
  Indirect_Attainment = CES_Level for (CO)i
  Overall_Attainment = (Direct_Attainment * 0.8) + (Indirect_Attainment * 0.2)

4.4.9 CO Target Achievement Check
FOR each CO:
  Target_Value = Fetch from Target Table
  IF Overall_Attainment >= Target_Value:
    Target_Achieved = Yes

ELSE:
Target_Achieved = No

4.4.10 Action Plan for COs
FOR each CO:
IF Target_Achieved == No:
Recommend Action_Plan = "Improve question design / reinforce concepts / additional assignments"
ELSE:
Action_Plan = "Maintain current strategy"

4.4.11 PO and PSO Attainment
FOR each PO:
Direct_PO = Σ (Direct CO_Attainment × CO-to-PO weight)
Indirect_PO = Σ (Overall CO_Attainment × CO-to-PO weight)
IF Direct_PO >= Target_PO:
Target_Achieved = Yes
ELSE:
Target_Achieved = No
Repeat above for each PSO.

4.4.12 Action Plan for PO/PSO
FOR each PO/PSO:
IF Target_Achieved == No:
Action_Plan = "Strengthen related COs or improve CES feedback"
ELSE:
Action_Plan = "Maintain approach"

5. Testing

5.1 Tools Used

- **Postman**: Used for testing API endpoints by making HTTP requests (GET, POST, PUT and DELETE) and verifying responses.
- **Excel (.xlsx files)**: Used as test datasets to simulate real inputs. These files were uploaded through the UI/backend to verify that:
    - File upload functionality works correctly.
    - All backend formulas and calculations (for marks, evaluations, etc.) are processed correctly.
    - Data is parsed and stored accurately in the database.

5.2 Test Cases

- **API Testing with Postman:**
    - Checked login and user authentication with /api/auth.
    - Tested protected routes with and without authentication tokens.
    - Tested creation of new faculty, course, section and student objects for admin profile.
    - Tested assignment of coordinator and HoD roles by admin profile.
    - Tested ability to view all courses and students of a department for HoD profile.
    - Tested correct mapping of faculty to section objects.
    - Tested input of attainment targets, and, CO-PO and CO-PSO mapping by course co-ordinator profile.
    - Test ability to input root cause analysis and action plan by faculty profiles.
- **File Upload Testing:**
    - Uploaded .xlsx files like tempass1.xlsx, tempass2.xlsx, etc.
    - Confirmed backend correctly processed and extracted data from each file.
    - It was validated that calculated results matched expected outputs from formulas.

5.3 Results

- All API endpoints returned the expected data and proper status codes.
- Role-based access control worked correctly using middleware.
- Uploaded .xlsx files were parsed successfully.
- All mark calculations and feedback aggregation logic produced correct output.
- Root cause analysis and Action plan once provided by faculty is persistent in the database.

6. Conclusion and Future Enhancements

The “Automated Web-Based System for Course and Program Outcome Tracking and Analysis” Tool has been successfully implemented to streamline the evaluation process for academic performance, feedback, and curriculum mapping. It integrates essential modules such as student data uploads, CO-PO, CO-PSO attainment calculations, assignment tracking, and feedback analysis. The backend and frontend work in tandem to offer a role-based, structured, and secure platform for professors and admins. Through thorough testing using Postman and real-world test datasets (Excel files), the system has been validated to function as intended across all major components.

6.1 Future Enhancements:

6.1.1 UI Improvements:

- Make the interface more intuitive and user-friendly, especially for faculty unfamiliar with modern UI workflows.
- Improve sidebar to include contextual details like designation, professor name, employee code, etc.
- Use dashboard colors and visual cues to indicate pending or incomplete tasks (e.g., missing uploads, unverified data).
- Add in-line guidance or tooltips to help with uploading Excel files, especially format expectations.

6.1.2 Backend Improvements

- Optimize database schema structure and API structure to be able to serve large number of concurrent users.

These improvements aim to enhance usability, reduce confusion, and add a more structured and maintainable workflow suitable for our institution at scale.

7. Screenshots and Output

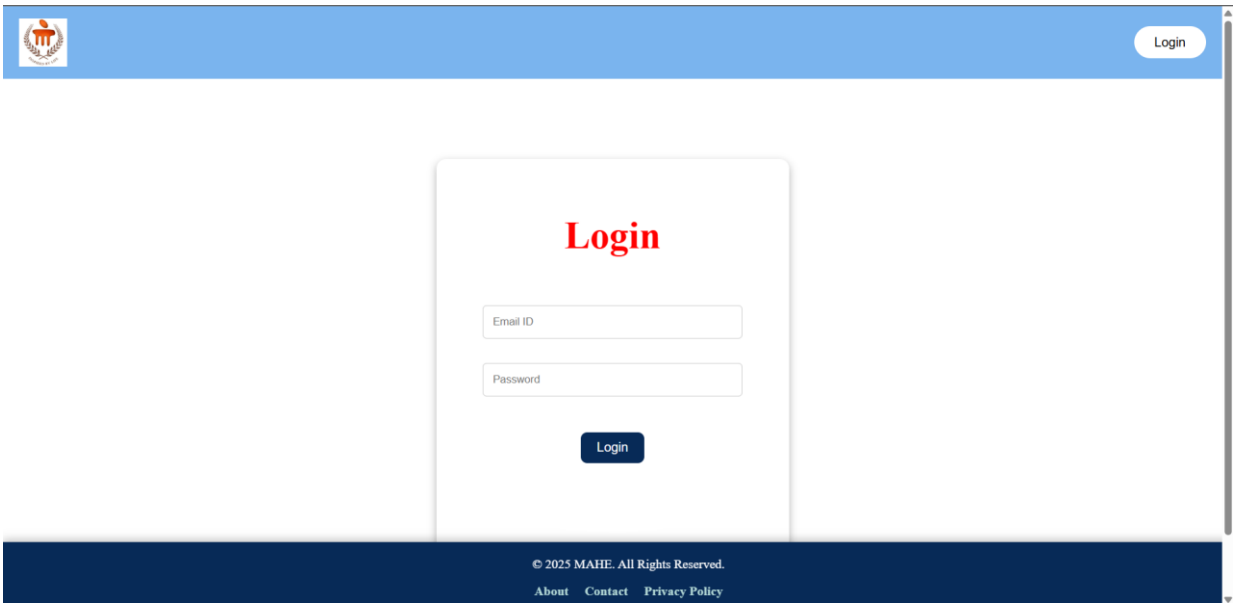

Figure 7.1 Login page of our portal.

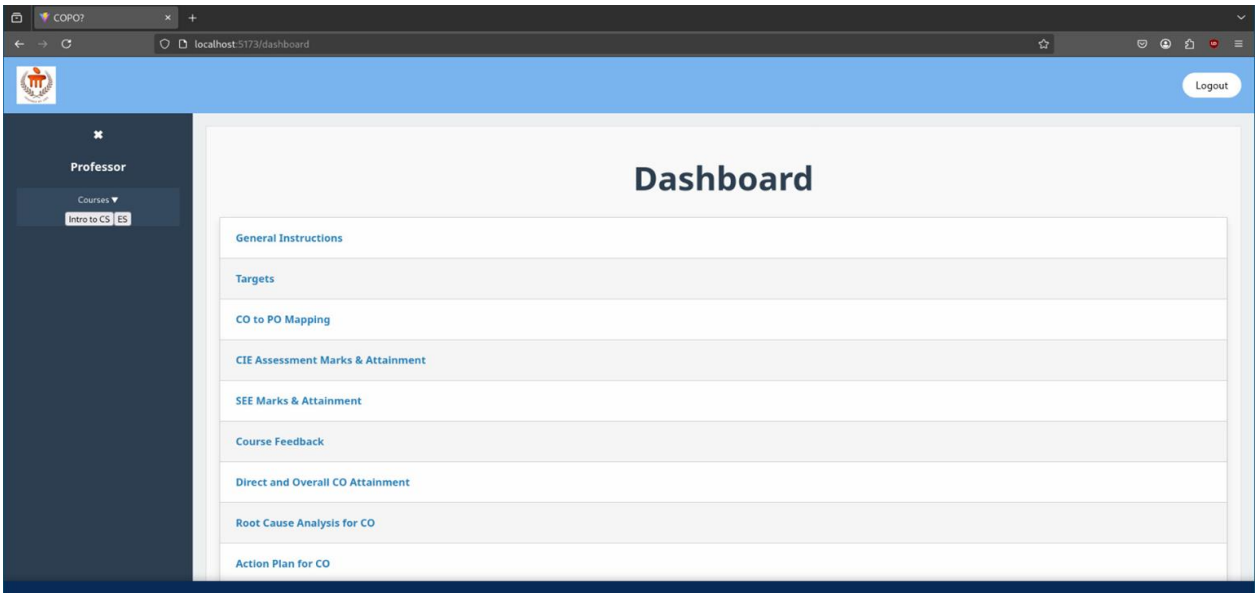

Figure 7.2 Main dashboard for faculty.

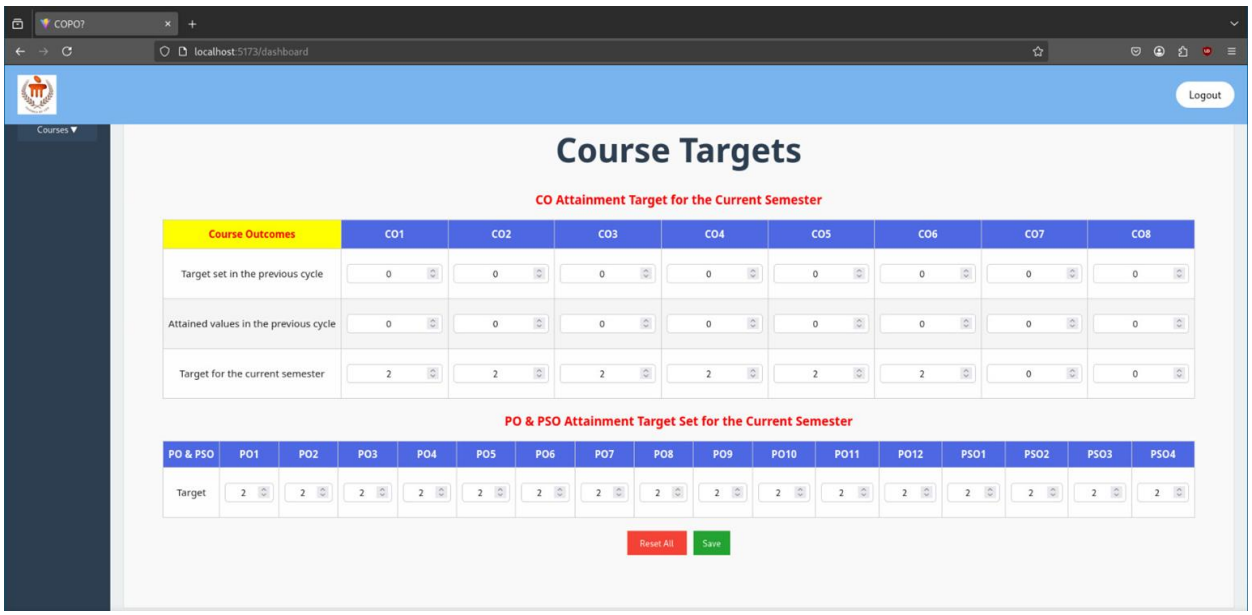

Figure 7.3 Webpage where course targets set by coordinator are displayed. Faculty are unable to edit these values.

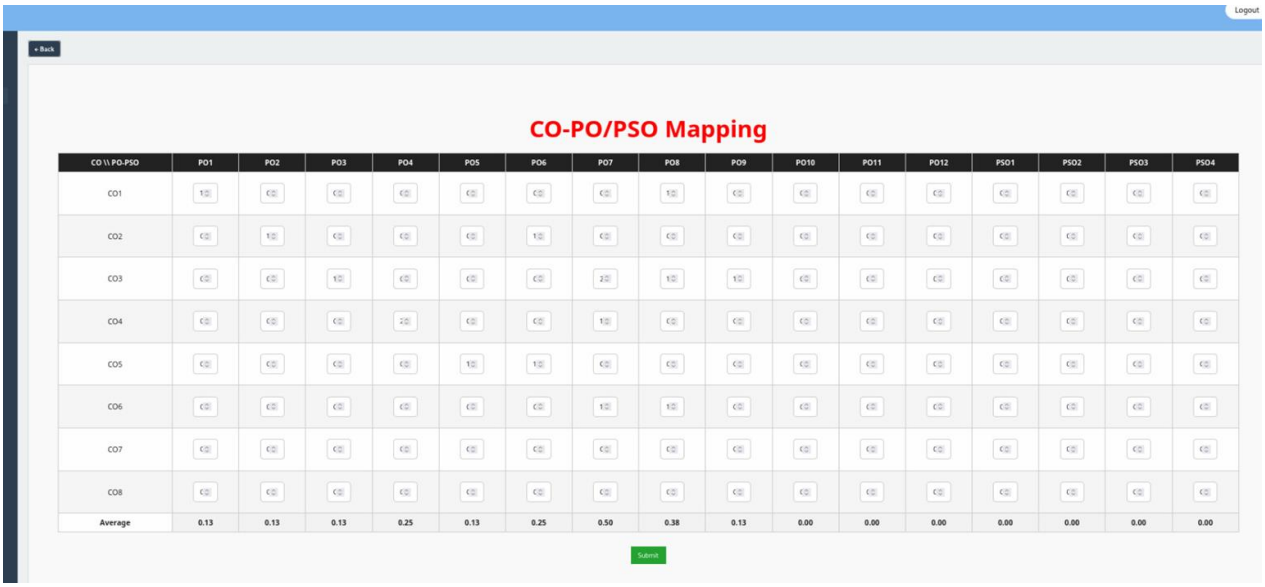

Figure 7.4 CO-PO/PSO mapping page. Values set by coordinator are displayed. Faculty are unable to edit these values.

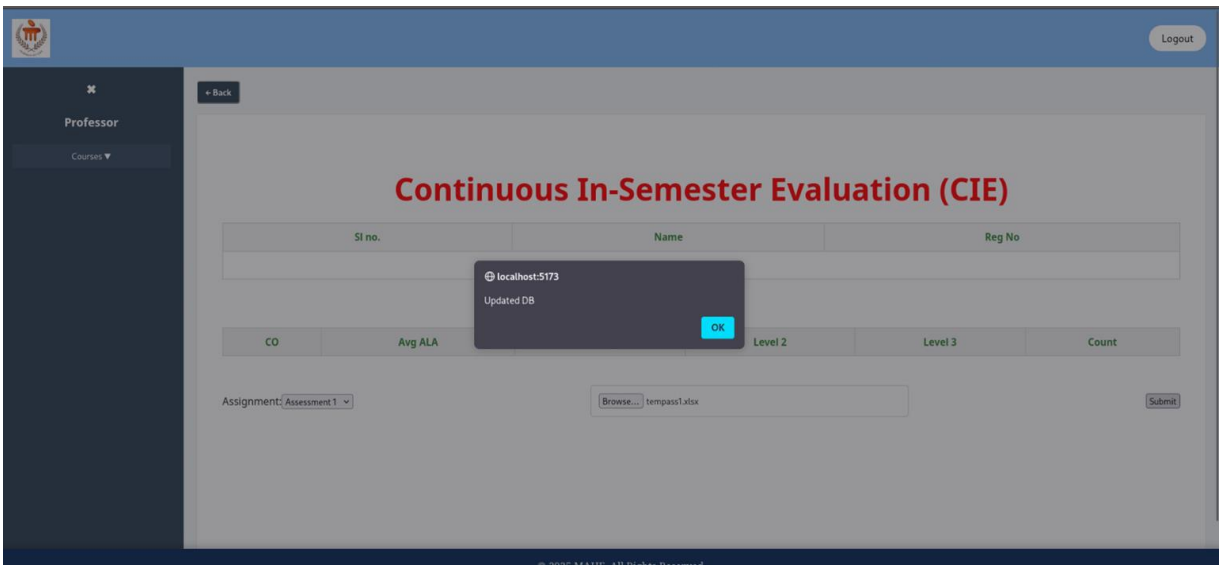

Figure 7.5 Alert on successful upload of CIE marks by faculty.

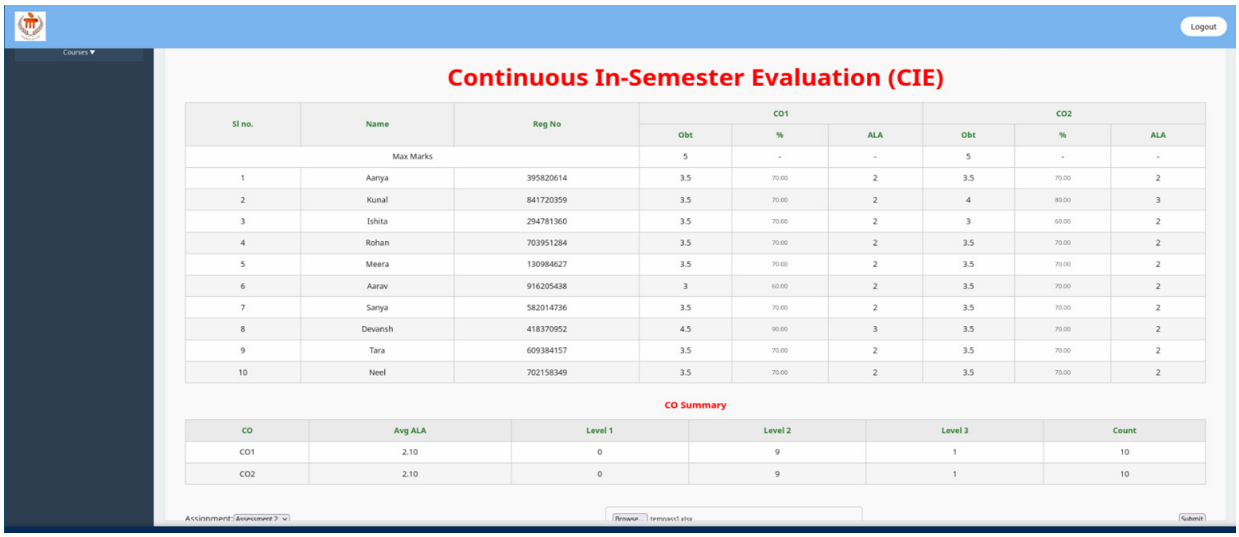

Figure 7.6 CIE marks obtained and maximum marks CO-wise along with ALA on CIE page.

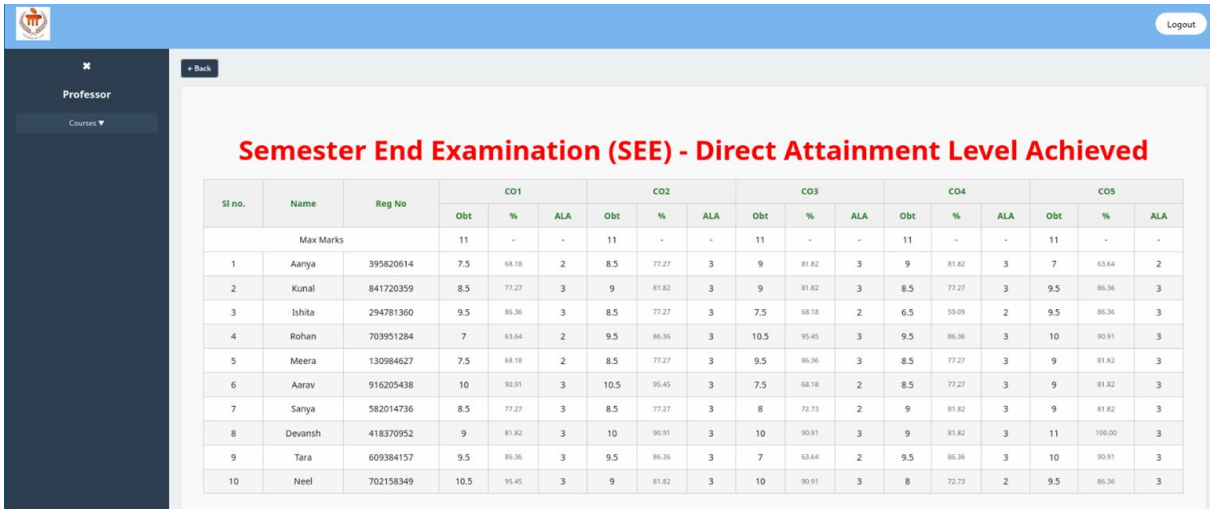

Figure 7.7 SEE marks obtained and maximum marks CO-wise along with ALA on SEE page.

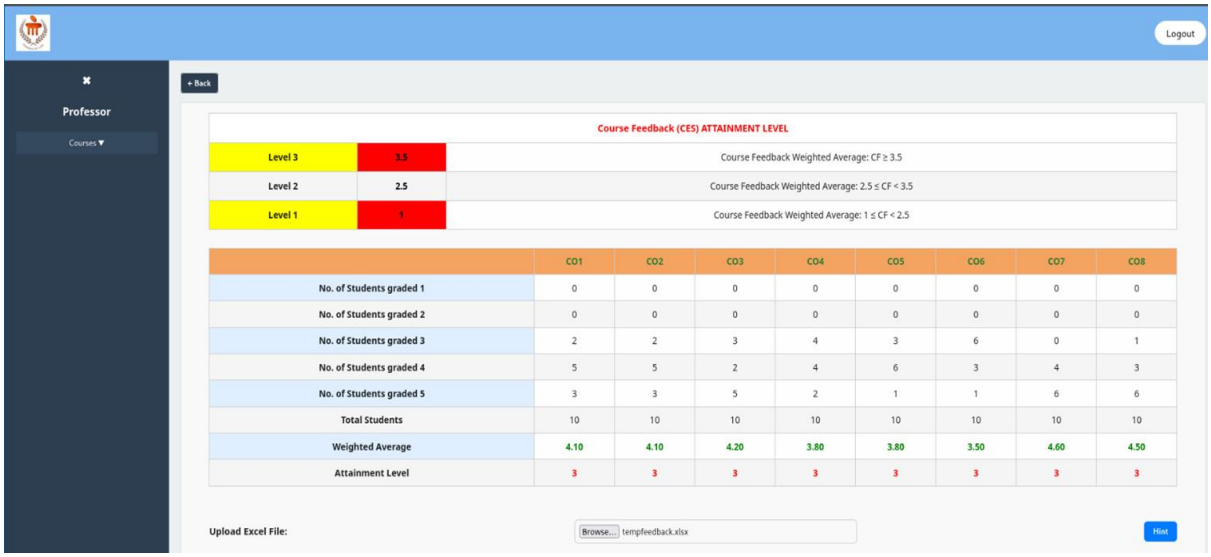

Figure 7.8 Course feedback uploaded for a section by faculty, calculated CO-wise average being displayed.

Course Feedback (CES): Individual Student Responses

Sl. No	Name	Reg Number	CO1	CO2	CO3	CO4	CO5	CO6	CO7	CO8
1	Aanya	395820614	4	5	3	4	4	3	4	5
2	Kunal	841720359	3	4	5	4	3	3	5	4
3	Ishita	294781360	5	4	4	3	5	4	5	3
4	Rohan	703951284	3	3	5	3	4	3	4	5
5	Meera	130984627	4	5	5	5	4	5	5	5
6	Aarav	916205438	4	3	5	4	4	3	5	5
7	Sanya	582014736	5	4	3	3	4	3	5	4
8	Devansh	418370952	4	4	5	4	3	4	4	5
9	Tara	609384157	5	5	4	3	4	3	4	4
10	Neel	702158349	4	4	3	5	3	4	5	5

Figure 7.9 Feedback values for each student.

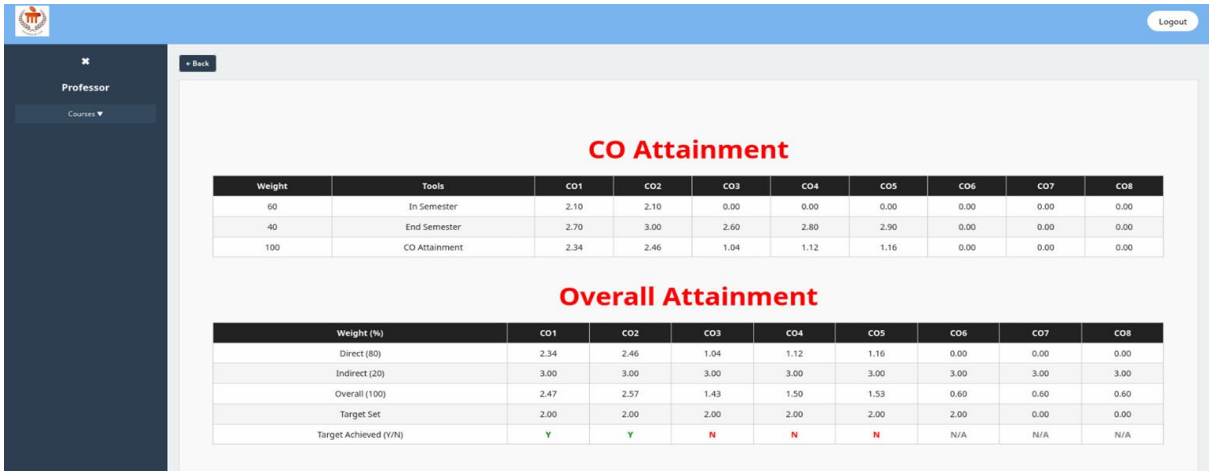

Figure 7.10 Direct, Indirect and Overall CO attainment values calculated from CIE, SEE and feedback scores for a section.

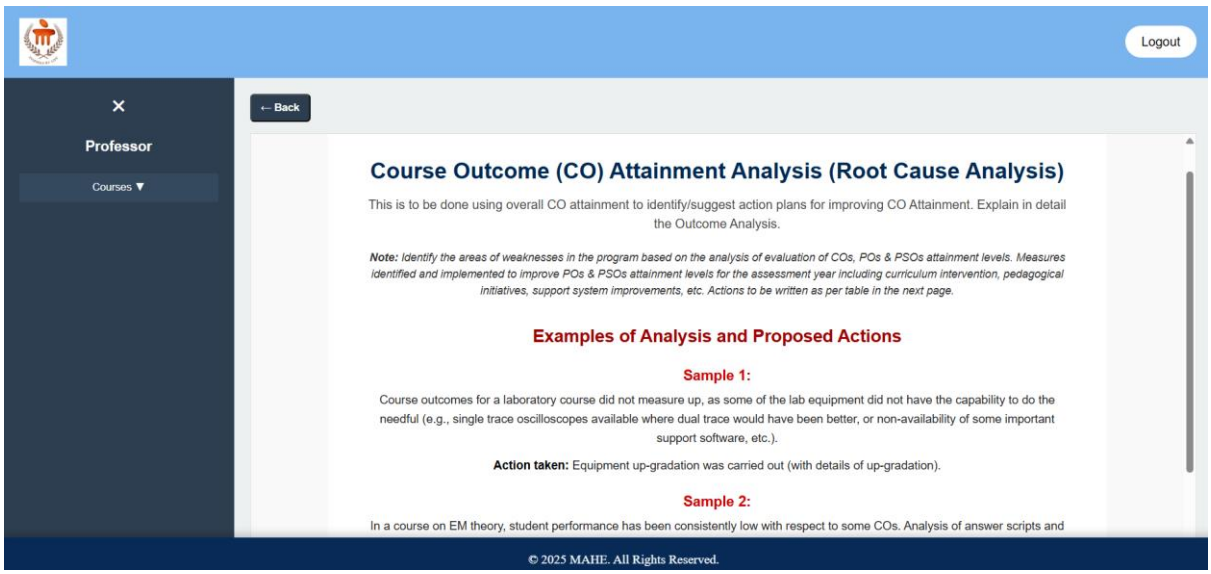

Figure 7.11 Root cause analysis for CO sample.

CO Action Plan

CO	CO Statement	Target Set	Target Attained	Target Achieved (Y/N)	Action Plan
CO1	Understand basic programming concepts	2.00	2.47	Y	Enter action plan
CO2	Apply logic to solve computational problems	2.00	2.57	Y	Enter action plan
CO3	Analyze algorithms for efficiency	2.00	1.43	N	better techniques
CO4	Design simple software applications	2.00	1.50	N	more real world app
CO5	Evaluate different data structures	2.00	1.53	N	easier openeing structs
CO6	Implement modular programming techniques	2.00	0.60	N	more detailed explanations
CO7	Communicate technical ideas clearly	0.00	0.60	Y	Enter action plan
CO8	Work effectively in team-based environments	0.00	0.60	Y	Enter action plan

Save

Figure 7.12 Sample action plan input for underperforming CO's (in red).

← Back

Direct PO & PSO Attainment

Direct CO Att.	CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
2.34	CO1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
2.46	CO2	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1
1.04	CO3	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
1.12	CO4	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
1.16	CO5	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
0.00	CO6	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
0.00	CO7	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
0.00	CO8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Attainment		2.34	2.46	1.04	1.12	1.16	0.00	0.00	0.00	0.00	0.00	1.16	1.12	1.04	2.46	2.34	2.46
Target Set		2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
Target Attained		Y	Y	N	N	N	NA	NA	NA	NA	NA	N	N	N	Y	Y	Y

Figure 7.13 Direct PO and PSO attained values, with their target values. Values in red indicate underperforming PO's and PSO's.

Overall (Indirect) PO & PSO Attainment

Overall CO Att.	CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
2.47	CO1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
2.57	CO2	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1
1.43	CO3	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
1.50	CO4	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
1.53	CO5	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
0.60	CO6	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
0.60	CO7	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
0.60	CO8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Attainment		2.47	2.57	1.43	1.50	1.53	0.60	0.60	0.60	0.60	0.60	1.53	1.50	1.43	2.57	2.47	2.57
Target Set		2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
Target Attained		Y	Y	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y

Figure 7.14 Overall PO and PSO attained values, with their target values. Values in red indicate underperforming PO's and PSO's.

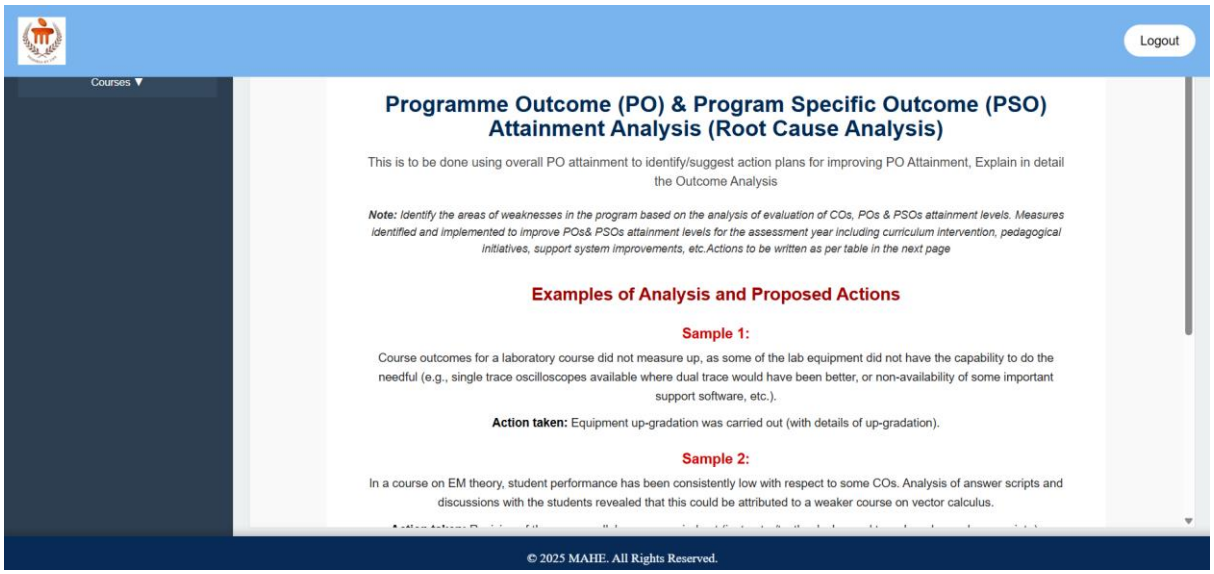

Figure 7.15 Root cause analysis for PO's and PSO's sample.

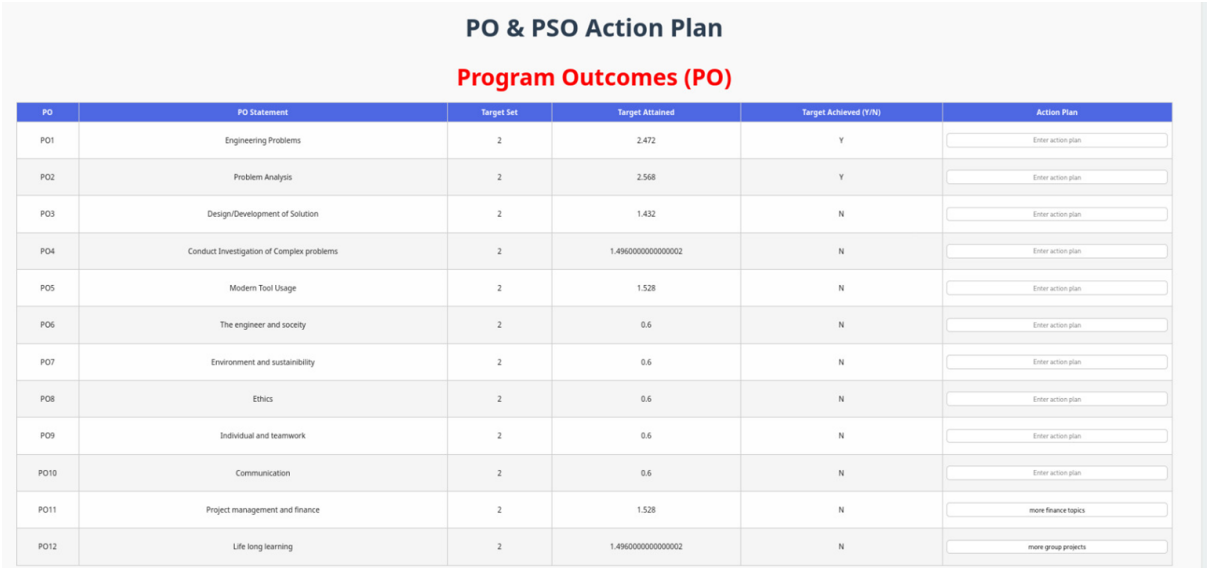

Figure 7.16 Page to enter action plan for underperforming PO's.

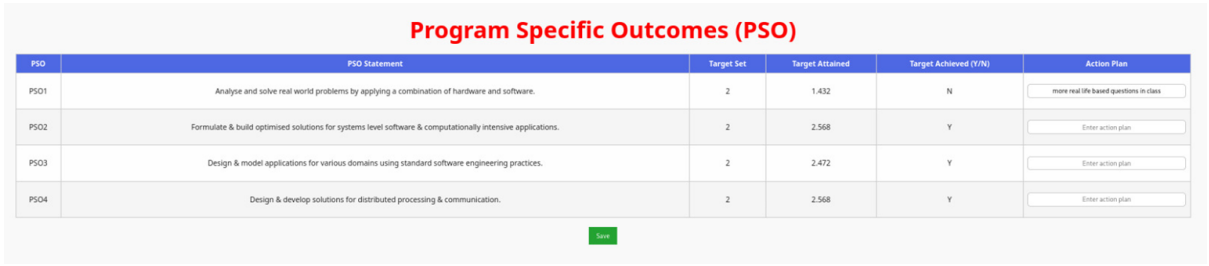

Figure 7.17 Page to enter action plan for underperforming PSO's.

8. References

8.1 Postman
https://learning.postman.com/docs/introduction/overview/
Last accessed: 4 April 2025

8.2 Node.js
https://nodejs.org/docs/latest/api/
Last accessed: 1 April 2025

8.3 React
https://react.dev/reference/react
Last accessed: 2 April 2025

8.4 Express
https://expressjs.com/en/5x/api.html
Last accessed: 3 April 2025

8.5 MongoDB
https://www.mongodb.com/docs/atlas/getting-started/
Last accessed: 4 April 2025

8.6 Rajib Mall, Fundamentals of Software Engineering (4e), PHI Learning 2014

8.7 JWT Authentication
https://learn.microsoft.com/en-us/aspnet/core/security/authentication/configure-jwt-bearer-authentication?view=aspnetcore-9.0
Last accessed: 3 April 2025